Fast Estimation of Sparse Quantum Noise

Robin Harper,[1,*] Wenjun Yu,[2] and Steven T. Flammia[3]

[1] *Centre for Engineered Quantum Systems, School of Physics, University of Sydney, Sydney, NSW 2006, Australia*
[2] *Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China*
[3] *AWS Center for Quantum Computing, Pasadena, California 91125, USA*

(Received 2 August 2020; revised 22 December 2020; accepted 13 January 2021; published 10 February 2021)

As quantum computers approach the fault-tolerance threshold, diagnosing and characterizing the noise on large-scale quantum devices is increasingly important. One of the most important classes of noise channels is the class of Pauli channels, for reasons of both theoretical tractability and experimental relevance. Here we present a practical algorithm for estimating the s nonzero Pauli error rates in an s-sparse, n-qubit Pauli noise channel, or more generally the s largest Pauli error rates. The algorithm comes with rigorous recovery guarantees and uses only $O(n^2)$ measurements, $O(sn^2)$ classical processing time, and Clifford quantum circuits. We experimentally validate a heuristic version of the algorithm that uses simplified Clifford circuits on data from an IBM 14-qubit superconducting device and our open-source implementation. These data show that accurate and precise estimation of the probability of arbitrary-weight Pauli errors is possible even when the signal is 2 orders of magnitude below the measurement noise floor.

DOI: 10.1103/PRXQuantum.2.010322

I. INTRODUCTION

Estimating noise in quantum computers is becoming increasingly important as we begin to test quantum error correction (QEC) on current noisy intermediate-scale devices [1]. Much of the current effort in noise estimation is focused on identifying methods that will remain tractable as the system size increases beyond the few qubit regime [2–15]. In such larger systems it is important to identify not only the errors that occur when qubits are operated in isolation or in small groups but also the additional errors that occur when the device is implementing fault-tolerant QEC circuits and nontrivial quantum algorithms. If we are able to characterize the noise and noise types (such as control errors, decoherence, and crosstalk errors) in such a system then that will allow us to better diagnose and fix such errors, for instance by enabling calibration in the presence of crosstalk. Characterization of the noise will also allow the construction of tailored quantum error-correcting codes and decoders and customized fault-tolerance protocols designed to counteract the specific noise in the system. Such bespoke systems have been shown to outperform their generic counterparts at quantum error correction [16–21].

Noise estimation is possible in principle using quantum process tomography [22], but in practice this is often not desirable for several reasons. First, even using methods such as compressed sensing [23–28], the enormous Hilbert space of a multiqubit machine makes it difficult to efficiently estimate all possible parameters beyond a handful of qubits. Second, standard tomography protocols are susceptible to state preparation and measurement (SPAM) errors [29], which limit the accuracy in estimating noise in quantum gates.

One promising approach to make noise characterization more tractable is to reduce the noise to a smaller set of relevant parameters that can be estimated in a SPAM-free way. A natural candidate for this approach is to learn the *Pauli projection* of a quantum noise channel. This is the channel obtained when the noise channel is twirled over the set of n-qubit Pauli operators. The remaining parameters of the channel, known as the Pauli error rates, are the most relevant parameters for near-term applications of QEC and fault tolerance because of the dominant role played by stabilizer codes [30]. Moreover, practical methodologies have been developed to implement the Pauli projection without substantially changing the average error rate in a given round of gates [31–33]. Furthermore, QEC tends to make noise less coherent [34–36], which further justifies the Pauli approximation at the logical level. Finally, Pauli error rates can be learned in a SPAM-free way [37,38].

[*] robin.harper@sydney.edu.au

Focusing on Pauli channels reduces the number of parameters required for complete noise estimation to 4^n, where n is the number of qubits of the device. Although this has better scaling than other SPAM-robust methods that attempt to learn an entire noise channel (e.g., Refs. [39,40]), this is unfortunately already too large to be tractable for some present-day quantum devices [41]. There are several ways to try to reduce this parameter count even further while still capturing the most relevant parameters for fault tolerance and QEC. For example, when the Pauli error rates form a bounded-degree Markov field, then the channel can be learned efficiently in n [37]; this algorithm was experimentally validated in Ref. [38]. Reference [37] also gave an efficient algorithm for estimating the class of s-sparse Pauli channels, i.e., those with at most s nonzero Pauli error rates.

These two classes of Pauli channels are motivated by the fact that quantum devices approaching the fault-tolerant regime will have very few significant errors (and therefore are approximately sparse) and will have errors that are only weakly correlated (and therefore are approximated by a low-degree Markov field).

However, the algorithm for estimating the class of s-sparse Pauli channels discussed in Ref. [37] required a prohibitively large number of samples in practice, and could only guarantee recovery for $O(\epsilon^{-4})$ samples. In order to recover Pauli error rates as large as 10^{-3}, over 10^{12} measurements are required, rendering the algorithm impractical on modern noisy, intermediate-scale quantum (NISQ) devices. As we discuss shortly the scaling for the present algorithm is orders of magnitude better in every regime of interest.

A. Main results

In this paper, we give a new algorithm for estimating s-sparse Pauli channels that is distinct from Ref. [37]. This algorithm can reconstruct an s-sparse Pauli channel with a recovery guarantee that is much more efficient than any existing algorithm. We note that for a Pauli channel each Pauli is an eigenvector of the channel, the respective eigenvalue representing how faithfully that Pauli is transmitted through the channel [see Eq. (4) and related discussion]. Then we first assume that an experiment can be modeled as having access to a noisy oracle that can return an eigenvalue of an unknown Pauli channel with some independent Gaussian noise with variance ξ^2. Then using at most $O(sn)$ queries to the noisy oracle, the algorithm returns s estimated error rates $\hat{p}_j$ that agree with the channel error rates p_j with precision $|\hat{p}_j - p_j| \leq O\left(\xi/\sqrt{s}\right)$. In fact, the bound is slightly stronger than this. The precise statement is given in Theorem 1, together with Assumption 1, which lay out the precise mathematical assumptions used in the derivation.

We then show how to replace the abstract Pauli eigenvalue oracle with a concrete experimental procedure that uses only Clifford quantum circuits and computational basis measurements, detailing the process to determine the required information and how to perform the entire estimation efficiently. We show that noisy eigenvalues can be estimated to within variance ξ^2 by using only Clifford quantum circuits and computational basis measurements. Our results use modifications of the algorithm from Ref. [37] and show how the relevant noisy eigenvalue queries can be obtained with only $O\left(n^2/\xi^2\right)$ measurements.

Next, we validate these algorithms using experimental data from a 14-qubit superconducting device [38]. The original experiment exhaustively estimated the averaged eigenvalues in this device. We use these data to construct our eigenvalue oracle. We then simulate various levels of measurement noise on top of this "true" experimental signal to validate our algorithms. Our results are depicted in Fig. 1. We show that when the noise added to the eigenvalues has any standard deviation in the range of 10^{-3}–10^{-5} then we can accurately recover Pauli error rates as small as *2 orders of magnitude* less than the noise added on the eigenvalues. Importantly, even when we artificially add arbitrary many-body Pauli errors with comparable error probabilities, we still recover these strongly correlated errors with high relative precision.

Our results suggest that practical characterization of all Pauli error rates with probabilities greater than 10^{-4} or 10^{-5} in a quantum device with 10–20 qubits can be achieved with around 10^6 or 10^7 experimental measurements. In such quantum devices having submicrosecond gate times and submillisecond state preparation and readout times, this puts practical noise characterization within reach on a time scale of hours, not days or weeks.

Finally, we write open-source code, available on GitHub [42], which reproduces all the figures in this paper and contains other examples, which explain how to use the algorithms in real experiments.

The remainder of this paper is organized as follows. We provide some notation and background in Sec. II followed by an intuitive overview of our recovery algorithm in Sec. III. We state our precise recovery guarantees in Sec. IV. We describe the circuits we use for practical eigenvalue estimation and provide details of our validation results in Secs. VI and V. We defer the precise definition of the algorithm until Sec. VII and the proofs until Secs. VIII and IX. We conclude in Sec. X.

II. NOTATION AND BACKGROUND

Given a set of n qubits with Hilbert space dimension 2^n, we can introduce the following notation. Let $\mathcal{P}^n$ denote the group of Pauli operators on all n qubits and $\mathbf{P}^n = \mathcal{P}^n/\langle i \rangle$ be the Pauli modulo phase. There is a natural isomorphism

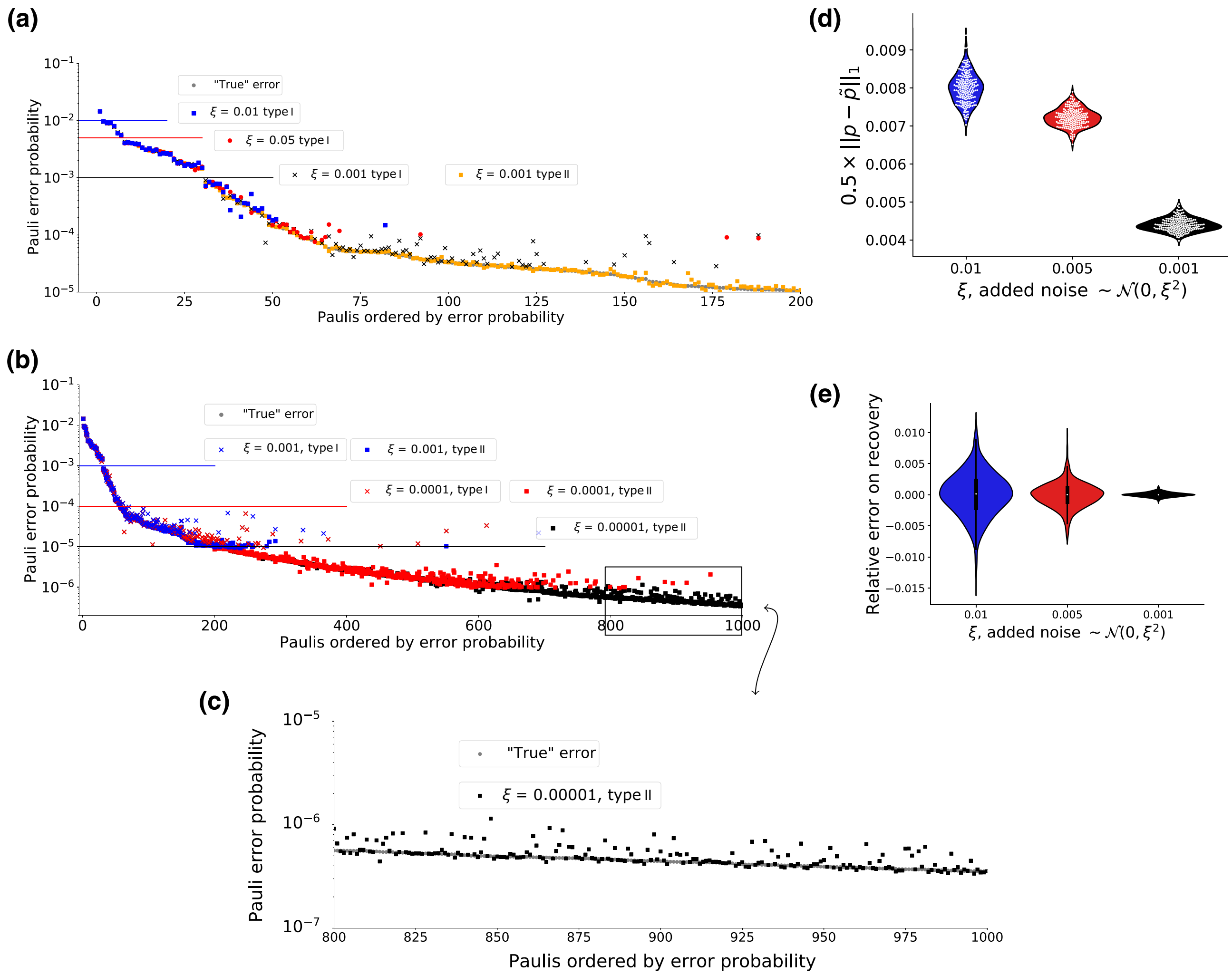

FIG. 1. (a) This figure shows the ability of the reconstruction algorithms to recover sparse Pauli error rates from experimental data. The "actual" error rates (barely visible beneath the data points representing the estimated error rates) are constructed using data from a 14-qubit experiment [38], as described in the main text (Sec. V). Dots indicate recovered Pauli error rates using our algorithm with artificially added zero mean normally distributed noise (variance ξ^2) on top of the true error rates to simulate finite sampling and other noise sources. The reconstruction used two experimental designs using a number of randomized benchmarking style experiments: type I (see Sec. V) used 58 and type II (see Sec. V A) used 365 such experiments, each with the same number of samples per experiment. The type-II experimental design runs more experiments and therefore takes more data overall, but allows recovery of an increasing number of Paulis while keeping constant the number of measurements per experiment. (b) The recovery of 1000 different error rates as low as 10^{-7} with high relative precision, when the experimental noise varies between $\xi = 10^{-3}$–10^{-5}. Notably, the error rates are recovered with a precision almost 2 orders of magnitude below the standard deviation ξ of the noise added to the signal. (c) a more detailed look at the recovery in the regime between 10^{-6} and 10^{-7} with noise levels of $\xi = 10^{-5}$. (d) Violin plot of the total variational (1-norm) distance between the original probability distribution ($\mathbf{p}$) and the reconstructed probability distribution ($\tilde{\mathbf{p}}$), being $\frac{1}{2}\|\mathbf{p} - \tilde{\mathbf{p}}\|_1$. The charts show the spread of recovery error over 200 different randomly generated samples of noise. As can be seen, the entire probability distributions are consistently recovered to high precision. (e) A separate experiment where four distinct uniformly random many-body Paulis are added to the oracle with error rates chosen randomly from a normal distribution $\mathcal{N}(0.005, 0.001)$. The algorithm is run with this additional signal to test if it can recover these Paulis as well. In all cases the planted Paulis are recovered with small relative error, as shown.

between multiplication on $\mathbf{P}^n$ and bit-wise addition of $2n$-bit strings $\mathbb{F}_2^{2n}$ given by

$$a \in \mathbb{F}_2^{2n}, \quad a \longleftrightarrow P_a = P_{a_x a_z} = i^{a_x \times a_z} X[a_x] Z[a_z], \quad (1)$$

where $a_x, a_z \in \mathbb{F}_2^n$ and X and Z are the standard single-qubit Pauli matrices, and $P_a \in \mathcal{P}^n$ is understood to be a canonical coset representative. Here $X[a_x] = X^{a_{x1}} \otimes \ldots \otimes X^{a_{xn}}$, and similar for $Z[a_z]$. Using this isomorphism, we can directly use $a \in \mathbb{F}_2^{2n}$ to denote the Pauli matrix P_a.

For any two Pauli matrices P_a and P_b, we have $P_aP_b = (-1)^{\langle a,b\rangle}P_bP_a$ where the symplectic inner product

$$\langle a,b\rangle = a_x \times b_z + a_z \times b_x \mod 2 \tag{2}$$

is symmetric and bilinear.

We define a *stabilizer group* S to be a linear subspace of $\mathbb{F}_2^{2n}$ such that for all $a, b \in \mathsf{S}$, $\langle a,b\rangle = 0$. Thus a stabilizer group forms a commuting subgroup of the full Pauli group by the mapping in Eq. (1).

An n-qubit *Pauli channel* $\mathcal{E}$ acting on a quantum state ρ is of the form

$$\mathcal{E}(\rho) = \sum_j p_j P_j \rho P_j, \tag{3}$$

where p_j is the error rate associated with the Pauli operator P_j. The *Pauli error rates* p_j form a probability distribution over all $N = 4^n$ elements of the n-qubit Pauli group modulo phases. These are closely related to, but distinct from, the *Pauli channel eigenvalues*, which are defined as

$$\lambda_j = \frac{1}{2^n}\mathrm{Tr}\left[P_j\mathcal{E}(P_j)\right]. \tag{4}$$

Because it is clear from the context, we often refer to these simply as the "error rates" and the "eigenvalues." Thus, when a state ρ is subjected to the noisy channel $\mathcal{E}$, the error rate p_j describes the probability of a multiqubit Pauli error P_j affecting the system. In contrast, the eigenvalues describe how faithfully a given multispin Pauli operator is transmitted through the channel. The error rates p_j and eigenvalues λ_j are related by a *Walsh-Hadamard transform* (WHT). From Eqs. (3) and (4) and the orthogonality relations of the Pauli group, we can compute the Walsh-Hadamard transform coefficients:

$$\lambda_k = \sum_{j\in\mathbb{F}_2^{2n}} (-1)^{\langle k,j\rangle} p_j. \tag{5}$$

The symmetrical nature of the Walsh-Hadamard transform means we also have the inverse relation:

$$p_j = \frac{1}{N}\sum_{k\in\mathbb{F}_2^{2n}} (-1)^{\langle j,k\rangle}\lambda_k. \tag{6}$$

Note that our WHT is ordered by Pauli commutation relations—see Appendix for a further discussion of this subtlety. Finally, for any natural number N, we then write $[N]$ to mean $\{0,\ldots,N-1\}$.

In an analogy with discrete Fourier transforms, the error rates can be thought of as the frequency domain components of the time domain signal, which in this case is the eigenvalues. Our goal is to sparsely sample the dense time-domain signal (the eigenvalues) and reconstruct the entire (but sparse) frequency domain (the error rates). The theory of compressed sensing allows us to do this in principle with very few measurements, namely $O(s\log N)$. However the standard reconstruction methods either use convex optimization [which requires poly(N) classical computation] or use sampling methods and cannot therefore guarantee exact support recovery [43]. Therefore, unlike in compressed sensing, we must reconstruct the sparse frequency-domain signal using only poly$(s, \log N)$ resources for our algorithm to be considered efficient, which is indeed what we achieve here.

Throughout this paper, we restrict to a sparsity regime with only $s \ll 4^{n/2}$ nonzero error rates, each having probabilities greater than a specified cutoff ϵ_0. (In our proofs, we assume that any error rate less than ϵ_0 is identically zero, although the heuristic algorithm is more forgiving.) This is what we mean when we refer to an s-sparse model. This allows our algorithm to perform in the regime where s is exponential in n. When such an exponential scaling holds, it makes our algorithm inefficient in n, but this is also a relevant regime if we wish to estimate Pauli channels with an extensive entropy. Distributions with extensive entropy will generally require an exponential number of error rates to estimate them with arbitrary accuracy.

Our recovery methodology builds on one of the main results of Ref. [37] and an adaptation of the classical algorithms described in Refs. [44,45]. In Ref. [37], the authors show how to recover all $N = 4^n$ Pauli channel eigenvalues to *relative* precision ϵ using $O\left(\epsilon^{-2}n2^n\right)$ measurements. The recovery of all N eigenvalues would require $2^n + 1$ applications of depth $O(m + n^2/\log n)$ Clifford circuits, or 3^n applications of depth $O(m)$ Clifford circuits. The factor of m in the circuit depth must be large enough to resolve the decay curve of the largest eigenvalue in the channel that we wish to faithfully estimate [46]. While the depth of this algorithm is efficient, the number of distinct circuits required is clearly not scalable in n. A single individual eigenvalue can still be learned to relative precision ϵ using only $O\left(\epsilon^{-2}\right)$ measurements, however. It is the need to sweep through $2^n + 1$ (or more) sets that leads to the factor of $O\left(n2^n\right)$ in the sample complexity.

In Ref. [37], the authors also derived what is essentially a variant of the Kushilevitz-Mansour algorithm [47] for learning decision trees via the Fourier spectrum and applied it to the case of Pauli channels. The idea is to breadth-first search through the marginal Pauli error rates, keeping those with large probability mass and pruning the search tree when the mass is below a threshold. This algorithm is theoretically efficient in s and n, however our numerical experiments using this algorithm suggest that the number of eigenvalues required per recovered error rate will make it difficult to use in practice, at least in its current instantiation and in the relevant regime for quantum-computing applications.

III. ALGORITHM OVERVIEW

The problem of reconstructing a sparse set of Pauli error rates by measuring few eigenvalues is closely related to a classical problem of computing a sparse Walsh-Hadamard transform. This problem was studied by Scheibler *et al.* [44] and later (in the regime of noisy signals) by Li *et al.* [45] by decoding a signal $x \in \mathbb{R}^N$, which contains 2^n points indexed by $j \in \mathbb{F}_2^n$. In our circumstances we are not analyzing the frequency domain of a signal, but rather the global probability distribution of the Pauli error rates in a quantum device and the eigenvalue distribution of the Paulis in a superoperator representation of a Pauli noise channel, so this formalism requires some adaptation.

Given the WHT mapping in Eqs. (5) and (6), the algorithms presented in Refs. [44,45] are broadly applicable, but require some modifications. We note where adjustments have to be made. One major difference is our inability to simultaneously measure noncommuting Pauli operators. Below we give a broad overview of the reconstruction algorithms as applicable to our needs. A complete and rigorous analysis can be found in Sec. VIII, but the main recovery guarantee is stated below in Theorem 1. We first deal with the noiseless case.

The main idea behind the algorithm is to note that each eigenvalue is made up of a linear combination of all the error rates. By subsampling the eigenvalues, we are able to split up the error rates, figuratively creating "bins" of error rates, where each bin contains a linear combination of a smaller number of error rates. Provided that there are sufficient bins, then in the sparse regime most of these bins will only contain a few error rates with weight $\geq \epsilon_0$. Using *aliasing*, we can identify these bins and can therefore evaluate these error rates. This information will allow us to reconstruct all the sparse error rates. With this in mind, the reconstruction algorithm can be broken down into three main steps.

1. Determine the *subsampling* bins and perform the experiments to measure the required eigenvalues.
2. Calculate and measure the *aliased* bins to enable identification of single-Pauli bins (*singletons*) and the Pauli error rates that occupy them.
3. Run a decoder to "*peel back*" singletons, converting multi-Pauli bins to single-Pauli bins and repeat until all error rates are identified.

We describe these three steps in an intuitive manner below and relegate the analysis and proofs to Sec. VII.

Step 1—Subsampling. The intuition behind the first step is that it is possible to sample a specific pattern of eigenvalues that will allow the reconstruction of the global probability vector, but where various probabilities are binned (i.e., added together). For instance, given a global probability vector with $N = 4^n$ values it is possible to rewrite this as a "reduced" vector ($\tilde{\mathbf{p}}$) with $B = 2^b$ values, each value being composed of the summation of N/B of the original global probability values (possibly with signs). In the regime where our sparsity is $s < 2^n$ then we show that with appropriate random sampling a large number of these reduced vector values (which we call *bins*), will be composed of none or one of our sparse Pauli errors, i.e., those with a weight $\geq \epsilon_0$ for a parameter ϵ_0 to be chosen later. In what follows we always choose $B = 2^n$, but we occasionally use the notation $B = 2^b$ (so that $b = n$) to illustrate where a given numerical factor originates from.

Whereas Ref. [44] imagined using specific bit patterns of binary strings to index the requisite eigenvalues to sample, we wish to exploit the ability of a quantum device with independent measurement on each qubit to sample from a bit string of 2^n values. As previously discussed, the protocol in Ref. [37] shows how to measure, to multiplicative precision, the Pauli eigenvalues of 2^n commuting Paulis using one randomized-benchmarking style experiment with n-bit spin measurements at the output. The constraint that the Paulis measured be mutually commuting is exactly the constraint we require for the subsampling to allow us to create the required reduced probability vector $\tilde{\mathbf{p}}$.

Suppose we have a specific stabilizer group S. We postpone how to choose this group until later. We can represent the entire stabilizer group by an $n \times 2n$ binary matrix S whose jth row is the stabilizer generator s_j.

Now let $v \in \mathbb{F}_2^n$ label the elements of the chosen stabilizer group, for example via the mapping $v.S$, where v is thought of as a row vector. Our reduced probability vector $\tilde{\mathbf{p}}$ then consists of B bins each containing a sum of N/B distinct Pauli errors. It is labeled by a string $j \in \mathbb{F}_2^b$ and is given by

$$\tilde{p}_j = \frac{1}{B} \sum_{v \in \mathbb{F}_2^n} \lambda_{v.S} (-1)^{j \cdot v}. \tag{7}$$

The effect of this is that the sampled Pauli eigenvalues from the stabilizer group, when transformed by the Walsh-Hadamard transform, give us B bins each containing a sum of $2^n = N/B$ error rates, many of which will be zero in general.

The binning is chosen in such a way that with high probability there will be a large number of bins that only contain a *single* Pauli error rate with a weight $\geq \epsilon_0$ (the other Pauli errors allocated to that bin being, effectively, zero). This will depend on the size of the bins and the sparsity of the Pauli error rates, and is discussed further in Sec. VII. A simple example of the subsampling and binning idea is shown in Fig. 2.

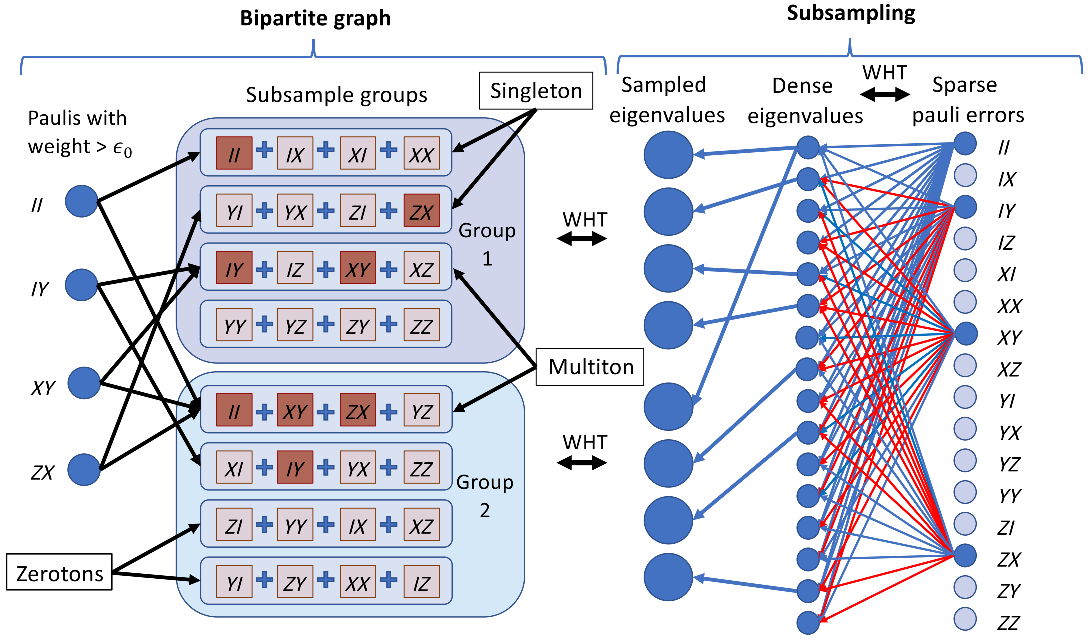

FIG. 2. Illustrative diagram of the bipartite graph that is used to extract information from the subsampling bins. Here we show a simple example for two qubits, where only three nontrivial Paulis (IY, XY, and ZX) have errors. To follow the diagram, we have on the right-hand side all 16 possible Pauli errors. The ones with weight $> \epsilon_0$ are darker colored. The WHT transform combines these Pauli errors, forming the "dense eigenvalues." The color of the arrow indicates whether the Pauli is added or subtracted by the transform—although the detail is not necessary to follow here. The subsampling algorithm then selects two sets of stabilizer groups. In this example the first corresponds to the II, IX, XI, and XX stabilizers, the second to the II, ZY, XZ, and YX stabilizers. These form the two sets of *sampled eigenvalues*. Each of these two groups of four sampled eigenvalues can then be transformed back using a WHT transform yielding four numbers for each group. Each of these four numbers is made up of the sum of four possible Pauli errors, which are indicated inside the inner boxes. For ease of reference the Paulis with weight greater then ϵ_0 are colored red. This is shown in the right-hand side of the "bipartite graph" section of the diagram. To summarize, at this stage, the subsampling algorithm has split the Paulis up as shown (II, IX, XI, and XX stabilizers for group 1 and II, ZY, XZ, and YX for group 2), separating them into bins consisting of singletons, multitons, and zerotons, as described in the text. In this example, it can be seen that the IY Pauli exists as a singleton in group 2, allowing its value to be recovered. It can then be "peeled" from the third bin in group 1 converting that bin from a multiton to a singleton containing just Pauli XY. This then allows Pauli XY to be recovered as well, which would not otherwise be possible, as the signal would conflate with that of IY (if one were looking at group 1 in isolation). Iterative peeling in this fashion will eventually recover all of the nonzero Pauli error rates. Details as to how we determine if a bit is a singleton or multiton are contained in the text. Workbooks in [42] contain example implementations.

So how do we construct our stabilizer group? The most obvious way is to sample a random n-qubit Clifford (see Refs. [48,49] for how to do this). However as n grows past a few qubits, then on current devices the number of single and multiqubit gates required to construct a generic element of the Clifford group requires circuits of depth $O(n^2/\log n)$, and if these circuits are noisy then this will wash out the signal required to estimate the eigenvalues. A better way for current devices is to use a random subset of n-qubit stabilizers that can be formed from a single round of nonoverlapping two-qubit Clifford gates. This has the added advantage of making it trivial to work out how to perform step 2.

Step 2—Aliasing. The question then becomes: how do we detect which bins contain a single Pauli error rate? To do this the reconstruction algorithm uses the shift and modulation property of the WHT. Specifically, if we let $\{p_k\}$ be the WHT of $\{\lambda_m\}$ we have

$$\lambda_{m+n} \xleftrightarrow{\text{WHT}} (-1)^{\langle n,k\rangle} p_k. \tag{8}$$

By taking each element of the stabilizer group and offsetting the sample with a shifting bit pattern (e.g., for four qubits the sample would be offset by the five following bit patterns $[0,0,0,0]$, $[1,0,0,0]$, $[0,1,0,0]$, $[0,0,1,0]$, $[0,0,0,1]$) then the Pauli error rates consigned to that bin are no longer merely summed but rather are added or subtracted depending on whether the inner product of their "bit strings" and the relevant pattern is zero or one. This result is illustrated in more detail in Algorithm 2 and Lemma 1, where we also discuss how to use bit-flip error-detection codes to make the decoding more robust to noise.

This leads to a number of remarkable effects. If the bin is empty (i.e., contains no Pauli error rates with nonzero errors) each of the *offset bins* (i.e., for a particular j each $\tilde{p}_{j,d}, d \in \{2^0 \dots 2^{2n}\}$) will also be zero. If the bin contains only one nonzero Pauli error rate then the magnitude of the sum of each of the offset bins will be constant, and the sign of the sums will identify exactly which Pauli has the nonzero error rate. (For example using the four qubit offsets shown above, if the absolute values of the bins were all 0.001 and the signs of the four offset bins were $(+, -, -, +)$, this could only be caused by a single Pauli error rate of 0.001, with a bit string of 0110.) In every other case, the bin contains multiple Pauli error rates (a *multiton bin*), which leads us to the PEELING decoder (see step 3).

So how can we construct the experiments that will allow us to extract the "shifted" eigenvalues? For instance, one might note that for any particular stabilizer group $\mathcal{S}$, the offset bit pattern applied to each of the elements of the group are unlikely to form a stabilizer group.

It transpires that where we use a stabilizer group created by local two-qubit Cliffords (on each qubit pair), we can do this simply by iterating each distinct qubit pair through four further (different) two-qubit stabilizer patterns. There are five two-qubit stabilizer groups, the union of whose bases form a complete set of mutually unbiased bases; let us label them $S_{1\dots5}^{\otimes 2}$. We set out a specific choice of these groups in detail, together with the two-qubit circuit needed to create them, in Fig. 3. The initial stabilizer is chosen by selecting randomly from $S_{1,2}^{\otimes 2}$ for each qubit pair. This becomes the stabilizer for the purpose of step 1. This circuit is used to conduct the first experiment and extract 2^n Pauli eigenvalues. The offset pattern required for this in step 2 is constructed by iterating over each qubit pair, and replacing the circuit chosen in step 1 with one of the other five (for a total of four further experiments per qubit pair). The total number of experiments required is therefore $2n + 1$. By analyzing each of the experiments formed we will be able to pull out all of the eigenvalues determined by such experiments. Figure 3 shows the circuits used for each experiment and illustrates the method described above.

Step 3—Peeling. If we use a variety of subsampling matrices (that is we repeat steps 1 and 2 for more than one random initial choice of Cliffords) we are now in the position where we have identified a number of Pauli error rates (from bins that contain only one Pauli error rate) and we will also have a number of bins that contain more than one Pauli error rate (multiton bins). In general, for any two stabilizer groups, different Pauli error rates will get hashed into different bins. Where we identify a single Pauli error rate under, say, stabilizer group 1, that same error rate may be in a different bin under stabilizer group 2, a bin it shares with one or more different high weight Paulis (i.e., it may be in a multiton bin under stabilizer group 2). However, because we know the value of this Pauli error rate (since it is a *singleton* under stabilizer group 1), we can remove it from the bin created by stabilizer group 2 by simple subtraction. After this removal, some bins that were previously multiton bins will now become singletons, or at the very least they will be closer to being singleton in that we are left with a bin that now has one fewer Pauli error rate in it. This removal of the value of a previously identified singleton from a different stabilizer group's bin is known as "peeling back" the known values, giving the PEELING decoder its name. The goal is that when we peel back our identified error rates, we create more and more bins that now contain only one Pauli error rate. This can be applied in an iterative fashion. We can then iterate this until we either identify all the Pauli error rates (all the bins are empty) or until we have no further single Pauli error rates to peel back. All of these steps can be viewed in Algorithm 3. In the latter case the reconstruction algorithm has failed, although we will at least know the magnitude of the error rates we have failed to identify, and can perform additional experiments to try to learn them.

A. Dealing with noise

Using the ideas in Ref. [45], we can modify the reconstruction algorithm to handle noise of the form

$$\boldsymbol{\lambda} \rightarrow \boldsymbol{\lambda} + \mathbf{w}, \tag{9}$$

where $\mathbf{w}$ is a Gaussian distributed noise vector, $\mathbf{w} \sim \mathcal{N}(0, \xi^2 \mathbb{1})$. It is only for simplicity in the proof that we consider the isotropic case, and small dependencies and correlations do not substantially affect the observed numerical performance.

In our case, the noise arises as the estimation error in our eigenvalues caused by finite sampling. These finite sampling errors occur because of the limited number of random sequences and measurement shots per sequence occurring when the eigenvalue estimation experiments are carried out. Errors of this nature have been analyzed in Ref. [37]. To reduce noise, the number of sequences and shots per experiment needs to be increased, and this sample complexity was also bounded in Ref. [37]. In the relevant regime of high precision, the estimation error on the eigenvalues will be approximately normally distributed, and empirical estimates of the variance and covariance can be determined by bootstrapping from the observed measurement outcomes [38].

The PEELING decoder only requires two adjustments to account for such noise: the zero-Pauli verification and the single-Pauli search protocols.

For the former, in the noiseless model we identify a bin as being empty if the value of the bin (and each of the offset bins) is zero. Where we have noise, we simply relax the requirement that the bins are exactly equal to zero before identifying them as empty. We can bound an acceptable

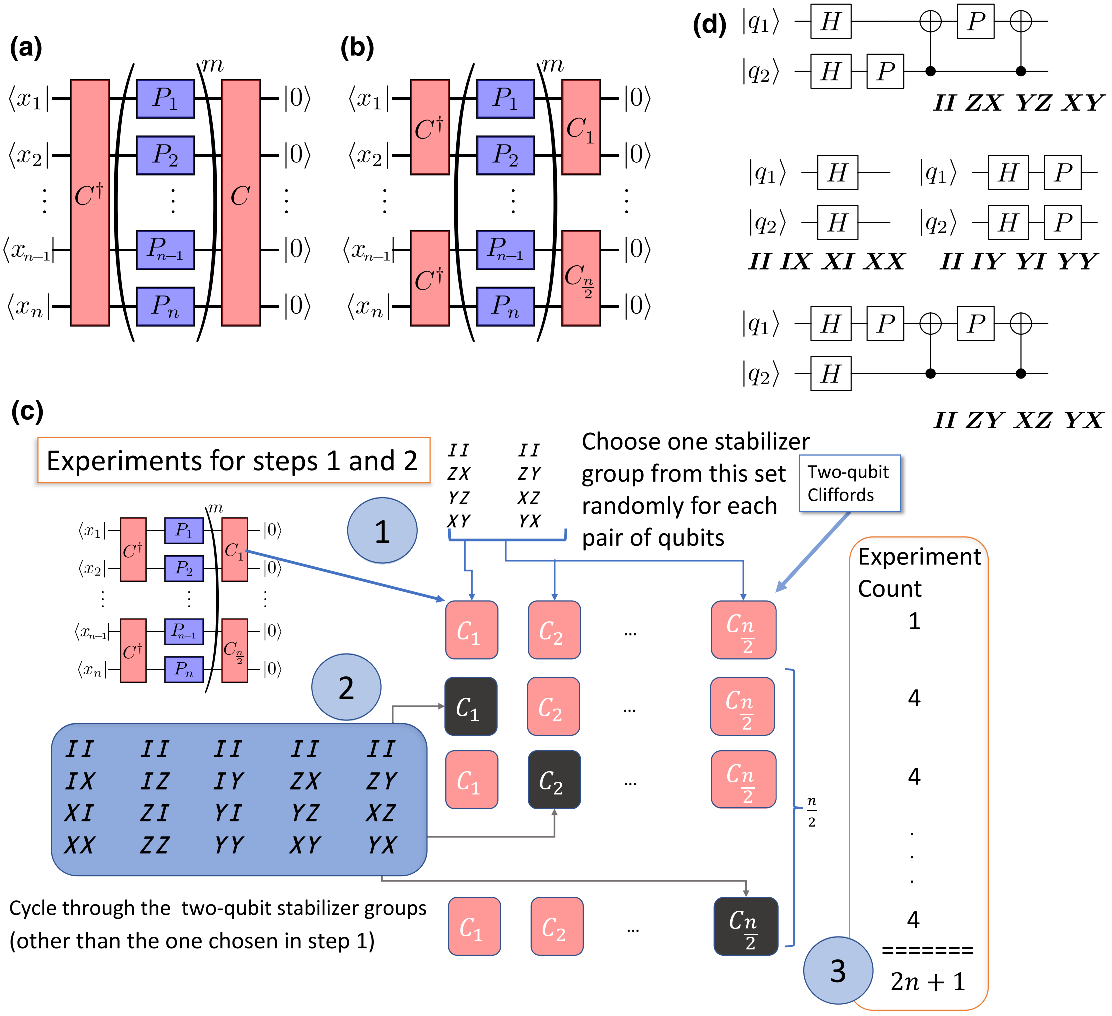

FIG. 3. (a) The type of circuit described in Ref. [37] that allows the recovery of 2^n eigenvalues of the averaged noise channel of the device. The random Pauli gates (blue) are used to twirl the channel and by averaging over a number of random choices of Pauli gates, the noise channel in the device is transformed into a Pauli channel. In a similar way to randomized benchmarking, by repeating the twirl for a certain number of Pauli gates (m) and then returning the system into the computation basis by choosing the Pauli inverting the twirl, a decay curve will be induced and this can then be fit to determine the eigenvalues independently of state preparation and measurement errors. The single n-qubit Clifford (and its inverse at the end) determines which of the 4^n Paulis are sampled and an appropriate Clifford can be used to select any n-qubit stabilizer set. (b) A further modification of the circuit, where instead of using a generic n-qubit Clifford only two-qubit Cliffords are used. (Where the device has an odd-number of qubits, a single Clifford can be used on one of the qubits.) As discussed in the text, for each chosen value of m the circuit is repeated for multiple sequences with different randomly chosen Paulis, but for fixed Cliffords. Collectively each of the runs for multiple choices of Paulis carried out over several different lengths of m are defined as *an experiment*. (c) How, once an experiment has been chosen in step 1 of the procedure, further experiments are created in order to determine the offsets required to identify the Pauli (see text). As shown in step 2, each of the two qubit Cliffords needs to be cycled sequentially through the four other two-qubit stabilizer groups (i.e., the four that are different from the initial choice). This means that for each sequence in step 1, a further $2n$ experiment needs to be performed, leading to $2n+1$ experiments per chosen stabilizer group. For the second group an offset of one qubit should be chosen, meaning the local stabilizer groups now span different qubit pairs. Simple Pauli twirls can be carried out on any odd or isolated qubits. (d) Some example subcircuits required to perform the transform into the local stabilizer group listed in (c)—step 2. The inverse gate will be of a similar form.

small value as indicating an empty bin, given the number of “noisy” zeros in the bin and our estimate of the noise variance. This will lead to a noise floor of Pauli error weights we can recover. That is, we are unlikely to recover those Pauli errors with a value so small they are swamped by the noise in the bins. This is an inevitable consequence of the noise.

The latter case of single Pauli identification has two aspects that need to be considered. The first is “does the bin contain only a single Pauli?”, and the second is “if so, which Pauli?". For a noisy version the first question is dealt with the same way as the noisy zero, i.e., we require only the magnitudes of the offset bins to match to within some estimated noise window. While this runs the risk of not noticing some small Pauli error rates that are also in the bin, it appears to work well in practice. The second is more akin to a noisy bit-flip channel, in that the noise may cause us to incorrectly identify a "1" as a zero or vice versa. (This is more likely when the noise is commensurate with or greater than the Pauli error weight.) One simple method of dealing with this is to repeat sample with different offsets, and then take a majority vote, however our numerical simulations do not suggest that this is necessary. Finally we can use a number of random offsets and some additional fixed offsets chosen in such a way they form a classical error-correction code to further protect the algorithm from noise. When an appropriate classical code is chosen this does not alter the sample complexity scaling, though it does increase slightly the number of experiments. It also comes with a robust recovery guarantee as described in the next section and Sec. VII.

IV. RECOVERY GUARANTEE FROM NOISY EIGENVALUES

Using the algorithm illustrated above and leveraging some proofs contained in Ref. [45], we can construct the following recovery guarantee that relates our ability to recover Pauli error rates with bounded error to the noise in the estimated Pauli eigenvalues. The intuition behind the guarantee is that by increasing the number of offset observations we can reduce the chance of incorrectly detecting whether the bin occupancy is zero, or one, or more than one. If the bin detection succeeds, then the peeling step will succeed with high probability for appropriate choices of the subsampling and aliasing designs.

Our recovery guarantee does however rely on several assumptions, which we now state explicitly.

Assumptions 1. *Let* $\mathbf{p} \in \mathbb{R}^N$ *be the target Pauli error rates with support* $\mathcal{K} = \mathrm{supp}(\mathbf{p})$ *and sparsity* $s = |\mathcal{K}|$.

A1 *(Random sparse support.) The support set* $\mathcal{K}$ *is chosen uniformly at random from all subsets of* $[N]$ *of size exactly s, where* $s = 4^{\delta n}$ *is sublinear in the dimension* $N = 4^n$ *for some* $0 < \delta < 1/2$.

A2 *(Independent Gaussian noise.) Each queried Pauli eigenvalue* λ_j *has noise given by independent Gaussian noise centered around the eigenvalue with variance* ξ^2.

A3 *(Good signal to noise.) Each error rate* p_m *for* $m \in \mathcal{K}$ *is lower bounded by* $p_m \geq \epsilon_0$ *for some* $\epsilon_0 > 0$, *the eigenvalue noise variance is upper bounded as* $\xi^2 \leq \min(\frac{B}{s^2}, 1)$, *and the two are related via* $\epsilon_0 \geq 2\xi/\sqrt{B}$. *Here B is the number of bins in a single subsampling group.*

Our main theorem is then the following.

Theorem 1. *Suppose Assumptions 1 hold for an unknown Pauli channel with eigenvalues* $\boldsymbol{\lambda}$ *and error rates* $\mathbf{p}$. *Then with failure probability* $\mathbb{P}_F \leq e^{-O(n)}$, *Algorithms 2–4 estimate the s-sparse Pauli error rates* $\widehat{\mathbf{p}}$ *such that* $\|\widehat{\mathbf{p}} - \mathbf{p}\|_\infty \leq 2\xi/\sqrt{B}$ *using* $O(sn)$ *eigenvalue queries and* $O(sn^2)$*-time classical computation.*

Proof. The proof is given in Sec. VIII. ■

Note that our main theorem references a noisy eigenvalue oracle rather than a direct sample complexity for estimating the eigenvalues. From Ref. [37], $O(sn)$ queries to the eigenvalue oracle can be approximated to within variance ξ^2 using only $O\left(n^2/\xi^2\right)$ samples. While a variant of the protocol in Ref. [37] can make the noise independent, it will not be exactly isotropic Gaussian noise, so we can only heuristically claim this as the sample complexity. This is why we state the formal main result in terms of query complexity.

It is worth remarking on the strength of the assumptions that go into the statement of the theorem. Assumption A1 is mathematically convenient, but is certainly too strong physically since most errors in near-term quantum devices are likely to have low weight. This could in principle be compensated by incorporating a randomizing permutation into the experimental design. However, our experiments (see the next section) do not seem to require such a compensation for convergence. Assumption A2 is again mathematically convenient, and it will only ever be approximately true in practice. We believe that other error models with weak correlations and bounded variance will have similar guarantees, but an analysis of this would introduce significant complications without elucidating anything about the algorithm. Weakening A2 in this way would be interesting future work, as it would let us make direct formal statements about the sample complexity. As for our final assumption, a signal-to-noise assumption along the lines of Assumption A3 seems to be a mathematical necessity for convergence. However, it may be possible that a guarantee could still be proven with a smaller signal-to-noise ratio or with weaker restrictions on ϵ_0 and ξ. For example, a simple corollary of our result is a guarantee in the total variation distance (1-norm) such that $\frac{1}{2}\|\widehat{\mathbf{p}} - \mathbf{p}\|_1 \leq s\xi/\sqrt{B}$, which is nontrivial exactly when $\xi < \sqrt{B}/s$ (cf. A3). It might be easier (and more natural) to directly prove this implication of our result, or it may be possible to prove this using weaker assumptions.

V. EXPERIMENTAL VALIDATION

To validate our algorithm we use data extracted from a 14-qubit superconducting device build by IBM. In Ref. [38] the complete distribution of locally averaged Pauli error rates in the device was estimated. In this work, we recycle the data from that experiment to validate our new algorithms.

The data set from Ref. [38] consists of 2^{14} *locally averaged* eigenvalue estimates, meaning that each eigenvalue is labeled by a 14-bit string that labels the presence or absence of a nontrivial Pauli on each corresponding qubit. This is in contrast to the full eigenvalues, each of which would require a 28-bit label and could additionally resolve the entire set of 2^{28} Pauli eigenvalues, without local averaging. Although we could run our algorithm on the 2^{14} locally averaged eigenvalues, to make it more challenging we look at random self-consistent extrapolations of the data onto the full set of $2^{28} \approx 2.7 \times 10^8$ eigenvalues.

The random interpolation proceeds as follows. From the estimated eigenvalues of Ref. [38], we reconstruct the locally averaged error rates. (In fact, this step was already done in Ref. [38].) For each locally averaged error rate, we pick a uniformly random point in the probability simplex of the Paulis supporting the local average. This defines a new probability distribution on the full set of 2^{28} Paulis. Every such extrapolation has the property that locally averaging it will return the original experimentally observed data. We construct a "true" set of known Pauli channel eigenvalues by transforming (using the Walsh-Hadamard transform) on these extrapolated error rates. This gives us a family of experimentally derived eigenvalue oracles that we can use to validate our numerical reconstructions.

The data from Ref. [38] have a "no-error" probability of about 0.86, and upon extrapolation they have approximately 200 Paulis with an error rate above 10^{-5}, about 600 above 10^{-6}, and about 2000 above 10^{-8}. Although the original estimation cannot resolve error rates as small as 10^{-8} with meaningful error bars, our eigenvalue oracle still has access to these numbers as part of the simulation. For this discussion, we focus on reconstructing errors in the regime above 10^{-5}, as these are the most relevant. This corresponds to a sparsity $s = 4^{\delta n}$ with roughly $\delta \approx \frac{1}{4}$.

As can be seen from Fig. 1, the sparse recovery protocol performs well in this regime ($\delta \lesssim 0.25$), requiring only a fraction of the eigenvalues that would be required for a full recovery of all Pauli error rates. The limiting factor in this regime is the noise in the oracle, which equates directly to the number of measurements and sequences sampled as part of the original experiment (see Ref. [37] for relevant reconstruction guarantees). It appears that the effect of the protocol is to allow recovery of the Pauli error rates to (approximately) an order of magnitude or more *less* than the noise in the oracle.

Importantly, if a device has unexpected many-body correlations (for example through unexpected qubit interactions or crosstalk), then we should also be able to find these errors whenever their probability is above our noise floor. We validate this feature of the algorithms as well by injecting known high-weight Pauli errors into the oracle. Our algorithm reveals and evaluates such Pauli error rates to a high degree of relative accuracy, as shown in Fig. 1(e).

Section V A discusses the regime where ($\delta \geq 0.25$). In that case continued recovery of Paulis with low error rates requires some changes to the local stabilizer groups used (or a switch to global random stabilizer groups).

A. Experiments in the regime $\frac{1}{4} < \delta < \frac{1}{2}$

While the experimental protocol presented above is likely to be all that is required in most practical regimes, if the number of Paulis to be recovered is large then a slight modification might be needed. Unlike the situation where one is using completely random stabilizer groups, the local stabilizer protocol can fail when trying to reconstruct many low-error Paulis that differ only in one or two Paulis, in such a way that they cannot be separated by the local stabilizers. This might occur, for instance, in the regime where $\delta > 0.25$. In such circumstances, one can cycle each distinct set of two qubit pairs through the five stabilizer groups identified in Fig. 3(c), and then generate the offset bins for each of them. The number of experiments that need to be performed are the original experiment (1), then a further four for each qubit pair (4), times the number of qubit pairs ($n/2$), times the number of experiments needed to generate the offsets on the remaining $n-2$ qubits [$4(n-2)$], for a total of $1 + 8n(n-2) = O(n^2)$ total experiments. The eigenvalues gathered this way allow the creation of $n/2$ properly offset subsampling matrices of 2^{n+2} bins each containing 2^{n-2} Paulis. Empirically, this appears to be sufficient to exactly recreate the global probabilities up to $\delta = 0.5$. Figure 1(b) illustrates the extra recovery power available in the highest precision regime.

VI. HEURISTIC NOISE RECONSTRUCTION

Here we describe in more detail the intuition behind the algorithm, the experiments prescribed and a simplified, practical extraction algorithm. Our GitHub repository [42] contains code and examples showing how the algorithms can be used to recreate the figures in this paper.

A. Determining a suitable number of subsampling groups

Our proofs relating to the recovery of s-sparse Pauli errors require an assumption that each element in the support set $\mathcal{K}$ is chosen independently and uniformly at random from $[N]$. At first glance it may appear that this is not likely to be the case in a quantum device as the Pauli errors are likely to cluster around low-weight Pauli errors rather than be uniformly distributed over the 4^n different possible Pauli errors. However where we choose random n-qubit stabilizers (global stabilizer groups) as the basis for sampling the Pauli eigenvalues, this effectively randomizes the bin into which we consign any specific Pauli error rate, which (empirically) allows us to satisfy the uniformly random distribution requirement.

Given this we can continue to use the "balls-and-bins" model utilized in Ref. [45, Appendix B]. We can use this insight, together with our ability to simultaneously sample 2^n commuting eigenvalues, to determine practical values for the number of subsampling groups C given our bin size of $B = 2^n$.

Since the sparsity $s \ll N$, we have that the expected number of Paulis (balls) in one bin will be $s/N \times N/B = s/B$, which in the sparsity regime of interest will be < 2. Assuming we have an s-sparse distribution, with $B = 2^n$ being the number of bins sampled, we define the sparsity coefficient $\eta = B/s$, which is at least 1. This means that we require only C, the number subsampling groups, to be 2 in order to recover all of the edges in $O(s)$ iterations with probability at least $1 - O(1/s)$ (see [45, Appendix B]). It can then be seen that as each experimental run recovers 2^n eigenvalues, we need to perform at least one experimental run for each bin plus one for each of the offsets of the bin times the number of subsampling groups. This means that the minimum number of experimental runs is $2(2n + 1)$. As we discuss later, by increasing the number of offsets we can increase the recovery guarantees, but at the cost of sampling more eigenvalues (although still only scaling proportional to n).

In the case where our device is too large for $\eta \geq 1$, for instance where we have had to marginalize over the measurements as $\log_2(B) \geq 30$, then we can increase our effective C by marginalizing over randomly chosen qubits and creating our subsampling matrices from such randomly chosen subsamples of the measurement outcomes. This will allow us to retain the recovery guarantees without increasing the number of experiments on the device, although this incurs an increased computational cost in setting up and performing the peeling decoder.

B. Heuristic algorithm using local circuits

Our numerical simulations based on the data collected in the experiments from Ref. [38] indicate that randomly chosen global stabilizers are not in fact necessary to distribute the Pauli errors widely enough to allow recovery. It appears that *local* stabilizer groups suffice. This allows us to dramatically reduce circuit complexity while keeping the number of experiments required to a minimum. Figure 3(c) details the local two-qubit stabilizer groups that can be selected to perform an extraction experiment that is viable on most current devices. In Fig. 3(d) we show the local Clifford circuits that can be used (with their corresponding inverse) at the beginning (end) of the measurement circuits to create these local stabilizer groups. In Ref. [37] it was shown that using such circuits we can estimate 2^n Pauli eigenvalues with relative precision ϵ, using $O(\epsilon^{-2}n)$ measurements.

Having chosen the series of stabilizers to measure, together with the circuits for offsets, we will have all the relevant eigenvalues required to use the noisy peeling decoder.

In Algorithm 1 we show how to operate the decoder on a practical level, assuming that there are a large number

```
Require: M ← Paulis in groups ($C$ sets),   ▷ Figure 3
Require: $\lambda_{\Psi_{l,c}}$ for $l \in M_c$ $c \in [C]$,   ▷ Eigenvalues
Require: $\lambda_{(\Psi_{l,c} \oplus b)}$ for $l \in M_c$ $c \in [C]$,$b \in [2^{2n}]$   ▷ Offsets
Require: $\delta_z, \delta_s$   ▷ Relaxation parameters (see text)
$Z_s \leftarrow$ initial zero sensitivity
$S_s \leftarrow$ initial singleton sensitivity
$\mathcal{P} \leftarrow$ initialize empty list of Paulis + errors
    ▷ Set up quasi probability ‘bins’.
for c=1, ..., $C$ do
    $\tilde{p}_{j,c} \leftarrow \frac{1}{2^n} \sum_{l \in [2^n]} \lambda_{\Psi_{l,c}} (-1)^{\langle j,l \rangle}, j \in \mathbb{F}_2^n$   ▷ Equation (7)
    $\tilde{p}_{j,c,b} \leftarrow \frac{1}{2^n} \sum_{l \in [2^n]} \lambda_{(\Psi_{l,c} \oplus b)} (-1)^{\langle j,l \rangle}, j \in \mathbb{F}_2^n, b \in 2^{[2n]}$
end for
    ▷ Populate $\mathcal{P}$ with singletons.
for _ = 1, 2, ..., arbitrary do
    for c=1, ..., $C$ do
        for $\tilde{p} \in [\tilde{p}_{j,c}, \tilde{p}_{j,c,b}], j \in \mathbb{F}_2^n, b \in 2^{[2n]}$ do
            if not IsCloseToZero($\tilde{p}, Z_s$) then
                if IsSingleton($\tilde{p}, S_s$) then
                    (P,E) ← SingletonPauliAndSize($\tilde{p}$)
                    $\mathcal{P}$ ←(P,E)
                        ▷ Remove from other sets
                    PeelBack(M$_{[C]/c}$,(P,E))
                end if
            end if
            if $\sum \mathcal{P} \approx 1$ then
                return $\mathcal{P}$   ▷ Success!
            end if
        end for
    end for
    if no new Paulis added since previous iteration then
        ▷ Relax search requirements.
        $Z_s \leftarrow Z_s - \delta_z$
        $S_s \leftarrow S_s + \delta_s$
    end if
end for
return Incomplete $\mathcal{P}$   ▷ !Success
```

Algorithm 1. Noisy peeling decoder.

of Paulis sitting below the level of interest (i.e., with an error rate $\ll \epsilon_0$). To understand how it works one should note that there are two main components to dealing with the noise. The first is when deciding if the bin is zero, i.e., when the only values in the bin and its offsets are noise. We initially start willing to assume this is the case and slowly become less willing (by δ_z) to accept that the bin is really zero as we start to try and recover smaller and smaller error rates. This means that initially the decoder will concentrate only on bins that have relatively large Pauli errors in them and will be less likely to mistake noise as indicative of an error.

For instance, assuming a reconstruction error normally distributed with a standard deviation of 0.01, then if a bin contained 2^{14} 0-error Paulis with such noise, we would expect the mean of such a bin to be centered around 0, with a standard deviation of $\sqrt{(}0.01^2/2^{14}) \approx 7.8 \times 10^{-5}$. Therefore, allowing 3 standard deviations we expect the square of the noise in the bin to be less than approximately 5.5×10^{-8}. By ignoring bins with a squared value less than 5.5×10^{-8} and slowly decreasing this number to, say, 5.5×10^{-10} one can ensure that higher error rate Paulis are first recovered, before exploring possibly empty bins for low error rate Paulis.

The second component is the willingness to accept that there is only one value in the bin offsets. This time we start with a strict check, and we accept only a Pauli if the noise is below a threshold, then slowly relax this (by δ_s) as we aim to recover Paulis that happen to have increasing amounts of noise in the bins with them.

VII. PROVABLE RECOVERY ALGORITHM

In this section, we describe in detail a hashing-based subsampling recovery algorithm for which we prove a recovery guarantee. For convenience, we collect the notation used in the provable recovery algorithm into a glossary of symbols in Table I.

Consider a Walsh-Hadamard transformation among n-qubit Pauli eigenvalues and Pauli error rates like Eq. (5). In order to recover a set of sparse error rates with noisy eigenvalues, it is necessary to consider a noisy variation

$$\widehat{\lambda}_k = \sum_{m \in \mathbb{F}_2^{2n}} (-1)^{\langle k,m \rangle} p_m + w_k, \quad k \in \mathbb{F}_2^{2n}. \tag{10}$$

Note that we employ w_k to indicate the sampling errors of the eigenvalues, and for simplicity we are assuming that they are all independent Gaussian random variables with distribution $\mathcal{N}(0, \xi^2)$. The proposed algorithm follows the SPRIGHT framework of Ref. [45]. It first samples Pauli eigenvalues and forms several groups of bins. The algorithm then implements the peeling-decoder Algorithm 4 to recover individual Pauli error rates from the sampled bins.

To subsample bins from noisy eigenvalues, this algorithm employs C *subsampling groups*. Each subsampling group is specified by a binary matrix $\mathbf{M}_c \in \mathbb{F}_2^{2n \times b}$ for $c \in [C]$ and a set of P *offsets*. The binary matrices $\mathbf{M}_c$ serve as hash functions to isolate individual Pauli error rates into B bins with high probability. In our experimental setting, this B is always chosen as $B = 2^n$ for convenience, while

TABLE I. Glossary of symbols used throughout the proof.

Glossary of symbols	
n	number of qubits
N	4^n, the number of Pauli operators modulo phase
s	sparsity, $s = 4^{\delta n}$ for $0 < \delta < \frac{1}{2}$
δ	related to sparsity by $s = 4^{\delta n}$ and $0 < \delta < \frac{1}{2}$
B	number of bins in a single subsampling group, in the experimental regime described in this paper, typically $B = 2^n$
b	$B = 2^b$ for $0 < b \leq 2n$. Typically $B = 2^n$, but this b is used to indicate the bin number
η	the sparsity coefficient, being B/s
C	is the number of subsampling groups
P_1	the number of random offsets chosen for each subsampling group
P_2	the number of extra offsets chosen for each subsampling group to form an error-correcting code
P	$P = P_1 + P_2$
$U_{c,t}[j]$	the bin generated from subsampling group $c \in [C]$ with index $j \in \mathbb{F}_2^b$, and the subscript $t \in [P]$ indicates the offset this bin uses
ξ	the standard deviation of the noise in the estimated Pauli eigenvalues created by shot noise from the original experiments
σ	the standard deviation of the noise in the Pauli error rates created by WHT from the noisy Pauli eigenvalues
ν	the standard deviation of the noise in a given bin, created by subsampling the noisy Pauli eigenvalues
ϵ_0	the lower bound on the nonzero Pauli error rates, required by assumption A3
w_k	Noise on the eigenvalue λ_k, distributed like $\mathcal{N}(0, \xi)$.
W_m	Noise on error rate p_m, induced by the WHT.

in the following proof it is sufficient for B to be as large as s, and increasing B exponentially to s is enough to find out the magnitude of s. Therefore in the proof, we use the general case that $B = O(s)$. The P offsets provide redundancy designed to make the recovery algorithm robust to limited amounts of sampling noise.

Before constructing explicit algorithms, we shall introduce a method for choosing offsets by using good error-correcting codes [45].

Definition 1 (Offsets)**.** *Let $P = P_1 + P_2$ with $P_i = O(n)$ for $i = 1, 2$. We choose P_1 random offsets $\mathbf{d}_t$ for $t = 0, \ldots, P_1 - 1$ chosen independently and uniformly at random over $\mathbb{F}_2^n$, and P_2 coded offsets $\mathbf{d}_t$ for $t = P_1, \ldots, P - 1$ such that the offset matrix $\mathbf{G} = [\cdots; \mathbf{d}_t; \cdots;] \in \mathbb{F}_2^{P_2 \times 2n}$ constitutes a generator matrix of a linear code with parameters $[P_2, 2n, \beta P_2]$ with $\beta > \mathbb{P}$. Here $\mathbb{P}$ is an upper bound on the probability that the sample error will change the sign of a singleton bin (i.e., a bin with a single nonzero Pauli error rate).*

It is convenient in what follows to define a $2n \times 2n$ matrix J_n given by

$$J_n = X \otimes I_n = \begin{pmatrix} 0_n & I_n \\ I_n & 0_n \end{pmatrix}, \tag{11}$$

where 0_n is the $n \times n$ zero matrix and I_n is the $n \times n$ identity matrix. This is the symplectic form that controls the commutation relations in the Pauli group. That is, if $p, q \in \mathbb{F}_2^{2n}$, then

$$\langle p, q \rangle = p^T J_n q, \tag{12}$$

where the arithmetic is implicitly modulo 2.

We now introduce Algorithm 2 to use for data preprocessing and bin construction. The indices on each array are considered to be modulo their respective dimension, and each element of the summation $\mathbf{M}_c' \ell + \mathbf{d}_{c;t}$ is calculated in the field $\mathbb{F}_2$. The algorithm calculates bin coefficients using the corresponding binary matrices and by taking sums over the whole space $\mathbb{F}_2^b$. After this subsampling process, each subsampling group contains P sets of $B = 2^b$ bins, where b is a free parameter. The result of applying Algorithm 2 is summarized in the following lemma.

Lemma 1 (**B**asic observation model)**.** *The B-point WHT subsampled bin coefficients with index $j \in \mathbb{F}_2^b$ can be written as*

$$U_{c;t}[j] = \sum_{m:\, \mathbf{M}_c^T m = j} p_m (-1)^{\langle \mathbf{d}_{c;t}, m \rangle} + W_{c;t}[j],\ \forall t \in [P]. \tag{13}$$

```
Input: Offsets $\mathbf{d}_{c;t}$ for observation index $t \in [P]$ and subsampling index $c \in [C]$;
Input: Subsampling matrices $\mathbf{M}_c \in \mathbb{F}_2^{2n \times b}$ for some $b > 0$ and $c \in [C]$.
Modify: $\mathbf{M}_c' \leftarrow J_n \mathbf{M}_c J_b \ \forall c \in [C]$.
for all $c \in [C]$, $t \in [P]$, and $\ell \in \mathbb{F}_2^b$ do
    $k \leftarrow \mathbf{M}_c' \ell + \mathbf{d}_{c;t}$
    Estimate : $\widehat{\lambda}_k$
end for
$B \leftarrow 2^b$
for all $c \in [C]$ and $t \in [P]$ do
    $U_{c;t}[j] \leftarrow \frac{1}{B} \sum_{\ell \in \mathbb{F}_2^b} (-1)^{\langle j, \ell \rangle} \widehat{\lambda}_{\mathbf{M}_c' \ell + \mathbf{d}_{c;t}}$
    Return $U_{c;t}[j]$
end for
```

Algorithm 2. Subsampling and WHT.

Moreover, the sample error is as follows:

$$W_{c;t}[j] = \sum_{m:\, \mathbf{M}_c^T m = j} W_m (-1)^{\langle \mathbf{d}_{c;t}, m \rangle},$$

where W_m is the noise of the Pauli error rate p_m.

We remark that the noise W_m on the error rate p_m is induced by the Gaussian noise $\{w_k\}$ on the noisy eigenvalues $\{\lambda_k\}$ by the WHT. It is important to keep these two noise sources separate, although we do not make much direct use of w_k in the remainder of the paper.

Proof. Denote $p_m + W_m$ by $\tilde{p}_m$, so the noisy Pauli eigenvalues can be transformed to

$$\widehat{\lambda}_k = \sum_{m \in \mathbb{F}_2^{2n}} (-1)^{\langle k, m \rangle} \tilde{p}_m, \quad \forall\, k \in \mathbb{F}_2^{2n}.$$

From Algorithm 2, a specific bin $U_{c;t}[j]$ for some $c \in [C]$, $t \in [P]$, and $j \in \mathbb{F}_2^{2n}$ is constructed as follows:

$$U_{c;t}[j] = \frac{1}{B} \sum_{\ell \in \mathbb{F}_2^b} \sum_{m \in \mathbb{F}_2^{2n}} (-1)^{\langle m, \mathbf{M}_c' \ell \rangle} (-1)^{\langle j, \ell \rangle} (-1)^{\langle m, \mathbf{d}_{c;t} \rangle} \tilde{p}_m.$$

A key observation is

$$\begin{aligned} \langle m, \mathbf{M}' \ell \rangle &= m^T J_n \mathbf{M}' \ell \\ &= m^T J_n \times J_n \mathbf{M} J_b \ell \\ &= (\mathbf{M}^T m)^T J_b \ell \\ &= \langle \mathbf{M}^T m, \ell \rangle, \end{aligned}$$

where the first equation is from the definition of the symplectic inner product in Sec. II, and the second equation comes from the Modify part in Algorithm 2, and the third is due to the following property of J:

$$J_n \times J_n = I_{2n} \quad \forall\, n \in \mathbb{N}. \tag{14}$$

Thus the bin can be simplified as follows:

$$\begin{aligned}U_{c;t}[j] &= \frac{1}{B}\sum_{m\in\mathbb{F}_2^{2n}}\sum_{\ell\in\mathbb{F}_2^{b}}(-1)^{\langle \mathbf{M}_c^T m+j,\ell\rangle}(-1)^{\langle m,\mathbf{d}_{c;t}\rangle}\tilde{p}_m\\ &= \sum_{m:\,\mathbf{M}_c^T m=j}\tilde{p}_m(-1)^{\langle m,\mathbf{d}_{c;t}\rangle}\\ &= \sum_{m:\,\mathbf{M}_c^T m=j} p_m(-1)^{\langle \mathbf{d}_{c;t},m\rangle} + W_{c;t}[j],\end{aligned}$$

where $W_{c,t}[j]$ is defined in the lemma. ■

We note that from Line 10 in Algorithm 2, the fact that the original noise $\mathbf{w}$ is isotropic, and the fact that the ℓ-bit WHT is proportional to an orthogonal transformation, it follows that the noise in each bin $\mathbf{W}_c[j]$ remains Gaussian distributed, but according to the distribution $\mathcal{N}(0,\nu^2\mathbb{1})$ where $\nu^2 = \xi^2/B$. Moreover, we can combine each $U_{c;t}[j]$ for different $t \in [P]$, and get a vector

$$\mathbf{U}_c[j] := (U_{c;0}[j],\ldots,U_{c;P-1}[j])^T,$$

and an analogous vectorization can be implemented on the offsets

$$\mathbf{D}_c := [\mathbf{d}_{c;0}\cdots\mathbf{d}_{c;P-1}].$$

Therefore, Lemma 1 can be rewritten as follows.

Lemma 2 (Bin observation model). *The B-point WHT subsampled bin with index $j \in \mathbb{F}^b$ in the* c*th subsampling group is*

$$\mathbf{U}_c[j] = \sum_{m:\,\mathbf{M}_c^T m=j} p_m(-1)^{\langle \mathbf{D}_c,m\rangle} + \mathbf{W}_c[j], \tag{15}$$

where the noise $\mathbf{W}_c[j] = \sum_{m:\,\mathbf{M}_c^T m=j} W_m(-1)^{\langle \mathbf{D}_c,m\rangle}$ is distributed as $\mathbf{w}_c[j] \sim \mathcal{N}(0,\nu^2\mathbb{1})$ with $\nu^2 = \xi^2/B$, and W_m is the WHT noise of Pauli error rate p_m.

Proof. This is a variation of Lemma 1. ■

After subsampling and calculating bins, it is straightforward to design a protocol to extract information from these bins. The idea is to construct a bipartite graph G, as in Fig. 2, with s left nodes representing nonzero Pauli error rates and BC right nodes representing bin vectors $\mathbf{U}_c[j]$.

We draw an edge from each left node (a nonzero Pauli error rate) to every right node that contains that Pauli. Each Pauli error rate will occur exactly once in each subsampling group, the degree of the left nodes is therefore C. We can use the resulting degrees of the right-hand nodes to partition them into three types. We call a bin with exactly one nonzero Pauli error rate a *singleton*, and similarly there are *zeroton* and *multiton* bins that contain exactly zero or contain more than one Pauli error rate, respectively. (Recall that this graph is depicted in Fig. 2.) Shortly we describe in detail a method to detect which type of bin a particular node has been partitioned into. After invoking such a bin detector, the peeling decoder can be designed to peel out the detected singleton Pauli error rates by subtracting them from every multiton bin in which they appear, removing the associated edge from the graph. This will reduce the degree of that right-hand node, potentially turning it from a multiton bin into a singleton bin. For the range of parameters that we choose and the assumptions outlined above, iterating this decoder to discover new singletons and reduce multitons will converge to reduce the graph to only zeroton and singleton bins with high probability.

In Algorithm 3, we apply an array $\mathbf{T}$ that indicates the variance of the propagated noise part, $W_{c;t}[j]$, in each bin. These numbers help track the propagation of error in the bin detector from the calculation in Line 12 of Algorithm 3. The equation in Line 11 of that algorithm describes how to update $\mathbf{T}$. Lemma 7 below shows the need and utility of this parameter.

One subtlety to applying the peeling decoder to this graph is that the graph might have cycles. Peeling on a graph with cycles will in general lead to dependencies in the random variables, which complicates the analysis. However, as we show below in Lemma 5, large local neighborhoods of the peeling graph look locally treelike with high probability, therefore we can peel for a large number of steps before encountering a cycle. With the correct choice of parameters, the treelike neighborhood can be made large enough throughout the graph to ensure convergence of the peeling decoder.

As we previously mention, the peeling-decoder algorithm is based on a subroutine that we call *bin detector* (it is set out in Algorithm 4). We denote it by $\mathrm{BD}(\mathbf{U}_c, \mathbf{D}_c, T)$. The subroutine, BD, will take a bin, the offsets chosen, and a noise parameter T as inputs, and it will output an estimate for the type (zeroton, singleton, or multiton) of the bin $\widehat{\mathfrak{B}}$, and if the bin is a singleton it also returns the estimated index $\widehat{m}$ and Pauli error rate $\widehat{p_{\widehat{m}}}$. The subroutine BD also depends on two parameters γ_1 and γ_2, but these can be chosen as arbitrary constants in the interval $(0,1)$. Their only purpose is to ensure exponential decay of the failure probability of bin detection, as we discuss in Lemma 10.

By using the sparsity assumption and our choice of subsampling matrices, this peeling process will succeed with

```
Input : observation vectors $\mathbf{U}_c[j]$, offsets $\mathbf{D}_c$ and array
    $\mathbf{T}_c[j]$ initialized by 1 for $j \in \mathbb{F}_2^b$, $c \in [C]$;
Input : the number of peeling iterations $I$;
$\mathcal{P} \leftarrow$ initialize empty list of Paulis $(\widehat{m}, \widehat{p}_{\widehat{m}})$
for $i \in [I]$ do
    for all $c \in [C]$ and $j \in \mathbb{F}_2^b$ do
        $(\widehat{\mathfrak{B}}, \widehat{m}, \widehat{p}_{\widehat{m}}) \leftarrow \mathrm{BD}(\mathbf{U}_c[j], \mathbf{D}_c, \mathbf{T}_c[j])$
        if $\widehat{\mathfrak{B}} =$ single-ton then
            $\mathcal{P} \leftarrow (\widehat{m}, \widehat{p}_{\widehat{m}})$
            for all $c' \in [C]$ and $c' \neq c$ do
                Locate bin index $j_{c'} \leftarrow \mathbf{M}_{c'}^T \widehat{m}$
                $\mathbf{T}_{c'}[j_{c'}] \leftarrow \mathbf{T}_{c'}[j_{c'}] + \frac{\mathbf{T}_c[j]}{P_1} + \frac{(P_1-1)B}{P_1 N}$
                $\mathbf{U}_{c'}[j_{c'}] \leftarrow \mathbf{U}_{c'}[j_{c'}] - \widehat{p}_{\widehat{m}}(-1)^{\langle \mathbf{D}_{c'}, \widehat{m} \rangle}$
            end for
        else if $\widehat{\mathfrak{B}} \neq$ single-ton then
            continue to next $j$.
        end if
    end for
end for
Return: $\mathcal{P}$
```

Algorithm 3. Subsampling and WHT.

high probability. Intuitively we can see this from our ability to choose subsampling matrices **M** in such a way that we can find bins that typically contain only zero or one nonzero Pauli error rate. In Ref. [45] the authors provide a proof that if a bin-detector algorithm always returns an exactly correct answer, then the oracle-based peeling decoder has a failure probability that vanishes with the signal size. So it suffices to propose a suitable design for the bin detector and a corresponding recovery guarantee.

In designing such a bin detector we need to estimate the index of the relevant Pauli in the bin. For index estimation in the setting where there is noise we need to make our estimation robust. One approach is to use some repetition of detection and a majority voting. A better approach is to use some form of error-correcting code for the offsets, as discussed above in Definition 1. In what follows, we use the following definition of a sign function:

$$sgnx = \begin{cases} 0 & \text{if } x \geq 0, \\ 1 & \text{if } x < 0. \end{cases} \tag{16}$$

With this definition, we have the following lemma, which confirms that offsets chosen in accordance with Definition 1 can be used to estimate the indices.

Lemma 3. *Given a singleton bin (m, p_m) observed with noise*

$$U = p_m(-1)^{\langle \mathbf{d}, m \rangle} + W, \tag{17}$$

and supposing that the variance in each row (offset, **d***) of the bin is equal to $T\nu^2$, then the sign of each observation satisfies*

$$\mathrm{sgn}\,[U] = \langle \mathbf{d}, m \rangle \oplus Z, \tag{18}$$

where Z is a Bernoulli random variable with probability $\Pr(Z=1) \leq \mathbb{P}_m := \sqrt{T\nu^2/2\pi p_m^2}\, e^{-p_m^2/2T\nu^2}$.

Proof. The first term in Eq. (18) follows trivially from the sign of the power of minus one in Eq. (17) and the fact that p_m, being a probability, is always positive. The second term, Z, will be 1 if and only if $|W|$ is larger than p_m so that it can change the sign generated by the first term of Eq. (17). Therefore, Z is Bernoulli distributed with a probability that we can bound as follows. Recalling that W is a Gaussian random variable, we can use the relevant tail bounds for our assumption on the variance (for details see Ref. [50]) to obtain

$$\begin{aligned} \Pr(Z=1) &= \frac{1}{2}\Pr(|W| > p_m) = \Pr(W > p_m) \\ &\leq \sqrt{\frac{T\nu^2}{2\pi p_m^2}}\, e^{-p_m^2/2T\nu^2} = \mathbb{P}_m, \end{aligned} \tag{19}$$

where T is the number extracted from the array **T** in Algorithm 3. ■

Remark 1. *If we assume that the maximum degree of right nodes in the bipartite graph G is not larger than $\frac{1}{2}P_1$, $\mathbb{P}_m < \frac{1}{2}$ is satisfied for all $m \in \mathbb{F}_2^{2n}$ (using* **A1** *in Assumptions 1 and Lemma 9). We bound the probability that this right-hand degree assumption fails in Lemma 8.*

In what follows, we ignore the subscripts c and indices j of the bins when it does not lead to any misunderstanding.

Now Lemma 3 can be used to identify $\widehat{m}$, the index of the Pauli error rate in a singleton bin. Given the offsets chosen in Definition 1 and recalling Lemma 2, we have the following equation from the code generator **G** for the signs of every element in a bin,

$$\begin{bmatrix} \mathrm{sgn}\,[U_{P_1}] \\ \vdots \\ \mathrm{sgn}\,[U_{P-1}] \end{bmatrix} = \langle \mathbf{G}, m \rangle \oplus \begin{bmatrix} Z_{P_1} \\ \vdots \\ Z_{P-1} \end{bmatrix}. \tag{20}$$

Since the bit length of the index m is $2n$, we can choose the number P_2 as follows. We choose any linear code with

```
Input: bin $\mathbf{U}_c$, offsets $\mathbf{D}_c$ and the number $T$ to indicate error size;
Parameter: real numbers $\gamma_1, \gamma_2 \in (0,1)$;
if $\frac{1}{P_1}\sum_{t=0}^{P_1-1} U_{c;t}^2 \leq T(1+\gamma_1)\nu^2$ then
    $\widehat{\mathfrak{B}} \leftarrow$ zero-ton
    Return ($\widehat{\mathfrak{B}}$, nil, nil)                ▷ zero-ton verification
end if
$\widehat{m} \leftarrow$ Decode([sgn $[U_{P_1}], \cdots,$ sgn $[U_{P-1}]]^T$)
$\widehat{p}_{\widehat{m}} \leftarrow \frac{1}{P_1}\sum_{t=0}^{P_1-1}(-1)^{\langle \mathbf{d}_{c;t},\widehat{m}\rangle} U_{c;t}$    ▷ single-ton search
if $\frac{1}{P_1}\sum_{t=0}^{P_1-1}(U_{c;t} - (-1)^{\langle \mathbf{d}_{c;t},\widehat{m}\rangle}\widehat{p}_{\widehat{m}})^2 \leq T(1+\gamma_2)\nu^2$ then
    $\widehat{\mathfrak{B}} \leftarrow$ single-ton
    Return ($\widehat{\mathfrak{B}}$, $\widehat{m}$, $\widehat{p}_{\widehat{m}}$)          ▷ single-ton verification
else
    $\widehat{\mathfrak{B}} \leftarrow$ multi-ton
    Return ($\widehat{\mathfrak{B}}$, nil, nil)
end if
```

Algorithm 4. Bin Detector: BD($\mathbf{U}_c, \mathbf{D}_c, T$).

rate R and distance d, and a decoder that can decode up to at least a minimum distance $\beta 2n/R$ for parameters $\beta, R = \Theta(1)$. Obviously this requires $d \geq \beta 2n/R$. The additional constraint on β is that β is larger than the probability $\mathbb{P}$ of any of the Bernoulli random variables Z_i to be 1. Then we can choose $P_2 = 2n/R$. That is, we are looking for a classical linear code with parameters $[2n/R, 2n, d \geq \beta 2n/R]$. There are a number of pre-existing candidate codes that can be decoded up to a constant fraction of the minimal distance in linear time in the length of the code exist that satisfy these stringent conditions. For example, expander codes [51] can be implemented to construct the code generator $\mathbf{G}$ and the parity check matrix $\mathbf{H}$, and the greedy linear-time decoder [51] can correct errors with weight up to $d/4$. The decoder of the corresponding code is required to retrieve the estimate $\widehat{m}$. Since the manner of coding and decoding is flexible, here we use only Decode to indicate the decoder.

$$\widehat{m} = \texttt{Decode}\left(\begin{bmatrix} \texttt{sgn}\,[U_{P_1}] \\ \vdots \\ \texttt{sgn}\,[U_{P-1}] \end{bmatrix}\right). \tag{21}$$

With this we can specify the modified algorithm to detect the bin U with the offsets as in Definition 1 along with the corresponding number T.

We are now ready to give a precise specification of the bin-detector algorithm.

VIII. PROOF OF MAIN THEOREM

We now repeat the statement of the main theorem.

Theorem 2. *Suppose Assumptions 1 hold for an unknown Pauli channel with eigenvalues* $\boldsymbol{\lambda}$ *and error rates* $\mathbf{p}$. *Then with failure probability* $\mathbb{P}_F \leq e^{-O(n)}$, *Algorithms 2–4 estimate the s-sparse Pauli error rates* $\widehat{\mathbf{p}}$ *such that* $\|\widehat{\mathbf{p}} - \mathbf{p}\|_\infty \leq 2\xi/\sqrt{B}$ *using* $O(sn)$ *eigenvalue queries and* $O(sn^2)$*-time classical computation.*

Recall that using the protocol in Ref. [37], we can estimate $O(sn)$ eigenvalues to within variance ξ^2 by doing $O(n^2/\xi^2)$ measurements. Therefore, heuristically, the entire algorithm needs $O(n^2/\xi^2)$ measurements to achieve this recovery guarantee.

Proof. Firstly we consider the stated query and computational complexities. From [45, Theorem 2], it is shown that the oracle-based peeling decoder succeeds with probability $1 - O(1/s)$ for a random sparse set (obeying assumption **A1**) as long as $C = O(1)$ and $B = O(s)$. Therefore, to prepare these bins, the number of queried eigenvalues is $BPC = O(Ps)$.

To construct the bins and the corresponding graph, the computational complexity can be calculated by the complexity from the construction algorithm. Note there are P offset coefficients $\mathbf{d}$ and each $\mathbf{U}_{c,t}[j]$ comes from the sum of B samples in Algorithm 2. To construct the total set $\{\mathbf{U}_{c,t}[j]\}_{C,P,B}$, we can use a fast WHT [which has complexity $O(B\log B)$ to calculate a B-point WHT] for each offset. Therefore, the computational complexity for this part is

$$Z_1 = O(PB\log B) = O(Psn).$$

The second part of computational complexity comes from the computation of Algorithm 3. Each step in the bin detector checks the type of the bin with $O(P)$ calculations, and there are $O(B)$ iterations. Accordingly, the complexity is

$$Z_2 = O(PB) = O(Ps).$$

Therefore, the total computational complexity is $Z = Z_1 + Z_2 = O(Psn)$.

Let us denote by E_{bin} the event that any invocation of the bin detector (in the execution of Algorithm 3) returns one or more of the following: (a) an incorrect identification of the type of bin; (b) wrong indices for a detected singleton, or (c) a misestimate of the Pauli error rates of a detected singleton by more than $2\nu = 2\xi/\sqrt{B}$. Furthermore, let D denote the event that the maximum degree of the right nodes in the graph G is less than or equal to P_1. Let H be the event that all the peeling routes in the procedure are cycle-free. Then utilizing the law of total probability we can bound the failure rate of the entire algorithm as

$$\mathbb{P}_F \leq \Pr\left(\text{peeling decoder fails}\middle|E_{\text{bin}}^c\right) + \Pr\left(E_{\text{bin}}|D,H\right) + \Pr\left(D^c\right) + \Pr\left(H^c\right). \tag{22}$$

Here the subscript c denotes the complement of the event, e.g., E^c_{bin}, denotes that no bin detection error occurred in the entire execution of Algorithm 3.

The first term in Eq. (22) is the chance that the oracle-based peeling decoder fails, even though the bin decoder is always correct. This probability scales as $O(1/s)$ (Proposition 4 in Ref. [45]).

To bound the second term, it is more convenient to consider the probability that every invocation of a bin detector works correctly given D and H. Let M denote the number of times the peeling decoder calls the bin-detector subroutine. This probability can be expressed as follows:

$$\begin{aligned}
&\Pr\left(\bigcap_{i=1}^{M} E_i^c \middle| D, H\right)\\
&= \Pr\left(E_M^c \middle| \bigcap_{i=1}^{M-1} E_i^c, D, H\right) \Pr\left(\bigcap_{i=1}^{M-1} E_i^c \middle| D, H\right)\\
&= \Pr\left(E_M^c \middle| \bigcap_{i=1}^{M-1} E_i^c, D, H\right) \cdots \Pr\left(E_1^c | D, H\right),
\end{aligned}$$

where E_i denotes the event that the ith call of bin detector returns a wrong answer. According to Lemma 7, the parameter T will always correctly estimate the variance if all the earlier bin detectors worked correctly. From Lemma 10, each term in the above equation will be lower bounded by

$$\Pr\left(E^c | D, V, H\right) \geq 1 - e^{-O(P_1)},$$

where V here just indicates that all the previous bin detectors work correctly. So we have

$$\Pr\left(\bigcap_{i=1}^{M} E_i^c \middle| D, H\right) \geq \left[1 - e^{-O(P_1)}\right]^M.$$

Moreover, since M, the number of times the bin detector routine is called, is at most $O(BCs)$, the upper bound of the second term is

$$\begin{aligned}
\Pr\left(E_{\text{bin}} | D, H\right) &\leq 1 - \left[1 - e^{-O(P_1)}\right]^M\\
&\leq O(BCs)e^{-O(n)} \leq e^{-O(n)}.
\end{aligned}$$

Lemma 8 provides that the third term in Eq. (22) is also exponentially decaying with P_1:

$$\mathrm{P}\left(D^c\right) \leq e^{-O(P_1)} \leq e^{-O(n)},$$

where the last inequality comes from the definition of P_1 from Definition 1. Similarly Lemma 5 and Remark 2 provide the bound on the probability of H^c:

$$\Pr(H^c) \leq O\left[\frac{\log^{\log\log(s)} s}{s}\right] \leq e^{-O(n)}.$$

Therefore, the total failure probability of our peeling-decoder algorithm is vanishing exponentially with the number of qubits n. And from Definition 1, the total number of offsets consists of $P_1 = \Theta(n)$ random offsets and $P_2 = \Theta(n)$ coding offsets, thus $P = \Theta(n)$ and stated complexities have been proven. ■

IX. TAIL BOUNDS

In this section, we prove several statements bounding the failure probabilities of various events that can cause the bin detector outlined as Algorithm 4 to fail. One of the main lemmas that we need is the following tail bound on Gaussian random variables.

Lemma 4 (Tail bound [45, Lemma 11). *Given* $\mathbf{g}, \mathbf{k} \in \mathbb{R}^N$ *where* $\mathbf{k}$ *is an isotropic Gaussian random variable* $\mathbf{k} \sim \mathcal{N}(0, \nu^2 \mathbb{1}_N)$, *then the following tail bound holds:*

$$\Pr\left(\frac{1}{N}\|\mathbf{g}+\mathbf{k}\|^2 \geq \tau_1\right) \leq e^{-(N/4)\left(\sqrt{2\tau_1/\nu^2-1}-\sqrt{1+2\theta_0}\right)^2} \tag{23}$$

$$\Pr\left(\frac{1}{N}\|\mathbf{g}+\mathbf{k}\|^2 \leq \tau_2\right) \leq e^{-(N/4)\left(1+\theta_0-\tau_2/\nu^2\right)^2/(1+2\theta_0)}, \tag{24}$$

for τ_1, τ_2*, and* θ_0 *satisfying*

$$\tau_1 \geq \nu^2(1+\theta_0), \quad \tau_2 \leq \nu^2(1+\theta_0), \quad \theta_0 = \frac{\|\mathbf{g}\|^2}{N\nu^2}.$$

Since we use this Lemma 4 to get a failure bound, it is critical to show the sample errors within the different offsets for a particular bin are independent. However, given that bins are created as shown in Line 12 in Algorithm 3, it is not immediately clear that the sample errors remain independent. To show this independence let us first extend the definition of the sample errors $W_{c;t}[j]$ to take into account the effect of peeling-decoder Algorithm 3. Recall that for any particular bin we have $P = P_1 + P_2$ offset bins.

Definition 2. *For a specific bin, regard* $\mathbf{U}_c[j]$ *as a vector of length* P *as in Lemma 2. At a given time step in Algorithm 2, denote the set of indices of the current contained nonzero Pauli error rates by* $\mathcal{P}$. *Define the random* *offset errors and the coded offset errors by the equation*

$$W_{c;t}[j] := U_{c;t}[j] - \sum_{m\in\mathcal{P}}(-1)^{\langle \mathbf{d}_{c;t,m}\rangle}p_m,$$

where $t\in[P_1]$ for the random offset errors and $t\in P_1+[P_2]$ for the coded offset errors. We can combine all of these sample errors to define $\mathbf{W}_c[j]$ following the manner of Lemma 2.

In order to discuss the independence of errors and the evolution of their variance, we first introduce some results to rule out some delicate situations. To define this more rigorously, consider the *directed neighborhood* $\mathcal{N}_e^l$ in the bipartite graph, which consists of nonzero Pauli error rates (left nodes) and all the bins (right nodes). The neighborhood $\mathcal{N}_e^{2l}$ with length $2l$ and an edge $e=(v,c)$ is an induced subgraph containing all the edges and nodes on paths $e_1, e_2\cdots, e_{2l}$ from node v where $e\neq e_1$.

Denote by $\mathcal{T}_l$ the event that for every edge in the bipartite graph, this subgraph $\mathcal{N}_e^{2l}$ is cycle-free. If $\mathcal{T}_l$ occurs, then all the first l peeling iterations will progress independently and there will be no initial error propagating to any bin more than once in the first l iterations. It has been shown in Ref. [45] that with sufficiently large s and N, the effective part of the subsampling-based bipartite graph, similar to Fig. 2, is sufficiently cycle-free for our purposes, as the following lemma illustrates.

Lemma 5 (Ref. [45, Lemma 6]). *For any iteration l, the probability of the complement of $\mathcal{T}_l$ is bounded as*

$$\Pr(\mathcal{T}_l^c) \le c_0 \times \frac{\log^l s}{s},$$

for some constant c_0.

Remark 2. *Reference [45] shows that the probability p_l that an arbitrary edge remains after l peelings given that the event $\mathcal{T}_l$ is true can be calculated recursively as*

$$p_l = \begin{cases} 1 & l=0 \\ \left(1-e^{-p_{l-1}/\eta}\right)^{C-1} & \text{otherwise,}\end{cases} \tag{25}$$

where η is the factor B/s and C is the number of subsampling groups.

To illustrate the convergence of Algorithm 3 given the event $\mathcal{T}_l$, consider taking $C=6$ and $\eta=1$, which are reasonable choices in the regime of interest. Then the probability of an edge will decrease to approximately machine precision in only three iterations. In general this p_l vanishes exponentially with an exponent of l. Using the law of total probability, the probability that there exist any edges after $l=\Omega(\log\log s)$ iterations is

$$p_l + \Pr(\mathcal{T}_l^c) = O\left(\frac{\log^{\log\log s} s}{s}\right). \tag{26}$$

Therefore, the event that there exist bins getting peeled by some earlier bins in a cyclic manner during the whole process happens with probability of the same magnitude of Eq. (26), which converges to zero with s.

Lemma 6. *For an arbitrary timestamp in Algorithm 3, sample errors in each of the random offset errors for a particular bin $\mathbf{U}_c[j]$ remain independent of each other given that the peeling route is cycle-free.*

Proof. For an initial bin subsampled from Algorithm 2, consider an arbitrary pair of offsets labeled by $t_1, t_2\in[P_1]$ in the same bin $\mathbf{U}_c[j]$

$$W_{c;t_1}[j] = \sum_{m:\,\mathbf{M}_c^T m=j} W_m(-1)^{\langle \mathbf{d}_{c;t_1},m\rangle},$$

$$W_{c;t_2}[j] = \sum_{m:\,\mathbf{M}_c^T m=j} W_m(-1)^{\langle \mathbf{d}_{c;t_2},m\rangle}.$$

Since all the errors W_m are i.i.d. Gaussian random variables $\mathcal{N}(0,\xi^2/N)$, it is obvious that $\mathbb{E}(W_{c;t_1}[j]\times W_{c;t_2}[j])=0$. So they are independent given that the expected values of the samples errors are 0.

The peeling decoder in Line 12 in Algorithm 3 causes errors in the estimate of $W_{c;t}[j]$ to propagate in the following manner:

$$W_{c;t}[j] \leftarrow W_{c;t}[j] + (p_m - \widehat{p}_{\widehat{m}})(-1)^{\langle \mathbf{d}_{c;t},\widehat{m}\rangle}.$$

We now proceed by induction. As discussed above, the noise is initially independent, so the base case is satisfied. Now assume that the sample errors before peeling are independent of each other. Observing that the updated error still has mean zero, we can calculate the expected value of a product between an arbitrary pair of sample errors in the offsets of a bin to show independence between the offset bins. For convenience, denote the updated error by $W_{c;t}[j]'$. Then we have

$$\begin{aligned}\mathbb{E}(&W_{c;t_1}[j]' \times W_{c;t_2}[j]')\\ &= \mathbb{E}\big\{W_{c;t_1}[j]\times W_{c;t_2}[j]\\ &\quad + W_{c;t_1}[j]\times (p_m-\widehat{p}_{\widehat{m}})(-1)^{\langle \mathbf{d}_{c;t_2},\widehat{m}\rangle}\\ &\quad + W_{c;t_2}[j]\times (p_m-\widehat{p}_{\widehat{m}})(-1)^{\langle \mathbf{d}_{c;t_1},\widehat{m}\rangle}\\ &\quad + (p_m-\widehat{p}_{\widehat{m}})^2(-1)^{\langle \mathbf{d}_{c;t_2}+\mathbf{d}_{c;t_1},\widehat{m}\rangle}\big\} = 0.\end{aligned}$$

The first three terms vanish because the noise has zero mean and the peeling route is cycle-free, and the last term vanishes because the expectation over the independent random offset phases is $\mathbb{E}\left[(-1)^{\langle \mathbf{d}_{c;t_i}, m\rangle}\right] = 0$ for any $t_i \in [P_1]$ and m. ■

In Algorithm 3, we employ an array $\mathbf{T}$ to keep track of the variance of sample error for each bin. This array gets updated whenever the algorithm peels a bin using an estimated Pauli error rate. We now show that this does indeed correctly track the variance of the sample errors in the bins.

Lemma 7. *Suppose that at a given arbitrary timestamp in Algorithm 3 all the bin-detector subroutines called earlier have correctly identified their bins and the peeling route is cycle-free. Then for each bin and its corresponding offsets* $\mathbf{U}_c[j]$*, the sample error for that bin and each of its offsets* $\mathbf{W}_c[j]$ *have the same variance* $T_c[j] \times \nu^2$*.*

Proof. Since this statement is based on the premise that all of the earlier bin-detector runs are accurate, we can assume that the index $\widehat{m} = m$ is correct. We still write $\widehat{m}$ to distinguish these index estimates from the original index m. The idea is to calculate the variance after a peel by induction with the fact that for any two random variables,

$$\text{Var }(X+Y) = \text{Var }(X) + \text{Var }(Y) + 2\text{Cov }(X,Y)\,.$$

Assume that the statement in the lemma holds before a peeling. Then we need to show that if we subtract an estimated Pauli $\widehat{p_{\widehat{m}}}$ from a bin in a different subsampling group that contains this Pauli, the statement is preserved when updating $\mathbf{T}_c[j]$. To do this we work out the variance of the sample error in each term and the covariance between these sample errors. Armed with this we are able to prove that the statement is preserved after peeling.

The peeling process (Line 12 in Algorithm 3) causes error propagation for the error part of the bin $\mathbf{U}_c[j]$ as follows:

$$W_{c;t}[j] \leftarrow W_{c;t}[j] + (p_m - \widehat{p_{\widehat{m}}})(-1)^{\langle \mathbf{d}_{c;t}, \widehat{m}\rangle}. \tag{27}$$

The variance of the first term is by induction $\mathbf{T}_c[j]\nu^2$, while the variance of the second term is not so trivial. The estimated Pauli comes from the *singleton search*, where all the first P_1 observations get summed after adding a random sign. Since all the random signs of the $W_{\widehat{m}}$ terms are annihilated before summation, and all the other error parts still remain random, the variance of this second term in Eq. (27) is

$$\text{Var }(p_m - \widehat{p_{\widehat{m}}}) = \frac{T_{c'}[j'] \times \nu^2}{P_1} + \frac{(P_1-1)B \times \nu^2}{P_1 N}. \tag{28}$$

Because of the assumption that all the initial errors in different bins are independent and the condition that the every peeling route is cycle-free, the covariance term vanishes. Therefore, we calculate variance as follows:

$$\text{Var}\left(W_{c;t}[j]\right)_{\text{after}}/\nu^2 = T_c[j] + \frac{\mathbf{T}_{c'}[j']}{P_1} + \frac{(P_1-1)B}{P_1 N},$$

which proves the lemma. ■

In order to prove Theorem 1 and find a bound on the variances of the sample errors, it is necessary to find an upper bound on the parameters $\mathbf{T}_c[j]$, which need to be analyzed for both the graph and the algorithm. Denote by G the bipartite graph of which each right node represents a bin observation, each left node represents a nonzero Pauli error rate, and edges come from the hash function relation:

$$\mathbf{M}_c^T m = j\,. \tag{29}$$

That is, a bin-observation-node $\mathbf{U}_c[j]$ is connected to error-rate-node p_m if and only if it holds that $\mathbf{M}_c^T m = j$. A right node is a singleton node if and only if it has a single edge connected to it. Every time we peel a left node (that is, we identify a Pauli error) we remove the edges connecting it and the right nodes. Each peeling therefore decreases the degree of the right nodes.

The following two lemmas help us bound the integer array $\mathbf{T}$ as we peel along the graph G. We first bound the right degree of G.

Lemma 8. *The maximum degree of the right nodes in G is less or equal than* $P_1/2$ *with probability* $1 - e^{-O(n)}$*.*

Proof. Put the right nodes in some sequential order, and define events $\{X_i\}_{i=1}^{BC}$ where X_i denotes the ith node and is linked to more than $P_1/2$ left nodes. According to the bin-observation model, Eq. (15), each bin connects with N/B Pauli error rates (most of which will be zero), so the expected degree of a right node is s/B where s is the number of left nodes. Since the algorithm chooses $B = O(s)$, the expected degree $s/B = O(1)$.

Note that by Assumption 1, the support of the Pauli error rates is chosen randomly. Therefore, concentrating on a specific bin i, we can introduce a random variable d_i that denotes the degree of bin-observation-node i (a right node), and introduce the variable d_{ij}, which is 0 if the corresponding jth Pauli error rate is zero or if p_j is not in the ith bin, and 1 otherwise. Then we have the relation

$$d_i = \sum_j d_{ij}\,, \tag{30}$$

since this counts the support in the ith bin. These variables d_{ij} are actually all Bernoulli variables and the only correlation among them comes from the constraint that there are exactly s elements in the entire support. This constraint means that the d_{ij} are negatively correlated, and so the probability of $d_{ij} = 1$ can be upper bounded by considering the event that all the other Pauli rates linked with this bin are zero,

$$\Pr\left(d_{ij} = 1\right) \leq \frac{sB}{N(B-1)}.$$

Now consider another set of i.i.d. Bernoulli variables $\{d'_{ij}\}_j$ each of which is 1 with probability $sB/N(B-1)$. We then have

$$\Pr\left(X_i\right) = \Pr\left(\sum_{j=1}^{N/B} d_{ij} \geq \frac{P_1}{2}\right) \leq \Pr\left(\sum_{j=1}^{N/B} d'_{ij} \geq \frac{P_1}{2}\right).$$

Since the expected value of the sum of $\{d'_{ij}\}$ is $s/(B-1)$, the Chernoff bound is suitable for this case, and we find

$$\Pr\left(X_i\right) \leq \Pr\left(\sum_{j=1}^{N/B} d'_{ij} - \frac{s}{B-1} \geq \frac{P_1}{2} - \frac{s}{B-1}\right)$$
$$\leq e^{-[(B-1)P_1-2s]^2/2(B-1)[(B-1)P_1+2s]}.$$

According to the union bound, the event X, which denotes that there exist some left nodes with degree larger than $P_1/2$, follows from the upper bound,

$$\mathrm{P}\left(X\right) \leq BC\mathrm{P}\left(X_i\right) \leq BCe^{-[(B-1)P_1-2s]^2/2(B-1)[(B-1)P_1+2s]}.$$

That this is at most $e^{-O(n)}$ follows since $B = \Theta(s)$, $s = \Theta(N^\delta)$, and $P_1 = \Theta(n)$. ■

The degree bound we have just proven allows us to bound the maximum element of the array $\mathbf{T}$.

Lemma 9. *Suppose the maximum degree of the right nodes in G is not larger than $P_1/2$. Then for any time step in Algorithm 3, assuming all previous bin detections succeed, the maximum element of the array* $\mathbf{T}$ *is at most 4.*

Proof. The recursive equation for $\mathbf{T}$ (Line 11 in Algorithm 3) is

$$\mathbf{T}_{c'}[j_{c'}] \leftarrow \mathbf{T}_{c'}[j_{c'}] + \frac{\mathbf{T}_c[j]}{P_1} + \frac{(P_1-1)B}{P_1 N}.$$

The algorithm defines a time sequential order for each nonzero Pauli error rate to be detected. For each step i, let $\mathbf{U}_{z_i}$ be the next bin in which we find a nonzero Pauli rate. Let $T_{\max}$ be the current maximum element in a subset $\mathbf{T}_{\text{peeled}}$ of $\mathbf{T}$. Then $\mathbf{T}_{\text{peeled}}$ contains those $\mathbf{T}_c[j]$ of bins in which we already find a nonzero Pauli error. Also, we use $T'_{\max}$ to indicate the maximum $\mathbf{T}_c[j]$ that will exist after the next step.

According to the assumption, the maximum degree of an arbitrary right node is less than $\min(P_1/2, N/B)$, and the number of peels needed is never more than the maximum degree of that node. Note we assume that the previous bins have been accurately detected, so the process will always choose nonzero Pauli error rates (and the corresponding bins) to peel. The noise in the peeling bin will increase by at most $T_{\max}/P_1 + (P_1-1)B/P_1N$. Each $\mathbf{T}$ is initialized as 1, and denote the number of peelings by κ. Therefore

$$T'_{\max} \leq 1 + \kappa \times \left[\frac{T_{\max}}{P_1} + \frac{(P_1-1)B}{P_1 N}\right]$$
$$\leq 1 + \frac{T_{\max}}{2} + 1.$$

Then by induction, because initially the maximum element in $\mathbf{T}_{\text{peeled}}$ is equal to 0 and in each step this value will increase via the above formula, the maximum element we can get is the limit of this recursive inequality, which is $T_{\max} \leq 4$. ■

According to the above two lemmas, the integer $T_{\max}$ indicates that the upper bound of the maximum element in the array $\mathbf{T}$ is not larger than 4 with high probability for a sufficiently large N. In order to compute the failure rates and make the algorithm realizable, we assume (as in Assumption **A3**) that the minimum nonzero Pauli error rate ϵ_0 satisfies $\epsilon_0^2 \geq 4\nu^2 = 4\xi^2/B$, which makes a distinctive barrier between Pauli error rates and any noise.

In anticipation of applying the union bound, let us define the following error categories and their probabilities. For brevity, we denote a zeroton, singleton, or multiton by just the letters $\mathtt{z}$, $\mathtt{s}$, or $\mathtt{m}$, and we denote the true value of the bin by $\mathfrak{B}$.

Definition 3 (Failure modes for bin detection)**.** *The bin-detection-algorithm failure modes are defined as follows:*

(a) *The singleton false negative probability:*

$$\Pr(\mathrm{SFN}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{z}|\mathfrak{B} = \mathtt{s}) + \Pr(\widehat{\mathfrak{B}} = \mathtt{m}|\mathfrak{B} = \mathtt{s}).$$

(b) *The singleton false positive probability:*

$$\Pr(\mathrm{SFP}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{s}|\mathfrak{B} = \mathtt{z}) + \Pr(\widehat{\mathfrak{B}} = \mathtt{s}|\mathfrak{B} = \mathtt{m}).$$

(c) *The multiton ↔ zeroton confusion probability:*

$$\Pr(\mathrm{MZ}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{z}|\mathfrak{B} = \mathtt{m}) + \Pr(\widehat{\mathfrak{B}} = \mathtt{m}|\mathfrak{B} = \mathtt{z}).$$

(d) *The index error probability:*

$$\Pr(\mathrm{I}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{s}, \widehat{m} \neq m|\mathfrak{B} = \mathtt{s}, m).$$

(e) *The value error probability:*

$$\Pr(\mathrm{V}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{s}, |\hat{p}_{\hat{m}} - p_m| > 2\xi/\sqrt{B}|\mathfrak{B} = \mathtt{s}, m, p_m).$$

Of course these probabilities are not all independent. However, by the union bound it suffices to bound each of these bad events individually and the total failure probability will be at most the sum of the probabilities of these failure modes. We show that all of these failure probabilities decay exponentially with P_1, the number of randomly chosen offsets.

Lemma 10. *Let E denote the event that an arbitrary bin detection with inputs as those in Algorithm 3 returns either the wrong bin type, the wrong index or an estimated Pauli error rate with error larger than $2\nu = 2\xi/\sqrt{B}$. Let D be the event that the maximum degree of the right nodes in G is not larger than $P_1/2$. Let V be the event that all prior bin detections succeed. And denote the event that every peeling route is cycle-free by H. Then*

$$\Pr(E|D,V,H) \le O(1)e^{-O(n)}. \tag{31}$$

Proof. This theorem means that the bin-detector algorithm succeeds with high probability whenever D, V, and H occur. To show it, we have to bound the failure probabilities for each failure mode of the bin-detection algorithm and then apply the union bound. We prove most of our statements by bounding failure probabilities with expressions of the form $e^{-O(P_1)}$. This is equivalent to a bound of the form $e^{-O(n)}$ since $P_1 = \Theta(n)$ by Definition 1. Also note that conditioning on events D, V, and H allows the use of Lemmas 7 and 9, specifically that the variance of noise in each row of this bin is $T\nu^2$ from Lemma 7, and that this T is no more than 4 according to Lemma 9.

We first consider the *singleton false negative* probability in Definition 3. Note in this case the underlying bin contains only one Pauli error rate along with noise, that is

$$\mathbf{U}_c = p_m(-1)^{\langle \mathbf{D}_c, m\rangle} + \mathbf{W}. \tag{32}$$

Let $f_1 = \Pr\left(\widehat{\mathfrak{B}} = \mathtt{z}|\mathfrak{B} = \mathtt{s}\right)$. Then by Line 2 in Algorithm 4, the probability can be upper bounded by the probability of a singleton bin passing the zeroton verification:

$$f_1 \le \Pr\left[\frac{1}{P_1}\left\|p_m\mathbf{s}_{c,m} + \mathbf{W}\right\|^2 \le T(1+\gamma_1)\nu^2\right],$$

where $\mathbf{s}_{c,m}$ is the vector such that $\mathbf{s}_{c,m}[t] = (-1)^{\langle m, d_{c;t}\rangle}$. Since here the noise vector $\mathbf{W}$ comes from the sum of noise w, it is obvious that all the elements of $\mathbf{W}$ are Gaussian distributed with variance $T\nu^2$. Therefore, according to the tail bounds of Lemma 4, the following holds as long as $\gamma_1 < \epsilon_0^2/T\nu^2$.

$$f_1 \le e^{-(P_1/4)\left(\epsilon_0^2/T\nu^2 - \gamma_1\right)^2/(1+2\epsilon_0^2/T\nu^2)}. \tag{33}$$

Now let $f_2 = \Pr\left(\widehat{\mathfrak{B}} = \mathtt{m}|\mathfrak{B} = \mathtt{s}\right)$. This kind of failure happens if and only if the singleton bin fails during singleton verification,

$$f_2 = \Pr\left[\frac{1}{P_1}\left\|\mathbf{U}_c - \widehat{p}_{\widehat{m}}\mathbf{s}_{c,\widehat{m}}\right\|^2 \ge T(1+\gamma_2)\nu^2\right].$$

Considering the underlying structure of this bin (32), this probability can be bounded using a conditional probability. We first denote the event $\{|\widehat{p}_{\widehat{m}} - p_m| > \sqrt{4\nu^2} \text{ or } \widehat{m} \ne m\}$ by E_0. Then we observe

$$f_2 \le \Pr(E_0) + \Pr\left[\frac{1}{P_1}\left\|\mathbf{U}_c - \widehat{p}_{\widehat{m}}\mathbf{s}_{c,\widehat{m}}\right\|^2 \ge T(1+\gamma_2)\nu^2\middle|E_0^c\right].$$

Using the tail bound, Eq. (23), we have that

$$\begin{aligned}&\Pr\left[\frac{1}{P_1}\left\|\mathbf{U}_c - \widehat{p}_{\widehat{m}}\mathbf{s}_{c,\widehat{m}}\right\|^2 \ge T(1+\gamma_2)\nu^2\middle|E_0^c\right]\\ &\quad\le e^{-(P_1/4)\left(\sqrt{1+2\gamma_2}-\sqrt{1+2\times 4/T}\right)^2}.\end{aligned} \tag{34}$$

Then using union bound, we can deal with the first term

$$\begin{aligned}\Pr(E_0) &\le \Pr\left(|\widehat{p}_{\widehat{m}} - p_m| > \sqrt{4\nu^2}\right) + \Pr\ (\widehat{m} \ne m)\\ &\le \Pr\ \left(|\widehat{p}_{\widehat{m}} - p_m| > \sqrt{4\nu^2}\middle|\widehat{m} = m\right)\\ &\quad + 2\Pr(\widehat{m} \ne m).\end{aligned} \tag{35}$$

Note above, the estimated Pauli error rate can be calculated according to Algorithm 4, so we obtain the bound

$$\begin{aligned}&\Pr\left(|\widehat{p}_{\widehat{m}} - p_m| > \sqrt{4\nu^2}\middle|\widehat{m} = m\right)\\ &\quad = \mathrm{P}\left(\left|\frac{\mathbf{s}_{c,m}^T\mathbf{U}_c}{P_1} - p_m\right| > \sqrt{4\nu^2}\right)\\ &\quad = \mathrm{P}\left(|Y|/P_1 > \sqrt{4\nu^2}\right) \le 2e^{-\frac{4P_1}{2T}},\end{aligned} \tag{36}$$

where Y is the sum of P_1 i.i.d. Gaussian variables with $\mathcal{N}\left\{0, [T + (P_1-1)B/N]\,\nu^2\right\}$ like Eq. (28), and the last inequality comes from the Chernoff-Hoeffding bound [52]. According to Lemma 9, exponents in (34) and (36) are both scaling linearly with P_1, thus the probabilities decay exponentially with P_1.

Since the second term in Eq. (35), $\Pr(\widehat{m} \ne m)$, is essentially the probability of the index error, the failure probability of such a decoding process also decays exponentially with P_2. In accordance with Eq. (20), the sign vector $\left[\mathtt{sgn}\left[U_{P_1}\right], \ldots, \mathtt{sgn}\left[U_{P-1}\right]\right]^T$ is the sum of a codeword $\langle \mathbf{G}, m\rangle$ and a vector of noise. Since the decoding process fails only if the weight of the noise is larger than the code distance βP_2 and each element of the noise is an independent Bernoulli random variable with error probability upper bounded by $\mathbb{P}$, the index error probability can be bounded by the Chernoff-Hoeffding bound:

$$\Pr\left(\widehat{m} = m\right) \leq e^{-[(\beta/\mathbb{P}-1)^2/3]P_2}. \tag{37}$$

Moreover, as noted in Remark 1, we have $\mathbb{P}_m < \frac{1}{2}$ for all $m \in \mathbb{F}_2^{2n}$. Given the assumptions of D, V, and H, we choose the maximum $\mathbb{P}_m$ to be $\mathbb{P} = \max_m \mathbb{P}_m$. Therefore, using the law of total probability we have

$$f_2 \leq e^{-\frac{P_1}{4}\left(\sqrt{1+2\gamma_2}-\sqrt{1+2\times 4/T}\right)^2} + 2e^{-\frac{4P_1}{2T}} + 2e^{-\frac{(\beta/\mathbb{P}-1)^2}{3}P_2}. \tag{38}$$

Recall from Definition 1 that P_1 and P_2 are proportional to n, so we have a bound $f_2 = e^{-O(n)}$.

We now turn to the case that the bin-detection algorithm incorrectly recognizes a zeroton or a multiton bin as a singleton bin, i.e., we consider the *singleton false positive* probability. For this, we need to consider the general underlying bin structure

$$\mathbf{U}_c = \mathbf{S}_c\mathbf{p} + \mathbf{W}, \tag{39}$$

where $\mathbf{U}_c$ is either zeroton or multiton, and only contains the P_1 fully random offsets when choosing as in Definition 1. Here $\mathbf{S}_c \in \{\pm 1\}^{P_1 \times N/B}$ is the sign matrix constructed according to Lemma 2.

Now consider the probability of the bin detector falsely detecting a zeroton as a singleton, and denote $\Pr\left(\widehat{\mathfrak{B}} = \mathtt{s} | \mathfrak{B} = \mathtt{z}\right)$ by f_3. By Line 2 in Algorithm 4, the probability of f_3 can be bounded by the probability of zeroton verification failing

$$f_3 \leq \Pr\left[\frac{1}{P_1}\|\mathbf{W}\|^2 \geq T \times (1+\gamma_1)\nu^2\right].$$

According to the tail bound, Eq. (23), this failure probability can be bounded by an exponentially decaying function

$$f_3 \leq e^{-(P_1/4)(\sqrt{1+2\gamma_1}-1)^2}. \tag{40}$$

Now let $f_4 = \Pr\left(\widehat{\mathfrak{B}} = \mathtt{s} | \mathfrak{B} = \mathtt{m}\right)$. This error probability can be evaluated under the multiton model when it passes the singleton verification step for some estimated index-value pair $(\widehat{m}, \widehat{p}_{\widehat{m}})$. Using Line 11 in Algorithm 4,

$$f_4 \leq \Pr\left[\frac{1}{P_1}\left\|\mathbf{U}_c - \widehat{p}_{\widehat{m}}\mathbf{s}_{c,\widehat{m}}\right\|^2 \leq T \times (1+\gamma_2)\nu^2\right].$$

Let

$$\mathbf{g} = \mathbf{S}_c\mathbf{p} - \widehat{p}_{\widehat{m}}\mathbf{s}_{c,\widehat{m}}, \tag{41}$$

and let the sample error be $\mathbf{W} = \mathbf{k}$. Then the law of total probability can be used as follows:

$$\begin{aligned} f_4 &= \Pr\left[\frac{1}{P_1}\|\mathbf{g}+\mathbf{k}\|^2 \leq T(1+\gamma_2)\nu^2\right] \\ &\leq \Pr\left[\frac{1}{P_1}\|\mathbf{g}+\mathbf{k}\|^2 \leq T(1+\gamma_2)\nu^2 \middle| \frac{\|\mathbf{g}\|^2}{P_1} \geq 2T\gamma_2\nu^2\right] \\ &\quad + \Pr\left(\frac{\|\mathbf{g}\|^2}{P_1} \leq 2T\gamma_2\nu^2\right). \end{aligned} \tag{42}$$

Note that the first term can be bounded by Eq. (24) since the conditional part shows the lower bound of the parameter θ_0 as defined in Lemma 4

$$\Pr\left[\frac{1}{P_1}\|\mathbf{g}+\mathbf{k}\|^2 \leq T(1+\gamma_2)\nu^2 \middle| \frac{\|\mathbf{g}\|^2}{P_1} \geq 2T\gamma_2\nu^2\right] \leq e^{-\frac{P_1}{4}\frac{\gamma_2^2}{1+4\gamma_2}}. \tag{43}$$

The second term can be bounded as follows. Let $\alpha = \mathbf{p} - \widehat{p}_{\widehat{m}}\mathbf{e}_{\widehat{m}}$, and we have

$$\Pr\left(\frac{\|\mathbf{g}\|^2}{P_1} \leq 2T\gamma_2\nu^2\right) = \Pr\left(\frac{\|\mathbf{S}_c\alpha\|^2}{P_1} \leq 2T\gamma_2\nu^2\right).$$

Here $\mathbf{e}_k$ is the vector with support only on the kth element. We denote the support set of the vector α by $\mathcal{L}_0$, and define the ϵ_0-essential support of α to be

$$\mathcal{L} = \left\{i \in \mathbb{F}_2^{2n} \middle| |\alpha_i| \geq \epsilon_0\right\}. \tag{44}$$

Denote the cardinality of $\mathcal{L}$ by L. Then the above probability can be bounded by an application of the Chernoff-Hoeffding bound.

With the same argument as in the proof of Lemma 6, the sample error in each row in vector $\mathbf{g}$ is independent, and so is the square of that error. When we calculate $\|\mathbf{g}\|^2$, we can regard it as a sum of P_1 independent random variables. Also, each term in this sum contains the same structure, and identically distributed parameters, so we can claim each term is identically distributed.

Therefore, we first analyze the expected value E of each variable in this sum. Take one of these terms X_i as an example,

$$X_i := \left[\sum_{j\in\mathcal{L}_0}(-1)^{\langle d_i, j\rangle}\alpha_j\right]^2, \tag{45}$$

where $\{d_i\}$ is a set of independent random $2n$-bit strings. The expected value E of X_i satisfies the following bound:

$$E = \mathbb{E}(X_i) \geq L\epsilon_0^2. \tag{46}$$

Note that above we use the fact that any random strings are independent.

Moreover, since we want to show this term will be large with high probability, we should consider the random cross terms in each X_i, and that is

$$R_i = \sum_{\substack{u>v\\ u,v\in\mathcal{L}_0}} (-1)^{\langle d_i, u+v\rangle} 2\alpha_u\alpha_v, \tag{47}$$

where the order is in lexicographical order. Note the remaining part of X_i is a deterministic one, so we calculate only the variance for this R_i. It is straightforward that we have

$$\mathrm{Var}\,(R_i) = \mathbb{E}[R_i^2], \tag{48}$$

and the only contributed terms are those without random signs in R_i^2. For example, if we consider a specific u, v and the term $(\alpha_u\alpha_v) \times (\alpha_w\alpha_x)$ with some $(w, x) \neq (u, v)$ (assume $w > x$), this term will contribute to the expected value only if $w + x + u + v = 0$. Therefore, the only potential effective terms are those with four different Pauli error rates. Moreover, since $w + x + u + v$ must be in the null space of M_c^T according to Lemma 1, of which the size is $N/B = (1/\eta)e^{(1-\delta)n}$, we can estimate this probability using a balls and bins model.

Regardless of the square terms, the number of potential terms is $\binom{L}{4}$. Let G_i be the probability that the number of terms in bin 0 is at least a chosen constant η_0. Then G_i can be bounded as

$$\Pr\,(G_i) \leq \binom{L}{4} \times \left(\frac{B}{N}\right)^{\eta_0} = \binom{L}{4} \times \frac{e^{-(1-\delta)n\eta_0}}{\eta},$$

which is decaying exponentially with n. Given that the complementary event G_i^c happens and defining the set of contributing terms to be $\mathcal{A}_i$, the variance of R_i is the sum

$$\mathrm{Var}\,(R_i) = \sum_{\substack{(u,v,w,x)\in\mathcal{A}_i\\ u>v>w>x}} 8\alpha_u\alpha_v\alpha_w\alpha_x + \sum_{\substack{u>v\\ u,v\in\mathcal{L}_0}} 4\alpha_u^2\alpha_v^2.$$

Then by Cauchy-Schwarz we have $(\alpha_u\alpha_v)^2 + (\alpha_w\alpha_x)^2 \geq 2\alpha_u\alpha_v\alpha_w\alpha_x$. Averaging this over the other distinct partitions and using $|\mathcal{A}_i| \leq \eta_0$, we find

$$\mathrm{Var}\,(R_i) \leq \frac{\eta_0 + 3}{3} \sum_{\substack{u>v\\ u,v\in\mathcal{L}_0}} 4\alpha_u^2\alpha_v^2.$$

Now we can use the Hoeffding bound to obtain

$$\begin{aligned}
&\Pr\left(\frac{\|\mathbf{g}\|^2}{P_1} \leq 2T\gamma_2\nu^2\right)\\
&\quad \leq \Pr\left(\frac{\|\mathbf{g}\|^2}{P_1} \leq 2T\gamma_2\nu^2 \middle| G^c\right) + \Pr\,(G)\\
&\quad \leq e^{-3P_1(L\epsilon_0^2 - 2T\gamma_2\nu^2)^2/2(3+\eta_0)L^2\epsilon_0^4} + O\left[P_1 L^4 e^{-(1-\delta)n\eta_0}\right].
\end{aligned}$$

The last inequality uses the fact that

$$\mathbb{E}[X_i]^2 \geq \sum_{\substack{u>v\\ u,v\in\mathcal{L}_0}} 4\alpha_u^2\alpha_v^2 \geq 2L(L-1)\epsilon_0^4.$$

For any nontrivial signal, we have that $1 \leq L < P_1/2$. As long as $0 < \gamma_2 < \epsilon_0^2/2T\nu^2$ and choosing $\eta_0 = 6$, for any multiton we have

$$\begin{aligned}
f_4 \leq{}& e^{-(P_1/4)[\gamma_2^2/(1+4\gamma_2)]} + e^{-P_1(\epsilon_0^2 - 2T\gamma_2\nu^2)^2/6\epsilon_0^4}\\
&+ O\left[e^{-6(1-\delta)n}\right].
\end{aligned} \tag{49}$$

Therefore, $f_4 \leq O(1)e^{-O(n)}$.

Next we consider the *multiton–zeroton confusion* probability

$$\Pr(\mathrm{MZ}) := \Pr(\widehat{\mathfrak{B}} = \mathtt{z}|\mathfrak{B} = \mathtt{m}) + \Pr(\widehat{\mathfrak{B}} = \mathtt{m}|\mathfrak{B} = \mathtt{z}).$$

Denote the first term $\Pr(\widehat{\mathfrak{B}} = \mathtt{z}|\mathfrak{B} = \mathtt{m})$ by f_5 and the second $\Pr(\widehat{\mathfrak{B}} = \mathtt{m}|\mathfrak{B} = \mathtt{z})$ by f_6. For f_5, recognizing a multiton as a zeroton, we have the following inequality:

$$f_5 \leq \Pr\left[\frac{1}{P_1}\|\mathbf{U}\|^2 \leq T \times (1 + \gamma_1)\nu^2\right].$$

Note this probability can be analyzed in just the same way as f_4, and the only difference is that when we consider f_5, the α is just based on several underlying Pauli error rates without any subtraction, so $L \geq 2$ for this case. As long as $0 < \gamma_1 < \epsilon_0^2/T\nu^2$, then for any multiton we have the bound

$$\begin{aligned}
f_5 \leq{}& e^{-(P_1/4)[\gamma_1^2/(1+4\gamma_1)]} + e^{-P_1[(\epsilon_0^2 - T\gamma_1\nu^2)^2/6\epsilon_0^4]}\\
&+ O\left[e^{-6(1-\delta)n}\right],
\end{aligned} \tag{50}$$

so $f_5 \leq O(1)e^{-O(n)}$. Moreover, it is clear that the failure probability of recognizing a zeroton bin as a multiton bin, namely f_6, is smaller than f_3.

Next, consider the *index error* probability, and denote $\Pr\left(\widehat{\mathfrak{B}} = \mathtt{s}, \widehat{m} \neq m|\mathfrak{B} = \mathtt{s}, m\right)$ by f_7. This probability can be bounded by the probability of estimating a wrong index

$\widehat{m}$ and some Pauli error rate, and still passing the singleton verification

$$f_7 \leq \Pr[(\widehat{m} \neq m) \ \wedge \ (\widehat{m}, \widehat{p_{\widehat{m}}}) \text{ passes verification}]$$
$$\leq \Pr(\widehat{m} \neq m) \leq e^{-\frac{(\beta/\mathbb{P}-1)^2}{3}P_2}. \qquad (51)$$

Note the last inequality is just Eq. (37), and according to Remark 1, $\mathbb{P}_m < \frac{1}{2}$ for all $m \in \mathbb{F}_2^{2n}$ given that events D and V happen and we choose the maximum one to be $\mathbb{P}$.

Finally, let us consider the *value error* probability, and denote $\Pr(\widehat{\mathfrak{B}} = \mathtt{s}, |\hat{p}_{\hat{m}} - p_m| > \sqrt{4\nu^2} | \mathfrak{B} = \mathtt{s}, m, p_m)$ by f_8. Note that we have chosen $\sqrt{4\nu^2}$ as the error bound for the Pauli error rate, so similar to the *index error* probability, this f_8 can be bounded by the probability of estimating a noisy Pauli error rate and passing the singleton verification. We can loosen this bound by only considering the first event, and we obtain the inequality

$$f_8 \leq \Pr\left(|\widehat{p_{\widehat{m}}} - p_m| > \sqrt{4\nu^2} \middle| \widehat{m} = m\right) + \Pr(\widehat{m} \neq m)$$
$$\leq 2e^{-4P_1/2T} + e^{-[(\beta/\mathbb{P}-1)^2/3]P_2} \leq 3e^{-O(n)}. \qquad (52)$$

Note the middle inequality comes from a combination of Eqs. (36) and (37). According to Remark 1, we again have $\mathbb{P}_m < \frac{1}{2}$ for all $m \in \mathbb{F}_2^{2n}$ given that events D, V, and H happen and we choose the maximum one to be $\mathbb{P}$.

Following the taxonomy in Definition 3, we treat all of the failure cases of the bin-detector algorithm. Using the union bound, we can get the following inequality:

$$\Pr(E) \leq \sum_{i=1}^{8} f_i.$$

As we illustrate at the beginning of this proof, events D, V, and H have shown that the variance of the noise in each row of this bin is $T\nu^2$ from Lemma 7, and that this T is no more than 4 according to Lemma 9. Furthermore, they imply that θ_m is strictly smaller than $\frac{1}{2}$ for all $m \in \mathbb{F}_2^{2n}$ in Eqs. (38), (51), and (52). Since constraining the peeling graph G to obey this event is independent of the above analysis of the failure probabilities of the bin detector, it follows that

$$\mathrm{P}(E|D, V, H) \leq O(1)e^{-O(P_1)}.$$

This completes the proof. ■

X. CONCLUSION

We show that for sparse Pauli channels we can learn all the significant Pauli errors, even those associated with high-weight Pauli strings, using realistic experimental resources that scale with the sparsity of the Pauli errors rather than the dimension. In particular, we demonstrate that using only a few local two-qubit gates and a number of quantum experiments that scales linearly (with a factor of about 4), we can recover up to $4^{\delta n}$ of the largest error rates, where $\delta \lesssim 0.25$. Our numerical analysis indicates in the regime where $0.25 < \delta < 0.5$ we can still recover these errors with a number of experiments that only scales as $O(n^2)$.

We support these experimental protocols by defining and analyzing an algorithm with rigorous performance guarantees. This provable algorithm confirms that, with explicitly stated assumptions, high-precision reconstruction is possible when querying only a number of Pauli eigenvalues that scales like $O(sn)$. Moreover, the heuristic practical circuits used above are able to approximate the relevant noisy eigenvalue queries with sufficient precision ξ using only $O(n^2/\xi^2)$ measurements. These circuits exploit the protocols presented in Refs. [37,38] to learn up to 2^n commuting Pauli eigenvalues per experiment, and greatly reduces the required experimental resources.

This work provides an experimentally realizable method of identifying the relevant Pauli errors in large-scale quantum devices even if there are unexpected long-range correlations between the qubits. The ability to do so will be vital as we seek to mitigate the errors in such devices, to learn the noise patterns that exist when such devices are operated holistically and will allow better designing and tailoring of error correction and fault tolerance in such devices.

Many interesting open questions remain.

For example, what about very large-scale devices? The practicality of the algorithm in the regime of greater than (say) 30 qubits, where memory storage becomes an issue, could potentially be addressed as follows. We keep the protocol executed on the device identical as system size increases. However, we can take advantage of the fact that the WHT commutes with the marginalization of the observed probabilities and the process of fitting required to ascertain the SPAM-free eigenvalues (see Ref. [38]). The actual observations require only n bits of data to store. We can, therefore, marginalize the observations to obtain overlapping sets of 2^m eigenvalues (where we choose m to be the largest computationally tractable number for our classical computer). This will mean that we have multiple sets of 2^m bins, each potentially containing 2^{2n-m} Pauli error rates. Given this, the s-sparse assumption now becomes $s < 2^m$. It would be extremely interesting to implement this version of the algorithm on real data.

Another approach to dealing with very large-scale devices is to incorporate our algorithm as a subroutine in a larger algorithm that builds a globally consistent Pauli error distribution from estimations of marginal error rates. For example, as proposed in Ref. [37], one could efficiently estimate a Markov random field description of a Pauli channel if the underlying graphical model has bounded degree correlations. This idea has been performed experimentally on 14 qubits in Ref. [38]. We believe using the algorithm presented here would improve the estimation of the core subroutines and lead to better performance of the global reconstruction.

There are also several open mathematical questions about the reconstruction of sparse (or approximately sparse) Pauli channels. For example, it would be interesting to relax the random sparsity assumption on the support, or to allow for prior information in the distribution of the support. It would also be interesting to treat more general noise on the eigenvalue oracle. In particular, treating the case of noise with bounded variance seems to be the most relevant for providing recovery guarantees that relate to practical experimental capabilities. It might also be possible to weaken our assumptions about the signal-to-noise ratio. A lower bound would help to clarify where the limits are to these types of algorithms.

A further important open question is understanding the power of the structured circuits that we use for eigenvalue estimation. When using shallow depth Clifford circuits to prepare stabilizer bases for eigenvalue estimation, what recovery guarantees are possible? Is it still possible to efficiently reconstruct arbitrary sparse Pauli channels? Our heuristics suggest that pseudorandom and relatively shallow Clifford circuits allow sufficient randomness in the support that the algorithm can still have provable convergence guarantees, but it would be interesting to establish this rigorously.

Finally, the most important open problem is to use our algorithms on real experiments to characterize noise, improve calibration of a device, or customize an error-correction procedure.

ACKNOWLEDGMENTS

This work is supported by the US Army Research Office Grants No. W911NF-14-1-0098 and No. W911NF-14-1-0103, and the Australian Research Council Centre of Excellence for Engineered Quantum Systems (EQUS) Grant No. CE170100009.

APPENDIX: WALSH-HADAMARD ORDERING

In this paper we use a variant of the Walsh-Hadamard transform where the ordering is determined by the commutation relations between the Paulis. The natural (bit-wise) ordering of a WHT matrix can be calculated from the tensor product as

$$H_n\,(\text{natural ordering}) = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes n}. \tag{A1}$$

In this case, like the sequency order and dyadic order variants of the WHT, we reorder the columns of the transform matrix. Unless otherwise expressly noted, we use a WHT where the (i,j)th entry of the Hadamard transform matrix is given by $(-1)^{\langle i,j\rangle}$, where the inner product is the symplectic inner product introduced above. The advantages of using this variant of the WHT is that when it is used to transform eigenvalues to error rates and vice versa [Eqs. (5) and (6)], the position of each Pauli in the transformed vector remains constant.

[1] J. M. Martinis, Qubit metrology for building a faulttolerant quantum computer, npj Quantum Inf. **1** , 15005 (2015).

[2] M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y.-K. Liu, Efficient quantum state tomography, Nat. Commun. **1**, 149 (2010).

[3] B. P. Lanyon, C. Maier, M. Holzapfel, T. Baumgratz, C. Hempel, P. Jurcevic, I. Dhand, A. S. Buyskikh, A. J. Daley, M. Cramer, M. B. Plenio, R. Blatt, and C. F. Roos, Efficient tomography of a quantum many-body system, Nat. Phys. **13**, 1158 (2017).

[4] J. Helsen, X. Xue, L. M. Vandersypen, and S. Wehner, A new class of efficient randomized benchmarking protocols, npj Quantum Inf. **5**, 71 (2019).

[5] T. J. Proctor, A. Carignan-Dugas, K. Rudinger, E. Nielsen, R. Blume-Kohout, and K. Young, Direct Randomized Benchmarking for Multi-Qubit Devices, Phys. Rev. Lett. **123**, 030503 (2019).

[6] A. Erhard, J. J. Wallman, L. Postler, M. Meth, R. Stricker, E. A. Martinez, P. Schindler, T. Monz, J. Emerson, and R. Blatt, Characterizing large-scale quantum computers via cycle benchmarking, Nat. Commun. **10**, 5347 (2019).

[7] M. Sarovar, T. Proctor, K. Rudinger, K. Young, E. Nielsen, and R. Blume-Kohout, Detecting crosstalk errors in quantum information processors, arXiv:1908.09855 (2019).

[8] E. Bairey, I. Arad, and N. H. Lindner, Learning a Local Hamiltonian from Local Measurements, Phys. Rev. Lett. **122**, 020504 (2019).

[9] E. Bairey, C. Guo, D. Poletti, N. H. Lindner, and I. Arad, Learning the dynamics of open quantum systems from their steady states, New J. Phys. **22**, 032001 (2020).

[10] E. F. Dumitrescu and P. Lougovski, Hamiltonian assignment for open quantum systems, arXiv:1911.11092 [quant-ph] (2019).

[11] T. J. Evans, R. Harper, and S. T. Flammia, Scalable Bayesian Hamiltonian learning, arXiv:1912.07636 [quant-ph] (2019).

[12] H.-Y. Huang, R. Kueng, and J. Preskill, Predicting many properties of a quantum system from very few measurements, Nat. Phys. **16**, 1050 (2020).

[13] K. E. Hamilton, T. Kharazi, T. Morris, A. J. McCaskey, R. S. Bennink, and R. C. Pooser, Scalable quantum processor noise characterization, arXiv:2006.01805 [quant-ph] (2020).

[14] G. Torlai, C. J. Wood, A. Acharya, G. Carleo, J. Carrasquilla, and L. Aolita, Quantum process tomography with unsupervised learning and tensor networks, arXiv:2006.02424 [quant-ph] (2020).

[15] P. V. Klimov, J. Kelly, J. M. Martinis, and H. Neven, The snake optimizer for learning quantum processor control parameters, arXiv:2006.04594 [cs.DC] (2020).

[16] P. Aliferis and J. Preskill, Fault-Tolerant Quantum Computation against Biased Noise, Phys. Rev. A **78**, 052331 (2008).

[17] D. K. Tuckett, S. D. Bartlett, and S. T. Flammia, Ultrahigh Error Threshold for Surface Codes with Biased Noise, Phys. Rev. Lett. **120**, 050505 (2018).

[18] S. Puri, L. St-Jean, J. A. Gross, A. Grimm, N. E. Frattini, P. S. Iyer, A. Krishna, S. Touzard, L. Jiang, A. Blais, S. T. Flammia, and S. M. Girvin, Bias-preserving gates with stabilized cat qubits, arXiv:1905.00450 [quant-ph] (2019).

[19] J. Guillaud and M. Mirrahimi, Repetition Cat Qubits for Fault-Tolerant Quantum Computation, Phys. Rev. X **9**, 041053 (2019).

[20] D. K. Tuckett, A. S. Darmawan, C. T. Chubb, S. Bravyi, S. D. Bartlett, and S. T. Flammia, Tailoring Surface Codes for Highly Biased Noise, Phys. Rev. X **9**, 041031 (2019).

[21] D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, Fault-Tolerant Thresholds for the Surface Code in Excess of 5% Under Biased Noise, Phys. Rev. Lett. **124**, 130501 (2020).

[22] I. L. Chuang and M. A. Nielsen, Prescription for experimental determination of the dynamics of a quantum black box, J. Mod. Opt. **44**, 2455 (1997).

[23] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum State Tomography via Compressed Sensing, Phys. Rev. Lett. **105**, 150401 (2010).

[24] A. Shabani, R. L. Kosut, M. Mohseni, H. Rabitz, M. A. Broome, M. P. Almeida, A. Fedrizzi, and A. G. White, Efficient Measurement of Quantum Dynamics via Compressive Sensing, Phys. Rev. Lett. **106**, 100401 (2011).

[25] S. T. Flammia, D. Gross, Y.-K. Liu, and J. Eisert, Quantum tomography via compressed sensing: Error bounds, sample complexity, and efficient estimators, New J. Phys. **14**, 095022 (2012).

[26] A. V. Rodionov, A. Veitia, R. Barends, J. Kelly, D. Sank, J. Wenner, J. M. Martinis, R. L. Kosut, and A. N. Korotkov, Compressed sensing quantum process tomography for superconducting quantum gates, Phys. Rev. B **90**, 144504 (2014).

[27] A. Kalev, R. L. Kosut, and I. H. Deutsch, Quantum tomography protocols with positivity are compressed sensing protocols, npj Quantum Inf. **1**, 15018 (2015).

[28] C. A. Riofrío, D. Gross, S. T. Flammia, T. Monz, D. Nigg, R. Blatt, and J. Eisert, Experimental quantum compressed sensing for a seven-qubit system, Nat. Commun. **8**, 15305 (2017).

[29] S. T. Merkel, J. M. Gambetta, J. A. Smolin, S. Poletto, A. D. Córcoles, B. R. Johnson, C. A. Ryan, and M. Steffen, Self-Consistent Quantum Process Tomography, Phys. Rev. A **87**, 062119 (2013).

[30] B. M. Terhal, Quantum error correction for quantum memories, Rev. Mod. Phys. **87**, 307 (2015).

[31] E. Knill, Quantum computing with realistically noisy devices, Nature **434**, 39 (2005).

[32] J. J. Wallman and J. Emerson, Noise Tailoring for Scalable Quantum Computation via Randomized Compiling, Phys. Rev. A **94**, 052325 (2016).

[33] M. Ware, G. Ribeill, D. Riste, C. A. Ryan, B. Johnson, and M. P. da Silva, Experimental demonstration of Pauliframe randomization on a superconducting qubit, arXiv:1803.01818 (2018).

[34] E. Huang, A. C. Doherty, and S. Flammia, Performance of Quantum Error Correction with Coherent Errors, Phys. Rev. A **99**, 022313 (2019).

[35] S. Beale, J. Wallman, M. Gutiérrez, K. R. Brown, and R. Laflamme, Coherence in Quantum Errorcorrecting Codes, Phys. Rev. Lett. **121**, 190501 (2018).

[36] J. K. Iverson and J. Preskill, Coherence in logical quantum channels, arXiv:1912.04319 [quant-ph] (2019).

[37] S. T. Flammia and J. J. Wallman, Efficient estimation of Pauli channels, arXiv:1907.12976 (2019).

[38] R. Harper, S. T. Flammia, and J. J. Wallman, Efficient learning of quantum noise, arXiv:1907.13022 (2019).

[39] S. Kimmel, M. P. da Silva, C. A. Ryan, B. R. Johnson, and T. Ohki, Robust Extraction of Tomographic Information via Randomized Benchmarking, Phys. Rev. X **4**, 011050 (2014).

[40] R. Blume-Kohout, J. K. Gamble, E. Nielsen, K. Rudinger, J. Mizrahi, K. Fortier, and P. Maunz, Demonstration of qubit operations below a rigorous fault tolerance threshold with gate set tomography, Nat. Commun. **8** (2016).

[41] F. Arute *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[42] R. Harper, Github: Sparse Pauli Reconstruction (2020), https://github.com/rharper2/sparsePauliReconstruction.

[43] A. C. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss, in *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing - STOC '02* (ACM Press, 2002).

[44] R. Scheibler, S. Haghighatshoar, and M. Vetterli, A fast Hadamard transform for signals with sublinear sparsity in the transform domain, IEEE Trans. Inf. Theory **61**, 2115 (2015).

[45] X. Li, J. K. Bradley, S. Pawar, and K. Ramchandran, in *2014 IEEE International Symposium on Information Theory* (IEEE, Honolulu, Hawaii, USA, 2014).

[46] Choosing $m = O(1/\Delta)$ with $\Delta = 1 - \max_{j\neq 0} \lambda_j$ being the spectral gap of the channel is always sufficient, but this choice depends on the unknown channel. One could alternatively consider m to be a constant independent of the channel, but in doing so one would accept that any eigenvalue $\lambda > 1 - O(1/m)$ might incur an error of size $O(1/m)$. For our resource accounting, we adopt this latter perspective and treat m as a constant, $m = O(1)$.

[47] E. Kushilevitz and Y. Mansour, Learning decision trees using the Fourier spectrum, SIAM J. Comput. **22**, 1331 (1993).

[48] R. Koenig and J. A. Smolin, How to efficiently select an arbitrary Clifford group element, J. Math. Phys. **55**, 122202 (2014).

[49] S. Bravyi and D. Maslov, Hadamard-free circuits expose the structure of the Clifford group, arXiv:2003.09412 [quant-ph] (2020).

[50] W. Feller, , *An Introduction to Probability Theory and its Applications* (John Wiley & Sons, Hoboken, New Jersey, USA, 2008), Vol. 2.

[51] M. Sipser and D. A. Spielman, Expander codes, IEEE Trans. Inf. Theory **42**, 1710 (1996).

[52] W. Hoeffding, Inequalities for Sums of Bounded Random Variables, J. Am. Stat. Assoc. **58**, 13 (1963).