Foliated Quantum Error-Correcting Codes

A. Bolt,[1] G. Duclos-Cianci,[2] D. Poulin,[2] and T. M. Stace[1,*]
[1]*ARC Centre for Engineered Quantum System, Department of Physics, University of Queensland, Brisbane, Queensland 4072, Australia*
[2]*Département de Physique, Université de Sherbrooke, Québec J1K 2R1, Canada*

(Received 10 May 2016; published 10 August 2016)

We show how to construct a large class of quantum error-correcting codes, known as Calderbank-Steane-Shor codes, from highly entangled cluster states. This becomes a primitive in a protocol that *foliates* a series of such cluster states into a much larger cluster state, implementing foliated quantum error correction. We exemplify this construction with several familiar quantum error-correction codes and propose a generic method for decoding foliated codes. We numerically evaluate the error-correction performance of a family of finite-rate Calderbank-Steane-Shor codes known as turbo codes, finding that they perform well over moderate depth foliations. Foliated codes have applications for quantum repeaters and fault-tolerant measurement-based quantum computation.

DOI: 10.1103/PhysRevLett.117.070501

Quantum error correction is critical to building practical quantum-information processors (QIPs). In an influential series of papers, Raussendorf and co-workers described a measurement-based approach to fault-tolerant quantum processing using highly entangled *cluster states*, defined on a 3D lattice [1–4]. Raussendorf's 3D cluster state can be visualized as a *foliation* of Kitaev's surface code [5,6], i.e., a sequence of 2D surface-code "sheets," stacked together to form a 3D lattice. This is evident in Ref. [1], where it is shown that measuring the "bulk" qubits of a 3D cluster state leaves the two logical surface-code qubits encoded in the boundary faces in an entangled Bell pair.

Raussendorf's 3D cluster gained prominence for its high fault-tolerant computational error thresholds $\lesssim 1\%$. It has applications in various QIP tasks, including long-range entanglement sharing, in which surface-code cluster states are created at regularly spaced local nodes, which are linked by medium-range optical channels into a 3D cluster state [7]. It is capable of fault-tolerant, measurement-based quantum computation, using an elegant geometric construction that braids defects in the interior of the 3D cluster state to produce robust Clifford gates. Universality is afforded by magic state injection and distillation [3,4,8].

The robustness of Refs. [3,4] is inherited from the underlying surface code, which has a high error-correction threshold $\sim 11\%$ [6,9–11]. The surface code has large distance and zero rate (in regards to the asymptotic ratio of the number of logical and physical qubits), reflecting the trade-off between distance and rate in two spatial dimensions [12]. It is natural to ask how to adapt the foliated structure of Refs. [1,2] to use other underlying codes that could achieve a higher encoding rate.

Another motivation for our work is recent fault-tolerant schemes that produce a universal gate set by code deformation and code switching [13–15]. Extending code foliation to codes that circumvent magic state distillation [8] may produce cluster states with a lower resource overhead for fault-tolerant measurement-based QIPs.

In this Letter we show that all Calderbank-Steane-Shor (CSS) codes can be *clusterized*, meaning that they can be derived, using single-qubit measurements, from a larger cluster state [16,17] defined over the code qubits plus additional ancilla qubits. We use this fact to develop our main result: generalizing Raussendorf's 3D lattice to a foliation of any clusterized CSS code. This is a larger cluster state comprising alternating copies of a clusterized CSS code and its dual. We demonstrate this construction for some familiar CSS codes and present a general decoding algorithm for foliated codes, utilizing the underlying code's decoder. Finally, we apply the construction to a family of finite-rate CSS codes called turbo codes [18,19], and present Monte Carlo simulations of the error-correction performance of foliated turbo codes.

Background.—CSS code stabilizer generators are classified into two sets: $S_Z \in \{I,Z\}^{\otimes n}$ and $S_X \in \{I,X\}^{\otimes n}$, where Z and X denote the Pauli matrices [20]. An $[[n,k,d]]$ CSS code satisfies $k = n - (|\mathcal{S}_X| + |\mathcal{S}_Z|)$. We write a stabilizer in $\mathcal{S}_Z$ as $Z_{\vec{b}} \equiv \otimes_{\vec{b}} Z^{b_j}$ for some binary vector $\vec{b} = (b_1, b_2, \dots, b_n)$ with $b_j = 1$ if qubit j is in the stabilizer $Z_{\vec{b}}$, and $b_j = 0$ otherwise, i.e., $\vec{b}$ is a row of the code's parity-check matrix, B_Z. Similarly, a stabilizer in $\mathcal{S}_X$ is given by $X_{\vec{c}}$ for some binary list $\vec{c}$. The associated dual code is derived from the primal code by exchanging X and Z operators in the stabilizer generators, $X \leftrightarrow Z$.

A cluster state is defined on a collection of qubits located at the vertices of a graph [16,17,21]. A qubit at vertex v is associated with a cluster stabilizer $C_v = X_v(\otimes_{\mathcal{N}_v} Z) \equiv X_v Z_{\mathcal{N}_v}$, acting on it and its neighbors, $\mathcal{N}_v$. The cluster state is the $+1$ eigenstate of the C_v's.

Clusterized CSS codes.—An $[[n,k,d]]$ CSS code can be generated from a larger *progenitor* cluster state, i.e., clusterized. The progenitor cluster is simply the cluster state associated with the Tanner graph of $\mathcal{S}_Z$ [22], i.e., a bipartite graph $G=(V,E)$ whose vertices V are labeled by code qubits j, or ancilla qubits a, each associated with a stabilizer $Z_{\mathcal{N}_a}\in\mathcal{S}_Z$, so that $|V|=n+|\mathcal{S}_Z|$. E contains the graph edge (a,j) if $[B_Z]_{a,j}=1$. We now show that a code state of the CSS code is obtained by measuring the ancilla qubits of the progenitor cluster in the X basis.

In the above definition, the cluster stabilizer associated with ancilla a is $C_a=X_aZ_{\mathcal{N}_a}$. Measurement of a in the X basis with outcome ±1 projects adjacent code qubits onto an eigenstate of the code stabilizer $Z_{\mathcal{N}_a}\in\mathcal{S}_Z$. Thus, $\mathcal{S}_Z$ is generated by ancilla measurements.

Because $\mathcal{S}_X$ and $\mathcal{S}_Z$ mutually commute, the progenitor cluster is also an eigenstate of the generators in $\mathcal{S}_X$. To see this, take an element $X_{\vec{c}}\in\mathcal{S}_X$ and consider the product of cluster stabilizers centered at each code qubit $c_j\in\vec{c}$, given by $C_{\vec{c}}\equiv\otimes_{\vec{c}}\,C_{c_j}=\otimes_{\vec{c}}\,(X_{c_j}Z_{\mathcal{N}_{c_j}})$. The neighborhood, $\mathcal{N}_{c_j}$, of the code qubit c_j consists only of ancilla. The code stabilizer $X_{\vec{c}}=\otimes_{\vec{c}}\,X_{c_j}$ has an even overlap with any Z-like stabilizer, $Z_{\vec{b}_a}$ (which is generated by measurement of ancilla a in the X basis). It follows that the intersection of $\vec{c}$ and $\vec{b}_a$ has an even number of qubits. Any ancilla qubit a thus appears in the product $\otimes_{\vec{c}}\,C_{c_j}$ an even number of times, so $\otimes_{a\in\mathcal{N}_{c_j}}Z_a=\otimes_{a\in\mathcal{N}_{c_j}}\mathbb{I}_a$, and $C_{\vec{c}}=X_{\vec{c}}$. Thus, $\mathcal{S}_X$ is generated by cluster stabilizers.

The same argument implies that logical X operators of the CSS code (which are products of local X operators that commute with $\mathcal{S}_Z$) are also generated by cluster stabilizers. It follows that ancilla measurements project the cluster state onto a logical X code state.

The surface code [5] exemplifies the relationship between a CSS code and a progenitor cluster state. Starting from the cluster state defined on the lattice shown in Fig. 1(c), and measuring the ancilla qubits (the red squares) in the X basis results in a new state on the remaining code qubits (the blue circles) which is stabilized by the surface-code plaquette operators, e.g., $Z_2Z_4Z_5Z_7\in\mathcal{S}_Z^{\text{surf}}$, and vertex operators, e.g., $X_4X_6X_7X_9\in\mathcal{S}_X^{\text{surf}}$. It is, therefore, a code state of the surface code [23].

Other examples of clusterized CSS codes are shown in Fig. 1(a) for Steane's seven-qubit code [24], for which $\mathcal{S}_Z^{\text{Steane}}=\{Z_1Z_2Z_6Z_7,Z_2Z_3Z_4Z_7,Z_4Z_5Z_6Z_7\}$ (which is also a minimal example of the color code [25,26]), and Fig. 1(b) for Shor's nine-qubit code [27,28], for which $\mathcal{S}_Z^{\text{Shor}}=\{Z_1Z_2,Z_2Z_3,Z_4Z_5,Z_5Z_6,Z_7Z_8,Z_8Z_9\}$.

The examples in Fig. 1 illustrate the fact that X measurements of the ancilla qubits project out code stabilizers in $\mathcal{S}_Z$, while each stabilizer in $\mathcal{S}_X$ comes "for free" simply by considering products of C_j's acting on the corresponding code qubits, and noting that these products act trivially on the ancilla qubits. For example, in the Steane code cluster of Fig. 1(a), it is straightforward to check that $C_1C_2C_6C_7=X_1X_2X_6X_7\in\mathcal{S}_X^{\text{Steane}}$.

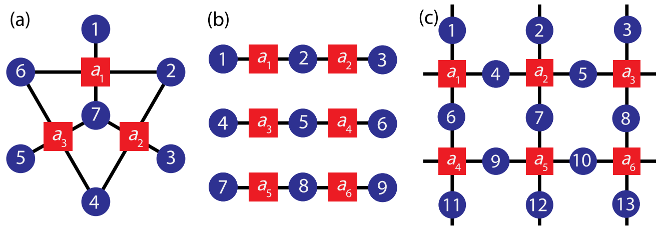

FIG. 1. Examples of progenitor clusters for clusterized CSS codes. (a) Clusterized Steane code. (b) Clusterized Shor code. (c) Clusterized surface code. Code qubits (the blue circles) are connected by cluster bonds (the black lines) to ancilla qubits (the red squares). An X-basis measurement of ancilla a_k projects neighboring code qubits onto an eigenstate of $\otimes_{\mathcal{N}_{a_k}}Z\in\mathcal{S}_Z$.

Foliated codes.—Raussendorf's 3D cluster state construction [1–4], Fig. 2(c), can be viewed as a foliation of the surface-code cluster state shown in Fig. 1(c). Alternating sheets of the primal surface-code cluster state and its dual are stacked together [29], with additional cluster bonds (the green lines) extending between code qubits in each sheet and the corresponding code qubits in the adjacent dual sheets.

We now generalize this construction to arbitrary CSS codes. Take an alternating stack of sheets of clusterized primal and dual codes, and link the sheets together by creating additional cluster bonds between primal code qubits in a given sheet, m, and the corresponding dual code qubits in the adjacent sheets, $m\pm1$. We call this a *foliated code*. The number of *layers*, L, in the foliated construction counts the number of primal-dual sheet pairs, so that $1\le m\le 2L+1$.

Figure 2(a) shows the example of a foliated Steane code (which is self-dual, so the primal and dual sheets are identical). Figure 2(b) shows the foliated Shor code, for which primal and dual clusters are different. One can readily verify that this definition preserves the key feature of Rausendorf's construction: measuring the bulk qubits and the boundary ancilla qubits leaves the two boundary sheets in an encoded Bell state [1]. Because this feature enables fault-tolerant measurement-based quantum computation and long-range entanglement sharing, our generalization has immediate applications in these settings, offering additional flexibility in the choice of code.

Errors.—Errors may arise during construction of the cluster or storage of the qubits, or during single-qubit measurement. In the error models that we consider, preparation and measurement errors can be mapped onto possibly correlated storage errors [9]. Furthermore, after the cluster is created, since we perform single-qubit X measurements, X errors do not affect the measurement outcome; only Z errors on the final foliated cluster act nontrivially. We note that X errors during cluster construction are equivalent to correlated Z errors in the final cluster [9,10]. This asymmetry between the X and Z errors is a consequence of the asymmetry in the definition of the

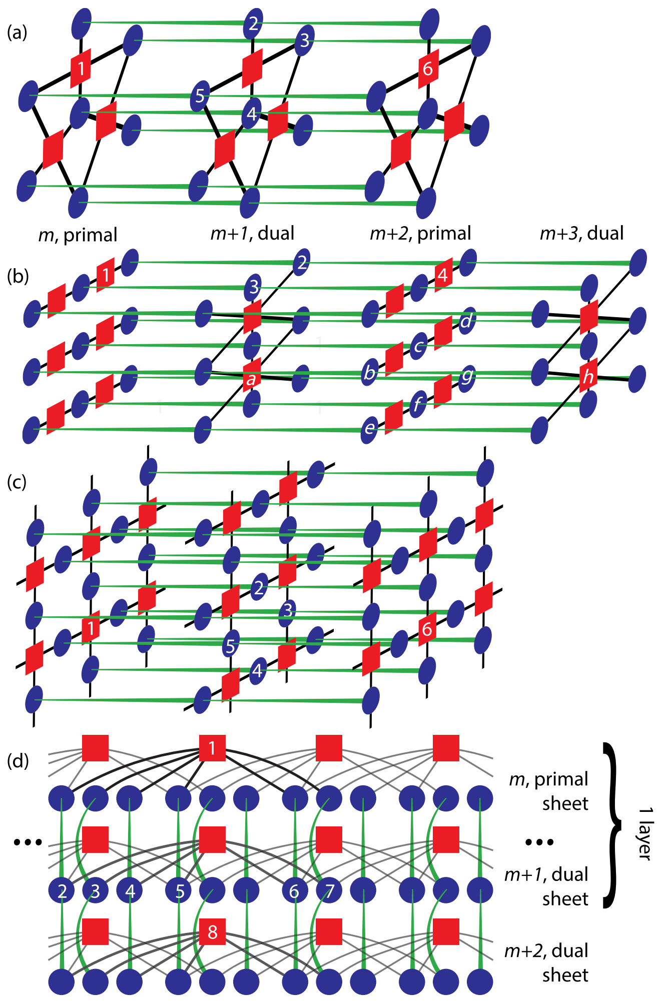

FIG. 2. Examples of foliated, clusterized CSS codes. Code qubits (the blue circles) share cluster bonds (the black lines) with ancilla qubits (the red squares) in the same sheet, and with code qubits in adjacent sheets (the green lines). (a) Foliated Steane code. Being self-dual, primal and dual sheets are identical. The product of cluster stabilizers centered on the numbered qubits generates parity-check operators $C_1C_2\cdots C_6 = X_1X_2\cdots X_6$. (b) Foliated Shor code. This code is not self-dual, so primal and dual sheets are different, and there are two kinds of parity-check operators: $C_aC_b\cdots C_h = X_aX_b\cdots X_h$, centered on primal sheets, and $C_1C_2C_3C_4 = X_1X_2X_3X_4$, centered on dual sheets. (c) Foliated surface code [1], with parity-check operators $C_1\cdots C_6 = X_1\cdots X_6$. (d) Foliated self-dual convolutional code with parity-check operator $C_1\cdots C_8 = X_1\cdots X_8$. Stabilizers are generated by translations of the kernel (indicated by thick edges) across frames (here, the frame length is 3).

cluster stabilizers. Correlated or asymmetric errors may also arise in specific applications, such as long-range repeaters where internode quantum transmission errors are much worse than those within a node, which can be mitigated by a suitable choice of code [30].

Parity-check operators.—Errors in the foliated cluster are detected by parity-check operators: a Z error will flip one or more parity checks, giving a nontrivial error syndrome for the foliated cluster. Importantly, the parity-check measurement outcomes can be inferred from sets of single-qubit X measurements.

Each parity-check operator is associated with a CSS code stabilizer within a code sheet. To construct a parity-check operator, consider the CSS code stabilizer $X_{\vec{c},m} \in \mathcal{S}_X$, in sheet m of a foliated cluster state. The product of foliated cluster stabilizers centered on each of the code qubits indicated by $\vec{c}$ is $C_{\vec{c},m} = X_{\vec{c},m}Z_{\mathcal{N}_{\vec{c},m}} = Z_{\vec{c},m-1}X_{\vec{c},m}Z_{\vec{c},m+1}$. The dual code sheets, $m \pm 1$, each have a cluster stabilizer $C_{a_{\vec{c}},m\pm1} = X_{a_{\vec{c}},m\pm1}Z_{\vec{c},m\pm1}$ centered on an ancilla qubit $a_{\vec{c}}$ associated with $\vec{c}$. Thus, $\hat{P}_{\vec{c},m} \equiv C_{a_{\vec{c}},m-1}C_{\vec{c},m}C_{a_{\vec{c}},m+1} = X_{a_{\vec{c}},m-1}X_{\vec{c},m}X_{a_{\vec{c}},m+1}$ defines a parity check for the foliated cluster, centered on code stabilizer $X_{\vec{c},m}$. Note that parity-check operators centered on primal sheets share no common qubits with those centered on dual sheets.

This generalizes the construction of the parity-check operators for Raussendorf's 3D cubic lattice, which are formed by products of X operators on the faces of the cubic unit cells, as shown in Fig. 2(c) (exemplified by numbered qubits). Parity-check operators for other foliated CSS codes are exemplified by labeled qubits in other panels of Fig. 2. In a non-self-dual code, such as the Shor code, primal and dual parity-check operators may have different weights [Fig. 2(b)].

Logical code operators within a sheet commute with the parity-check operators. It follows that for an underlying $[[n,k,d]]$ code, there are weight-d undetected error chains on the foliated cluster, as in Refs. [1–4,6]. Since the structure of the code in the direction of foliation is a simple repetition, it follows that the foliated cluster inherits the distance of the underlying code.

Decoding.—A nontrivial error syndrome indicates the presence of Z errors. If the error probability is sufficiently small, the most likely class of errors can be inferred from the syndrome with high probability, facilitating error recovery. Small codes can be decoded by brute force, but this is not computationally scalable in n.

There are a number of computationally efficient, near-optimal decoders available for both the 2D surface code and its 3D foliation, including hard decoders (which return a specific high-likelihood error pattern) based on perfect matching [6,9], and soft decoders (which return a probability distribution over error patterns) based on renormalization methods [11,31].

Surface-code decoders naturally generalize to the 3D Raussendorf lattice, as exemplified by matching-based decoders. While generic CSS codes cannot typically be efficiently decoded, many exact or heuristic decoders are known for specific code constructions [6,11,19,32]. The problem we address here is to use a soft decoder for the underlying CSS code—which we presume to be efficient—as a subroutine in a decoder for the foliated construction. We describe a heuristic method based on belief propagation (BP) that may work in many cases [33,34]. We assume the existence of soft decoders for the underlying CSS primal and dual codes, which, given a physical error model, calculates the probability of a Pauli error σ on code qubit j, $P(\sigma_j|S_{\rm CSS})$, conditioned on a syndrome, $S_{\rm CSS}$, which may itself be unreliable.

In the foliated case, consider a parity-check operator $\hat{P}_{\vec{c},m} = X_{a_{\vec{c}},m-1}X_{\vec{c},m}X_{a_{\vec{c}},m+1}$ centered on primal sheet m. A nontrivial syndrome can arise because of errors on code qubits $\vec{c}$ within code sheet m, or due to errors on the corresponding ancilla qubits, $a_{\vec{c}}$ in adjacent dual sheets $m \pm 1$. If the dual-sheet ancilla qubits were error free, then all the parity-check failures would be due solely to in-code qubit errors, so that the parity-check outcomes centered on sheet m would be in direct correspondence with the CSS code syndrome for that sheet. The code syndrome could then be used in the CSS decoder to calculate a soft error model on sheet m, from which an error-correction strategy could be determined.

However, errors on the dual-sheet ancilla qubits mean that the in-sheet syndrome passed to the CSS decoder is itself unreliable. To account for dual-sheet ancilla errors, we embed the CSS decoder in a BP routine as follows.

Step 1.—For each code qubit, j, in sheet m, the CSS decoder calculates an in-sheet error model probability distribution, $P_m(\sigma_j|S_m \cup P_{m\pm1}(a_k))$, subject to both the measured code syndrome, S_m, which is derived from the foliated parity-check operators centered on sheet m, and an assumed error model, $P_{m\pm1}(a_k)$, for errors on ancilla qubits, a_k, in adjacent dual sheets.

Step 2.—Using the result of step 1 we fix the code qubit error model, $P_m(\sigma_j)$, and calculate an error model on the dual-sheet ancilla qubits, $P_{m\pm1}(a_k|S_{m\pm1} \cup P_m(\sigma_j))$.

Step 3.—We iterate step 1, using the result of step 2 for $P_{m\pm1}(a_k)$, repeating until each error model converges.

Turbo codes.—We now consider the class of turbo codes, which are finite-rate CSS codes with bounded-weight stabilizers [19,33,35]. These are capable of encoding an arbitrary number of logical qubits with a finite rate, $r = k/n$. Essentially, turbo codes are formed from a concatenation of two *convolutional* codes: an inner code G_I and an outer code G_O [32,36–38], each of which can be decoded with soft trellis decoders [19,33,35,38,39].

Convolutional codes are defined over an ordered set of qubits. The code stabilizers are generated by a kernel which is repeatedly translated across *frames* (i.e., blocks of the underlying physical qubits). For illustrative purposes, Fig. 2(d) shows a foliation of three sheets of a $d = 3$, $r = 1/3$, weight-6 self-dual CSS convolutional code cluster. The code stabilizer kernel is indicated by the dark cluster edges within a sheet. Turbo codes are conceptually similar, albeit with more complicated Tanner graphs.

Turbo codes provide a platform for testing the foliated construction on codes that are quite different from the surface code. Since they are a code family, we analyze the performance of the codes as a function of the code size $k = nr$ and the number of foliated layers, L. A soft trellis decoder [38,40] for the underlying code is embedded as a subroutine in a BP decoder spanning the sheets of the foliation. The BP decoder run time is linear in L; however, the trellis decoder complexity is exponential in the size of the turbo code frame length, making simulations practically slow.

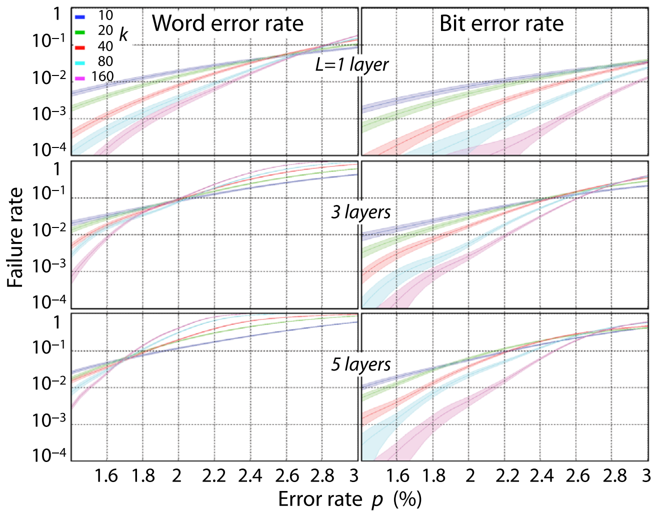

FIG. 3. Numerical performance results for a foliated $r = \frac{1}{25}$, $d = 25$ turbo code, for different numbers of foliated layers, L (rows). Different colors correspond to different code sizes, $k = nr$; shading indicates $\pm 1\sigma$. A layer consists of a primal code sheet and a dual code sheet [see Fig. 2(d)]. Word error rate (the left column) counts any errors across all k logical qubits. Bit error rate (right column) counts the failure rate per logical qubit.

Figure 3 shows the performance of a $d = 25$, $r = 1/25$, self-dual foliated turbo code, based on Monte Carlo simulations of errors. As noted earlier, X errors on the foliated cluster commute with parity-check measurements. Thus, for our simulations we assume a phenomenological error model in which uncorrelated Z errors are distributed independently across the cluster with rate p. The decoder performance is quantified in terms of both the word error rate (WER), which is the probability of one or more errors across all k encoded qubits, and the bit error rate (BER), which is the probability of an error in each of the encoded qubits.

For each L, there is a threshold error rate around $p \sim 2\%$, below which the code performance improves with code length (up to at least 160 encoded logical qubits per code sheet), consistent with pseudothreshold behavior seen in turbo codes [19]. As L increases, the threshold decreases, more pronouncedly for the WER than the BER. The range of k and L that we can simulate is limited by computational time, so we cannot explore the asymptotic performance for large L's. Nevertheless, numerics indicate that foliated turbo codes perform quite well for moderate depth foliations.

We note that the foliated construction transforms a clusterized code into a fault-tolerant resource state, but with a consequent reduction in threshold. This is seen in Fig. 3 and in Raussendorf's construction, in which the fault-tolerant threshold $\lesssim 1\%$ is smaller than the $\sim 11\%$ threshold for the surface code on which it is based.

The ~1% threshold observed for foliated surface codes is obtained by scaling the code distance d and the foliation depth L together. Here, the code distance is fixed at $d = 25$, which is responsible for the observed decreasing value of the pseudothreshold with an increasing L.

Our main motivation for studying turbo codes is to demonstrate the foliated construction and BP decoder in an extensible, finite-rate code family. Practically, these and other finite-rate codes may have applications in fault-tolerant quantum repeater networks [7,41], where local nodes create optimal clusterized codes to reduce resource overheads or error tolerance [42].

In conclusion, we have shown how to clusterize arbitrary CSS codes. We have shown how to foliate clusterized codes, generalizing Raussendorf's 3D foliation of the surface code. We have described a generic approach to decoding errors that arise on the foliated cluster using an underlying soft decoder for the CSS code as a subroutine in a BP decoder, and we have applied it to error correction by means of a foliated turbo code. This construction may have applications where codes with a finite rate are useful, such as long-range quantum repeater networks.

This work was funded by the ARC Centre of Excellence for Engineered Quantum Systems, NSERC, and the Canadian Institute for Advanced Research. We thank Sean Barrett, Andrew Doherty, Terry Rudolph, Clemens Mueller, and Stephen Bartlett for the helpful discussions.

[*]stace@physics.uq.edu.au

[1] R. Raussendorf, S. Bravyi, and J. Harrington, Long-range quantum entanglement in noisy cluster states, Phys. Rev. A **71**, 062313 (2005).
[2] R. Raussendorf, J. Harrington, and K. Goyal, A fault-tolerant one-way quantum computer, Ann. Phys. (Amsterdam) **321**, 2242 (2006).
[3] R. Raussendorf and J. Harrington, Fault-Tolerant Quantum Computation with High Threshold in Two Dimensions, Phys. Rev. Lett. **98**, 190504 (2007).
[4] R. Raussendorf, J. Harrington, and K. Goyal, Topological fault-tolerance in cluster state quantum computation, New J. Phys. **9**, 199 (2007).
[5] A. Kitaev, Fault-tolerant quantum computation by anyons, Ann. Phys. (Amsterdam) **303**, 2 (2003).
[6] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, J. Math. Phys. (N.Y.) **43**, 4452 (2002).
[7] Y. Li, S. D. Barrett, T. M. Stace, and S. C. Benjamin, Fault Tolerant Quantum Computation with Nondeterministic Gates, Phys. Rev. Lett. **105**, 250502 (2010).
[8] S. Bravyi and A. Kitaev, Univeral quantum computation with ideal clifford gates and noisy ancillas, Phys. Rev. A **71**, 022316 (2005).
[9] T. M. Stace and S. D. Barrett, Error correction and degeneracy in surface codes suffering loss, Phys. Rev. A **81**, 022317 (2010).
[10] S. D. Barrett and T. M. Stace, Fault Tolerant Quantum Computation with Very High Threshold for Loss Errors, Phys. Rev. Lett. **105**, 200502 (2010).
[11] G. Duclos-Cianci and D. Poulin, Fast Decoders for Topological Quantum Codes, Phys. Rev. Lett. **104**, 050504 (2010).
[12] S. Bravyi, D. Poulin, and B. Terhal, Tradeoffs for Reliable Quantum Information Storage in 2D Systems, Phys. Rev. Lett. **104**, 050503 (2010).
[13] H. Bombin, Dimensional jump in quantum error correction, New J. Phys. **18**, 043038 (2016).
[14] A. Paetznick and B. W. Reichardt, Universal Fault-Tolerant Quantum Computation with Only Transversal Gates and Error Correction, Phys. Rev. Lett. **111**, 090505 (2013).
[15] J. T. Anderson, G. Duclos-Cianci, and D. Poulin, Fault-Tolerant Conversion between the Steane and Reed-Muller Quantum Codes, Phys. Rev. Lett. **113**, 080501 (2014).
[16] R. Raussendorf and H. J. Briegel, A One-Way Quantum Computer, Phys. Rev. Lett. **86**, 5188 (2001).
[17] R. Raussendorf, D. E. Browne, and H. J. Briegel, Measurement-based quantum computation on cluster states, Phys. Rev. A **68**, 022312 (2003).
[18] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, TDA Progress Report No. 42-124, 1996.
[19] D. Poulin, J.-P. Tillich, and H. Ollivier, Quantum serial turbo codes, IEEE Trans. Inf. Theory **55**, 2776 (2009).
[20] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, England, 2000).
[21] H. Briegel, D. Browne, W. Dür, R. Raussendorf, and M. Van den Nest, Measurement-based quantum computation, Nat. Phys. **5**, 19 (2009).
[22] R. Tanner, A recursive approach to low complexity codes, IEEE Trans. Inf. Theory **27**, 533 (1981).
[23] We allow ±1 eigenstates of $Z_{\bar{b}_a} \in \mathcal{S}_Z$ in this definition; a trivial syndrome corresponds to the agreement between $Z_{\bar{b}_a}$ and the associated ancilla X_a measurement outcome.
[24] A. M. Steane, Error Correcting Codes in Quantum Theory, Phys. Rev. Lett. **77**, 793 (1996).
[25] G. Duclos-Cianci, H. Bombin, and D. Poulin, Local equivalence of topological order: Kitaev's code and color codes, American Physical Society, APS March Meeting, 2011.
[26] M. Kargarian, H. Bombin, and M. Martin-Delgado, Topological color codes and two-body quantum lattice hamiltonians, New J. Phys. **12**, 025018 (2010).
[27] A. R. Calderbank and P. W. Shor, Good quantum error-correcting codes exist, Phys. Rev. A **54**, 1098 (1996).
[28] P. Shor, Scheme for reducing decoherence in quantum computer memory, Phys. Rev. A **52**, R2493 (1995).
[29] The dual to the surface code is also a surface code, albeit on the lattice geometrically rotated by 90°.
[30] C. Cafaro and S. Mancini, Quantum stabilizer codes for correlated and asymmetric depolarizing errors, Phys. Rev. A **82**, 012306 (2010).
[31] G. Duclos-Cianci and D. Poulin, Fault-tolerant renormalization group decoder for abelian topological codes, Quantum Inf. Comput. **14**, 721 (2014).
[32] H. Ollivier and J.-P. Tillich, Description of a Quantum Convolutional Code, Phys. Rev. Lett. **91**, 177902 (2003).
[33] R. McEliece, D. MacKay, and J. Cheng, Turbo decoding as an instance of pearl's "belief propagation" algorithm, IEEE J. Sel. Areas Commun. **16**, 140 (1998).
[34] D. Poulin and Y. Chung, On the iterative decoding of sparse quantum codes, Quantum Inf. Comput. **8**, 987 (2008).

[35] S. Papaharalabos, P. Sweeney, and B. G. Evans, SISO algorithms based on Max-Log-MAP and Log-MAP turbo decoding, IET Commun. **1**, 49 (2007).

[36] H. Ollivier and J.-P. Tillich, Description of a Quantum Convolutional Code, Phys. Rev. Lett. **91**, 177902 (2003).

[37] J.-P. Tillich and G. Zemor, Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength, IEEE Trans. Inf. Theory **60**, 1193(2014).

[38] P. Tan and J. Li, Efficient ML decoding for quantum convolutional codes, arXiv:1004.0174.

[39] J. Geldmacher and J. Gotze, On fault tolerant decoding of turbo codes, in *Proceedings of the 7th International Symposium on Turbo Codes and Iterative Information Processing (ISTC), Gothenburg, Sweden, 2012* (IEEE, New York, 2012).

[40] J. Tu, Zhenyu. Li, and R. Blum, An efficient SF-ISF approach for the slepian-wolf source coding problem, EURASIP J. Appl. Signal Process. **6**, 961 (2005).

[41] L.-M. Duan, M. D. Lukin, J. I. Cirac, and P. Zoller, Long-distance quantum communication with atomic ensembles and linear optics, Nature (London) **414**, 413 (2001).

[42] T. Satoh, K. Ishizaki, S. Nagayama, and R. Van Meter, Analysis of quantum network coding for realistic repeater networks, Phys. Rev. A **93**, 032302 (2016).