A Distillation–Teleportation Protocol for Fault-Tolerant QRAM

Alexander M. Dalzell*, András Gilyén†, Connor T. Hann*, Sam McArdle*,
Grant Salton*‡, Quynh T. Nguyen§, Aleksander Kubica*¶, Fernando G.S.L. Brandão*‖
*AWS Center for Quantum Computing, Pasadena, CA, USA
†HUN-REN Alfréd Rényi Institute of Mathematics, Budapest, Hungary
‡Amazon Quantum Solutions Lab, Seattle, WA, USA
§School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA
¶Yale Quantum Institute & Department of Applied Physics, New Haven, CT, USA
‖Institute for Quantum Information and Matter, Caltech, Pasadena, CA, USA

***Abstract*—We present a protocol for fault-tolerantly implementing the logical quantum random access memory (QRAM) operation, given access to a specialized, noisy QRAM device. For coherently accessing classical memories of size 2^n, our protocol consumes only $\mathrm{poly}(n)$ fault-tolerant quantum resources (logical gates, logical qubits, quantum error correction cycles, etc.), avoiding the need to perform active error correction on all $\Omega(2^n)$ components of the QRAM device. This is the first rigorous conceptual demonstration that a specialized, noisy QRAM device could be useful for implementing a fault-tolerant quantum algorithm. In fact, the fidelity of the device can be as low as $1/\mathrm{poly}(n)$. The protocol queries the noisy QRAM device $\mathrm{poly}(n)$ times to prepare a sequence of n-qubit QRAM resource states, which are moved to a general-purpose $\mathrm{poly}(n)$-size processor to be encoded into a QEC code, distilled, and fault-tolerantly teleported into the computation. To aid this protocol, we develop a new gate-efficient streaming version of quantum purity amplification that matches the optimal sample complexity in a wide range of parameters and is therefore of independent interest.**

The exponential reduction in fault-tolerant quantum resources comes at the expense of an exponential quantity of purely classical complexity—each of the n iterations of the protocol requires adaptively updating the 2^n-size classical dataset and providing the noisy QRAM device with access to the updated dataset at the next iteration. We show that this classical operation can be parallelized to $\mathrm{poly}(n)$ classical circuit depth, but only in a model where classical sparse matrix-vector multiplication for 2^n-dimensional vectors can be as well. While our protocol demonstrates that QRAM is more compatible with fault-tolerant quantum computation than previously thought, the need for significant classical computational complexity exposes potentially fundamental limitations to realizing a truly $\mathrm{poly}(n)$-cost fault-tolerant QRAM.

***Index Terms*—quantum information science, quantum algorithm, random access memory, fault tolerant computing, error correction, teleportation**

I. Introduction

The development of fast and large-scale random access memory (RAM) has played an indispensable role in the development of conventional computing. Early RAM devices assisted in the first demonstrations of stored-program electronic computers [1]–[3], and today, the availability of efficient high-speed RAM enables data-intensive computing applications in areas like machine learning [4], [5]. At an abstract level, RAM performs the following operation: take as input an n-bit address x specifying a location in memory, and retrieve one of the 2^n data items $f(x)$, labeled by x. In practice, the access time for RAM is astonishingly fast: modern RAM chips can achieve latency of 10 nanoseconds or faster.[1] Furthermore, the RAM runtime is independent of the location of the data within the memory, and the latency can remain nearly unchanged even as the overall size of the memory is scaled up.

The idea of quantum random access memory (QRAM) [7], [8] is to achieve something similar even when the n-qubit address register is in a quantum superposition $\sum_x \alpha_x|x\rangle$ of all 2^n addresses. For simplicity, we consider the most basic version of QRAM: applying a phase $(-1)^{f(x)}$ onto basis state $|x\rangle$, where the 2^n binary values $f(0), f(1), \ldots, f(2^n-1)$ are stored in classical memory.

$$\sum_{x\in\{0,1\}^n} \alpha_x|x\rangle \quad \overset{V(f)}{\longmapsto} \quad \sum_{x\in\{0,1\}^n} (-1)^{f(x)}\alpha_x|x\rangle. \tag{1}$$

We refer to the n-bit Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ as the classical *dataset* or *data table* we want to query. For each f, the n-qubit unitary $V(f)$ that implements the QRAM operation is diagonal in the computational basis, with diagonal ± 1 entries determined by f. We note that the controlled $V(f)$ operation[2] can be used to implement the more familiar

[1]RAM latency is a complex topic, due to the many kinds of memory that are used in a computer, each type offering benefits and drawbacks on a number of dimensions including speed, physical size per bit, volatility, and price. See Ref. [6] and the associated datasheet found there for a concrete example of an asynchronous static RAM chip achieving 10 nanoseconds latency on 2^{20} memory locations (each storing 8 bits).

[2]Controlled $V(f)$ is equivalent to (non-controlled) $V(\hat{f})$ for a dataset $\hat{f}$ with $n+1$ address bits; see Appendix A of the full version [9].

We refer to arxiv.org/abs/2505.20265 for the full version of this paper.

formulation of QRAM, which reads the classical data into an ancilla register as $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$.[3]

Whereas a single classical RAM query can access at most one entry of a data table, a single QRAM query suffices to create a superposition over all 2^n entries. The assumption that QRAM is cheap and available underlies a number of proposed quantum algorithms (see Refs. [8], [10]–[12] for relevant surveys), which leverage this ability to offer up to exponential speedups over their classical counterparts. Often, the need for QRAM in these algorithms is contained within an unspecified oracle or data access assumption. For instance, quantum machine learning algorithms for support vector machines [13], Gaussian process regression [14], and recommendation systems [15] require only a polylogarithmic (in the size of the dataset) number of queries to an oracle that accesses (in superposition) the entries of a classical matrix or vector. Similarly, quantum algorithms for solving differential equations [16]–[21] discretize the equations and invert the resulting linear systems [22], in some cases incurring only a polylogarithmic (in the size of the linear system) number of queries to the classical data defining the instance, such as object geometries and boundary conditions. As a final example, quantum algorithms for solving optimization problems like semidefinite and linear programs [23]–[30], with applications in logistics and finance [31], [32], require coherent oracle access to the classical matrices defining the optimization problem. In all of these areas, the claimed speedup is typically dependent upon the assumption that—at least at an abstract level—the cost of QRAM is similar to that of RAM.

Cheap QRAM assumption: For an arbitrary data table f, the computational cost of implementing the unitary operation $V(f)$ from Eq. (1) is $\mathrm{poly}(n)$.

Here, the term *computational cost* is intentionally vague—depending on the context, it might refer to circuit depth, physical runtime, energy dissipated, or some other metric—one must define it more precisely before justifying the assumption (see discussion in Ref. [8]). Focusing on physical runtime/latency as a metric, the assumption of $\mathrm{poly}(n)$ cost is roughly valid in the case of RAM: one can write down classical circuits for RAM that have $O(n)$ depth, and in practice actual RAM chips maintain extremely fast latency even at very large scale. However, for QRAM, the validity of this assumption has been the source of significant controversy [8], [10], [33], [34]. At the root of the issue is the fact that, unlike RAM, QRAM must be implemented in such a way that information about *which* address is being queried is not leaked to the environment, which would lead to decoherence. Strategies for preventing this decoherence without also reducing QRAM's relative power have so far proved to be elusive.

One might try to justify the cheap QRAM assumption by writing down an $O(n)$-depth quantum circuit for the n-qubit unitary $V(f)$ [35]–[38], and then running that circuit on a general-purpose fault-tolerant quantum processor; assuming gates can be implemented in parallel, $O(n)$ latency is achievable. This strategy—referred to as "circuit QRAM" in Ref. [8]—has significant drawbacks. In particular, it requires $\Omega(2^n)$ logical ancilla qubits and $\Omega(2^n)$ classical co-processors to control the system and perform active error correction on all its components in parallel. Each logical ancilla may require dozens or hundreds of physical qubits, leading to an extremely large device footprint, a conclusion that is further exacerbated by the presence of a large number of magic state factories for implementing in parallel the non-Clifford T or Toffoli gates in the circuit, of which there must be at least $\Omega(\sqrt{2^n})$ [39]. One estimate for a surface code approach found that *quadrillions* of physical qubits would be needed for querying an 8-gigabyte memory [35]. The opportunity cost of these quantum and classical resources is steep. For example, the $O(2^n)$ classical co-processors can perform complex tasks like sparse matrix-vector multiplication for $2^n \times 2^n$ matrices in $\mathrm{poly}(n)$ time [8], [10], [34]. Consequently, for circuit QRAM, the cheap QRAM assumption is only justifiable in a cost model that essentially precludes the possibility of quantum advantage in many proposed applications.

Ideally, the QRAM operation would instead be carried out by a specialized hardware element, separate from the main general-purpose quantum processor, mirroring how RAM is performed in a different way than computation on the main CPU. While the physical size of the QRAM hardware element would scale as $\Omega(2^n)$—this is necessary simply to store the dataset f—the runtime could be as little as $O(n)$. Furthermore, since the device is specialized for QRAM, it could in principle be performed *passively* and *ballistically* [8], that is, implemented automatically by natural evolution of the system while requiring at most $\mathrm{poly}(n)$ external interventions from classical control and dissipating at most $\mathrm{poly}(n)$ energy.[4] Constructing such a device is a formidable engineering challenge; there are currently no fully convincing proposals on how it could be done, but nothing rules it out in theory;[5] see Fig. 1a for an abstract picture of how such a device might be structured.

Yet, even if a passive, physical QRAM device did exist, it is unclear how it could actually be useful to a fault-

[3]In some places in the literature [8], the operation $|x\rangle|0\rangle \mapsto |x\rangle|f(x)\rangle$ is referred to as QRACM, where the additional C emphasizes the fact that the data $f(x)$ are classical and thus the states $|f(x)\rangle$ are computational basis states. This distinguishes QRACM from its generalization, QRAQM, where each $|f(x)\rangle$ can be an arbitrary (possibly multiqubit) quantum state. In this paper, we do not consider QRAQM, and we refer to QRAM interchangeably with QRACM.

[4]Reading from a classical RAM can be viewed as a passive operation: although the circuit for RAM has $\Omega(2^n)$ gates/components, these gates are etched onto the chip and are performed without any external intervention—one simply needs to set the voltages on the n input pins specifying the desired address. It is possible to design a RAM circuit that dissipates only $O(n)$ energy, although since this does not represent a bottleneck in practical systems, actual RAM chips are better modeled as dissipating $O(\sqrt{2^n})$ energy (memory is laid out in 2D and in practice an entire row/column is activated, rather than just a single memory cell) [8], [40].

[5]We ignore speed-of-light constraints, which we expect only to be relevant at large QRAM size [41]. At large enough scale, the speed of light would prevent both RAM and QRAM from achieving query latency $O(n)$, since the dataset of size 2^n must be embedded in 2 or at most 3 spatial dimensions, and the time needed for information to travel across the device would be at least $\Omega(2^{n/3})$ (in the case of a 3D embedding).

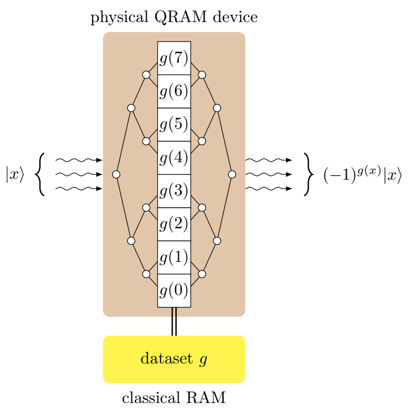
tolerant quantum computation (FTQC). The ability to apply the physical QRAM operation at computational cost $\mathrm{poly}(n)$ is not sufficient to justify the cheap QRAM assumption, even if there are no errors in the QRAM device itself (which is not realistic anyway). The problem is that, in FTQC, we need to perform the logical QRAM operation, denoted by $\overline{V(f)}$, onto an address register encoded into some quantum error-correcting (QEC) code. Naively, we could implement $\overline{V(f)}$ by un-encoding the n logical qubits into n physical qubits, running the physical qubits through the physical QRAM device, and re-encoding the output. However, the un-encoding and re-encoding processes introduce uncorrectable errors, and any noise in the physical QRAM device will also propagate into logical errors on the re-encoded state. An alternative would be to find a QEC code where the logical $\overline{V(f)}$ is a transversal gate, meaning it can be implemented as a tensor product of $\mathrm{poly}(n)$ physical $V(f)$ gates without the need for un-encoding and re-encoding. Unfortunately, there are known challenges to finding such codes [8]. A general n-qubit QRAM gate is in the n-th level of the Clifford hierarchy (see Section II-B and Appendix D of the full version [9]), and all known examples of codes supporting transversal implementation of a gate in the n-th level have $O(2^n)$ qubits [42]–[44]. In fact, there is at least one example of a gate in the n-th level—the single-qubit $\pi/2^n$ rotation gate—where a matching lower bound of $\Omega(2^n)$ qubits has been shown for a strong form of transversality [45], leading one to speculate that a similar lower bound may hold for QRAM, as well.

Our main contribution is to devise a protocol that implements the logical operation $\overline{V(f)}$ fault tolerantly, using $\mathrm{poly}(n)$ queries to a noisy device that can implement the physical QRAM operation with at least $1/\mathrm{poly}(n)$ fidelity, as well as $\mathrm{poly}(n)$ fault-tolerant operations on a general-purpose quantum processor—exponentially fewer than the number of fault-tolerant quantum operations required for circuit QRAM. The protocol generalizes well-known distillation–teleportation protocols for non-Clifford gates like the T gate and the CCZ gate. First, the physical $V(f)$ gate is used to prepare many copies of a faulty physical n-qubit QRAM resource state. Next, the physical resource states are encoded into a QEC code (the protocol is agnostic to which one) and distilled into a single high-fidelity logical resource state. Finally, the high-fidelity logical resource state is teleported into the computation to enact the logical QRAM gate, up to a correction which can be computed classically—our protocol outsources this calculation to a classical processor as depicted in Fig. 1b. The required correction is a different logical QRAM gate $\overline{V(f')}$, where f' is determined by f and random measurement outcomes obtained during the teleportation procedure. The correction $\overline{V(f')}$ is then implemented in the same way, requiring a correction of its own, $\overline{V(f'')}$, where f'' is again dependent on f' and random measurement outcomes. We show that after iterating this process for n rounds, no further correction is necessary. This is a consequence of the fact that, despite its exponential circuit complexity, the unitary $V(f)$ lies in the n-th level of the Clifford hierarchy [47] for every f, which implies that the first correction $V(f')$ is in the $(n-1)$-th level, the second correction $V(f'')$ is in the $(n-2)$-th level, and so on. Our insights are (i) to notice that these corrections always lie within the family of QRAM gates of Eq. (1), allowing for a straightforward recursive implementation, and (ii) to devise a method for preparing the high-fidelity encoded resource states, completing the end-to-end workflow for fault-tolerant $\overline{V(f)}$. A no-go theorem in Ref. [8] ruled out a wide class of QRAM distillation–teleportation protocols; our protocol sidesteps this theorem by being adaptive and querying the physical QRAM on different datasets (f, f', f'', etc.) in each round. We provide a more complete informal overview of the protocol in Section II and a detailed error analysis of each step in Section IV.

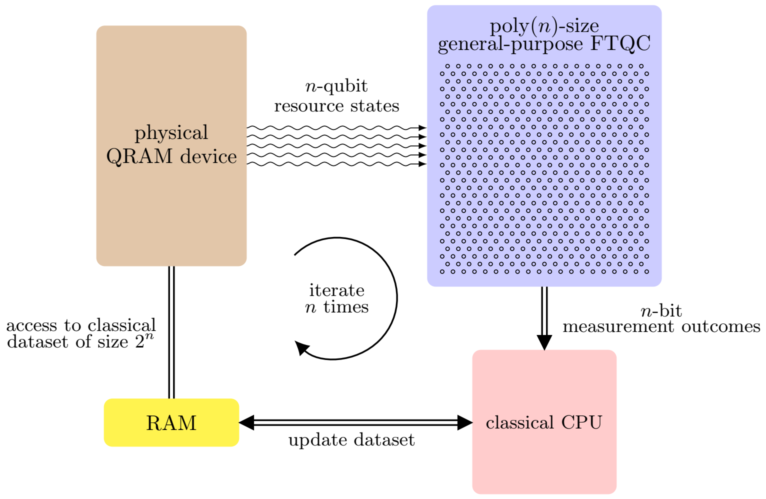
By showing how to perform logical $\overline{V(f)}$ using $\mathrm{poly}(n)$ calls to physical QRAM, our protocol salvages the potential utility of the specialized, faulty QRAM device, and it encourages a model of quantum computation where QRAM is performed separately from the main quantum processing unit.

Our protocol also *partially* justifies the cheap QRAM assumption, provided that a passive QRAM device can be constructed. Indeed, if noisy physical QRAM has computational cost $\mathrm{poly}(n)$, then the quantum resources required to implement fault-tolerant QRAM via our protocol also scales only as $\mathrm{poly}(n)$. The main caveat is that running our protocol requires a non-negligible amount of adaptive *classical* computation of complexity $O(2^n)$ to compute the required correction operations (and "reload" the passive QRAM device, so that it has access to the new classical dataset at the next round of the protocol), although this complexity may be amenable to some degree of parallelization. We explore the nuances of this caveat in Section V: like the RAM operation, this classical update operation has $\Omega(2^n)$ overall gate complexity, but admits an $O(n)$-depth classical circuit. However, certain features—namely, the wire density—of this parallel circuit are more demanding than that of RAM. We show an equivalence between the update rule and sparse matrix-vector multiplication, which clarifies when our scheme can and cannot provide advantage in applications. For our protocol, this adaptive classical computation and QRAM reloading appears necessary in order to avoid revealing which address is being queried even while using a noisy QRAM device. In Section VII, we pose the question of whether this reflects an inevitable limitation of fault-tolerant QRAM or whether a stronger justification of the cheap QRAM assumption, where both quantum and classical resources are $\mathrm{poly}(n)$, may be possible.

In any case, our protocol can be viewed as trading $O(2^n)$ quantum resources for $O(2^n)$ classical resources. That is, our protocol does not require the $O(2^n)$ actively error-corrected quantum resources incurred in circuit QRAM (fault-tolerant quantum gates, ancilla qubits, magic state factories, control wiring, classical co-processors, etc.). Instead, it requires $\mathrm{poly}(n)2^n$ purely classical resources in addition to only $\mathrm{poly}(n)$ fault-tolerant quantum resources and $\mathrm{poly}(n)$ queries



(a) Possible structure of physical QRAM device passively implementing $V(g)$ from Eq. (1)


physical QRAM device
n-qubit resource states
poly(n)-size general-purpose FTQC
iterate n times
access to classical dataset of size 2^n
n-bit measurement outcomes
RAM
update dataset
classical CPU

(b) Setup of our protocol

Fig. 1: (a) Ideally, the physical QRAM operation $V(g)$ of Eq. (1) is performed passively by a specialized device. We may imagine, for example, encoding the address state $\sum_x \alpha_x|x\rangle$ into the polarization states of n photons, and then sending them into a pre-manufactured device, where they return having picked up a -1 phase only on branches of the superposition where $g(x) = 1$ [8], [46]. This might be accomplished by placing the classical bits $g(0), g(1), \ldots, g(2^n - 1)$ at the leaves of a binary tree, selectively routing the photons to the correct leaf based on their polarization, picking up a phase if a photon exists at location x and $g(x) = 1$, and then unrouting the n photons. (b) Our protocol utilizes a specialized, physical QRAM device, which is separate from the general-purpose fault-tolerant quantum processor. The QRAM device is used to create QRAM resource states on n physical qubits which are moved onto the main processor. The main processor encodes, distills, and teleports these resource states, generating classical n-bit measurement outcomes, which are sent to a classical CPU. The classical CPU performs a calculation to update the dataset stored in classical memory (RAM), which is queried by the physical QRAM device at the next iteration of the protocol.

to a faulty QRAM device. There may be applications where such a tradeoff is beneficial, since quantum devices will be significantly slower and more expensive than classical devices for the foreseeable future.

II. Overview of Protocol

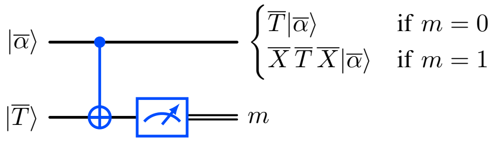
A. *Warm-up: distillation–teleportation protocol for the T gate*

In many schemes for FTQC, it is relatively cheap to implement logical Clifford gates (e.g., they can often be done transversally). On the other hand, non-Clifford logical gates like the T gate and the CCZ gate are more expensive; these gates can instead be performed using distillation–teleportation protocols. In this section, we review such a protocol for the T gate, a diagonal gate mapping $|0\rangle \mapsto |0\rangle$ and $|1\rangle \mapsto e^{i\pi/4}|1\rangle$. Although the CCZ gate is actually a special case of the QRAM operation from Eq. (1) and thus more directly related to our protocol, the T gate provides a gentler introduction because it is a single-qubit gate.

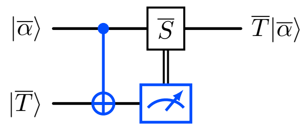
We discuss distillation and gate teleportation separately, beginning with gate teleportation. Henceforth, we denote logical states and operations with an overline, for example, we denote the logical T gate by $\overline{T}$. Teleporting $\overline{T}$ into a quantum computation requires the preparation of a resource state, also known as a magic state, which is $\overline{T}$ applied to the equal superposition state:

$$|\overline{T}\rangle = \overline{T}|\overline{+}\rangle = \frac{1}{\sqrt{2}}\left(|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle\right), \tag{2}$$

where $|\overline{0}\rangle$ and $|\overline{1}\rangle$ denote the encoded computational basis for some QEC code (here we are agnostic to which code), and $|\overline{+}\rangle = \frac{1}{\sqrt{2}}(|\overline{0}\rangle + |\overline{1}\rangle)$. The $\overline{T}$ gate can then be applied to an arbitrary quantum state $|\overline{\alpha}\rangle$ (on one logical qubit) by entangling $|\overline{\alpha}\rangle$ with $|\overline{T}\rangle$ and making a (logical) measurement, as follows:


$|\overline{\alpha}\rangle$
$|\overline{T}\rangle$
$\begin{cases}\overline{T}|\overline{\alpha}\rangle & \text{if } m=0\\ \overline{X}\,\overline{T}\,\overline{X}|\overline{\alpha}\rangle & \text{if } m=1\end{cases}$
m
(3)

The logical CNOT gate (a Clifford gate) and the single-qubit logical measurement are both performed fault-tolerantly within the QEC code to ensure negligible chance of logical error. Direct computation verifies that the single-qubit measurement outcome $m \in \{0, 1\}$ is uniformly random, regardless of the state $|\overline{\alpha}\rangle$. If the measurement outcome is $m = 0$, the gate $\overline{T}$ is exactly implemented on the top wire. However, if the outcome $m = 1$ is obtained, the wrong phase was applied to the state, equivalent to the gate $\overline{X}\,\overline{T}\,\overline{X}$ instead of $\overline{T}$ (where X, Y, Z denote the Pauli operators). To fix this, one must apply a *correction* operation when the measurement outcome is 1. The correction required to undo the erroneous $\overline{X}\,\overline{T}\,\overline{X}$ gate and re-do the $\overline{T}$ gate is the $\overline{T}\,\overline{X}\,\overline{T}^\dagger\overline{X}$ gate, which is equal to the phase gate $\overline{S} = \overline{T}^2$, up to a global phase. Crucially, the phase gate is a Clifford gate, and thus the logical $\overline{S}$ can typically be implemented fault tolerantly in a more direct fashion. The full gate teleportation circuit with the correction is then given by:


$|\overline{\alpha}\rangle$
$\overline{S}$
$\overline{T}|\overline{\alpha}\rangle$
$|\overline{T}\rangle$
(4)

The benefit of performing the $\overline{T}$ gate via gate teleportation is that the difficulty is reduced to preparing a high-fidelity $|\overline{T}\rangle$ state. This state can be prepared through a multistep process of physical preparation, encoding, and then distillation. For concreteness, one can consider magic state injection schemes for the surface code [48]–[51]. Here, the first step is to prepare the $|T\rangle$ state on a single physical qubit. Next, an encoding procedure is performed, that is, $|T\rangle$ is mapped to $|\overline{T}\rangle$, which is encoded in a $d \times d$ surface code patch. This can be realized by, for instance, preparing a product state and performing appropriate stabilizer measurements. This procedure is generally not fault-tolerant; if the underlying hardware has error rate p, the logical error on the prepared state is $O(p)$, but the logical error can be kept independent of how large one makes the code distance d. Some of the possible logical errors are heralded—they can be detected by applying certain checks, in which case the procedure can be restarted from scratch, improving the postselected fidelity. The final step is magic state distillation, whereby multiple noisy $|\overline{T}\rangle$ states are consumed to produce a smaller number of higher-fidelity $|\overline{T}\rangle$ states. For example, the 15-to-1 magic state distillation protocol uses 15 input magic states of error rate p_{in}, succeeds with probability $1 - O(p_{\text{in}})$, and conditioned on success, produces a single output magic state of error rate $p_{\text{out}} = O(p_{\text{in}}^3)$ [51]–[53]. By recursively applying this protocol, one can distill $|\overline{T}\rangle$ states with arbitrarily low error rate even when all physical components have noise rate $p = O(1)$, provided that p is below a certain threshold for state distillation.

In some instances, one may not want to perform the corrective $\overline{S}$ gate directly.[6] In this case, another option is to perform the $\overline{S}$ gate also via gate teleportation, using the resource state

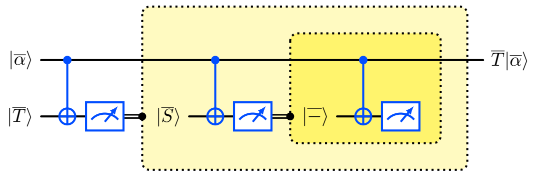
$$|\overline{S}\rangle = \overline{S}|\overline{+}\rangle = \frac{1}{\sqrt{2}}(|\overline{0}\rangle + \mathrm{i}|\overline{1}\rangle)\,. \tag{5}$$

[6]For some codes, such as the color code [54], the $\overline{S}$ gate is transversal; for the surface code, however, it is fold-transversal [55], [56], and therefore more challenging to implement. One may also prefer to use autocorrected gadgets [57], [58] that avoid direct implementation of $\overline{S}$.

The conditional correction required when teleporting $\overline{S}$ is the gate $\overline{S}\,\overline{X}\,\overline{S}^\dagger\overline{X} \propto \overline{S}^2 = \overline{Z}$. While implementing the Pauli $\overline{Z}$ gate is typically easy for FTQC schemes, it could in principle also be implemented by teleporting the resource state

$$|\overline{-}\rangle = \overline{Z}|\overline{+}\rangle = \frac{1}{\sqrt{2}}(|\overline{0}\rangle - |\overline{1}\rangle)\,. \tag{6}$$

Teleporting the $|\overline{-}\rangle$ state requires no correction, regardless of the measurement outcome, since $\overline{Z}\,\overline{X}\,\overline{Z}\,\overline{X} \propto \overline{\mathbb{I}}$, where $\overline{\mathbb{I}}$ is the (logical) identity operator. Following this strategy, we can write the following circuit, which implements the $\overline{T}$ gate via three successive teleportations, where the second and third teleportations are applied only if all prior measurement outcomes are 1.


$|\overline{\alpha}\rangle$
$|\overline{T}\rangle$
$|\overline{S}\rangle$
$|\overline{-}\rangle$
$\overline{T}|\overline{\alpha}\rangle$
(7)

This approach may strike the reader as unnecessary, but designing the procedure in this iterative way will mirror the structure of our full protocol for QRAM.

B. Teleportable gates and the Clifford hierarchy

Not all gates can be teleported in the manner of circuit (4). The key reason it works is that the correction operation, $\overline{S}$, is a Clifford gate. In general, if one attempts to teleport a diagonal single-qubit logical gate $\overline{G}$ in this fashion, the conditional correction is $\overline{G}\,\overline{X}\,\overline{G}^\dagger\,\overline{X}$ [59, Appendix A.1]. Early work on gate teleportation [60] characterized the set of teleportable gates. It identified a hierarchy of teleportable gates known as the Clifford hierarchy. Focusing here on logical gates for consistency with the rest of this section, the logical Clifford hierarchy is a sequence of sets $\mathcal{C}_k$ for $k = 1, 2, \ldots$, where $\mathcal{C}_1$ is the set of logical Pauli gates, and $\mathcal{C}_k$ is defined recursively by

$$\mathcal{C}_k = \{\overline{G} \colon \overline{G}\,\overline{P}\,\overline{G}^\dagger \in \mathcal{C}_{k-1} \text{ for all } \overline{P} \in \mathcal{C}_1\}\,. \tag{8}$$

That is, the k-th level of the Clifford hierarchy are gates that, under conjugation, transform Pauli gates into gates in the $(k-1)$-th level. We may recognize $\mathcal{C}_2$ as the set of gates that transform Paulis to Paulis—that is, the set of Clifford gates. The $\overline{T}$ gate lies in $\mathcal{C}_3$ because $\overline{T}\,\overline{X}\,\overline{T}^\dagger = \mathrm{e}^{-\mathrm{i}\pi/4}\overline{S}\,\overline{X}$ is Clifford, $\overline{T}\,\overline{Y}\,\overline{T}^\dagger = \mathrm{e}^{-\mathrm{i}\pi/4}\overline{S}\,\overline{Y}$ is Clifford, and $\overline{T}\,\overline{Z}\,\overline{T}^\dagger = \overline{Z}$ is Pauli (and therefore Clifford).

Focusing here on single-qubit diagonal gates, if a gate $\overline{G}$ lies in $\mathcal{C}_k$, then the teleportation procedure calls to use the state $\overline{G}|\overline{+}\rangle$ as a resource state. The conditional correction $\overline{G}\,\overline{X}\,\overline{G}^\dagger\,\overline{X}$ is also diagonal and lies in $\mathcal{C}_{k-1}$. As pointed out already in Refs. [59], [60], this immediately yields a recursive procedure for implementing any gate in $\mathcal{C}_k$: prepare $\overline{G}|\overline{+}\rangle$; teleport; if outcome 1 is obtained, classically compute the required correction $\overline{G}' = \overline{G}\,\overline{X}\,\overline{G}^\dagger\,\overline{X} \in \mathcal{C}_{k-1}$; prepare

$\overline{G}'|\overline{+}\rangle$; teleport; if outcome 1 is obtained, compute the required correction $\overline{G}'' = \overline{G}'\,\overline{X}\,\overline{G}'^\dagger\,\overline{X} \in \mathcal{C}_{k-2}$, etc. Each correction is one level lower in the hierarchy than the last. After enough rounds, no further correction will be required, as in circuit (7).

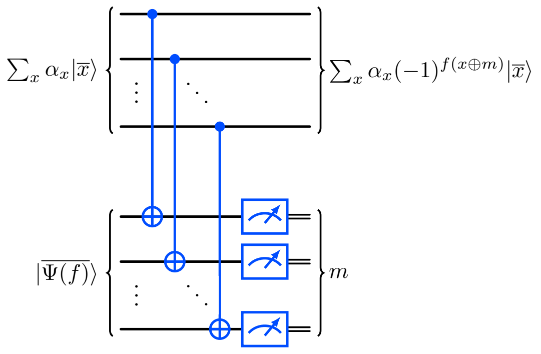
C. Teleporting the QRAM gate

The teleportation strategy for single-qubit diagonal gates can also be applied to multi-qubit diagonal gates (see, e.g., Ref. [59, Appendix A.1]), such as the QRAM unitary $\overline{V(f)}$ from Eq. (1). We define *QRAM resource states* analogously to the resource state $|T\rangle$ (cf. Eq. (2)).

$$|\Psi(f)\rangle = V(f)|+\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_{x\in\{0,1\}^n} (-1)^{f(x)}|x\rangle\,. \tag{9}$$

We denote the encoded logical QRAM resource state by $|\overline{\Psi(f)}\rangle$.

Assuming that we can prepare the encoded resource state $|\overline{\Psi(f)}\rangle$, then we may teleport the QRAM gate into an arbitrary encoded n-qubit state $\sum_x \alpha_x|\overline{x}\rangle$ by making an entangled measurement, as in the following circuit (cf. circuit (3)).



$\sum_x \alpha_x|\overline{x}\rangle$
$\sum_x \alpha_x(-1)^{f(x\oplus m)}|\overline{x}\rangle$
$|\overline{\Psi(f)}\rangle$
m

(10)

Here we emphasize that in the context of our protocol, the circuit is a logical circuit: all qubits are logical qubits, and both the upper and lower sets of n logical qubits are constructed out of $n' > n$ physical qubits using some QEC code. For example, one could choose to encode each logical qubit into its own $d \times d$ surface code patch, giving $n' = nd^2$ for that example. The n logical CNOT gates and n logical single-qubit measurements in circuit (10) are performed fault tolerantly within this code, allowing us to neglect the chance of logical error.

Each possible n-bit measurement outcome $m \in \{0,1\}^n$ is obtained with uniform probability $1/2^n$, regardless of the state $\sum_x \alpha_x|\overline{x}\rangle$. If $m = 0^n$ is obtained, then by direct calculation (see Section IV-E), we can verify that the gate $\overline{V(f)}$ has been correctly applied, yielding the state $\sum_x \alpha_x(-1)^{f(x)}|\overline{x}\rangle$. However, most of the time, we obtain a nonzero measurement outcome $m \neq 0^n$, in which case the phase $(-1)^{f(x)}$ is applied onto the basis state $|\overline{x\oplus m}\rangle$ rather than $|\overline{x}\rangle$, yielding the state $\sum_x \alpha_x(-1)^{f(x\oplus m)}|\overline{x}\rangle$. Here and throughout, $\oplus$ denotes bitwise addition, modulo 2.

To correct for this, we need to apply the phase $(-1)^{f(x\oplus m)\oplus f(x)}$ onto the basis state $|\overline{x}\rangle$ for each x; that is, we need to implement the correction operation $\overline{V(f')}$, where f' is a Boolean function defined by the rule

$$f'(x) = f(x) \oplus f(x\oplus m)\,. \tag{11}$$

The function f' depends on m, and thus it can only be determined after the teleportation of $|\overline{\Psi(f)}\rangle$ has been performed. To tie back to the case of a single-qubit diagonal gate $\overline{G}$ discussed in Section II-B, where the conditional correction was $\overline{G}' = \overline{G}\,\overline{X}\,\overline{G}^\dagger\,\overline{X}$, we can note that $\overline{V(f)}^\dagger = \overline{V(f)}$ and rewrite

$$\overline{V(f')} = \overline{V(f)}\,\overline{X}^m\,\overline{V(f)}^\dagger\,\overline{X}^m\,, \tag{12}$$

where X^m denotes the n-qubit Pauli operator with Pauli-X in positions where $m_i = 1$ and identity operator $\mathbb{I}$ in positions where $m_i = 0$, such that $\overline{X}^m|\overline{x}\rangle = |\overline{x\oplus m}\rangle$.

It now suffices to observe that for any f, the n-qubit unitary $\overline{V(f)}$ is in the n-th level of the logical Clifford hierarchy [8], [47]; we provide a self-contained proof of this in Appendix D of the full version [9]. This guarantees that the correction $\overline{V(f')}$ will be in the $(n-1)$-th level. As explained in Appendix D of the full version [9], the reason this holds is related to the degree of the Boolean functions f and f', when they are expanded as a polynomial of their n input bits. Specifically, we may observe that the highest-degree monomials in the expansion of $f(x)$ are the same as those in the expansion of $f(x\oplus m)$. Thus, when $f'(x)$ is defined as $f(x)\oplus f(x\oplus m)$, the highest-degree monomials all cancel out, leaving only monomials of a lower degree. That is, the degree of f' is smaller than the degree of f by at least one. We use this fact to prove in general that if a Boolean function h has degree d, then $\overline{V(h)} \in \mathcal{C}_d$. In particular, since $\overline{V(f)} \in \mathcal{C}_n$ (the maximum possible degree of any function is n), we have that $\overline{V(f')} \in \mathcal{C}_{n-1}$.

Our protocol proposes to implement the correction $\overline{V(f')}$ in the same fashion as $\overline{V(f)}$: by preparing the resource state $|\overline{\Psi(f')}\rangle$ and teleporting as in circuit (10). This will also produce a correction, associated with a Boolean function f'' of degree $n-2$. As we iterate, we descend the Clifford hierarchy, and the degree of our correction function is reduced. Once we have performed n rounds of teleportation, our correction function has degree zero. If a Boolean function h is constant, this implies that $\overline{V(h)} \propto \overline{\mathbb{I}}$; thus, once we have reduced the correction function to degree zero, we may cease iterating the protocol.

Later, in Section IV-E, we perform a more complete analysis of the teleportation channel; for example, we quantify the error in the teleportation channel when an imperfect resource state is teleported instead of $|\overline{\Psi(f)}\rangle$.

D. Preparing the encoded QRAM resource state

The analysis above shows how we can implement the logical $\overline{V(f)}$ gate, provided that we can adaptively prepare the resource states $|\overline{\Psi(g)}\rangle$, up to low error, for any particular Boolean function g. At first glance, this seems like a tall task. There are 2^{2^n} different states that we may need to prepare. By a simple counting argument, the quantum circuit complexity of at least one of these states is at least $\Omega(2^n/n)$. The innovation of our protocol is to outsource this complexity to a single-purpose, faulty (and ideally passive) QRAM device, which may be able to exploit the unique structure of QRAM to implement $V(g)$ cheaply, but imperfectly.

We propose a three-step procedure for preparing these states, analogous to the preparation of the $|\overline{T}\rangle$ state: physical preparation, encoding, and distillation.

- **Physical preparation**: we assume that we have access to a QRAM device that can implement an approximation to $V(g)$ at the physical level, as discussed in Section I and Fig. 1. By running this device on the initial input state $|+\rangle^{\otimes n}$, we produce the physical resource state $|\Psi(g)\rangle$ of Eq. (9). The device can be faulty. In fact, our protocol can succeed even when the device produces states that have low (asymptotically vanishing) $1/\mathrm{poly}(n)$ minimum fidelity with respect to $|\Psi(g)\rangle$.
- **Encoding**: The physical n-qubit state is not protected by a QEC code, and thus it is vulnerable to error. We immediately encode it into (an approximation of) the logical state $|\overline{\Psi(g)}\rangle$ using some number $n' > n$ of physical qubits on our main quantum processor. This step incurs some additional logical error because encoding arbitrary states is not fully fault tolerant. However, for topological codes like the surface code, there exist effective methods for encoding a physical qubit into a logical qubit [50]. The logical error due to encoding is $O(p)$—independent of the code distance—where p is the physical error rate. In Section IV-B, we use the general results of Ref. [61] to formalize the error in this step. Since the state $|\overline{\Psi(g)}\rangle$ is an n-qubit state, we expect the total logical error incurred from encoding to be $O(np)$, although for the case of general codes, we can only show $O(n\sqrt{p})$. The physical error rate must be $p = O(1/n)$ or $p = O(1/n^2)$, so that the total error from encoding remains $O(1)$, but for relevant sizes of n (e.g., $n = 43$ already corresponds to one terabyte of QRAM), the $p = O(1/n)$ condition is already met on devices that exist today.
- **Distillation**: Distillation procedures [52], [53] for the $|\overline{T}\rangle$ (or $|\overline{\mathrm{CCZ}}\rangle$) state leverage the existence of QEC codes where T (or CCZ) is transversal. The overhead, that is, the number of noisy copies of $|\overline{T}\rangle$ needed to distill one $\varepsilon_{\mathrm{dist}}$-good copy of $|\overline{T}\rangle$ is $\mathrm{polylog}(1/\varepsilon_{\mathrm{dist}})$, and this can be improved to $O(1)$ overhead using high-rate codes [62]–[64]. For the $V(g)$ gate, we do not know of any suitable codes that would enable this kind of approach. However, we can still distill $|\overline{\Psi(g)}\rangle$ using state-agnostic *quantum purity amplification* methods [65]–[71], which take many copies of an arbitrary mixed state $\overline{\rho}$ and produce one $\varepsilon_{\mathrm{dist}}$-good copy of the pure state $|\overline{\Xi}\rangle\langle\overline{\Xi}|$, where $|\overline{\Xi}\rangle$ is the principal component (i.e., top eigenvector) of $\overline{\rho}$. These methods do not leverage or learn any properties of $|\overline{\Xi}\rangle$, and it is known that the optimal overhead achievable in such settings is $\Theta(1/\varepsilon_{\mathrm{dist}})$ [70]. In Section IV-D, we discuss several specific state-agnostic approaches. We first consider the iterated swap test purification method studied in Refs. [68], [69], [71], [72], which is appealing for its simplicity. In the regime where the physical preparation and encoding steps prepare states with high (but still imperfect) fidelity, the iterated swap test approach is nearly optimal. On the other hand, as the fidelity of the undistilled input states decreases, the overhead of the iterated swap test rapidly increases, scaling exponentially in the inverse input fidelity. To alleviate this issue, we propose a new gate-efficient state-agnostic quantum purity amplification procedure based on quantum principal component analysis [73], [74], which achieves nearly optimal sample complexity even in the regime of low input fidelity, while still being compatible with the streaming model (i.e., where the undistilled input states are processed one at a time, rather than all at once as in the known sample-optimal protocol [70]).

To apply these state-agnostic distillation approaches within our protocol, it must be the case that the state that is output by the physical preparation and encoding processes has the ideal resource state $|\overline{\Psi(g)}\rangle$ as its principal component. Evaluating this assertion requires specifying a noise model in our abstract QRAM device and in our main quantum processor. We suppose that our main processor is subject to circuit-level stochastic noise. For the QRAM device, the only assumption we make is that the noise is independent of the dataset, in the sense that, for dataset g, it enacts the n-qubit quantum channel $\mathcal{N}_2 \circ \mathcal{V}(g) \circ \mathcal{N}_1$, where $\mathcal{N}_{1,2}$ are g-independent noise channels, and $\mathcal{V}(g) = V(g)[\cdot]V(g)^\dagger$ is the ideal QRAM channel. In this case, we can ensure that the principal component of the state we prepare is $|\overline{\Psi(g)}\rangle$ by performing a *partial Clifford-twirl* of the unitary $V(g)$. This method leverages the fact that for any Clifford circuit C formed from Z, X, CZ, and CX (i.e., CNOT) gates, we have $|\overline{\Psi(g)}\rangle = C|\overline{\Psi(g_C)}\rangle$ for some dataset g_C; the idea is to choose a random C, compute the dataset g_C, query g_C with the QRAM device, and then apply $\overline{C}$ fault-tolerantly to restore $|\overline{\Psi(g)}\rangle$. Partial Clifford twirling is not necessary under the stronger assumption that the noise in the QRAM device naturally guarantees that the ideal resource state is the principal component.

It is important that our protocol can work even when the QRAM device has low (at least inverse polynomial) fidelity. Given the engineering challenges associated with building a reliable physical QRAM device, it is much easier to imagine realizing our protocol in practice, especially as n grows, if the physical QRAM device need only have a small correlation with the correct output. Along these lines, another key benefit of a distillation–teleportation approach to fault-tolerant QRAM is that one always has the option to restart the preparation, encoding, and distillation procedure if an error is detected. For instance, if the physical QRAM device recognizes certain errors (e.g., photon loss), one can simply postselect on these events not occurring, improving the effective fidelity of the device from the perspective of our protocol.

E. Full summary of protocol and statement of results

To summarize, our main result is a protocol for implementing the logical QRAM operation $\overline{V(f)}$, up to arbitrarily high fidelity, using many queries to a device that can perform the physical QRAM operation $V(g)$ (for any/all g) with a lower nonzero fidelity. As discussed in Section I, the physical QRAM operation could be accomplished with a dedicated subcomponent of the larger quantum device specialized for QRAM, which need not be capable of universal fault-tolerant quantum computation.

The protocol to implement $\overline{V(f)}$ cycles at most n times through five steps discussed in the previous subsections: (i) physical preparation, (ii) encoding, (iii) distillation, (iv) teleportation, and (v) adaptive classical computation of the correction. Step (v) uses measurement outcomes from step (iv) to transform the dataset according to the *update rule* (UR) of Eq. (11), prior to returning to step (i). The entire protocol is depicted in Fig. 2, where each of the five steps is shown in a different color. A more detailed specification and formal error analysis of each step is provided in Section IV. We arrive at the following statement of the cost of implementing $\overline{V(f)}$.

Theorem 1 (Main result (informal)): For any data table f with 2^n entries, and any error parameter $\varepsilon > 0$, the protocol performs the logical QRAM operation $\overline{V(f)}$ up to error ε (in diamond distance), under the assumption that the physical QRAM device implementing physical $V(f)$ has noise independent of f. The quantum resources required are:

- $\text{poly}(n)/\varepsilon$ calls to a device that performs the physical $V(g)$, for various g (determined adaptively) with any nonzero minimum fidelity $F \geq 1/\text{poly}(n)$.
- $\text{poly}(n)/\varepsilon$ calls to a $\text{poly}(n)$-cost fault-tolerant encoding procedure that encodes n-qubit physical states into a suitable QEC code capable of FTQC, while incurring at most $O(1)$ logical error.
- $\text{poly}(n)/\varepsilon$ fault-tolerant one- and two-qubit logical gates, single-qubit logical $|\overline{0}\rangle$ state preparations, and single-qubit logical measurements.

The classical resources required are:

- n applications of the classical update rule, each of which has $O(2^n)$ complexity in a standard RAM model.
- $\text{poly}(n)/\varepsilon$ twirling operations on the dataset, each of which has complexity $\text{poly}(n)2^n$ in a standard RAM model.

After each update rule and twirling operation, the physical QRAM device must be "reloaded" or otherwise given access to the updated classical dataset.

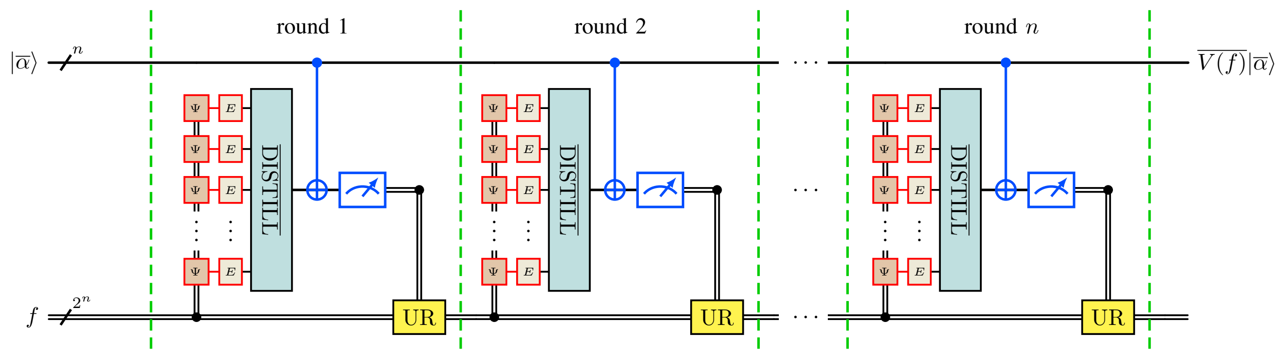
The main implication of this result is the following: suppose that a quantum algorithm calls the QRAM operation $V(f)$ at most $T = \text{poly}(n)$ times, and suppose that one has access to a QRAM device that approximately performs the physical QRAM operation with at least $1/\text{poly}(n)$ fidelity, at computational cost $\text{poly}(n)$ (similar to the cost of RAM). Then, one may take $1/\varepsilon = O(T) = \text{poly}(n)$, and conclude that the algorithm can be implemented fault-tolerantly using only $\text{poly}(n)$ quantum resources. As a result, our protocol provides a step toward justifying the cheap QRAM assumption, and it provides a method of fault-tolerantly implementing quantum algorithms that depend on QRAM.

Even in a cost model where noisy physical QRAM incurs computational cost $\Omega(2^n)$—for instance, if the physical QRAM has $\Omega(2^n)$ active gates each requiring $\Omega(1)$ energy input—our protocol still provides the benefit that the exponential quantum complexity is contained entirely to physical quantum operations that can be optimized specifically to perform QRAM. There is no need for an exponential amount of QEC and the associated overheads it incurs.

1) Caveat: classical complexity: The main caveat of our protocol is that it requires a non-negligible amount of purely classical adaptive computation. In particular, after receiving random measurement outcome m, the protocol requires replacing the value $g(x)$ with the value $g(x) \oplus g(x \oplus m)$ for all 2^n addresses x of the dataset, as in Eq. (11). While computing the new value is easy for any individual x, the sheer number of different x means the complexity—in terms of classical circuit size or RAM calls to the dataset g—is at least $\Omega(2^n)$.

However, we must recall that naively, the logical QRAM requires $\Omega(2^n)$ fault-tolerant quantum resources, if implemented as a fault-tolerant circuit. One unit of fault-tolerant quantum resources, such as one fault-tolerant quantum gate, is expected to be several orders of magnitude more expensive in terms of both financial cost and computational runtime than one unit of classical computation, such as a classical gate or floating point operation [75]. Thus, trading $\Omega(2^n)$ quantum for $\Omega(2^n)$ classical resources may lead to an overall cheaper and faster computation.

Furthermore, we expect that although it formally has $\Omega(2^n)$ computational cost, the complexity of the classical update rule has significantly better constant prefactors than the classical computation required to power active QEC of an entire QRAM circuit. As mentioned previously, circuit QRAM with $\text{poly}(n)$ latency would require a fault-tolerant quantum computer with $\Omega(2^n)$ logical qubits. Such a device would likely require $\Omega(2^n)$ full-fledged *classical* chips to be co-located with the logical qubits, in order to process in parallel the QEC syndrome data generated by the computation in real time. For example, in the surface code operating at a 1 MHz QEC cycle rate, the amount of syndrome data generated by 2^{20} logical qubits, each encoded into its own patch at code distance 11, would be more than 15 terabytes per second. Specialized classical decoding algorithms must be run continuously to identify and correct errors as they occur. In contrast, for a dataset of size 2^{20} bits, the classical update rule in our protocol is a single structured transformation of a 120 kilobyte dataset. The only interaction between this dataset and the quantum



round 1
round 2
round n
$|\overline{\alpha}\rangle$
n
$\overline{V(f)}|\overline{\alpha}\rangle$
Ψ
E
DISTILL
UR
f
2^n

Fig. 2: Quantum circuit depiction of the protocol for implementing the logical diagonal QRAM operation $\overline{V(f)}$ (see Eq. (1)) fault tolerantly for a data table f. The protocol cycles through n rounds, and each round has five steps: preparation, encoding, distillation, teleportation, and classical update, depicted in different colors. All gates are fault-tolerant, logical gates, except the query to the noisy QRAM device Ψ and the encoding step E, outlined in red. The solid black wires represent encoded logical quantum registers of n logical qubits, the red wires represent unencoded quantum registers of n physical qubits, and the double black lines represent classical registers. For simplicity, we have not depicted the twirling step in the figure (which may not be necessary in practice), where prior to each application of Ψ, the dataset is modified by an independently chosen random transformation, which is corrected for after E has been applied; see circuit (48).

processor is the reloading of the physical QRAM device with the updated dataset.

Additionally, because of the structure of the classical update rule $g(x) \mapsto g(x) \oplus g(x \oplus m)$, it is conceivable that dedicated classical chips could be built to parallelize the process of performing the update and the reloading of the QRAM device. In Section V, we analyze the complexity of the update rule, and illustrate how it can be implemented with a classical circuit of depth $\mathrm{poly}(n)$, although embedding this circuit into 2 or 3 spatial dimensions leads to asymptotically growing wire density. We show how in a model of parallel computation, the update rule is equivalent (up to $\mathrm{poly}(n)$ factors) to performing sparse matrix-vector multiplication, with a particularly close connection to the (fast) Walsh–Hadamard transform. This fact helps to understand the expected difficulty of parallelization and it clarifies the opportunity cost of these classical resources.

2) Comments on scalability: An additional caveat is the fact that our protocol is likely not to be fully scalable for indefinitely large n. This stems from two aspects, the physical QRAM device, and the encoding step.

First, the physical size of an (ideally passive) physical QRAM device would need to grow as $\Omega(2^n)$, yet we need it to produce physical QRAM resource states with at least $1/\mathrm{poly}(n)$ fidelity. Thus, the fidelity of the individual device components needs to improve as n grows. It is known that certain architectural approaches to QRAM, namely, bucket brigade QRAM, possess a certain noise resilience property: the overall infidelity of the physical QRAM operation scales as $O(qn^2)$ [37], where q is the error rate of the individual router components that compose the device. This is exponentially better than $O(q2^n)$, which would be the naive expectation, given the exponential number of error-prone routers in the device. This noise resilience is a crucial fact for the possibility of practical QRAM. If this kind of scaling is achieved, then the physical per-component error rate q must decrease asymptotically roughly as $O(1/n^2)$ to be useful for our protocol.

Second, the physical QRAM resource state is an n-qubit state, and in a noise model where each operation on our main quantum processor fails with probability p, the encoding of this n-qubit physical state into an n-qubit logical state necessarily incurs at least $\Omega(np)$ logical error. The formal analysis, later, shows how $O(n\sqrt{p})$ can be achieved regardless of the choice of QEC code. Either way, p must decrease as $O(1/n)$ or as $O(1/n^2)$ to keep this error of total size $O(1)$.

While any practical implementation of this protocol will certainly need to pay close attention to error rates at every step, this is not a hugely debilitating conceptual issue for QRAM. This is because we do not ever expect to need to build a QRAM device for very large values of n. For example, typical RAM devices in classical computers are of size roughly 10 gigabytes, corresponding to only $n = 36$. The physical error rate in state-of-the-art quantum devices in several different platforms already achieves p in the range of 10^{-3}–10^{-2}. Improving by roughly an order of magnitude to $p = q = 10^{-4}$ would be sufficient to enable our protocol at size $n = 36$, assuming that the dominant error contribution scales as $4qn^2$ (where the presumed constant prefactor of 4 is chosen to align with Ref. [37, Eq. (28)]).

3) Comments on applications: Our investigation has been primarily motivated by the goal of evaluating the viability of QRAM as a primitive for fault-tolerant quantum computation in an abstract sense. Nonetheless, in Section VI, we consider whether our protocol could provide a practical advantage over alternative methods in several concrete applications. Generally, although our protocol achieves asymptotically polynomial $\mathrm{poly}(n)/\varepsilon$ complexity, we find that this version of the protocol struggles to provide an immediate advantage. For example, in quantum machine learning scenarios, the $\Omega(2^n)$ cost of the classical update rule makes it difficult to find examples where an end-to-end speedup persists over alternative classical methods. The observation in Section V that the update rule is similar in power to a sparse matrix-vector multiplication clarifies that, in our search for super-polynomial quantum speedups, we must only target problems where the ability to perform classical $2^n \times 2^n$ sparse matrix-vector multiplications is not already sufficient to solve the problem in $\mathrm{poly}(n)$ time, which considerably reduces the set of candidates. See Section VI-B for comments on possible scenarios where this conclusion may be avoided.

On the other hand, in scenarios like quantum chemistry and cryptanalysis where QRAM is utilized—in that context often referred to as a "quantum lookup table" or "quantum read-only memory"—the $\Omega(2^n)$ classical cost is tolerable. In fact, in these instances, it is typically already being proposed to implement QRAM with a fully error-corrected quantum circuit of depth $\Omega(2^n)$ [76]. Our protocol could allow this exponential fault-tolerant complexity to be offloaded to a specialized physical QRAM device and a classical computer. However, in our preliminary resource analysis at relevant system sizes in Section VI, the $\mathrm{poly}(n)/\varepsilon$ cost is still too large to provide an actual advantage. Part of the issue is that if the QRAM operation is called T times, one must take $1/\varepsilon = \Omega(T)$, and hence the total cost of implementing all T QRAM queries scales as T^2. The discovery of a distillation protocol for QRAM resource states with overhead $\mathrm{polylog}(1/\varepsilon)$ instead of $1/\varepsilon$ would be extremely beneficial in this calculation.

4) Extension to multiple output bits: In Appendix A of the full version [9], we explain how our protocol can be straightforwardly extended to the case where b classical bits are stored at each of 2^n addresses, and one wishes to coherently read all b bits into a separate bus register. That is, we show how to fault-tolerantly perform the operation $\overline{U(f)}$ that implements $|\overline{x}\rangle|\overline{u}\rangle \mapsto |\overline{x}\rangle|\overline{u \oplus f(x)}\rangle$, with $f\colon \{0,1\}^n \to \{0,1\}^b$ here denoting a function with b output bits. The strategy is to observe that conjugating $U(f)$ by a Hadamard transform on the bus register yields a diagonal unitary with ± 1 on the diagonal that may be viewed as a generalization of $V(f)$ from Eq. (1). The unitary acts on $n+b$ qubits rather than n qubits, but importantly, the degree of the Boolean function is at most $n+1$, which can be much smaller than $n+b$. The protocol proceeds identically to how it is described in the main text, except that the resource states are larger, requiring $n+b$ qubits, which leads to greater gate complexity overhead when performing distillation and teleportation.

F. Relation to prior work

The idea of QRAM was first formalized by Giovannetti, Lloyd, and Maccone (GLM) in Refs. [7], [77] (although some primitive versions of QRAM had been sketched earlier, see e.g. Ref. [78, Chapter 6]). These works first introduced the idea of a dedicated QRAM hardware element—a device specially designed for QRAM and separate from the main quantum processor—by proposing implementations based on optical and atomic hardware. Many other proposals have followed, including proposals based on superconducting circuits [79]–[82], photonic systems [83], [84], and neutral atom arrays [85] (see Ref. [8] for a more detailed review). We highlight that notions of teleportation-based QRAM [83] and QRAM resource states [85]—albeit resource states of size exponential in n—have previously been proposed. Unfortunately, all of these proposed QRAM implementations face daunting practical challenges, and most are not passive, meaning they require active control over $\Omega(2^n)$ quantum components, which would undermine the cheap QRAM assumption. (As Ref. [8] notes, the proposal of Ref. [82] is a noteworthy example of a passive implementation, although it faces challenges of scalability and practicality.) To our knowledge, there has not yet been a proposed implementation of a physical QRAM device that is simultaneously practical, scalable, and passive.

The initial GLM QRAM papers also sparked a long-running debate about the practicality of QRAM and validity of the cheap QRAM assumption, especially in relation to error correction and fault tolerance. In particular, GLM proposed a specific QRAM architecture—the "bucket brigade" QRAM—that they argued was intrinsically robust to errors. This claim was initially met with some skepticism (see, e.g., Ref. [33]), but the robustness was later proven in Ref. [37], which showed that the overall error of a bucket-brigade QRAM query scaled only with $\mathrm{poly}(n)$, despite the fact that the QRAM itself is comprised of $\Omega(2^n)$ error-prone components. This robustness is key to the viability of our own proposal, since the passive QRAM device in Fig. 1a could indeed have only moderate overall error rates compatible with our distillation–teleportation scheme.

Even with some intrinsic robustness against errors, QRAM is still likely to require QEC in most applications. As mentioned in Section I, fault-tolerant implementations of QRAM based on circuit decompositions of the unitary $V(f)$ involve $\Omega(2^n)$ qubits and $\Omega(\sqrt{2^n})$ non-Clifford gates, and face serious questions of practicality at large-scales. To our knowledge, the survey of QRAM in Ref. [8] was the first to consider the possibility of achieving a fault-tolerant QRAM by a method other than circuit QRAM. They proved several no-go theorems that present barriers to finding a code where QRAM is transversal. They also proved a no-go theorem ruling out certain distillation–teleportation protocols. Specifically, they considered protocols that have a distillation phase that queries the physical QRAM gate $V(f)$ up to Q times to prepare a resource state $\chi(f)$, followed by a teleportation channel where $\chi(f)$ interacts with an arbitrary state $|\overline{\alpha}\rangle\langle\overline{\alpha}|$ in an attempt to prepare $\overline{V(f)}|\overline{\alpha}\rangle\langle\overline{\alpha}|\overline{V(f)}$. They showed that for this setup, $Q \geq \Omega(2^{2n})$ queries are required to succeed with high fidelity on all possible choices of $|\overline{\alpha}\rangle\langle\overline{\alpha}|$. Translating their logic into our language, they observed that regardless of the protocol and the function f, one can always find an f' which differs from f at only a few addresses, but where the resource states $\chi(f)$ and $\chi(f')$ are $O(\sqrt{Q}/2^n)$-close. This implies that if

$Q \ll 2^{2n}$, then the teleportation channel cannot produce well-distinguishable outputs when f is queried compared to when f' is queried (data processing inequality). Yet, if we suppose that f and f' differ at even one address j while agreeing on some other address k, then when $|\overline{\alpha}\rangle = \frac{1}{\sqrt{2}}(|\overline{j}\rangle + |\overline{k}\rangle)$, the $\overline{V(f)}$ and $\overline{V(f')}$ gates should lead to distinguishable orthogonal states, a contradiction.

Each of the n rounds within our protocol individually fits into the framework of the no-go theorem of Ref. [8]. Our protocol circumvents this result because it adaptively updates the QRAM function being queried in each round, based on measurement outcomes obtained in prior rounds. If two functions f and f' are different, even at a single address, there will be at least one round where the resource states being teleported by our protocol are far away from each other; in fact, if f and f' differ at exactly one address in round $r = 1$, then they will differ at exactly 2^{r-1} addresses in each round $r = 2, 3, 4, \ldots, n$, provided that all of the n-bit random measurement outcomes obtained up until round r form a linearly independent set.

Certain elements of our protocol also connect with prior work outside the context of QRAM. For example, the task of quantum purity amplification has been extensively studied, and we comment more on this in Section IV-D. Additionally, while the n-qubit states $|\Psi(f)\rangle$ from Eq. (9) have—to the best of our knowledge—not previously been proposed as QRAM resource states,[7] they have been utilized in other contexts, often by the name of "phase states." For example, phase states have been studied as pseudorandom quantum states in the context of cryptography [87], [88], as targets for quantum state tomography [89], and as a mechanism for showing search-to-decision reductions in quantum complexity theory [72].

III. Error models and physical protocol requirements

The theory of FTQC shows how a quantum processor can perform an arbitrary quantum computation through a sequence of noisy physical operations on a set of physical qubits, provided that the physical noise is sufficiently uncorrelated and its rate p is below a constant threshold [90]–[92]. Our protocol augments this by assuming that we also have access to a physical QRAM device that can perform an approximate $V(g)$ gate on n physical qubits, for any function g, as illustrated in Fig. 1b. We require that g can be modified from one query to the next via classical communication with the device. Also, we require that the quantum information contained in the n physical qubits output by the device can be transported into n physical qubits on the main quantum processor without significant degradation of its fidelity, whether by physically moving the qubits output by the device to the main processor, or by some other means.

[7]After the public release of the first version of this paper, Ref. [86] independently proposed using states $|\Psi(f)\rangle$ as resource states for data lookup via teleportation. However, their motivation was quite different, focusing on saving constant factors on the gate complexity of data lookup implementations with $\mathrm{poly}(n)$ footprint and $\Omega(2^n)$ depth. In their context, it is not useful to apply our recursive teleportation scheme, and their work involves only one round of teleportation.

In this section, we specify the noise models we consider for each of these two components, and we define the setup for the FTQC part of our protocol on the main processor. The protocol is agnostic to many of the details here, including which QEC code is used, and we attempt to keep it as general as possible.

A. Error model of the physical QRAM device

Without a more concrete implementation in mind, we cannot fully model the noise in the device. However, we consider a general noise model that assumes only that the noise is independent of g, the function being queried. This noise model was also employed for some of the results in Ref. [8].

Definition 1 (Dataset-independent QRAM noise): A physical QRAM device that implements channel $\widetilde{\mathcal{V}}(g)$ on input g is said to have dataset-independent noise if there exists $\mathcal{N}_1$ and $\mathcal{N}_2$ independent of g for which

$$\widetilde{\mathcal{V}}(g) = \mathcal{N}_2 \circ \mathcal{V}(g) \circ \mathcal{N}_1 \,, \tag{13}$$

where $\mathcal{V}(g) = V(g)[\cdot]V(g)^\dagger$ is the ideal unitary channel.

We defend the plausibility of this noise model by appealing to the presumed structure of shallow-depth physical QRAM implementations. One imagines that the bits of classical memory corresponding to the values $g(0)$, $g(1)$,..., $g(2^n-1)$ are distributed in memory cells over 1D or 2D space—for example, the illustration in Fig. 1a distributes them in 1D space. As discussed in Ref. [8], at a high level, a $\mathrm{poly}(n)$-depth QRAM implementation requires a routing step, a readout step, and an unrouting step. In the routing step, the n-qubit address information $|x\rangle$ is used to (coherently) activate a path to the memory cell corresponding to address x. In the readout step, a qubit must interact with the classical bit of information $g(x)$ stored in that cell, gaining a -1 phase if and only if $g(x) = 1$. Then, in the unrouting step, the activated routers must be coherently reset before being traced out to ensure the overall operation maintains coherence between different $|x\rangle$.

The important fact to notice is that the routing and unrouting steps are not at all dependent on the dataset g. Thus, to justify the validity of Definition 1 in this model, any noise that occurs during the routing step could be propagated backward to the beginning of the circuit and contribute to $\mathcal{N}_1$, while any noise in the unrouting step could be propagated forward to the end of the circuit and contribute to $\mathcal{N}_2$.

The only step that can be dependent on g is the readout step, but this step is generally considered to be simpler to implement than the routing step [8]. For example, in the bucket-brigade QRAM circuit of Ref. [37, Figure 10], the readout step is performed in a single circuit layer by a set of parallel single-qubit Pauli-X gates: at memory cell x, an X gate is applied if $g(x) = 1$, and an identity gate

$\mathbb{I}$ is applied if $g(x) = 0$. (This classically controlled X gate would ideally be applied passively via interaction with a non-volatile memory storing $g(x)$.) Since the identity $\mathbb{I}$ and Pauli-X gates are simple, it may be plausible that, in some implementations, the noise can be independent of which of them is applied, justifying Definition 1 for the physical QRAM device. Furthermore, we note that even if the noise is not identical for the $\mathbb{I}$ and X gates, it is plausible that the dataset-independent noise property *effectively* holds in the context of our protocol, thanks to the protocol's *partial Clifford twirling* (Section IV-C) that effectively randomizes the data being queried—intuitively this randomization should remove dependence of g from the average noise channel.

We leave to future work the task of more rigorously showing that certain microscopic (i.e., component-level) noise models lead to dataset-independent noise in the form of Definition 1. However, we note that the set of noise processes that fall into this category can include some counterintuitive members. For example, we may suppose that a physical QRAM device is constructed from a depth-n binary tree of routing elements, but that one of the routers in the tree is "dead." The QRAM attempts to activate a path through the tree to a particular address x at one of the leaves, and if this path passes through the dead router, it causes catastrophic failure of the device, leading the device to instead output the maximally mixed state $\mathbb{I}/2^n$. Let $\mathcal{X} \subset \{0,1\}^n$ denote the set of addresses that cause the dead router to activate when they are queried. Let $\Pi_{\mathcal{X}} = \sum_{x\in\mathcal{X}} |x\rangle\langle x|$ be the projector onto these address states, and let $\Pi_{\overline{\mathcal{X}}} = \mathbb{I} - \Pi_{\mathcal{X}}$. Then, we may write the channel implemented by the noisy device as

$$\widetilde{\mathcal{V}}(g)[\rho] = V(g)\Pi_{\overline{\mathcal{X}}}\rho\Pi_{\overline{\mathcal{X}}}V(g)^\dagger + \mathrm{tr}(\Pi_{\mathcal{X}}\rho)\frac{\mathbb{I}}{2^n} \tag{14}$$

$$= \Pi_{\overline{\mathcal{X}}}V(g)\rho V(g)^\dagger\Pi_{\overline{\mathcal{X}}} + \mathrm{tr}(\Pi_{\mathcal{X}}\rho)\frac{\mathbb{I}}{2^n}\,, \tag{15}$$

where the second equality follows since $V(g)$ is a diagonal unitary, and thus it commutes with the diagonal projector $\Pi_{\overline{\mathcal{X}}}$. We may then rewrite the noisy channel $\widetilde{\mathcal{V}}(g)$ in a dataset-independent fashion as $\widetilde{\mathcal{V}}(g) = \mathcal{N}_2 \circ \mathcal{V}(g) \circ \mathcal{N}_1$, with $\mathcal{N}_1 = \mathcal{I}$ (the identity channel) and

$$\mathcal{N}_2[\rho] = \Pi_{\overline{\mathcal{X}}}\rho\Pi_{\overline{\mathcal{X}}} + \mathrm{tr}(\Pi_{\mathcal{X}}\rho)\frac{\mathbb{I}}{2^n}\,. \tag{16}$$

Furthermore, in the context of our protocol, the device is always queried on the input state $\rho = |+\rangle\langle+|^{\otimes n}$. Thus, this particular noise process has fidelity given by

$$\mathrm{tr}\Big(|\Psi(g)\rangle\langle\Psi(g)|\widetilde{\mathcal{V}}(g)[|+\rangle\langle+|^{\otimes n}]\Big)$$
$$= \frac{(2^n - |\mathcal{X}|)(2^n - 1 - |\mathcal{X}|)}{2^{2n}} + \frac{1}{2^n} \geq 1 - \frac{2|\mathcal{X}|}{2^n}\,. \tag{17}$$

That is, if the dead router is deep in the binary tree and $|\mathcal{X}| \ll 2^n$, then the fidelity of the device remains close to 1.

The fact that our protocol can work in such a scenario is counterintuitive because the dead router would seem to completely block access to the information $g(x)$ for any $x \in \mathcal{X}$ and thus make it impossible to correctly implement the QRAM when the address register has high overlap with the support of $\Pi_{\mathcal{X}}$. As we will show in Section IV-C, our protocol handles this with partial Clifford twirling, a technique that scrambles the dataset g into a new dataset g_C so that the information $g(x)$ is contained in $g_C(y)$, and the location y is uniformly random, and in particular, the probability that $y \in \mathcal{X}$ is $|\mathcal{X}|/2^n$. Thus, for every x, the dead router only compromises the information $g(x)$ with small probability, even for $x \in \mathcal{X}$.

B. Error model of the main quantum processor

Our main quantum processor acts on a set of noisy physical qubits with a quantum circuit, that is, a sequence of noisy physical quantum operations including initializations, 1- and 2-qubit gates, and measurements, as well as classical computation and adaptive classical feedback. We will assume that our main processor is subject to circuit-level stochastic noise, defined below.

Definition 2 (Circuit-level stochastic noise, Section 2.5 of Ref. [61]): A physical implementation of a quantum circuit V is said to be subject to parameter-p stochastic noise if the following holds.

- Purely classical components are implemented perfectly without any errors.
- Each quantum component $\mathcal{P}$, including (classically controlled-)gates, qubit initializations, measurements, is realized by $\widetilde{\mathcal{P}} = (1-p)\mathcal{P} + p\mathcal{N}_{\mathcal{P}}$, where $\mathcal{N}_{\mathcal{P}}$ is a quantum channel of the same input and output registers as $\mathcal{P}$.

Circuit-level stochastic noise is a special case of the more standard model called local-stochastic noise model [92], which additionally allows some correlations between the gate faults. We choose circuit-level stochastic noise over local stochastic noise because this allow us to analyze a fault-tolerant logical state preparation procedure using the results of Ref. [61], for which the theorems require independent noise. However, we will require that the QEC code and FTQC scheme are able to correct against the more general local stochastic noise.

C. Fault-tolerant quantum computation

Our protocol is agnostic to which QEC code family and FTQC scheme is utilized, as long as it is capable of a universal set of fault-tolerant logical gates. Specifically, there must be a nonzero threshold p_0 such that, if the processor is subject to local stochastic noise with error parameter $p < p_0$, then for any target error rate and any logical quantum circuit, one can choose the code parameters (e.g., distance) large enough to ensure the ideal logical circuit is simulated up to the required error [90], [92]–[96].

As we wish to enact the logical n-qubit QRAM operation of Eq. (1), we assume we have chosen some QEC code family that encodes n logical qubits into some number $n' > n$ of physical qubits, along with a scheme for performing fault-tolerant gates. Regardless of our choice, we can identify the following ingredients.

- Encoding: There is an encoding isometry E, which maps any n-qubit physical state $|\psi\rangle$ to its associated encoded n-qubit logical state $|\overline{\psi}\rangle$ (of n' physical qubits). The map E is injective and its image is the codespace of the code. We let $\mathcal{E}$ denote the corresponding quantum channel $E[\cdot]E^\dagger$ that encodes density matrices. Our protocol will also require a fault-tolerant encoding gadget $\mathcal{E}_{\rm FT}$, which implements $\mathcal{E}$ in the absence of noise, but is constructed in such a way that its noisy implementation $\widetilde{\mathcal{E}}_{\rm FT}$ is resilient to errors (see Section IV-B).
- QEC: There is a QEC projector $\mathcal{Q}$ that maps states outside of the codespace to states in the codespace, that is, a map that detects and corrects physical errors on encoded states. In FTQC, the projector $\mathcal{Q}$ is implemented with a fault-tolerant QEC gadget, denoted $\mathcal{Q}_{\rm FT}$, which enacts the map $\mathcal{Q}$ in the absence of noise and a map $\widetilde{\mathcal{Q}}_{\rm FT}$ in the presence of noise. The gadget $\mathcal{Q}_{\rm FT}$ is applied after each location in the logical circuit to prevent the buildup and propagation of physical errors; it must satisfy certain formal properties to guarantee the existence of a threshold; see Ref. [93].
- Gates: For each physical gate G, we let $\overline{G}$ denote the associated logical operation on the codespace of the QEC code, and we let $\overline{\mathcal{G}} = \overline{G}[\cdot]\overline{G}^\dagger$ denote the associated map. We let $\overline{\mathcal{G}}_{\rm FT}$ denote a fault-tolerant gate gadget for G. In the absence of noise, the gadget $\overline{\mathcal{G}}_{\rm FT}$ implements $\overline{\mathcal{G}}$ when acting on states in the codespace, and in the presence of noise, it implements a map denoted $\widetilde{\mathcal{G}}_{\rm FT}$, while obeying certain properties related to propagation of physical errors [93]. A scheme for universal FTQC requires the specification of a fault-tolerant gate gadget for a universal set of gates.

We now expand more on the formal properties that FTQC guarantees about these maps. First of all, each subset $S \subset [n']$ of physical qubits is either a correctable or an uncorrectable subset, depending on whether there exists an error channel $(\mathcal{I}_{S^c} \otimes \mathcal{W}_S)$ acting trivially on S^c and nontrivially only on qubits within S that can induce a logical error, in the sense that $\mathcal{Q} \neq \mathcal{Q} \circ (\mathcal{I}_{S^c} \otimes \mathcal{W}_S) \circ \mathcal{Q}$. The assumption that the FTQC scheme has a threshold p_0 against local stochastic noise indicates that whenever $p < p_0$,

$$\sum_{S \text{ uncorrectable}} (1-p)^{n'-|S|} p^{|S|} \leq \Gamma(\mathcal{E})\,, \tag{18}$$

where $\Gamma(\mathcal{E})$ a quantity that depends on p but has the property that, for fixed p, $\Gamma(\mathcal{E})$ can be exponentially driven to zero simply by choosing a larger QEC code (indicated by the encoding map $\mathcal{E}$) from the code family, incurring at most a polylogarithmic overhead (which is defined as the factor by which the number of physical qubits and physical gates must increase).

Furthermore, abstracting away from physical errors, we can similarly quantify the logical error incurred by performing a fault-tolerant gate as

$$\frac{1}{2}\left\| \overline{\mathcal{G}} \circ \mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\rm FT} - \mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\rm FT} \circ \widetilde{\mathcal{G}}_{\rm FT} \circ \widetilde{\mathcal{Q}}_{\rm FT} \right\|_\diamond \leq \Gamma(\mathcal{E})\,, \tag{19}$$

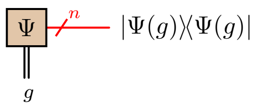
where again $\Gamma(\mathcal{E})$ is used to denote a quantity that vanishes with increasing code size—the overhead of achieving logical error $\Gamma(\mathcal{E}) \leq \delta$ is at worst $\mathrm{polylog}(1/\delta)$. The equation above states that performing the noisy gate $\widetilde{\mathcal{G}}_{\rm FT}$ sandwiched between noisy rounds of QEC and then projecting onto the codespace is nearly equivalent to simply performing noisy QEC, projecting onto the codespace, and applying the perfect logical gate.

IV. Distillation–Teleportation Protocol: Details and Error Analysis

In this section, we examine each step of our protocol in more detail, propagating errors and proving formal statements about the performance of each step. Whereas in previous sections we labeled inputs and outputs of quantum circuits with kets, here we label them with density matrices, since we will be investigating the impact of noise, which may lead states to lose their purity. It should be understood that the unitary circuit elements are applying the associated unitary channel on the density matrices on which they act.

A. Noisy physical resource state preparation with QRAM device

As discussed, we assume we have access to a noisy physical QRAM device that attempts to perform the QRAM operation of Eq. (1) on the state $|+\rangle^{\otimes n}$ at the physical level, in order to produce the resource state $|\Psi(g)\rangle$ of Eq. (9), given as input the dataset g. We depict this relation pictorially via the following circuit, where the red wire indicates that the outgoing quantum register is unencoded and the qubits are physical qubits.

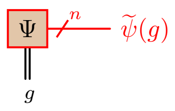


(20)

However, it is unrealistic to assume that the QRAM device can be implemented perfectly at the physical level. Rather, the state it produces is a state $\widetilde{\psi}(g)$ that has some nonzero fidelity with $|\Psi(g)\rangle\langle\Psi(g)|$. We assume that the noise in the device is independent of the dataset, as in Definition 1, so that the QRAM channel decomposes as $\mathcal{N}_2 \circ \mathcal{V}(g) \circ \mathcal{N}_1$. Then, we have

$$\widetilde{\psi}(g) = \mathcal{N}_2\Big[V(g)\mathcal{N}_1\Big[|+\rangle\langle+|^{\otimes n}\Big]V(g)^\dagger\Big] \tag{21}$$

In the circuit notation, we indicate the existence of noise by outlining the gate and its output in red.



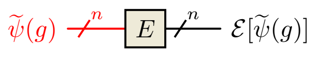
(22)

We characterize the noise strength by the infidelity $1 - F(g)_{\rm phys}$, where $F(g)_{\rm phys}$ is the fidelity of the state $\widetilde{\psi}(g)$ with respect to the ideal state $|\Psi(g)\rangle\langle\Psi(g)|$, given by

$$F(g)_{\rm phys} = \langle\Psi(g)|\widetilde{\psi}(g)|\Psi(g)\rangle\,. \tag{23}$$

B. Encoding physical resource states into logical resource states

The noisy resource state $\widetilde{\psi}(g)$ on n physical qubits is unprotected from errors, and once prepared it must immediately be encoded into a QEC code. Ideally, this encoding process would prepare the state $\mathcal{E}[\widetilde{\psi}(g)]$, which is in the codespace of the code, as in the figure below, where the black wire with a black slash-n indicates that the register contains n logical qubits encoded into some number $n' > n$ of physical qubits.

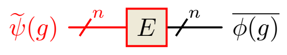


(24)

However, the encoding process may not be fault tolerant, that is, it may introduce additional errors into the logical output state that are proportional to the physical error rate p of the hardware and the number of gates in the encoding circuit that implements $\mathcal{E}$; this logical error cannot be suppressed simply by growing the code size. Hence, we need an encoding procedure $\mathcal{E}_{\mathrm{FT}}$ that is fault-tolerant against the physical noise model, in the sense specified in Proposition 2 below. When we attempt to realize $\mathcal{E}_{\mathrm{FT}}$ on faulty hardware, we instead implement a channel $\widetilde{\mathcal{E}}_{\mathrm{FT}}$. After applying $\widetilde{\mathcal{E}}_{\mathrm{FT}}$, we apply the (noisy) fault-tolerant QEC gadget—implementing the map $\widetilde{\mathcal{Q}}_{\mathrm{FT}}$—on the now-encoded state to correct for physical errors that occurred during $\widetilde{\mathcal{E}}_{\mathrm{FT}}$. The output state is $\widetilde{\mathcal{Q}}_{\mathrm{FT}} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\widetilde{\psi}(g)]$ (we could alternatively think of $\widetilde{\mathcal{Q}}_{\mathrm{FT}}$ as part of $\widetilde{\mathcal{E}}_{\mathrm{FT}}$ and omit it from the expression[8]). This state is still not necessarily in the codespace, but the action of the noisy-but-fault-tolerant $\widetilde{\mathcal{Q}}_{\mathrm{FT}}$ brings it close to the codespace, in the robust sense required by formal proofs of fault tolerance, for example, in Ref. [93]. The closest codespace state is obtained by applying the QEC projector $\mathcal{Q}$, resulting in

$$\overline{\phi(g)} = \mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\mathrm{FT}} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\widetilde{\psi}(g)]\,. \tag{25}$$

Of course, we cannot apply noiseless $\mathcal{Q}$ on actual hardware, but the probability of logical deviation from $\overline{\phi(g)}$ can be suppressed to an arbitrarily small quantity simply by growing the code size (incurring only logarithmic overheads or even constant overhead if one uses constant-rate codes [92]) provided that the physical error rate is below the threshold. Since the rest of our protocol is performed using fault-tolerant logical gates, we identify $\overline{\phi(g)}$ as the logical output of the noisy encoding, denoted pictorially by outlining the encoding gate in red.



(26)

We define the encoding error as the maximum (over arbitrary n-qubit input ρ) trace distance between the state $\mathcal{E}[\rho]$ obtained from perfect encoding and the state $\mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\mathrm{FT}} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\rho]$ obtained from noisy encoding, as follows.

Definition 3 (Encoding error): Consider a QEC code encoding n logical qubits, specified by an encoding isometry $\mathcal{E}$. Let $\mathcal{Q}$ denote a QEC projector for the code $\mathcal{E}$, and let $\widetilde{\mathcal{Q}}_{\mathrm{FT}}$ denote the noisy implementation of a fault-tolerant QEC gadget for $\mathcal{Q}$. Given a fault-tolerant encoding procedure $\mathcal{E}_{\mathrm{FT}}$ and its noisy implementation $\widetilde{\mathcal{E}}_{\mathrm{FT}}$, the encoding error is defined by the following expression

$$\varepsilon_{\mathrm{enc}} = \sup_{\rho} \frac{1}{2} \| \mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\mathrm{FT}} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\rho] - \mathcal{E}[\rho] \|_1\,, \tag{27}$$

where the supremum is taken over all n-qubit physical states ρ. Note that $\varepsilon_{\mathrm{enc}}$ is expected to have a dependence on n, as well as the physical error rate p of the hardware.

The goal of the encoding step is to take a physical state $\widetilde{\psi}(g)$ with some nonzero fidelity $F(g)_{\mathrm{phys}} = \langle \Psi(g) | \widetilde{\psi}(g) | \Psi(g) \rangle$, and map it to an encoded state $\overline{\phi(g)}$ with a smaller but still nonzero fidelity

$$F(g)_{\mathrm{enc}} = \langle \overline{\Psi(g)} | \overline{\phi(g)} | \overline{\Psi(g)} \rangle\,. \tag{28}$$

Using the definition of the encoding error, we can make an additive bound $F(g)_{\mathrm{enc}} \geq F(g)_{\mathrm{phys}} - \varepsilon_{\mathrm{enc}}$, which is sufficient when $F(g)_{\mathrm{phys}}$ is large enough that the right-hand side is greater than zero. However, if $F(g)_{\mathrm{phys}}$ is smaller than $\varepsilon_{\mathrm{enc}}$, then this bound is no longer meaningful. We can instead show a multiplicative bound roughly of the form $F(g)_{\mathrm{enc}} \geq (1 - O(\varepsilon_{\mathrm{enc}}))F(g)_{\mathrm{phys}}$, which is more powerful in the small-fidelity regime. To show this, we have to manually perform a Pauli twirl of the encoding operation, which allows us to guarantee that the twirled encoding channel is stochastic (i.e., it acts as identity with some probability $1 - O(\varepsilon_{\mathrm{enc}})$ and as some other CPTP channel otherwise); see, for example, Ref. [97, Lemma 5.2.4] and Ref. [98, Lemma 3]. Without Pauli twirling, small encoding error alone is not sufficient to rule out the possibility that the encoding channel has coherent error, such as small unitary rotations, which can degrade the fidelity in an additive rather than multiplicative way.

The Pauli twirl involves applying a randomly chosen physical Pauli operator, encoding, and then applying the same Pauli operator but on the logical level. The set of physical Pauli operators can cause the fidelity to degrade by a factor $(1-p)^n$ (which is on the order of $1 - O(\varepsilon_{\mathrm{enc}})$ anyway). We capture this in the following proposition, which states that if we already have an encoding method $\mathcal{E}'_{\mathrm{FT}}$ with small encoding error, we can construct a $\mathcal{E}_{\mathrm{FT}}$ that degrades the fidelity in this multiplicative way. The formal proof is provided in Appendix B of the full version [9].

Proposition 1 (Pauli twirling the encoding channel): Denote the fidelity of the physical state by $F(g)_{\mathrm{phys}} = \langle \Psi(g) | \widetilde{\psi}(g) | \Psi(g) \rangle$. Suppose the processor is subject to circuit-level stochastic noise with strength p (Definition 2), and et $\mathcal{E}'_{\mathrm{FT}}$ be a fault-tolerant encoding channel with encoding error $\varepsilon_{\mathrm{enc}}$ (as in Definition 3). Then, there exists another fault-tolerant encoding channel $\mathcal{E}_{\mathrm{FT}}$ (formed by Pauli-twirling $\mathcal{E}'_{\mathrm{FT}}$), for

[8]However, note that physical errors in $\widetilde{\mathcal{E}}_{\mathrm{FT}}$ can combine with physical errors in $\widetilde{\mathcal{Q}}_{\mathrm{FT}}$ that create a logical difference between $\mathcal{Q} \circ \widetilde{\mathcal{Q}}_{\mathrm{FT}} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\widetilde{\psi}(g)]$ and $\mathcal{Q} \circ \widetilde{\mathcal{E}}_{\mathrm{FT}}[\widetilde{\psi}(g)]$.

which

$$\langle\overline{\Psi(g)}|\overline{\phi(g)}|\overline{\Psi(g)}\rangle$$
$$\geq (1-p)^n\Big((1-3\varepsilon_{\mathrm{enc}})F(g)_{\mathrm{phys}} - 2\Gamma(\mathcal{E})\Big)\,, \tag{29}$$

where $\overline{\phi(g)}$ is defined from $\mathcal{E}_{\mathrm{FT}}$ as in Eq. (25), and $\Gamma(\mathcal{E})$ is a quantity that vanishes with increasing code size, provided the physical error rate p is below a constant threshold, as discussed in Section III-C.

Next, we explain how to achieve an encoding with manageable $\varepsilon_{\mathrm{enc}}$. It should be noted that for fixed p, it is unavoidable for $\varepsilon_{\mathrm{enc}}$ to grow at least linearly with the number of logical qubits n, simply because we start with an unencoded state $|\psi\rangle$ on n faulty physical qubits. Remarkably, it is possible to construct a fault-tolerant encoding procedure $\mathcal{E}_{\mathrm{FT}}$ such that in the presence of noise, the logical encoding error $\varepsilon_{\mathrm{enc}}$ has no direct dependence on the block size and the distance of the code $\mathcal{E}$ that we are encoding into. Such fault-tolerant encoding procedures exist for specific families of quantum codes such as the surface code [50]. Here, to keep our main results as agnostic to the underlying quantum code $\mathcal{E}$ as possible, we opt to use a fault-tolerant encoding procedure that works for any quantum code [61]. This procedure is based on concatenated-code quantum fault tolerance [90], and its fault tolerance for quantum input–quantum output tasks was recently proven under the circuit-level stochastic noise model defined in Definition 2.

The circuit-level stochastic noise model in Definition 2 is a special case of a more general model called local-stochastic noise model [92], which additionally allows for some correlations between gate faults. For our purposes, Proposition 2 below will be using a result of Ref. [61] that is proven under this circuit-level stochastic noise model. However, as is often the case in fault tolerance analysis, we expect the same statement extends to the local-stochastic noise model. Since implementing this extension is beyond the scope of our work, we keep the discussion simple by working with Definition 2 here.

Proposition 2 (Error in fault-tolerant encoding channel for general codes): Consider a family of quantum error-correcting codes encoding n logical qubits into a codespace, and suppose that this family has a threshold p_0 with respect to local stochastic noise (implying Eq. (18)). Correspondingly, for a particular instance of the family (labeled by its encoding map $\mathcal{E}$), let $\mathcal{Q}$ be the ideal QEC projector, $\mathcal{Q}_{\mathrm{FT}}$ be the fault-tolerant QEC gadget, and $\Gamma(\mathcal{E})$ be the logical error suppression function (see Section III-C). Then, there exists a fault-tolerant encoding procedure $\mathcal{E}_{\mathrm{FT}}$ of size $|\mathcal{E}|\cdot\mathrm{poly}(k)$, such that, when implemented under the circuit-level stochastic noise model (Definition 2), the encoding error as defined in Definition 3 satisfies

$$\varepsilon_{\mathrm{enc}} = \sup_\rho \frac{1}{2}\|\mathcal{Q}\circ\widetilde{\mathcal{Q}}_{\mathrm{FT}}\circ\widetilde{\mathcal{E}}_{\mathrm{FT}}[\rho] - \mathcal{E}[\rho]\|_1 \tag{30}$$
$$\leq \Gamma(\mathcal{E}) + 2\sqrt{cp}n + 2|\mathcal{E}|(cp)^k, \tag{31}$$

where C is an absolute constant and k can take values from a sequence of geometrically increasing integers, provided that the physical error rate p is below some constant threshold.

The formal proof is provided in Appendix B of the full version [9]. We remark that the final term in the expression above is similar to the $\Gamma(\mathcal{E})$ term, in the sense that that for any δ, one can choose $k = \mathrm{polylog}(n/\delta)$ and ensure that it is smaller than δ. Thus, the overhead is only polylogarithmic and we neglect its contribution in our analysis elsewhere in the protocol. These two propositions together allow us to show that the encoded state maintains substantial fidelity with the ideal resource state, for use in our protocol.

Corollary 1: Suppose the quantum processor is subject to circuit-level stochastic noise with error rate p (defined in Definition 2). Let

$$F_{\min} = (1 - np - 6n\sqrt{cp})\min_g\langle\Psi(g)|\widetilde{\psi}(g)|\Psi(g)\rangle\,. \tag{32}$$

Then, there exists a fault-tolerant encoding procedure $\mathcal{E}_{\mathrm{FT}}$, such that for all g we have

$$\langle\overline{\Psi(g)}|\overline{\phi(g)}|\overline{\Psi(g)}\rangle \geq F_{\min}, \tag{33}$$

where $\overline{\phi(g)}$ is defined from $\mathcal{E}_{\mathrm{FT}}$ as in Eq. (25). Moreover, the qubit and gate overhead of applying $\mathcal{E}_{\mathrm{FT}}$ is $\mathrm{poly}(n)$.

Proof: This follows by defining $\mathcal{E}'_{\mathrm{FT}}$ to be the encoding procedure shown to exist in Proposition 2, and then applying Proposition 1 to form $\mathcal{E}_{\mathrm{FT}}$. Note that $(1-p)^n > 1 - np + \delta$ for $\delta = O(n^2p^2)$. When the value of k is taken to be $\mathrm{polylog}(n/\delta)$, and the QEC code size is taken to be $n\,\mathrm{polylog}(n/\delta)$, then the error terms $\Gamma(\mathcal{E})$ and $2|\mathcal{E}|(cp)^k$ can be made $O(\delta)$, and the stated fidelity $F_{\min}$ can be guaranteed. ■

C. Partial Clifford twirling

The preparation and encoding processes produce a state $\overline{\phi(g)}$, but the distillation process discussed later can only distill $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$ from many copies of $\overline{\phi(g)}$ if the principal eigenvector of $\overline{\phi(g)}$ is $|\overline{\Psi(g)}\rangle$, which we cannot guarantee in general from the noise model we have assumed.

One solution to this challenge is to use twirling [99], also known as randomized compiling. This technique can convert general noise into better-behaved noise by inserting random gates from a certain set into a quantum circuit, and compensating for their effect by modifying other gates in the circuit. We already saw an example of this process in the encoding step (Proposition 1), where the insertion of Pauli gates led the noise to become stochastic. It is well known that twirling by random Clifford gates can lead to stronger results, converting arbitrary (gate-independent) noise into depolarizing noise. For example, Ref. [100] showed how randomized compiling, when actively applied within the physical QRAM device, can help mitigate the impacts of coherent errors on fidelity. However, in our setting—where we ideally consider a fully passive QRAM device and thus can only apply twirling "outside" the device—we cannot use the full Clifford group since, for example, if we conjugate the QRAM unitary $V(g)$ by the H gate, we do not obtain another QRAM unitary. Our twirling set will instead be the subset of the n-qubit Clifford gates generated by Z, X, CZ and CX (CNOT) gates, which do map QRAM unitaries to QRAM unitaries.

1) Setup and important lemmas: In what follows, we make use of the properties of $\{0,1\}^n$ as an n-dimensional vector space over the field $\mathbb{F}_2$.

Definition 4 (Partial Clifford twirling set): Suppose we are given A, B, u, and v, where

- A is an $n \times n$ invertible matrix over $\mathbb{F}_2$,
- B is an upper triangular $n \times n$ matrix (with zeros on the diagonal) with entries in $\mathbb{F}_2$,
- $u \in \mathbb{F}_2^n$,
- $v \in \mathbb{F}_2^n$.

Define M_A to be the quantum gate that enacts $M_A : |x\rangle \mapsto |Ax\rangle$ for all $x \in \mathbb{F}_2^n$, which can be composed from $O(n^2)$ CX gates via Gaussian elimination, and define the diagonal unitary

$$Q_B = \prod_{1 \le i < j \le n} \mathrm{CZ}_{ij}^{B_{ij}} , \tag{34}$$

where CZ_{ij}^k is the identity gate when $k = 0$ and the CZ gate between qubits i and j when $k = 1$. Then, define the n-qubit quantum gate that corresponds to (A, B, u, v) as

$$C = Z^v Q_B M_A^\dagger X^u, \tag{35}$$

That is, C is a product of $O(n)$ single-qubit Pauli-Z gates determined by the entries of v, $O(n^2)$ CZ gates determined by the entries of B, $O(n)$ Pauli-X gates determined by the entries of X, and $O(n^2)$ CX gates determined by the entries of A (via M_A).

Let the *twirling set* $\mathbb{T}$ consist of all gates C constructed in this fashion from some choice of A, B, u, v. When we say to generate a random gate from $\mathbb{T}$, we mean to generate a uniformly random A, B, u, and v and choose the corresponding $C \in \mathbb{T}$.

We call this partial Clifford twirling because the set $\mathbb{T}$ is a subset of the n-qubit Clifford group. In particular, the set $\mathbb{T}$ is generated by $X, Z, \mathrm{CX}, \mathrm{CZ}$, and the Clifford group is obtained by adding the Hadamard H and phase gate S to the generating set.

Proposition 3: Let $g : \mathbb{F}_2^n \to \mathbb{F}_2$ be a data table (Boolean function) and let $C \in \mathbb{T}$ be a twirling gate corresponding to choice (A, B, u, v). Let M_A and Q_B be defined as in Definition 4. Then, we have

$$|\Psi(g)\rangle = V(g)|+\rangle^{\otimes n} = C\,V(g_C)|+\rangle^{\otimes n} = C|\Psi(g_C)\rangle \tag{36}$$

where for any $x \in \mathbb{F}_2^n$

$$g_C(x) = g(Ax \oplus u) \oplus [x \cdot v] \oplus \left[x^\top Bx\right] \tag{37}$$

Proof: We note that $X^u M_A|+\rangle^{\otimes n} = |+\rangle^{\otimes n}$ since X^u and M_A are made from X and CX gates. The gates X^u and M_A simply permute the computational basis states, viewed as vectors in $\mathbb{F}_2^n$, by an affine transformation, and we can further note that $M_A^\dagger X^u V(g) X^u M_A = V(g')$ where $g'(x) = g(Ax \oplus u)$. Finally, the operation $Z^v Q_B$ is diagonal, and equal to $V(h)$ where $h(x) = [x \cdot v] \oplus \left[x^\top Bx\right]$. We have the composition rule $V(h)V(g') = V(g' \oplus h) = V(g_C)$. The statement follows from these facts as

$$\begin{aligned} C\,V(g)|+\rangle^{\otimes n} &= Z^v Q_B M_A^\dagger X^u V(g)|+\rangle^{\otimes n} && (38)\\ &= Z^v Q_B M_A^\dagger X^u V(g) X^u M_A|+\rangle^{\otimes n} && (39)\\ &= Z^v Q_B V(g')|+\rangle^{\otimes n} && (40)\\ &= V(h)V(g')|+\rangle^{\otimes n} && (41)\\ &= V(g_C)|+\rangle^{\otimes n} && (42)\end{aligned}$$

■

Next, we will examine how random choice of $C \in \mathbb{T}$ spreads Pauli operators. We will examine the set of signed Pauli operators, and specific subsets of it.

Definition 5 (Signed Pauli set and noteworthy subsets): Define $\mathbb{P}$ to be the set of 2^{2n+1} signed Pauli strings written in the canonical form $\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a$, where $s \in \mathbb{F}_2$, $a, b \in \mathbb{F}_2^n$. The factor of i ensures that each operator in $\mathbb{P}$ is Hermitian. We partition $\mathbb{P}$ into several nonoverlapping subsets

$$\mathbb{P}_0 = \{\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a \in \mathbb{P} : a = b = 0^n, s = 0\} = \{\mathbb{I}\} \tag{43}$$

$$\mathbb{P}_1 = \{\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a \in \mathbb{P} : a = b = 0^n, s = 1\} = \{-\mathbb{I}\} \tag{44}$$

$$\mathbb{P}_Z = \{\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a \in \mathbb{P} : a \neq 0^n, b = 0^n\} \tag{45}$$

$$\mathbb{P}_{\mathrm{even}} = \{\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a \in \mathbb{P} : b \neq 0^n, a \cdot b = 0\} \tag{46}$$

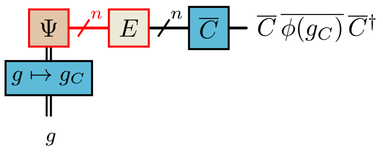
$$\mathbb{P}_{\mathrm{odd}} = \{\mathrm{i}^{a \cdot b}(-1)^s X^b Z^a \in \mathbb{P} : a \cdot b = 1\} \tag{47}$$

We argue that the Pauli operators are spread uniformly over the subset of $\mathbb{P}$ to which it belongs. The full proof of the following proposition is provided in Appendix C.1 of the full version [9].

Proposition 4 (Twirling spreads Paulis uniformly): Let $C \sim \mathbb{T}$ denote choosing C randomly from $\mathbb{T}$ as described in Definition 4. Given a fixed $P \in \mathbb{P}$, for any $C \in \mathbb{T}$, $CPC^\dagger \in \mathbb{P}$ since C is Clifford. Furthermore, let $Q \in \mathbb{P}$ be a random variable formed by choosing $C \sim \mathbb{T}$ and defining $Q = CPC^\dagger$. Then, the distribution over Q is the uniform distribution over the subset of $\mathbb{P}$ (i.e., $\mathbb{P}_0$, $\mathbb{P}_1$, $\mathbb{P}_Z$, $\mathbb{P}_{\mathrm{even}}$, or $\mathbb{P}_{\mathrm{odd}}$) to which P belongs.

Proof idea: Consider a Pauli $P \in \mathbb{P}$ written in canonical form as $P = \mathrm{i}^{a \cdot b}(-1)^s X^b Z^a$, and a Clifford $C = Z^v Q_B M_A^\dagger X^u \in \mathbb{T}$. We explicitly compute the Pauli $CPC^\dagger = \mathrm{i}^{a' \cdot b'}(-1)^{s'} X^{b'} Z^{a'}$ and give formulas for s', a', b' in terms of s, a, b, v, B, A, u. Then, we use these formulas to verify that for $t \in \{0, 1, Z, \mathrm{odd}, \mathrm{even}\}$ if $P \in \mathbb{P}_t$ and v, B, A, u are chosen uniformly at random, then s', a', b' are uniformly random over all values consistent with the definition of $\mathbb{P}_t$. ■

2) Modification to circuit: We now explain precisely how our protocol changes when we implement partial Clifford twirling. We want to implement $V(g)$. Each time we query the physical QRAM device, we generate a C uniformly at random from the Clifford subset $\mathbb{T}$, as defined in Definition 4. We update the function g to be g_C using Eq. (37). In particular, the value at each address x may need to be updated, but each one can be computed with a simple $\mathrm{poly}(n)$-time classical computation and a single query to learn $g(y)$ for a particular y. We use the QRAM device to produce the noisy physical resource state $\widetilde{\psi}(g_C)$, and then we encode that resource state, yielding $\overline{\phi(g_C)}$, as in Eq. (25). Only then do we fault tolerantly apply the gate $\overline{C}$, which consists of $O(n^2)$ logical Clifford gates. This full procedure is depicted in the following circuit.



(48)

Each time the QRAM device is called, an independent random C is chosen. Thus, the output state may be modeled as the mixture

$$\overline{\phi(g)}_{\mathrm{twirl}} = \mathop{\mathbb{E}}_{C\sim\mathbb{T}} \overline{C}\,\overline{\phi(g_C)}\,\overline{C}^\dagger \,. \tag{49}$$

We have not included the twirling step in the main circuit of Figure 2; partly because it would clutter the figure, and partly because we feel that the twirling step may not be necessary in practice if the QRAM device can be constructed in such a way that $\overline{\phi(g)}$ already has $|\overline{\Psi(g)}\rangle$ as its principal eigenvector.

3) Twirling ensures the principal eigenvector is correct: Now, we are ready to present the main finding of this section. The full proof is provided in Appendix C.2 of the full version [9].

Proposition 5 (Correct top eigenvector): Suppose that for every g, the state $\overline{\phi(g)}$ defined in Eq. (25) satisfies $\langle\overline{\Psi(g)}|\overline{\phi(g)}|\overline{\Psi(g)}\rangle \geq F_{\min}$, and suppose that the faulty QRAM device is subject to dataset-independent noise (Definition 1). Let $C \sim \mathbb{T}$ denote drawing C randomly from the twirling set as described in Definition 4, and let $\mathbb{E}$ denote expectation value. For each g, let $\overline{\phi(g)}_{\mathrm{twirl}}$ be defined as in Eq. (49). Then $\overline{\phi(g)}_{\mathrm{twirl}}$ satisfies the eigenvalue equation

$$\overline{\phi(g)}_{\mathrm{twirl}}|\overline{\Psi(g)}\rangle = \lambda_{\mathrm{twirl}}|\overline{\Psi(g)}\rangle \,, \tag{50}$$

with $\lambda_{\mathrm{twirl}} \geq F_{\min}$. Furthermore, all other eigenvalues of $\overline{\phi(g)}_{\mathrm{twirl}}$ are no larger than 2^{-n+1}.

Proof idea: Due to the dataset-independent assumption, the noise in the physical QRAM device and the encoding step can be consolidated into a noise matrix $\chi_{P,P'}$ (where $P, P' \in \mathbb{P}$) for which it is always possible to write

$$\overline{\phi(g_C)} = \sum_{P,P'\in\mathbb{P}} \chi_{P,P'}\overline{P}|\overline{\Psi(g_C)}\rangle\langle\overline{\Psi(g_C)}|\overline{P}' \,. \tag{51}$$

Noting $|\overline{\Psi(g_C)}\rangle = \overline{C}^\dagger|\overline{\Psi(g)}\rangle$ (Proposition 3), we can then say

$$\overline{\phi(g)}_{\mathrm{twirl}} = \sum_{P,P'\in\mathbb{P}} \chi_{P,P'} \mathop{\mathbb{E}}_{C\sim\mathbb{T}} \overline{C}\,\overline{P}\,\overline{C}^\dagger|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|\overline{C}\,\overline{P}'\,\overline{C}^\dagger \,. \tag{52}$$

We now use the fact that randomly choosing C leads $CPC^\dagger$ to uniformly cover a large subset of of $\mathbb{P}$ (Proposition 4). The offdiagonal terms with $P \neq \pm P'$ vanish due to the uniformly random sign. This conclusion did not require the full size of $\mathbb{T}$; it is a consequence of the fact that $\mathbb{T}$ contains the Pauli group, and Pauli-twirling leads the effective channel to become a Pauli channel. If $\mathbb{T}$ were the entire Clifford group, then $CPC^\dagger$ would be a uniformly random Pauli, and we would immediately be able to use the 1-design property of the Pauli set to say that, for any g and for any case where $P \neq \pm\mathbb{I}$, the quantity $\mathbb{E}_{C\sim\mathbb{T}}\,\overline{C}\,\overline{P}\,\overline{C}^\dagger|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|\overline{C}\,\overline{P}\,\overline{C}^\dagger$ is equal to the maximally mixed state. This would then immediately imply the statement we seek, and also that all other eigenvalues λ_{other} satsify $\lambda_{\mathrm{other}} \leq 2^{-n}$. Generally, this would be a manifestation of the fact that Clifford twirling transforms any noise channel into a depolarizing channel. However, as $\mathbb{T}$ is not the full Clifford group, we have to do more work; some subsets of $\mathbb{P}$ may be underweighted or overweighted. Nevertheless, we show that there is enough uniformity to recover a similar result, albeit with the bound on λ_{other} suffering a factor-of-2 overhead. ■

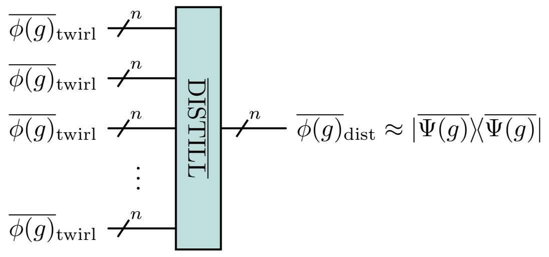
D. Distillation of logical resource states

The encoding step, combined with twirling, produces the logical encoded states $\overline{\phi(g)}_{\mathrm{twirl}}$, which are guaranteed to have $|\overline{\Psi(g)}\rangle$ as their principal eigenvector (Proposition 5). The remaining steps of the protocol act directly on the encoded states (using fault-tolerant gadgets for a universal set of gates): all physical errors created during the distillation and teleportation portions of the protocol can be prevented from turning into logical errors by growing the code distance. Thus, we describe the distillation and teleportation protocol by their logical quantum circuits, and in the description of our protocol, we keep the overline notation to remind the reader that these operations are meant to be performed fault tolerantly on the encoded Hilbert space.

However, the distillation ideas presented here apply generally, regardless of whether/how the states and operations are encoded with QEC. Later in the section, including some of the proposition statements, we drop the overlines, since the statements could be of independent interest.

The procedures we consider for distilling the logical QRAM resource state are state agnostic. Namely, given many copies of an arbitrary state $\overline{\rho}_{\mathrm{in}}$, the distillation protocol prepares (up to small trace distance error) the pure state $|\overline{\Xi}\rangle\langle\overline{\Xi}|$, where $|\overline{\Xi}\rangle$ is the principal eigenvector of $\overline{\rho}_{\mathrm{in}}$. In other words, our distillation procedure is equivalent to the task of *quantum purity amplification* [70].

Pictorially, the distillation step accomplishes the following operation within our protocol. For simplicity, the circuit below depicts all copies being prepared at the beginning and processed at once; in practice, it is possible to prepare the states in a streaming fashion with less space requirement, as we discuss later.



$\overline{\phi(g)}_{\rm twirl}$
n
DISTILL
$\overline{\phi(g)}_{\rm dist} \approx |\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$

(53)

Now we present the main result of this section, as applied to our protocol, utilizing the general techniques discussed later in the section.

Proposition 6: Suppose that the QRAM device is subject to dataset-independent noise, as in Definition 1, and that for every g, the states $\overline{\phi(g)}$ produced on input g (see Eq. (25)) satisfy $\langle\overline{\Psi(g)}|\overline{\phi(g)}|\overline{\Psi(g)}\rangle \geq F_{\rm min}$, with $F_{\rm min} \geq 2^{-n+2}$. Then, for any error parameter $\varepsilon_{\rm dist}$, by applying a carefully crafted sequence of subsequent (fractional) swap operations on $O(\frac{1-F_{\rm min}}{F_{\rm min}^2}(\frac{1}{\varepsilon_{\rm dist}} + \frac{1}{F_{\rm min}}))$ copies of $\overline{\phi(g)}_{\rm twirl}$ (from Eq. (49)) we can distill a state $\overline{\phi(g)}_{\rm dist}$ that satisfies

$$\frac{1}{2}\||\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\rm dist}\|_1 \leq \varepsilon_{\rm dist}\,. \tag{54}$$

The protocol requires $O(1)$ single-qubit gates and $O(n)$ controlled swap operations, per copy consumed.

Proof: This is based on Proposition 5, which shows that $\overline{\phi(g)}_{\rm twirl}$ has the correct top eigenvector, combined with some state-agnostic quantum purity amplification protocol to distill the top eigenvector. When $F_{\rm min}$ is close to 1 the iterated swap test achieves this with a close-to-optimal $\approx (1-F_{\rm min})/\varepsilon_{\rm dist}$ number of copies of $\overline{\phi(g)}_{\rm twirl}$ and the same order of swap tests, that is, circuit (56)—see Proposition 7 and Lemma 1. However, for smaller values of $F_{\rm min}$, the iterated swap test incurs an $\exp(\Theta(1/F_{\rm min}))$ overhead, see the discussion at the end of Section IV-D1.

In the general case, we can exploit the fact that the second largest eigenvalue of $\overline{\phi(g)}_{\rm twirl}$ is upper bounded by $2^{-n+1} \leq F_{\rm min}/2$ due to Proposition 5, and use a protocol based on quantum principal component analysis (Proposition 10), achieving the stated copy complexity utilizing a matching number of swap-test-like gadgets from Fig. 4.

Both Proposition 10 and Proposition 7 are described as producing a state for which the largest eigenvalue is close to 1. We can turn this into a trace distance bound via Lemma 1. The stated gate complexity follows from the observation that Fig. 4 has 3 controlled swap operations (targetting $(n+1)$-qubit registers) and 4 single-qubit gates. ∎

To convert to trace distance as in Eq. (54), our analysis uses the fact that a mixed state is as close to its principal eigenvector as its principal eigenvalue is to 1, captured in the following lemma.

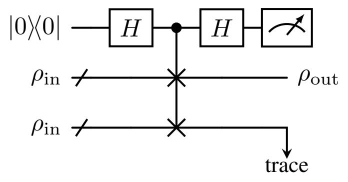
Lemma 1: A quantum state $\overline{\rho}_{\rm out}$, whose principal eigenvector is $|\Xi\rangle$ with eigenvalue $1-\eta$, satisfies

$$\frac{1}{2}\|\overline{\rho}_{\rm out} - |\Xi\rangle\langle\Xi|\|_1 = \eta. \tag{55}$$

Proof: Let $\lambda_1, \ldots, \lambda_d$ be the eigenvalues of the state $\overline{\rho}_{\rm out}$, with $\lambda_1 = 1-\eta$ the principal eigenvector. They satisfy $\sum_{j=1}^d \lambda_j = 1$, so $\sum_{j=2}^d \lambda_j = \eta$. The operator $\overline{\rho}_{\rm out} - |\Xi\rangle\langle\Xi|$ has the same eigenvectors as $\overline{\rho}_{\rm out}$, and the eigenvalues are $-\eta, \lambda_2, \ldots, \lambda_d$. The sum of the singular values is thus equal to 2η, which completes the proof. ∎

1) Distillation with the iterated swap test: In this subsection, we drop the overlines in our notation, and speak generally about the task of quantum purity amplification. We first consider the iterated swap test, a quantum purity amplification procedure studied in detail in Ref. [69] (a similar procedure was discussed in Ref. [72]), although there the analysis assumed that the input states $\rho_{\rm in}$ are a mixture of a rank-1 pure state and the maximally mixed state, that is, of the form $\rho_{\rm dep} = (1-\delta)|\Xi\rangle\langle\Xi| + \frac{\delta}{d}\mathbb{I}$, with d the Hilbert space dimension and $\mathbb{I}/d$ the maximally mixed state. We do not make this assumption here. See also Ref. [71] for an analysis of the iterated swap test without this assumption.

The basic ingredient of this distillation procedure is the swap test (which in our application would be performed fault tolerantly using fault-tolerant gadgets for controlled swap, Hadamard, and measurement).



(56)

We say that the swap test passes if the first qubit is measured in $|0\rangle$ at the end of the circuit. Acting on two copies of the trace-1 state $\rho_{\rm in}$, the probability of the swap test passing is given by

$$\Pr[\text{swap test passes}] = \frac{1+\mathrm{Tr}(\rho_{\rm in}^2)}{2}, \tag{57}$$

and the state one obtains conditioned on the swap test passing is

$$\rho_{\rm out} = \frac{\rho_{\rm in} + \rho_{\rm in}^2}{1+\mathrm{Tr}(\rho_{\rm in}^2)}. \tag{58}$$

If the principal eigenvalue of $\rho_{\rm in}$ is $1-\eta_{\rm in}$, then we can bound

$$\Pr[\text{swap test passes}] \geq \frac{1+(1-\eta_{\rm in})^2}{2} \geq 1-\eta_{\rm in}. \tag{59}$$

It is easy to verify that $\rho_{\rm out}$ and $\rho_{\rm in}$ commute, and they also have the same eigensubspaces since $[0,1] \ni x \to x + x^2$ is strictly monotone. Moreover, denoting the principal eigenvalue of ρ_{out} by $1-\eta_{\text{out}}$, we have

$$\begin{aligned}\eta_{\text{out}} &= 1-\frac{(1-\eta_{\text{in}})+(1-\eta_{\text{in}})^2}{1+\operatorname{Tr}(\rho_{\text{in}}^2)}\\ &\le 1-\frac{(1-\eta_{\text{in}})+(1-\eta_{\text{in}})^2}{1+(1-\eta_{\text{in}})^2+\eta_{\text{in}}^2}\\ &= \frac{\eta_{\text{in}}}{2}\left(\frac{1+\eta_{\text{in}}}{1-\eta_{\text{in}}+\eta_{\text{in}}^2}\right), \qquad (60)\end{aligned}$$

which is about $\eta_{\text{in}}/2$ for $\eta_{\text{in}} \ll 1$.

The idea of the distillation protocol studied in Refs. [69], [71], [72] is to iterate the swap test by feeding two copies of ρ_{out} that both passed the swap test back into the swap test as ρ_{in}, and repeating. Each time we successfully pass the swap test, the output state has higher purity and also the probability that the swap test passes at the next level is closer to 1. By continuing this process for sufficiently many iterations and consuming sufficiently many copies of ρ_{in}, we can produce a state with purity arbitrarily close to 1.

For small η_{in} (i.e., the regime of high input fidelity), the number of copies needed scales as $O(\eta_{\text{in}}/\varepsilon_{\text{dist}})$. Circuit (53) depicts creating all copies of the input state at the beginning of the protocol. For the swap test approach, if swap tests on disjoint pairs can be performed in parallel, the overall depth of the protocol could be $O(\log(\eta_{\text{in}}/\varepsilon_{\text{dist}}))$. However, as described in Ref. [69], the swap test approach can also be implemented in a streaming fashion, allowing one to reduce the space requirements to $O(\log(\eta_{\text{in}}/\varepsilon_{\text{dist}}))$ at the expense of requiring $O(\eta_{\text{in}}/\varepsilon_{\text{dist}})$ depth. We now give the formal statement of performance and costs for this type of distillation when $\eta_{\text{in}} < 1/4$, for general input states.

Proposition 7: Consider a qudit in a mixed state ρ_{in} with its principal eigenvector $|\Xi\rangle$ having eigenvalue $1-\eta_{\text{in}}$. After k successful iterations of the swap test procedure we obtain a state ρ_k such that $[\rho_{\text{in}},\rho_k]=0$ and if $\eta_{\text{in}}<1/4$, then $\langle\Xi|\rho_k|\Xi\rangle \ge 1-\frac{2^{-k}\eta_{\text{in}}}{1-4\eta_{\text{in}}}$. The expected number of copies of ρ_{in} consumed is upper bounded by $2^k/\sqrt{1-4\eta_{\text{in}}}$, and the expected number of required individual swap tests is upper bounded by $(2^k-1)/\sqrt{1-4\eta_{\text{in}}}$, moreover the protocol needs to store at most 1 qubit and $k+1$ qudits for preparing ρ_k.

Proof: Let $\rho_0=\rho_{\text{in}}$ and $\eta_0=\eta_{\text{in}}$. The fact that $[\rho_0,\rho_k]=0$ follows directly from Eq. (58). For the stated space-efficient implementation, observe that until ρ_k is prepared it suffices to store at most one copy of each of $\rho_0,\rho_1,\ldots,\rho_{k-1}$, except for a single ρ_i where a swap test is performed, see Ref. [69, Algorithm 3].

Our proof is inspired by the calculations of Ref. [69]. Let us define $\eta_i := 1-\langle\Xi|\rho_i|\Xi\rangle$. We can verify by induction that $\eta_i \le \frac{2^{-i}\eta_0}{1-4\eta_0+2^{2-i}\eta_0}\left(\le\frac{2^{-i}\eta_0}{1-4\eta_0}\right)$. This trivially holds for $i=0$ and the induction step can be verified as follows:

$$\begin{aligned}\eta_{i+1} &\le \frac{\eta_i}{2}\left(\frac{1+\eta_i}{1-\eta_i+\eta_i^2}\right) && \text{(by (60))}\\ &\le \frac{\eta_i}{2-4\eta_i} && \text{(since } (1+\eta_i)(1-2\eta_i)\le 1-\eta_i+\eta_i^2)\\ &\le \frac{2^{-i}\eta_0}{2(1-4\eta_0+2^{2-i}\eta_0)-2^{2-i}\eta_0} && (61)\\ &= \frac{2^{-1-i}\eta_0}{1-4\eta_0+2^{1-i}\eta_0}, && (62)\end{aligned}$$

where the last inequality follows from the monotonicity of $\frac{x}{2-4x}$ and the induction hypothesis.

Let us denote by $p_i^{\text{succ}}=\frac{1+\operatorname{Tr}(\rho_{i-1}^2)}{2}$ the success probability Eq. (57) of the swap test on ρ_{i-1}. The expected number c_i of copies of ρ_0 needed for preparing ρ_i is $c_i = 2^i\prod_{j=1}^{i}\frac{1}{p_j^{\text{succ}}}$, which is easy to verify by induction. The $i=0$ case is trivial, and the induction step follows from the observation that to obtain ρ_{i+1} we need to repeat the swap test an expected number of $\frac{1}{p_{i+1}^{\text{succ}}}$ many times on two copies of ρ_i, i.e., $c_{i+1}=2\frac{c_i}{p_{i+1}^{\text{succ}}}$.

We can bound c_i by deriving the bound $\prod_{j=1}^{i}p_j^{\text{succ}} \ge \sqrt{1-4\eta_0}$ as follows

$$\begin{aligned}\left(\prod_{j=1}^{i}p_j^{\text{succ}}\right)^2 &\ge \prod_{j=1}^{i}(1-\eta_{j-1})^2 && \text{(by Eq. (59))}\\ &\ge \prod_{j=1}^{i}(1-2\eta_{j-1})\\ &\ge \prod_{j=1}^{i}\left(1-\frac{2^{2-j}\eta_0}{1-4\eta_0+2^{3-j}\eta_0}\right) && \text{(by Eq. (62))}\\ &= 1-4\eta_0+2^{2-i}\eta_0,\end{aligned}$$

where the last equality follows by induction due to the identity: $(1-\frac{2^{1-i}\eta_0}{1-4\eta_0+2^{2-i}\eta_0})(1-4\eta_0+2^{2-i}\eta_0)=(1-4\eta_0+2^{1-i}\eta_0)$. The expected number s_i of swap tests used for preparing ρ_i can also be bounded by $(2^i-1)\prod_{j=1}^{i}\frac{1}{p_j^{\text{succ}}}$ via induction:

$$\begin{aligned}s_{i+1} &= \frac{1+2s_i}{p_{i+1}^{\text{succ}}}\\ &\le \frac{1+2\cdot(2^i-1)\prod_{j=1}^{i}\frac{1}{p_j^{\text{succ}}}}{p_{i+1}^{\text{succ}}}\\ &= (2^{i+1}-1)\prod_{j=1}^{i+1}\frac{1}{p_j^{\text{succ}}}+\frac{1-\prod_{j=1}^{i}\frac{1}{p_j^{\text{succ}}}}{p_{i+1}^{\text{succ}}}\\ &\le (2^{i+1}-1)/\sqrt{1-4\eta_0}.\end{aligned}$$

■

Proposition 7 only applies when $\eta_{\text{in}} < 1/4$, but the iterated swap test can still be successful even when the input fidelity is lower. We now consider what happens when $\eta_{\text{in}} \gg 0$. Let $\gamma_{\text{in}} := 1-\eta_{\text{in}} = \langle\Xi|\rho_{\text{in}}|\Xi\rangle = \lambda_1(\rho_{\text{in}})$ denote the principal eigenvalue of ρ_{in}, where the notation $\lambda_i(\sigma)$ denotes the i-th largest eigenvalue of σ. Let us assume that $\frac{\lambda_2(\rho_{\text{in}})}{\lambda_1(\rho_{\text{in}})} \leq \alpha$ for some value $\alpha < 1$, then we get the following guarantee on $\gamma_{\text{out}} := \langle \Xi | \rho_{\text{out}} | \Xi \rangle$

$$\begin{aligned}\gamma_{\text{out}} &= \frac{\gamma_{\text{in}} + \gamma_{\text{in}}^2}{1 + \operatorname{Tr}(\rho_{\text{in}}^2)} \\ &\geq \frac{\gamma_{\text{in}} + \gamma_{\text{in}}^2}{1 + \gamma_{\text{in}}^2 + \alpha\gamma_{\text{in}}(1-\gamma_{\text{in}})} \\ &= \frac{\gamma_{\text{in}}(1+\gamma_{\text{in}})}{1 + \gamma_{\text{in}} + (\alpha - 1)\gamma_{\text{in}}(1-\gamma_{\text{in}})} \\ &= \frac{\gamma_{\text{in}}}{1 - (1-\alpha)\gamma_{\text{in}}\frac{1-\gamma_{\text{in}}}{1+\gamma_{\text{in}}}} =: p(\gamma_{\text{in}}) \end{aligned} \tag{63}$$

which is always greater than γ_{in} when $\gamma_{\text{in}} \in (0,1)$. Moreover $\frac{\lambda_2(\rho_{\text{out}})}{\lambda_1(\rho_{\text{out}})} = \frac{\lambda_2(\rho_{\text{in}})}{\lambda_1(\rho_{\text{in}})} \cdot \frac{1+\lambda_2(\rho_{\text{in}})}{1+\lambda_1(\rho_{\text{in}})} \leq \alpha$. Finally, by computing the derivative of Eq. (63) in γ_{in}, we get

$$\frac{1 + \gamma_{\text{in}}(2 - \gamma_{\text{in}} + 2\alpha\gamma_{\text{in}})}{(1 + \gamma_{\text{in}}(\alpha + (1-\alpha)\gamma_{\text{in}}))^2} > 0,$$

which means that $p(\gamma_{\text{in}})$ in Eq. (63) is monotonically increasing in γ_{in}; therefore, if we replace γ_{in} with a lower bound on γ_{in} we still get a valid lower bound on γ_{out}. Let's assume that we have an "easy" scenario, where $\alpha \leq 10^{-3}$; a direct calculation shows that if $\gamma_{\text{in}} \geq 0.2$, then $p(\gamma_{\text{in}}) > 0.23$, $p^{\circ 2}(\gamma_{\text{in}}) = p(p(\gamma_{\text{in}})) > 0.26, \ldots, p^{\circ 9}(\gamma_{\text{in}}) > \frac{5}{6}$. By using Eq. (59), similarly to the proof of Proposition 7 we can see that the expected number of copies for successfully completing the 9 iterations of the swap test is at most

$$2^9 \prod_{j=0}^{8} \frac{2}{1 + (p^{\circ j}(\gamma_{\text{in}}))^2} = 2^{18} \prod_{j=0}^{8} \frac{1}{1 + (p^{\circ j}(\gamma_{\text{in}}))^2} \tag{64}$$

$$< \frac{2^{18}}{22.6} < 11600. \tag{65}$$

Therefore, one can see that in the $\gamma_{\text{in}} = 1 - \eta_{\text{in}} \leq \frac{3}{4}$ regime the iterated swap test still works, but its efficiency degrades rapidly [71, Theorems 15 & 30]. In fact, even if we assume that all non-principal eigenvalues are the same, the protocol incurs an exponential cost in $1/\gamma_{\text{in}}$ because the initial swap tests make only a small increase in γ_{out} while they succeed with probability about $\frac{1}{2}$; see, for example, Ref. [69, Theorem 9]. Nevertheless, when $\gamma_{\text{in}} = 1 - \eta_{\text{in}}$ is lower bounded by a constant we get the desired asymptotically optimal complexity (see Section IV-D2), by first magnifying γ_{in} to at least $\frac{4}{5}$ as in Eq. (63)–(64), and then applying Proposition 7, stated formally as follows.

Proposition 8: In the setting of Proposition 7, suppose that ρ_{in} has principal eigenvalue $1-\eta_{\text{in}}$, where $1-\eta_{\text{in}}$ is greater than $\Omega(1)$. Furthermore, suppose that all other eigenvalues of ρ_{in} are bounded above by $\alpha(1-\eta_{\text{in}})$ for some constant $\alpha < 1$. Then the expected number of copies consumed and the expected number of swap tests is the same as stated in Proposition 7, up to a multiplicative $O(1)$ constant.

Proof: This follows from a generalization of the example above to arbitrary $\alpha < 1$. ■

2) An asymptotically optimal distillation protocol with simultaneous use of all copies: Ref. [70] describes a state-agnostic quantum purity amplification protocol based on the Schur transform, which processes all copies in parallel. The authors prove [70, Theorem II.3] that for a generic quantum state ρ_{in} with principal eigenvector $|\Xi\rangle$, their protocol's sample complexity for outputting a quantum state ρ_{out} such that $\langle \Xi | \rho_{\text{out}} | \Xi \rangle \geq 1 - \varepsilon_{\text{dist}}$ is asymptotically optimal in the $\varepsilon_{\text{dist}} \to 0$ limit[9] and their protocol has sample complexity

$$\underset{\varepsilon_{\text{dist}}\to 0}{\sim} \frac{1}{\varepsilon_{\text{dist}}} \sum_{i=2}^{d} \frac{\lambda_i(\rho_{\text{in}})}{(\lambda_1(\rho_{\text{in}}) - \lambda_i(\rho_{\text{in}}))^2} + O(1). \tag{66}$$

When $\lambda_1(\rho_{\text{in}}) = 1 - \eta_{\text{in}}$, the above expression is maximized by $\lambda_2(\rho_{\text{in}}) = \eta_{\text{in}}$, resulting in complexity

$$\underset{\varepsilon_{\text{dist}}\to 0}{\sim} \frac{1}{\varepsilon_{\text{dist}}} \frac{\eta_{\text{in}}}{(1 - 2\eta_{\text{in}})^2} + O(1),$$

which is rather close to the complexity achieved by Proposition 7.

If we only assume that $\lambda_2(\rho_{\text{in}}) \leq \alpha\lambda_1(\rho_{\text{in}})$, then the expression in Eq. (66) is maximized when all nonzero, non-principal eigenvalues equal $\alpha\lambda_1(\rho_{\text{in}})$, giving rise to the complexity expression

$$\underset{\varepsilon_{\text{dist}}\to 0}{\sim} \frac{1}{\varepsilon_{\text{dist}}} \frac{1 - \gamma_{\text{in}}}{(1-\alpha)^2\gamma_{\text{in}}^2} + O(1). \tag{67}$$

As noted in Ref. [70], this is exponentially better in the $\gamma_{\text{in}} \to 0$ regime (i.e., low input fidelity) than the iterated swap test protocol described in Section IV-D1. However, a major drawback of the corresponding protocol of Ref. [70] is that it requires storing and processing all copies in parallel, resulting in a large space complexity.

The authors of Ref. [71] note that there is no known protocol that can be applied in a streaming fashion but gets close to the complexity of Eq. (67) in the $\gamma_{\text{in}} \ll 1$ regime. In the following subsection, we derive such a protocol that uses only two qudits of memory while matching the above asymptotically optimal sample complexity.

3) Improved distillation in the regime of small input fidelity via quantum PCA: Now we show that a gate-efficient procedure inspired by quantum principal component analysis (PCA) [73], [74] requires only two qudits plus three qubits of storage to output the top eigenstate with fidelity at least $1 - \varepsilon_{\text{dist}}$ using

$$O\left(\left(\frac{1}{\varepsilon_{\text{dist}}} + \frac{1}{\gamma_{\text{in}}}\right) \frac{1 - \gamma_{\text{in}}}{(1-\alpha)^2\gamma_{\text{in}}^2}\right)$$

copies of ρ_{in} in expectation, which matches the optimal asymptotic complexity of Eq. (67) up to a constant factor.

Intuitively speaking, the additional $1/\gamma_{\text{in}}$ term next to $1/\varepsilon_{\text{dist}}$ comes from the fact that we need to find the top

[9]This means that for any fixed spectrum $S = (\lambda_1, \lambda_2, \ldots, \lambda_d)$ the optimal sample complexity is $\frac{1}{\varepsilon_{\text{dist}}} \sum_{i=2}^{d} \frac{\lambda_i}{(\lambda_1 - \lambda_i)^2} + O(1)$ given that ρ_{in} has spectrum S. However, this does not say much about what happens for, say, constant $\varepsilon_{\text{dist}} \approx 1$, see Ref. [70, Appendix D].

eigenstate within the states stored in memory. Since the protocol of Ref. [70] stores and processes all of the required copies in parallel, a tighter analysis might reveal that it performs better in the $\varepsilon_{\mathrm{dist}} \gg \gamma_{\mathrm{in}}$ regime. However, once we pay the $1/\gamma_{\mathrm{in}}$ price of postselection, we can very efficiently distill further, so the overhead does not multiply with the high-precision-induced $1/\varepsilon_{\mathrm{dist}}$ cost. We expect that in the single-pass constant-storage setting, our protocol is essentially optimal, but we leave this problem of optimality as an open question. Finally, we speculate that one may be able to improve this protocol's sample complexity in the following way: if an attempt of locating the top eigenstate failed, one may reuse the earlier copies that were used for density matrix exponentation in earlier rounds.

a) Density matrix exponentiation: Quantum PCA [73], [74] is based on the core primitive of density matrix exponentiation using a fractional swap operation $\exp(-\mathrm{i}\mathbb{S}t) = \cos(t)\mathbb{I} - \mathrm{i}\sin(t)\mathbb{S}$, where $\mathbb{I}$ denotes the identity operation on a two-qudit system, and $\mathbb{S}$ denotes the swap operation of two qudits. Suppose that we have a density operator ς on systems A and S_1, and we get a copy of ϱ on system S_2 matching the dimension of S_1. The Lloyd–Mohseni–Rebentrost (LMR) density matrix exponentiation primitive applies a fractional swap of S_1 and S_2, then discards S_2, and the resulting state can be described as follows [74]:

$$\begin{aligned}
&\mathrm{Tr}_{S_2}\Big[\big(\mathbb{I}_A\otimes\exp(-\mathrm{i}\mathbb{S}t)\big)\left(\varsigma\otimes\varrho\right)\big(\mathbb{I}_A\otimes\exp(\mathrm{i}\mathbb{S}t)\big)\Big] \qquad (68)\\
&= \mathrm{Tr}_{S_2}\big[\ \cos^2(t)\varsigma\otimes\varrho\\
&\qquad + \mathrm{i}\cos(t)\sin(t)(\varsigma\otimes\varrho)(\mathbb{I}_A\otimes\mathbb{S})\\
&\qquad - \mathrm{i}\cos(t)\sin(t)(\mathbb{I}_A\otimes\mathbb{S})(\varsigma\otimes\varrho)\\
&\qquad + \sin^2(t)(\mathbb{I}\otimes\mathbb{S})(\varsigma\otimes\varrho)(\mathbb{I}_A\otimes\mathbb{S})\ \big]\\
&= \ \cos^2(t)\varsigma\\
&\quad + \mathrm{i}\cos(t)\sin(t)\varsigma(\mathbb{I}_A\otimes\varrho)\\
&\quad - \mathrm{i}\cos(t)\sin(t)(\mathbb{I}_A\otimes\varrho)\varsigma\\
&\quad + \sin^2(t)\,\mathrm{Tr}_{S_1}[\varsigma]\otimes\varrho. \qquad \text{(see Fig. 3)}
\end{aligned}$$

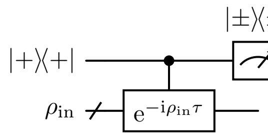
This can be viewed as density matrix exponentiation since it represents the map $\varsigma \mapsto (\mathbb{I}_A\otimes \mathrm{e}^{-\mathrm{i}\varrho t})\varsigma(\mathbb{I}_A\otimes \mathrm{e}^{\mathrm{i}\varrho t})$ up to corrections of order $O(t^2)$. The procedure consumes one copy of ϱ in order to approximately implement the unitary evolution generated by the Hermitian operator ϱ for a short time t. One can also approximately implement controlled density matrix exponentiation $\varsigma \mapsto \big(\mathbb{I}_A\otimes(|0\rangle\langle 0|\otimes\mathbb{I}+|1\rangle\langle 1|\otimes \mathrm{e}^{-\mathrm{i}\varrho' t})\big)\varsigma\big(\mathbb{I}_A\otimes(|0\rangle\langle 0|\otimes\mathbb{I}+|1\rangle\langle 1|\otimes \mathrm{e}^{\mathrm{i}\varrho' t})\big)$ by choosing $\varrho = |1\rangle\langle 1|\otimes\varrho'$; see Ref. [74, Appendix C]. We will exploit this trick in our protocol below.

For an efficient implementation of the fractional swap operation, note that the unitary $(\mathrm{e}^{\mathrm{i}\theta_+ Y}\otimes\mathbb{I})\mathrm{C}\mathbb{S}(\mathrm{e}^{-\mathrm{i}\theta_- X}\otimes\mathbb{I})$ is a block-encoding of the operator $\exp(-\mathrm{i}\mathbb{S}t)/2$, with $\theta_\pm$ defined below and $\mathrm{C}\mathbb{S}$ the controlled swap gate with the first register acting as the control; thus, the fractional swap gate can be implemented using 3 $\mathrm{C}\mathbb{S}$ gates and 4 single-qubit gates

$$\begin{aligned}
&|0\rangle\langle 0|\otimes(\cos(t)\mathbb{I}-\mathrm{i}\sin(t)\mathbb{S}) + |1\rangle\langle 1|\otimes(\mathrm{i}\sin(t)\mathbb{I}+\cos(t)\mathbb{S})\\
&\ = -(\mathrm{e}^{\mathrm{i}\theta_+ Y}\otimes\mathbb{I})\cdot\mathrm{C}\mathbb{S}\,(\mathrm{e}^{-\mathrm{i}\theta_- X}Z\mathrm{e}^{\mathrm{i}\theta_- X}\otimes\mathbb{I})\cdot\\
&\qquad\qquad\qquad \cdot\mathrm{C}\mathbb{S}\,(\mathrm{e}^{-\mathrm{i}\theta_+ Y}Z\mathrm{e}^{\mathrm{i}\theta_+ Y}\otimes\mathbb{I})\cdot\\
&\qquad\qquad\qquad \cdot\mathrm{C}\mathbb{S}\,(\mathrm{e}^{-\mathrm{i}\theta_- X}\otimes\mathbb{I}),\\
&\text{where}\quad \theta_\pm = \frac{\arccos\left(\frac{\cos(t)-\sin(t)}{2}\right)\pm\arccos\left(\frac{\cos(t)+\sin(t)}{2}\right)}{2},
\end{aligned}$$

which corresponds to one iteration of oblivious amplitude amplification. As elsewhere in the paper, X, Y, and Z refer to the single-qubit Pauli operators.

b) A simple (suboptimal) protocol for the case $\lambda_2(\rho_{\mathrm{in}}) \ll \lambda_1(\rho_{\mathrm{in}})\sqrt{\lambda_1(\rho_{\mathrm{in}})\varepsilon_{\mathrm{dist}}}$: We now show how to use a simple version of Kitaev's phase estimation [102] to distill the top eigenstate of ρ_{in} when we are promised that $\lambda_1(\rho_{\mathrm{in}}) \in [\gamma, 3\gamma]$, $\varepsilon_{\mathrm{dist}} \le (1-\gamma)$ and $\lambda_2(\rho_{\mathrm{in}}) \ll \gamma\sqrt{\gamma\varepsilon_{\mathrm{dist}}/(1-\gamma)}$, for a known value of γ. Specifically, the protocol aims to implement the following circuit involving controlled density matrix exponentiation, which may be viewed as phase estimation to one bit of precision.



(69)

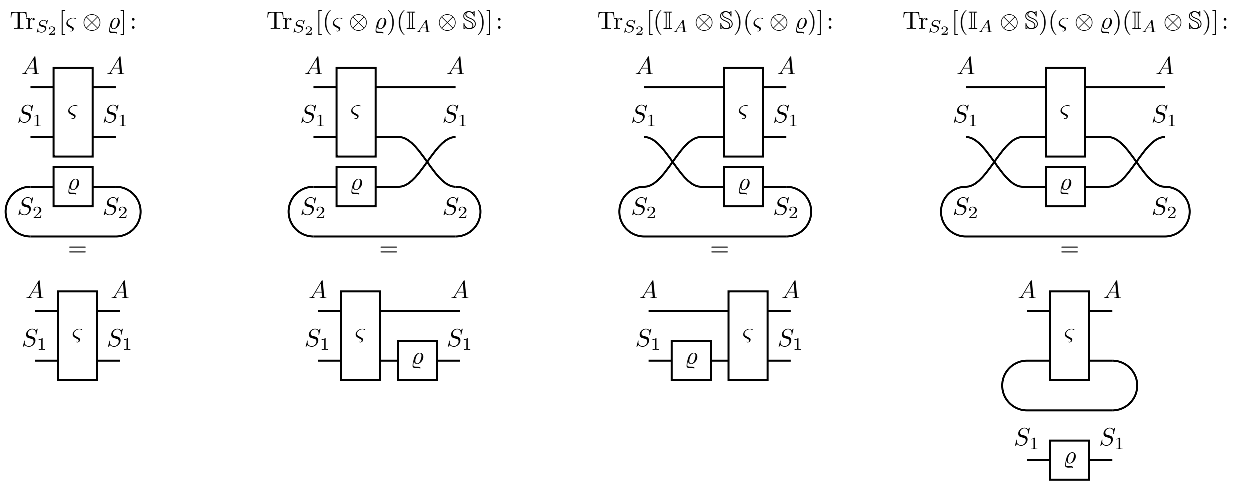
With the right choice of τ, if density matrix exponentiation is performed in small enough steps t, then there is a substantial chance of measuring the first register in $|-\rangle\langle -|$, and when this occurs, the non-principal eigenstates of the input state ρ_{in} are appropriately suppressed in the output.

The controlled density matrix exponentiation in circuit (69) is approximated with $r = \frac{\tau}{t}$ applications of the LMR procedure, giving the circuit in Fig. 4. Namely, consider the effect of the LMR protocol of Eq. (68) in the special case when the system A is not present, and the protocol is repeatedly applied on the initial state $\varsigma = \rho^{(0)} := |+\rangle\langle+|\otimes\rho_{\mathrm{in}}$ on system S_1, using copies of the amended mixed state $\varrho = |1\rangle\langle 1|\otimes\rho_{\mathrm{in}}$ on system S_2. Denote the state (on system S_1) after r iterations of the LMR protocol by $\rho^{(r)}$, which can be written as

$$\rho^{(r)} = \sum_j \sigma_j^{(r)}\otimes\lambda_j|\psi_j\rangle\langle\psi_j|, \qquad (70)$$

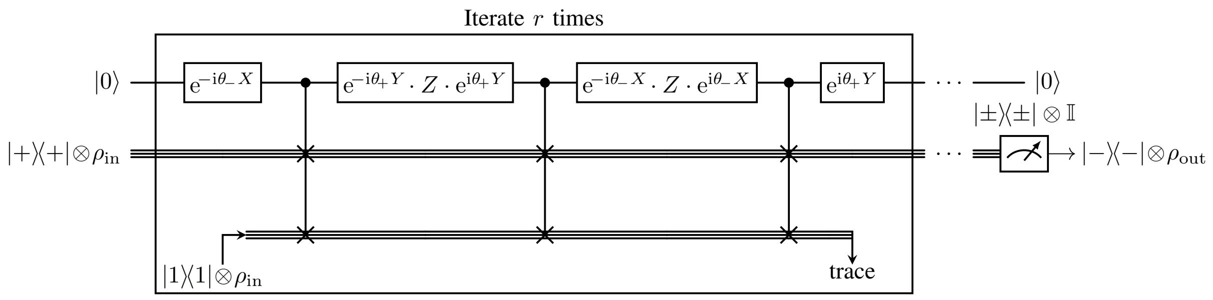
where $\rho_{\mathrm{in}} = \sum_j \lambda_j|\psi_j\rangle\langle\psi_j|$ is the eigendecomposition of ρ_{in} and $\sigma_j^{(r)}$ is a normalized single-qubit density operator, for example, $\sigma_j^{(0)} = |+\rangle\langle+|$. According to Eq. (68), we find that

$$\begin{aligned}
\rho^{(r+1)} = &\ \cos^2(t)\rho^{(r)}\\
&+ \mathrm{i}\cos(t)\sin(t)\rho^{(r)}(|1\rangle\langle 1|\otimes\rho_{\mathrm{in}})\\
&- \mathrm{i}\cos(t)\sin(t)(|1\rangle\langle 1|\otimes\rho_{\mathrm{in}})\rho^{(r)}\\
&+ \sin^2(t)(|1\rangle\langle 1|\otimes\rho_{\mathrm{in}}),
\end{aligned}$$



$\mathrm{Tr}_{S_2}[\varsigma \otimes \varrho]$:
$\mathrm{Tr}_{S_2}[(\varsigma \otimes \varrho)(\mathbb{I}_A \otimes \mathbb{S})]$:
$\mathrm{Tr}_{S_2}[(\mathbb{I}_A \otimes \mathbb{S})(\varsigma \otimes \varrho)]$:
$\mathrm{Tr}_{S_2}[(\mathbb{I}_A \otimes \mathbb{S})(\varsigma \otimes \varrho)(\mathbb{I}_A \otimes \mathbb{S})]$:
A
S_1
S_2
ς
ϱ
=

Fig. 3: Diagrammatic representation [101], and simplification of the four terms in Eq. (68). For a derivation by direct computation consider that $\mathrm{Tr}_{S_2}[(\varsigma \otimes \varrho)(\mathbb{I}_A \otimes \mathbb{S})] = \sum_i (\mathbb{I}_{AS_1} \otimes \langle i|)(\varsigma \otimes \varrho)(\mathbb{I}_A \otimes \mathbb{S})(\mathbb{I}_{AS_1} \otimes |i\rangle)$ which is $\sum_i (\varsigma(\mathbb{I}_A \otimes |i\rangle)) \otimes (\langle i|\varrho) = \sum_i \varsigma(\mathbb{I}_A \otimes |i\rangle\langle i|\varrho) = \varsigma(\mathbb{I}_A \otimes \varrho)$.



Iterate r times
$|0\rangle$
$|+\rangle\langle+|\otimes\rho_{\mathrm{in}}$
$\mathrm{e}^{-\mathrm{i}\theta_- X}$
$\mathrm{e}^{-\mathrm{i}\theta_+ Y} \cdot Z \cdot \mathrm{e}^{\mathrm{i}\theta_+ Y}$
$\mathrm{e}^{-\mathrm{i}\theta_- X} \cdot Z \cdot \mathrm{e}^{\mathrm{i}\theta_- X}$
$\mathrm{e}^{\mathrm{i}\theta_+ Y}$
$|0\rangle$
$|\pm\rangle\langle\pm| \otimes \mathbb{I}$
$|-\rangle\langle-|\otimes\rho_{\mathrm{out}}$
$|1\rangle\langle 1|\otimes\rho_{\mathrm{in}}$
trace

Fig. 4: A simple procedure based on density matrix exponentiation for extracting the top eigenstate of an unknown density operator ρ_{in}. The procedure approximately implements one step of Kitaev's phase estimation of circuit (69). The parameters $\theta_\pm = \frac{1}{2}\arccos\left(\frac{\cos(t)-\sin(t)}{2}\right) \pm \frac{1}{2}\arccos\left(\frac{\cos(t)+\sin(t)}{2}\right)$ determine the length t of the approximated density matrix evolution-time segment per iteration. Each iteration consumes a fresh copy of ρ_{in} and the first ancilla qubit returns to state $|0\rangle$ after each iteration. After all iterations are completed, the second ancilla qubit is measured in the $|\pm\rangle$ basis and we only accept the $|-\rangle$ outcome.

in particular $\rho^{(r+1)} = \sum_j \sigma_j^{(r+1)} \otimes \lambda_j |\psi_j\rangle\langle\psi_j|$ where

$$\begin{aligned}\sigma_j^{(r+1)} = \; & \cos^2(t)\sigma_j^{(r)} \\ & + \mathrm{i}\cos(t)\sin(t)\sigma_j^{(r)}(\lambda_j|1\rangle\langle 1|) \\ & - \mathrm{i}\cos(t)\sin(t)(\lambda_j|1\rangle\langle 1|)\sigma_j^{(r)} \\ & + \sin^2(t)\,\mathrm{Tr}\left[\sigma_j^{(r)}\right]|1\rangle\langle 1|. \qquad (71)\end{aligned}$$

This means that $\sigma_j^{(r+1)} = \Phi_j[\sigma_j^{(r)}]$ for a quantum channel Φ_j, which we analyze in the full version [9] in detail and prove the following.

Proposition 9: Suppose we are given real numbers $\gamma, \varepsilon_{\mathrm{dist}} \in (0,1)$, and copies of a qudit density operator ρ_{in} such that $\lambda_1(\rho_{\mathrm{in}}) \in [\gamma, 3\gamma]$, $\varepsilon_{\mathrm{dist}} \leq 1-\gamma$ and $\lambda_2(\rho_{\mathrm{in}}) \leq \gamma\sqrt{\frac{8\gamma\varepsilon_{\mathrm{dist}}}{3\pi^2(1-\gamma)}}$. Choosing $r = \left\lceil \frac{3\pi^2(1-\gamma)}{2\gamma^3\varepsilon_{\mathrm{dist}}} \right\rceil$ and $t = \frac{\pi}{2r\gamma}$ the protocol of Fig. 4 succeeds with probability at least $\frac{\gamma}{3}$, and upon success produces a state $\rho_-^{(r)}$ such that $[\rho_{\mathrm{in}}, \rho_-^{(r)}] = 0$, and $\langle\Xi|\rho_-^{(r)}|\Xi\rangle \geq 1-\varepsilon_{\mathrm{dist}}$, where $|\Xi\rangle$ is the principal eigenvector of ρ_{in}. The protocol uses an expected number of copies of ρ_{in} which is at most $\frac{3}{\gamma}(r+1)$, and the same order of controlled qudit swap and single-qubit gates. The space complexity is two qudits and three qubits of storage.

Note that if we only know that $\lambda_1 \geq \gamma$ and $\lambda_2 \leq \gamma\sqrt{\frac{8\gamma\varepsilon_{\mathrm{dist}}}{3\pi^2(1-\gamma)}}$, but don't know whether $\lambda_1 \leq 3\gamma$, we can still perform the distillation using the above protocol through combining it with standard techniques, such as exponential search to guess the right order of $\frac{\lambda_1}{\gamma}$. The resulting protocol shall even have the same asymptotic complexity up to constant factors.

c) An improved protocol for the general case: We can improve the overhead in the previous protocol by recursively filtering the smaller eigenvalues in a similar fashion to Ref. [103]. As we show, it suffices to filter a constant fraction of the unwanted eigenstates in each iteration. This relieves the burden of error magnification due to the postselection on a small probability $\approx \lambda_1$ event hindering the previous simple variant described in the proof of Proposition 9. For this purpose, we can use Kitaev's phase estimation [102] combined by standard error reduction techniques, which requires a total simulation time of $\Theta(\frac{\log(1/\epsilon)}{\delta})$ in controlled density matrix exponentiation to output, with probability at least $1-\epsilon$, a phase estimate that is less than δ off [104]. For practical considerations one might also consider using improved iterative phase estimation variants [105] or eigenstate filtering techniques via quantum singular value transformation or quantum signal processing [106], [107].

The controlled Hamiltonian simulation can be performed using the same circuit as before (Fig. 4), however the subsequent applications of the protocol require a slightly adapted analysis because we need to track the state of multiple qubits and/or the entire measurement history.

Proposition 10: Suppose we are given real numbers $\gamma, \varepsilon_{\text{dist}}, \alpha \in (0,1)$, and can request copies of a qudit density operator ρ_{in} such that $\lambda_1(\rho_{\text{in}}) \geq \gamma$, $\varepsilon_{\text{dist}} < (1-\gamma)$ and $\lambda_2(\rho_{\text{in}}) \leq \alpha\gamma$. By iteratively applying the circuit of Fig. 4 with appropriate choices of r and t we can prepare a state ρ_{out} such that $[\rho_{\text{in}}, \rho_{\text{out}}] = 0$, and $\langle \Xi|\rho_{\text{out}}|\Xi\rangle \geq 1 - \varepsilon_{\text{dist}}$, where $|\Xi\rangle$ is the principal eigenvector of ρ_{in}. The protocol uses an expected number of copies of ρ_{in} which is at most $O\left(\frac{1-\gamma}{(1-\alpha)^2\gamma^2}\left(\frac{1}{\varepsilon_{\text{dist}}} + \frac{1}{\gamma}\right)\right)$, and the same order of controlled qudit swap gates and single-qubit gates. The space complexity is two qudits and three qubits of storage.

The core of our analysis is to understand what happens when we apply the LMR controlled density matrix exponentiation protocol to a mixed state whose reduced density matrix commutes with $\rho_{\text{in}} = \sum_j \lambda_j |\psi_j\rangle\langle\psi_j|$, where $|\Xi\rangle = |\psi_1\rangle$.

Lemma 2: Let A label a system of arbitrary finite dimension, and let C label a two-dimensional (qubit) system. Suppose that we have a quantum algorithm that receives as input a normalized state $\rho^{(\text{start})} := \sum_j \sigma_j^{(\text{start})} \otimes |\psi_j\rangle\langle\psi_j|$, where $\sigma_j^{(\text{start})}$ are subnormalized quantum states on the AC register and $|\psi_j\rangle\langle\psi_j|$ are the eigenstates of ρ_{in}. If the algorithm only interacts with the final register through controlled Hamiltonian simulation $\mathbb{I}_A \otimes |0\rangle\langle 0|_C \otimes \mathbb{I} + \mathbb{I}_A \otimes |1\rangle\langle 1|_C \otimes \mathrm{e}^{-\mathrm{i}\rho_{\text{in}}\tau_k}$, then the output state can be written as $\rho^{(\text{end})} := \sum_j \sigma_j^{(\text{end})} \otimes |\psi_j\rangle\langle\psi_j|$. Moreover, if the total simulation time is $T = \sum_k \tau_k$ and we approximate each such controlled Hamiltonian simulation step by the LMR protocol with step size at most $t \leq \frac{\pi}{2}$, then the output state can be written as $\tilde{\rho}^{(\text{end})} := \sum_j \tilde{\sigma}_j^{(\text{end})} \otimes |\psi_j\rangle\langle\psi_j|$, where it holds that $\|\tilde{\sigma}_j^{(\text{end})} - \sigma_j^{(\text{end})}\|_1 \leq 3Tt \max(\mathrm{Tr}\left[\sigma_j^{(\text{start})}\right], \lambda_j)$.

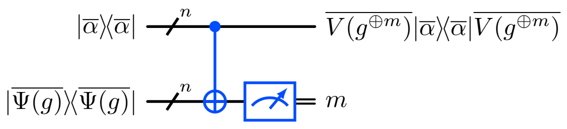
The above lemma is proven in the full version of the paper [9], and then it is used in the detailed analysis of the above propositon, which we omit here for breity.

E. Resource state teleportation

Once we have prepared the resource state $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$ (up to low error), the next step is to consume this resource state to enact a transformation on the encoded address qubits. Suppose the n-qubit encoded register on which we want to apply the operation $\overline{V(g)}$ is in an arbitrary pure state

$$|\overline{\alpha}\rangle = \sum_{w\in\{0,1\}^n} \alpha_w |\overline{w}\rangle\,, \qquad \sum_w |\alpha_w|^2 = 1\,. \tag{72}$$

The teleportation procedure calls for applying a set of n disjoint logical CNOT gates, each controlled by one of the logical qubits in the register holding $|\overline{\alpha}\rangle$ and targeting one of the logical qubits in the resource state, and then measuring the resource state. This was depicted in circuit (10), which is reproduced more compactly as


$|\overline{\alpha}\rangle\langle\overline{\alpha}|$
n
$\overline{V(g^{\oplus m})}|\overline{\alpha}\rangle\langle\overline{\alpha}|\overline{V(g^{\oplus m})}$
$|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$
n
m

(73)

On the output label of the circuit, we have used the notation $g^{\oplus m}$ to represent the Boolean function defined for any x by the equation

$$g^{\oplus m}(x) = g(x \oplus m)\,. \tag{74}$$

To verify the circuit's output and the assertions in Section II-C, note that the CNOT gates enact the transformation

$$\begin{aligned}|\overline{\alpha}\rangle|\overline{\Psi(g)}\rangle &= \frac{1}{2^{n/2}} \sum_{w,z\in\{0,1\}^n} (-1)^{g(z)}\alpha_w|\overline{w}\rangle|\overline{z}\rangle \\ \longmapsto \quad & \frac{1}{2^{n/2}} \sum_{w,z\in\{0,1\}^n} (-1)^{g(z)}\alpha_w|\overline{w}\rangle|\overline{w\oplus z}\rangle \qquad (75) \\ &= \frac{1}{2^{n/2}} \sum_{w,m\in\{0,1\}^n} (-1)^{g(w\oplus m)}\alpha_w|\overline{w}\rangle|\overline{m}\rangle\,. \qquad (76)\end{aligned}$$

A logical measurement is performed on the resource state register, obtaining outcome m. Regardless of the state $|\overline{\alpha}\rangle$ and the function g, the distribution over measurement outcomes m is the uniform distribution, and the post-measurement state on the first register is

$$\begin{aligned}\overline{V(g^{\oplus m})}|\overline{\alpha}\rangle &= \frac{1}{2^{n/2}} \sum_{w\in\{0,1\}^n} (-1)^{g^{\oplus m}(w)}\alpha_w|\overline{w}\rangle \\ &= \frac{1}{2^{n/2}} \sum_{w\in\{0,1\}^n} (-1)^{g(w\oplus m)}\alpha_w|\overline{w}\rangle\,. \qquad (77)\end{aligned}$$

Examining the definition of $g^{\oplus m}$, we can see that it is defined by the same underlying table of data that defines g; however, the addresses to which the data items have been assigned are scrambled by adding (modulo 2) the random measurement outcome m to each address. Nevertheless, we may conclude that the teleportation procedure successfully inserts information about g into the phases of the state $|\overline{\alpha}\rangle$, albeit in an obfuscated way, due to the fact that the phase applied is $(-1)^{g(w\oplus m)}$ rather than the desired $(-1)^{g(w)}$, where m is uniformly random.

We now wish to understand what happens in the teleportation step if we use the state $\overline{\phi(g)}_{\mathrm{dist}}$ (discussed in Section IV-D) as our resource state, which is an imperfect approximation to $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$. First, we define the ideal teleportation channel $\overline{\mathcal{T}(g)}$ as the procedure that perfectly prepares $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$, applies the logical CNOTs, and measures the resource register (equivalently, applies a completely dephasing channel). This has the action

$$\overline{\mathcal{T}(g)}[\rho] = \frac{1}{2^n}\sum_{m\in\{0,1\}^n} \overline{\mathcal{V}(g^{\oplus m})}[\rho]\otimes|m\rangle\langle m|\,, \tag{78}$$

where we have used the channel definition $\mathcal{V}(h)[\rho] = V(h)\,\rho\,V(h)^\dagger$ with the overlined version $\overline{\mathcal{V}(h)}$ denoting the logical version of the channel.

We may now argue that as long as the error of $\overline{\phi(g)}_{\mathrm{dist}}$ is low, relative to $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$, the channel enacted is close to $\overline{\mathcal{T}(g)}$.

Proposition 11 (Teleportation with approximate resource state): Let $\overline{\mathcal{T}(g)}_{\mathrm{appr}}$ be the channel realized by using the state $\overline{\phi(g)}_{\mathrm{dist}}$ in place of $|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|$ in the teleportation protocol depicted in circuit (73). Then, we have

$$\frac{1}{2}\|\overline{\mathcal{T}(g)} - \overline{\mathcal{T}(g)}_{\mathrm{appr}}\|_\diamond \le \frac{1}{2}\||\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\|_1\,, \tag{79}$$

where $\|\cdot\|_\diamond$ indicates the diamond norm distance between channels.

Proof: Define the quantum operation $\overline{\mathcal{C}}$ as the operation on $2n$ logical qubits enacted by the CNOT gates and measurements (i.e., completely dephasing channel) in circuit (73). Thus, for arbitrary n-qubit state $\overline{\sigma}$ in the codespace of the first register, we have

$$\overline{\mathcal{T}(g)}[\overline{\sigma}] = \overline{\mathcal{C}}\left[\overline{\sigma}\otimes|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}|\right] \tag{80}$$

$$\overline{\mathcal{T}(g)}_{\mathrm{appr}}[\overline{\sigma}] = \overline{\mathcal{C}}\left[\overline{\sigma}\otimes\overline{\phi(g)}_{\mathrm{dist}}\right]\,. \tag{81}$$

Now, introduce an environment register of a qubits, and consider an arbitrary state $\overline{\rho}$ on $a+n$ logical qubits (whose reduced density matrix after tracing out the environment is in the codespace of the n-logical-qubit code). Let $\mathcal{I}_{\overline{E}}$ be the identity channel on the environment register. The diamond norm distance is defined as the supremum (over $\overline{\rho}$ for all possible sizes a of the environment) in the trace distance between the action of the two channels

$$\begin{aligned}
&\frac{1}{2}\|\overline{\mathcal{T}(g)} - \overline{\mathcal{T}(g)}_{\mathrm{appr}}\|_\diamond \\
&= \max_a \sup_{\overline{\rho}} \frac{1}{2}\|(\mathcal{I}_{\overline{E}}\otimes\overline{\mathcal{T}(g)})[\overline{\rho}] - (\mathcal{I}_{\overline{E}}\otimes\overline{\mathcal{T}(g)}_{\mathrm{appr}})[\overline{\rho}]\|_1 && (82)\\
&= \max_a \sup_{\overline{\rho}} \frac{1}{2}\left\|(\mathcal{I}_{\overline{E}}\otimes\overline{\mathcal{C}})\left[\overline{\rho}\otimes\left(|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\right)\right]\right\|_1 && (83)\\
&\le \max_a \sup_{\overline{\rho}} \frac{1}{2}\|\overline{\rho}\otimes\left(|\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\right)\|_1 && (84)\\
&= \max_a \sup_{\overline{\rho}} \frac{1}{2}\||\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\|_1 && (85)\\
&= \frac{1}{2}\||\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\|_1\,, && (86)
\end{aligned}$$

where the inequality follows from monotonicity of the trace distance under quantum channels. This completes the proof. ■

F. Adaptive correction and classical update rule

Recall that the goal is to implement the diagonal logical QRAM unitary $\overline{V(f)}$, given a data table (Boolean function) f. The previous section established that for any function g, the teleportation channel $\overline{\mathcal{T}(g)}$ receives a uniformly random measurement outcome m, and then, conditioned on m, enacts the unitary transformation $|\overline{\alpha}\rangle \mapsto \overline{V(g^{\oplus m})}|\overline{\alpha}\rangle$.

Our protocol repeats the teleportation process over a sequence of n rounds. In each round, the function g will be different, and the measurement outcome m will be sampled independently and uniformly at random. The value of g used in round j will be denoted $g^{(j)}$, and the value of m denoted by $m^{(j)}$.

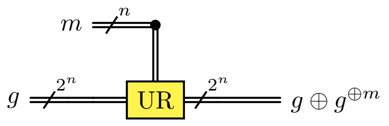
We begin in round 1 setting $g = g^{(1)} = f$, receiving measurement outcome $m = m^{(1)} \in \{0,1\}^n$, and enacting logical unitary $\overline{V(g^{\oplus m})}$. The key observation is that $\overline{V(g^{\oplus m})}^2 = \overline{\mathbb{I}}$, the logical identity operation, and hence

$$\begin{aligned}
\overline{V(g)} &= \overline{V(g)}\;\overline{V(g^{\oplus m})}\;\overline{V(g^{\oplus m})}\\
&= \overline{V(g\oplus g^{\oplus m})}\;\overline{V(g^{\oplus m})}\,,
\end{aligned} \tag{87}$$

where we have invoked the composition rule $V(h)V(h') = V(h\oplus h')$. Thus, given that we have already implemented $\overline{V(g^{\oplus m})}$, we must now implement the *correction unitary* $\overline{V(g\oplus g^{\oplus m})}$, which is also a member of the family of diagonal (logical) QRAM operators. We thus compute a new Boolean function by the *classical update rule* UR, which takes as input a data table g and a bit string $m\in\{0,1\}^n$ and outputs a new data table

$$\mathrm{UR}(g,m) = g\oplus g^{\oplus m}\,, \tag{88}$$

which is depicted as a circuit as



(89)

and we update $g \leftarrow g^{(2)} = \mathrm{UR}(g^{(1)}, m^{(1)})$ to use for round 2. Consequently, the correction unitary is now equal to $\overline{V(g)}$ and the goal of round 2 is again simply to implement the unitary $\overline{V(g)}$, just as it was in round 1. As before, we receive a new measurement outcome $m = m^{(2)}$, we compute $g^{(3)} = \mathrm{UR}(g^{(2)}, m^{(2)})$, and we update $g \leftarrow g^{(3)}$ for round 3.

We then iterate this process a number of times. We note that, if after applying the update rule we ever obtain $g = \mathbf{0}$, the zero function, then we may terminate the procedure, because the correction unitary will be $\overline{V(\mathbf{0})} = \overline{\mathbb{I}}$ at the next round. Moreover, $g = \mathbf{1}$ (the constant function that outputs 1 on all inputs), then the correction unitary is $-\overline{\mathbb{I}}$, which is equivalent to $\overline{\mathbb{I}}$ up to an unphysical global sign. We claim that after at most n rounds, we will be certain to obtain $g \in \{\mathbf{0}, \mathbf{1}\}$, based on the following proposition.

Proposition 12: Let g be an n-bit Boolean function. Define $\deg(g)$ to be the degree of g when it is expanded as a polynomial of its input bits over the field $\mathbb{F}_2$. Suppose that $\deg(g) = d$. Let $m \in \{0,1\}^n$, and let $h = \mathrm{UR}(g, m)$, as defined in Eq. (88). Then, we have

$$\deg(h) \leq d - 1\,. \tag{90}$$

Proof: This is a consequence of the reasoning in Appendix D of the full version [9]. ■

The proposition establishes that each application of the update rule $g \leftarrow \mathrm{UR}(g, m)$ decreases the degree of g by at least 1. Recall that in round 1, we have $\deg(g) = \deg(f) \leq n$, simply by virtue of the fact that f is an n-bit function. Thus, we may assert that in round j we have $g = g^{(j)}$ and

$$\deg(g^{(j)}) \leq n + 1 - j\,. \tag{91}$$

In particular, after applying the update rule in round n with $g = g^{(n)}$ and $m = m^{(n)}$, we are guaranteed to obtain $h = \mathrm{UR}(g, m)$, which leads to a degree $\deg(h) = 0$. If the degree of h is 0, this implies that either $h = \mathbf{0}$ or $h = \mathbf{1}$.

G. Total complexity of protocol

We have now explained the action of each component of the protocol, and may we state its overall complexity.

Theorem 2 (Main result): Let f be an arbitrary dataset (n-bit Boolean function) for which we wish to implement the fault-tolerant diagonal QRAM unitary $\overline{V(f)}$ of Eq. (1), and let ε be an error parameter. Suppose that we have access to a noisy physical QRAM device subject to dataset-independent noise (Definition 1) which on input g produces state $\widetilde{\psi}(g)$ on n physical qubits achieving fidelity $\langle \Psi(g)|\widetilde{\psi}(g)|\Psi(g)\rangle \geq F$ for all g. Suppose further that we have the capability to reload the QRAM device with a new dataset, and that we have the capability to move the n-qubit output state to a fault-tolerant quantum processor subject to circuit-level stochastic noise (Definition 2) with error rate p. If p is below a constant threshold p_0 determined by the QEC code family, and separately if pn^2 is below a different constant threshold related to the fault-tolerant encoding procedure, then there exists an adaptive distillation–teleportation procedure that implements a quantum channel $\mathcal{P}_{\mathrm{DT}}$ for which

$$\frac{1}{2}\|\mathcal{P}_{\mathrm{DT}} - \overline{V(f)}[\cdot]\overline{V(f)}^{\dagger}\|_{\diamond} \leq \varepsilon\,. \tag{92}$$

The protocol uses (in expectation over internal randomness) Q queries to the noisy physical QRAM device, Q applications of the encoding process $\mathcal{E}_{\mathrm{FT}}$ (Corollary 1), and Q' additional fault-tolerant operations (controlled-SWAP, Hadamard, CNOT, single-qubit logical state preparations, and single-qubit logical measurements), where

$$Q = O\left(\frac{n(1-F)}{F^2}\left(\frac{n}{\varepsilon} + \frac{1}{F}\right)\right) \tag{93}$$

$$Q' = O\left(n^2 Q\right)\,. \tag{94}$$

Additionally, the protocol applies the classical update rule (Eq. (88)) at most n times, and the classical partial Clifford twirling operation $g \mapsto g_C$ (Eq. (37)) Q times, each time on a data table of size 2^n classical bits.

Proof: The protocol is depicted as a quantum circuit in Fig. 2. We begin with correctness. In each of the n rounds indexed by $j = 1, \ldots, n$, it implements a channel $\overline{\mathcal{T}(g^{(j)})}$. As discussed in Section IV-F, if all resource states are prepared perfectly, the procedure is guaranteed to implement the unitary $\overline{V(f)}$, up to a global sign which does not impact the channel $\overline{V(f)}[\cdot]\overline{V(f)}^{\dagger}$.

However, the teleportation channel $\overline{\mathcal{T}(g^{(j)})}$ is not implemented perfectly by the protocol. To ensure the overall diamond norm error is ε, it suffices to choose parameters such that $\overline{\mathcal{T}(g)}$ is implemented up to ε/n diamond distance for all g, since the errors from each of the n channel applications add linearly in the worst case, when performed in succession. By Proposition 11, it suffices to distill resource states $\overline{\phi(g)}_{\mathrm{dist}}$ that satisfy

$$\frac{1}{2}\||\overline{\Psi(g)}\rangle\langle\overline{\Psi(g)}| - \overline{\phi(g)}_{\mathrm{dist}}\|_1 \leq \varepsilon_{\mathrm{dist}} \tag{95}$$

with distillation error $\varepsilon_{\mathrm{dist}} = \varepsilon/n$. Meanwhile, by Proposition 6, this is accomplished by the distillation protocol using $O(\frac{1-F_{\min}}{F_{\min}^2}(\frac{1}{\varepsilon_{\mathrm{dist}}} + \frac{1}{F_{\min}}))$ copies of the input states $\overline{\phi(g)}_{\mathrm{twirl}}$, defined in Eq. (49), provided that for all g $\langle\overline{\Psi(g)}|\overline{\phi(g)}|\overline{\Psi(g)}\rangle \geq F_{\min}$ for some $F_{\min}$. We are guaranteed from Corollary 1 that $F_{\min} \geq (1 - O(np) - O(n\sqrt{p}))F$, which can be replaced by $\Omega(F)$ as long as pn^2 is below a certain constant. The number of gates required by distillation is a factor of $O(n)$ larger than the number of copies.

Since there are n rounds, we require n calls to the distillation procedure. Thus, the total number of queries to the noisy QRAM device and the $\mathrm{poly}(n)$-cost encoding procedure

$\mathcal{E}_{\mathrm{FT}}$ is

$$Q = O\left(\frac{n(1-F)}{F^2}\left(\frac{n}{\varepsilon}+\frac{1}{F}\right)\right) \tag{96}$$

The total fault-tolerant gate complexity from distillation is $O(nQ)$. The teleportation procedure also requires n fault-tolerant CNOT gates in each of the n rounds. Finally, the Clifford twirling step requires the application of $O(n^2)$ fault-tolerant Clifford gates for each of the Q copies. In total, these Clifford gates dominate the fault-tolerant gate count, which is $Q' = O(n^2 Q)$. Each round requires only one call to the update rule, and each of the Q copies requires classically applying a partial Clifford update $g \mapsto g_C$. This completes the proof. ■

V. Complexity of the classical update rule

The classical update rule calls for updating a data table g to the data table $h = \mathrm{UR}(g,m) = g \oplus g^{\oplus m}$, for a certain fixed measurement outcome $m \in \{0,1\}^n$, as in Eq. (88). That is, the entry at address x in the data table should be updated from $g(x)$ to $g(x) \oplus g(x \oplus m)$. In this section, we analyze the complexity of this transformation under several different frameworks. The guiding question is to understand the ways in which the classical update rule is a more complex operation than a RAM query.

We note that the partial Clifford twirling step also requires substantial classical computation to randomly transform the dataset. We do not specifically analyze this step here, because we view it as less fundamental to our protocol. For example, if the physical QRAM device and encoding step were error free (or if they have errors but the principal eigenvalue of the resulting state is correct), then partial Clifford twirling is not necessary, but the need for the update rule remains.

A. Classical circuit complexity

The update rule takes $2^n + n$ bits as input and produces 2^n bits as output, as in circuit (89). It is straightforward to see that a classical circuit built from elementary gates (NOT, AND, NAND, etc.) would require $\Omega(2^n)$ gates to implement the update rule. Observe that for any fixed nonzero value $m \neq 0^n$, we have for every x

$$h(x) = h(x \oplus m) = g(x) \oplus g(x \oplus m) \tag{97}$$

That is, the 2^n addresses are partitioned into pairs $\{x, x \oplus m\}$, where h takes the same value on both elements of the pair, and its value is equal to the parity of the input bits at locations x and $x \oplus m$. Each of these 2^{n-1} parity calculations is independent and requires at least one elementary gate.

B. Complexity in a classical RAM model

The need for $\Omega(2^n)$ circuit complexity does not alone entail that the update rule is an expensive classical calculation. After all, the standard RAM operation also has $\Omega(2^n)$ circuit complexity, but it is commonplace to consider a model of classical computation where RAM has unit cost.

However, the single-bit RAM operation takes as input n bits (an address) and returns just 1 bit, so it appears to be a much simpler operation than the classical update rule. Since (i) the output of the update rule has 2^n bits, (ii) the output depends on all 2^n input bits $g(x)$, and (iii) each RAM query can access only 1 of the bits, we conclude that, in a model where the input data g can only be accessed via RAM queries, implementing the update rule requires $\Omega(2^n)$ RAM queries.

C. Classical circuit depth in an all-to-all model

The structured nature of the update rule suggests it may still be amenable to some degree of parallelization. Ideally, one could design a specialized shallow circuit that directly implements the update, rather than relying on RAM. Indeed, if one imposes no restrictions on spatial layout of the $2^n + n$ input bits and 2^n output bits (i.e., one allows all-to-all gates), then one can perform the update rule with a classical circuit of depth $O(n)$ comprised of elementary gates each acting on only $O(1)$ bits.

We provide one possible way to accomplish this. The construction requires $O(n2^n)$ ancilla bits and has the following steps; we provide an $n = 3$ example to assist with understanding each step in Fig. 5 (and in the description of each step, we reference the colors in that figure for clarity of explanation). Let e_i refer to the length-n bit string with a 1 in the i-th position and 0 in the other $n-1$ positions.

1) For each $x \in \{0,1\}^n$ in parallel, the (blue) bit holding $g(x)$ is copied into a (yellow) ancilla bit, accomplished in depth 1.
2) For each $i = 1, \ldots, n$ in parallel, the (green) input bit holding m_i is copied into $2^{n-1} - 1$ (gray) ancilla bits, accomplished in depth $n-1$ (using a tree-like approach, the number of copies of m_i can be doubled with each additional circuit layer).
3) For each $i = 1, \ldots, n$ in series, and for each of the 2^{n-1} address pairs $\{x, x \oplus e_i\}$ in parallel, if $m_i = 1$ then the (yellow) ancilla bit which held the copy of $g(x)$ at the beginning of this step is swapped with the (yellow) ancilla bit which held the copy of $g(x \oplus e_i)$ at the beginning of this step. Each value of i requires depth 1: the availability of 2^{n-1} (green and gray) copies of the m_i bit created in step 2 allows parallelization of the m_i-controlled $\{x, x \oplus e_i\}$ swaps. Thus, the overall depth of this step is n. Over the course of the n steps, the (yellow) ancilla bit that originally held $g(x)$ before this step undergoes the transformations
$$g(x) \overset{i=1}{\mapsto} g(x \oplus m_1 e_1) \overset{i=2}{\mapsto} g(x \oplus m_1 e_1 \oplus m_2 e_2) \overset{i=3}{\mapsto} \cdots$$
$$\cdots \overset{i=n}{\mapsto} g(x \oplus \bigoplus_{i=1}^{n} m_i e_i) = g(x \oplus m)\,, \tag{98}$$
that is, for each x, the (yellow) ancilla bit originally holding $g(x)$ now holds $g(x \oplus m)$.
4) For each $x \in \{0,1\}^n$ in parallel, the (yellow) ancilla bit that originally held the copy of $g(x)$ after step 1 (which now holds $g(x \oplus m)$) is added modulo 2 into the (blue) bit holding $g(x)$, incurring depth 1.

5) The original 2^n (blue) input bits are taken to be the 2^n output bits; the ancilla bits are discarded.

The registers (blue) holding the original bits $g(x)$ have now been updated to $h(x) = g(x) \oplus g(x \oplus m)$, which is the desired output of the update rule.

D. Classical circuit depth in a spatially local model

If the input and output bits of the shallow circuit are embedded into d spatial dimensions, then the $O(n)$-depth circuit above requires spatially nonlocal gates. For example, we may recognize the action of step 3 as a re-arrangement of the 2^n (yellow) ancilla bits by traversing edges of the n-dimensional Boolean hypercube. In $d = n$ spatial dimensions, this could be done using spatially local gates, and each (yellow) classical bit need only interact with $O(n)$ of the 2^n other (yellow) bits (its neighbors on the hypercube). Unfortunately, real classical circuits must be embedded into $d \leq 3$ spatial dimensions. In this case, step 3 cannot be performed exclusively with spatially local gates for more than d of the values of $i \in \{1, \ldots, n\}$.

In fact, we can show that if gates are local in d spatial dimensions, then the depth required is at least $\Omega(2^{n/d})$. This follows from the fact that without knowing the value of m, the circuit must be prepared to connect the bit at address x to all $2^n - 1$ of the other bits; for any pair $x, y \in \{0,1\}^n$, if $m = x \oplus y$, then the circuit must be able to compute the parity of $g(x)$ and $g(y)$. If the bits storing $g(x)$ and $g(y)$ live on opposite sides of the d-dimensional array, computing this parity will require a classical circuit with depth at least $\Omega(2^{n/d})$. Ultimately, this essentially amounts to a speed of light–type restriction, where depth is restricted due to the fact that information can only move so quickly through space. This kind of argument could also be used to show that classical RAM requires a circuit of depth $\Omega(2^{n/d})$ in a local model, in d spatial dimensions, so it does not represent a fundamental limitation that is unique to QRAM.

E. Wire density

While speed of light–type restrictions can be relevant for RAM at large scales, they are not a factor in practice at small or intermediate scales. If one ignores the speed of light, one can simply build long wires into the circuit and enable all-to-all connectivity. These long wires should not be thought of as completely free, however. For example, electronic ciruits typically dissipate energy and lead to heating in proportion to their total wire length. The need to cool electronic chips is a key limitation in practical computer systems. Thus, a cost metric worth considering [8] is the wire density of the circuit, which we define as the total wire length divided by the total spacetime of the circuit, where the spacetime is defined as the number of bits the circuit uses (henceforth referred to as the circuit width) multiplied by the circuit depth.

Classical circuits for RAM can have depth $O(n)$, width $O(2^n)$, and total wire length $O(n2^n)$ [8], even when embedded in one spatial dimension. Thus, the wire density is a constant with respect to n, suggesting the circuit can be scaled without causing heating issues.

We now examine the circuit for the update rule described in Section V-C. It has depth $O(n)$ and width $O(n2^n)$. Steps 1 and 2 perform copying of the bits and contribute wire length $O(n2^n)$. If the circuit is embedded in n spatial dimensions, then step 3 can also be accomplished with total wire length $O(n2^n)$, as each gate is local and has $O(1)$ wire length. However, in $d \leq 3$ spatial dimensions, the wire length is asymptotically larger. To implement step 3, the (yellow) ancilla bits storing the copy of $g(x)$ must be connected to the (yellow) ancilla bits initially storing the copies of $g(x \oplus e_1)$, $g(x \oplus e_2)$, $\ldots$, $g(x \oplus e_n)$—essentially, an embedding of the n-dimensional Boolean hypercube into d dimensions. Consequently, the total wire length of gates acting on each (yellow) ancilla bit will be at least $\Omega(2^{n/d})$. Since there are 2^n (yellow) ancilla bits, the overall wire length of the circuit is at least $\Omega(2^{n(1+1/d)})$ and thus the wire density grows with n as $\Omega(2^{n/d}/\text{poly}(n))$, a fundamentally different outcome than the case of classical RAM.

F. Relation to matrix-vector multiplication and the Walsh–Hadamard transform

Matrix-vector multiplication for $2^n \times 2^n$ matrices is an operation with 2^n inputs (the entries of the input vector) and 2^n outputs (the entries of the output vector). This feature is similar to the update rule, although the inputs and outputs for matrix multiplication would typically each be multiple bits (e.g., an integer or floating point number), rather than just a single bit. Moreover, the analysis of Ref. [8] demonstrated how classical sparse matrix-vector multiplication requires a growing wire density, consistent with the observation above for the update rule.

Here, we will argue that the update rule is in a certain sense equivalent to a sparse matrix-vector multiplication, and specifically it is equivalent to the Walsh–Hadamard (WH) transform up to factors of $\text{poly}(n)$. This equivalence holds under a parallel model of computation. Namely, we assume that we have 2^n classical co-processors, each with $\text{poly}(n)$-size local memory, which may perform local arithmetic on their memory in parallel, but cannot communicate with one another, except through joint application of the update rule or through joint application of a sparse matrix-vector multiplication. When jointly applying the update rule, each of the 2^n processors supplies one entry $g(x)$ and receives the output $h(x) = g(x) \oplus g(x \oplus m)$ for a fixed global m. When jointly applying sparse matrix-vector multiplication, each processor provides one of the 2^n entries of the input vector and receives one of the 2^n entries of the output vector. The remainder of this subsection aims to justify this claim of equivalence.

1) The (fast) Walsh–Hadamard transform: The WH transform is the multiplication of a length-2^n vector by a $2^n \times 2^n$ matrix denoted by H, where the matrix element

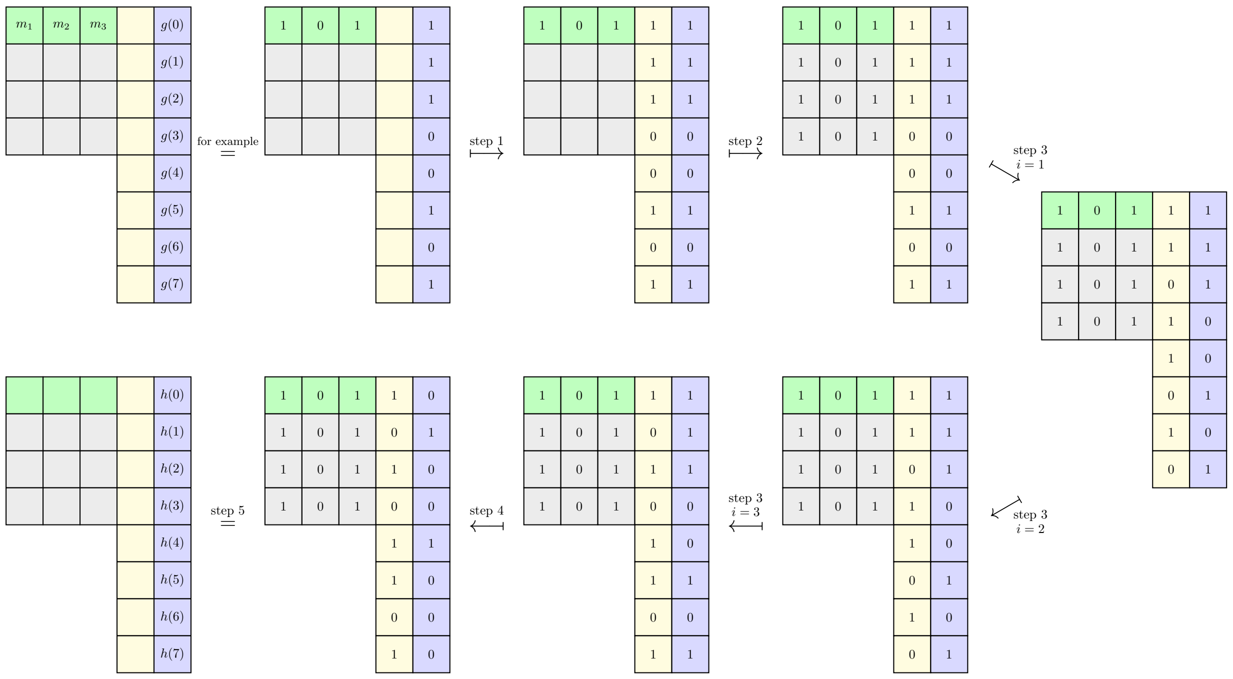


Fig. 5: Step-by-step action of the $O(n)$-depth classical circuit that implements the update rule $g \mapsto h = g \oplus g^{\oplus m}$, for a particular $n = 3$ example input with $m = (1, 0, 1)$. Each box stores one bit of information, and each step modifies some subset of these bits with parallelized layers of local gates (note that step 2 requires $n-1$ layers to create all $2^{n-1}-1$ copies of each m_i). The inputs to the update rule are the n bits of m (green) and the 2^n bits in the classical data table storing g (blue). The circuit utilizes $2^n + n(2^{n-1}-1)$ ancilla bits (gray and yellow).

associated with the transition from n-bit input address y to n-bit output address x is given by

$$H_{xy} = \frac{1}{2^{n/2}}(-1)^{x\cdot y} \tag{99}$$

We can see that H_{xy} is a dense matrix—all of its entries are nonzero—however, it can be shown that it is the product of n sparse matrices. Specifically, let $H^{(i)}$ be defined as[10]

$$H^{(i)}_{xy} = \begin{cases} 1 & \text{if } x = y \oplus e_i \\ (-1)^{x_i y_i} & \text{if } x = y \\ 0 & \text{otherwise} \end{cases} \tag{100}$$

Then, it holds that

$$H = \frac{1}{2^{n/2}} H^{(n)} H^{(n-1)} \cdots H^{(2)} H^{(1)}\,, \tag{101}$$

This fact can be verified by noting that the nonzero offdiagonal entries of $H^{(i)}$ are matrix elements $H^{(i)}_{uv}$ where u and v differ only on the i-th bit. Thus, any offdiagonal transition element H_{xy} can only be obtained in the product $H^{(n)} \cdots H^{(1)}$ by choosing the corresponding offdiagonal entry of $H^{(i)}$, (equal to 1) whenever $x_i \neq y_i$ differ, and the diagonal entry $(-1)^{x_i y_i}$ of $H^{(i)}$ whenever $x_i = y_i$. This gives precisely the quantity $(-1)^{\sum_i x_i y_i} = (-1)^{x\cdot y}$.

[10]We note that the matrix $H^{(i)}$ is proportional (by a factor $\sqrt{2}$) to the transformation applied to the amplitudes of a quantum state when a single-qubit Hadamard gate is applied to the i-th qubit, and identity is applied to the other $n-1$ qubits. The full WH transform is the product of the $H^{(i)}$ matrices, that is, the Hadamard gate on all n qubits.

Matrix multiplication by $H^{(i)}$ requires only $O(2^n)$ arithmetic operations, since each row of $H^{(i)}$ has only 2 nonzero entries. The fast WH transform utilizes this decomposition to implement multiplication by H in classical time $\mathrm{poly}(n)2^n$, much smaller than the $\Omega(2^{2n})$ time required to multiply general dense matrices that do not have this kind of decomposition.

2) Reduction from update rule to Walsh–Hadamard transform: Now, we show how the update rule can be accomplished with two applications of the WH transform and parallel local arithmetic. Let $\mathbf{g}$ denote the length-2^n vector with entry $g(x)$ at index x, and let $\mathbf{h}$ denote the length-2^n vector with entry $g(x) + g(x \oplus m)$ at entry x. Note here that we are using normal addition $+$, rather than modular addition $\oplus$, and we allow $\mathbf{h}$ to take integer values in $\{0, 1, 2\}$. We have that the x-th entry of the matrix-vector product $H\mathbf{h}$ is

$$(H\mathbf{h})_x = \frac{1}{2^{n/2}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y}(g(y) + g(y \oplus m)) \tag{102}$$

$$= \left[\frac{1}{2^{n/2}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot y} g(y) + \frac{(-1)^{m\cdot x}}{2^{n/2}} \sum_{y\in\{0,1\}^n} (-1)^{x\cdot(y\oplus m)} g(y\oplus m)\right] \tag{103}$$

$$= (H\mathbf{g})_x + (-1)^{m\cdot x}(H\mathbf{g})_x \tag{104}$$

$$= \begin{cases} 2(H\mathbf{g})_x & \text{if } m\cdot x = 0 \\ 0 & \text{if } m\cdot x = 1 \end{cases} \tag{105}$$

That is, the WH transform $H\mathbf{h}$ of the update rule output $\mathbf{h}$ can be easily computed from the WH transform $H\mathbf{g}$ of the update rule input $\mathbf{g}$.

Let $M^{(m)}$ be the diagonal matrix for which $M^{(m)}_{xx} = 2$ if $m\cdot x = 0$ and $M^{(m)}_{xx} = 0$ if $m\cdot x = 1$. Then, we may use the equation above to say that $H\mathbf{h} = MH\mathbf{g}$. Since $H^2 = \mathbb{I}$, we then have

$$\mathbf{h} = HH\mathbf{h} = HM^{(m)}H\mathbf{g} \tag{106}$$

This gives a straightforward way to transform $\mathbf{g} \mapsto \mathbf{h}$ in the model where 2^n classical co-processors can perform local arithmetic in parallel or jointly perform the WH transform. First, we assume that 2^n processors each locally store the bit $g(x)$ for one address x, and an n-bit copy of m (note that a copy of m could be pre-distributed to each of the 2^n processors via a tree-like circuit of constant wire density, as in step 2 of Section V-C). Then, $\mathbf{g} \mapsto \mathbf{h}$ is accomplished by applying the WH transform, applying the diagonal transformation $M^{(m)}$, and finally applying the WH transform again. The x-th entry of the diagonal matrix $M^{(m)}$ can be computed in $O(1)$ depth locally by the x-th processor acting on its internal memory. We recall that $\mathbf{h}$ may have entries in $\{0, 1, 2\}$; to recover binary entries, we simply take every entry modulo 2 via parallel local arithmetic, which does not discard any important information, owing to the fact that $(-1)^{g(x)+g(x\oplus m)} = (-1)^{g(x)\oplus g(x\oplus m)}$.

The conclusion is that $O(1)$ applications of the WH transform, along with parallel local arithmetic, is sufficient to implement the classical update rule, or in other words, the classical update rule is no harder than the WH transform.

3) Reduction from WH transform to update rule: Now, we show the converse: that the WH transform can be implemented using $\mathrm{poly}(n)$ applications of the update rule along with parallel local arithmetic. Specifically, we aim to implement the matrix transformation $H^{(i)}$. Suppose we are given a vector $\mathbf{g}$ of integers each represented by at most n bits.[11] Let $g(x)$ denote the integer corresponding to the entry of $\mathbf{g}$ at address x, which is stored in the local memory of one of the 2^n processors. Let $g_j(x)$ denote the j-th bit of the integer $g(x)$. We perform the following steps

1) For each address x, the co-processor storing $g(x)$ makes a copy of $g(x)$ in its local memory.
2) We fix global n-bit string $m = e_i$ (known by all processors) and for each j, the 2^n co-procesoors jointly apply the update rule $\mathrm{UR}(g_j, e_i)$ onto the set of 2^n bits $g_j(x)$ (i.e., bit-wise application of update rule). For each $x \in \{0,1\}^n$, and each j, the co-processor at address x now has in its memory the value $g_j(x)$ and the value $g_j(x) \oplus g_j(x\oplus e_i)$.
3) For each x and each j, the co-processor at address x performs local arithmetic to add (modulo 2) the local register holding $g_j(x)$ into the local register holding $g_j(x)\oplus g_j(x\oplus e_i)$ so that the two registers now hold $g_j(x)$ and $g_j(x\oplus e_i)$. Effectively, this step and the previous step have together performed a swap $g(x) \leftrightarrow g(x\oplus e_i)$ between the various co-processors.
4) For each address x, the co-processor at address x takes the local register holding integer $g(x)$ and flips it to $-g(x)$ only if $x_i = 1$.
5) Finally, for each address x, the co-processor at address x adds the local register holding $(-1)^{x_i} g(x)$ into the local register holding $g(x\oplus e_i)$, such that the latter register now holds $g(x\oplus e_i) + (-1)^{x_i} g(x)$.

The latter register in the local memory of co-processor x now holds precisely the value $(H^{(i)}\mathbf{g})_x$, indicating that the co-processors have successfully managed to apply one step of the fast WH transform. Accomplishing this required one application of the update rule for each bit of the entries of the vector. Assuming the integer entries have at most $\mathrm{poly}(n)$ bits, this means $H^{(i)}$ can be accomplished with $\mathrm{poly}(n)$ applications of the update rule, and local arithmetic, specifically, copying (step 1), bit-wise addition mod 2 (step 3), negation (step 4), and integer addition (step 5).

Since H is the product of n matrices $H^{(i)}$ up to a proportionality constant, we conclude that $\mathrm{poly}(n)$ applications of the update rule are sufficient to implement the WH transform in this model of computation; in other words, the WH transform is no harder than the update rule, up to a factor of $\mathrm{poly}(n)$.

4) Performing general sparse matrix-vector multiplication using the update rule: Multiplication by $H^{(i)}$ is in some sense easier than multiplication by an arbitrary sparse matrix, since (i) the location of the nonzero entries is highly structured and (ii) all entries are ± 1, avoiding the need for any integer or floating-point multiplications. The only arithmetic required is addition and subtraction. Here we discuss how the update rule can also be used for arbitrary sparse matrix-vector multiplications.

Lemma A.3 of Ref. [8] examines performing general sparse matrix-vector multiplication using parallel classical co-processors, each capable of local addition and multiplication.

[11]The assumption of integer entries is without loss of generality. If the entries are not integers but rather non-integer numbers expressed in binary with a finite number of bits of precision, then we can always multiply by a power of 2 so that all the entries are integers, and divide by that power of 2 after performing the calculation.

If the parallel processors can communicate with each other via links that form a sorting network, then the sparse matrix-vector multiplication can be accomplished in roughly the time required to perform a sort using the sorting network. As shown in steps 2 and 3 of the procedure in Section V-F3, the update rule and parallel bitwise xor together allow a swap of data at location x and location $x \oplus e_i$ for all x in parallel. By copying the data before performing the swap, and then choosing whether to discard the original copy or the swapped copy, one can use the update rule to swap in parallel any *subset* of the location pairs $(x, x \oplus e_i)$. Thus, the ability to perform the update rule enables access to parallel swaps along a Boolean hypercube connectivity in n dimensions, a model for which it is known that sorting can be completed in $\mathrm{poly}(n)$ time [108], [109]. Together with Ref. [8, Lemma A.3] this shows how $\mathrm{poly}(n)$ applications of the update rule and $\mathrm{poly}(n)$ rounds of parallel local arithmetic on the database entries enables arbitrary sparse matrix-vector multiplication.

Unlike for the WH transform, this procedure for arbitrary sparse matrix-vector multiplication also requires the local arithmetic to include multiplication, rather than just addition. However, $\mathrm{poly}(n)$-bit integer multiplication can be accomplished with only $\mathrm{poly}(n)$ integer additions. In any case, this suggests that the update rule is essentially equivalent to a general sparse matrix-vector multiplication, at least in this model where parallel local arithmetic is possible. We note that it could still be possible that in practice that the constant and $\mathrm{poly}(n)$ prefactors for the update rule are substantially better than those for general sparse matrix-vector multiplication.

G. Concluding comments on parallelization of the update rule

The above arguments establish that the update rule can be parallelized to $O(n)$ depth, but only in a model where all-to-all gates are possible. Embedding this all-to-all circuit into a finite number of spatial dimensions causes the circuit to have wire density that grows exponentially with n, a feature that suggests the parallelized update rule is less scalable than classical RAM. It is unclear whether this growing wire density is problematic for practical sizes of n.

Relatedly, we have shown that in a parallel model of computation, the ability to apply the update rule is roughly equivalent to the ability to apply a sparse matrix-vector multiplication, and there is a particularly close connection to the WH transform. In many instances, sparse matrix-vector multiplication can be parallelized very successfully in practice, for example, by leveraging graphics processing units (GPUs), where the parallelization is hardwired into the chip. This comes despite the fact that, asymptotically speaking, the wire density of a sparse matrix-vector multiplication would increase exponentially with n [8]. Since the classical update rule seems to be an operation that is no harder than a sparse matrix-vector multiplication, we are hopeful that it would be possible to effectively parallelize the classical update rule in practice.

However, this argument also clarifies the opportunity cost of the classical resources dedicated to performing the update rule. For example, if the quantum algorithm requiring fault-tolerant QRAM aims to solve a certain linear algebra problem, one must consider whether the classical device that performs the classical update rule is capable of solving that problem on its own, without the need for a quantum computer at all. After all, many linear algebra problems, such as solving sparse linear systems, can be solved using a small number of sparse matrix-vector multiplications; see Section VI-B and Ref. [8].

VI. Applications

In this section, we make a coarse attempt at estimating the resources required by our protocol in several applications. The goal is to shed light on which aspects of our protocol are most in need of improvement for it to be useful.

Generally, these applications come in two flavors—first, there are big-data applications that heavily rely on QRAM and require the cheap QRAM assumption from Section I to have a significant quantum speedup. For these applications, conceptually speaking, the limiting aspect of using our protocol is the exponential *classical* computation required to do the update rule (and the Clifford twirling), as this prevents a true exponential speedup and full justification of the cheap QRAM assumption. One must always consider that the classical resources required to perform the update rule could be repurposed directly toward solving the computational problem; in Section V, we discussed how the classical update rule is in a sense equivalent in complexity to a sparse matrix-vector multipliation.

The second flavor of application are those where the QRAM operation of Eq. (1) (or its b-bit generalization, discussed in Appendix A of the full version [9]) is required, but the cheap QRAM assumption is not essential—cryptanalysis and chemistry, below. In fact, in these applications, the fault-tolerant QRAM operation is typically compiled as a space-efficient quantum circuit, requiring only $O(n)$ logical qubits and $\Omega(2^n)$ depth—in this context, the operation is often referred to as QROM (quantum read-only memory) [76], rather than QRAM. Assigning QRO(A)M cost $\Omega(2^n)$ does not necessarily jeopardize the possibility of quantum advantage. This is an appealing place to apply our protocol, because it may be viewed as offloading exponential resources from the quantum processor to the classical processor, which is typically a favorable trade. The issue we face here is that the $O(1/\varepsilon)$ overhead for distillation in our method compares unfavorably to the $\mathrm{polylog}(1/\varepsilon)$ overhead achieved for distillation of T and CCZ magic states, and ε may need to be taken quite small if QRAM is applied many times. Furthermore, the values of n encountered in practice may not be sufficiently large for the asymptotic advantage of our method to kick in, although future improvements could cut down on the overhead of our protocol.

A. Arbitrary quantum state preparation

Our protocol for fault-tolerant QRAM could be used to prepare an arbitrary n-qubit state, given a list of its 2^n

amplitudes stored in classical memory. This subroutine could be useful for preparing initial states for the simulation of the dynamics of many-body systems or ansatz states for quantum phase estimation or variational algorithms. It would also be useful in certain algorithms for quantum machine learning or solving differential equations.

The process for creating an arbitrary state with QRAM goes roughly as follows [15]. First, one classically pre-processes the 2^n complex amplitudes that define the state (comprising $2^{n+1}-2$ independent real degrees of freedom, accounting for normalization and an unphysical global phase) to compute a list of $2^{n+1}-2$ rotation angles. The arbitrary state can be prepared through a sequence of steps labeled by $i=1,\dots,n$. On step i, a single-qubit rotation is performed on qubit i by one of 2^{i-1} angles; which angle to use depends on the setting of qubits $1,\dots,i-1$. If the angles are accessible via QRAM, the quantum algorithm can compute the angle with a single query by using qubits $1,\dots,i-1$ as the address. Once the angle has been read in, the single-qubit rotation on qubit i can be efficiently performed, and then a second query to the QRAM can be made to uncompute the angle. For more details on this strategy, see for example Refs. [12], [15], [39], [110]. At most $2n$ fault-tolerant QRAM queries would be required, two queries each to datasets of sizes $1,2,4,\dots,2^n$ angles. These $O(n)$ queries could be implemented fault-tolerantly by our protocol—by Theorem 2, each of the n fault-tolerant queries can be implemented up to error ε/n (so that the total error is ε) using $O(n^3/\varepsilon)$ queries to the physical QRAM device that can coherently access datasets of size up to 2^n, giving a total of $O(n^4/\varepsilon)$ physical queries. However, we note that it is possible the n dependence could be improved by leveraging the fact that only a small fraction of these queries truly require the full 2^n-size QRAM.

Thus, our protocol could represent a great improvement over the $\Omega(2^n)$ fault-tolerant quantum gates (of which at least $\Omega(\sqrt{2^n})$ must be non-Clifford gates [39]) required by a standard circuit approach to state preparation. Since state preparation is typically only one component of a larger algorithm, whether this would be a worthwhile approach within end-to-end applications may depend on the details of the application.

B. *Quantum machine learning*

The quantum linear system algorithm [22] prepares a quantum state $|\mathbf{x}\rangle$ encoding the solution to a well-conditioned $2^n\times 2^n$ linear system $A\mathbf{x}=\mathbf{b}$ using only $\mathrm{poly}(n)$ queries to data access oracles for the entries of the matrix A and the vector $\mathbf{b}$. Thus, remarkably, the number of queries needed can be exponentially smaller than the size of the matrices and vectors themselves, due to the ability of the quantum algorithm to explore the exponentially large Hilbert space in superposition. This insight, and the broader framework of quantum linear algebra [32], [106], [111], has led to a number of quantum algorithms in the realm of machine learning [10], [11], where linear algebra problems are ubiquitous.

When the entries of A and $\mathbf{b}$ have succinct formulas, the data access oracles can be implemented with $\mathrm{poly}(n)$ gate complexity directly on a fault-tolerant quantum processor. However, in big-data applications, it is more relevant to consider A and $\mathbf{b}$ as having arbitrary entries determined by the data, which is stored in classical memory. In these cases, the data access oracles, for example, a unitary block-encoding of A [106], [110], [111] or a state-preparation unitary for $|\mathbf{b}\rangle$ (see Section VI-A) can be implemented with $\mathrm{poly}(n)$ QRAM queries. Since these algorithms will require many quantum gates and many calls to the data access oracles to solve an end-to-end machine learning problem, they will only be possible once fault-tolerant quantum computers are available, and furthermore, to potentially provide exponential speedups, they will require the ability to perform QRAM fault-tolerantly at cost $\mathrm{poly}(n)$ (cheap QRAM assumption from Section I).

It is worth briefly mentioning that many of these quantum machine learning algorithms have been dequantized [112]–[117], in the sense that quantum-inspired classical algorithms can also achieve $\mathrm{poly}(n)$ complexity for problems involving datasets of size 2^n, using the ability to query individual entries of the dataset (e.g., via RAM) and also to sample an entry with probability proportional to its magnitude (a classical analogue of arbitrary state preparation). In these cases, the quantum algorithm cannot provide an exponential speedup. Polynomial speedups may still be possible if the cheap QRAM assumption holds. Moreover, these classical methods do not apply in every case; notably, when the matrices involved are sparse and high rank, the possibility of exponential quantum speedup persists [118], provided that the cheap QRAM assumption is true.

To see how our protocol impacts the outlook of these applications, we suppose generically that a quantum machine learning algorithm requires $\mathrm{poly}(n)$ fault-tolerant quantum gates and $\mathrm{poly}(n)$ fault-tolerant queries to QRAM to complete its task. By implementing QRAM using our fault-tolerant QRAM protocol with error parameter $\varepsilon = 1/\mathrm{poly}(n)$, the problem can be solved with $\mathrm{poly}(n)$ fault-tolerant gates and $\mathrm{poly}(n)$ calls to the faulty physical QRAM device, which keeps open the possibility of superpolynomial speedup in quantum resources, here assuming that the computational cost of the physical query is also $\mathrm{poly}(n)$.

However, our protocol also requires classical computations of complexity $O(2^n)$ to perform the update rule. Although this classical complexity may be parallelized, as we discussed in Section V, it is essentially equivalent to the ability to perform a sparse matrix-vector multiplication for a vector of size 2^n. This is a crucial caveat for machine learning, since sparse matrix-vector multiplication often suffices to efficiently solve the problem in the first place (see discussion in Ref. [8]). For example, the classical conjugate gradient method [119], [120] can invert well-conditioned, sparse linear systems $A\mathbf{x}=\mathbf{b}$ with $\mathrm{poly}(n)$ sparse matrix-vector multiplications. In fact, the number of sparse matrix-vector multiplications required by conjugate gradient has better asymptotic complexity (scaling as the square root of the condition number for positive semidefinite A) than the number of QRAM calls required by the quantum linear system algorithm (scaling at least linearly in the condition number [121]). This suggests that in a generic instance of the sparse linear system problem, it would be more efficient to apply $\Omega(2^n)$ classical computational resources directly toward solving the linear algebra problem, rather than to perform the update rule required by our protocol.

That said, a few opportunities remain. For instance, one can consider the case that the sparse $2^n \times 2^n$ matrix A has repeated entries or some other kind of high-level structure, such that the number of classical degrees of freedom is asymptotically less than 2^n. For concreteness, suppose that the size of the dataset determining A is only $\sqrt{2^n}$. Then, the QRAM operation needs only to be applied for size-$\sqrt{2^n}$ datasets, and the classical complexity of the update rule is only $O(\sqrt{2^n})$, quadratically cheaper than the number of arithmetic operations required for a full matrix-vector multiplication by the matrix A. In this situation, our protocol may enable an end-to-end solution with $\mathrm{poly}(n)$ quantum cost and $O(\sqrt{2^n})$ classical cost, which could represent a quadratic speedup in terms of classical complexity and an exponential speedup in terms of quantum complexity. While quadratic *quantum* speedups are typically thought of as insufficient to overcome the large disadvantage in constant prefactor for quantum computation [75], a quadratic *classical* speedup faces no such obstacles, and could be considered quite large. Generalizing this line of thinking, we may expect to find end-to-end speedups in situations where the best achievable classical complexity is asymptotically greater than the number of classical degrees of freedom of the problem. Toward this end, another example worth exploring may be the inversion of dense matrices with $O(2^{2n})$ degrees of freedom, which generally requires $2^{\omega n}$ classical arithmetic operations with $\omega > 2$. It remains to find concrete end-to-end examples where quantum advantages of this kind may come to fruition.

C. Cryptanalysis

Shor's algorithm for factoring or computing the discrete logarithm relies on performing coherent modular arithmetic. Gidney's windowed quantum arithmetic [122] has been used to reduce the cost of Shor's algorithm by replacing some of this costly arithmetic with coherent reads from a quantum lookup table of classically precomputed values [123], [124]. The algorithm can be expressed as repeated blocks of a coherent lookup table read, each followed by a coherent addition.[12] We investigate the performance of our distillation-based QRAM scheme for implementing the coherent lookup table reads in Shor's algorithm. Elliptic curve cryptography (ECC) is a more attractive target than factoring because of the smaller number of calls to the lookup table, which allows a less stringent target ε, as well as the high cost of elliptic curve addition compared to modular addition. We follow the presentation of Ref. [125], where Shor's algorithm for ECC is presented as two applications of quantum phase estimation on unitaries of the form:

$$U_X|R\rangle = |R + X\rangle, \tag{107}$$

where $R = (x, y)$ is an elliptic curve point, and X denotes either the base point P or the public key Q. The goal is to compute integer j such that $Q = [j]P$, where the notation $[j]$ indicates that we are performing elliptic curve scalar multiplication of the point P j times (see Ref. [125] for full definitions of the addion and multiplication operations). Quantum phase estimation uses controlled unitaries of the form $U_X^{2^j}$ for $0 \leq j \leq k-1$, where k denotes the number of bits in the ECC scheme (e.g., ECC-256). In Ref. [125], windowed quantum arithmetic is used to replace blocks of 16 controlled unitaries with a single QROM load of 2^{16} classically pre-computed values. The load is followed by an elliptic curve addition operation. The cost of loading N pieces of classical data from a QROM is N Toffoli gates. For $N = 2^{16}$, this is approximately 6.5×10^4 Toffolis. We note that this is much smaller than the cost of the elliptic curve addition operation, which is approximately 8.34×10^6. The minimum Toffoli cost of the algorithm is obtained by loading groups of 2^{19} values, while the minimum active volume cost is obtained at 2^{16} values.

For the sake of calculation, we suppose that the physical QRAM device produces states with minimum fidelity of $F = 50\%$, and we seek to achieve error $\varepsilon_{\mathrm{tot}} = 0.1$ (where $\varepsilon_{\mathrm{tot}}$ is the sum of the errors of all QRAM loads in the algorithm). We use the generalized b-bit version of our protocol from Appendix A of the full version [9]. If we choose the iterated swap test distillation protocol for its simplicity (each swap test requiring exactly $n + b$ non-Clifford Toffoli gates), for non-vanishing F, the number of non-Clifford gates for distillation scales roughly as $O((1-F)n^2(n+b)/\varepsilon))$ with ε the error per fault-tolerant QRAM query (note that the twirling and teleportation steps are entirely Clifford). For $F \sim 50\%$, and assuming for simplicity a constant prefactor of 1, we estimate the non-Clifford cost as $n^2(n+b)/(2\varepsilon)$. We use this expression to calculate that the QRAM-based approach achieves its minimum Toffoli and active volume cost at a group size of 2^{64} values, which is impractically large for the classical update step of our scheme. A more realistic size of 2^{32} values only increases the costs by approximately 10%. Nevertheless, both the minimum Toffoli and active volume costs of the existing QROM-based approach are approximately a factor of $1.8\times$ and $1.6\times$ lower than the minimum costs of our QRAM-scheme, respectively, and we have not even considered the computational cost of applying the QRAM device itself. A major contribution to the cost of our method stems from the large amount of data to be loaded ($b = 512$), which features multiplicatively in our Toffoli costs. In contrast, the Toffoli cost of the QROM scheme is independent of the value of b. Improved methods for distillation, especially when b is large, could make our scheme more competitive in this application.

[12]This could be regular addition, modular addition, or elliptic curve point addition.

D. *Chemistry*

Coherent data access using a quantum lookup table has become a key subroutine in modern algorithms for quantum chemistry [76]. These algorithms assume access to a block-encoding of the Hamiltonian. This block-encoding is typically implemented by writing the Hamiltonian as a linear combination of unitaries $H = \sum_{j=0}^{L-1} c_j U_j$ and using oracles of the form:

$$\text{PREPARE}\,|0^{\lceil \log_2(L) \rceil}\rangle = \frac{1}{\sqrt{\lambda}} \sum_{j=0}^{L-1} \sqrt{|c_j|}|j\rangle \tag{108}$$

$$\text{SELECT} = \sum_{j=0}^{L-1} |j\rangle\langle j| \otimes \text{sign}(c_j) U_j + \sum_{j=L}^{2^{\lceil \log_2(L) \rceil}-1} |j\rangle\langle j| \otimes \mathbb{I}, \tag{109}$$

where $\lambda = \sum_j |c_j|$ is the normalization factor of the block-encoding. The LCU approach to block-encodings still works if PREPARE results in each computational basis state $|j\rangle$ being entangled with a garbage register. The technique of coherent alias sampling, introduced in Ref. [76], provides an efficient approach for implementing PREPARE in this way, with a cost of $O(L + \log(1/\delta))$ Toffoli gates using QROM, where δ is the largest error in $\sqrt{|c_j|}$. The dependence on L can be improved quadratically using the techniques of Ref. [39]. In the most straightforward application of these techniques (referred to as "sparse qubitization" [126]) the complexity of PREPARE dominates the algorithm, and its cost depends on the cost to load the Hamiltonian coefficients from a quantum lookup table.

Sparse qubitization requires $O(\lambda/\Delta)$ calls to the block-encoding of the Hamiltonian, where Δ is the precision on the energy estimate of the Hamiltonian. For typical λ values of small molecules, in the range 10^2–10^4 Hartree, and $\Delta = 10^{-3}$ Hartree, this implies at least 10^5 calls to the quantum lookup table. If we were to replace the calls to the quantum lookup table with our distillation-based QRAM scheme for the example of FeMo-co ($\lambda = 7614$, $N = L = 179{,}498$, $b = 84$ [126]), using the same methodology as in the previous section, we conclude that we would require at least 2.5×10^{11} Toffoli gates per call to the block-encoding, substantially larger than the value of 10^4 Toffolis in Ref. [126]. We note that even if the dependence of our approach on the error ε per query could be reduced to $O(\log(1/\varepsilon))$, it is unclear whether our approach would improve over existing methods, as the amount of data (i.e., Lb) loaded for the chemistry Hamiltonian is sufficiently modest that the QROM scaling of $O(\sqrt{Lb})$ is comparable to the scaling of our protocol $O(\log^2(L)(\log(L)+b))$ (see Appendix A of the full version [9]).

The sparse qubitization approach has been superseded in some applications by more efficient methods [127], [128], which also use a lookup table to coherently load angles for basis rotations that are used in the SELECT oracle. We refer to Refs. [128], [129] for a detailed accounting of the contribution of the lookup table reads to the total gate and space complexity. It is unlikely that our aproach to implementing QRAM would reduce the costs of these algorithms, at least in its current form.

VII. Outlook on the Cheap QRAM Assumption

A central goal of this research program is to determine whether QRAM can be considered equally cheap as RAM—at least in an abstract, asymptotic sense—or whether its quantum nature makes QRAM fundamentally more difficult, preventing a justification of the cheap QRAM assumption from Section I. The central aspect of QRAM that differentiates it from RAM is the need to protect the quantum information about which address (or superposition of addresses) is being queried, even when the hardware has errors. To emphasize this distinction, it is worth noting that a fault-tolerant RAM could be easily constructed from a faulty RAM device using a simple repetition code: by repeatedly querying the faulty RAM device on the same input address, one can efficiently boost the probability of a successful RAM query, provided that the device has nonzero bias in favor of the correct answer. In contrast, this same strategy fails for QRAM because even a single query to a faulty QRAM device may leak the address information and decohere the address register. Formally speaking, we might say that logical (Q)RAM is a transversal gate for the repetition code, but that this is not sufficient for fault-tolerant QRAM, since the repetition code does not protect against phase-flip errors. For fault-tolerant QRAM, a more sophisticated strategy is required.

Our protocol provides such a method for fault-tolerant QRAM, successfully protecting which address state $\sum_x \alpha_x |\overline{x}\rangle$ is being queried without performing QEC on the $\Omega(2^n)$ components of the QRAM device. It does this by relying on resource states (see Eq. (9)) that are equal superpositions of all 2^n addresses, independent of which superposition of addresses one wants to query—the noisy device is prevented from direct interaction with the address information. However, these resource states have exponentially small amplitude on any individual address. Consequently, if the data at one address is modified, the ideal resource state barely changes. This begs the question: if these resource states are insensitive to the underlying bits in the dataset, how, then, can they be used to insert information about the dataset into the quantum state? Our protocol achieves this by iteratively and adaptively inserting *global* information about the dataset f—that is, properties of f that depend on all 2^n data entries—into the quantum state. One way to see this is by decomposing $f(x)$ into a sum over global, oscillatory contributions of the form $(-1)^{k \cdot x}$, that is, its Fourier expansion

$$f(x) = \frac{1}{2^{n/2}} \sum_{k \in \{0,1\}^n} (-1)^{k \cdot x} \tilde{f}(k), \tag{110}$$

where $\tilde{f}$ is the Walsh–Hadamard transform of f. When we teleport the resource state $|\overline{\Psi(f)}\rangle$, we enact the QRAM unitary $\overline{V(f^{\oplus m})}$ instead of $\overline{V(f)}$, where m is uniformly random and $f^{\oplus m}$ is defined by $f^{\oplus m}(x) = f(x \oplus m)$. While the datasets

$f^{\oplus m}$ and f are not guaranteed to agree on any of the addresses, they have a close relationship in frequency space. One can see from Eq. (110) that

$$\tilde{f}^{\oplus m}(k) = (-1)^{k \cdot m} \tilde{f}(k) \,. \tag{111}$$

Thus, $\tilde{f}^{\oplus m}(k) = \tilde{f}(k)$ for half the values of k and $\tilde{f}^{\oplus m}(k) = -\tilde{f}(k)$ for the other half (except in the unlikely case that $m = 0^n$, in which case they would agree on all values). In a sense, we may say that by implementing $\overline{V(f^{\oplus m})}$, we have successfully inserted half of the information about the function f into the quantum state, regardless of which random m is obtained. To insert the other half of the information, the protocol updates the dataset to $f' = f \oplus f^{\oplus m}$. We may observe that f' is periodic in translation by m (i.e., $f'(x) = f'(x \oplus m)$ for all x), and thus $\tilde{f}'(k) = 0$ for any k for which $k \cdot m = 1$ (half the values of k). Each successive correction function will have half as many nonzero Fourier components as the previous one (assuming the n-bit measurement outcomes in previous rounds form a linearly independent set), until finally after n rounds there is no remaining frequency information left to insert. This global approach to information insertion appears crucial for correctly applying QRAM on any input state, even while not knowing or learning what that input state is.

The cost of this global approach is adaptivity and classical computation. At each iteration, we do not have control over which half of the frequency information we insert into the quantum state; this is determined by a uniformly random measurement outcome m. After receiving m, the protocol must adaptively update the *entire* dataset to essentially remove the half of the global information that has already been applied to the quantum state. This removal requires touching all 2^n entries of the dataset, and then giving the physical QRAM device access to the new dataset. As discussed in Section V, this updating of frequency information is a non-negligible classical computation, essentially equivalent in complexity to performing the Walsh–Hadamard transformation on the dataset. While the Walsh–Hadamard transform may be parallelizable, it appears to be a harder calculation than a normal RAM query; for example, it requires fundamentally greater wire density than RAM.

The main theoretical open question, then, is to determine if the classical complexity of the protocol can be reduced to a quantity more similar to a RAM query, or otherwise find a way to justify that the overall (i.e., both classical and quantum) cost of QRAM is $\mathrm{poly}(n)$. Our protocol makes some progress in this direction, and it offers clear advantages over actively error corrected circuit QRAM at the practical level—enabling the usage of a specialized low-fidelity QRAM device, and eliminating the need for massively parallel QEC. However, in a theoretical sense, both our protocol and circuit QRAM ultimately require the same $\Omega(2^n)$ scaling of classical resources to protect the address information from decoherence. One is left to wonder whether this scaling of classical complexity could be a fundamental requirement for achieving fault-tolerant QRAM.

Acknowledgment

The authors thank Mario Berta, Joe Iverson, Sam Jaques, Michael Kastoryano, Fernando Pastawski, Samson Wang, and John Wright for helpful conversations. We also thank Simone Severini, James Hamilton, Nafea Bshara, Peter DeSantis, and Andy Jassy for their involvement and support of the research activities at the AWS Center for Quantum Computing.

References

[1] F. C. Williams and T. Kilburn, "Electronic digital computers," *Nature*, vol. 162, no. 4117, pp. 487–487, 1948.

[2] W. F. C. and K. T., "A storage system for use with binary-digital computing machines," *Proceedings of the IEE—Part II: Power Engineering*, vol. 96, no. 50, pp. 183–200, 2025/03/17 1949.

[3] Engineering and Technology History Wiki, "Milestones: Manchester University "Baby"' computer and its derivatives, 1948–1951," 2022, https://ethw.org/Milestones:Manchester_University_%22Baby%22_Computer_and_its_Derivatives,_1948-1951, accessed 2025-03-17.

[4] S. Kim, C. Hooper, T. Wattanawong, M. Kang, R. Yan, H. Genc, G. Dinh, Q. Huang, K. Keutzer, M. W. Mahoney, S. Shao, and A. Gholami, "Full stack optimization of transformer inference," in *Architecture and System Support for Transformer Models (ASSYST)*, 2023, https://openreview.net/forum?id=GtyQbLUUagE, accessed: 2025-03-17. arXiv:2302.14017.

[5] C. Silvano, D. Ielmini, F. Ferrandi, L. Fiorin, S. Curzel, L. Benini, F. Conti, A. Garofalo, C. Zambelli, E. Calore, S. Schifano, M. Palesi, G. Ascia, D. Patti, N. Petra, D. De Caro, L. Lavagno, T. Urso, V. Cardellini, G. Cardarilli, R. Birke, and S. Perri, "A survey on deep learning hardware accelerators for heterogeneous HPC platforms," *ACM Comput. Surv.*, 4 2025, arXiv:2306.15552.

[6] Infineon, "CY7C1069G-10BVXIT," 2025, https://www.infineon.com/cms/en/product/memories/sram-static-ram/asynchronous-sram/cy7c1069g-10bvxit/, accessed: 2025-03-17.

[7] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Phys. Rev. Lett.*, vol. 100, no. 16, p. 160501, 2008, arXiv:0708.1879.

[8] S. Jaques and A. G. Rattew, "QRAM: A survey and critique," 2023, arXiv:2305.10310.

[9] A. M. Dalzell, A. Gilyén, C. T. Hann, S. McArdle, G. Salton, Q. T. Nguyen, A. Kubica, and F. G. S. L. Brandão, "A distillation-teleportation protocol for fault-tolerant QRAM," 2025, arXiv:2505.20265 (Full version of this paper).

[10] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, "Quantum machine learning: A classical perspective," *Proc. R. Soc. A*, vol. 474, no. 2209, p. 20170551, 2018, arXiv:1707.08561.

[11] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, pp. 195–202, 2017, arXiv:1611.09347.

[12] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica, G. Salton, S. Wang, and F. G. S. L. Brandão, *Quantum algorithms: A survey of applications and end-to-end complexities*. Cambridge University Press, 2025, arXiv:2310.03011.

[13] P. Rebentrost, M. Mohseni, and S. Lloyd, "Quantum support vector machine for big data classification," *Phys. Rev. Lett.*, vol. 113, no. 13, p. 130503, 2014, arXiv:1307.0471.

[14] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, "Quantum-assisted Gaussian process regression," *Phys. Rev. A*, vol. 99, no. 5, p. 052331, 2019, arXiv:1512.03929.

[15] I. Kerenidis and A. Prakash, "Quantum recommendation systems," in *ITCS*, 2017, pp. 49:1–49:21, arXiv:1603.08675.

[16] D. W. Berry, "High-order quantum algorithm for solving linear differential equations," *J. Phys. A*, vol. 47, no. 10, p. 105301, 2014, arXiv:1010.2745.

[17] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang, “Quantum algorithm for linear differential equations with exponentially improved dependence on precision,” *Commun. Math. Phys.*, vol. 356, no. 3, pp. 1057–1081, 2017, arXiv:1701.03684.

[18] A. M. Childs and J.-P. Liu, “Quantum spectral methods for differential equations,” *Commun. Math. Phys.*, vol. 375, no. 2, pp. 1427–1457, 2020, arXiv:1901.00961.

[19] H. Krovi, “Improved quantum algorithms for linear and nonlinear differential equations,” *Quantum*, vol. 7, p. 913, 2023, arXiv:2202.01054.

[20] D. Jennings, M. Lostaglio, R. B. Lowrie, S. Pallister, and A. T. Sornborger, “The cost of solving linear differential equations on a quantum computer: Fast-forwarding to explicit resource counts,” *Quantum*, vol. 8, p. 1553, 12 2024, arXiv:2309.07881.

[21] D. W. Berry and P. C. S. Costa, “Quantum algorithm for time-dependent differential equations using Dyson series,” *Quantum*, vol. 8, p. 1369, 6 2024, arXiv:2212.03544.

[22] A. W. Harrow, A. Hassidim, and S. Lloyd, “Quantum algorithm for linear systems of equations,” *Phys. Rev. Lett.*, vol. 103, no. 15, p. 150502, 2009, arXiv:0811.3171.

[23] F. G. S. L. Brandão and K. M. Svore, “Quantum speed-ups for solving semidefinite programs,” in *FOCS*, 2017, pp. 415–426, arXiv:1609.05537.

[24] F. G. S. L. Brandão, A. Kalev, T. Li, C. Y.-Y. Lin, K. M. Svore, and X. Wu, “Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning,” in *ICALP*, 2019, pp. 27:1–27:14, arXiv:1710.02581.

[25] J. van Apeldoorn, A. Gilyén, S. Gribling, and R. de Wolf, “Quantum SDP-solvers: Better upper and lower bounds,” *Quantum*, vol. 4, p. 230, 2020, earlier version in *FOCS'17*. arXiv:1705.01843.

[26] J. van Apeldoorn and A. Gilyén, “Quantum algorithms for zero-sum games,” 2019, arXiv:1904.03180.

[27] ——, “Improvements in quantum SDP-solving with applications,” in *ICALP*, 2019, pp. 99:1–99:15, arXiv:1804.05058.

[28] I. Kerenidis and A. Prakash, “A quantum interior point method for LPs and SDPs,” *ACM Trans. Quantum Comput.*, vol. 1, no. 1, 2020, arXiv:1808.09266.

[29] I. Kerenidis, A. Prakash, and D. Szilágyi, “Quantum algorithms for second-order cone programming and support vector machines,” *Quantum*, vol. 5, p. 427, 2021, arXiv:1908.06720.

[30] B. Augustino, G. Nannicini, T. Terlaky, and L. F. Zuluaga, “Quantum interior point methods for semidefinite optimization,” *Quantum*, vol. 7, p. 1110, 9 2023, arXiv:2112.06025.

[31] I. Kerenidis, A. Prakash, and D. Szilágyi, “Quantum algorithms for portfolio optimization,” in *AFT*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 147–155, arXiv:1908.08040.

[32] A. M. Dalzell, B. D. Clader, G. Salton, M. Berta, C. Y.-Y. Lin, D. A. Bader, N. Stamatopoulos, M. J. A. Schuetz, F. G. S. L. Brandão, H. G. Katzgraber, and W. J. Zeng, “End-to-end resource analysis for quantum interior-point methods and portfolio optimization,” *PRX Quantum*, vol. 4, p. 040325, 11 2023, arXiv:2211.12489.

[33] S. Arunachalam, V. Gheorghiu, T. Jochym-O'Connor, M. Mosca, and P. V. Srinivasan, “On the robustness of bucket brigade quantum RAM,” *New J. Phys.*, vol. 17, no. 12, p. 123010, 2015, arXiv:1502.03450.

[34] D. S. Steiger and M. Troyer, “Racing in parallel: Quantum versus classical,” in *APS March Meeting Abstracts*, vol. 2016, 2016, pp. H44–010.

[35] O. Di Matteo, V. Gheorghiu, and M. Mosca, “Fault-tolerant resource estimation of quantum random-access memories,” *IEEE Trans. Quantum Eng.*, vol. 1, pp. 1–13, 2020, arXiv:1902.01329.

[36] A. Paler, O. Oumarou, and R. Basmadjian, “Parallelizing the queries in a bucket-brigade quantum random access memory,” *Phys. Rev. A*, vol. 102, p. 032608, 9 2020, arXiv:2002.09340.

[37] C. T. Hann, G. Lee, S. Girvin, and L. Jiang, “Resilience of quantum random access memory to generic noise,” *PRX Quantum*, vol. 2, p. 020311, 4 2021, arXiv:2012.05340.

[38] P. Mukhopadhyay, “A quantum random access memory (qram) using a polynomial encoding of binary strings,” *Sci. Rep.*, vol. 15, no. 1, p. 11002, 2025, arXiv:2408.16794.

[39] G. H. Low, V. Kliuchnikov, and L. Schaeffer, “Trading T gates for dirty qubits in state preparation and unitary synthesis,” *Quantum*, vol. 8, p. 1375, 6 2024, arXiv:1812.00954.

[40] R. C. Jaeger and T. N. Blalock, *Microelectronic circuit design*, 5th ed. McGraw-Hill New York, 2016, vol. 97.

[41] Y. Wang, Y. Alexeev, L. Jiang, F. T. Chong, and J. Liu, “Fundamental causal bounds of quantum random access memories,” *npj Quant. Inf.*, vol. 10, no. 1, p. 71, 2024, arXiv:2307.13460.

[42] B. Zeng, X. Chen, and I. L. Chuang, “Semi-Clifford operations, structure of $\mathcal{C}_k$ hierarchy, and gate complexity for fault-tolerant quantum computation,” *Phys. Rev. A*, vol. 77, no. 4, p. 042313, 2008, arXiv:0712.2084.

[43] M. B. Hastings and J. Haah, “Distillation with sublogarithmic overhead,” *Phys. Rev. Lett.*, vol. 120, no. 5, p. 050504, 2018, arXiv:1709.03543.

[44] A. Kubica and M. E. Beverland, “Universal transversal gates with color codes: A simplified approach,” *Phys. Rev. A*, vol. 91, p. 032330, 2015, arXiv:1410.0069.

[45] S. Koutsioumpas, D. Banfield, and A. Kay, “The smallest code with transversal T,” 2022, arXiv:2210.14066.

[46] G. Kuperberg, “Another subexponential-time quantum algorithm for the dihedral hidden subgroup problem,” in *TQC*, ser. Leibniz International Proceedings in Informatics (LIPIcs), S. Severini and F. Brandao, Eds., vol. 22. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013, pp. 20–34, arXiv:1112.3333.

[47] S. X. Cui, D. Gottesman, and A. Krishna, “Diagonal gates in the Clifford hierarchy,” *Phys. Rev. A*, vol. 95, p. 012329, 1 2017, arXiv:1608.06596.

[48] D. Horsman, A. G. Fowler, S. Devitt, and R. V. Meter, “Surface code quantum computing by lattice surgery,” *New J. Phys.*, vol. 14, no. 12, p. 123011, 2012, arXiv:1111.4022.

[49] Y. Li, “A magic state's fidelity can be superior to the operations that created it,” *New J. Phys.*, vol. 17, no. 2, p. 023037, 2015, arXiv:1410.7808.

[50] J. Łodyga, P. Mazurek, A. Grudka, and M. Horodecki, “Simple scheme for encoding and decoding a qubit in unknown state for various topological codes,” *Sci. Rep.*, vol. 5, no. 1, p. 8975, 2015, arXiv:1404.2495.

[51] D. Litinski, “Magic state distillation: Not as costly as you think,” *Quantum*, vol. 3, p. 205, 12 2019, arXiv:1905.06903.

[52] S. Bravyi and A. Kitaev, “Universal quantum computation with ideal Clifford gates and noisy ancillas,” *Phys. Rev. A*, vol. 71, no. 2, p. 022316, 2005, arXiv:quant-ph/0403025.

[53] E. Knill, “Fault-tolerant postselected quantum computation: Schemes,” 2004, arXiv:quant-ph/0402171.

[54] H. Bombín and M. A. Martin-Delgado, “Topological quantum distillation,” *Phys. Rev. Lett.*, vol. 97, p. 180501, 2006, arXiv:quant-ph/0605138.

[55] A. Kubica, B. Yoshida, and F. Pastawski, “Unfolding the color code,” *New J. Phys.*, vol. 17, no. 8, p. 083026, 2015, arXiv:1503.02065.

[56] J. E. Moussa, “Transversal Clifford gates on folded surface codes,” *Phys. Rev. A*, vol. 94, p. 042316, 2016, arXiv:1603.02286.

[57] D. Litinski, “A game of surface codes: Large-scale quantum computing with lattice surgery,” *Quantum*, vol. 3, p. 128, 3 2019, arXiv:1808.02892.

[58] C. Gidney and A. G. Fowler, “Flexible layout of surface code computations using AutoCCZ states,” 2019, arXiv:1905.08916.

[59] M. Beverland, E. Campbell, M. Howard, and V. Kliuchnikov, “Lower bounds on the non-Clifford resources for quantum computations,” *Quantum Sci. Technol.*, vol. 5, no. 3, p. 035009, 2020, arXiv:1904.01124.

[60] D. Gottesman and I. L. Chuang, "Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations," *Nature*, vol. 402, no. 6760, pp. 390–393, 1999, arXiv:quant-ph/9908010.

[61] M. Christandl, O. Fawzi, and A. Goswami, "Fault-tolerant quantum input/output," 2024, arXiv:2408.05260.

[62] A. Wills, M.-H. Hsieh, and H. Yamasaki, "Constant-overhead magic state distillation," 2024, arXiv:2408.07764.

[63] Q. T. Nguyen, "Good binary quantum codes with transversal CCZ gate," 2024, arXiv:2408.10140.

[64] L. Golowich and V. Guruswami, "Asymptotically good quantum codes with transversal non-Clifford gates," 2024, arXiv:2408.09254.

[65] J. I. Cirac, A. K. Ekert, and C. Macchiavello, "Optimal purification of single qubits," *Phys. Rev. Lett.*, vol. 82, pp. 4344–4347, 5 1999, arXiv:quant-ph/9812075.

[66] M. Keyl and R. F. Werner, "The rate of optimal purification procedures," *Annales Henri Poincaré*, vol. 2, no. 1, pp. 1–26, 2001, arXiv:quant-ph/9910124.

[67] J. Fiurášek, "Optimal probabilistic cloning and purification of quantum states," *Phys. Rev. A*, vol. 70, p. 032308, 9 2004, arXiv:quant-ph/0403165.

[68] H. Fu, "Quantum state purification," Master's thesis, University of Waterloo, 2016.

[69] A. M. Childs, H. Fu, D. Leung, Z. Li, M. Ozols, and V. Vyas, "Streaming quantum state purification," *Quantum*, vol. 9, p. 1603, 2025, arXiv:2309.16387.

[70] Z. Li, H. Fu, T. Isogawa, and I. Chuang, "Optimal quantum purity amplification," 2024, arXiv:2409.18167.

[71] D. Grier, D. Leung, Z. Li, H. Pashayan, and L. Schaeffer, "Streaming quantum state purification for general mixed states," 2025, arXiv:2503.22644.

[72] S. Irani, A. Natarajan, C. Nirkhe, S. Rao, and H. Yuen, "Quantum search-to-decision reductions and the state synthesis problem," in *CCC*, ser. CCC '22. Dagstuhl, DEU: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2022, arXiv:2111.02999.

[73] S. Lloyd, M. Mohseni, and P. Rebentrost, "Quantum principal component analysis," *Nat. Phys.*, vol. 10, pp. 631–633, 2014, arXiv:1307.0401.

[74] S. Kimmel, C. Y.-Y. Lin, G. H. Low, M. Ozols, and T. J. Yoder, "Hamiltonian simulation with optimal sample complexity," *npj Quant. Inf.*, vol. 3, no. 1, p. 13, 2017, arXiv:1608.00281.

[75] R. Babbush, J. R. McClean, M. Newman, C. Gidney, S. Boixo, and H. Neven, "Focus beyond quadratic speedups for error-corrected quantum advantage," *PRX Quantum*, vol. 2, p. 010103, 3 2021, arXiv:2011.04149.

[76] 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," *Phys. Rev. X*, vol. 8, no. 4, p. 041015, 2018, arXiv:1805.03662.

[77] V. Giovannetti, S. Lloyd, and L. Maccone, "Architectures for a quantum random access memory," *Phys. Rev. A*, vol. 78, no. 5, p. 052310, 2008, arXiv:0807.4994.

[78] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2000.

[79] D. Weiss, S. Puri, and S. Girvin, "Quantum random access memory architectures using 3d superconducting cavities," *PRX Quantum*, vol. 5, no. 2, p. 020312, 2024, arXiv:2310.08288.

[80] C. T. Hann, C.-L. Zou, Y. Zhang, Y. Chu, R. J. Schoelkopf, S. M. Girvin, and L. Jiang, "Hardware-efficient quantum random access memory with hybrid quantum acoustic systems," *Phys. Rev. Lett.*, vol. 123, no. 25, p. 250501, 2019, arXiv:1906.11340.

[81] Z. Wang, H. Qiao, A. N. Cleland, and L. Jiang, "Quantum random access memory with transmon-controlled phonon routing," 2024, arXiv:2411.00719.

[82] A. Sala Cadellans, "A transmon based quantum switch for a quantum random access memory," Master's thesis, Leiden University, 2015.

[83] K. C. Chen, W. Dai, C. Errando-Herranz, S. Lloyd, and D. Englund, "Scalable and high-fidelity quantum random access memory in spin-photon networks," *PRX Quantum*, vol. 2, no. 3, p. 030319, 2021, arXiv:2103.07623.

[84] F.-Y. Hong, Y. Xiang, Z.-Y. Zhu, L.-z. Jiang, and L.-n. Wu, "Robust quantum random access memory," *Phys. Rev. A*, vol. 86, no. 1, p. 010306, 2012, arXiv:1201.2250.

[85] F. Cesa, H. Bernien, and H. Pichler, "Fast and error-correctable quantum RAM," 2025, arXiv:2503.19172.

[86] W. J. Huggins, T. Khattar, and N. Wiebe, "Productionizing quantum mass production," 2025, arXiv:2506.00132.

[87] Z. Ji, Y.-K. Liu, and F. Song, "Pseudorandom quantum states," in *CRYPTO*, H. Shacham and A. Boldyreva, Eds. Cham: Springer International Publishing, 2018, pp. 126–152, arXiv:1711.00385.

[88] Z. Brakerski and O. Shmueli, "(Pseudo) random quantum states with binary phase," in *Theory of Cryptography*, D. Hofheinz and A. Rosen, Eds. Cham: Springer International Publishing, 2019, pp. 229–250, arXiv:1906.10611.

[89] S. Arunachalam, S. Bravyi, A. Dutt, and T. J. Yoder, "Optimal algorithms for learning quantum phase states," in *TQC*, ser. Leibniz International Proceedings in Informatics (LIPIcs), O. Fawzi and M. Walter, Eds., vol. 266. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, pp. 3:1–3:24, arXiv:2208.07851.

[90] D. Aharonov and M. Ben-Or, "Fault-tolerant quantum computation with constant error rate," *SIAM J. Comp.*, vol. 38, no. 4, pp. 1207–1282, 7 2008, earlier version in *STOC'97*. arXiv:quant-ph/9906129.

[91] P. W. Shor, "Fault-tolerant quantum computation," in *FOCS*. IEEE Comput. Soc. Press, 1996, pp. 56–65, arXiv:quant-ph/9605011.

[92] D. Gottesman, "Fault-tolerant quantum computation with constant overhead," *Quantum Inf. Comput.*, vol. 14, no. 15–16, pp. 1338–1372, 2014, arXiv:1310.2984.

[93] ——, "An introduction to quantum error correction and fault-tolerant quantum computation," in *Proceedings of Symposia in Applied Mathematics*, vol. 68, 2010, pp. 13–58, arXiv:0904.2557.

[94] O. Fawzi, A. Grospellier, and A. Leverrier, "Constant overhead quantum fault tolerance with quantum expander codes," *Commun. ACM*, vol. 64, no. 1, p. 106–114, 12 2020, earlier version in *FOCS'18*. arXiv:1808.03821.

[95] H. Yamasaki and M. Koashi, "Time-efficient constant-space-overhead fault-tolerant quantum computation," *Nat. Phys.*, vol. 20, no. 2, pp. 247–253, 2024, arXiv:2207.08826.

[96] Q. T. Nguyen and C. A. Pattison, "Quantum fault tolerance with constant-space and logarithmic-time overheads," 2024, arXiv:2411.03632.

[97] C. Dankert, "Efficient simulation of random quantum states and operators," 2005, arXiv:quant-ph/0512217.

[98] S. T. Flammia and J. J. Wallman, "Efficient estimation of Pauli channels," *ACM Trans. Quantum Comput.*, vol. 1, no. 1, 12 2020, arXiv:1907.12976.

[99] J. J. Wallman and J. Emerson, "Noise tailoring for scalable quantum computation via randomized compiling," *Phys. Rev. A*, vol. 94, p. 052325, 11 2016, arXiv:1512.01098.

[100] R. Mehta, G. Lee, and L. Jiang, "Analysis and suppression of errors in quantum random access memory errors under extended noise models," 2024, arXiv:2412.10318.

[101] C. J. Wood, J. D. Biamonte, and D. G. Cory, "Tensor networks and graphical calculus for open quantum systems," *Quantum Inf. Comput.*, vol. 15, no. 9–10, pp. 759–811, 2015, arXiv:1111.6950.

[102] A. Y. Kitaev, "Quantum measurements and the abelian stabilizer problem," 1995, arXiv:quant-ph/9511026.

[103] Y. Chen, A. Gilyén, and R. de Wolf, "A quantum speed-up for approximating the top eigenvectors of a matrix," in *SODA*, 2025, pp. 994–1036, arXiv:2405.14765.

[104] N. S. Mande and R. de Wolf, "Tight bounds for quantum phase estimation and related problems," in *ESA*, vol. 274, 2023, pp. 81:1–81:16, arXiv:2305.04908.

[105] N. Wiebe and C. Granade, "Efficient Bayesian phase estimation," *Phys. Rev. Lett.*, vol. 117, no. 1, p. 010503, 2016, arXiv:1508.00869.

[106] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, "Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics," in *STOC*, 2019, pp. 193–204, full version in arXiv:1806.01838.

[107] Y. Dong, L. Lin, and Y. Tong, "Ground-state preparation and energy estimation on early fault-tolerant quantum computers via quantum eigenvalue transformation of unitary matrices," *PRX Quantum*, vol. 3, no. 4, p. 040305, 2022, arXiv:2204.05955.

[108] M. Ajtai, J. Komlós, and E. Szemerédi, "An $o(n \log n)$ sorting network," in *STOC*, ser. STOC '83. New York, NY, USA: Association for Computing Machinery, 1983, p. 1–9.

[109] R. Beals, S. Brierley, O. Gray, A. W. Harrow, S. Kutin, N. Linden, D. Shepherd, and M. Stather, "Efficient distributed quantum computing," *Proc. R. Soc. A*, vol. 469, no. 2153, p. 20120686, 2013, arXiv:1207.2307.

[110] B. D. Clader, A. M. Dalzell, N. Stamatopoulos, G. Salton, M. Berta, and W. J. Zeng, "Quantum resources required to block-encode a matrix of classical data," *IEEE Trans. Quantum Eng.*, vol. 3, pp. 1–23, 2022, arXiv:2206.03505.

[111] L. Lin, "Lecture notes on quantum algorithms for scientific computation," 2022, arXiv:2201.08309.

[112] E. Tang, "A quantum-inspired classical algorithm for recommendation systems," in *STOC*, 2019, pp. 217–228, arXiv:1807.04271.

[113] ——, "Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions," *Phys. Rev. Lett.*, vol. 127, no. 6, p. 060503, 2021, arXiv:1811.00414.

[114] N.-H. Chia, A. P. Gilyén, T. Li, H.-H. Lin, E. Tang, and C. Wang, "Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning," *J. ACM*, vol. 69, no. 5, pp. 1–72, 2022, earlier version in *STOC'20*. arXiv:1910.06151.

[115] C. Shao and A. Montanaro, "Faster quantum-inspired algorithms for solving linear systems," *ACM Trans. Quantum Comput.*, vol. 3, no. 4, 2022, arXiv:2103.10309.

[116] E. Tang, "Dequantizing algorithms to understand quantum advantage in machine learning," *Nat. Rev. Phys.*, vol. 4, no. 11, pp. 692–693, 2022.

[117] ——, "Quantum machine learning without any quantum," Ph.D. dissertation, University of Washington, 2023.

[118] H. Yamasaki, S. Subramanian, S. Sonoda, and M. Koashi, "Learning with optimized random features: Exponential speedup by quantum machine learning without sparsity and low-rank assumptions," in *NeurIPS*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 13 674–13 687, arXiv:2004.10756.

[119] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Natl. Bur. Stand.*, vol. 49, no. 6, pp. 409–435, 1952.

[120] W. Hackbusch, *Iterative solution of large sparse systems of equations*, 2nd ed. Springer, 2016, vol. 95.

[121] D. Orsucci and V. Dunjko, "On solving classes of positive-definite quantum linear systems with quadratically improved runtime in the condition number," *Quantum*, vol. 5, p. 573, 11 2021, arXiv:2101.11868.

[122] C. Gidney, "Windowed quantum arithmetic," 2019, arXiv:1905.07682.

[123] C. Gidney and M. Ekerå, "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits," *Quantum*, vol. 5, p. 433, 4 2021, arXiv:1905.09749.

[124] T. Häner, S. Jaques, M. Naehrig, M. Roetteler, and M. Soeken, "Improved quantum circuits for elliptic curve discrete logarithms," in *PQCrypto*, J. Ding and J.-P. Tillich, Eds. Cham: Springer International Publishing, 2020, pp. 425–444, arXiv:2001.09580.

[125] D. Litinski, "How to compute a 256-bit elliptic curve private key with only 50 million Toffoli gates," 2023, arXiv:2306.08585.

[126] D. W. Berry, C. Gidney, M. Motta, J. R. McClean, and R. Babbush, "Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization," *Quantum*, vol. 3, p. 208, 12 2019, arXiv:1902.02134.

[127] V. von Burg, G. H. Low, T. Häner, D. S. Steiger, M. Reiher, M. Roetteler, and M. Troyer, "Quantum computing enhanced computational catalysis," *Phys. Rev. Res.*, vol. 3, no. 3, p. 033055, 2021, arXiv:2007.14460.

[128] J. Lee, D. W. Berry, C. Gidney, W. J. Huggins, J. R. McClean, N. Wiebe, and R. Babbush, "Even more efficient quantum computations of chemistry through tensor hypercontraction," *PRX Quantum*, vol. 2, p. 030305, 7 2021, arXiv:2011.03494.

[129] I. H. Kim, Y.-H. Liu, S. Pallister, W. Pol, S. Roberts, and E. Lee, "Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules," *Phys. Rev. Res.*, vol. 4, p. 023019, 4 2022, arXiv:2104.10653.