Partial Syndrome Measurement for Hypergraph Product Codes

Noah Berthusen and Daniel Gottesman

Joint Center for Quantum Information and Computer Science, NIST/University of Maryland, College Park, Maryland 20742, USA

Hypergraph product codes are a promising avenue to achieving fault-tolerant quantum computation with constant overhead. When embedding these and other constant-rate qLDPC codes into 2D, a significant number of nonlocal connections are required, posing difficulties for some quantum computing architectures. In this work, we introduce a fault-tolerance scheme that aims to alleviate the effects of implementing this nonlocality by measuring generators acting on spatially distant qubits less frequently than those which do not. We investigate the performance of a simplified version of this scheme, where the measured generators are randomly selected. When applied to hypergraph product codes and a modified small-set-flip decoding algorithm, we prove that for a sufficiently high percentage of generators being measured, a threshold still exists. We also find numerical evidence that the logical error rate is exponentially suppressed even when a large constant fraction of generators are not measured.

1 Introduction

Quantum computers have the theoretical potential to solve problems intractable for classical computers. However, realizing this potential requires dealing with the noise inherent in near- and far-term devices. One way of doing this is to redundantly encode the quantum information in a quantum error-correcting code (QECC) and manipulate the encoded states to do computation. The threshold theorem [1–3] guarantees that such a procedure can work for arbitrarily long circuits as long as the noise rate of the system is below some threshold. Although polylogarithmic overhead is needed in the general case, it was later shown that the use of asymptotically *good* quantum low-density parity-check (qLDPC) codes could reduce the overhead to a constant [4]. The question of whether such codes existed was unanswered until recently [5–8]; however, these constructions are currently more theoretical than practical.

When implementing QECCs on hardware it is especially advantageous to use one that is qLDPC, as its stabilizer generators act on a constant number of qubits, and its qubits are involved in a constant number of stabilizer generators. For certain architectures, such as nuclear magnetic resonance or superconducting qubits, another desirable code property is *locality*. A code is considered local in $\mathbb{Z}^2$ if, when embedded in a grid of size $\sqrt{n} \times \sqrt{n}$, its generators act on qubits within a ball of constant radius. Recently, a popular choice when implementing a code family with these properties has been the surface code and its variations [9, 10]. While it has local, weight-four generators and a favorable $\Theta(\sqrt{n})$ distance scaling, the surface code has a rate, k/n, which tends to zero as n approaches infinity. A qLDPC code family that avoids this issue is hypergraph product (HGP) codes [11]. This construction has the same $\Theta(\sqrt{n})$ distance scaling, but now with a constant rate; the tradeoff, however, is that the stabilizer generators of HPG codes are very nonlocal. It was shown in Refs. [12, 13] that there is an intimate relationship between locality and the corresponding code parameters. In particular, the distance d for a local code in $\mathbb{Z}^2$ is bounded above by $O(\sqrt{n})$, and the number of logical qubits k obeys the relation $kd^2 = O(n)$. As such, the surface code saturates these bounds. Later work [14, 15] more precisely quantified the amount of nonlocality required to surpass them.

Noah Berthusen: nfbert@umd.edu

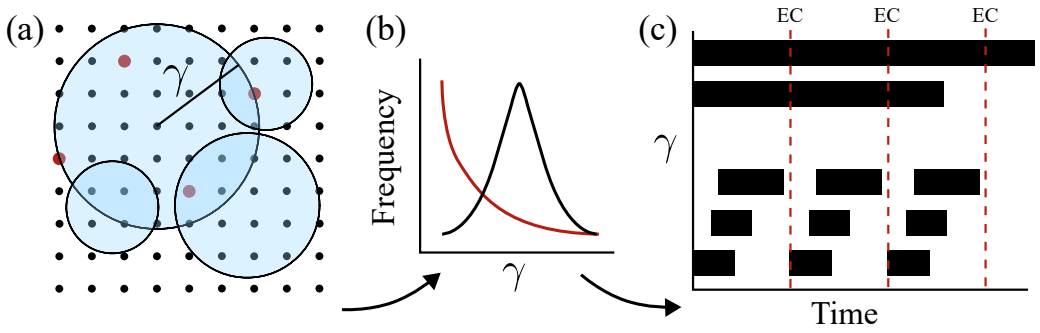

Figure 1: Overview of the stacked model. (a) After embedding a quantum code into $\mathbb{Z}^2$, each stabilizer generator has a parameter γ that denotes the radius of the ball containing the qubits in its support. (b) Two possible distributions of γ over the set of generators. The most advantageous distributions for this scheme are those where the relative frequency decays exponentially with increasing γ (red curve). (c) An example *schedule* for the generator measurements. The syndrome extraction circuits for the smaller generators are able to be prepared quickly, and so their syndromes are available during every round of error correction (red dashed lines). The larger generators require more time to build their syndrome extraction circuits, so this is done over a period of time that may stretch over several error correction rounds. More practically, priority is given to the smaller generators, and after completing them, the larger generators are worked on using any remaining time before an error correction round.

In this paper, we show through analytic and numerical evidence that repeated quantum error correction with HGP codes still provides a threshold even when a constant fraction of generators are measured only after many rounds of error correction. This result suggests that it may be possible to build a fault-tolerant quantum computer with nonlocal qLDPC codes on architectures restricted to 2D local gates with a procedure based on the *stacked model* [14]. After embedding a QECC in a grid of size $\sqrt{n} \times \sqrt{n}$, the stabilizer generators are partitioned into a stack of layers based on the radius of the ball containing the qubits they act on. The bottom layer of the stack contains local generators, and as we move up the stack, the interaction radius increases while the number of generators of that size decreases. Ideally, we can use codes which when embedded into $\mathbb{Z}^2$ have the property that the number of generators decreases exponentially with increasing radius; that is, a (large) constant fraction of the generators act on qubits within a support of constant radius. It was also shown in Ref. [14] that any code constrained to the above model has a distance that is bounded by $\widetilde{O}(n^{2/3})$ and obeys the relation $k^3 d^4 = \widetilde{O}(n^5)$. HGP codes satisfy this trade-off. [1]

The stacked model has a natural application when performing fault-tolerant quantum computations. To convert a quantum circuit into a fault-tolerant version, the qubits are first encoded in some QECC, and then each operation in the original circuit is replaced with a fault-tolerant *gadget*. Errors may still occur in the individual gates, so after each time step of the circuit, a round of fault-tolerant error correction is performed. To do this, the eigenvalues of the stabilizer generators of the code are measured to learn the syndrome, which is then used by a decoder to deduce and correct the error. Measurement of the generators at the bottom of the stack takes constant time, since they are local. The corresponding syndrome information is then available during every round of error correction. As we move up the stack, the interaction radius increases. The important distinction to make is that while the generators are nonlocal, we are still measuring them with only 2D local gates, and so extracting these syndromes takes longer than for local generators. These nonlocal generators are measured less frequently than those lower in the stack, and their syndrome is not always available. This scheme is depicted in Fig. 1.

Several recent works have provided evidence against the possibility of doing error correction on architectures restricted to 2D local gates. Delfosse *et al.* [16] investigated the problem of performing syndrome extraction circuits of HGP codes using 2D local gates and classical communication and presented numerical simulations suggesting that the resulting overhead was pro-

[1] $\widetilde{O}(\cdot)$ is a variant of big O notation that ignores log factors. $f(n) \in \widetilde{O}(h(n))$ is equivalent to $\exists k : f(n) \in O(h(n) \log^k n)$.

hibitive. However, when considering the same problem in the context of the stacked model, it is possible that significant reductions in overhead could be gained by not measuring the generators with a large interaction radius every error correction cycle, since most of the work required is due to these very nonlocal generators.

We can roughly approximate the amount of work required to perform syndrome measurement using 2D local gates by estimating the number of SWAP gates in the extraction circuits. For a generator with interaction radius γ, the total number of SWAP gates needed to perform the syndrome measurement is proportional to γ. As a concrete example, consider a qLDPC code on $n = 100,000$ qubits which when embedded into $\mathbb{Z}^2$ results in a generator distribution where the number of generators decays exponentially with increasing γ. Drawing $O(n)$ generators from this distribution and summing the radii of the smallest 90% is $\sim 3\%$ of the total sum across all generators. Thus, we can estimate that the syndrome of these smallest 90% of generators can be obtained using only $\sim 3\%$ of the SWAP gates required to perform all of the syndrome measurements. Obtaining the remaining 10% of the syndromes requires the majority of the work, but these circuits are built up over time (see Fig. 1(c)), allowing for a significant portion of the full error correction capabilities to be available during each error correction round. Although the resulting logical error rates will be strictly larger than when using a full syndrome, the reductions in overhead may outweigh the increases in the logical error rate. A rigorous investigation of this question is the focus of further research [17].

Baspin *et al.* [18] provide further evidence against 2D local implementations of qLDPC codes by deriving bounds on the amount of overhead needed to perform error correction at a given logical error rate. They show that the restriction to 2D local gates incurs polynomial overhead. However, they also note that their definition of error rate is very restrictive and that computations not satisfying this definition might not obey the overhead bound. It therefore remains possible that the stacked model could be used to perform these computations with constant overhead. Apart from this brief discussion, we do not rigorously prove the feasibility of the stacked model as a whole or refute the claims put forth by these authors. This work only addresses the question of partial syndromes and their effect on performing error correction in the phenomenological noise model.

The remainder of the work is structured as follows. In Section 2 we give a brief review of classical and quantum coding theory and introduce the families of codes relevant to this work. Section 3 introduces the idea of masking and contextualizes it with respect to the stacked model. In Section 4, we apply previous results to provide some analytical bounds on using masking during multi-round error correction. Section 5 provides empirical evidence to suggest that the analytical thresholds are better in practice. Finally, we conclude in Section 6 with a summary and discussion of the remaining problems.

2 Background

2.1 Classical and Quantum Codes

An $[n, k, d]$ binary linear code $\mathcal{C}$ encodes k classical bits in a k-dimensional subspace of the n bit, n-dimensional space, $\mathbb{F}_2^n$. Codewords are the binary vectors $v \in \mathbb{F}_2^n$ that satisfy the equation $H \cdot v = \mathbf{0}$, where H is a full rank binary matrix of size $(n-k) \times n$ called the *parity check matrix*. The distance d of a linear code is the minimum Hamming weight of a nonzero codeword. We can also represent the code $\mathcal{C}$ with its *Tanner graph*, a bipartite graph $G = (V \sqcup C, E)$ whose biadjacency matrix is H.

An $[[n, k, d]]$ quantum error correcting code $\mathcal{Q}$ encodes k logical qubits into a 2^k-dimensional subspace of the n qubit, 2^n-dimensional Hilbert space, $(\mathbb{C}^2)^{\otimes n} = \mathbb{C}^{2^n}$. A commonly used class of QECCs are *stabilizer codes* [19, 20]. A stabilizer code is defined by its *stabilizer* S, consisting of elements of the Pauli group

$$\mathcal{P}_n = \{I, X, Y, Z\}^{\otimes n} \times \{\pm 1, \pm i\}, \tag{1}$$

whose action is the identity on the codewords of $\mathcal{Q}$. To have a codespace at all, we require that $-I \notin S$ and that S forms an abelian subgroup of $\mathcal{P}_n$. Denote by $N(S)$ the normalizer of S, the set of Paulis that commute with everything in the stabilizer, $N(S) = \{N \in \mathcal{P}_n \mid [N, M] = 0 \ \forall M \in S\}$. The distance d of $\mathcal{Q}$ is then defined to be the minimum weight of an operator in $N(S) \setminus S$. S is generated by $m = n - k$ independent *stabilizer generators* $S = \langle S_1, ..., S_m \rangle$, which when

measured provide an error syndrome of length m used to deduce the error. We note that the syndrome labels the 2^m cosets of $P_n/N(S)$.

The *binary symplectic representation* of a Pauli $P \in \mathcal{P}_n/\{\pm 1, \pm i\}$ is a bitstring consisting of two n-bit binary vectors, $(x|z) \in \mathbb{F}_2^{2n}$. The ith component of x is 0 if P acts on qubit i with I or Z and 1 if P acts on qubit i with X or Y. Similarly, the ith component of z is 0 if P acts on qubit i with I or X and 1 if P acts on qubit i with Z or Y. This transformation allows us to use techniques from classical coding theory on QECCs. In particular, we can represent the stabilizer generators as a $m \times 2n$ binary parity check matrix, H. If we consider then some error $E = (x|z) \in \mathbb{F}_2^{2n}$, the corresponding syndrome is $\sigma(E) = H \cdot E$, where multiplication and addition are performed over $\mathbb{F}_2$.

CSS codes [21] are a subclass of stabilizer codes where the stabilizer generators consist entirely of tensor products of X and I or Z and I. As such, these codes have parity check matrices of the symplectic form $H = \left(\begin{smallmatrix} H_Z & 0 \\ 0 & H_X \end{smallmatrix}\right)$, with $H_Z \cdot H_X^T = H_X \cdot H_Z^T = \mathbf{0}$ to enforce the abelian structure of S. In this form, it can be seen that decoding CSS codes can be broken down into decoding the two classical codes with parity check matrices H_Z and H_X separately, where H_Z corrects bit-flip errors and H_X corrects phase-flip errors. In this case, separate syndromes are needed to decode an error $E = (x|z)$,

$$\sigma(E) = (\sigma_Z(x), \sigma_X(z)) = (H_Z \cdot x, H_X \cdot z). \quad (2)$$

In this work, it may be unclear with respect to which stabilizer generators a syndrome is measured. Where clarification is needed, we slightly abuse notation and write a syndrome taken from a subset of the stabilizer $U \subseteq S$ as $\sigma_U(E)$. Using this notation, we do not explicitly specify the type of error we are measuring, but in all cases we will only consider one type. We let the Tanner graph of a CSS code, $G = (V \sqcup C_X \sqcup C_Z, E)$, to be the bipartite graph defined by its parity check matrix in symplectic form. The two disjoint sets of check nodes, C_X, C_Z, correspond to the $X-$ and $Z-$type stabilizer generators, respectively.

A classical or quantum code is considered a *low density parity check* code if the weights of the rows and columns of its parity check matrix are bounded by a constant. Specifically, an $[[n, k, d]]$ stabilizer code is considered a $(\Delta_V, \Delta_C)-$qLDPC code if, for some constants Δ_V and Δ_C, each qubit is involved in at most Δ_V stabilizer generators and each generator measures at most Δ_C qubits. We can equivalently say that the Tanner graph has bit node degree bounded by Δ_V and check node degree bounded by Δ_C.

2.2 Quantum Expander Codes

Hypergraph product (HGP) codes [11] are CSS type codes made by taking the graph product of two classical codes $\mathcal{C}_1, \mathcal{C}_2$. When $\mathcal{C} := \mathcal{C}_1 = \mathcal{C}_2$ is a binary linear code with parameters $[n, k, d]$ and a full-rank parity check matrix, the parameters of the resulting hypergraph product code are $[[n^2 + (n-k)^2, k^2, d]]$. If the input code is $(\Delta_V, \Delta_C)-$LDPC with $\Delta_V \leq \Delta_C$, then the resulting quantum code is $(2\Delta_C, \Delta_V + \Delta_C)-$qLDPC. Furthermore, when the base code is replaced with a classical expander code [22], the resulting quantum code is deemed a *quantum expander code* and is equipped with a linear time decoding algorithm which we now describe.

The small-set flip (SSF) decoding algorithm [23] aims to imitate the classical flip decoding algorithm used to decode classical expander codes. Let $\mathcal{F}$ be the set of powersets of qubits in $X-$type generators and let E be the initial $X-$type error. A single round takes as input a guessed error $\hat{E}_i$ and the syndrome of the remaining error $\sigma_i := \sigma_Z(E \oplus \hat{E}_i)$. The decoder then goes through all 'small-sets' $f \in \mathcal{F}$ and finds the one that when flipped maximizes the decrease in syndrome weight, which is then applied to the guessed error for the next round. The algorithm succeeds if the final error has zero syndrome and is not a logical operation; otherwise, it fails. In other words, decoding is considered a success if the guessed error $\hat{E}$ is equivalent to the actual error E, that is $E \oplus \hat{E}$ belongs to the stabilizer group. The success of the decoder is guaranteed for errors of size less than the distance, as well as random errors of linear size [24] provided the underlying classical codes are sufficiently expanding.

The complete decoding procedure is listed as pseudo-code in Algorithm 1. It takes as input a tuple (E, D), where $E \subseteq V$ is an $X-$type error, and $D \subseteq C_Z$ is a potential syndrome error—that is the algorithm runs instead on the syndrome where some values have been flipped, $\sigma_Z(E) \oplus D$. We make one small change to the algorithm for the purposes of using it in the context of the

Algorithm 1: Small-set flip decoding algorithm [23]

Require: (E, D)

while $\exists F \in \mathcal{F} : |\sigma_i| - |\sigma_i \oplus \sigma_Z(F)| > 0$ **do**
$F_i = \max_{F \in \mathcal{F}} \frac{|\sigma_i| - |\sigma_i \oplus \sigma_Z(F)|}{|F|}$
$\hat{E}_{i+1} = \hat{E}_i \oplus F_i$
$\sigma_{i+1} = \sigma_i \oplus \sigma_Z(F_i)$
$i = i + 1$
end while

return $\hat{E}_i$

stacked model. Specifically, we exchange using the full syndrome for one taken from some subset of the stabilizer generators $U \subseteq S$. We still search through every opposite type generator when looking for small-sets F to flip; however, the effect of flipping will only be visible on the restricted set of generators $\sigma_U(F)$. We overload the meaning of having the input (E, D) when $D \subseteq C_Z$ is interpreted as a mask, in which case the available syndrome is $\sigma_D(F)$. The chosen interpretation will be clear from context.

2.3 Fault-Tolerance

A quantum circuit is considered fault-tolerant if it prevents errors from propagating throughout the circuit; in this way, it keeps the size of the error manageable for the QECC. We can convert a circuit into a fault-tolerant version by replacing each element of the original circuit with a fault-tolerant *gadget* performing an equivalent operation on the encoded state. Fault-tolerant circuits can be naturally broken down into time steps, where a single time step consists of gadgets applied in parallel followed by error correction.

To investigate how an error propagates throughout a fault-tolerant circuit, we abstract the above model and instead work with the procedure described in Algorithm 2. For the purposes of analysis and simulation, we condense all gadgets, except error correction, into a single event that has an error with probability p_{phys}. We also assume that the error correction itself is ideal and that there is no syndrome error, except the artificially imposed error coming from the generators that have been masked with probability p_{mask}, which we now define. We later discuss how to

Algorithm 2: A simplified fault-tolerance scheme

Apply a mask D with probability p_{mask}

for t = 1, ..., τ **do**
Generate an error F_t with probability p_{phys} and apply F_t to the current error:

$$E'_t := F_t \oplus E_{t-1}$$

Run Algorithm 1 on the input (E_t, D) and correct using the decoded error $\hat{E}_t$:

$$E_t := E'_t \oplus \hat{E}_t$$

end for

Generate an error F_t with probability p_{phys} and apply to the current error:

$$E_\tau := F_\tau \oplus E_{\tau-1}$$

Run Algorithm 1 on the input $(E_T, \varnothing)$

make this scheme more realistic, but for the purposes of determining the effects of performing error correction with partial syndromes this simplified model is sufficient.

3 Syndrome masking

The notion of *masking* has recently been introduced as a way of describing fault-tolerant protocols for space-time codes [25]. We use the same idea here, although in a different context. An element of the stabilizer is considered masked if we cannot measure its eigenvalue during an error correction round. We follow the definition from [25] and define two subgroups of the stabilizer, U and T, where $U \subseteq T \subseteq S$. The *always unmasked* subgroup, U, are the stabilizers whose eigenvalues can be measured in a constant number of rounds, whereas the *temporarily unmasked* subgroup, T, are the stabilizers whose eigenvalues can be measured in a number of rounds that can scale with the size of the code, n. In general, it could be the case that $T \subsetneq S$ where the set $S \backslash T$ contains stabilizers that cannot be measured on any time scale. In this work, we consider the case $U \subset T = S$. The subgroups form valid stabilizer codes, and as

such can be described by their parameters. Defining k for these codes has no real meaning since logical information is not being stored in the subspace; however, we can define the corresponding distances, where

$$d_U = \min |N(U) \setminus S| \quad d_T = \min |N(T) \setminus S|. \quad (3)$$

In other words, d_U (d_T) is the weight of the smallest Pauli operator outside of the full group that has zero syndrome when measuring only the stabilizer generators of U (T). We call d_U and d_T masked distances, whereas d is the unmasked distance. Note that $d_U \leq d_T \leq d$.

Since not every generator is measured, the resulting syndrome may have less information about the error than would otherwise be available if the full set of stabilizer generators were measured. For any number of masked generators, there is a set of *invisible* errors that have zero syndrome on the generators of U (or T) while having a nonzero syndrome in S. In particular, the new normalizer $N(U)$ contains $N(S)$ as well as all cosets of $\mathcal{P}_n/N(S)$ labeled with undetectable error syndromes. Furthermore, errors that were previously correctable may no longer be uniquely identifiable with the syndrome of U or T. Note that errors with a zero syndrome for U do not immediately cause logical errors, unlike errors with a zero syndrome for all of S. If an error has a nonzero syndrome for T, it will eventually be detected, once the generators of $T \setminus U$ are unmasked. The risk is that such errors will accumulate over time and become logical errors before they can be corrected.

3.1 Masking and the Stacked Model

Identifying which layers of the stack are available during an error correction round corresponds to specifying the temporarily unmasked subgroup, T_t, at each time step in the circuit, $t = 1, ..., \tau$. The always unmasked subgroup, $U \subset T_t$, is static over the execution of the circuit and so can be specified at the beginning. This set contains all local generators, as their eigenvalues can be measured in constant time. T_t will contain U as well as any additional layers that have completed syndrome extraction between time $t-1$ and t. Since, in general, we want to measure all generators throughout the course of the circuit, $\bigcup_t T_t = S$; however, it may not be the case that any one time step has all generators available.

An equivalent interpretation is to specify $S \setminus T_t$, the set of generators whose eigenvalues are not available during time step t. For the remainder of the work, we consider 'applying' a mask D to be specifying this set, $S \setminus T_t$.

4 Analytic results

In this section, we consider previous results on HGP codes and the SSF decoder in the context of masking in a multi-round error correction procedure. We consider qubit errors and syndrome masks that follow a local stochastic noise model.

Definition 1. *(Local stochastic error model). We say that an error (E, D) is local stochastic if there are error parameters (p_{phys}, p_{synd}) such that for any F and L, $\Pr[F \subseteq E, L \subseteq D] \leq p_{phys}^{|F|} p_{synd}^{|L|}$.*

HGP codes in conjunction with the SSF decoder have several desirable properties that make them a strong contender for fault-tolerance with constant overhead. Most relevant to us is the fact that they can tolerate random qubit errors and syndrome errors of linear size, as stated in the following theorem.

Theorem 1. *(modified from Fawzi, Grospellier, Leverrier [26]). There exists a non-zero constant $p_0 > 0$ such that the following holds. Suppose that the error (E, D) each satisfy a local stochastic noise model with parameters p_{phys} and p_{synd} where $p_{phys} < p_0$ and $p_{synd} < p_0$. If we run Algorithm 1 on the input (E, D) then there exists a random variable $E_{ls} \subseteq V$ with a local stochastic distribution with parameter $p_{ls} := p_{synd}^{\Omega(1)}$ such that:*

$$\Pr\left[E_{ls} \text{ and } E \oplus \hat{E} \text{ are not equivalent}\right] \leq e^{-\Theta(\sqrt{n})} \quad (4)$$

In the analysis for the above theorem, Fawzi *et al.* consider an error D in the syndrome to be a subset of the stabilizer generators whose measurement results have been flipped. Very briefly, the argument requires that the syndrome error does not form clusters on the syndrome adjacency graph [4] for it to be tolerable. As such, p_0 must be below the percolation threshold of the syndrome adjacency graph of $\mathcal{Q}$. This value is a constant that depends only on Δ_V and Δ_C of the code. We can turn the result of a masked measurement into the above form

by randomly assigning measurement outcomes to the generators included in the mask. Thus, in this context, we can say that Theorem 1 holds when a mask—turned syndrome error—D satisfies a local stochastic noise model with parameter $p_{\text{mask}} < p_0$.

The above analysis is sufficient in the case where we do a single round of masked error correction; however, when we use the same mask over several rounds, we have to be more careful about accounting for the correlations between error sources. Following the notation of Algorithm 2, in each round t we have the syndrome error from the mask D, any error that was not fully corrected in the previous round E_{t-1}, and a new error F_t. When considered individually, all three error sources are local stochastic described by parameters $p_{\text{mask}}, p_{\text{res}}$, and p_{phys}, respectively. When looked at together, the new error and the syndrome error are bounded by

$$\Pr[F \subseteq E \text{ and } L \subseteq D] \leq p_{\text{phys}}^{|F|} p_{\text{mask}}^{|L|} \tag{5}$$

and similarly for the residual error and the new error, as per the definition of a locally stochastic error. However, we would expect to see correlations arise between the residual error and the syndrome error over the rounds, and so together they no longer obey a local stochastic noise model. Instead, they are bounded by:

$$\Pr[F \subseteq E \text{ and } L \subseteq D] \leq \min\{p_{\text{res}}^{|F|},\ p_{\text{mask}}^{|L|}\} \tag{6}$$

When $\max\{p_{\text{res}}, p_{\text{mask}}\} < p_0$, the threshold from Theorem 1, we can say that the probability of clustering is at most $e^{-\Theta(\sqrt{N})}$ by plugging the error bound in Eq. (6) into Theorem 17 ([24]). With this, we can apply Lemma 26 ([26]) to bound the probability of the residual error obeying a local stochastic distribution, $\Pr[S \subseteq E_{\text{ls}}]$. Besides the requirement that $E \cup D$ forms clusters with low probability, we need that $\Pr[L \subseteq D] \leq p_{\text{mask}}^{|L|}$. Since we assumed that the mask was chosen according to a local stochastic error model, this statement is satisfied for all rounds $t \leq \tau$. We are then able to apply Theorem 1 in an iterative manner, yielding the following result.

Theorem 2. *(Grospellier [27]). Let p_0 be the threshold of Theorem 1, and let p_{mask} and p_{phys} be such that:*

$$p_{mask} < \left(\frac{p_0}{2}\right)^{\Omega(1)} \quad \text{and} \quad p_{phys} < \frac{p_0}{2}. \tag{7}$$

Then Algorithm 2 fails with probability at most $(\tau+1)e^{-\Theta(\sqrt{n})}$.

If the conditions for Theorem 2 are satisfied, then we can make the failure probability for the procedure arbitrarily small by using larger codes. This result is perhaps surprising given the following two claims:

Claim 1. *Applying a random mask D with parameter p_{mask} to a qLDPC code $\mathcal{Q}$ results in a code $\mathcal{Q}' = \mathcal{Q}(S \backslash D)$ whose Tanner graph has the following degree distribution:*

$$\Pr(\deg(\mathcal{Q}'_{|v}) = i) = \binom{\deg(\mathcal{Q}_{|v})}{i} p_{mask}^{\deg(\mathcal{Q}_{|v})} (1-p_{mask})^i \tag{8}$$

Here, we use the notation $\deg(Q_{|v})$ to mean the degree of node v in $\mathcal{Q}$. Since we assume $\mathcal{Q}$ to be LDPC, $\deg(v)$ is bounded by a constant Δ_V for all v, but the values may differ between vertices.

Corollary 1. *Randomly masking a constant fraction of generators results in a masked distance of $d_U = 1$ whp.*

Proof. Applying Claim 1 with $p_{\text{mask}} = O(1)$ gives a degree distribution where $\Pr(\text{Degree of qubit } v = 0) = p_{\text{mask}}^{\deg(\mathcal{Q}_{|v})} = \Omega(1)$ for all qubits. In this case, an error on such a qubit has zero syndrome on the remaining unmasked generators, U. As this error would not be an element of the stabilizer, $d_U = 1$. □

Although the always unmasked subgroup U has a bad distance d_U with high probability, it is capable of performing enough error correction in the intermediate steps to prevent the accumulation of errors. When the full set of stabilizer generators are unmasked at the end of the multi-round decoding procedure, it is then likely able to correct any residual errors. As we will now show, we see similar behavior at masking percentages well above what is guaranteed analytically.

5 Numerical simulations

In this section, we report on the results of numerical simulations of a multi-round decoding protocol as described in Section 2.3. Previous work has investigated the single-round performance of HGP codes using a variety of decoders [28–31]

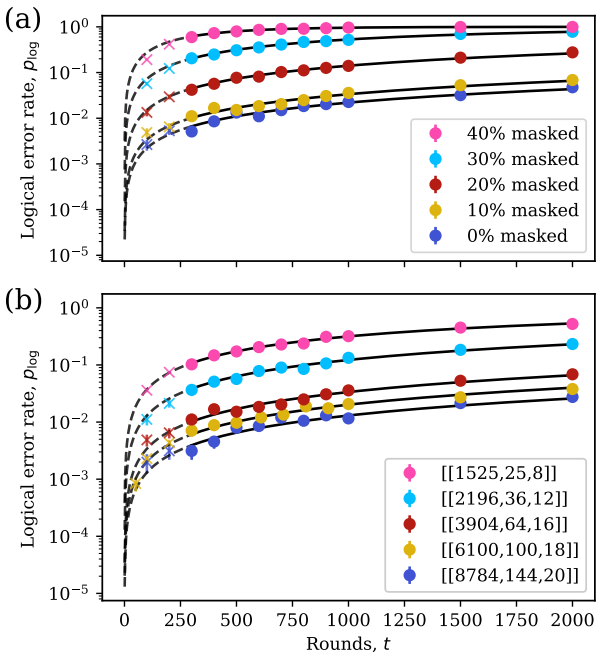

Figure 2: (a) Semilog plot of logical error rate as a function of the number of rounds for a $[[3904, 64, 16]]$ code and the simple unmasking schedule. (b) Logical error rate as a function of rounds across the $(12, 11)$-qLDPC code family with fixed $p_{\text{mask}} = 10\%$ and the simple unmasking schedule. Both panels include fits of Eq. (9), for which we only include data with $t \geq 300$.

and gives evidence for thresholds at near-state-of-the-art error rates. Here, we provide alternative evidence of exponential error suppression in both masked and unmasked cases following the methodology of Ref. [32].

We investigate a family of HGP codes constructed from a single classical expander code family and decode them using the small-set flip decoding algorithm. HGP codes—being CSS codes—can have bit- and phase-flip errors decoded independently. Furthermore, HGP codes constructed from a single base code have equivalent parity check matrices H_X, H_Z, and therefore, without loss of generality, we focus on the problem of decoding $X-$type errors. The specific quantum code family considered is constructed from a base (5, 6)-LDPC code family, resulting in (12, 11)-qLDPC codes. These codes have a rate of $1/61 \approx 0.016$. The results presented here correspond to a specific (un)masking *schedule*, which is a potential modification of Algorithm 2 and a way of specifying when and by how much to apply a mask to the syndrome. In particular, we study the following two models:

- *Simple scheduling.* Apply a mask D with a masking percentage of p_{mask} to use for all τ error correction rounds. After τ rounds, remove the mask completely and perform one error correction round with the fully unmasked syndrome.

- *Iterative scheduling.* Apply a mask D with masking percentage p_{mask}. After a multiple of 10^{t-1} rounds, for $t > 0$, remove $1 - 10^{-(t-1)}\%$ of the mask. For each of these instances, remove the same portion of the mask each time. On rounds $10^{t-1} + 1$, all generators that were temporarily unmasked in the previous round are re-masked until another 10^{t-1} rounds have passed. After τ rounds, again remove the mask completely and perform one round of error correction.

In Fig. 2(a) we show the logical error rate, $p_{\log}$, as a function of the number of rounds for a $[[3904, 64, 16]]$ code and the simple unmasking schedule. Data is obtained by running Algorithm 2 for a fixed number of rounds with an error rate of $p = 0.001$ while varying the masking percentage, p_{mask}, and then recording the percentage of samples that end with a logical error. A sample is considered to end with a logical error if the final state is not equal to the initial state, up to stabilizer elements. We extract the logical error per round, ϵ_L, by fitting the data to the exponential

$$p_{\log} = 1 - (1 - \epsilon_L)^t. \tag{9}$$

The error bars on the fits are taken from the standard error of sampling a binomial distribution, $\sqrt{p_{\log}(1 - p_{\log})/N}$.

In the bottom panel of Fig. 2, we now fix $p_{\text{mask}} = 0.1$ and show the performance of the simple unmasking schedule across the code family. The codes are labelled with their parameters as described in Section 2.2. While finding the distance of a code is generally hard, we are able to exhaustively search through the codewords of the base classical code to determine the distance of it, as well as the corresponding HGP code. Here, we observe even spacing between curves on the semilog plot showing exponential error suppression with code size. This behavior is more easily seen as the linear downwards trend in Fig. 3, which we now more precisely quantify.

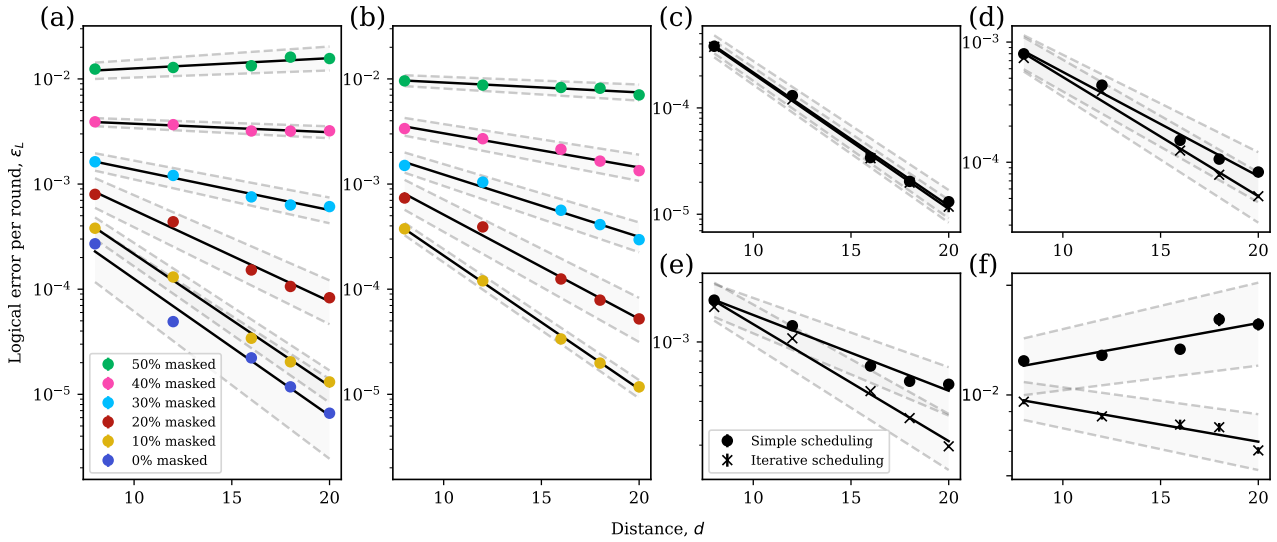

Figure 3: (a) Semilog plot of logical error rate per round, ϵ_L, as a function of code distance for the simple unmasking schedule and an error rate of $p = 0.001$. The fits are of a linearized Eq. (10) with log ϵ_L. (b) Similar results for iterative scheduling. Note that we do not include 0% masking in this case because it is equivalent to the simple schedule. Panels (c)-(f) plot the same data from panels (a)-(b) and provide easier comparisons between the simple (dot markers) and iterative (x markers) scheduling for $p_{\text{mask}} = \{10\%, 20\%, 30\%, 50\%\}$, respectively. The shaded region for all panels indicates error bars for C and Λ.

p_{mask}	Simple scheduling	Iterative scheduling
0%	1.820 ± 0.046	-
10%	1.782 ± 0.019	1.794 ± 0.010
20%	1.490 ± 0.026	1.579 ± 0.026
30%	1.193 ± 0.015	1.314 ± 0.018
40%	1.038 ± 0.007	1.161 ± 0.015
50%	0.956 ± 0.014	1.044 ± 0.009

Table 1: Extracted values of Λ for different masking percentages and schedules.

We can relate a code family and values for logical error per round with an exponential error suppression factor Λ. For simple models, the equation

$$\epsilon_L = \frac{C}{\Lambda^{(d+1)/2}}, \tag{10}$$

where C is a fitting constant and d is the distance of the code, heuristically describes this relationship well. In Fig. 3(a) and (b), we show the logical error per round as a function of code distance for the simple and iterative schedules, respectively. For each masking percentage, we fit a linearized Eq. (10) with log ϵ_L to obtain Λ. These values are listed in Table 1. A value of $\Lambda > 1$ is a clear indication of operating below the threshold, as increasing the code size gives an exponential decrease in the logical error rate per round. For simple scheduling, we find that for masking percentages below 50%, Λ is in this regime. Increasing p_{mask} decreases Λ, and between 40% and 50% we see a transition where $\Lambda < 1$. In this case, it is no longer advantageous to increase the code size, as it actually causes more logical errors to occur.

The results of the iterative unmasking schedule are shown in Fig. 3(b), where we find that it outperforms the simple schedule. For smaller masking percentages, it is not as advantageous to use a schedule with more unmasking, as there is less difference in performance between small masking percentages and completely unmasking (see Fig. 2(a)). However, larger masking percentages appear to benefit more from using a more frequent unmasking schedule. In fact, with iterative scheduling, it is now the case that 50% masked is back in the $\Lambda < 1$ regime, although with very little error suppression. Fig. 3(c)-(f), highlights this difference between schedules.

In both cases, we find that the results exceed the guarantees provided by Theorem 2. We find that the percolation threshold of this family of $(12, 10)-$qLDPC codes is around 2%; however, we see exponential error suppression at error rates up to $\sim 50\%$, well above this threshold.

5.1 2D Hyperbolic Surface Codes

As a comparison, we benchmark the performance of a 2D hyperbolic surface code on the multi-round decoding protocol. Although codes based on tilings of closed hyperbolic surfaces have a comparatively poor asymptotic distance, $d = \Theta(\log n)$, they have a constant encoding rate. These parameters violate the Bravyi-Poulin-Terhal bounds [13], and therefore embedding these codes in 2D Euclidean space is not possible without nonlocal connections. However, they are in some sense close to being local, and so they make a good candidate for the stacked model. For the construction and threshold simulations of these codes, we point the interested reader to Ref. [33]. As the SSF decoder is not known to work for 2D hyperbolic surface codes, we instead use the minimum-weight perfect matching (MWPM) decoder [34]. While we no longer have the guarantees of Theorem 2, the MWPM decoder can be modified to work with masked stabilizer generators. To do this, we set the nodes corresponding to masked generators as boundaries in the matching graph and set the corresponding syndrome bits to zero. Decoding normally, it is then possible to match unpaired syndrome nodes to the boundary. Note that the standard solution to decoding with syndrome noise of building a 3D matching graph with a time dimension does not work since the mask is fixed from round to round, and the repeated measurements provide no additional information.

The code we investigate comes from a family of $\{5, 4\}$-codes with an asymptotic rate of $1/10$ and has parameters $[[360, 38, 8]]$. In Fig. 4 we show the results of running Algorithm 2 with this code and an error rate of $p = 0.003$ for several iterative unmasking schedules. We compare completely unmasked decoding (blue markers) with a schedule that alternates between performing no error correction and 0% masked each round (yellow markers) and one where the masking percentage alternates between 10% and 0% masked each round (red markers). For these codes and decoder, we find that it is actually better to do nothing and let the errors accumulate rather than try to correct the errors with the partial syndrome. We note that we did not observe this behavior for HGP codes, even at the higher error rate. This result is interesting as it seems to imply that the masking behavior for HPG codes is non-trivial.

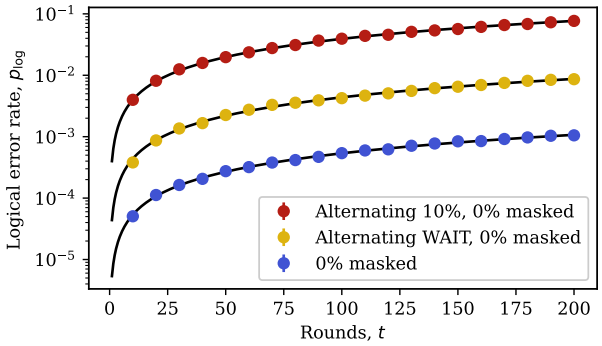

Figure 4: Semilog plot of logical error rate as a function of the number of rounds for a $[[360, 25, 8]]$ 2D hyperbolic surface code and an error rate of $p = 0.003$. We compare fully unmasked decoding performance (blue markers) with two iterative unmasking schedules. Yellow markers denote a schedule consisting of alternating between a round where no error correction is performed and a round where the entire syndrome is available. Red markers denote a schedule where masks of 10% and 0% are used to decode, alternating each round. Fits are of Eq. (9).

One possible explanation for the difference is the single-shot [35, 36] property of the SSF decoder, a property not found in the MWPM decoder. Intuitively, this means that the syndrome has redundancies that make it more resilient to syndrome errors and masks. Over a multi-round decoding procedure, the single-shot property also ensures that the size of any residual error is proportional to the size of the syndrome error. Consequently, misdiagnosing an error cannot have immediate effects throughout the system, since the size of the resulting error is bounded. This is not the case with the MWPM decoder, where a well-placed syndrome error could result in a long error chain across the lattice.

6 Discussion

In this paper, we investigated the feasibility of performing error correction with partial syndromes and found reasonable performance while masking a large constant fraction of the generators. With these results, we have motivated a new practical protocol based on the stacked model for implementing nonlocal qLDPC codes on quantum hardware restricted to 2D local gates. We note that while this limitation has been the main motivation for this work, it is possible that architectures where connectivity is not as much of

a constraint might still benefit from such a protocol. Even for architectures like neutral atoms or trapped ions with effectively all-to-all connectivity, nonlocal gates are still costly in the sense that transport of the qubits is required to perform them. Limiting the number of these operations could provide overhead improvements. There are a number of questions that need to be answered to determine whether this procedure is feasible in general.

- *What families of codes are amenable to the stacked model?* Theoretically, the parameters for HGP codes built from classical expander codes are allowable in this model. In the preparation of this work, some effort was given to find specific embeddings in $\mathbb{Z}^2$ that yielded good generator size distributions; however, the resulting distributions instead often favored mid-sized generators. The consequence of this is that the largest $p_{\text{mask}}\%$ of generators take roughly $p_{\text{mask}}\%$ of the work to measure. To take full advantage of the stacked model, we would instead want those largest generators to take $\gg p_{\text{mask}}\%$ of the work to measure. It is possible that other code families fit better into this model. One option are codes based on tessellations of closed, 4D hyperbolic manifolds [37] which are equipped with an efficient, single-shot decoder [38, 39]. Another option are generalized bicycle codes [40, 41], which might have favorable embeddings.

- *What do the syndrome extraction circuits look like for the stacked model?* The central idea of the stacked model is that the nonlocal generators are being prepared while the local generators are being used for decoding. Careful thought has to be put into the syndrome extraction circuit to ensure that we do not fall into the same pitfall of accumulating too many errors while the nonlocal generators are being prepared. A naïve syndrome extraction circuit consisting of SWAP gates will take $\omega(1)$ time to prepare generators of size $\omega(1)$, which is prohibitively long. Alternatively, one could use the syndrome extraction circuits of Ref. [16]; this method solves the scaling issue by utilizing ancilla qubits to perform long-range CNOT gates in constant depth. Remaining technicalities include the use of entanglement distillation [42, 43] to ensure the resulting long-range CNOT gates are of high enough fidelity.

- *How long does it take to perform a set of masked syndrome measurements?* As discussed in the previous question, performing the syndrome extraction of a single generator can be accomplished in constant time. However, when restricted to $O(n)$ ancilla qubits, the same cannot be said for a growing number of nonlocal generators. Bounds on the depth of 2D local circuits needed to measure the *full* syndrome of a stabilizer code were developed in Ref. [16]. Extending these bounds to include specifying generator size distributions will help inform explicit unmasking schedules, which may provide better performance than the arbitrarily chosen ones studied in this work. These three questions form the basis for a practical implementation of the stacked model and are the focus of future work [17].

Several decoders for HGP codes including belief propagation [31] and ordered statistic decoding [30, 44] have been shown to perform better than the SSF decoder. An interesting question is whether these decoders work as well with the addition of masked generators. Further improvements to the simulation can be gained by using a more realistic fault-tolerance model; in general, the error correction itself can be noisy and result in errors on the qubits and syndrome. Ultimately, performing noisy, circuit-level simulations of the syndrome extraction similar to those done in Ref. [16] will determine whether this protocol is possible as a whole.

Acknowledgements

We thank Antoine Grospellier and Anirudh Krishna, whose code was useful for performing the simulations. D.G. is partially supported by the National Science Foundation (RQS QLCI grant OMA-2120757).

Data Availability

The source code and data to generate the figures in the paper are provided freely

at https://github.com/noahberthusen/hgp_partial_syndrome.

References

[1] D. Aharonov and M. Ben-Or. Fault-tolerant quantum computation with constant error. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, page 176–188. Association for Computing Machinery, 1997. DOI: 10.1145/258533.258579.

[2] A Yu Kitaev. Quantum computations: algorithms and error correction. *Russian Mathematical Surveys*, 52(6):1191, 1997. DOI: 10.1070/RM1997v052n06ABEH002155.

[3] Emanuel Knill, Raymond Laflamme, and Wojciech H. Zurek. Resilient quantum computation: Error models and thresholds. *Proc. R. Soc. Lond. A.*, 454(1969):365–384, 1998. DOI: 10.1098/rspa.1998.0166.

[4] Daniel Gottesman. Fault-tolerant quantum computation with constant overhead. *Quantum Info. Comput.*, 14(15-16):1338–1372, 2014. DOI: 10.26421/QIC14.15-16-5.

[5] Nikolas P. Breuckmann and Jens N. Eberhardt. Balanced product quantum codes. *IEEE Transactions on Information Theory*, 67(10):6653–6674, 2021. DOI: 10.1109/TIT.2021.3097347.

[6] Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, page 375–388, 2022. DOI: 10.1145/3519935.3520017.

[7] A. Leverrier and G. Zemor. Quantum tanner codes. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science*, pages 872–883. IEEE Computer Society, 2022. DOI: 10.1109/FOCS54457.2022.00117.

[8] Irit Dinur, Min-Hsiu Hsieh, Ting-Chun Lin, and Thomas Vidick. Good quantum LDPC codes with linear time decoders. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, page 905–918, 2023. DOI: 10.1145/3564246.3585101.

[9] S. B. Bravyi and A. Yu. Kitaev. Quantum codes on a lattice with boundary. *arXiv preprint arXiv:quant-ph/9811052*, 1998. DOI: 10.48550/arXiv.quant-ph/9811052.

[10] A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003. DOI: 10.1016/s0003-4916(02)00018-0.

[11] Jean-Pierre Tillich and Gilles Zémor. Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, 2014. DOI: 10.1109/TIT.2013.2292061.

[12] Sergey Bravyi and Barbara Terhal. A no-go theorem for a two-dimensional self-correcting quantum memory based on stabilizer codes. *New Journal of Physics*, 11(4):043029, 2009. DOI: 10.1088/1367-2630/11/4/043029.

[13] Sergey Bravyi, David Poulin, and Barbara Terhal. Tradeoffs for reliable quantum information storage in 2D systems. *Phys. Rev. Lett.*, 104:050503, Feb 2010. DOI: 10.1103/PhysRevLett.104.050503.

[14] Nouédyn Baspin and Anirudh Krishna. Quantifying nonlocality: How outperforming local quantum codes is expensive. *Phys. Rev. Lett.*, 129:050505, Jul 2022. DOI: 10.1103/PhysRevLett.129.050505.

[15] Nouédyn Baspin and Anirudh Krishna. Connectivity constrains quantum codes. *Quantum*, 6:711, 2022. DOI: 10.22331/q-2022-05-13-711.

[16] Nicolas Delfosse, Michael E. Beverland, and Maxime A. Tremblay. Bounds on stabilizer measurement circuits and obstructions to local implementations of quantum LDPC codes. *arXiv preprint arXiv:2109.14599*, 2021. DOI: 10.48550/arXiv.2109.14599.

[17] Noah Berthusen, Dhruv Devulapalli, Eddie Schoute, Andrew M. Childs, Michael J. Gullans, Alexey V. Gorshkov, and Daniel Gottesman. Toward a 2D local implementation of quantum LDPC codes. *arXiv preprint arXiv:2404.17676*, 2024. DOI: 10.48550/arXiv.2404.17676.

[18] Nouédyn Baspin, Omar Fawzi, and Ala Shayeghi. A lower bound on the overhead of quantum error correction in low dimensions. *arXiv preprint arXiv:2302.04317*, 2023. DOI: 10.48550/arXiv.2302.04317.

[19] Daniel Gottesman. Stabilizer codes and quantum error correction. *arXiv preprint*

arXiv:quant-ph/9705052, 1997. DOI: 10.48550/arXiv.quant-ph/9705052.

[20] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane. Quantum error correction and orthogonal geometry. *Phys. Rev. Lett.*, 78:405–408, 1997. DOI: 10.1103/PhysRevLett.78.405.

[21] A. R. Calderbank and Peter W. Shor. Good quantum error-correcting codes exist. *Phys. Rev. A*, 54:1098–1105, 1996. DOI: 10.1103/PhysRevA.54.1098.

[22] M. Sipser and D.A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996. DOI: 10.1109/18.556667.

[23] Anthony Leverrier, Jean-Pierre Tillich, and Gilles Zémor. Quantum expander codes. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 810–824, 2015. DOI: 10.1109/FOCS.2015.55.

[24] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Efficient decoding of random errors for quantum expander codes. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, page 521–534, 2018. DOI: 10.1145/3188745.3188886.

[25] Daniel Gottesman. Opportunities and challenges in fault-tolerant quantum computation. *arXiv preprint arXiv:2210.15844*, 2022. DOI: 10.48550/arXiv.2210.15844.

[26] Omar Fawzi, Antoine Grospellier, and Anthony Leverrier. Constant overhead quantum fault tolerance with quantum expander codes. *Commun. ACM*, 64(1):106–114, 2020. DOI: 10.1145/3434163.

[27] Antoine Grospellier. Constant time decoding of quantum expander codes and application to fault-tolerant quantum computation. *PhD Thesis*, 2019. Sorbonne Université.

[28] Alexey A. Kovalev, Sanjay Prabhakar, Ilya Dumer, and Leonid P. Pryadko. Numerical and analytical bounds on threshold error rates for hypergraph-product codes. *Phys. Rev. A*, 97:062320, 2018. DOI: 10.1103/PhysRevA.97.062320.

[29] Antoine Grospellier and Anirudh Krishna. Numerical study of hypergraph product codes. *arXiv preprint arXiv:1810.03681*, 2019. DOI: 10.48550/arXiv.1810.03681.

[30] Joschka Roffe, David R. White, Simon Burton, and Earl Campbell. Decoding across the quantum low-density parity-check code landscape. *Phys. Rev. Res.*, 2:043423, Dec 2020. DOI: 10.1103/PhysRevResearch.2.043423.

[31] Antoine Grospellier, Lucien Grouès, Anirudh Krishna, and Anthony Leverrier. Combining hard and soft decoders for hypergraph product codes. *Quantum*, 5:432, 2021. DOI: 10.22331/q-2021-04-15-432.

[32] Zijun Chen et al. Exponential suppression of bit or phase errors with cyclic error correction. *Nature*, 595(78677867):383–387, 2021. DOI: 10.1038/s41586-021-03588-y.

[33] Nikolas P. Breuckmann and Barbara M. Terhal. Constructions and noise threshold of hyperbolic surface codes. *IEEE Transactions on Information Theory*, 62(6):3731–3744, 2016. DOI: 10.1109/TIT.2016.2555700.

[34] Oscar Higgott and Craig Gidney. Sparse blossom: correcting a million errors per core second with minimum-weight matching. *arXiv preprint arXiv:2303.15933*, 2023. DOI: 10.48550/arXiv.2303.15933.

[35] Héctor Bombín. Single-shot fault-tolerant quantum error correction. *Phys. Rev. X*, 5:031043, 2015. DOI: 10.1103/PhysRevX.5.031043.

[36] Earl T Campbell. A theory of single-shot error correction for adversarial noise. *Quantum Science and Technology*, 4(2):025006, 2019. DOI: 10.1088/2058-9565/aafc8f.

[37] Larry Guth and Alexander Lubotzky. Quantum error correcting codes and 4-dimensional arithmetic hyperbolic manifolds. *J. Math. Phys.*, 55(8), 2014. ISSN 0022-2488. DOI: 10.1063/1.4891487.

[38] Matthew B. Hastings. Decoding in hyperbolic spaces: quantum LDPC codes with linear rate and efficient error correction. *Quantum Info. Comput.*, 14(13–14):1187–1202, 2014. DOI: 10.48550/arXiv.1312.2546.

[39] Nikolas P. Breuckmann and Vivien Londe. Single-shot decoding of linear rate LDPC quantum codes with high performance. *IEEE Trans. Inf. Theory*, 68(1):272–286, 2022. DOI: 10.1109/TIT.2021.3122352.

[40] Alexey A. Kovalev and Leonid P. Pryadko. Quantum kronecker sum-product low-density parity-check codes with finite rate.

Phys. Rev. A, 88:012311, 2013. DOI: 10.1103/PhysRevA.88.012311.

[41] Sergey Bravyi, Andrew W. Cross, Jay M. Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J. Yoder. High-threshold and low-overhead fault-tolerant quantum memory. *Nature*, 627:778–782, 2024. DOI: 10.1038/s41586-024-07107-7.

[42] Charles H. Bennett, Gilles Brassard, Sandu Popescu, Benjamin Schumacher, John A. Smolin, and William K. Wootters. Purification of noisy entanglement and faithful teleportation via noisy channels. *Phys. Rev. Lett.*, 76:722–725, 1996. DOI: 10.1103/PhysRevLett.76.722.

[43] Charles H. Bennett, David P. DiVincenzo, John A. Smolin, and William K. Wootters. Mixed-state entanglement and quantum error correction. *Phys. Rev. A*, 54:3824–3851, 1996. DOI: 10.1103/PhysRevA.54.3824.

[44] Pavel Panteleev and Gleb Kalachev. Degenerate quantum LDPC codes with good finite length performance. *Quantum*, 5:585, 2021. DOI: 10.22331/q-2021-11-22-585.