Quantum serial turbo-codes

David Poulin*, Jean-Pierre Tillich†, and Harold Ollivier‡
* Center for the Physics of Information, California Institute of Technology, Pasadena, CA 91125, USA.
† INRIA, Projet Codes, Domaine de Voluceau BP 105, F-78153 Le Chesnay cedex, France.
‡ Perimeter Institute for Theoretical Physics, Waterloo, ON, N2J 2W9, Canada.

Abstract— **We present a theory of quantum serial turbo-codes, describe their decoding algorithm, and study their performances numerically on a depolarization channel. These codes can be considered as a generalization of classical serial turbo-codes. As their classical cousins, they can be iteratively decoded and with well chosen constituent convolutional codes, we observe an important reduction of the word error rate as the number of encoded qubits increases.**

Our construction offers several advantages over quantum LDPC codes. First, the Tanner graph used for decoding can be chosen to be free of 4-cycles that deteriorate the performances of iterative decoding. Secondly, the iterative decoder makes explicit use of the code's degeneracy. Finally, there is complete freedom in the code design in terms of length, rate, memory size, and interleaver choice.

We address two issues related to the encoding of convolutional codes that are directly relevant for turbo-codes, namely the character of being recursive and non-catastrophic. We define a quantum analogue of a state diagram that provides an efficient way to verify these properties on a given quantum convolutional encoder. Unfortunately, we also prove that all recursive quantum convolutional encoder have catastrophic error propagation. In our constructions, the convolutional codes have thus been chosen to be non-catastrophic and non-recursive. While there is no guarantee that the resulting families of turbo-codes have a minimum distance growing with the number of encoded qubits, from a pragmatic point of view the effective minimum distances of the codes that we have simulated are large enough not to degrade the iterative decoding performance up to reasonable word error rates and block sizes.

I. Introduction

For the fifty years that followed Shannon's landmark paper [38] on information theory, the primary goal of the field of coding theory was the design of practical coding schemes that could come arbitrarily close to the channel capacity. Random codes were used by Shannon to prove the existence of codes approaching the capacity – in fact he proved that the overwhelming majority of codes are good in this sense. For symmetric channels this can even be achieved by linear codes. Unfortunately, decoding a linear code is an NP-hard problem [5], so they have no practical relevance. To render the decoding problem tractable thus imposes the use of codes with even more structure.

The first few decades were dominated by algebraic coding theory. Codes such as Reed-Solomon codes [37] and Bose-Chaudhuri-Hocquenghem codes [20], [7] use the algebraic structure of finite fields to design codes with large minimal distances that have efficient minimal distance decoders. The most satisfying compromise nowadays is instead obtained from families of codes (sometimes referred to as "probabilistic codes") with some element of randomness but sufficiently structured to be suitable for iterative decoding. They display good performances for a large class of error models with a decoding algorithm of reasonable complexity. The most prominent families of probabilistic codes are Gallager's low density parity-check (LDPC) codes [15] and turbo-codes [6]. They are all decoded by a belief propagation algorithm, which albeit sub-optimal has been shown to have astonishing performance even at rates very close to the channel capacity. Moreover, the randomness involved in the code design can facilitate the analysis of their average performance. Indeed, probabilistic codes are in many aspect related to quench-disordered physical systems, so standard statistical physics tools can be called into play [45], [28].

Quantum information and quantum error correction [40], [43], [4], [16], [22] are much younger theories and differ from their classical cousins in many aspects. For instance, there exists a quantum analogue of the Shannon channel capacity called the quantum channel capacity [12], [39], [25], which sets the maximum rate at which quantum information can be sent over a noisy quantum channel. Contrarily to the classical case, we do not know how to efficiently compute its value for channels of practical significance, except for quite peculiar channels such as the quantum erasure channel where it is equal to one minus twice the erasure probability [3]. For the depolarizing channel – the quantum generalization of the binary symmetric channel – random codes do not achieve the optimal transmission rate in general. Instead, they provide a lower bound on the channel capacity, often referred to as the hashing bound. In fact, coding schemes have been designed to reliably transmit information on a depolarization channel in a noise regime where the hashing bound is zero [13], [41].

The stabilizer formalism [16] is a powerful method in which a quantum code on n qubits can be seen as classical linear codes on $2n$ bits, but with a parity-check matrix whose rows are orthogonal relative to a symplectic inner product. Moreover, a special class of stabilizer codes, called CSS codes after their inventors [8], [42], can turn any pair of dual classical linear code into a quantum code with related properties. The stabilizer formalism and the CSS construction allow to import a great deal of knowledge directly from the classical theory, and one may hope to use them to leverage the power of probabilistic coding to the quantum domain. In particular, one may expect that, as in the classical case, quantum analogues of LDPC codes or turbo-codes could perform under iterative decoding as well as random quantum codes, i.e. that they could come arbitrarily close to the hashing bound.

For this purpose, it is also necessary to design a good iterative decoding algorithm for quantum codes. For a special class of noise models considered here – namely Pauli noise models – it turns out that a version of the classical belief propagation algorithm can be applied. For CSS codes in particular, each code in the pair of dual codes can be decoded independently as a classical code. However, this is done at the cost of neglecting some correlations between errors that impact the coding scheme's performances. For some class of stabilizer codes, the classical belief propagation can be improved to exploit the coset structure of degenerate errors which improve the code's performances. This is the case for concatenated block codes [34] and the turbo-codes we consider here, but we do not know how to exploit this feature for LDPC codes for instance. Finally, a quantum belief propagation algorithm was recently proposed [24] to enable iterative decoding of more general (non-Pauli) noise models. As in the classical case, quantum belief propagation also ties in with statistical physics [19], [23], [24], [35].

We emphasize that a fast decoding algorithm is crucial in quantum information theory. In the classical setting, when error correction codes are used for communication over a noisy channel, the decoding time translate directly into communication delays. This has been the driving motivation to devise fast decoding schemes, and is likely to be important in the quantum setting as well. However, there is an important additional motivation for efficient decoding in the quantum setting. Quantum computation is likely to require active stabilization. The decoding time thus translates into computation delays, and most importantly in error suppression delays. If errors accumulate faster than they can be identified, quantum computation may well become infeasible: fast decoding is an essential ingredient to fault-tolerant computation.

The first attempts at obtaining quantum analogues of LDPC codes [27], [9], [18] have not yielded results as spectacular as their classical counterpart. This is due to several reasons. First there are issues with the code design. Due to the orthogonality constraints imposed on the parity-check matrix, it is much harder to construct quantum LDPC codes that classical ones. In particular, constructing the code at random will certainly not do. The CSS construction is of no help since random sparse classical codes do not have sparse duals. In fact, it is still unknown whether there exist families of quantum LDPC codes with non-vanishing rate and unbounded minimum distance. Moreover, all known construction seem to suffer from a poor minimum distances for reasons which are not always fully understood. Second, there are issues with the decoder. The Tanner graph associated to a quantum LDPC code necessarily contains many 4-cycles which are well known for their negative effect on the performances of iterative decoding. Moreover, quantum LDPC codes are by definition highly degenerate but their decoder does not exploit this property: rather it is impaired by it [36].

On the other hand, generalizing turbo-codes to the quantum setting first requires a quantum analogue of convolutional codes. These have been introduced in [10], [11], [30], [31] and followed by further investigations [14], [17], [1]. Quantum turbo-codes can be obtained from the interleaved serial concatenation of convolutional codes. This idea was first introduced in [32]. There, it was shown that, on memoryless Pauli channels, quantum turbo-codes can be decoded similarly to classical serial turbo-codes. One of the motivation behind this work was to overcome some of the problems faced by quantum LDPC codes. For instance, graphical representation of serial quantum turbo-codes do not necessarily contain 4-cycles. Moreover, there is complete freedom in the code parameters. Both of these points are related to the fact that there are basically no restrictions on the choice of the interleaver used in the concatenation. An other advantage over LDPC codes is that the decoder makes explicit use of the coset structure associated to degenerate errors.

Despite these features, the iterative decoding performance of the turbo-code considered in [32] was quite poor, much poorer in fact that results obtained from quantum LDPC codes. The purpose of the present article is to discuss in length several issues omitted in [32], to provide a detailed description of the decoding algorithm, to suggest much better turbo-codes than the one proposed there, and, most importantly, to address the issue of catastrophic error propagation for recursive quantum convolutional encoders.

Non-catastrophic and recursive convolutional encoders are responsible for the great success of parallel and serial classical turbo-codes. In a serial concatenation scheme, an inner convolutional code that is recursive yields turbo-code families with unbounded minimum distance [21], while non-catastrophic error propagation is necessary for iterative decoding convergence. The last point can be circumvented in several ways (by doping for instance, see [44]) and some of these tricks can be adapted to the quantum setting, but are beyond the scope of this paper.

The proof [21] that serial turbo-codes have unbounded minimal-distance carries almost verbatim to the quantum setting. Thus, it is possible to design quantum turbo-codes with polynomially large minimal distances. However, we will demonstrate that all recursive quantum convolutional encoders have catastrophic error propagation. This phenomenon is related to the orthogonality constraints which appear in the quantum setting and to the fact that quantum codes are in a sense coset codes. As a consequence, such encoders are not suitable for (standard) serial turbo-codes schemes.

In our constructions, the convolutional codes are therefore chosen to be non-catastrophic and non-recursive, so there is no guarantee that the resulting families of turbo-codes have a minimum distance which grows with the number of encoded qubits. Despite these limitations, we provide strong numerical evidence that their error probability decreases as we increase the block size at fixed rate – and this up to rather large block sizes. In other words, from a pragmatic point of view, the minimum distances of the codes that we have simulated are large enough not to degrade the iterative decoding performance up to moderate word error rates ($10^{-3} - 10^{-5}$) and block sizes

$(10^2 - 10^4)$.

The style of our presentation is motivated by the intention to accommodate a readership familiar with either classical turbo-codes or quantum information science. This unavoidably implies some redundancy and the expert reader may want to skip some sections, or perhaps glimpse at them to pick up the notation. In particular, the necessary background from classical coding theory and convolutional codes is presented in the next section using the circuit language of quantum information science, and yet requires little departure from more conventional presentations. This, we hope, allows a very smooth transition between classical codes and quantum codes, which are the subject of Sec. III. Whenever possible, the definitions used in the quantum setting directly mirror those established in the classical setting.

Section IV uses the circuit representation to define quantum convolutional codes and their associated state diagram. The state diagram is an important tool to understand the properties of a convolutional code. In particular, the detailed analysis of the state diagram of recursive convolutional encoders performed in Sec. IV-E will lead to the conclusion that they all have catastrophic error propagation. Section V is a detailed presentation of the iterative decoding procedure used for quantum turbo-codes. Finally, our numerical results on the codes' word error rate and spectral properties are presented at Sec. VI.

II. Classical preliminaries

The main purpose of this section is to introduce a circuit representation of convolutional encoders which simplifies the generalization of several crucial notions to the quantum setting. For instance, it allows to define in a straightforward way a state diagram for the quantum analogue of a convolutional code which arises naturally from this circuit representation. This state diagram will be particularly helpful for defining and studying fundamental issues related to turbo-codes such as recursiveness and non-catastrophicity of the constituent convolutional encoders. The circuit representation is also particularly well suited to present the decoding algorithm of quantum convolutional codes.

A. Linear block codes

A rate $\frac{k}{n}$ classical linear code C can be specified by a full-rank $(n-k) \times n$ parity-check matrix H over $\mathbb{F}_2$:

$$C = \{\overline{c} \mid H\overline{c}^T = 0\}. \tag{1}$$

Alternatively, the code can be specified by fixing the encoding of each information word $c \in \mathbb{F}_2^k$ through a linear mapping $c \mapsto \overline{c} = cG$ for some full-rank $k \times n$ generator matrix G over $\mathbb{F}_2$ that satisfies $GH^T = 0$. Since G has rank k, there exists an $n \times k$ matrix over $\mathbb{F}_2$ that we denote by a slight abuse of notation by G^{-1} satisfying $GG^{-1} = \mathbb{1}_k$ where for any integer k, $\mathbb{1}_k$ denotes the $k \times k$ identity matrix. Similarly, since H has rank $n-k$, there exists a $n \times (n-k)$ matrix H^{-1} over $\mathbb{F}_2$ satisfying $HH^{-1} = \mathbb{1}_{n-k}$.

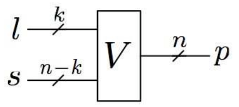

Fig. 1. Circuit representation of encoder $(l:s)V = p$. Slashed wires with integer superscript j indicate a j-bit input/output. The l-bit input are called the logical bits, the other $(n-k)$-bit input are called syndrome or stabilizer bits, and the n-bit output are the physical bits. The string $p \in \mathbb{F}_2^n$ is a codeword if and only if $s = 0_{n-k}$.

Lemma 1: The right inverses H^{-1} and G^{-1} can always be chosen such that $(H^{-1})^T G^{-1} = 0$.

Proof: Let $B = (H^{-1})^T G^{-1}$. The substitution $H^{-1} \to H^{-1} + G^T B^T$ preserves the property $HH^{-1} = \mathbb{1}$ and fulfills the desired requirement. ■

We will henceforth assume that the right inverses H^{-1} and G^{-1} are chosen to fulfill the condition of Lemma 1.

To study the analogy between classical linear binary codes and stabilizer codes, we view a rate $\frac{k}{n}$ classical linear code and its encoding in a slightly unconventional fashion. We specify the encoding by an $n \times n$ invertible *encoding* matrix V over $\mathbb{F}_2$. The code space is defined as

$$C = \big\{\overline{c} = (c : 0_{n-k})V \mid c \in \mathbb{F}_2^k\big\}, \tag{2}$$

where we use the following notation.

Notation 1: For an n-tuple $a \in \mathscr{A}^n$ and an m-tuple $b \in \mathscr{A}^m$ over some alphabet $\mathscr{A}$, we denote by $a : b$ the $n+m$-tuple formed by the concatenation of a followed by b.

Given the generator matrix G and parity check matrix H of a code, the encoding matrix V can be fixed to

$$V = \begin{pmatrix} G \\ (H^{-1})^T \end{pmatrix}. \tag{3}$$

This matrix is invertible:

$$V^{-1} = \big(G^{-1}, H^T\big) \tag{4}$$

and satisfies $VV^{-1} = \mathbb{1}_n$ following Lemma 1. Clearly, the encoding matrix $V : \mathbb{F}_2^n \to \mathbb{F}_2^2$ specifies both the code space and the encoding. The output $b = aV$ of the encoding matrix V is in the code space if and only if the input is of the form $a = (c : 0_{n-k})$ where $c \in \mathbb{F}_2^k$. This follows from the equalities $aV = cG = \overline{c} \in C$ and $(c : s)VH^T = s$.

The encoding matrix also specifies the syndrome associated to each error. When transmitted on a bit-flip channel, a codeword $\overline{c}$ will result in the message $m = \overline{c} + p$ for some $p \in \mathbb{F}_2^n$. The error p can be decomposed into an error syndrome $s \in \mathbb{F}_2^{n-k}$ and a logical error $l \in \mathbb{F}_2^k$ as $pV^{-1} = (l : s)$. This is conveniently represented by the circuit diagram shown at Fig. 1, in which time flows from left to right. In such diagrams, the inverse V^{-1} is obtained by reading the circuit from right to left, running time backwards. This circuit representation is at the core of our construction of quantum turbo codes, it greatly simplifies all definition and analysis.

A probability distribution $\mathbf{P}(p)$ on the error p incurred during transmission induces a probability distribution on logical

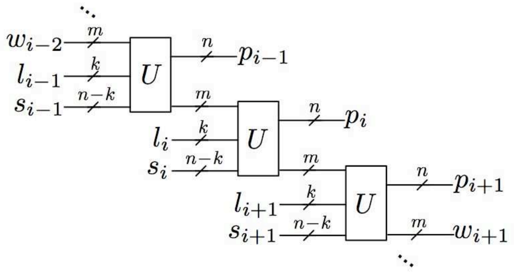

Fig. 2. Circuit diagram of a convolutional encoder with seed transformation U.

transformation and syndromes

$$\mathbf{P}(l,s) = \mathbf{P}(p)\Big|_{p=(l:s)V^{-1}}. \tag{5}$$

We call $\mathbf{P}(l,s)$ the *pullback* of the probability $\mathbf{P}(p)$ through the gate V. Maximum likelihood decoding $l_{ML} : \mathbb{F}_2^{n-k} \to \mathbb{F}_2^k$ consists in identifying the most likely logical transformation l given the syndrome s

$$l_{ML}(s) = \text{argmax}_l \mathbf{P}(l|s) \tag{6}$$

where the conditional probability is defined the usual way

$$\mathbf{P}(l|s) = \frac{\mathbf{P}(l,s)}{\sum_{l'} \mathbf{P}(l',s)}. \tag{7}$$

Similarly, we can define the bit-wise maximum likelihood decoder $l^i_{ML} : \mathbb{F}_2^{n-k} \to \mathbb{F}_2$ which performs a local optimization on each logical bit

$$l^i_{ML}(s) = \text{argmax}_{l^i} \mathbf{P}(l^i|s), \tag{8}$$

where the marginal conditional probability is defined the usual way

$$\mathbf{P}(l|s) = \sum_{l^1,\ldots l^{i-1},l^{i+1},\ldots l^k} \mathbf{P}(l^1,\ldots l^k|s). \tag{9}$$

B. Convolutional codes

We define now a convolutional code as a linear code whose encoder V has the form shown at Fig. 2. The circuit is built from repeated uses of a linear invertible seed transformation $U : \mathbb{F}_2^{n+m} \to \mathbb{F}_2^{n+m}$ shifted by n bits. In this circuit, particular attention must be paid to the order of the inputs as they alternate between syndrome bits and logical bits. The total number of identical repetition is called the duration of the code and is denoted N. The m bits that connect gates from consecutive "time slices" are called memory bits. The encoding is initialized by setting the first m memory bits to $w_0 = 0_m$. There are several ways to terminate the encoding, but we here focus on a padding technique. This simply consists in setting the k logical bits of the last t time slices $i = N+1, N+2, \ldots N+t$ equal to $l_i = 0_k$, where t is a free parameter independent of N. The rate of the code is thus $k/n + O(1/N)$.

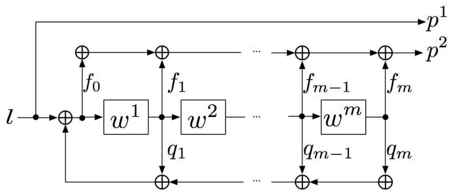

Fig. 3. Representation of convolutional encoder as a linear filter. The labels f and q take value 0 and 1 and indicate respectively the absence or presence of the associated wire. Although linear, this transformation is not invertible.

Note that in this diagram, we use a *subscript* to denote the different elements of a stream. For instance, p_i denotes the n-bit output string at time i. The jth bits of p_i would be denoted by a *subscript* as p_i^j, or simply p^j when the particular time i is clear from context. This convention will be used throughout the paper.

This definition of convolutional code differs at first sight from the usual one based on linear filters built from shift register and feed-back lines. An example of a linear filter for a rate 1/2 (systematic and recursive) convolutional encoder is shown at Fig. 3. An other common description of this encoder would be in terms of its rational transfer function which related the D-transform of the output $p(D)$ to that of the input $l(D)$. Remember that the D-transform of a bit stream $x_1 : x_2 : x_3 : \ldots$ is given by $x(D) = \sum_i x_i D^i$. For the code of Fig. 3, the output's D-transforms are

$$p^1(D) = l(D) \tag{10}$$
$$p^2(D) = \frac{f_0 + f_1 D + \ldots + f_m D^m}{1 + q_1 D + \ldots + q_m D^m} l(D) \tag{11}$$

where the inverse is the Laurent series defined by long division. The code can also be specified by the recursion relation

$$\begin{aligned}
w_i^j &= w_{i-1}^{j-1} \text{ for } j > 1 \\
w_i^1 &= l_i + \sum_{j=1}^m q_j w_{i-1}^j \\
p_i^2 &= f_0(\sum_{j=1}^m q_j w_{i-1}^j + l_i) + \sum_{j=1}^m f_j w_{i-1}^j \\
&= f_0 l_i + \sum_{j=1}^m (f_j + f_0 q_j) w_{i-1}^j.
\end{aligned}$$

These definitions are in fact equivalent to the circuit of Fig. 2 with the seed transformation U specified by Fig. 4. Note that we can assume without lost of generality that $f_m = 1$ or $q_m = 1$ (or both), and these two cases lead to different seed transformations. The generalization to arbitrary linear filters is straightforward. In terms of matrices, the seed transformation associated to this convolutional code encodes the relation $(p_i : w_i) = (w_{i-1} : l_i : s_i)U$ with U given by

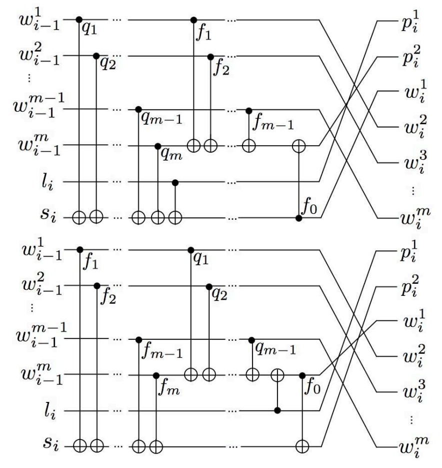

Fig. 4. Seed transformation circuit for convolutional code of Fig 3. Top: Case $f_m = 1$. Bottom: Case $q_m = 1$. The labels f and q take value 0 and 1 and indicate respectively the absence or presence of the associated gate. Both circuits are entirely built from controlled-nots, and is therefore invertible. As its name indicates, the controlled-not acts by negating the target bit $\oplus$ if and only if the control bit $\bullet$ is in state 1.

$$U = \left(\begin{array}{cc} \overbrace{\mu_{\rm P}}^{n} & \overbrace{\mu_{\rm M}}^{m} \\ \Lambda_{\rm P} & \Lambda_{\rm M} \\ \Sigma_{\rm P} & \Sigma_{\rm M} \end{array} \right) \begin{array}{l} \}_m \\ \}_k \\ \}_{n-k} \end{array} . \tag{12}$$

where

$$\mu_{\rm P} = \begin{pmatrix} 0 & f_1 + f_0 q_1 \\ \vdots & \vdots \\ 0 & f_m + f_0 q_m \end{pmatrix}, \ \mu_{\rm M} = \begin{pmatrix} q_1 & & & \\ q_2 & & \mathbb{1}_{m-1} & \\ \vdots & & & \\ q_m & 0 & 0 & 0 \end{pmatrix},$$

$\Lambda_{\rm P} = (1, f_0)$, and $\Lambda_{\rm M} = (1\ 0_{m-1})$. The two other components depend on whether $f_m = 1$ or $q_m = 1$. In the former case $\Sigma_{\rm P} = (0, f_0)$ and $\Sigma_{\rm M} = (1\ 0_{m-1})$ while in the latter case $\Sigma_{\rm P} = (0, 1)$ and $\Sigma_{\rm M} = (0_m)$.

Not only does the circuit of Fig. 2 produce the same encoding as the linear filter of Fig. 3, it also has the same memory states. More precisely, the value contained in the jth shift register at time i in Fig. 3 is equal to the value of the jth memory bit between gate i and $i+1$ on Fig. 2. This is important because it allows to define the state diagram (see Sec. IV-B) directly from the circuit diagram Fig. 4.

Of particular interest are systematic recursive encoders that are defined as follows.

Definition 1 (Systematic encoder): An encoder is *systematic* when the input stream is a sub-stream of the output stream.

Definition 2 (Recursive encoder): A convolutional encoder is *recursive* when its rational transfer function involves genuine Laurent series (as opposed to simple polynomials).

Systematic encoders copy the input stream in clear in one of the output stream. Typically they have transfer functions of the form $p^j(D) = l^j(D)$ for $j = 1, \ldots, k$ and arbitrary $p^j(D)$ for $j > k$, so p_i^j is a copy of l_i^j. The systematic character of the code considered in the above example is most easily seen from Fig. 3: p^1 is a copy of the input l. Systematic encoders are used to avoid catastrophic error propagation. This term will be defined formally in the quantum setting, but it essentially means that an error affecting a finite number of physical bits is mapped to a logical transformation on an infinite number of logical bits by the encoder inverse. Catastrophic encoders cannot be used directly in standard turbo-code schemes. The problem is that the first iteration of iterative decoding does not provide information on the logical bits. This is due to the fact that as the length of the convolutional encoder tends to infinity and in the absence of prior information about the value of the logical bits, the logical bit error rate after decoding tends to $\frac{1}{2}$.

A recursive encoder has an infinite impulsive response: on input l of Hamming weight 1, it creates an output of infinite weight for a code of infinite duration N. Recursiveness is also related to the presence of feed-back in the encoding circuit, which is easily understood from the linear filter of Fig. 3. Except when the polynomial $\sum q_i D^i$ factors $\sum f_i D^i$, an encoder with feed-back will be recursive. It is essential to use as constituent recursive convolutional codes in classical turbo-codes schemes to obtain families of turbo-codes of unbounded minimum distance and with performances which improve with the block size.

III. Quantum Mechanics and Quantum Codes

In this section, we review some basic notions of quantum mechanics, the stabilizer formalism, and the decoding problem for quantum codes. The stabilizer formalism is introduced in two steps. In Sec. III-B, stabilizer codes are defined the usual way, as subspaces of the Hilbert space stabilized by an Abelian subgroup of the Pauli group. In the following Sec. III-C, an equivalent definition of a stabilizer code is derived in terms of its action on the effective Pauli group. Although redundant for quantum information scientists, this establishes a clear relation between classical codes and quantum stabilizer codes, and allows to highlight the crucial distinctions. Particular emphasis is put on the role of the encoder because it leads to a straightforward quantization of convolutional codes and turbo-codes. The encoder also provides an intuitive picture for the logical cosets, which are an important distinction between classical codes and quantum stabilizer codes.

A. Qubits and the Pauli group

A *qubit* is a physical system whose state is described by a unit-length vector in a two-dimensional Hilbert space. The two vectors of a given orthonormal basis are conventionally denoted by $|0\rangle$ and $|1\rangle$. We identify the Hilbert space with $\mathbb{C}^2$ in the usual way with the help of such a basis. The state of a system comprising n qubits is an unit-lenght vector in the tensor product of n two-dimensional Hilbert spaces. It is a space of dimension 2^n which can be identified with $(\mathbb{C}^2)^{\otimes n} \simeq \mathbb{C}^{2^n}$. It has a basis given by all tensor products of the form $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$, where the $x_i \in \{0, 1\}$ and the inner product between two basis elements $|x_1\rangle \otimes \cdots \otimes |x_n\rangle$ and $|y_1\rangle \otimes \cdots \otimes |y_n\rangle$ is the product of the inner products of $|x_i\rangle$ with the corresponding $|y_i\rangle$. In other words, this basis is orthonormal.

The error model we consider in this paper is a Pauli-memoryless channel which is defined with the help of the three Pauli matrices

$$\mathcal{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \; \mathcal{Y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \; \mathcal{Z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

These matrices anti-commute with each other and satisfy the following multiplication table

$\times$	$\mathcal{X}$	$\mathcal{Y}$	$\mathcal{Z}$
$\mathcal{X}$	$\mathcal{I}$	$i\mathcal{Z}$	$-i\mathcal{Y}$
$\mathcal{Y}$	$-i\mathcal{Z}$	$\mathcal{I}$	$i\mathcal{X}$
$\mathcal{Z}$	$i\mathcal{Y}$	$-i\mathcal{X}$	$\mathcal{I}$

where $\mathcal{I}$ denotes the 2×2 identity matrix. The action of these operators on the state of a qubit is obtained by right multiplication $|\psi\rangle \to \mathcal{P}|\psi\rangle$, with $|\psi\rangle$ viewed as an element of $\mathbb{C}^2$.

These matrices generate the Pauli group $\mathscr{G}_1$ which is readily seen to be the set

$$\{\pm\mathcal{I}, \pm i\mathcal{I}, \pm\mathcal{X}, \pm i\mathcal{X}, \pm\mathcal{Y}, \pm i\mathcal{Y}, \pm\mathcal{Z}, \pm i\mathcal{Z}\}.$$

They also form all the errors which may affect one qubit in our error model. If we have an n-qubit system then the errors which may affect it belong to the Pauli group $\mathscr{G}_n$ over n qubits which is defined by

$$\begin{aligned} \mathscr{G}_n &= \mathscr{G}_1^{\otimes n} \\ &= \{\epsilon\mathcal{P}_1 \otimes \cdots \otimes \mathcal{P}_n | \epsilon \in \{\pm 1, \pm i\}, \mathcal{P}_i \in \{\mathcal{I}, \mathcal{X}, \mathcal{Y}, \mathcal{Z}\}\} \end{aligned}$$

This group is generated by i and the set of $\mathcal{X}_i$'s and $\mathcal{Z}_i$'s for $i = 1, 2, \dots, n$ which are defined by:

Notation 2:

$$\begin{aligned} \mathcal{X}_i &\triangleq \overbrace{\mathcal{I} \otimes \cdots \otimes \mathcal{I}}^{i-1 \text{ times}} \otimes \mathcal{X} \otimes \overbrace{\mathcal{I} \otimes \cdots \otimes \mathcal{I}}^{n-i \text{ times}} \\ \mathcal{Z}_i &\triangleq \overbrace{\mathcal{I} \otimes \cdots \otimes \mathcal{I}}^{i-1 \text{ times}} \otimes \mathcal{Z} \otimes \overbrace{\mathcal{I} \otimes \cdots \otimes \mathcal{I}}^{n-i \text{ times}} \end{aligned}$$

In quantum mechanics two states are physically indistiguishable if they differ by a multiplicative constant. This motivates the definition another group of errors, called the effective Pauli group, obtained by taking the quotient of $\mathscr{G}_n$ by $\{\pm\mathcal{I}, \pm i\mathcal{I}\}$.

Definition 3 (Effective Pauli group): The effective Pauli group G_n on n qubits is the set of equivalence classes $[\mathcal{P}]$ for $\mathcal{P}$ in $\mathscr{G}_n$, where the equivalence class $[\mathcal{P}]$ is the set of elements of $\mathscr{G}_n$ which differ from $\mathcal{P}$ by a multiplicative constant. We will also use the notation $I \triangleq [\mathcal{I}], X \triangleq [\mathcal{X}], Y \triangleq [\mathcal{Y}], Z \triangleq [\mathcal{Z}]$ and $X_i = [\mathcal{X}_i], Z_i = [\mathcal{Z}_i]$.

All the effective Pauli groups G_n are Abelian. $(G_1, +)$ is isomorphic to $(\mathbb{F}_2 \times \mathbb{F}_2, +)$ where the group operation of G_1 corresponds to bitwise addition over $\mathbb{F}_2 \times \mathbb{F}_2$. As a consequence effective Pauli operators will always be represented in the following way: $I \leftrightarrow (0,0)$, $X \leftrightarrow (1,0)$, $Y \leftrightarrow (1,1)$ and $Z \leftrightarrow (0,1)$. Since $G_n \cong G_1^n$, we view $P \in G_n$ either as an n-tuple $(P^i)_{i=1}^n$ with entries in G_1 or as $2n$-tuple with entries in $\mathbb{F}_2$ obtained by replacing each P_i by its corresponding binary representation. G_n is generated by the X_i and Z_i, and we introduce the following notation.

Notation 3: For P in G_n, we denote by P^x and P^z the only elements of G_n satisfying:

1) $P = P^x + P^z$, and
2) $P^x \in \{I, X\}^n, P^z \in \{I, Z\}^n$.

An important property of $\mathscr{G}_n$ is that any pair of elements $\mathcal{P}, \mathcal{Q}$ either commutes or anti-commutes. This leads to the definition of an inner product "$\star$" for elements $P = (P_i)_{1 \le i \le n}$ and $Q = (Q_i)_{1 \le i \le n}$ of G_n such that $P \star Q = \sum_{i=1}^n P_i \star Q_i \bmod 2$. Here, $P_i \star Q_i = 1$ if $P_i \neq Q_i$, $P_i \neq I$ and $Q_i \neq I$; and $P_i \star Q_i = 0$ otherwise.

Fact 1: $\mathcal{P}, \mathcal{Q} \in \mathscr{G}_n$ commute if and only if $[\mathcal{P}] \star [\mathcal{Q}] = 0$.

This product can also be defined with the help of the following matrix which will appear again later in the definition of symplectic matrices.

Notation 4:

$$\Lambda_n \triangleq \mathbb{1}_n \otimes \mathcal{X}.$$

By viewing now elements of G_n as binary $2n$-tuples we have:

Definition 4 (Inner product): Define the inner product $\star : G_n \times G_n \to \mathbb{F}_2$ by $P \star Q = P\Lambda_n Q^T$.

G_n is an $\mathbb{F}_2$-vector space and we use the $\star$ inner product to define the orthogonal space of a subspace of G_n as follows.

Definition 5 (Orthogonal subspace): Let V be a subset of G_n. We define $V^\perp$ by

$$V^\perp \triangleq \{P \in G_n : P \star Q = 0 \text{ for every } Q \in V\}.$$

$V^\perp$ is always a subspace of G_n and if the space spanned by V is of dimension t, then $V^\perp$ is of dimension $2n - t$.

From the fact that two states are indistiguishable if they differ by a mutiplicative constant, a Pauli error may only be specified by its effective Pauli group equivalence to which it belongs. A very important quantum error model is the *depolarizing channel*. It is in a sense the quantum analogue of the binary symmetric channel.

Definition 6 (Depolarizing channel): The depolarizing channel on n qubits of error probability p is an error model where all the errors which occur belong to G_n and the probability that a particular element P is chosen is equal to $(1-p)^{n-\text{weight}(P)} \left(\frac{p}{3}\right)^{\text{weight}(P)}$ where weight(P) is the number of coordinates of P which differ from I.

In other words, the coordinates of the error are chosen independently: there is no error on a given coordinate with probability $1-p$ and there is an error on it of type X, Y or Z each with probability $\frac{p}{3}$.

B. Stabilizer codes: Hilbert space perspective

A rate $\frac{k}{n}$ quantum error correction code is a 2^k dimensional subspace $\mathscr{C}$ of $(\mathbb{C}^2)^{\otimes n}$. It can be specified by a unitary transformation $\mathcal{V} : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$:

$$\mathscr{C} = \left\{ |\overline{\psi}\rangle = \mathcal{V}(|\psi\rangle \otimes |0_{n-k}\rangle) \mid |\psi\rangle \in \mathbb{C}^{2^k} \right\}. \tag{13}$$

This definition directly reflects Eq. (2). As in the classical case, the matrix $\mathcal{V}$ specifies not only the code but also the encoding, that is the particular embedding $(\mathbb{C}^2)^{\otimes k} \to (\mathbb{C}^2)^{\otimes n}$.

An importance distinction however is that in the quantum case, the dimension of the matrix $\mathcal{V}$ is exponential in the number of qubits n. To obtain an efficiently specifiable code, we choose $\mathcal{V}$ from a subgroup of the unitary group over $(\mathbb{C}^2)^{\otimes n}$ called the Clifford group. In fact, not only are Clifford transformations over n qubits efficiently specifiable, they can also be implemented efficiently by a quantum circuit involving only $O(n^2)$ elementary quantum on 1 and 2 qubits (see Theorem 10.6 in [29] for instance).

Definition 7 (Clifford transformation and Clifford group): A *Clifford transformation* over n qubits is a unitary transform $\mathcal{V}$ over $(\mathbb{C}^2)^{\otimes n}$ which leaves the Pauli group over n qubits globally invariant by conjugation

$$\mathcal{V}\mathscr{G}_n\mathcal{V}^\dagger = \mathscr{G}_n.$$

The set of Clifford transformations is a group and is called the *Clifford group* over n qubits.

This definition naturally leads to the action of the Clifford group on elements of the Pauli group.

Definition 8 (Action of Clifford transformation on Pauli): A Clifford transformation $\mathcal{V}$ acts on the Pauli group as

$$\begin{array}{rcl} \mathscr{G}_n & \rightarrow & \mathscr{G}_n \\ \mathcal{P} & \mapsto & \mathcal{P}' = \mathcal{V}\mathcal{P}\mathcal{V}^\dagger \end{array}$$

It also acts on the effective Pauli group by the mapping $[\mathcal{P}] \mapsto [\mathcal{P}']$.

The last mapping is $\mathbb{F}_2$-linear and there is a square binary matrix V of size $2n$ which is such that

$$[\mathcal{V}\mathcal{P}\mathcal{V}^\dagger] = [\mathcal{P}]V.$$

This matrix will be called the *encoding matrix*.

Definition 9 (Encoding matrix): The encoding matrix V associated to an encoding operation $\mathcal{V}$, which is a Clifford transformation over n qubits, is the binary matrix V of size $2n \times 2n$ such that for any $\mathcal{P} \in \mathscr{G}_n$ we have

$$[\mathcal{V}\mathcal{P}\mathcal{V}^\dagger] = [\mathcal{P}]V.$$

Clearly then, a Clifford transformation on n qubits can be specified by its associated encoding matrix V on $\mathbb{F}_2^{2n}$ together with a collection of $2n$ phases. This shows that Clifford transformations are efficiently specifiable as claimed. It can readily be verified that the rows of V, denoted V_i $i = 1, 2, \ldots, 2n$, are equal to

$$\begin{aligned} V_{2i-1} &= [\mathcal{V}\mathcal{X}_i\mathcal{V}^\dagger], && (14)\\ V_{2i} &= [\mathcal{V}\mathcal{Z}_i\mathcal{V}^\dagger]. && (15) \end{aligned}$$

Since conjugation by a unitary matrix $\mathcal{V}$ does not change the commutation relations, the above equations implies that the encoding matrix is a symplectic matrix, whose definition is recalled below.

Definition 10 (Symplectic transformation): A n-qubit *symplectic transformation* is a $2n \times 2n$ matrix U over $\mathbb{F}_2$ that satisfies

$$U\Lambda_n U^T = \Lambda_n.$$

By definition, symplectic transformation are invertible and preserve the inner product $\star$ between n-qubit Pauli group elements. Conversely, every symplectic matrices always correspond to a (non-unique) Clifford transformation.

A stabilizer code is thus a quantum code specified by Eq. (13), but with $\mathcal{V}$ in the Clifford group. The code $\mathscr{C}$ (but not the encoding) can equivalently be specified with $n-k$ independent mutually commuting elements of $\mathscr{G}_n$ of order 2 as follows:

Definition 11 (Stabilizer code): The stabilizer code $\mathscr{C}$ associated to the stabilizer set $\{\mathcal{H}_i, i = 1..n-k\}$, where the $\mathcal{H}_i$'s are independent mutually commuting elements of $\mathscr{G}_n$ of order 2 and different from -1, is the subspace of $(\mathbb{C}^2)^{\otimes n}$ of elements stabilized by the $\mathcal{H}_i$'s, that is

$$\mathscr{C} = \{|\overline{\psi}\rangle \mid \mathcal{H}_i|\overline{\psi}\rangle = |\overline{\psi}\rangle, 1 \le i \le n-k\}. \qquad (16)$$

This is the usual definition of stabilizer codes. The $\mathcal{H}_i$ play a role analogous to the rows of the parity-check matrix of a classical linear code, and this connection will be formalized in the next section. To see the equivalence between this definition and Eq. (13), set $\mathcal{H}_i = \mathcal{V}Z_{k+i}\mathcal{V}^\dagger$. These operators are independent and of order 2 since they are conjugate to the $\mathcal{Z}_i$ which are independent and of order 2. Now, consider a $|\overline{\psi}\rangle \in \mathscr{C}$ as defined in Eq. (13). For all $\mathcal{H}_i$, we have

$$\begin{aligned} \mathcal{H}_i|\overline{\psi}\rangle &= \mathcal{V}\mathcal{Z}_{i+k}\mathcal{V}^\dagger\mathcal{V}(|\psi\rangle \otimes |0_{n-k}\rangle) && (17)\\ &= \mathcal{V}(|\psi\rangle \otimes \mathcal{Z}_i|0_{n-k}\rangle) = |\overline{\psi}\rangle, && (18) \end{aligned}$$

where we used the fact that $\mathcal{Z}|0\rangle = |0\rangle$. Hence, $|\overline{\psi}\rangle$ satisfies the condition of Def. 11. Conversely, for any state $|\overline{\psi}\rangle \in \mathscr{C}$ according to Def. 11, we have

$$\begin{aligned} \mathcal{Z}_{k+i}\mathcal{V}^\dagger|\overline{\psi}\rangle &= \mathcal{V}^\dagger\mathcal{H}_i|\overline{\psi}\rangle && (19)\\ &= \mathcal{V}^\dagger|\overline{\psi}\rangle, && (20) \end{aligned}$$

which implies that the $k+i$th qubit of $\mathcal{V}^\dagger|\overline{\psi}\rangle$ must be in state $|0\rangle$. Since this holds for all $i = 1, 2, \ldots, n-k$, we conclude that the two definitions are equivalent. This equivalence has the following consequence:

Fact 2: A stabilizer code of length n associated to $n-k$ independent generators $\mathcal{H}_i$ is of dimension 2^k.

Since $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ are all of order 2, all the generators of order 2 in $\mathscr{G}_n$ are of the form $\pm\mathcal{P}$ where $\mathcal{P}$ is a tensor product of n matrices all chosen among the set $\{\mathcal{I}, \mathcal{X}, \mathcal{Y}, \mathcal{Z}\}$. Thus, we can specify the generators $\mathcal{H}_i$ of the stabilizer code by giving only the associated effective Pauli group elements together with a sign for each generator. Changing the sign of a stabilizer generator changes the code, but not its properties[1]. More precisely, the set of Pauli errors which can be corrected by such a code does not depend on the signs which have been chosen. Hence, we can specify a family of "equivalent" codes by specifying instead of the $\mathcal{H}_j$'s the set of $H_j \triangleq [\mathcal{H}_j] = VZ_{j+k}$. It is important to note that these elements have to be orthogonal: the fact that the $\mathcal{H}_i$'s commute translate into the orthogonality condition $H_i \star H_j = 0$. Thus, the H_i span a linear space called the stabilizer space, that we denote $C(I)$ for reasons that will become apparent later.

[1]This is strictly true for Pauli channels which are considered here. For a general noise model, error correcting properties may actually depend on the sign of the stabilizer generators.

Thus, in analogy with classical linear codes, a stabilizer code (or more precisely an equivalent class thereof) can be efficiently specified by an encoding matrix V on $\mathbb{F}_2^{2n}$. This matrix also provides an efficient description of the encoding up to a set of phases. There is another analogy with a classical encoding matrix that will be crucial for our definition of quantum turbo-codes. Assume that we concatenate two stabilizer codes and that these codes are encoded by Clifford transformations. The result of the concatenation is also a stabilizer code (because Clifford transformations form a group) and the resulting encoding matrix is just the product of the two encoding matrices of each constituent code. This reflects the fact that the encoding matrices provide a representation of the Clifford group.

Fact 3: Let $\mathcal{V}_1$ and $\mathcal{V}_2$ be two Clifford transformations over n qubits with encoding matrices V_1 and V_2 respectively. Then $\mathcal{V}_2\mathcal{V}_1$ is a Clifford transformation with encoding matrix V_1V_2.

Proof: Consider the Clifford transformation $\mathcal{V} \triangleq \mathcal{V}_2\mathcal{V}_1$. It suffices to verify the statement on a generating set of the Pauli group:

$$\begin{aligned}\left[\mathcal{V}\mathcal{X}_i\mathcal{V}^\dagger\right] &= \left[\mathcal{V}_2\mathcal{V}_1\mathcal{X}_i\mathcal{V}_1^\dagger\mathcal{V}_2^\dagger\right] \\ &= \left[\mathcal{V}_1\mathcal{X}_i\mathcal{V}_1^\dagger\right]V_2 \qquad (21)\\ &= X_iV_1V_2\end{aligned}$$

Equation (21) uses the fact that $\mathcal{V}_1\mathcal{X}_i\mathcal{V}_1^\dagger$ belongs to $\mathscr{G}_n$. The same kind of result holds for the $\mathcal{Z}_i$'s and this completes the proof. ■

C. Stabilizer codes: effective Pauli group perspective

As presented in the previous section, stabilizer codes are a special class of quantum codes that, in analogy to classical linear codes, can be specified by an encoding matrix V on $\mathbb{F}_2^{2n}$. However, the analogy between stabilizer codes and classical linear codes is much richer and is best seen by considering stabilizer codes from the perspective of the effective Pauli group. The main advantage of this perspective is that the effective Pauli group G_n is discrete, its elements can be specified by $2n$-bit strings. This section will present stabilizer codes from this perspective, which will then be used in the rest of this paper. Whenever possible, the presentation will directly mirror that of section II.

The crucial point is that the Pauli operators form a linear basis for the linear operators on $(\mathbb{C}^2)^{\otimes n}$, any $2^n \times 2^n$ matrix can be expressed as

$$A = \sum_{\mathcal{P}\in\mathscr{G}_n} \alpha_{\mathcal{P}}\mathcal{P} \qquad (22)$$

for some set of complex $\alpha_{\mathcal{P}}$. Some operators on $(\mathbb{C}^2)^{\otimes n}$leave $\mathscr{C}$ globally invariant. They are obtained from Eq. (22) with $\alpha_{\mathcal{P}} \neq 0$ only for those $\mathcal{P}$ satisfying $[\mathcal{P}] \star H_j = 0$ for all $j = 1, 2, \ldots, n-k$. Indeed, it is easy to see that such operators map $\mathscr{C}$ to itself. Consider a state $|\overline{\psi}\rangle \in \mathscr{C}$, and suppose that $[\mathcal{P}] \star H_j = 0$ for all $j = 1, 2, \ldots, n-k$. Then

$$\begin{aligned}\mathcal{H}_i\mathcal{P}|\overline{\psi}\rangle &= (-1)^{([\mathcal{H}_i]\star[\mathcal{P}])}\mathcal{P}\mathcal{H}_i|\overline{\psi}\rangle \qquad (23)\\ &= \mathcal{P}|\overline{\psi}\rangle \qquad (24)\end{aligned}$$

so $\mathcal{P}|\overline{\psi}\rangle$ is in $\mathscr{C}$. The fact that all operators that map $\mathscr{C}$ to itself have this form follows directly from the fact that the Pauli group forms a basis of operators.

Hence, the effect of the noise on $\mathscr{C}$ can be fully characterized by its effect on the Pauli operators that leave the codespace globally invariant. This motivates the following definition of a stabilizer code:

Definition 12 (Discrete stabilizer code): The discrete stabilizer code C associated to the stabilizer set $\{H_i, i = 1..n-k\}$, where the H_i's are independent mutually orthogonal elements of G_n, is the subspace of G_n orthogonal to the H_i, that is

$$C = \{P \in G_n \mid H_i \star P = 0, 1 \leq i \leq n-k\}, \qquad (25)$$

or more succinctly $C = C(I)^\perp$.

Notation 5: Given a quantum code C, a *codeword* is an effective Pauli operator that is an element of the code.
Thus, the code consists of those effective Pauli operators that map $\mathscr{C}$ to itself. When the combined effect of a noise operator followed by error correction leaves each element of C invariant, then we can conclude that it does not affect the elements of $\mathscr{C}$. This is the main advantage of the stabilizer formalism: it allows to discretize a seemingly continuous problem by characterizing the errors on the (continuous) set $\mathscr{C}$ through their effect on the discrete set C. Viewing effective Pauli operators as elements of $\mathbb{F}_2^{2n}$, we can define the parity-check matrix of the stabilizer code as follows.

Definition 13 (Parity-check matrix): The parity-check matrix H of a quantum code with stabilizer set $\{H_1, \ldots, H_{n-k}\}$ is the binary matrix of size $(n-k) \times 2n$ with rows $H_1, \ldots, H_{n-k}$.

The encoding matrix V provides a succinct definition of the stabilizer space $C(I)$ and the code space C:

$$\begin{aligned}C(I) &= \left\{P = (I_n : S^z)V \mid S^z \in \{I, Z\}^{n-k}\right\}\\ C &= \left\{\overline{L} = (L : I_{n-k})V \mid L \in G_k\right\} + C(I).\end{aligned}$$

Notice how this definition of the code C is almost identical to the classical definition Eq. (2), save for the presence of the additional $C(I)$. Just like in the classical case, the parity-check matrix is a sub-matrix of the encoding matrix V:

Fact 4: The parity-check matrix of a stabilizer with encoding matrix V is the sub-matrix obtained from the rows $\{j = 2i : i = k+1, k+2, \ldots n\}$ of V.
This follows from the fact that $\mathcal{H}_j = \mathcal{V}Z_{k+j}\mathcal{V}^\dagger$ and Eq. (14).

D. Decoding

As in the classical setting, decoding a quantum code is syndrome-based. When a Pauli error $\mathcal{P}_n \in \mathscr{G}_n$ affects the encoded register, it is possible to perform a physical measurement whose outcome is called the *error syndrome*, which reveals partial information about the error. The error syndrome can be expressed in terms of the parity-check matrix:

Definition 14 (Syndrome): The *syndrome* $s(\mathcal{P})$ of an error $\mathcal{P} \in \mathscr{G}_n$ for a stabilizer code with parity-check matrix $\{H_1, \ldots, H_{n-k}\}$ is defined by

$$s(\mathcal{P}) \triangleq ([\mathcal{P}] \star H_i)_{1\leq i\leq n-k}.$$

The set of (effective) Pauli errors with zero syndrome is the code itself $C = C(I)^\perp$. It forms a $\mathbb{F}_2$-linear subspace. The orthogonality properties of the H_i's imply that $C(I) \subset C(I)^\perp = C$. Just like in the classical setting, C forms the set of *undetectable errors*. There is however an important distinction: not all undetectable errors different from the identity are harmful. Indeed errors in $C(I)$ cannot be detected, but they do not affect the encoded state by definition, c.f. Eq. (16). Hence, $C(I)$ is the set of *harmless undetectable errors*. It may be checked that all other undetectable affect at least a subspace of the code. Hence, $C \setminus C(I)$ is the set of *harmful undetectable errors*. Let us now describe the decoding problem in some details.

When transmitted on a Pauli channel, an encoded state $|\overline{\psi}\rangle = \mathcal{V}(|\psi\rangle \otimes |0_{n-k}\rangle)$ (where $|\psi\rangle$ belongs to $\mathbb{C}^{2^k}$) will result in a state $\mathcal{P}|\overline{\psi}\rangle$ for some $\mathcal{P} \in \mathscr{G}_n$. Upon inverting the encoding we obtain the state

$$\begin{aligned} \mathcal{V}^\dagger \mathcal{P}|\overline{\psi}\rangle &= \mathcal{V}^\dagger \mathcal{P}\mathcal{V}\mathcal{V}^\dagger|\overline{\psi}\rangle & (26)\\ &= \mathcal{V}^\dagger \mathcal{P}\mathcal{V}(|\psi\rangle \otimes |0_{n-k}\rangle) & (27)\\ &= (\mathcal{L}|\psi\rangle) \otimes (\mathcal{S}|0_{n-k}\rangle) & (28)\end{aligned}$$

for some $\mathcal{L} \in \mathscr{G}_k$ and $\mathcal{S} \in \mathscr{G}_{n-k}$. Since global phases are irrelevant, we can consider instead the effective Pauli error $P = [\mathcal{P}]$ with corresponding decomposition $PV^{-1} = (L : S)$, with $L \in G_k$ and $S \in G_{n-k}$. Like in the classical case, this is conveniently represented by the circuit diagram of Fig. 5.

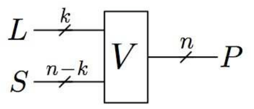

Fig. 5. Circuit representation of encoder $(L : S)V = P$. The operator $P \in G_n$ is a codeword if and only if $S \in \{I, Z\}_{n-k}$.

At this point however, the analogy with the classical case partially breaks down. As described in Section II, in the classical setting a bit-flip error p can be decomposed as $pV^{-1} = (l : s)$. In that case, s is the error syndrome and is therefore known. Decoding then consists in identifying the most likely l given knowledge of s. In the quantum case however, S is only partially determined by the error syndrome $s(P)$. Indeed, we can decompose S as $S = S^x + S^z$ (c.f. Notation 3), and notice that $s(P)$ reveals only S^x. More precisely, we have the relation

$$S^x = \sum_{i=1}^{n-k} (X_i)^{s_i(P)}$$

where $(X)^0 = I$, $(X)^1 = X$, and $s_i(P)$ is the i-th entry of $s(P)$. In other words, S^x has a X at position i if the ith syndrome bit $s_i(P) = 1$, and otherwise it has a I. This is due to two facts:

1) Using $X_i \star Z_j = \delta_{ij}$, we know that the i-th component of S^x is X if $(L : S) \star Z_{i+k} = 1$ and I otherwise,
2) The i-th error syndrome bit is given by (c.f. Definition 14) $s_i(P) = P \star H_i = (PV^{-1}) \star (H_i V^{-1}) = (L : S) \star Z_{i+k}$ where we have used the fact that symplectic transformations preserve the symplectic inner product.

Hence, two physical errors $P = (L : S^x + S^z)V$ and $P' = (L : S^x + S'^z)V = P + (I_k : S^z + S'^z)V$ have the same error syndrome S^x, so cannot be distinguished. However, they also yield the same logical transformation L, so they can be corrected by the same operation (namely applying $L = L^{-1}$ again). Therefore, they *cannot* and *need not* be distinguished by the error syndrome: such errors are called degenerate. This reflects the fact that the $H_i = Z_{i+k}V \in C(I)$ are harmless undetected errors, and motivates the following definition.

Definition 15 (Logical coset): Given an encoding matrix V, the *logical coset* $C(L)$ associated to the logical transformation $L \in G_k$ is defined as

$$\begin{aligned} C(L) &= \{P = (L : S^z)V \mid S^z \in \{I, Z\}^{n-k}\} & (29)\\ &= (L : I_{n-k})V + C(I). & (30)\end{aligned}$$

The union of these cosets is the code itself.

$$C = \bigcup_{L \in G_k} C(L). \tag{31}$$

The fact that quantum codes are coset codes will have important repercussions. In particular, maximum likelihood decoding consists in identifying the *logical coset* $C(L)$ with highest probability given S^x. As in the classical setting (c.f. Eq. (5)) the pullback of the probability distribution on physical errors $\mathbf{P}(P)$ induces a probability distribution on logical transformations and error syndromes

$$\mathbf{P}(L, S^x) = \sum_{S^z} \mathbf{P}(P)\Big|_{P=(L:S^x+S^z)V^{-1}} \tag{32}$$

and maximum likelihood decoding consists in identifying the most likely logical transformation L given the syndrome S^x. More formally:

Definition 16 (Maximum likelihood decoder): The maximum likelihood decoder $L_{ML} : \{I, X\}^{n-k} \to G_k$ is defined by

$$L_{ML}(S^x) = \text{argmax}_L \mathbf{P}(L|S^x) \tag{33}$$

where the conditional probability is defined the usual way

$$\mathbf{P}(L|S^x) = \frac{\mathbf{P}(L, S^x)}{\sum_{L'} \mathbf{P}(L', S^x)}. \tag{34}$$

Definition 17 (Qubit-wise maximum likelihood decoder): The qubit-wise maximum likelihood decoder $L^i_{ML} : \{I, X\}^{n-k} \to G_1$ is defined by

$$L^i_{ML}(S^x) = \text{argmax}_{L^i} \mathbf{P}(L^i|S^x) \tag{35}$$

where the marginal conditional probability is defined the usual way

$$\mathbf{P}(L^i|S^x) = \sum_{L^1,\ldots L^{i-1}, L^{i+1},\ldots L^k} \mathbf{P}(L^1, \ldots L^k|S^x). \tag{36}$$

Equation (32) differs from its classical analogue Eq. (5) by a summation over S^z which reflects the coset structure of the code. Aside from this distinction, the maximum-likelihood decoders are defined as in the classical case.

IV. Quantum turbo-codes

In this section, we describe quantum turbo-codes obtained from interleaved serial concatenation of quantum convolutional codes. This first requires the definition of quantum convolutional codes. We will define them through their circuit representation as in [30] rather than through their parity-check matrix as in [14], [17], [1]: this allows to define in a natural way the state diagram and is also quite helpful for describing the decoding algorithm.

A. Quantum convolutional codes

A quantum convolutional encoder can be defined quite succinctly as a stabilizer code with encoding matrix V given by the circuit diagram of Fig. 6. The circuit is built from repeated uses of the seed transformation U shifted by n qubits. In this circuit, particular attention must be paid to the order of the inputs as they alternate between stabilizer qubits and logical qubits. This is a slight deviation from the convention establish in the previous section, and it is convenient to introduce the following notation to label the different qubits appearing in the encoding matrix of a quantum stabilizer code.

Definition 18: The positions corresponding to L are called the *logical positions* and the positions corresponding to S are called the *syndrome positions*.
The total number of identical repetition of the seed transformation U is called the duration of the code and is denoted N. The m qubits that connect gates from consecutive time slices are called memory qubits. The encoding is initialized by setting the first m memory qubits in the $|0_m\rangle$ state. To terminate the encoding, set the k information qubits of the last t time slices in the $|0_k\rangle$ state, where t is a free parameter independent of N. The rate of the code is thus $kN/(n(N+t)+m)$ which is of the form $k/n+O(1/N)$ for fixed t.

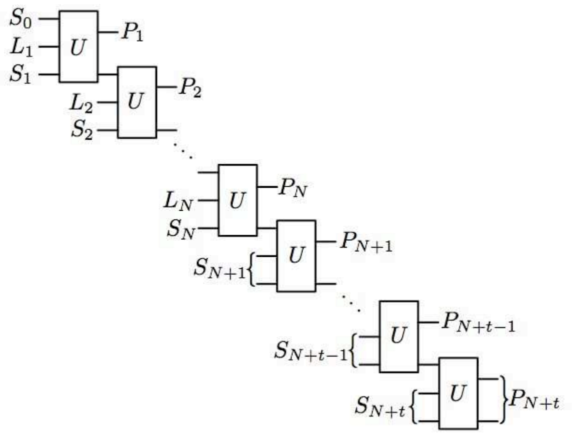

Fig. 6. Circuit diagram of a quantum convolutional encoder with seed transformation U. The superscript indicating the number of qubits per wire are omitted for clarity, and can be found on Fig. 7.

Formally, a quantum convolutional code can be defined as follows.

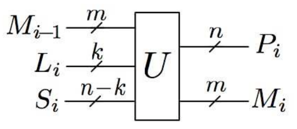

Fig. 7. Seed transformation circuit.

Definition 19 (Quantum convolutional encoder): Let n, k, m, and t be integers defining the parameters of the code, and N the duration of the encoding. Let U be an $(n+m)$-qubit symplectic matrix called the seed transformation. The encoding matrix V of the quantum convolutional encoder is a symplectic matrix over $m+n(N+t)$ qubits given by

$$V = \prod_{i=1}^{N+t} U_{[(i-1)n+1..in+m]}$$

where $[a..b]$ stands for the integer interval $\{a, a+1, \dots, b\}$ and where $U_{[(i-1)n+1..in+m]}$ acts on an element $(P_1, \dots, P_{m+n(N+t)}) \in G_{m+n(N+t)}$ such that its image $(P'_1, \dots, P'_{m+n(N+t)})$ satisfies: $(P'_{(i-1)n+1}, \dots, P'_{in+m}) = (P_{(i-1)n+1}, \dots, P_{in+m})U$ and all other P_i are given by $P'_i = P_i$. The syndrome symbols correspond to the positions belonging to $[1..m] \cup \bigcup_{i\in[1..N]}[(i-1)n+m+k+1..in+m] \cup \bigcup_{i\in[N+1..(N+t)]}[(i-1)n+m..in+m]$.

It will be convenient to decompose an element P in $G_{n(N+t)+m}$ as $P = (P_1 : P_2 : \dots : P_{N+t})$ where the P_i belong to G_n for i in $\{1, 2, \dots, N+t-1\}$ and P_{N+t} belongs to G_{n+m}. This decomposition directly reflects the structure of the output wires appearing on the right-hand-side of the circuit diagram of Fig. 6.

Similarly, we will decompose the Pauli-stream obtain by applying the inverse encoder to P as

$$(S_0 : L_1 : S_1 : \dots : L_N : S_N : S_{N+1} : \dots : S_{N+t}) \triangleq PV^{-1},$$

where S_0 belongs to G_m, the L_i's all belong to G_k, the S_i's belong to G_{n-k} for i in $\{1, \dots, N\}$ and the S_{N+j}'s belong to G_n for j in $\{1, \dots, t\}$. This decomposition directly reflects the structure of the input wires appearing on the left-hand-side of the circuit diagram of Fig. 6.

While the P_j are related to the L_j and S_j via a matrix V of dimension $2(N+t)n+2m$, the convoluted structure of V can be exploited to recursively compute this transformation without the need to manipulate objects of size increasing with N. This requires the introduction of auxiliary memory variables $M_j \in G_m$. The recursion is initialized by setting

$$(M_{N+t-1} : S_{N+t}) \triangleq P_{N+t}U^{-1}. \tag{37}$$

The S_j for $i \in \{N+1, \dots, N+t-1\}$ are obtained by recursion on i:

$$(M_{i-1} : S_i) \triangleq (P_i : M_i)U^{-1} \tag{38}$$

and the M_{i-1}, L_i, S_i for i in $\{1, \dots, N\}$ are obtained from the recursion

$$(M_{i-1} : L_i : S_i) \triangleq (P_i : M_i)U^{-1} \tag{39}$$

Finally, set

$$S_0 = M_0. \tag{40}$$

Any Clifford transformation U on $n+m$ qubits can be used as a seed transformation and defines a convolutional code. It will be useful to decompose U into blocks of various sizes

$$U = \left(\begin{array}{cc} \overbrace{\mu_{\rm P}}^{2n} & \overbrace{\mu_{\rm M}}^{2m} \\ \Lambda_{\rm P} & \Lambda_{\rm M} \\ \Omega_{\rm P} & \Omega_{\rm M} \end{array} \right) \begin{array}{l} \}_{2m} \\ \}_{2k} \\ \}_{2(n-k)} \end{array} . \tag{41}$$

Just like in the classical case, this definition of quantum convolutional code can easily be seen to be equivalent to the ones that have previously appeared in the literature [14], [17], [1]. In particular, the D-transform associated to the code can easily be obtained from the sub-matrices of V appearing in Eq. (41). However, these concepts will not be important for our analysis.

Our definition of convolutional code is stated in terms of their encoding matrix V. From this perspective, convolutional codes are ordinary, albeit very large, stabilizer codes. However, there are important aspects of convolutional codes that distinguish them from generic stabilizer codes.

As mentioned in the previous section, stabilizer codes have in general encoding circuits using a number of elements proportional to the square of the number of physical qubits. Convolutional codes have by definition circuit complexity that scales *linearly* with N for fixed m: each application of the seed transformation U requires a constant number of gates, and this transformation is repeated $N+t$ times.

The most important distinction however has to do with the decoding complexity. The maximum-likelihood decoder of a stabilizer code consists in an optimization over the logical cosets, of which there are 4^K where K denotes the the number of encoded qubits. Without any additional structure on V, maximum-likelihood decoding is an NP-hard problem [5]. Quantum convolutional codes on the other hand have decoding complexity that scales *linearly* with K. The algorithm that accomplishes this task will be described in details in Sec. V.

B. State diagram

We will now define some properties of convolutional codes that will play important roles in the analysis of the performance of turbo codes. Most of these definitions rely on the the state diagram of a convolutional code, which is defined similarly as in the classical case.

Definition 20 (State diagram): The state diagram of an encoder with seed transformation U and parameters (n, k, m) is a directed multigraph with 4^m vertices called *memory-states*, each labeled by a $M \in G_m$. Two vertices M and M' are linked by an edge $M \to M'$ with label (L, P) if and only if there exists $L \in G_k$, $P \in G_n$ and a $S^z \in \{I, Z\}^{n-k}$ such that

$$P : M' = (M : L : S^z)U, . \tag{42}$$

The labels L and P are referred to as the logical label and physical label of the edge respectively.

Thus, the state diagram represents partial information about the transformation $(M : L : S^z) \to (P : M')$ generated by the seed transformation U. Partial information because all information about S^s is discarded. Note that $S^z \in \{I, Z\}^{n-k}$, so the state diagram only contains information about the streams of Pauli operators that remain in the set of codewords C. The restriction on the S^z input can be lifted if we instead consider the *effective seed transformation*

$$U_{\rm eff} \triangleq \left[\begin{array}{cc} \overbrace{\mu_{\rm P}}^{2n} & \overbrace{\mu_{\rm M}}^{2m} \\ \Lambda_{\rm P} & \Lambda_{\rm M} \\ \Sigma_{\rm P} & \Sigma_{\rm M} \end{array} \right] \begin{array}{l} \}_{2m} \\ \}_{2k} \\ \}_{n-k} \end{array} . \tag{43}$$

where the matrix $[\Sigma_{\rm P} : \Sigma_{\rm M}]$ is obtained by removing every second row from the matrix $[\Omega_{\rm P} : \Omega_{\rm M}]$ (i.e. the rows which represent the action on the $\mathfrak{X}_i$). This definition will be convenient for later analysis.

The state diagram of the seed transformation represented at Fig. 8 is shown at Fig. 9. For instance, the self-loop at I labeled (I, II) represents the trivial fact that $(I : I : I)U = (II : I)$. The edge from Y to I labeled (Y, XZ) represents the transformation $(Y : Y : I)U = (XZ : I)$, and so on.

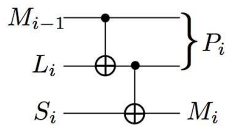

Fig. 8. Seed transformation for an $n = 1$, $k = 1$, and $m = 1$ quantum convolutional code. It corresponds to a unitary transform which maps $|a\rangle \otimes |b\rangle \otimes |c\rangle$ to $|a\rangle \otimes |a + b\rangle|a + b + c\rangle$ for $a, b, c \in \{0, 1\}$. Therefore the seed transformation U acts as follows on the Z_i and X_i: $Z_1U = (Z, I, I)U = (Z, I, I), X_1U = (X, X, X), Z_2U = (Z, Z, I), X_2U = (I, X, X), Z_3U = (I, Z, Z), X_3U = (I, I, X)$.

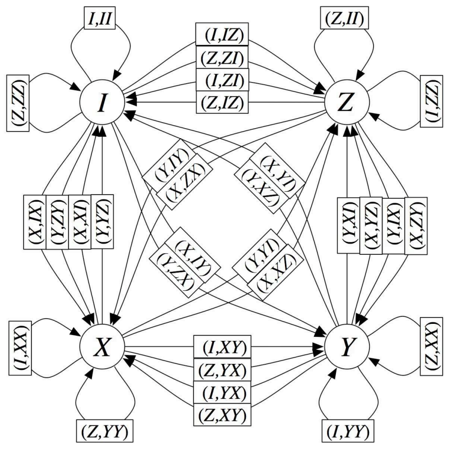

Fig. 9. State diagram for the seed transformation shown at Fig. 8.

The state diagram is crucial for analyzing the properties of the associated code, and also for defining some of its essential features. Here, we give some definitions based on the state diagram that will be important in our analysis.

Definition 21 (Path): A path in the state diagram is a sequence of vertices $M_1, M_2, \ldots$ such that $M_i \to M_{i+1}$ belongs to the state diagram.

Each element of C is naturally associated to a path in the state diagram, which corresponds to the memory states visited upon its encoding. The physical- and logical-weight of a codeword can be obtained by adding the corresponding weights of the edges in the path associated to the codeword. More generally, we will refer to the weight of a path as the sum of the weight of its edges.

Definition 22 (Zero-physical-weight cycle): A zero-physical-weight cycle is a closed path in the state diagram that uses only edges with zero-physical weight.

In the state diagram of Fig. 9 for example, there are two zero-physical-weight cycles corresponding to the transformations $(I:I:I)U = (II:I)$ and $(Z:Z:Z)U = (II:Z)$.

Definition 23 (Non-catastrophic encoder): An encoder C is *non-catastrophic* if and only if the only cycles in its state diagram with physical-weight 0 have logical weight 0.

We see for instance that the state diagram of our running example is catastrophic due to the presence of the self-loop with label (Z, II) at state Z: this cycle has physical weight 0 and logical weight 1. To understand the consequences of a catastrophic seed transformation, consider the act of inverting the encoding transformation of the associated convolutional encoder. This is done by running the circuit of Fig. 6 backwards. Suppose that a single Y error affected the transmitted qubits. More specifically, at time i, $1 \leq i \leq N$, there is a Y on the lower physical wire of the seed transformation of Fig. 8 (i.e. $P_i^2 = Y$) and everything else is I. Since $(IY:I)U^{-1} = (Z:Y:X)$, this will result in a Z in the memory qubit M_{i-1}, a Y in the logical qubit L_i, and a X in the stabilizer qubit S_i. The $S_i = X$ triggers a non-trivial syndrome, which signals the presence of an error. Moreover, because of the self-loop at $M = Z$ that has non-zero logical-weight but zero physical-weight, this error will continue to propagate *without triggering additional syndrome bits*, while creating Z's in L_{i-1} and M_{i-2}, and in L_{i-2} and M_{i-3}, and so on. Thus, an error of finite physical-weight results in an error of unbounded logical-weight, and a finite syndrome. This is the essence of catastrophic error propagation.

Catastrophic encoders may have large minimal distances, but perform poorly under iterative decoding. All the codes we have considered in our numerical simulations were non-catastrophic. In fact, they even satisfied a stronger condition:

Definition 24 (Completely non-catastrophic code): A *completely non-catastrophic code* is such that the only loop in its state diagram with physical-weight zero is the self-loop at I_m.

In the classical setting, non-catastrophicity is insured for instance by the use of systematic encoders. For such encoders, the logical string c is contained as a substring of the encoded string $\overline{c} = cV$. Systematic quantum encoding can be obtained by setting the first k columns of $\Lambda_{\mathrm{P}} = \mathbb{1}_k$ and the first k columns of $\Sigma_{\mathrm{P}} = \mu_{\mathrm{P}} = 0$ (c.f. Eq. (43)). However, this would imply that the stabilizers act trivially on the first k output qubits, resulting in a minimal distance equal to 1. We conclude that it is not possible to design a systematic quantum encoder with minimal distance greater than 1. Thus, non-catastrophicity is a condition that needs to be built in by hand. Fortunately, it can be efficiently verified directly on the state diagram and we have made great use of this fact.

In the classical setting, turbo-codes can be designed with a minimal distance that grows polynomially with N when the inner code is recursive. Recall that recursive means that the encoder has an infinite impulsive response: when a single 1 is inputed at any logical wire of the encoding circuit Fig. 2 and every other input is 0, the resulting output has infinite weight for a code of infinite duration. This definition can be generalized to the quantum setting.

Definition 25 (Quasi-recursive encoder): Consider executing the encoding circuit of Fig. 6 on an input containing a single non-identity Pauli operators on a logical wire, with all other imputs set to I. The corresponding encoder V is *quasi-recursive* when the resulting output has infinite weight when the code has infinite duration N.

However, it can be verified that this notion of recursiveness is too weak to derive a good lower bound on the minimal distance of turbo codes. This departure from the theory of classical codes stems from the fact that quantum codes are coset codes. As in the classical case, the proper definition of a recursive encoder demands that it generates an infinite impulsive response. The novelty comes from the fact that this must be true for every elements in the coset associated to the impulsive logical input: not only must the encoded version of X_i, Y_i, and Z_i have weight growing with the duration of the code N, but so must every elements of $C(X_i)$, $C(Y_i)$, and $C(Z_i)$. Formally, we can define recursive quantum convolutional encoders in two steps:

Definition 26 (Admissible path): A path in the state diagram is *admissible* if and only if its first edge is not part of a zero physical-weight cycle.

Definition 27 (Recursive encoder): A recursive encoder is such that any admissible path with logical-weight 1 starting from a vertex belonging to a zero physical-weight loop does not contain a zero physical-weight loop.

Once again, this property can be directly and efficiently tested given the seed transformation of the convolutional code by constructing its state diagram.

C. Interleaved serial concatenation

Quantum turbo codes are obtained from a particular form of interleaved concatenation of quantum convolutional codes. Interleaving is slightly more complex in the quantum setting since in addition to permuting the qubits it is also possible to perform a Clifford transformation on each qubit which amounts to permute X, Y and Z. More precisely:

Definition 28 (Quantum interleaver): A *quantum interleaver* Π of size N is an N-qubit symplectic transformation composed of a permutation π of the N qubit registers and a tensor product of single-qubit symplectic transformation. It acts as follows by multiplication on the right on G_N :

$$(P_1, \ldots, P_N) \mapsto (P_{\pi(1)}K_1, \ldots, P_{\pi(N)}K_N)$$

where $K_1, \ldots, K_N$ are some fixed symplectic matrices acting on G_1.

It follows that interleavers preserves the weight of N-Pauli streams. An interleaved serial concatenation of two quantum encoders has three basic components:

1) An *outer code* encoding k^{Out} qubits by embedding them in a register of n^{Out} qubits, with encoder V^{Out},
2) An *inner code* encoding k^{In} qubits by embedding them in a register of n^{In} qubits, with encoder V^{Out} and which is such that $k^{\text{In}} = n^{\text{Out}}$,
3) A *quantum interleaver* Π of size $N = n^{\text{Out}} = k^{\text{In}}$.

The resulting *encoding matrix of the interleaved concatenated code* is a symplectic matrix V acting on $G_{n^{\text{In}}}$ such that

$$V = V'^{\text{Out}}\Pi' V^{\text{In}},$$

with the action of V'^{Out} and Π' on $G_{n^{\text{In}}}$ being defined by

$$(L : S^{\text{Out}} : S^{\text{In}})V'^{\text{Out}} = ((L : S^{\text{Out}})V^{\text{Out}} : S^{\text{In}}) \qquad (44)$$

for $(L : S^{\text{Out}} : S^{\text{In}}) \in G_{k^{\text{Out}}} \times G_{n^{\text{Out}}-k^{\text{Out}}} \times G_{n^{\text{In}}-k^{\text{In}}}$, and

$$(L' : S^{\text{In}})\Pi' = (L'\Pi : S^{\text{In}}) \qquad (45)$$

for $L' \in G_{n^{\text{Out}}}$. These relations are summarized at Fig. 10.

The rate of the concatenated code is equal to $\frac{k^{\text{Out}}}{n^{\text{In}}} = \frac{k^{\text{Out}}}{n^{\text{Out}}}\frac{k^{\text{In}}}{n^{\text{In}}}$, that is the product of the rates of the inner code and the outer code.

A serial quantum turbo-code is obtained from this interleaved concatenation scheme by choosing V^{Out} and V^{In} as quantum convolutional encoders.

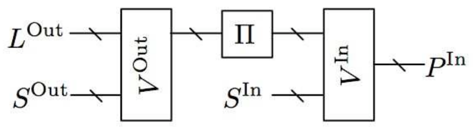

Fig. 10. Circuit diagram for a turbo encoder.

D. Figure of merit

There are a number of in-equivalent ways of characterizing the performance of a code. The minimum distance d of a code is the weight of the lowest weight Pauli error P in $C \setminus C(I)$. Using a minimum distance decoder (which is sub-optimal due to the presence of degenerate errors) allows to correct all errors of weight less or equal to $\frac{d-1}{2}$. Hence, codes with high d are good, although a large d is not necessary to achieve good error suppression. The minimal distance is arguably the most basic figure of merit of a code.

A quantity that is more informative is the weight enumerator, which counts the number of undetected harmful errors of each weight. For a convolutional code however, as the number N_L of encoded qubits tends to infinity, the weight enumerator will be infinite. Indeed, because of the translational invariance of the encoding circuit, finite error patterns come in an infinite number of copies obtained by translation. Instead, we can consider the *distance spectrum* of a non-catastrophic encoder, which is defined as follows.

Definition 29 (Distance spectrum): The *distance spectrum* $(F(w))_{w\geq 0}$ of a non-catastrophic convolutional encoder is a sequence for which $F(w)$ is the number of admissible paths in the state diagram starting and ending in memory states that are part of zero-weight cycles, and with physical weight w and logical weight greater than 0.

An other relevant quantity is the distance spectrum for logical-weight-one elements of C, which is defined similarly.

Definition 30 (Logical-weight-one distance spectrum): The *distance spectrum for logical-weight-one codewords* $F_1(w)$ of a non-catastrophic convolutional encoder is the number of admissible paths in the state diagram starting and ending in memory states that are part of zero-weight cycles, and with physical weight w and logical weight 1.

It can easily be seen that the minimum distance of a turbo-code obtained from the concatenation of two convolutional codes is no greater than $d_*^{\text{Out}} * d_1^{\text{In}}$ where $d_1 = \min_w\{F_1(w) > 0\}$ and the free minimal distance is $d_* = \min_w\{F(w) > 0\}$. The free distance is defined similarly to the classical case by the smallest weight of a harmful undetected error in the convolutional code with infinite time duration. It is so-to-speak a kind of typical minimal distance for convolutional codes, ignoring finite-size effects. To maximize the minimum distance of the turbo-code, we must use outer codes with large free distances d_* and inner encoders with large value of d_1. Recursive encoders for instance have d_1 proportional to N, and therefore serve as ideal inner codes. However, it happens that we cannot use recursive encoders as inner codes as we will see in the next section. Hence, a good rule of thumb is to use inner encoders that minimize the value of $F_1(w)$ at small w, and similarly use an outer code which minimizes the value of $F(w)$ at small w. These will result in a turbo-code with a distance spectrum that is small at low distances.

Finally, given an error model, the *word error rate* (WER) and *qubit error rate* (QER) provide a good operational figure of merit. The QER is the probability that an individual logical qubit is incorrectly decoded. In other words, the QER represents the fraction of logical qubits that have errors after the decoding. The WER is the probability that *at least one* qubit in the block is incorrectly decoded. We expect in general QER $\ll$ WER. Indeed, the WER is thus a much more strenuous figure of merit that the QER. For instance, if N qubits are encoded in N/k block codes for some constant k, then as N increases, the WER approaches 1 exponentially while the QER remains constant. As we will see, turbo-codes have a completely different behavior. In general, we will be interested in the WER averaged over the choice of interleaver π.

E. Recursive convolutional encoders are catastrophic

In the classical setting, non-catastrophic and recursive convolutional encoders are of particular interest. When used as the inner encoder of a concatenated coding scheme, the resulting code has a minimal distance that grows polynomially with their length and offer good iterative decoding performances. More precisely, random serial turbo-codes have a minimum distance which is typically of order $N^{\frac{d_*^{\rm Out}-2}{d_*^{\rm Out}}}$ when the inner encoder is recursive, where N is the length of the concatenated code and $d_*^{\rm Out}$ the free distance of the outer code [21]. That the encoder be non-catastrophic is important to obtain good iterative decoding performances.

This result and its proof would carry over the quantum setting almost verbatim with our definition of recursive encoders. The quantum case is slightly more subtle due to the coset structure of the code. Unfortunately, such encoders do not exist :

Theorem 1: Quantum convolutional recursive encoders are catastrophic.
This result is perhaps surprising since the notions of catastrophic and recursive are quite distinct in the classical setting. Nonetheless, the stringent symplectic constraints imposed to the seed transformation U gives rises to a conflicting relation between them. The proof of Theorem 1 is rather involved. Here we present its main steps and leave the details to the appendix.

The proof involves manipulation of the rows of the effective encoding matrix Eq. (43) and for that reason, it is more appropriate to view effective Pauli operators as elements of $\mathbb{F}_2^{2n}$. The proof proceeds directly by demonstrating that the state diagram of any recursive convolutional encoder contains a directed cycle with zero physical weight and non-zero logical weight. We first need a characterization of the memory states M that can be part of a zero physical weight cycle. We break this into three steps. First, we characterize the set of states that are the endpoint of edges in the state diagram with zero physical weight edges. In other words, we want to find all possible values for the memory element M' in $\mathbb{F}_2^{2m}$ such that there exist $M \in \mathbb{F}_2^{2m}$, $S \in \mathbb{F}_2^{n-k}$, and $L \in \mathbb{F}_2^{2k}$ such that $(M : L : S)U_{\rm eff} = (0_{2n} : M')$.

Lemma 2: Given a seed transformation U, let $\mathcal{S}$ to be the subspace of $\mathbb{F}_2^{2m}$ spanned by the rows of $\Sigma_{\rm M}$. The set of endpoints of edges with zero physical label is equal to $\mathcal{S}^\perp$ and conversely, any M in $\mathcal{S}^\perp$ is the end vertex in the state diagram of exactly one edge of zero physical weight.

If the state diagram contains a zero physical weight cycle, it is therefore necessarily supported on the subset of vertices $\mathcal{S}^\perp$. However, edges of zero physical weight with endpoints vertices in $\mathcal{S}^\perp$ may originate from vertices outside $\mathcal{S}^\perp$. Such edges are not part of zero-physical weight cycles. The next step is thus to characterize the set of endpoints of edges with zero physical label and starting point in $\mathcal{S}^\perp$. Since in the absence of other inputs each time interval modifies the memory state by $M \to M\mu_{\rm M}$, we intuitively expect this set to be $\mathcal{S}_0^\perp$, where $\mathcal{S}_0$ is the smallest subspace containing $\mathcal{S}$ and stable by $\mu_{\rm M}$. This is confirmed by the following lemma.

Lemma 3: Given a seed transformation U, let

$$\mathcal{S}_0 \triangleq \sum_{i=0}^{\infty} \mathcal{S}\mu_{\rm M}^i. \tag{46}$$

For any element M' of $\mathcal{S}_0^\perp$, there exists a unique element M in $\mathcal{S}_0^\perp$, such that there is an edge of physical weight 0 from M to M'.

This lemma narrows down the set of vertices in the state diagram that can support zero physical weight cycles. In particular, we can define a sub-graph of the state diagram obtained from the vertex set $\mathcal{S}_0^\perp$ and directed edges with trivial physical labels. This subgraph is guaranteed to have constant indegree 1 for all its vertices, but some of its vertices may have no outgoing edges. These would definitely not be part of a cycle. To ensure that all vertices in the subgraph have a positive number of outgoing edges, we must once more restrict its set of vertices. The (left) nullspace of $\mu_{\rm M}^i$'s will play a fundamental role.

Notation 6: Let μ be a linear mapping from $\mathbb{F}_2^{2m}$ to itself. We denote by ${\rm Null}(\mu)$ the (left) nullspace of μ, that is

$${\rm Null}(\mu) = \{M \in \mathbb{F}_2^{2m} | M\mu = 0_{2m}\}.$$

Notation 7: Let $\mathcal{N}_0 \triangleq \sum_{i=1}^{\infty} {\rm Null}(\mu_{\rm M}^i)$ and $\mathcal{V}_0 = \mathcal{S}_0 + \mathcal{N}_0$. Let $\mathcal{G}$ be a sub-graph of the state diagram obtained from the vertex set $\mathcal{V}_0^\perp$ and edges with trivial physical label. This graph is called the *kernel graph* of the quantum convolutional code with seed-transformation U.

By replacing the vertex set $\mathcal{S}_0^\perp$ by $\mathcal{V}_0^\perp$, our goal was to eliminate any vertex with no outgoing edge. This turns out to be successful as shown by the following lemma.

Lemma 4: The kernel graph has constant indegree 1 and positive outdegree for any vertex.

Thus, any cycle with zero physical weight must be supported on the kernel graph of the seed transformation. The next step in order to prove Theorem 1 is to demonstrate that when the quantum convolutional encoder is recursive, its corresponding kernel graph $\mathcal{G}$ does not only consist of the single zero vertex with a self-loop attached to it, corresponding to the trivial relation $(0_{2m} : 0_{2k} : 0_{n-k})U_{\rm eff} = (0_{2n} : 0_{2m})$.

Lemma 5: The kernel graph of a recursive quantum convolutional encoder has strictly more than one vertex.
This result is an essential distinction between the quantum and the classical case. In the classical case, when the memory state is non-zero, it is always possible to create a non-zero physical output for instance by copying the state of the memory at the output. But this is not possible quantum-mechanically.

Before proving that $\mathcal{G}$ contains a cycle with non-zero logical weight, we will first prove that it contains at least one edge with non-zero logical weight. For this purpose, let us characterize the subset of edges with zero physical weight and zero logical weight.

Lemma 6: Given a seed transformation U, let $\mathcal{L}$ be the subspace of $\mathbb{F}_2^{2m}$ spanned by the rows of $\Lambda_{\rm M}$ and $\Sigma_{\rm M}$. The set of endpoints of edges of zero physical and logical weight is equal to $\mathcal{L}^\perp$.
This result and its proof are structurally similar to Lemma 2, except that $\mathcal{S}$ has been replaced by $\mathcal{L}$. From this, we conclude:

Lemma 7: The kernel graph of a recursive quantum convolutional encoder contains an edge with non-zero logical weight.

Armed with this result, we are now in a position to prove the main result of this section.

Proof: (of Theorem 1) Consider a recursive quantum convolutional encoder and its associated kernel graph. By Lemma 7, this graph has at least one edge with non-zero logical weight. Let us say that it goes from M_0 to M_1. From Lemma 4, we can follow a directed path of arbitrary length l with (M_0, M_1) as starting edge :

$$M_0 \to M_1 \to \cdots \to M_{t-1} \to M_t.$$

If the length of the path is greater than the number of vertices of the graph it must contain at least twice the same vertex. Moreover, M_0 must be part of this cycle. Otherwise, we would have a path of the form $M_0 \to M_1 \to \dots M_j \to M_{j+1} \to \cdots \to M_l = M_j$ with $j > 0$. In this case, M_j would have indegree 2 which is impossible. In other words, there is a directed cycle in the state diagram with zero physical weight and non-zero logical weight. The corresponding convolutional encoder is therefore catastrophic. ∎

V. Decoding

This section describes the decoding procedure for turbo-codes operated on memoryless Pauli channels. With an n-qubit memoryless Pauli channel, errors are elements of G_n distributed according to a product distribution $\mathbf{P}(P_1 : P_2 : \dots : P_n) = f_1(P_1)f_2(P_2)\dots f_n(P_n)$. The depolarizing channel described in Section III is a particular example of such a channel where all f_j are equal. We note that our algorithm can be extended to non-Pauli errors using the belief propagation algorithm of [24], but leave this generalization for a future paper. The decoding algorithm we present is an adaptation to the quantum setting of the usual "soft-input soft output" algorithm used to decode serial turbo-codes (see [2]). It differs from the classical version in several points.

1) As explained in Subsection III-D, for decoding a quantum code we do not consider the state of the qubits directly (which belong to a continuous space and which cannot be measured without being disturbed) but instead consider the Pauli error (which is discrete) that has affected the quantum state. Decoding consists in inferring the transformation that has affected the state rather than inferring what the state should be.
2) Decoding a quantum code is related to classical "syndrome decoding" (see [26, chapter 47]) with the caveat that errors differing by a combination of the rows of the parity-check matrix act identically on the codewords. Thus, maximum-likelihood decoding consists in identifying the most likely error coset given the syndrome. The coset with largest probability can differ from the one containing the most likely Pauli error.
3) We cannot assume as in the classical case that the soft-input soft-output decoder of the convolutional quantum code starts at the zero-state and ends at the zero-state. This is related to the fact that the memory is described in terms of the Pauli error that has affected the qubits rather than reflecting a property of the encoded state. Instead, we perform a measurement which reveals partial information (the X component) about the first memory element.

Let us now describe how each constituent convolutional code is decoded with a soft-input soft-output decoder.

A. Decoding of convolutional codes

As stated in Definition 17, qubit-wise maximum likelihood consists in finding the logical operator L_i that maximizes the marginal conditional probability $\mathbf{P}(L_i|S^x)$. We call the algorithm that computes this probability – but without returning the L_i that optimizes it – a *soft input soft output (SISO) decoder*. The purpose of this section is to explain how such a decoder can be implemented efficiently for quantum convolutional codes.

We choose to base our presentation solely on the circuit description of the code. Our algorithm is essentially equivalent to a sum-product algorithm operated on the trellis of the code [32]. However, the novelties of quantum codes listed in the previous section requires some crucial modifications of the trellis-based decoding. We find that these complication are greatly alleviated when decoding is formulated directly in terms of the circuit.

Since the distinction between trellis-based and circuit-based decoding are technical rather conceptual, we will present the procedure in details and omit its derivation from first principles. As usual, when operated on a memoryless Pauli channel, the whole procedure is nothing but Bayesian updating of probabilities.

Consider a quantum convolutional code with parameters (n, k, m, t), seed transformation U and duration N as shown at Fig 6. We use the same notation as in Subsection IV-A and denote by V the associated encoding matrix. Let us recall that it maps $G_{n(N+t)+m}$ to itself. As in Subsection IV-A we decompose an element P in $G_{n(N+t)+m}$ (i.e. an error on the channel) as $P = (P_1 : P_2 : \cdots : P_{N+t})$ where the P_i's belong to G_n for i in $\{1, 2, \dots, N + t - 1\}$ and P_{N+t} belongs to G_{n+m}. It will be convenient to denote the coordinates of each P_i by P_i^j, i.e. $P_i = (P_i^1 : P_i^2 : \dots : P_i^n)$ where the P_i^j's belong to G_1.

Similarly, we will decompose the Pauli-stream obtain by applying the inverse encoder to P as

$$(S_0 : L_1 : S_1 : \cdots : L_N : S_N : S_{N+1} : \cdots : S_{N+t}) \triangleq PV^{-1},$$

where S_0 belongs to G_m, the L_i's all belong to G_k, the S_i's belong to G_{n-k} for i in $\{1, \dots, N\}$ and the S_{N+j}'s belong to G_n for j in $\{1, \dots, t\}$.

As explained in Subsection IV-A, the L_i and S_i can be obtained from the P_i via a recursion relation Eqs (37-39) which uses auxiliary memory variables M_i. This recursive procedure can be understood intuitively from the circuit diagram of Fig. 6. It simply consists in propagating the effective Pauli operator P from the right to the left-hand-side of the circuit. This can be done in $N + t$ steps, each step passing through a single seed transformation U, and the memory variables M_j simply represent the operators acting on the memory qubit

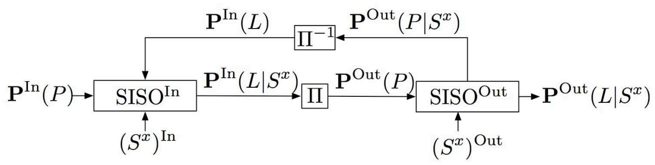
$\mathbf{P}^{\text{In}}(L)$
Π^{-1}
$\mathbf{P}^{\text{Out}}(P|S^x)$
$\mathbf{P}^{\text{In}}(P)$
SISO^{In}
$\mathbf{P}^{\text{In}}(L|S^x)$
Π
$\mathbf{P}^{\text{Out}}(P)$
SISO^{Out}
$\mathbf{P}^{\text{Out}}(L|S^x)$
$(S^x)^{\text{In}}$
$(S^x)^{\text{Out}}$

Fig. 11. Information flow in the iterative turbo decoding procedure.

between two consecutive seed transformations. The decoding algorithm actually follows the same logic. As explained in Sec. III-D, the probability on L and S is obtained from the pullback of $\mathbf{P}(P)$ through the encoder V (c.f. Eq. (32)). For a convolutional code, this pullback can be decomposed into elementary steps, each step passing through a single seed transformation U and computing intermediate probabilities on the memory variables.

In addition to the procedure just outlined, the decoder must also update the probability $\mathbf{P}(L)$ obtained from the pullback of $\mathbf{P}(P)$ conditioned on the value of the observed syndrome. This operation is slightly more subtle, and requires not only the pullback of probabilities through the circuit, but also their pushforward (propagating from the left to the right-hand-side of the circuit). For that reason, the decoding algorithm presented at Algorithm 1 will consist of three steps, a backward pass (Algorithm 2), a forward pass (Algorithm 3), and a local update (Algorithm 4). As indicated by their names, these respectively perform a pullback of probabilities, a pushforward of probabilities, and finally an operation that combines these two probabilities into the final result.

Our description of these algorithms make use of the following notation:

$$S \triangleq (S_i)_{0\le i\le N+t} \tag{47}$$
$$S_{\le i} \triangleq (S_j)_{0\le j\le i} \tag{48}$$
$$S_{>i} \triangleq (S_j)_{i<j\le N+t} \tag{49}$$

and we denote by U_P the binary matrix formed by the $2n$ first columns of U and by U_M the binary matrix formed by the $2m$ last columns of U. This means that

$$\begin{aligned} P_i &= (M_{i-1} : L_i : S_i)U_P \\ M_i &= (M_{i-1} : L_i : S_i)U_M, \end{aligned}$$

where the M_i are defined from Equations (37),(38) and (39). The notation $\mathbf{P}(M_i) \propto \ldots$ means that entries of the vector $((\mathbf{P}(M_i = \mu))_{\mu\in G_m}$ are proportional to the corresponding right-hand side term, the proportionality factor being given by normalization. Finally, for any integer n, we denote $[n] \triangleq \{1, 2, \ldots, n\}$.

B. Turbo decoder

A turbo code is built from the interleaved serial concatenation of two convolutional codes. The decoding of such a code uses the SISO decoder of its constituent convolutional codes in an iterative way that is schematically illustrated at Fig. 11.

The inner code is first decoded as described above but without any information on the logical random variables: $\mathbf{P}^{\text{In}}(L_i^j)$ is the uniform distribution. The distribution $\mathbf{P}^{\text{In}}(P_i^j)$ is given directly by the channel model. The only output which is used in the following step is the output distribution on the logical variables given the syndrome measured on the inner code : $\mathbf{P}^{\text{In}}(L_i^j|S^x)$ (S^x really refers to the part of the syndrome measured for the inner code and not to the whole syndrome, but we do not attach a "In" to it to avoid cumbersome notation).

Then, the outer code is decoded with the SISO algorithm, using as input distribution for the logical variables, as in the previous case, the uniform distribution. The input distribution of the physical variables $\mathbf{P}^{\text{Out}}(P_i^j)$ is deduced from the logical output distribution of the inner decoder :

$$\mathbf{P}^{\text{Out}}(P_{i^\pi}^{j^\pi} = \gamma) = \mathbf{P}^{\text{In}}(L_i^j K_i^j = \gamma|S^x),$$

where i^π and j^π are such that $(i^\pi, j^\pi) = \pi(i,j)$, and the K_i^j are the single-qubit symplectic transformations that appear in the quantum interleaver Π. This yields the output distributions $\mathbf{P}^{\text{Out}}(P_i^j|S^x)$ and $\mathbf{P}(L_i^j|S^x)$ (again, S^x only refers to the part of the syndrome attached to the outer code). This step is terminated by estimating the most likely error coset $\hat{L}$, setting

$$\hat{L}_i^j = \text{argmax}_\gamma \left\{\mathbf{P}^{\text{Out}}(L_i^j = \gamma|S^x)\right\}.$$

To iterate this procedure, use the output probability $\mathbf{P}^{\text{Out}}(P_i^j|S^x)$ as information on the logical variables of the inner code : in other words, set as input distribution for inner SISO decoding

$$\mathbf{P}^{\text{In}}(L_i^j = \gamma) = \mathbf{P}^{\text{Out}}(P_{i^\pi}^{j^\pi} = \gamma K_{j^\pi}^{i^\pi}|S^x),$$

and the distribution of the physical variables are set by the physical channel as before. This is represented by the feedback loop on Fig. 11 where information from the outer decoder is returned to the inner decoder.

This procedure can be repeated an arbitrary number of times, with each iteration yielding an estimate of the maximum-likelihood decoder of the outer code. The iterations can be halted after a fixed number of rounds, or when the estimate does not vary from one iteration to the next. Although the decoding scheme is exact for both constituent codes, the overall turbo-decoding is sub-optimal. The reason for this is that although $\mathbf{P}^{\text{In}}(P)$ is memoryless, the induced channel $\mathbf{P}^{\text{Out}}(P) =$ on the outer code obtained from $\mathbf{P}^{\text{Out}}(P_{i^\pi}^{j^\pi}) = \mathbf{P}^{\text{In}}(L_i^j K_i^j|S^x)$ is not. The decoder ignores this fact and only uses the marginals $\mathbf{P}^{\text{Out}}(P_i^j)$ of $\mathbf{P}^{\text{Out}}(P)$. This is the price to pay for an efficient decoding algorithm.

VI. Results

The convolutional codes we used for our construction of turbo-codes are for the most part generated at random. That is, we first generate a random seed transformation U of desired dimensions. Using its state diagram, we then test whether the corresponding encoder is catastrophic, and if so we reject it and start over. Non-catastrophicity is the only criterion that we systematically imposed.

```
Algorithm 1: The SISO algorithm for quantum convolutional codes

  INPUTS:
    $\mathbf{P}(P_i^j)$ for $i \in [N+t]$, $j \in [n]$, (and $j \in [n+m]$ when $i = N+t$)    From physical noise model
    $\mathbf{P}(L_i^j)$ for $i \in [N]$, $j \in [k]$                                            From turbo decoder
    $S^x$                                                                                       From syndrome measurement
  OUTPUTS:
    $\mathbf{P}(P_i^j|S^x)$ for $i \in [N+t]$, $j \in [n]$ (and $j \in [n+m]$ when $i = N+t$),
    $\mathbf{P}(L_i^j|S^x)$ for $i \in [N]$, $j \in [k]$
  ALGORITHM :
    backward pass
    forward pass
    local update
```

```
Algorithm 2: Backward pass

  INPUTS:
    Same as SISO algorithm
  OUTPUTS:
    $\mathbf{P}(M_i|S^x_{>i})$ for $i \in [N+t]$.
  ALGORITHM :
    {Initialization: $\mathbf{P}(M_{n+t})$ is given directly by the physical noise model.}
    for all $\gamma \in G_m$ do
      $\mathbf{P}(M_{n+t} = \gamma) \leftarrow \Pi_{j=1}^m \mathbf{P}(P_{N+t}^{n+j} = \gamma^j)$
    end for
    {Recursion: first $t$ steps}
    for $i = N+t-1$ to $N+1$ do
      $\mathbf{P}(M_i|S^x_{>i}) \propto \sum_{\sigma \in G_n : \sigma^x = S_i^x} \Big[\mathbf{P}(P_{i+1} = (M_i : \sigma)U_P)\mathbf{P}(M_{i+1} = (M_i : \sigma)U_M|S^x_{>i+1})\Big]$
    end for
    {Recursion: last $N$ steps}
    for $i = N$ to $1$ do
      $\mathbf{P}(M_i|S^x_{>i}) \propto \sum_{\substack{\lambda \in G_k \\ \sigma \in G_{n-k} : \sigma^x = S_i^x}} \Big[\mathbf{P}(L_i = \lambda)\mathbf{P}(P_{i+1} = (M_i : \lambda : \sigma)U_P)\mathbf{P}(M_{i+1} = (M_i : \lambda : \sigma)U_M|S^x_{>i+1})\Big]$
    end for
```

As a first sieve among the randomly generated non-catastrophic seed transformations, we can study their distance spectrums and make some heuristic test based on it. Example of good seed transformations obtained from this procedure are

$$U_{(3,1,3)} = \begin{bmatrix} 2085 \\ 926 \\ 2053 \\ 1434 \\ 910 \\ 3943 \\ 1484 \\ 2881 \\ 3212 \\ 2250 \\ 68 \\ 331 \end{bmatrix} \quad U_{(3,1,4)} = \begin{bmatrix} 13159 \\ 10335 \\ 13127 \\ 6554 \\ 10319 \\ 14441 \\ 10625 \\ 5835 \\ 832 \\ 13893 \\ 11916 \\ 11329 \\ 8204 \\ 5570 \end{bmatrix} \quad U_{(2,1,4)} = \begin{bmatrix} 610 \\ 3323 \\ 760 \\ 1591 \\ 2500 \\ 942 \\ 2290 \\ 794 \\ 1535 \\ 2202 \\ 2859 \\ 809 \end{bmatrix}$$

Algorithm 3: Forward pass

```
INPUTS:
  Same as SISO algorithm
OUTPUTS:
  $\mathbf{P}(M_i|S^x_{\leq i})$ for $i \in \{0,\ldots,N+t-1\}$.
ALGORITHM :
  {Initialization: }
  for all $\gamma \in G_m$ do
    if $\gamma^x = S^x_0$ then
      $\mathbf{P}(M_0 = \gamma|S^x_0) \leftarrow \frac{1}{2^m}$
    else
      $\mathbf{P}(M_0 = \gamma|S^x_0) \leftarrow 0$
    end if
  end for    {Recursion: }
  for $i = 1$ to $N+t+1$ do
    $\mathbf{P}(M_i|S^x_{\leq i}) \propto \sum_{\substack{\mu\in G_m,\lambda\in G_k\\ \sigma\in G_{n-k}:\sigma^x=S^x_i\\ M_i=(\mu:\lambda:\sigma)U_M}} \Big[\mathbf{P}(L_i=\lambda)\mathbf{P}\left(P_i=(\mu:\lambda:\sigma)U_P\right)\mathbf{P}\left(M_{i-1}=\mu|S^x_{\leq i-1}\right)\Big]$
  end for
```

Algorithm 4: Local update

```
INPUTS:
  Same as SISO algorithm
  $\mathbf{P}(M_i|S^x_{>i})$ for $i \in [N+t]$                      From backward pass
  $\mathbf{P}(M_i|S^x_{\leq i})$ for $i \in \{0,\ldots,N+t-1\}$    From forward pass
OUTPUTS:
  $\mathbf{P}(P^j_i|S^x)$ for $i \in [N+t]$, $j \in [n]$ (and $j \in [n+m]$ for $i = N+t$)
  $\mathbf{P}(L^j_i|S^x)$ for $i \in [N]$ and $j \in [k]$
ALGORITHM :
  for $i = 1$ to $N+t$ do
    $\mathbf{P}(L_i|S^x) \propto \sum_{\substack{\mu\in G_m\\ \sigma\in G_{n-k}:\sigma^x=S^x_i}} \Big[\mathbf{P}(L_i)\mathbf{P}\left(M_{i-1}=\mu|S^x_{\leq i-1}\right)\mathbf{P}\left(P_i=(\mu:L_i:\sigma)U_P\right)\mathbf{P}\left(M_i=(\mu:L_i:\sigma)U_m|S^x_{>i}\right)\Big]$
    $\mathbf{P}(P_i|S^x) \propto \sum_{\substack{\mu\in G_m,\lambda\in G_k\\ \sigma\in G_{n-k}:\sigma^x=S^x_i\\ P_i=(\mu:\lambda:\sigma)U_P}} \Big[\mathbf{P}(P_i)\mathbf{P}(L_i=\lambda)\mathbf{P}(M_{i-1}=\mu|S^x_{\leq i-1})\mathbf{P}\left(M_i=(\mu:\lambda:\sigma)U_M|S^x_{>i}\right)\Big]$
  end for
  {Marginalization: }
  Compute $\mathbf{P}(L^j_i|S^x)$ from $\mathbf{P}(L_i|S^x)$
  Compute $\mathbf{P}(P^j_i|S^x)$ from $\mathbf{P}(P_i|S^x)$
```

where each row is specified by the integer corresponding to the binary entry. The subscript on the encoders specify its parameters (n, k, m). Hence, the first two codes have rate $\frac{1}{3}$ but differ by the size of their memory. The third code has a higher rate of $\frac{1}{2}$. The first few values of the distance spectrum of logical-weight-one codewords for these quantum convolutional code are given at Table I, while the distance spectrum of all codewords are listed at Table II.

Based on those values, we conclude that the turbo-codes obtained from concatenation of code using seed transformation $U_{(3,1,4)}$ with itself has a minimal distance no greater that $6 \times 4 = 24$. Similarly, the codes obtained by the concatenation of $U_{(3,1,4)}$ with itself has minimal distance no greater than $7 \times 6 = 42$, and the one obtained from the concatenation of $U_{(2,1,4)}$ with itself has $8 \times 6 = 48$.

These are upper bounds on the minimal distance and do not translate directly into the performance of the code. On the one hand, the actual minimal distance of a turbo-code depends

w	$U_{(3,1,3)}$	$U_{(3,1,4)}$	$U_{(2,1,4)}$
0	0	0	0
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	2	0	0
7	4	3	0
8	8	0	2
9	16	7	0
10	35	0	3
11	70	34	2
12	143	0	0
13	295	156	2
14	634	0	10
15	1 362	586	12
16	2 802	0	37
17	5 714	2 827	38
18	11 526	0	121
19	23 674	11 430	86
20	48 817	0	280

TABLE I

DISTANCE SPECTRUM $F_1(w)$ OF LOGICAL-WEIGHT-ONE CODEWORDS

w	$U_{(3,1,4)}$	$U_{(3,1,4)}$	$U_{(2,1,4)}$
0	0	0	0
1	0	0	0
2	0	0	0
3	0	0	0
4	1	0	0
5	11	0	6
6	47	11	82
7	265	70	442
8	1 275	324	3 379
9	6 397	1 596	24 074
10	31 785	7 773	174 997
11	160 311	40 971	1 253 748
12	801 232	206 959	9 033 087

TABLE II

DISTANCE SPECTRUM $F(w)$

on the interleaver, which we chose completely at random. In all cases, there are most likely lower weight codewords than the estimate provided by these lower bounds, but those are atypical. On the other hand, the codes are not decoded with a minimum distance decoder, so even a true large minimal distance does not imply low WER.

The WER of a quantum turbo-code on a depolarization channel can be estimated using Monte Carlo methods. An error $P \in G_N$ is generated randomly according to the channel model probability distribution. The syndrome associated to this error are evaluated, and based on their value, the decoding algorithm (see Sec. V) is executed. The decoding algorithm outputs an error estimate P'. If $P - P' \in C(I)$, the decoding is accepted, otherwise it is rejected. In other words, the decoding is accepted only if all N_L encoded qubits are correctly recovered. The WER is then the fraction of rejected decodings.

The WERs as a function of the depolarizing probability p are shown for a selection of codes on Fig. 12-14. Perhaps the most striking features of those curves is the existence of a pseudo-threshold value of p below which the WER *decreases* as the number of encoded qubits is increases. Since the codes have a bounded minimal distance, this is not a true threshold in the sense that as we keep increasing the number of encoded qubits, the WER should start to increase. However, we see that for modest sizes N_L of up to 4000, this effect is not observed. We do see however that the improvement appears to be saturating around these values. The pseudo-threshold is particularly clear for the seed transformation $U_{(3,1,3)}$, where it is approximately 0.098, and for the seed transformation $U_{(2,1,4)}$ where it is approximately 0.067. Its value for the seed transformation $U_{(3,1,4)}$ is not as clear, but seams to be between 0.95 and 0.11.

These values should be compared with the hashing bound, whose value is approximately 0.16024 for a rate $\frac{1}{9}$ code and 0.12689 for rate $\frac{1}{4}$. We can also compare with the results obtained from LDPC codes in [27, Figure 10] by evaluating the depolarizing probability p at which the WER drops below 10^{-4}. For a rate $\frac{1}{4}$, this threshold was achieved at $p_{th} \approx 0.033$ (note the convention $f_m = \frac{2}{3}p$) for LDPC codes while the turbo-code shown at Fig. 14 has $p_{th} \approx 0.048$. It should also be noted that this improved threshold is achieved with a smaller block size than that used for the LDPC in [27]; a larger block should further improve this result.

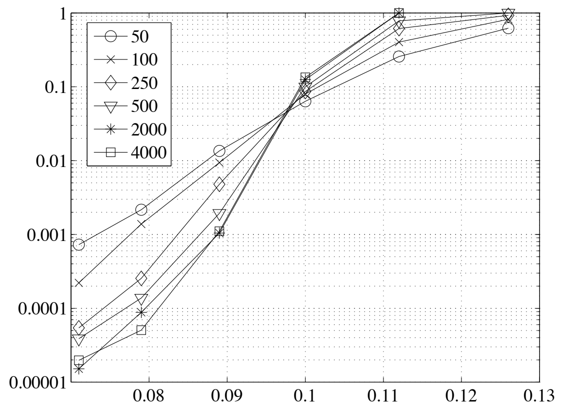

Fig. 12. WER *vs* depolarizing probability p for the quantum turbo-code obtained from the concatenation of the convolutional code with seed transformation $U_{(3,1,3)}$ with itself, for different number of encoded qubits N_L. Each constituent convolutional code has $m = 3$ qubits of memory and have rate $\frac{1}{3}$, so the rate of the turbo code is $\frac{1}{9}$.

As expected, changing the rate of the code directly affects the value of the pseudo threshold. This is seen by comparing either of Figs. 12 or 13 to Fig. 14. The effect of the memory size is however less obvious. Comparing Fig. 12 and 13, it appears that the effect of a larger memory is to sharpen the slope of the WER profile below the pseudo threshold for fixed N_L. In other words, the main impact of the memory size is not in the value of the pseudo threshold, but rather in the effectiveness of the error suppression below that threshold. This conclusion is somewhat supported by Fig. 15 where the WER is plotted for a variety of memory configurations. In all

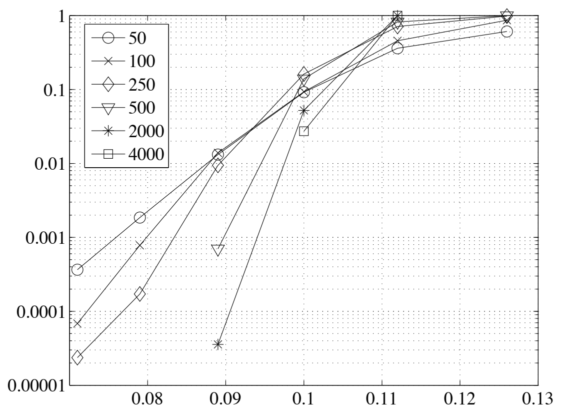

Fig. 13. WER *vs* depolarizing probability p for the quantum turbo-code obtained from the concatenation of the convolutional code with seed transformation $U_{(3,1,4)}$ with itself, for different number of encoded qubits N_L. Each constituent convolutional code has $m = 4$ qubits of memory and have rate $\frac{1}{3}$, so the rate of the turbo code is $\frac{1}{9}$.

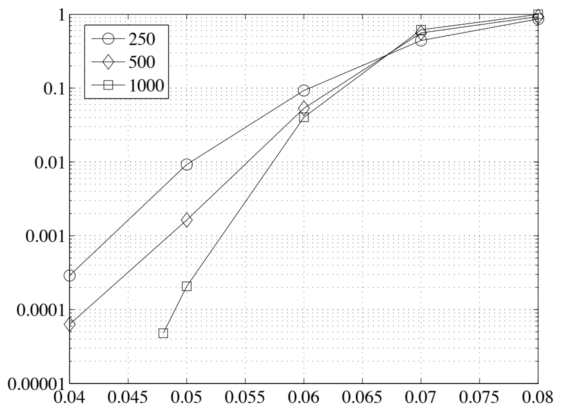

Fig. 14. WER *vs* depolarizing probability p for the quantum turbo-code obtained from the concatenation of the convolutional code with seed transformation $U_{(2,1,4)}$ with itself, for different number of encoded qubits N_L. Each constituent convolutional code has $m = 4$ qubits of memory and have rate $\frac{1}{2}$, so the rate of the turbo code is $\frac{1}{4}$.

cases, the slope of the WER increases with the memory size.

VII. Conclusion

In this article, we have presented a detailed theory of quantum serial turbo-codes based on the interleaved serial concatenation of quantum convolutional codes. The description and analysis of these codes was greatly simplified by the use of a circuit representation of the encoder. In particular, this representation provides a simple definition of the state diagram associated to a quantum convolutional code, and enables a simple and intuitive derivation of their efficient decoding algorithm.

By a detailed analysis of the state diagram, we have shown that all recursive convolutional encoders have catastrophic error propagation. Recursive convolutional encoders can be

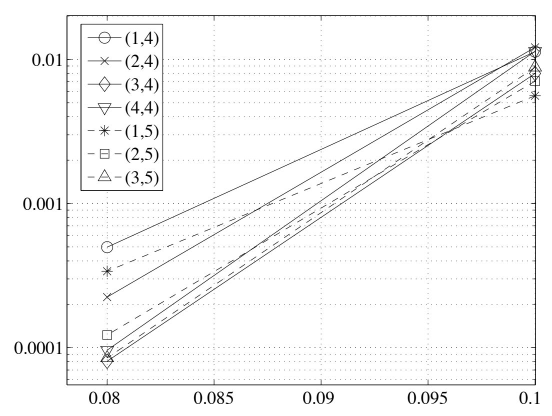

Fig. 15. WER *vs* depolarizing probability p for a quantum turbo code encoding $N_L = 100$ qubits and rate $\frac{1}{9}$ with different memory configurations $(m^{\text{In}}, m^{\text{Out}})$.

constructed and yield serial turbo-codes with polynomial minimal distances. However, they offer extremely poor iterative decoding performances due to their unavoidable catastrophic error propagation. The encoders we have used in our constructions are thus chosen to be non-catastrophic and non-recursive. While the resulting codes have bounded minimal distance, we have found that they offer good iterative decoding performances over a range of block sizes and word error rates that are of practical interest.

Compared to quantum LDPC codes, quantum turbo-codes offer several advantages. On the one hand, there is complete freedom in the code design in terms of length, rate, memory size, and interleaver choice. The freedom in the interleaver is crucial since it is the source of the randomness that is responsible for the success of these codes. On the other hand, the graphical representation of turbo-codes is free of 4-cycles that deteriorate the performances of iterative decoding. Finally, the iterative decoder makes explicit use of the code's degeneracy. This feature is important because turbo-codes, like LDPC codes, have low-weight stabilizers and are hence greatly degenerate.

In future work, we hope to surmount the obstacle of catastrophic error propagation. A concrete avenue is the generalized stabilizer formalism of operator quantum error correction [33], which could circumvent the conclusions of our theorem established in the context of subspace stabilizer codes. Doping [44] is an other possibility that we will investigate.

Acknowledgments — DP is supported in part by the Gordon and Betty Moore Foundation through Caltech's Center for the Physics of Information, by the National Science Foundation under Grant No. PHY-0456720, and by the Natural Sciences and Engineering Research Council of Canada.

Appendix

To prove Lemma 2, we first establish some simple facts:

Fact 5: The subspace of $\mathbb{F}_2^{2n+2m}$ orthogonal to all the rows of U_{eff} is the space spanned by the rows of its submatrix

$[\Sigma_{\rm P} : \Sigma_{\rm M}]$. Similarly, the subspace of $\mathbb{F}_2^{2n+2m}$ orthogonal to all the rows of $\begin{bmatrix} \mu_{\rm P} & \mu_{\rm M} \\ \Sigma_{\rm P} & \Sigma_{\rm M} \end{bmatrix}$ is the space spanned by the rows of $\begin{bmatrix} \Lambda_{\rm P} & \Lambda_{\rm M} \\ \Sigma_{\rm P} & \Sigma_{\rm M} \end{bmatrix}$.

Proof: The subspace V of $\mathbb{F}_2^{2n+2m}$ orthogonal to all the rows of $U_{\rm eff}$ is of dimension $2n+2m-(2m+2k+n-k) = n-k$. We observe now that the rows of $[\Sigma_{\rm P} : \Sigma_{\rm M}]$ are all independent and all orthogonal to the rows of $U_{\rm eff}$. They form therefore a basis of V. This finishes the proof of the first statement. The second one is obtained by similar arguments. ■

Proof: (of Lemma 2) Let $M' \in \mathbb{F}_2^{2m}$ be such that there exist $M \in \mathbb{F}_2^{2m}$, $S \in \mathbb{F}_2^{n-k}$, and $L \in \mathbb{F}_2^{2k}$ such that $(M : L : S)U_{\rm eff} = (0_{2n} : M')$. Notice now that $(0_{2n} : M')$ is spanned by the rows of $U_{\rm eff}$ and is therefore orthogonal to all the rows of the matrix $[\Sigma_{\rm P} : \Sigma_{\rm M}]$. This implies that M' belongs to $\mathcal{S}^{\perp}$. Conversely, any row vector of the form $(0_{2n} : M')$ with M' belonging to $\mathcal{S}^{\perp}$ is orthogonal to all the rows of $[\Sigma_{\rm P} : \Sigma_{\rm M}]$ and is therefore spanned by the rows of $U_{\rm eff}$. This implies that there exist $M \in \mathbb{F}_2^{2m}$, $S \in \mathbb{F}_2^{n-k}$, and $L \in \mathbb{F}_2^{2k}$ such that $(M : L : S)U_{\rm eff} = (0_{2n} : M')$. Furthermore, it can be noticed from the fact that the rows of $U_{\rm eff}$ are independent, that if such an $(M : L : S)$ exists, it is unique. ■

The proof of Lemma 3 requires a straightforward Fact and a Lemma.

Fact 6: For any $M, M' \in \mathbb{F}_2^{2m}$ we have

$$(M\mu_{\rm P} : M\mu_{\rm M}) \star (M'\mu_{\rm P} : M'\mu_{\rm M}) = M \star M'$$

Proof: This is straightforward consequence of the orthogonality relations satisfied by the first $2m$ rows of U. ■

Lemma 8: Let $T \in \mathbb{F}_2^{2m}$ and let M' be such that there exist $M \in \mathbb{F}_2^{2m}$, $S \in \mathbb{F}_2^{n-k}$, and $L \in \mathbb{F}_2^{2k}$ such that $(M : L : S)U_{\rm eff} = (0_{2n} : M')$. We have

$$M' \star T\mu_{\rm M} = M \star T. \tag{50}$$

Proof: We observe that

$$\begin{aligned} M' \star T\mu_{\rm M} &= (0_{2n} : M') \star (T\mu_{\rm P} : T\mu_{\rm M}) \\ &= (M\mu_{\rm P} + L\Lambda_{\rm P} + S\Sigma_{\rm P} : \\ &\quad M\mu_{\rm M} + L\Lambda_{\rm M} + S\Sigma_{\rm M}) \star (T\mu_{\rm P} : T\mu_{\rm M}) \\ &= (M\mu_{\rm P} : M\mu_{\rm M}) \star (T\mu_{\rm P} : T\mu_{\rm M}) \end{aligned}$$

where the last equation follows from the fact that any row of $[\Lambda_{\rm P} : \Lambda_{\rm M}]$ or $[\Sigma_{\rm P} : \Sigma_{\rm M}]$ is orthogonal to all the rows of $[\mu_{\rm P} : \mu_{\rm M}]$. From this, we conclude that

$$M' \star T\mu_{\rm M} = M \star T.$$

■

Proof: (of Lemma 3) Since $M' \in \mathcal{S}^{\perp}$, there exist by Lemma 2, $M \in \mathbb{F}_2^{2m}$, $S \in \mathbb{F}_2^{n-k}$, and $L \in \mathbb{F}_2^{2k}$ such that $(M : L : S)U_{\rm eff} = (0_{2n} : M')$. Let $T \in \mathcal{S}_0$. Using Lemma 8 we obtain

$$M' \star T\mu_{\rm M} = M \star T$$

Notice now that $M' \star T\mu_{\rm M} = 0$ since $T\mu_{\rm M} \in \mathcal{S}_0$. From this $M \star T = 0$. This shows that M belongs to $\mathcal{S}_0^{\perp}$ too. The unicity of M is a consequence of Lemma 2. ■

The following lemma is used in the proof of Lemma 4.

Lemma 9: Let μ be a linear mapping from $\mathbb{F}_2^{2m}$ to itself. Let V be a subspace of $\mathbb{F}_2^{2m}$ such that $\mu(V) \subset V$ and which contains the null space of any positive power of μ. Then for any M in $V^{\perp}$ there exists M' in $V^{\perp}$ such that for any T in $\mathbb{F}_2^{2m}$:

$$M \star T = M' \star T\mu.$$

Proof: We are first going to prove this statement in the case

$$V = \bigcup_{t=1}^{\infty} \text{Null}(\mu^t).$$

This is a subspace of $\mathbb{F}_2^{2m}$ since the Null(μ^t)'s are nested sets :

$$\text{Null}(\mu) \subset \text{Null}(\mu^2) \subset \cdots \subset \text{Null}(\mu^t) \subset \ldots.$$

Let us consider the space Im(μ^t) generated by the rows of μ^t. Since $\mathbb{F}_2^{2m} \supset \text{Im}(\mu) \supset \text{Im}(\mu^2) \supset \ldots$ there must exist a positive t such that $\text{Im}(\mu^t)) = \text{Im}(\mu^{t+1})$. In this case, $\mu(\text{Im}(\mu^t)) = \text{Im}(\mu^t)$. This implies that the restriction of μ to Im(μ^t) is a one-to-one mapping and that $\text{Null}(\mu^t) \cap \text{Im}(\mu^t) = \{0_{2m}\}$. Since $\dim(\text{Null}(\mu^t)) + \dim(\text{Im}(\mu^t)) = 2m$, we can form a basis $(T_1, \ldots T_l, T_{l+1}, \ldots, T_{2m})$ of $\mathbb{F}_2^{2m}$ such that $(T_1, \ldots, T_l)$ spans Null(μ^t) and $(T_{l+1}, \ldots, T_{2m})$ spans Im(μ^t). Moreover, all the Null(μ^v)'s are equal for v greater than or equal to t. This follows directly from the fact that the Im(μ^v)'s are all equal in this case. This can be checked by using the relations $\dim(\text{Null}(\mu^v)) + \dim(\text{Im}(\mu^v)) = 2m$. From these equalities, we deduce that $\dim(\text{Null}(\mu^t)) = \dim(\text{Null}(\mu^{t+1})) = \cdots = \dim(\text{Null}(\mu^v)) = \ldots$. The Null$(\mu^v)$'s are nested sets and therefore $\text{Null}(\mu^t) = \text{Null}(\mu^{t+1}) = \cdots = \text{Null}(\mu^v) = \ldots$. This implies that $V = \text{Null}(\mu^t)$. We define U_i for i in $\{l+1, \ldots, 2m\}$ as the unique element in $\mathbb{F}_2^{2m}$ such that $U_i\mu = T_i$. There exists a unique M' such that

$$\begin{aligned} M' \star T_i &= 0 \text{ for } i \in \{1, \ldots, l\} & (51) \\ M' \star T_i &= M \star U_i \text{ for } i \in \{l+1, \ldots, 2m\} & (52) \end{aligned}$$

This M' belongs to $V^{\perp}$ by Equation (51). Note now that we have defined M' in such a way that $M \star T$ coincides with $M' \star T\mu$ over the basis $(T_1, \ldots, T_l, U_{l+1}, \ldots, U_{2m})$. Therefore, by linearity of the $\star$ product, we have $M \star T = M' \star T\mu$ for all T in $\mathbb{F}_2^{2m}$.

The general case is direct consequence of this particular case. We define M' similarly by Equations (51) and (52) and it is readily checked that M' belongs to $V^{\perp}$. ■

Proof: (of Lemma 4) We know from Lemma 3 that for any element M' in $\mathcal{S}_0^{\perp}$, there exists a unique M in $\mathcal{S}_0^{\perp}$ such that there is an edge of zero physical weight in the state diagram which goes from M to M'. To prove that the kernel graph has constant indegree 1 we just have to prove that when M' belongs to the subset $\mathcal{V}_0^{\perp}$ of $\mathcal{S}_0^{\perp}$ the corresponding M also belongs to this subset. Since for any $T \in \mathcal{N}_0$ we have $M' \star T = 0$ and since $\mathcal{N}_0$ is stable by applying $\mu_{\rm M}$ to the left we obtain for a such a T, $M \star T = M' \star T\mu_{\mathrm{M}} = 0$. This shows that M also belongs to $\mathcal{N}_0^{\perp}$ which shows that M belongs to $\mathcal{V}_0^{\perp}$.

On the other hand, by applying Lemma 9 with $V = \mathcal{V}_0$, we know that for any vertex M of the kernel graph, there is an M' belonging also to $\mathcal{V}_0^{\perp}$ such that for any T in $\mathbb{F}_2^{2m}$:

$$M \star T = M' \star T\mu_{\mathrm{M}}.$$

Note that given such an M' there is a unique M which satisfies the aforementioned equality for all T. Therefore M is necessarily the starting vertex of the unique directed edge of physical weight 0 having as endpoint M'. ■

Proof: (of Lemma 5) We just have to prove that the set $\mathcal{V}_0$ is not equal to the whole space $\mathbb{F}_2^{2m}$. We proceed by contradiction. Assume that $\mathcal{V}_0 = \mathbb{F}_2^{2m}$. Notice now that there exists a finite number t such that

$$\mathcal{V}_0 = \mathrm{Null}(\mu_{\mathrm{M}}^t) + \sum_{i=0}^{t} \mathcal{S}\mu_{\mathrm{M}}^i.$$

For such a t, any M in $\mathbb{F}_2^{2m}$ can be expressed as a sum $M = N + \sum_{i=0}^{t} T_i\mu_{\mathrm{M}}^i$, where N is in $\mathrm{Null}(\mu_{\mathrm{M}}^t)$ and the T_i's all belong to $\mathcal{S}$, i.e. they are of the form $T_i = S_i\Sigma_{\mathrm{M}}$ for some $S_i \in \mathbb{F}_2^{n-k}$. Consider now a finite path starting at the origin with logical weight 1 and non-zero physical weight. We denote by M its endpoint (which is viewed as an element in $\mathbb{F}_2^{2m}$). We decompose $M\mu_{\mathrm{M}}^{t+1}$ as explained before

$$M\mu_{\mathrm{M}}^{t+1} = N + \sum_{i=0}^{t} S_{t-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i$$

where the S_i's belong to $\mathbb{F}_2^k$. The path of length t which starts at M and which corresponds to the sequence of pairs of logical transformations/stabilizer transformations $(0_{2k} : S_0) \to (0_{2k} : S_1) \to \cdots \to (0_{2k} : S_t)$ will go from point M to

$$M\mu_{\mathrm{M}}^{t+1} + \sum_{i=0}^{t} S_{t-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i = N.$$

By extending this path by feeding in t zero transformations $(0_{2k} : 0_{n-k})$ we go from vertex N to $\mu^t(N)$ which is equal to 0_{2m} by definition. This path may then continue by feeding in additional zero transformations and will stay at the zero vertex forever. This contradicts the fact that the quantum code is recursive. ■

Proof: (of Lemma 6) Let M' be an element of $\mathbb{F}_2^{2m}$ for which there exist $M \in \mathbb{F}_2^{2m}$ and $S \in \mathbb{F}_2^{n-k}$, such that $(M : 0^{2k} : S)U_{\mathrm{eff}} = (0_{2n} : M')$. $(0_{2n} : M')$ is spanned by the rows of $[\mu_{\mathrm{P}} : \mu_{\mathrm{M}}]$ and $[\Sigma_{\mathrm{P}} : \Sigma_{\mathrm{M}}]$. By Fact 6, this implies that $(0_{2n} : M')$ is orthogonal to all the rows of the matrices $[\Lambda_{\mathrm{P}} : \Lambda_{\mathrm{M}}]$ and $[\Sigma_{\mathrm{P}} : \Sigma_{\mathrm{M}}]$. Hence M' should belong to $\mathcal{L}^{\perp}$. On the other hand, any $(0_{2n} : M')$ for which M' belongs to $\mathcal{L}^{\perp}$ is orthogonal to all the rows of $[\Lambda_{\mathrm{P}} : \Lambda_{\mathrm{M}}]$ and $[\Sigma_{\mathrm{P}} : \Sigma_{\mathrm{M}}]$ and is therefore spanned by the rows of $[\mu_{\mathrm{P}} : \mu_{\mathrm{M}}]$ and $[\Sigma_{\mathrm{P}} : \Sigma_{\mathrm{M}}]$. ■

Proof: (of Lemma 7) This amounts to prove that there exists a vertex in the kernel graph which does not belong to $\mathcal{L}^{\perp}$. The set of vertices of the kernel graph is $\mathcal{V}_0^{\perp}$. Therefore, we need to find an element of $\mathcal{L}$ that is not in $\mathcal{V}_0$. In particular, we would be done if there existed a row of Λ_{M} which does not belong to $\mathcal{V}_0$.

Assume the opposite. Let t be the integer such that $\mathcal{V}_0 = \mathrm{Null}(\mu_{\mathrm{M}}^t) + \sum_{i=0}^{t} \mathcal{S}\mu_{\mathrm{M}}^i$. Then, for every L in $\mathbb{F}_2^{2m}$ of weight 1 and any integer k, there exists $S_0, S_1, \ldots, S_t$ in $\mathbb{F}_2^{n-k}$ and a N in $\mathrm{Null}(\mu_{\mathrm{M}}^t)$ such that

$$L\Lambda_{\mathrm{M}}\mu_{\mathrm{M}}^k = N + \sum_{i=0}^{t} S_{t-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i.$$

Consider a finite path of non-zero physical weight and logical weight 1 starting at the origin and ending at a vertex M. Assume that this path corresponds to the sequence of pairs of logical/stabilizer inputs

$$\begin{aligned}&(0_{2k} : S_0) \to (0_{2k} : S_1) \to \cdots \to \\ &(0_{2k} : S_{i-1}) \to (L : S_i) \to (0_{2k} : S_{i+1}) \to \cdots \to (0_{2k} : S_u),\end{aligned}$$

(i.e. the only time where the logical transformation is non-zero is at time i and is equal to L which is assumed to be of weight 1). The final memory state would then be

$$M = L\Lambda_{\mathrm{M}}\mu_{\mathrm{M}}^{u-i} + \sum_{i=0}^{u} S_{u-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i. \tag{53}$$

Since, by assumption, the rows of Λ_{M} are in $\mathcal{V}_0$, there exists $S_0', \ldots, S_t'$ in $\mathbb{F}_2^{n-k}$ and N' in $\mathrm{Null}(\mu_{\mathrm{M}}^t)$ such that

$$L\Lambda_{\mathrm{M}}\mu_{\mathrm{M}}^{u+t+1-i} + \sum_{i=0}^{u} S_{u-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^{i+t+1} = N' + \sum_{i=0}^{t} S_{t-i}'\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i. \tag{54}$$

Thus, if we extend the path by the sequence of inputs

$$(0_{2k} : S_0') \to (0_{2k} : S_1') \to \cdots \to (0_{2k} : S_t'),$$

we arrive at the vertex M' which satisfies

$$\begin{aligned} M' &= M\mu_{\mathrm{M}}^{t+1} + \sum_{i=0}^{t} S_{t-i}'\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i \\ &= L\Lambda_{\mathrm{M}}\mu_{\mathrm{M}}^{u+t+1-i} + \sum_{i=0}^{u} S_{u-i}\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^{i+t+1} \\ &+ \sum_{i=0}^{t} S_{t-i}'\Sigma_{\mathrm{M}}\mu_{\mathrm{M}}^i \\ &= N' \end{aligned}$$

Extending this whole sequence by adding t zero transformations $(0_{2k} : 0_{n-k})$ will bring this path back to the origin since N' in in the kernel of μ_{M}^t. Once at the origin, then encoder can remain in that state forever without any additional physical output. This implies that the code is non recursive, and completes the proof. ■

REFERENCES

[1] S. A. Aly, A. Klappenecker, and P. K. Sarvepalli, *On quantum and classical BCH codes*, IEEE Trans. Info. Theor., 53 (2007), p. 1183.

[2] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, *Serial concatenation of interleaved codes: performance analysis, design, and iterative decoding*, IEEE Trans. Info. Theor., 44 (1998).

[3] C. H. Bennett, D. P. DiVincenzo, and J. A. Smolin, *Capacities of quantum erasure channels*, Phys. Rev. Lett., 78 (1997), pp. 3217–3220.

[4] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, *Mixed state entanglement and quantum error-correcting codes*, Phys. Rev. A, 54 (1996), p. 3824, quant-ph/9604024.

[5] E. R. Berlekamp, R. J. McEliece, and H. van Tilborg, *On the inherent intractability of certain coding problems*, IEEE Trans. Info. Theor., 24 (1978), p. 384.

[6] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near shannon limit error-correcting coding and decoding*, in ICC'93, Genève, Switzerland, May 1993, pp. 1064–1070.

[7] R. C. Bose and D. K. Ray-Chaudhuri, *On a class of error-correcting binary group codes*, Info. Contr., 3 (1960).

[8] A. R. Calderbank and P. W. Shor, *Good quantum error-correcting codes exist*, Phys. Rev. A, 54 (1996), pp. 1098–1105, quant-ph/9512032.

[9] T. Camara, H. Ollivier, and J.-P. Tillich, *Constructions and performance of classes of quantum ldpc codes*, 2005, quant-ph/0502086.

[10] H. F. Chau, *Good quantum convolutional error correction codes and their decoding algorithm exist*, 1998, quant-ph/9806032.

[11] ———, *Quantum convolutional correcting codes*, Phys. Rev. A, 58 (1998), pp. 905–909.

[12] I. Devetak, *The private classical capacity and quantum capacity of a quantum channel*, IEEE Trans. Info. Theor., 51 (2005), p. 44.

[13] D. P. DiVincenzo, P. W. Shor, and J. A. Smolin, *Quantum-channel capacity of very noisy channels*, Phys. Rev. A, 57 (1998), pp. 830–839, quant-ph/9706061.

[14] J. G. D. Forney, M. Grassl, and S. Guha, *Convolutional and tail-biting quantum error-correcting codes*, 2005, quant-ph/0511016.

[15] R. G. Gallager, *Low Density Parity Check Codes*, M.I.T. Press, Cambridge, Massachusetts, 1963.

[16] D. Gottesman, *Stabilizer codes and quantum error correction*, PhD thesis, California Institute of Technology, Pasadena, CA, 1997, quant-ph/9705052.

[17] M. Grassl and M. Rötteler, *Non-catastrophic encoders and encoder inverses for quantum convolutional codes*, in Proc. ISIT, IEEE, 2006, p. 1109.

[18] M. Hagiwara and H. Imai, *Quantum quasi-cyclic ldpc codes*, 2007, quant-ph0701020.

[19] M. Hastings, *Quantum belief propagation*, 2007, arXiv:0706.4094.

[20] A. Hocquenghem, *Codes correcteurs d'erreurs*, Chiffres, 2 (1959), p. 147.

[21] N. Kahale and R. Urbanke, *On the minimum distance of parallel and serially concatenated codes*, in Proc. IEEE Int. Symp. Info. Theo. (ISIT'98), 1998, p. 31.

[22] E. Knill and R. Laflamme, *Theory of quantum error-correcting codes*, Phys. Rev. A, 55 (1997), p. 900, quant-ph/9604034.

[23] C. Laumann, A. Scardicchio, and S. Sondhi, *Cavity method for quantum spin glasses on the Bethe lattice*, 2007, arXiv:0706.4391.

[24] M. Leifer and D. Poulin, *Quantum graphical models and belief propagation*, to appear in Ann. Phys., (2007), arXiv:0708.1337.

[25] S. Lloyd, *Capacity of the noisy quantum channel*, Phys. Rev. A, 55 (1997), p. 1613.

[26] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*, Cambridge University Press, Cambridge, UK, October 2003.

[27] D. J. C. MacKay, G. Mitchison, and P. L. McFadden, *Sparse graph codes for quantum error-correction*, IEEE Trans. Info. Theor., 50 (2004), pp. 2315–2330, quant-ph/0304161.

[28] M. Mézard and A. Montanari, *Constraint Satisfaction Networks in Physics and Computation*, Clarendon Press, 2007.

[29] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, UK, 2000.

[30] H. Ollivier and J.-P. Tillich, *Description of a quantum convolutional code*, Phys. Rev. Lett., 91 (2003), p. 177902, quant-ph/0304189.

[31] ———, *Quantum convolutional codes: fundamentals*, 2004, quant-ph/0401134.

[32] ———, *Trellises for stabilizer codes: definition and uses*, Phys. Rev. A, 74 (2006), p. 032304, quant-ph/0512041.

[33] D. Poulin, *Stabilizer formalism for operator quantum error correction*, Phys. Rev. Lett., 95 (2005), p. 230504, quant-ph/0508131.

[34] D. Poulin, *Optimal and efficient decoding of concatenated quantum block codes*, Phys. Rev. A, 74 (2006), p. 052333, quant-ph/0606126.

[35] D. Poulin and E. Bilgin, *Belief propagation for interacting systems on sparse graphs*, 2007, arXiv:0710.4304.

[36] D. Poulin and Y.-J. Chung, *On the iterative decoding of sparse quantum codes.* in preparation.

[37] I. S. Reed and G. Solomon, *Polynomial codes over certain finite fields*, J. SIAM, 8 (1960), p. 300.

[38] C. E. Shannon, *A mathematical theory of communication*, Bell System Tech., 27 (1948), pp. 379, 623.

[39] P. Shor, *Msri workshop on quantum computation.* http://www.msri.org/publications, 2002.

[40] P. W. Shor, *Scheme for reducing decoherence in quantum computer memory*, Phys. Rev. A, 52 (1995), p. 2493.

[41] G. Smith and J. A. Smolin, *Degenerate coding for Pauli channels*, 2006, quant-ph/0604107.

[42] A. M. Steane, *Error correcting codes in quantum theory*, Phys. Rev. Lett., 77 (1996), p. 793.

[43] ———, *Simple quantum error correcting codes*, Phys. Rev. A, 54 (1996), p. 4741.

[44] S. ten Brink, *Designing iterative decoding schemes with the extrinsic information transfer chart*, AEU Int. J. Electron. Commun., 54 (2000), p. 389.

[45] J. S. Yedidia, *Advanced mean field methods: theory and practice*, MIT Press, 2001, ch. An idiosyncratic journey beyond mean field theory, p. 21.