Brazilian Journal of Physics (2026) 56:94
https://doi.org/10.1007/s13538-026-02007-9

A Short Introduction to Quantum Error Correction

Thiago Lucena de Macedo Guedes[1]

Received: 13 September 2025 / Accepted: 2 January 2026

Abstract
As quantum information processing grows into one of the currently most impactful fields in physics and engineering, it becomes indispensable for physicists to understand what challenges lie ahead. This work aims at introducing the main strategies to circumvent the detrimental effects of noise in quantum information processing: the field known as quantum error correction.

Keywords Quantum computing · Quantum information processing · Quantum error correction

1 Introduction

Quantum computing has arguably become one of the most popular buzzwords in scientific media and literature. What started as a hypothesis that quantum systems could be far more suitable for simulating other quantum systems when compared to computers [1–3], rapidly turned into a strong conviction after the pioneering works of Shor, Deutsch and others [4–16]. In fact, the potential advantages predicted to be provided by quantum algorithms led physicists and engineers to even set experimental milestones on the performance of quantum computers being developed: the point at which a quantum information processor will be able to solve a problem infeasible for any classical computer in a reasonable amount of time ("quantum supremacy"), and the point at which a quantum information processor will be able to solve a practically relevant problem faster than any classical computer ("quantum advantage") [17, 18]. Notwithstanding recent debatable claims of quantum supremacy being demonstrated in some quantum information processors [19, 20], it is largely believed within the physical and engineering communities that these two milestones will only ever be achieved once quantum information processors are able to efficiently suppress errors [21]. This comes as a consequence of the frail nature of quantum systems and of the rich variety of ways in which they can be afflicted by their environment. The pursue for noise-resilient quantum-information-processing technologies led to the advent of quantum error correction: the study and design of mechanisms to encode, preserve and decode quantum information in a way that reduces the effects of computational errors originating from noise in the quantum information processor.

The fundamental building blocks of quantum information processors, the qubits, store quantum information in the amplitudes of their quantum states: $|\psi(\phi,\theta)\rangle = \cos\theta|0\rangle + e^{i\phi}\sin\theta|1\rangle$, where θ and ϕ parametrize the amplitudes generating the probability distributions of measuring states in a given basis (e.g., $|0\rangle$, $|1\rangle$ or any linear combination thereof). It is the probability distributions that ultimately carry the information stored in the qubit. In a more general scenario, multilevel systems known as qudits are used instead. It is not hard to see that, due to interactions with whichever other systems composing the environment of the qubits/qudits, the parameters describing the probability distributions for the measurement of quantum states in a given basis can be distorted [22, 23]. These interactions, referred to as noise, are the source of errors in quantum information processing. Quantum error correction aims at remedying this problem by redundantly encoding the information of a single qubit or qudit into an ensemble of such quantum systems [24]. By "redundancy" it is meant that a much larger Hilbert space is employed to encode information about a few quantum states, the codewords, belonging to a subspace of the Hilbert space known as code space, and that the most probable error mechanisms will take the codewords out of the codespace in a way that one can, with

✉ Thiago Lucena de Macedo Guedes
t.guedes@fz-juelich.de

[1] International Institute of Physics, Federal University of Rio Grande do Norte, Natal 59078-970, Brazil

Published online: 14 February 2026

high probability, efficiently track and reverse. Each of the schemes for doing so is termed a code, and the processing of information gathered about errors and their histories to propose the most probable correction is done by a (usually classical) decoder.

The field of quantum error correction is vast and consists of different codes, multiple decoders, as well as a variety of physical and mathematical principles underlying the very construction mechanisms of these codes and decoders (e.g., Calderbank–Shor–Steane codes, Bacon-Shor codes, topological codes, non-Abelian codes, low-density parity-check codes, decoherence-free subspaces, etc) [7, 25–35]. While quantum-error-correction schemes were originally proposed as adaptations of classical ones to the limitations imposed by quantum mechanics [7, 25, 26], such as the no-cloning theorem, later developments led to unique designs and techniques, several of which are introduced every year [31, 32, 36, 37]. This rich environment, sometimes termed a "zoo" [1], makes a complete in-depth review of quantum error correction a daunting task, and we limit ourselves in this review to the far less ambitious goal of providing a short and comprehensive introduction to the most essential aspects of quantum error correction. We start with a brief introduction to classical-error-correction theory in Section 2, laying down the key ideas necessary to understand the challenges and clever solutions formulated when adopting error correction to quantum information processors. Section 3 introduces the basic mathematical machinery of quantum error correction and describes error models, as well as early quantum-error-correction designs such as the repetition and Shor codes. In section 4 we introduce stabilizer codes, discussing in further details the toric, surface and Steane codes. We end the section with a brief discussion of a few other interesting designs in quantum error correction. This work ends with a discussion of advances and perspectives in quantum error correction (Section 6), followed by some concluding remarks.

2 Classical Error-correction Codes

A classical error-correcting code (redundantly) encoding the information corresponding to k bits, known as logical bits, in a set of n physical bits is denoted by $[n, k, d]$. The code-specific number d is called (Hamming) distance and measures the smallest number w of corrupted bits (also known as weight) necessary to convert one codeword into another. Since decoding requires making a decision about which codeword one most probably had before errors occurred, errors of sufficiently large weight can trick the decoding agent into assuming that one actually started with a different codeword. But how large is sufficiently large? It is not hard to see that, if the two most similar codewords differ by an error of weight d, any error or weight larger than half of this, i.e., $w \geq \lceil (d+1)/2 \rceil$ (where $\lceil \cdot \rceil$ stands for the ceiling or rounding-up operation), will trick the decoder into inferring the wrong codeword. Conversely, any error of weight $w \leq t = \lfloor (d-1)/2 \rfloor$ ($\lfloor \cdot \rfloor$ represents the floor or rounding-down operation) can be successfully detected and corrected by the decoder. It is worth noting that there are codes solely capable of detecting errors, the so-called error-detection codes, for which a successful error detection forces one to restart the computations instead of inferring the correct codeword.

Given n bits, the total number of bit-strings one can generate is 2^n. These bit-strings are usually represented as vectors with n binary entries, i.e., $W = [W_0, W_1, \ldots, W_{n-1}]^T$ with $W_i \in \mathbb{Z}_2 \cong \{0, 1\}$ (with addition modulo 2) and T representing the transpose. The addition of bit-strings is performed bit-by-bit, each considering operations modulo 2. A code $[n, k, d]$ uses 2^k elements from the linear span of 2^n bit-strings as codewords $W = \sum_b \beta_b W^{(b)}$, spanned from a basis of k (n-bit) vectors $W^{(b)} = \sum_a \alpha_a^{(b)} V^{(a)}$, with $b \in \{0, 1, \ldots, k-1\}$, $a \in \{0, 1, \ldots, n-1\}$ and $\alpha_a^{(b)}, \beta_b \in \mathbb{Z}_2 \, \forall \, a, b$. As an example, one can choose each of the n vectors $V^{(a)}$ to have a 1 in its a-th entry and zeros elsewhere, then select a basis of k vectors $W^{(b)}$ best-suited for the encoding (not necessarily having a single non-zero entry each); as we will see later, for the repetition code with a single logical bit, $k = 1$, one naturally chooses $W^{(0)} = [1, 1, \ldots, 1]^T$, and the two possible codewords, $[0, 0, \ldots, 0]^T$ and $[1, 1, \ldots, 1]^T$, are set by the choice of the single parameter β_0 as 0 or 1. Codewords can be expressed in a compact form by collecting all the coefficients β_b into a vector $B = [\beta_0, \beta_1, \ldots, \beta_{k-1}]^T$ and all the logical-basis bit-strings $W^{(b)}$ into an $n \times k$ generating matrix $G = [W^{(0)}, W^{(1)}, \ldots, W^{(k-1)}]$, so that $W = GB$. Codes are classified as linear if they are closed with respect to addition of codewords, i.e., if $W^{(b)} + W^{(c)} \equiv [W_0^{(b)} + W_0^{(c)}, \ldots, W_{n-1}^{(b)} + W_{n-1}^{(c)}]^T = W^{(l)}$ for any basis vectors $W^{(b)}$ and $W^{(c)}$ and some basis vector $W^{(l)}$ (once again, addition should be considered modulo 2).

Mathematically, we describe the error-detection capabilities of a code via an $(n-k) \times n$ parity-check matrix, H. This matrix is composed of $n-k$ linearly independent vectors orthogonal to each of the codewords considered: $H = [H^{(0)}, H^{(1)}, \ldots, H^{(n-k-1)}]^T$, with $(H^{(l)})^T W = 0$ for $l \in \{0, \ldots, n-k-1\}$ (the orthogonality with every logical basis element $W^{(b)}$ suffices guarantee the orthogonality with all of the 2^k codewords through linearity). The meaning of the relation $HW = [0, \ldots, 0]^T$ is that, as long as the

[1] See https://errorcorrectionzoo.org.

information is encoded in the code space of 2^k codewords, it remains different from any bit-string in the linear span of the basis $\{H^{(i)}\}$, i.e., the code-space complement. As soon as an error occurs, which we represent by a bit-string E to be added to the codewords, the resulting bit-string $W + E$ is considered correctable if it can be detected by the parity-check matrix, i.e., $H(W + E) = HE \neq [0, \ldots, 0]^T$. If $H(W + E) = [0, \ldots, 0]^T$, it turns out that $W + E$ is a new codeword, and so the error E, called logical error, converted one codeword into another. Logical errors affect the encoded information in a way that cannot be detected by the code, as mathematically expressed by the orthogonality with each of the vectors composing the parity-check matrix. Detectable errors can be corrected by applying the same error sequence to the error-afflicted state, i.e., $W + E \rightarrow (W + E) + E = W$ (because any vector of bits added bit-wise modulo-2 to itself gives the zero vector). It is worth noting, however, that inference about which error has occurred comes from $H(W + E)$, known as syndrome, and if two or more errors generate the same syndrome, decoding might result in additional errors by introducing the wrong E to the error-afflicted bit-string $W + E'$, such that $E + E' \neq [0, \ldots, 0]^T$ [38].

One of the most intuitive error-correction codes is the repetition code $[n, 1, n]$. We all know from daily experience that, in the case of noisy communication, repeating a message increases the chances that the person on the receiving end successfully understands it. The repetition code makes use of the same principle: it repeats one bit of information n times, so that the 0 bit is encoded as $[0, \ldots, 0]^T$ and the 1 bit as $[1, \ldots, 1]^T$. As previously mentioned, the logical basis is spanned by the single vector $W^{(0)} = [1, \ldots, 1]^T$, and the two codewords are given by $W = \beta_0 W^{(0)}$ with $\beta_0 \in \{0, 1\}$. If we choose a basis of vectors with a single non-zero entry at the a-th position $V^{(a)}$ to span the space of n-bit vectors, we see that $W^{(0)} = \sum_a \alpha_a^{(0)} V^{(a)}$ for $\alpha_a^{(0)} = 1 \,\forall\, a$. For the specific case of $n = 3$, the parity-check matrix is given by

$$H = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} . \tag{1}$$

There are 8 possible error sequences that can afflict a 3-bit code, namely, $E_0 = [0, 0, 0]^T$, $E_1 = [0, 0, 1]^T$, $E_2 = [0, 1, 0]^T E_3 = [0, 1, 1]^T E_4 = [1, 0, 0]^T E_5 = [1, 0, 1]^T$, $E_6 = [1, 1, 0]^T$, and $E_7 = [1, 1, 1]^T$. We see that HE_i gives $[0, 1]^T$ for $i \in \{1, 6\}$, $[1, 0]^T$ for $i \in \{3, 4\}$, $[1, 1]^T$ for $i \in \{2, 5\}$ and $[0, 0]^T$ otherwise. In other words, each syndrome corresponds to a pair of possible error configurations in the repetition code. These syndromes also reveal the meaning of "parity check": each row of the matrix (1) corresponds to a measurement of the parity (even for a 0 entry in the syndrome and odd for a 1 entry) of two adjacent bits in the bit-string. Therefore, each entry of the syndrome tells us whether the two adjacent bits measured match value-wise or not. Since all bits are matched for the two codewords of the repetition code, the presence of non-zero syndrome entries, called defects, reveals the occurrence of *detectable* errors, which in our case belong to the set $\{E_1, E_2, E_3, E_4, E_5, E_6\}$. This makes it clear why errors E_1 and E_6, for example, give the same syndrome, since they both have the same parities for each pair of adjacent bits and only differ by all entries being flipped relative to each other (which does not affect the parities). The same holds for all other pairs of errors sharing the same syndromes. But then how should the decoder infer what correction to apply? If the codeword was afflicted by E_1, one would have to decide between applying E_1 or E_6 as correction, knowing that mistakenly choosing E_6 would result in the erroneous codeword $W + E_1 + E_6 = \overline{W}$, where $\overline{W}$ represents the codeword generated from W by flipping every bit (i.e., a logical-bit flip). The wrong correction will therefore induce an unwanted logical operation on the encoded information. Luckily, if errors on different physical bits are uncorrelated, we can assume that if a single error on any of the physical bits occurs with probability p, two errors, corresponding to any of the configurations $\{E_3, E_5, E_6\}$, happen with probability p^2 and 3 errors with probability p^3. Since $p < 1$, $p^3 < p^2 < p$ and single physical-bit flips are the most probable error configurations. In fact, for reliable devices $p \ll 1$ and multiple physical-bit flips are extremely rare. The decoder can therefore implement the correction corresponding to the most probable error to maximize its error-correction performance: E_1 for the syndrome $[0, 1]^T$, E_4 for $[1, 0]^T$, E_2 for $[1, 1]^T$, and no correction for $[0, 0]^T$. Clearly, E_0 corresponds to no error at all and E_7 represents an error that cannot be detected (its weight is the same as the code distance), since it flips all bits in the system and therefore induces a logical-bit flip. Note that, for arbitrary n, the parity-check matrix evaluates the parity of each of the $n - 1$ pairs of adjacent bits, but for the decoder to pick up the wrong correction associated to a given syndrome it would require $w > \lfloor (n - 1)/2 \rfloor$ physical-bit flips to occur, yet the probability of such an event is as small as p^w. The repetition-code error-correcting performance therefore increases exponentially with the system size.

A last concept from error-correction theory necessary to understand a large class of quantum-error-correction codes are dual codes. These are derived from a code $[n, k, d]$ with generating and parity-check matrices G and H, respectively, by considering the new matrices $G_\perp = H^T$ and $H_\perp = G^T$. The new matrices $G_\perp$ and $H_\perp$ define a code $[n, n - k, d']$, dual to $[n, k, d]$, with a new distance d' depending on the specific features of the dual code.

3 First Steps into Quantum Error Correction

Knowing that a theory of error correction is already available for classical information processing, it is only natural to expected that these techniques can be directly applied on quantum information processors. In reality, a few subtle features of quantum systems turn this expected straight path to quantum error correction into a rather rocky road. In order to better understand how classical error correction can be adapted to quantum systems, one should first understand what are these subtle features quantum systems possess that find no counterpart in classical information processing.

The idea of repeating information, just as implemented with the repetition code, seems like an ideal first attempt at implementing error correction on a quantum system. Instead of having the single-qubit state $|\psi\rangle = \cos\theta|0\rangle + e^{i\phi}\sin\theta|1\rangle$, one could consider n copies of it, $|\psi\rangle^{\otimes n}$. But how would one generate such a state? Since we are talking about quantum mechanics, evolution operations are usually considered to be unitary. Even in a more general non-unitary setting, all operators considered act linearly on states. If one assumes that the operator $\hat{O}$ is such that $\hat{O}|\psi\rangle|0\rangle = |\psi\rangle|\psi\rangle\,\forall\,|\psi\rangle$, it results that the action of this operator on the state $|\psi\rangle = |\psi_1\rangle + |\psi_2\rangle$ should give the state $(|\psi_1\rangle + |\psi_2\rangle)(|\psi_1\rangle + |\psi_2\rangle)$. Linearity, on the other hand, leads to $\hat{O}|\psi\rangle = \hat{O}|\psi_1\rangle + \hat{O}|\psi_2\rangle = |\psi_1\rangle|\psi_1\rangle + |\psi_2\rangle|\psi_2\rangle$. Clearly the assumption that $\hat{O}$ can clone any state from the considered Hilbert space violates the linearity of the operator action on this Hilbert space. This result is known as the "no-cloning theorem" [39] and states that no operator capable of copying arbitrary quantum states exists. This particularity of quantum mechanics prevents one from encoding a quantum logical state in the mold of the classical repetition code. Nonetheless, the encoding can be modified to comply with the requirement of linear operator action on states: rather than copying the state n times, one can work with a single state whose codewords are composed of (copy states of) n qubits, e.g., $|\Psi\rangle = \cos\theta|0\rangle^{\otimes n} + e^{i\phi}\sin\theta|1\rangle^{\otimes n}$ (for the quantum repetition code).

Another important feature of quantum systems to keep in mind is the fact that measurements usually affect quantum states. A single-qubit measurement in the Z-basis is described by projectors $\hat{P}_0 = |0\rangle\langle 0| = (1+\sigma_z)/2$ and $\hat{P}_1 = |1\rangle\langle 1| = (1-\sigma_z)/2$, with the spin-1/2 Pauli matrices being represented by σ_k. Such a measurement on the state $|\psi\rangle$ results in the state $|0\rangle$ ($|1\rangle$) whenever the measurement reads the Pauli-Z eigenvalue $+1$ (-1), regardless of what the state $|\psi\rangle$ is; the parameter θ describing $|\psi\rangle$ (and/or ϕ for the case of measurements in other bases) merely affects the probabilities of measuring each Pauli-Z eigenvalue. As a result, after a measurement the state is forced into an eigenstate of the observable measured. Now consider measuring a single qubit (the i-th one) of the fiducial state $|\Psi\rangle = \cos\theta|0\rangle^{\otimes n} + e^{i\phi}\sin\theta|1\rangle^{\otimes n}$, given in the form of its density matrix $\hat{\rho}_0 = |\Psi\rangle\langle\Psi|$, for the observable $\hat{O}_i$ with eigenvalues O_i. In the most general case, the measurement is described by a positive operator-valued measure (POVM) with operators $\hat{M}_{O_i} = \sum_\omega \hat{K}_{O_i}^{(\omega)\dagger}\hat{K}_{O_i}^{(\omega)}$ [22, 40] that generalize the concept of projectors, and the post-measurement density matrix is given by $\hat{\rho} = (\sum_{\omega'} \mathrm{tr}\{\hat{K}_{O_i}^{(\omega')}\hat{\rho}_0\hat{K}_{O_i}^{(\omega')\dagger}\})^{-1}\sum_\omega \hat{K}_{O_i}^{(\omega)}\hat{\rho}_0\hat{K}_{O_i}^{(\omega)\dagger}$ (the specific choice of $\hat{K}_{\hat{O}_i}^{(\omega)}$ depends on the details of the measurement). In the simple case of a projective measurement, $\hat{K}_{O_i}^{(\omega)} \to \hat{P}_{O_i}$ with a single ω value allowed, and it becomes clear that $\hat{\rho} \neq \hat{\rho}_0$ unless $|\Psi\rangle$ is an eigenvector with nonzero eigenvalue of $\hat{P}_{O_i}$, and therefore also an eigenvector of $\hat{O}_i$. For the state $|\Psi\rangle$ considered, one operator whose projection on the eigenvectors of eigenvalue $+1$ gives $\hat{\rho} = \hat{\rho}_0$ is the Z-parity operator $\sigma_{z,i}\otimes\sigma_{z,j}$ with support on the i-th and j-th qubits. One can readily see the similarities to the parity measurements performed by the parity-check matrix (1) in the classical scenario.

We can now envision how to construct the quantum version of the repetition code. On the one hand, one cannot simply copy the state $|\psi\rangle = \cos\theta|0\rangle + e^{i\phi}\sin\theta|1\rangle$ into n qubits, but can instead work with a logical qubit given in the basis spanned by the two codewords of the classical repetition code, $|\Psi\rangle = \cos\theta|0\rangle^{\otimes n} + e^{i\phi}\sin\theta|1\rangle^{\otimes n}$. On the other hand, measuring parity by checking each of the qubits in a pair independently affects the density matrix, corrupting the encoded information, yet measuring the parity as a single observable $\sigma_{z,i}\otimes\sigma_{z,i+1}$ leaves the state unchanged and allows one to extract syndromes (whose defects are the eigenstates of this observable) in a similar manner to the classical-error-correction protocol. Whenever $\sigma_{z,i}\otimes\sigma_{z,i+1}|\Psi\rangle = -|\Psi\rangle$, we know that the states of the i-th and $(i+1)$-th qubits are different, violating a basic symmetry of the codewords. This can only mean that a bit-flip error happened in either of these qubits. The precise location of the most probable bit-flip error(s) is revealed by a second location where a parity violation is detected within the string of n qubits. The most probable error configuration is the one for which the smallest subsets of qubits connecting pairs of defects have been afflicted by bit-flip errors. For $n = 3$, this translates to the same scenario already investigated in the classical repetition code. In practice, the measurement of these syndromes is performed with the aid of auxiliary qubits known as ancillas. For n physical data qubits, one needs $n-1$ ancilla qubits. Considering the i-th ancilla to encode the parity information of data qubits i and $i+1$, each ancilla qubit serves as target for two CNOT gates, one

controlled by the i-th data qubit and another by the $(i+1)$ -th data qubit. The process leads to

$$\begin{aligned}\hat{E}(\{e_i\})|\Psi\rangle\otimes|0\rangle^{\otimes n-1} &= (\bigotimes_{i=0}^{n-1}\sigma_{x,i}^{e_i})(\cos\theta|0\rangle^{\otimes n} \\ &+ e^{i\phi}\sin\theta|1\rangle^{\otimes n})\otimes|0\rangle^{\otimes n-1} \rightarrow (\bigotimes_{i}\sigma_{x,i}^{e_i})(\cos\theta|0\rangle^{\otimes n} \\ &+ e^{i\phi}\sin\theta|1\rangle^{\otimes n})\bigotimes_{i=0}^{n-2}|e_i+e_{i+1}\rangle_i \end{aligned} \quad (2)$$

in which the error operator is represented by $\hat{E}(\{e_i\}) = \bigotimes_i \sigma_{x,i}^{e_i}$, an n-qubit Pauli string generated from the classical error vector $E = [e_0, e_1, \ldots, e_{n-1}]^T$ with binary entries e_i. The arrow in (2) represents the action of all CNOTs mapping the defects to the ancillas. The corresponding quantum circuit for $n = 3$ is shown in Fig. 1. Since the ancillas are not entangled with the logical state, one can measure them in the Pauli-Z basis to extract the syndrome without affecting the logical information. Although one could directly correct the logical state based on the most probable error configuration deduced from the measured syndrome, since implementing gates on the logical states usually introduces additional errors due to gate noise, it is often preferred to account for the most probable correction only at the end of the computation, once all data qubits have been measured.

Since qubit states are parametrized by θ and ϕ, it is reasonable to expect that errors would be able to affect each of these two degrees of freedom. So far, we have only discussed bit-flip errors, which find a direct counterpart in classical

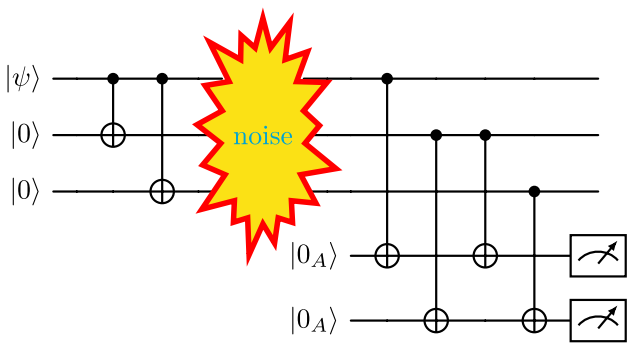

Fig. 1 Schematic representation of state encoding, noise application and syndrome measurement for the quantum repetition code [[3, 1, 1]]. On the left side, a single-qubit quantum state $|\psi\rangle$ is encoded into the repetition-code logical state $|\Psi\rangle$ via 2 CNOT gates. For simplicity, we assume that state encoding is noiseless (which is usually not the case). After state encoding, noise might afflict the logical state. The errors affecting the state are detected via CNOTs mapping parity information to two ancillas. The first and last pairs of consecutive CNOTs there can in general be applied simultaneously, leading to a single level of circuit depth. The parity information (defects) is collected via measurement of the ancilla states. These are not entangled to the logical states and solely contain classical information

error correction, but the reality is that qubits are also affected by phase flips. These are commonly represented by the Pauli operators $\sigma_{z,i}$ acting on qubit i as $\sigma_{z,i}|q_i\rangle = (-1)^{q_i}|q_i\rangle$, with $q_i \in \{0, 1\}$. In terms of parameters θ and ϕ, a bit flip effectively maps $(\theta, \phi) \rightarrow (\pi/2 - \theta, -\phi)$, while a phase flip maps $(\theta, \phi) \rightarrow (\theta, \pi + \phi)$, so that combining both error types, each parameter can be affected independently, as expected. The simultaneous occurrence of both types of error on the same qubit is described by the Pauli-Y operator $\sigma_{y,i} = i\sigma_{x,i}\sigma_{z,i}$. In fact, as a consequence of the interactions with the environment, it is often assumed that all 3 Pauli errors can affect any qubit with a certain probability. This process is mathematically described by an error channel involving Kraus operators given by the identity and Pauli matrices,

$$\varepsilon_i[\hat{\rho}] = (1 - \sum_{\alpha} p_\alpha)\hat{\rho} + \sum_{\alpha=x,y,z} p_\alpha \sigma_{\alpha,i}\hat{\rho}\sigma_{\alpha,i}\,, \quad (3)$$

in which p_α are the probabilities of occurrence of each error type. It is important to note that the channel (3) is trace-preserving, but not unitary: if the state $\hat{\rho}$ is pure, $\varepsilon_i[\hat{\rho}]$ is generally mixed. It is therefore called incoherent error channel [23, 24, 41]. The specific case for which all $p_\alpha = p/3$ is called depolarizing incoherent error channel. Physically, this error channel effectively simulates the noise afflicting qubits that either are left idling or have been interacting with the environment for a sufficiently long time. Likewise, miscalibration during state preparation or gate application is mathematically represented by a unitary coherent-error channel

$$\varepsilon_i[\hat{\rho}] = \hat{U}_i(\Theta)\hat{\rho}\hat{U}_i^\dagger(\Theta)\,, \quad (4)$$

in which $\hat{U}_i(\Theta) = e^{-i\Theta\vec{n}\cdot\vec{\sigma}_i/2}$ for some miscalibration angle Θ and direction $\vec{n}$, with $\vec{\sigma}_i = [\sigma_{x,i}, \sigma_{y,i}, \sigma_{z,i}]^\dagger$ being the vector of Pauli matrices for qubit i [24]. Error channels (3) and (4) assume no correlation between qubits, and therefore for a system of n qubits the total error channel is given by the composition of errors afflicting each qubit: $\varepsilon[\hat{\rho}] = \varepsilon_{n-1} \circ \varepsilon_{n-2} \circ \ldots \circ \varepsilon_0[\hat{\rho}]$. When performing operations on sets of qubits, it is expected that at least for an instant the noise affects multiple qubits in a correlated manner, therefore the channels (3) and (4) cannot properly describe this scenario. Instead, one commonly applies a depolarizing-error channel involving all possible Pauli strings of errors with support on the considered set of qubits, Q:

$$\begin{aligned}\varepsilon_Q[\hat{\rho}] &= \left(1 - \frac{4^{|Q|}p}{4^{|Q|}-1}\right)\hat{\rho} \\ &+ \frac{p}{4^{|Q|}-1}\sum_{\alpha_q=0,x,y,z}\left(\bigotimes_{q\in Q}\sigma_{\alpha_q,q}\right)\hat{\rho}\left(\bigotimes_{q\in Q}\sigma_{\alpha_q,q}\right)\,. \end{aligned} \quad (5)$$

In channel (5), $|Q|$ is the cardinality of the set (the number of qubits in it) and $\sigma_{0,q} = 1_q$ is the identity acting on qubit q [37]. The different ways in which one can implement the channels (3), (4), and (5) lead to different noise models. The simplest model to be considered is the so-called code-capacity noise, for which one merely considers a single application of the channel (3) or (4) (for incoherent or coherent noise, respectively) on every data qubit (i.e., excluding ancilla qubits) after encoding the logical state (state preparation). This model usually gives an upper bound to the quantum-error-correction capacity of a code and decoder. The phenomenological-noise model assumes that the error channel (3) or (4) is applied on every data qubit at the start of each quantum-error-correction cycle (i.e., each round of syndrome extraction and application of error-correcting gates, if any), as well as on ancilla qubits (or even data qubits) right before they are measured. Note that the probabilities of error occurrence for different processes, like (gates applied for) state preparation and/or a quantum-error-correction cycle, as well as measurements, may be different. Since gates are usually not perfect, as shown by the suboptimal fidelities observed in experiments [19, 20, 42–50], it is customary to consider the more complete circuit-level-noise model, which complements the phenomenological-noise model by additionally assuming that every gate application is succeeded by the channel (5) applied on qubits composing the support of each gate.

It is now clear that a quantum-error-correction code should be capable of detecting and enabling correction of at least the most probable error configurations involving both bit- and phase-flip errors. In this sense, the quantum repetition code discussed so far is not a complete quantum-error-correction code, since it cannot detect phase flips. Unlike the [n, 1, n] (classical) repetition code, its quantum counterpart is classified as an [[n, 1, 1]] code, with double brackets used to distinguish quantum codes from the classical ones. The latter has distance 1 because a single phase flip suffices to convert a codeword into another. In fact, for the quantum repetition code, a phase flip (or any odd number thereof) on a physical qubit is a logical operator, as can be seen by the change it causes in the encoded information: $(\theta, \phi) \to (\theta, \pi + \phi)$. It is not hard to adapt the quantum repetition code to detect phase flips instead. This can be achieved by applying Hadamard gates on every physical data qubit, $H_i|q_i\rangle = \sum_p (-1)^{pq_i}|p\rangle/\sqrt{2}$ for $p, q_i \in \{0, 1\}$, leading to $|\Psi\rangle \to \cos\theta|+\rangle^{\otimes n} + e^{i\phi}\sin\theta|-\rangle^{\otimes n}$. Transforming each CNOT gate from the data to the ancilla qubits with Hadamard unitaries results in CNOTs with control and target qubits swapped, therefore the syndromes for phase-flip errors are collected via two CNOTs controlled by each ancilla and targeting a pair of neighboring data qubits. It is easy to see that, starting with ancillas in the state $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$, a Pauli-Z error on a data qubit flips the corresponding ancilla state, and this process can be detected via a measurement in the Pauli-X basis, the basis of eigenstates of the σ_x matrix (or, equivalently, by applying a Hadamard gate before a Z-basis measurement). This change of basis, however, does not suffice to simultaneously detect and correct both bit and phase flips.

The efficiency of the quantum repetition code can be exponentially improved via concatenation: using logical qubits as data qubits for encoding a "higher-level" logical state in a hierarchic fashion. Starting with 9 data qubits, for example, one can, instead of encoding a single logical qubit as a [[9, 1, 1]] quantum repetition code, initially encode 3 (lower-level) logical qubits of [[3, 1, 1]] codes and then encode a 9-qubit (higher-level) logical state of a [[3, 1, 1]] quantum repetition code that uses 3 logical qubits from the lower level. Since the [[3, 1, 1]] can correct any single error of a given type, the probability of incorrectly decoding the information is $p^{(1)} = 3p^2(1-p) + p^3$ when each qubit can be affected by error with probability p. If one uses 3 such logical qubits to encode a higher-order one, then $p^{(2)} = 3(p^{(1)})^2(1-p^{(1)})^2 + (p^{(1)})^3$. If one continues this procedures, at the n-th hierarchy level one has a logical-error probability of $p_d^{(n)} = 3(p^{(n-1)})^2(1-p^{(n-1)}) + (p^{(n-1)})^3 \to (3p)^{2^n}/3|_{p\to 0}$. It can be shown that for $p \leq 1/2$ the concatenation exponentially improves the probability of correctly decoding the logical state of a quantum repetition code. This is a manifestation of the threshold theorem, which states that for a given choice of noise model, quantum-error-correction-code and decoder, there may be a noise level, called threshold, below which the logical-error probability can be arbitrarily suppressed at the cost of a hardware overhead[2] [56–58]. A construction similar to concatenation with 2 levels can actually render the quantum repetition code a full quantum-error-correction code capable of correcting both bit and phase flips. This is achieved by changing the basis of one of the hierarchy levels, so that bit and phase flips are detected at different levels. This construction, called the Shor code, leads to a [[9, 1, 3]] quantum-error-correction code[3] [7]. Its logical states are usually encoded as either $|\pm\rangle_L = 2^{-3/2}[|000\rangle \pm |111\rangle]^{\otimes 3}$ or $|q\rangle_L = 2^{-3/2}[|++\rangle + (-1)^q|--\rangle]^{\otimes 3}$ (for

[2] It is important to note that the derivation of the threshold theorem implicitly assumes the availability of effectively global connectivity between the qubits, implemented in the form of concatenation and/or of a global decoder. Strictly local quantum-error-correction strategies might lack a nonzero threshold [37, 51–55]. By strictly local it is meant that neither the quantum nor the classical auxiliary systems (e.g., the decoder) can have access to any kind of information of global character, such as the system size.

[3] For reasons that will become clear when discussing gauge codes, we adopt here a convention slightly different from the one presented in Shor's original work.

$q \in \{0,1\}$), with 3-qubit blocks represented respectively as rows or columns in Fig. 2. For the logical state $|\pm\rangle_L$, the parity measurements in the lower (higher) level, detecting bit (phase) flips, are conducted via the $\sigma_{z,i} \otimes \sigma_{z,i+1}$ operators for $i \in \{0,1,3,4,6,7\}$ ($\sigma_{x,i} \otimes \sigma_{x,i+1} \otimes \sigma_{x,i+2} \otimes \sigma_{x,i+3} \otimes \sigma_{x,i+4} \otimes \sigma_{x,i+5}$ operators for $i \in \{0,3\}$), acting on a pair of neighboring qubits (blocks) and shown as a purple (orange) solid rectangle in Fig. 2. Similarly, for $|q\rangle_L$, parity measurements on the lower (higher) level are given by $\sigma_{x,i} \otimes \sigma_{x,i+3}$ for $i \in \{0,1,2,3,4,5\}$ ($\sigma_{z,i} \otimes \sigma_{z,i+1} \otimes \sigma_{z,i+3} \otimes \sigma_{z,i+4} \otimes \sigma_{z,i+6} \otimes \sigma_{z,i+7}$ for $i \in \{0,1\}$), acting on the pair of neighboring qubits (blocks) represented as an orange (purple) dashed rectangle in Fig. 2. Considering the action of the logical operators X_L and Z_L to be $X_L|q\rangle_L = |q+1\rangle_L$ (modulo 2), $Z_L|q\rangle_L = (-1)^q|q\rangle_L$, $X_L|\pm\rangle_L = \pm|\pm\rangle_L$, and $Z_L|\pm\rangle_L = |\mp\rangle_L$, these are given by $X_L = \sigma_{x,i} \otimes \sigma_{x,i+1} \otimes \sigma_{x,i+2}$ ($i \in \{0,3,6\}$) and $Z_L = \sigma_{z,i} \otimes \sigma_{z,i+3} \otimes \sigma_{z,i+6}$ ($i \in \{0,1,2\}$) for both logical bases. The Shor code is arguably the most didactical quantum error correction code known, and a lot of its key features can be found in other codes, forming classes of codes with common relevant properties. One of these features is the usage of stabilizers, which are the parity operators for the Shor code. We shall therefore move to the broader study of stabilizer codes.

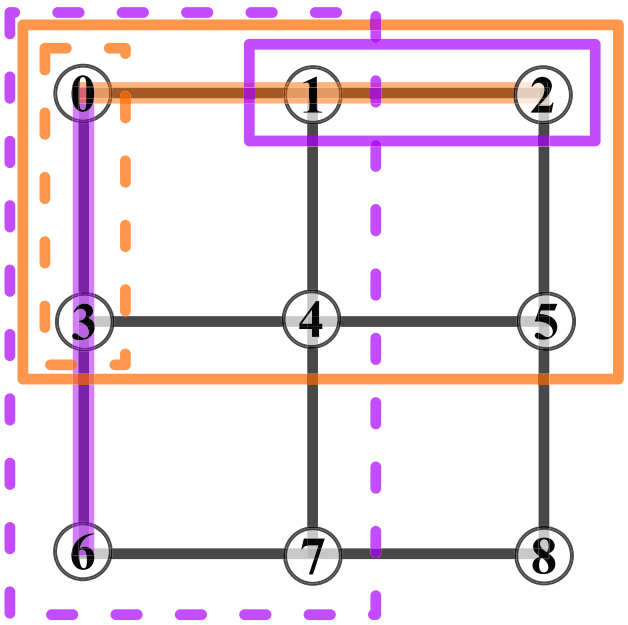

Fig. 2 Schematic representation of the [[9, 1, 3]] Shor code. The lattice or grid (black lines) represents the connectivity of qubits (numbered vertices). For the logical basis $|\pm\rangle_L = 2^{-3/2}[|000\rangle \pm |111\rangle]^{\otimes 3}$, whose blocks are encoded in rows, one X (higher hierarchy level) and one Z (lower hierarchy level) stabilizers are represented as orange and purple solid rectangles, respectively. For $|q\rangle_L = 2^{-3/2}[|+++\rangle + (-1)^q|---\rangle]^{\otimes 3}$ ($q \in \{0,1\}$), with blocks encoded in columns, one Z (higher hierarchy level) and one X (lower hierarchy level) stabilizers are respectively represented as purple and orange dashed rectangles. A specific choice of logical X and Z operators is also shown as orange horizontal and purple vertical lines (the other two parallel lines are equivalent for each direction). Since the X and Z logical operators intersect at exactly one qubit, one has $\{X, Z\} = 0$

4 Stabilizer Codes

In group theory, a stabilizer subgroup G_x is defined as the group formed by all elements $g \in G$ of a group G that fulfill the relation $gx = x$ for a given element $x \in H$ of a set H. In the context of quantum error correction, the set in question is usually the Hilbert space of multiqubit states, and the group acting on it is the Pauli group. The latter is defined as $P^{(n)} = \{i^k \bigotimes_{i=0}^{n-1} \sigma_{u,i} \,|\, k \in \{0,1,2,3\}, u \in \{0,x,y,z\}\}$ for a system of n qubits, but for practical purposes, one considers only its subgroups that do not contain the element $-\bigotimes_{i=0}^{n-1} 1_i$. Since $P^{(n)}$ is a closed set (under the action of any of its elements), the absence of $-\bigotimes_{i=0}^{n-1} 1_i$ excludes any element whose square is the negative identity. The remaining elements of the constrained Pauli group have the identity as their square, and therefore possess eigenvalues ± 1. The subgroups formed by Pauli strings with eigenvalue $+1$ when acting on a given common eigenstate are precisely stabilizer subgroups. The intersection of all these subgroups for the codeword states in the code space form the stabilizer group $S = \{\hat{S}_k\}$ of a stabilizer code. Elements of the constrained Pauli group that map stabilizers into stabilizers via conjugation form the normalizer subgroup, and any normalizer that is not a stabilizer is a logical operator. Pauli strings $\hat{E}_j$ not contained in the normalizer subgroup represent detectable errors and do not commute with at least one stabilizer (while all stabilizers commute with each other), therefore generating eigenstates of eigenvalue -1 when acting on codewords: $\hat{S}_k(\hat{E}_j|\Psi\rangle) = -\hat{E}_j(\hat{S}_k|\Psi\rangle) = -(\hat{E}_j|\Psi\rangle)$. The violation of the stabilizer condition provides a defect from which the (most probable) error can be inferred. It is common to describe stabilizer codes by explicitly giving the generators of its stabilizer group and of the logical operators [24, 33, 58, 59]. This is precisely the approach we will follow in this section.

4.1 Toric Code

A code that makes direct use of the properties of topological many-body systems to enable the protection of quantum information is Kitaev's $[[2L^2, 2, L]]$ toric code [28–30]. The qubits of a toric code are arranged (not necessarily physically, but rather in terms of connectivity) on the edges of a square tessellation of a torus. Assuming that the tessellation

is symmetric in terms of number L of squares counted along both meridional and longitudinal directions of the torus, a torus with L^2 square tiles, called plaquettes, contains $2L^2$ independent edges (effectively 2 per plaquette), each representing a physical qubit. The quantum information is encoded in a delocalized manner, such that error configurations attain a geometrical interpretation and only error chains that fully encircle the topologically non-trivial[4] (longitudinal or meridional) directions of the torus become logical, thus explaining the code distance L. The codespace basis is composed of 4 complicated linear combinations of states with similar topological properties, namely, the presence or absence of topologically non-trivial bit- or phase-flip chains that characterize logical operations. As there are two types of chains (composed of Pauli-Z or -X physical operations) and two directions on the torus, one has a total of 4 logical states or 2 logical qubits.

The toric-code stabilizers are local weight-4 Pauli strings whose non-identity elements encircle either a plaquette or a vertex of the tessellation. In general, bit flips are detected via plaquette stabilizers with support on the 4 qubits surrounding a plaquette p, $\hat{B}_p = \bigotimes_{l\in p} \sigma_{z,l}$ (here p labels a plaquette of choice whose edges are labelled by l, and we omit the identity operators applied on all other qubits). Similarly, phase flips are detected by vertex or star stabilizers with support on the 4 qubits connected to each other by a vertex v, $\hat{A}_v = \bigotimes_{v\in l} \sigma_{x,l}$. The stabilizers are represented in Fig. 3. It is easy to see why $\hat{A}_v$ and $\hat{B}_p$ stabilizers commute: if their supports do not overlap, the Pauli matrices in $\hat{A}_v$ and $\hat{B}_p$ are completely different, otherwise the two operators have overlapping support in exactly two qubits, so that the Pauli-matrix anti-commutation factor -1 happens to be squared.

Imagine an X error afflicts a single physical qubit corresponding to a certain edge (labeled l') of the tessellated torus. The two plaquette stabilizers with borders on this edge will act on the error-afflicted state as

$$\begin{aligned}\hat{B}_p[\hat{E}_x(\{e_{l'}=1, e_l=0\,\forall\, l\neq l'\})|\Psi\rangle] &= (\bigotimes_{m\in p}\sigma_{z,m})|_{l\in p}(\sigma_{x,l'}\bigotimes_{l\neq l'}1_l)|\Psi\rangle \\ &= -(\sigma_{x,l'}\bigotimes_{l\neq l'}1_l)(\bigotimes_{m\in p}\sigma_{z,m})|_{l\in p}|\Psi\rangle \\ &= -[(\sigma_{x,l'}\bigotimes_{l\neq l'}1_l)|\Psi\rangle]\,. \end{aligned} \tag{6}$$

The syndrome of a single Pauli-X error is therefore composed of two plaquette defects, also known as anyons. The defects originate from the Pauli-X error anticommuting with a single Pauli-Z matrix composing each of the plaquette stabilizers with support on the error-afflicted qubit at the considered edge. On the other hand, if the number of error-afflicted qubits on the support of a stabilizer is even, the anticommutation of Pauli matrices leads to a $(-1)^2 = +1$ commutation factor between the stabilizer and the local error configuration. One could therefore imagine that having errors on a pair of adjacent qubits instead would recover

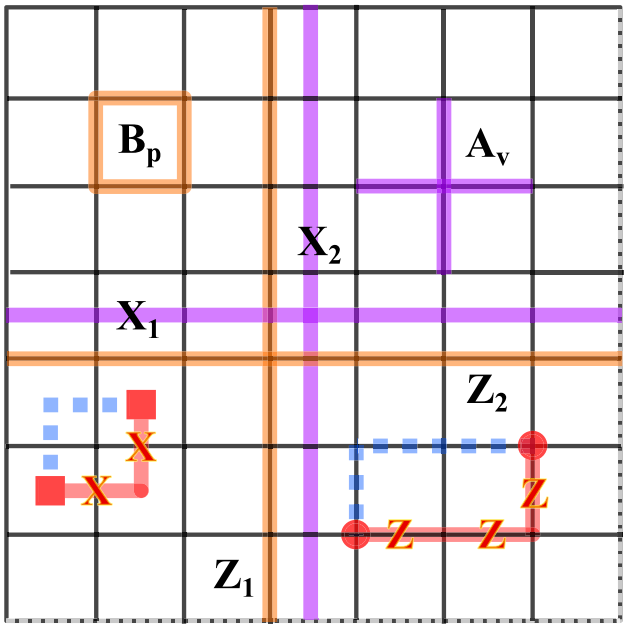

Fig. 3 Schematic representation of a [[98, 2, 7]] toric code. The lattice or grid (black lines) is a tessellation of a torus in which qubits are represented as edges. The bottom and right boundaries of the lattice are dashed because they have no qubits (periodic boundary conditions) and should be identified with the top and left boundaries, respectively. Plaquette and vertex stabilizers are represented as orange squares and purple crosses, respectively. A specific choice of logical X and Z operators of the two logical qubits (labeled by subscripts 1 and 2) is also shown as meridional and longitudinal lines, although any other such lines are equally suitable logical operators. Note that the purple lines intersect perpendicularly the edges representing qubits acted upon, while the orange lines intersect them tangentially. Since the logical operators X_1 and Z_1 intersect at exactly one qubit, one has $\{X_1, Z_1\} = 0$, and a similar argument shows that $\{X_2, Z_2\} = 0$, as expected from Pauli operators. The relations $[X_1, X_2] = [Z_1, Z_2] = 0$ come from the fact that X_i and Z_i are each composed of the same type of Pauli operators for $i \in \{1, 2\}$. The red strings mark error chains. Chains of Z-errors (marked in red on the bottom right of the lattice) can only be detected by vertex stabilizers at their ends, forming defects called electric anyons (red circles). Analogously, chains of X-errors (marked in red on the bottom left of the lattice) can only be detected by plaquette stabilizers at their ends, forming defects called magnetic anyons (red squares). Any chain of errors with the same endpoints gives the same syndrome, but the topological features of the toric code allow for correction along any other chain that closes the error Pauli string as a loop without winding around either of the homologically notrivial directions in the torus (an example is given by the dotted lines in blue)

[4] These topologically non-trivial loops are often called homologically non-trivial in the literature (see, e.g., Ref. [60]), but the reason for that requires some knowledge of group theory and homology theory. The idea can be summarized as homologically trivial loops being boundaries of a surface, while non-trivial loops have no such correspondence. Alternatively, one can think of these trivial loops as closed curves contractible to a point.

a codeword. This assumption, however, overlooks the fact that a neighboring plaquette stabilizer would detect a defect instead. For a (connected) open chain of errors, there are always two defects, one at each end of the chain (see red error chains in Fig. 3). The only way to recover a codeword is by closing these chains to form topologically trivial loops (their specific shapes are irrelevant). As a result, the several states in a superposition corresponding to one of the toric-code logical states contain all possible configurations of closed Pauli-operator loops. It is not hard to see that a similar reasoning applies for Pauli-Z errors. In fact, by taking the dual lattice to the considered tessellation of the torus, vertex stabilizers are mapped into plaquette ones and vice versa. The Pauli-Z loops that trigger no defects are dual-lattice ones, and cross perpendicularly the edges of the plaquettes (e.g., by encircling vertices) in the original tessellation. As topologically non-trivial loops also belong to the set of Pauli chains with no open ends, they do not trigger any defects under the action of stabilizers. Nonetheless, these loops do affect the encoded logical information, since the logical operations on the toric code are given by Pauli-Z and -X strings along the logitudinal and meridional directions of the torus (in fact, the logitudinal Pauli-X and the meridional Pauli-Z act on one logical qubit, and the logitudinal Pauli-Z and the meridional Pauli-X act on the other logical qubit). Logical X (Z) operators deformed by any number of vertex (plaquette) stabilizers (by multiplication of the corresponding Pauli strings or, equivalently, by merging the contour lines of these operators on the lattice, with overlapping contours "disappearing") are still valid logical operators (homologous) as a consequence of the topological features of the toric code.

The extraction of syndromes in its most standard design takes place with the aid of one ancilla qubit per stabilizer. Geometrically, these ancillas are portrayed as located in the center of the plaquettes or vertices of the lattice. The plaquette stabilizer is measured by applying 4 CNOT gates controlled by the plaquette-support qubits and targetting the ancillas initialized in the $|0\rangle$ state, followed by ancilla measurements in the Pauli-Z basis. The vertex stabilizer is measured with 4 CNOT gates controled by an ancilla initialized in the $|+\rangle$ state and targetting the 4 qubits forming the vertex support, and consecutively measuring the ancilla in the Pauli-X basis.

Under code-capacity noise, the threshold of the toric code decoded with minimum-weight perfect matching for depolarizing noise [cf. Eq. 3] is 15.5% [61]. For single-qubit noise levels below these thresholds, increasing the lattice size L leads to an exponential suppression on the logical-error probability. The hardware overhead in this case is the given by the additional data and ancilla qubits required, as well as gates to implement state preparation, syndrome extraction, logical operations and (if needed) error-correcting operations on data qubits.

4.2 Surface Code

For several quantum-information-processing platforms, the connectivity required to implement a toric code is rather demanding and impractical. Instead, similar codes with a similar phenomenology to the toric code but different boundary conditions have been proposed, the surface codes [29, 30, 62].

In its non-rotated form, the $[[L^2+(L-1)^2, 1, L]]$ surface code has its qubits connected as in a tessellation of a surface with asymmetric boundary conditions, with plaquette edges at the boundaries in one direction and boundaries cutting through the plaquette centers along the meridional or longitudinal direction [cf. Figure 4(left)]. When compared to the toric code, this modification allows for lower connectivity and qubit consumption, but comes at the cost of halving the number of logical qubits and modifying the stabilizers at the boundaries. Away from the boundaries, plaquette and vertex stabilizers are just the same as for the toric code. However, at each pair of opposite borders (left and right or top and bottom) one type of stabilizer will have its support reduced to 3 qubits. The logical operations are Pauli strings connecting opposite borders, with longitudinal and meridional directions corresponding to different Pauli types (X or Z), depending on how the asymmetric boundary conditions are chosen.

A rotated version of the surface code renders it a $[[L^2, 1, L]]$ code [63]. Its construction starting from the non-rotated distance-L surface code is achieved by removing $(L-1)^2/4$ qubits around each of the 4 lattice corners until an L^2-qubit diamond is left and then rotating the lattice by $\pi/4$ around the axis perpendicularly crossing the surface center [cf. Figure 4(right)]. The rotated surface code possesses $(L-1)/2$ modified 2-qubit stabilizers at each border (L odd). The logical operators in the rotated surface code also connect opposite borders, forming a diagonal or a zigzag path (or a composition of both).

The threshold for the rotated surface code under code-capacity noise decoded with minimum-weight perfect matching for depolarizing noise is 0.82% [64]. The suppression of logical errors with the increase in code distance for the rotated surface code has recently been experimentally shown for distances 3, 5 and 7 [43].

4.3 Steane Code and Color Codes

One interesting classical-error-correction code is the [7, 4, 3] Hamming code, for which a spatial distribution of its bits through a triangle (or a topologically equivalent

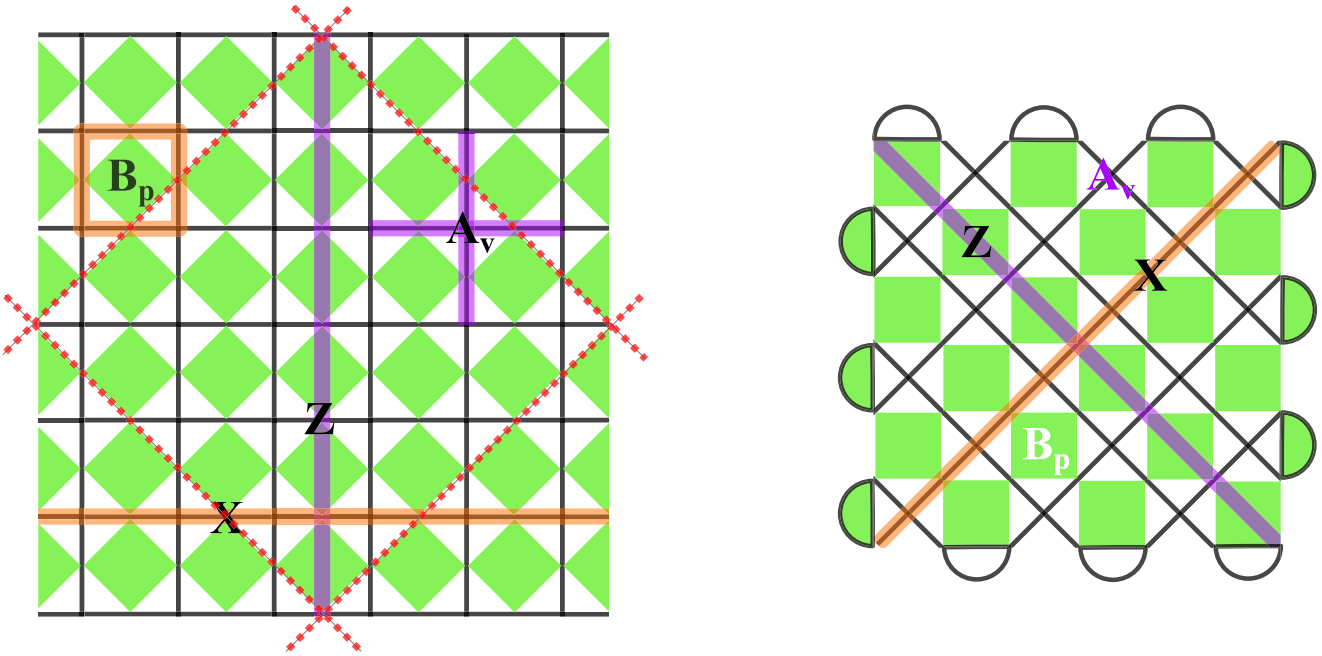

Fig. 4 Schematic representation of a [[85, 1, 7]] non-rotated (left) and a [[49, 1, 7]] rotated (right) surface codes. The lattice or grid of the non-rotated surface code (black lines) is a tessellation of a surface with asymmetric boundary conditions in which qubits are represented as edges. Equivalently, one can consider the chessboard-like grid formed by green and white diamonds, in which qubits are represented as diamond vertices. Plaquette and vertex stabilizers are represented as orange squares and purple crosses, respectively. Equivalently, they can be represented as green and white diamonds. The logical X and Z operators are also shown as meridional and longitudinal lines. Note that the purple line intersects perpendicularly the edges representing qubits acted upon, while the orange line intersects them tangentially. After cutting the left lattice along the red dashed lines and rotating the resulting lattice by $\pi/4$, one attains the rotated surface code shown on the right. Weight-2 stabilizers at the borders are represented as semi-circles. Green and white squares represent Z and X stabilizers (or, analogously, plaquette and vertex ones)

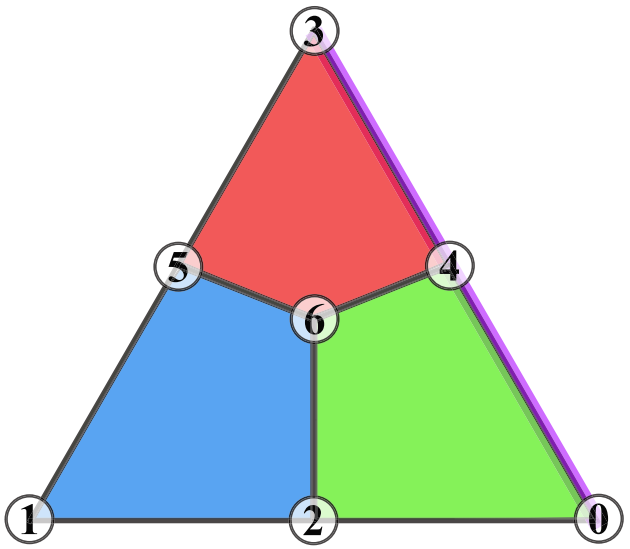

Fig. 5 Graphical representation of qubit connectivity in the [[7, 1, 3]] Steane code. The numbered vertices represent the 7 physical qubits. The green, red, and blue zones represent the support of both X and Z stabilizers (e.g., the blue stabilizers are given by $S_{b,t} = \sigma_{t,1} \otimes \sigma_{t,2} \otimes \sigma_{t,5} \otimes \sigma_{t,6}$ for $t = x, y$). The purple line covering qubits 0, 3, and 4 represents a possible logical operation consisting of either σ_x (for logical X) or σ_z (for logical Z)

hexagon), with 3 bits on each edge and one bit at the center, allow for a simple interpretation of its parity-check matrix

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} . \tag{7}$$

Numbering the bits at the triangle vertices as 0, 1 and 3, and the center bits of the 0-1, 1-3 and 3-0 edges as 2, 5, and 4, so that the bit at the center of the triangle is numbered 6, the 3 rows of the matrix (7) correspond to parity measurements involving the 4 bits around each of the triangle vertices, and any two such bit sets overlap on exactly two bits. A single bit flip at a vertex therefore only generates a defect on the corresponding bit set, while a bit at an edge center, where two sets overlap, triggers a defect in the two overlapping sets (two rows of the parity-check matrix) and a bit flip at the center bit is detected by all 3 parity checks, since all bit sets overlap there. These sets can be labeled as 3 colors: red, green and blue (cf. Figure 5).

A quantum-error-correction code can be constructed from two overlayed Hamming codes, one dealing with X and another with Z errors. This prescription follows the quantum-error-correction code design from two

classical-error-correction codes proposed by Calderbank, Shor, and Steane [25, 26] (forming a special class of stabilizer codes) and results in the [[7, 1, 3]] Steane code [65]. The 3 Pauli-X or Pauli-Z stabilizers from the Steane code, usually associated with colors, correspond to strings of Pauli-X or -Z matrices raised to powers given by the entries in each row of the parity-check matrix (7): $\hat{S}_{x,j} = \otimes_{i=0}^{6} \sigma_{x,i}^{H_{ji}}$ and $\hat{S}_{z,j} = \otimes_{i=0}^{6} \sigma_{z,i}^{H_{ji}}$. Logical operators are in principle given by Pauli-X or -Z operations applied on all physical qubits, but since stabilizers applied to logical operators generate equivalent logical operators, it is not hard to see that acting with Pauli operators on all 3 qubits of any of the triangle edges suffices to implement a logical operation, thus explaining the code distance 3.

The Steane code is the smallest known instance of a class of bidimensional quantum-error-correction codes known as color codes. These are defined on trivalent, 3-colorable lattices, such that each colored tile serves as support for both X and Z stabilizers [60, 66, 67]. The qubit connectivity in such codes is represented by lattices with qubits on their vertices. We assign a different color (e.g., red, green, or blue) to each of the 3 edges connected to each vertex, and color faces in such a way that two faces of the same color are connected by an edge of that color (cf. Figure 6). Correspondingly, neighboring faces always have different colors. Although faces can have varied shapes, we consider here solely lattices with certain symmetries.

Consider a hexagonal tessellation of a torus with colors attributed to its faces and edges as previously described (Fig. 6). A lattice with n hexagons is formed by $2n$ physical qubits. On each hexagonal face p, we define two plaquette stabilizers, one corresponding to a Pauli string of $\sigma_{x,i}$ operators and the other by $\sigma_{z,i}$. These stabilizers inherit the color labels $c \in \{r, g, b\}$ from the respective faces associated with their supports, i.e., $\hat{B}_{\tau,p^c} = \prod_{v \in p^c} \sigma_{\tau,v}$ (with $\tau \in \{x, z\}$) for the six vertices v of the c-colored face p^c. Since the product of all τ-type plaquette stabilizers of the same color has as support all the $2n$ qubits, i.e., $\prod_{p^c} \hat{B}_{\tau,p^c} = \prod_v \sigma_{\tau,v}$, it results that $\prod_{p^r} \hat{B}_{\tau,p^r} = \prod_{p^g} \hat{B}_{\tau,p^g} = \prod_{p^b} \hat{B}_{\tau,p^b}$, and two stabilizer generators of each type, X and Z, are unnecessary. The stabilizer group is therefore generated by $2(n-2)$ plaquette operators (the multiplicity 2 comes from the two possible choices of τ). The number of logical qubits is given by the difference between the numbers of physical qubits and constraints (stabilizer generators), so this hexagonal color code on a torus encodes 4 logical qubits. It is possible to show that color codes constructed on tessellations of tori of genus g encode $4g$ logical qubits [60, 66]. This is twice as many logical states as for a similarly generalized the toric code. One can therefore say that the color code on a torus represents "two colors of the toric code".

Like in the toric code, the logical operators of this color code on a torus are given by topologically (or homologically) non-trivial chains of Pauli X or Z operators winding

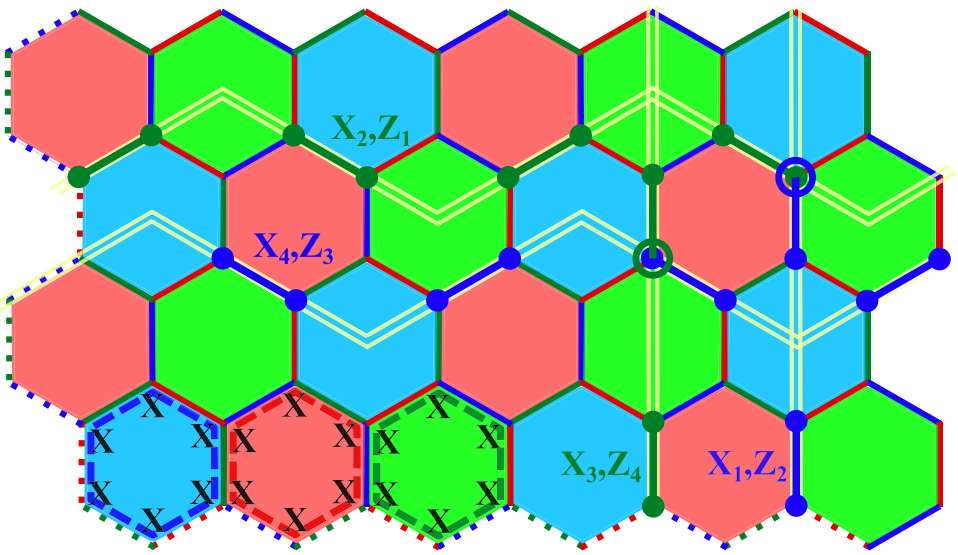

Fig. 6 Graphical representation of a [[48, 4, 4]] color code on a torus. Physical qubits are represented as vertices. Dotted edges on bottom and left sides of the lattice should be identified with top and right edges of same color and corresponding position (periodic boundary conditions). The red, green, and blue hexagons represent the support of each type and color of plaquette stabilizer, with the specific Pauli strings for the red, green and blue X-type stabilizers explicitly given on the bottom left corner (dashed colored hexagons with labeled vertices). Z stabilizers have the same support. The color scheme is such that at every vertex three edges of different colors meet, and these edges only connect vertices from pairs of faces of the same color. One possible choice of logical operators is shown as solid double yellow lines, with the qubits acted upon highlighted as solid circles. Note that the logical operators X_i and Z_j intersect at an odd number of qubits (solid circles within larger rings) when the indices match, while for different indices the overlap happens at zero or an even number of qubits

around either the longitudinal or meridional directions on the torus. Although each such chain can have three colors, it turns out that the product of one chain of each color for the same τ is equivalent (or homologous) to the identity (up to multiplication by some plaquette stabilizers). Consequently, only two colors of non-trivial loops are needed for each direction and type τ, resulting in eight logical operators, as expected for four logical qubits. it is completely arbitrary which colors to choose, and any homologically non-trivial chain along the same direction is equally good as a logical operator. In Fig. 6, an example is given for the choice of colors green and blue for the logical operators X_i and Z_i. Note that the same chain can be of either X or Z type (with different indices), and the logical operators with the same index intersect at exactly one qubit ($\{X_i, Z_i\} = 0$), while for different indices they intersect at either none or an even number of qubits ($[X_i, Z_j] = 0$ for $i \neq j$).

Syndrome extraction in the color code happens similarly to the toric code, with two ancillas needed per plaquette (one for each τ) and six CNOTs for the extraction of each type of defect, X or Z. In the simplest measurement scheme, one measures all plaquettes of a given color c and type τ at a time, and one quantum-error-correction cycle is performed after six rounds of measurements.

One can also design color codes with boundaries. These variants encode less qubits, but in exchange demand lower qubit connectivity and, for some specific designs, allow for the transversal implementation of a complete set of logical Clifford gates. Transversal logical gates are of the form $\bigotimes_{i\in D} \hat{U}_i$, with single-qubit unitaries $\hat{U}_i$ applied on a set D of qubits, what avoids error propagation from one qubit to another and therefore enables implementation of logical gates fault-tolerantly (i.e., designs in which failure of one circuit component does not compromise the entire computation [68]). These advantages justify briefly discussing triangular color codes beyond the Steane code.

One specific color-code design for which the entire logical Clifford group can be implemented transversally is the [[73, 1, 9]] code shown in Fig. 7 [60]. Interestingly, its logical X and Z gates can be implemented as Y-shaped stringnets formed by X or Z operators, respectively. These stringnets are composed of strings of all three colors, each connecting to a correspondingly colored boundary. Two stringnets of different types intersect at an odd number of qubits, resulting in the anticommutation of X and Z logical operators. The faces in this tessellation can be either octagons (two colors) or squares (one color), with supports of eight and four qubits, respectively. The stringnets can be deformed by multiplication with plaquette stabilizers to render equivalent logical operators that run solely through one boundary (any boundary is equally suitable) or that run from one boundary to the opposite triangle vertex.

5 Other Relevant Code Types

5.1 Gauge Codes

A family of codes with interesting stabilizer-like features is that of subsystem codes, sometimes also called gauge codes [69]. Arguably, the most famous such code is the $[[L_1L_2, 1, \min(L_1, L_2)]]$ Bacon-Shor code [7, 27]. Its working mechanism illustrates the general idea behind gauge codes: a quantum system with certain unconstrained

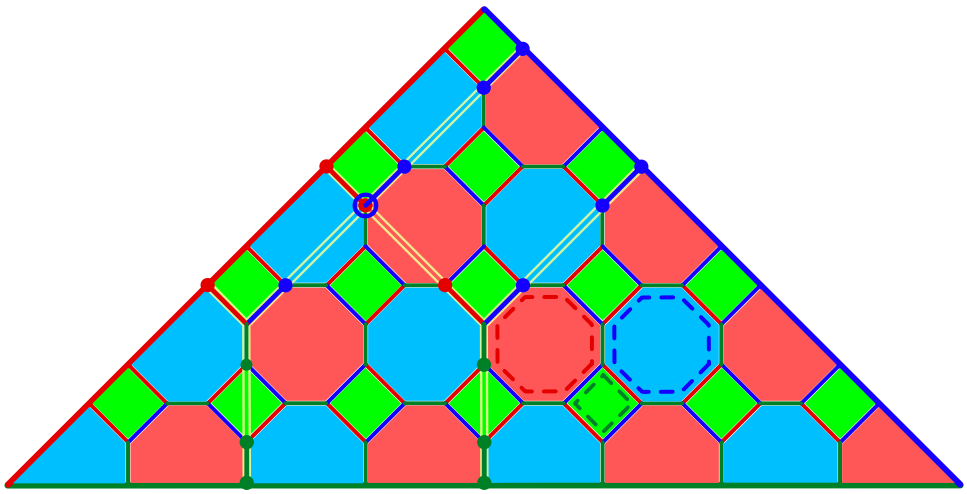

Fig. 7 Graphical representation of a [[73, 1, 9]] triangular color code on a 4-8-8 lattice with boundaries. Physical qubits are represented as vertices. Red, green, and blue faces represent the support of each type and color of plaquette stabilizer, with examples of red, green and blue stabilizers explicitly given as dashed colored polygons. One possible choice of logical operators (stringnets) is shown as solid double yellow lines, with the qubits acted upon highlighted as solid circles. Note that choosing one of the logical operators as X_i and another as Z_j, they intersect at an odd number of qubits (solid circles within larger rings)

degrees of freedom that allow for the existence of multiple equivalent codespaces. The Bacon-Shor code can be constructed from an extended Shor code by effectively assembling a stabilizer group out of the $L_1 - 1$ "horizontal" and $L_2 - 1$ "vertical" 2-block stabilizers from both $|\pm\rangle_L = 2^{-L_1/2}[\bigotimes_{i=0}^{L_2-1}|0\rangle_i \pm \bigotimes_{i=0}^{L_2-1}|1\rangle_i]^{\otimes L_1}$ and $|q\rangle = 2^{-L_2/2}[\bigotimes_{i=0}^{L_1-1}|+\rangle_i + (-1)^q \bigotimes_{i=0}^{L_1-1}|-\rangle_i]^{\otimes L_2}$ ($q \in \{0,1\}$) bases [70]. The remaining Shor-code stabilizers from both bases then compose the group of gauge operators. Note that the Bacon-Shor X (Z) stabilizers are tensor products of neighboring X-logicals (Z-logicals) of the Shor code. In fact, the Bacon-Shor and (extended) Shor codes share the same logical operators. The connectivity of their physical qubits is represented by arranging them as the vertices of an $L_1 \times L_2$ rectangular lattices (cf. Figure 8 for the case $L_1 = L_2 = 3$). X (Z) stabilizers have supports on each pair of consecutive rows (columns) of the lattice. Likewise, logical X (Z) operations have support on any lattice row (column). These codes possess an additional group of gauge operators (maps between equivalent codespaces) formed by pairs of X or Z operators oriented along columns or rows of the lattice. The two most common logical bases of the Shor code, $|\pm\rangle_L$ and $|q\rangle_L$, correspond to easy-to-prepare Bacon-Shor logical bases (in a fixed gauge) and have their blocks encoded as rows or columns, respectively, of the lattice shown in Fig. 8.

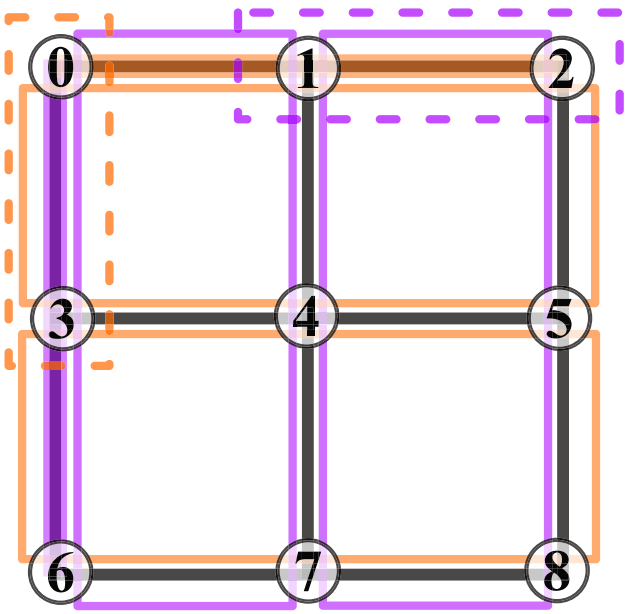

Fig. 8 Schematic representation of the [[9, 1, 3]] Bacon-Shor code. The lattice or grid (black lines) represents the connectivity of qubits (numbered vertices). X and Z stabilizers are represented as orange and purple solid rectangles, respectively. Two gauge operators are represented by dashed rectangles, with orange (purple) representing an X-type (Z-type) gauge. A specific choice of logical X and Z operators is also shown as orange horizontal and purple vertical lines (the other two parallel lines are equivalent for each direction). Since the X and Z logical operators intersect at exactly one qubit, one has $\{X, Z\} = 0$

5.2 Floquet Codes

The family of Floquet codes [36, 71–73] is composed of physical qubits connected as the vertices of a certain lattice, in which (usually periodic) low-weight parity-check measurements effectively compose dynamically generated stabilizers and logical states.

Consider qubits arranged on vertices of a hexagonal lattice with periodic boundary conditions (tessellating a torus) and three-colorable faces, similar to the one considered for the 2D color codes. One can label the lattice edges with the same color c of the faces they connect. We pictorially represent these colors as red, green and blue. We additionally label the edges $k = x, y, z$ according to their orientations along "southeast-northwest", "southwest-northeast" and "south-north", respectively (cf. Figure 9). For each edge, a so-called "check" (namely a parity-check operator $\hat{c}_{ij}^{(k)} = \sigma_{k,i} \otimes \sigma_{k,j}$ for neighboring qubits i and j) acts on its two qubits. The key idea behind the dynamical generation of an instantaneous stabilizer group with effective logical states is the cyclic and sequential measurement of checks of colors red, green and blue (some other orderings are also possible).

Starting the measurement routine with red check measurements applied on a prepared maximally mixed stated $\hat{\rho}_0$, one ends up with a collective eigenstate of all red checks. Note that each measurement can collapse the measured qubit pair into the check eigenstates of eigenvalues $+1$ or -1. In other words, the post-measurement state $\hat{\rho}_1^{\{\pm_{ij}\}_1} \propto [\prod_{e_{ij}^r}(1 \pm \hat{c}_{ij}^{(k)})]\hat{\rho}_0[\prod_{e_{ij}^k}(1 \pm \hat{c}_{ij}^{(k)})]$ is stabilized by either $\hat{c}_{ij}^{(k)}$ or $-\hat{c}_{ij}^{(k)}$ for each pair of qubits i, j belonging to a red edge, which we label e_{ij}^r (the measurement outcome for each pair is encoded in the set $\{\pm_{ij}\}_1$). Since we assume the first measurement round to consist of red checks, the corresponding (first) instantaneous stabilizer group is given by $\{\pm\hat{c}_{ij}^{(k)} | i, j \in e_{i,j}^r\}$ with signs corresponding to those in the set $\{\pm_{ij}\}_1$ and i, j chosen along red edges. A second measurement round consisting of green checks leads to the post-measurement states $\hat{\rho}_2^{\{\pm_{ij}\}_2} \propto [\prod_{e_{ij}^g}(1 \pm \hat{c}_{ij}^{(k')})]$ $[\prod_{e_{ij}^r}(1 \pm \hat{c}_{ij}^{(k)})]\hat{\rho}_0[\prod_{e_{ij}^r}(1 \pm \hat{c}_{ij}^{(k)})][\prod_{e_{ij}^g}(1 \pm \hat{c}_{ij}^{(k')})]$, with a larger set of measurement outcomes $\{\pm_{ij}\}_2$ including both rounds. It is important to notice that the cumulative projector $\frac{1}{4}[\prod_{e_{ij}^g}(1 + \hat{c}_{ij}^{(k')})][\prod_{e_{ij}^r}(1 + \hat{c}_{ij}^{(k)})] = [\prod_{e_{ij}^g}(1 + \hat{c}_{ij}^{(k')})]$ $[\prod_{p^r}(1 + \hat{B}_{p^r}^{(3)})]$ for red-colored plaquettes p^r defined on the corresponding hexagonal faces (with similar relations

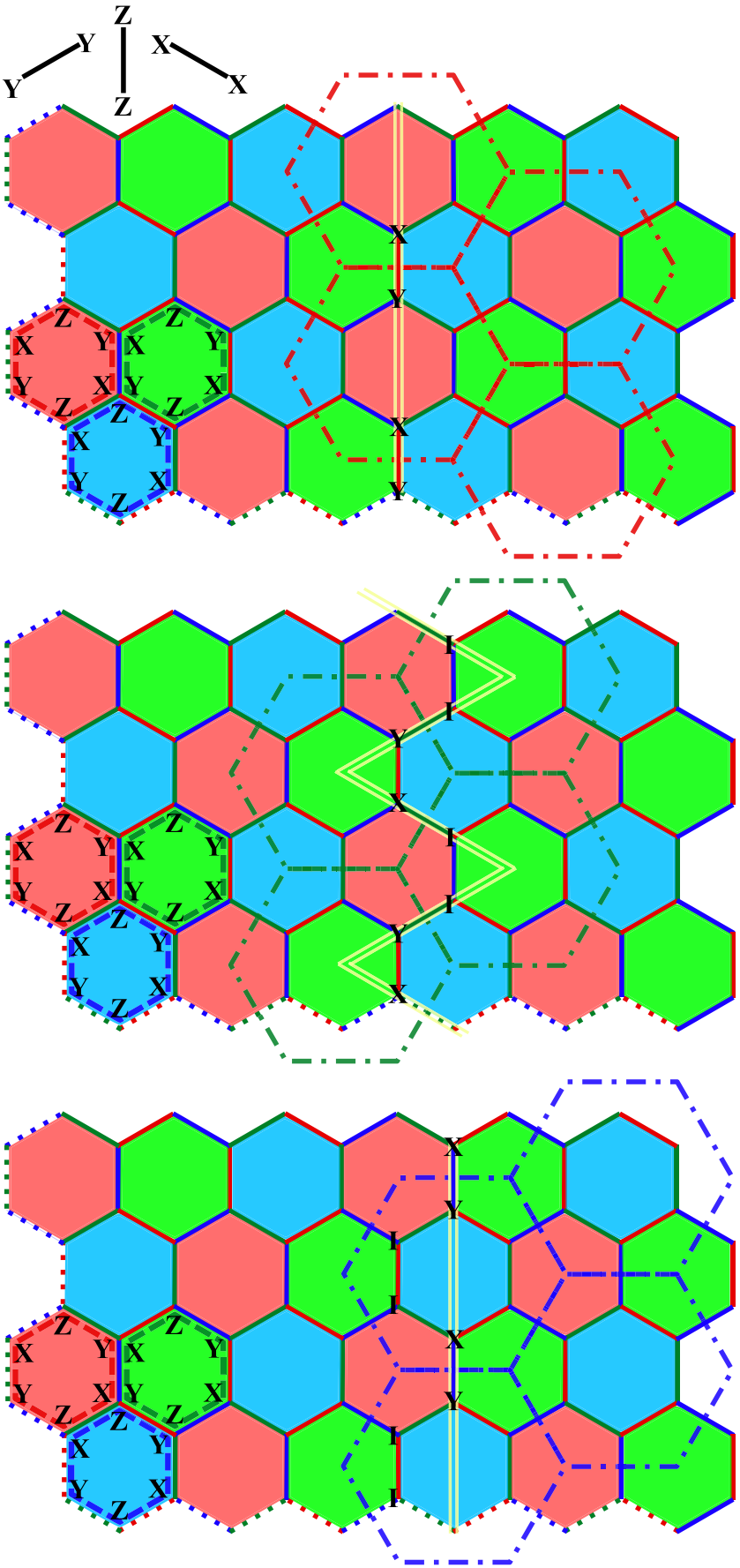

Fig. 9 Graphical representation of qubit connectivity in the 48-qubit Floquet code on a torus. Physical qubits are represented as vertices. Dotted edges on bottom and left sides of the lattice should be identified with top and right edges of same color and corresponding position (periodic boundary conditions). The red, green, and blue hexagons represent the support of each color of plaquette stabilizer, with the specific Pauli strings explicitly given on the bottom left corner (dashed colored hexagons). The dynamically generated toric-code superlattices spanned by red, green and blue checks are represented by dot-dashed hexagons of said colors. A logical operator is explicitly given by labeled qubits located on the yellow path

for other combinations of signs), so that the instantaneous stabilizer group is given by all green checks and all red plaquettes. The third round of blue-check measurements will likewise be stabilized by all blue checks and all red and green plaquettes (the former plaquette type commutes with the blue checks and the latter comes from the cumulative projections on green and then blue checks). Further measurement rounds will be stabilized by all three colors of plaquettes and by the specific check corresponding to the round.

A toric honeycomb lattice formed by $2n$ physical qubits contains n hexagonal faces (plaquettes) and n edges of each color (checks). One would therefore expect the instantaneous stabilizer group to be generated by $2n$ elements, checks and plaquettes. Nonetheless, the product of all plaquettes is the identity, and so is the case for the product of checks with plaquettes of the same color, so the instantaneous stabilizer group is actually generated by $2(n-1)$ elements. The difference between the numbers of physical qubits and constraints (stabilizer generators) therefore reveals that a Floquet code encodes 2 logical qubits. The distance of Floquet codes (or, more generally, dynamical codes, in case measurements are not necessarily periodic) is not easily defined [74].

A check measurement projects the state of the two involved qubits into a two-dimensional subspace (e.g., if the measurement gives the $+1$ eigenvalue, there are two corresponding eigenstates) [72]. One can therefore regard the qubit pair in the support of a check as a single effective qubit. This association becomes more concrete if one conjugates x- and z-type checks with CNOT unitaries and y checks with controlled-y unitaries, shrinking the supports of these checks down to single qubits (the choices of target and control are arbitrary) [36]. Assuming each such effective qubit of a given color (the color whose checks are in the instantaneous stabilizer group at a given measurement round, what we here call "active checks") is located (in terms of connectivity) on the center of a hexagonal-lattice edge, one obtains a superlattice that describes a hexagonal toric code, as shown in Fig. 9 for three measurement rounds: red, green and blue from top to bottom. The plaquettes of this dynamically generated toric code are centered at the same locations as the Floquet-code plaquettes whose colors correspond to the active checks. The Floquet-code plaquettes of the other two colors host 3-valent vertices of the dynamical toric code. The toric-code plaquette stabilizers are given by six operators $\widetilde{Z}_{ij}^{(k)} = 1_i \otimes \sigma_{k,j}$ (or $\sigma_{k,i} \otimes 1_j$) corresponding to the type k of each superlatice-plaquette edge, while the vertex stabilizers are given by three operators of the form $\widetilde{X}_{ij}^{(k)} = \epsilon_{kk'k''}\sigma_{k',i} \otimes \sigma_{k'',j}$, each associated with a k-type edge. Superlattice plaquettes and vertices are therefore violated by odd numbers of $\widetilde{X}_{ij}^{(k)}$ and $\widetilde{Z}_{ij}^{(k)}$ errors, respectively. The logical operators of the Floquet code are dynamically generated and correspond to longitudinal and meridional logical operators of the toric code. Since the logical operators commute with all instantaneous stabilizers, homologically non-trivial strings of $\widetilde{Z}_{ij}^{(k)}$ ($\widetilde{X}_{ij}^{(k)}$) errors run through the edges of active checks in such a way that they cross the vertices (plaquettes) of the superlattice similarly to the toric-code logical Z (X) operator, as shown in the middle (upper or lower) lattice in Fig. 9). An interesting feature of Floquet codes is the fact that the superlattice-toric-code logical operators going through plaquettes and vertices get mapped into each other with each new round of measurements of the Floquet code (i.e., magnetic and electric anyons get mapped into each other). To visualize this conversion, it is necessary to multiply the error string by a number of previous-round checks so that the new string commutes with all current-round active checks (see how the logical error changes between rounds in Fig. 9).

Errors in the Floquet code are detected via dynamical defects (plaquette violations). Since the actual measurements involve only pairs of qubits, the defect corresponding to a given plaquette is inferred by multiplying the outcomes from the two previous rounds of measurements of checks composing that plaquette. If in round r one measures red checks and in round $r+1$ measures green checks, one can infer at the end of the $(r+1)th$ measurement sequence the expectation value of the blue plaquettes. The whole process has a periodicity of 3 measurement rounds.

5.3 Quantum Computing and Error Correction with (Non-Abelian) Anyons

Non-Abelian topological codes generalize the toric and surface codes by allowing for the different flavours of defects (e.g., plaquette- or vertex-stabilizer violations in the toric code), commonly termed anyons, to possess a non-trivial braiding statistics (i.e., braiding a pair of anyons can change their flavours and generate superpositions of anyon types) [31, 32, 75]. In these systems, information can be encoded in the history of braidings and fusions of anyons, in the topologically non-trival loops of the substrate, or in both. Recent experiments were able to implement anyon-based quantum-information processing with information encoded in both the toric non-trivial loops and in a combination of anyons and surface-code states [76, 77].

The defects in the toric code can also be interpreted as quasiparticles known as anyons. The latter are characterized by exchange phases that can differ from the ones observed for either fermions on bosons. In fact, in the non-Abelian

case, exchanging or braiding anyons can also change the anyon type of the intermediate fusion outcomes. For the toric code, there are 3 non-trivial (yet Abelian) anyon types besides the vacuum (which we also count as an anyon type). These are the magnetic and electric anyons, corresponding respectively to defects on plaquettes and vertices, as well as the composite anyon formed by an electric and magnetic anyon pair. Not coincidentally, the number of anyon types in the toric code matches the degeneracy of its ground space.

It is known from topological quantum field theory that to every manifold one can associate a Hilbert space that depends solely on the manifold topology [78] (we restrict our analysis here to 2D manifolds). This Hilbert space corresponds to the code space, formed by ground states of the system if we consider anyons as excitations. That Hilbert space corresponding to the presence of particles on this manifold is the space corresponding to the manifold with labeled punctures representing the anyons. Therefore, besides the Hilbert space associated purely with the topology of the manifold, topological quantum field theories also associate (different) Hilbert spaces to that manifold in the presence of anyons (since punctures actually convert the original manifold into new ones). In other words, one can encode information in topological quantum systems through the degeneracy of its ground space, in the Hilbert space of anyon worldlines/histories that lead to a given collection of anyons (the so-called fusion trees), or in both!

Whenever two anyons come sufficiently close to be considered a single (composite) particle, one says they have fused. Labeling anyons by Greek letters, the fusion outcomes when bringing together α and β are usually represented by $\alpha \otimes \beta = \bigoplus_\gamma N^\gamma_{\alpha,\beta}\gamma$, in which the numbers $N^\gamma_{\alpha,\beta}$ are fusion multiplicities. The multiplicities represent the several ways in which fusing α and β can create an anyon γ, and the fusion relation runs over all possible outcomes. These multiplicies determine the dimension of the Hilbert space of a given topological system based on a certain manifold and collection of present anyons. For any pair of fusing anyons, if the multiplicity differs from zero for at least two outcomes or is larger than one for at least one outcome, we say that the theory is non-Abelian. The advantage of using such systems for quantum information processing, besides the fact that several such models allow for universal computation, is the fact that logical operations can only be performed in such systems by anyons with rather non-trivial worldlines, something that becomes increasingly more difficult in large systems where both winding around non-trivial loops in the manifold on braiding around other anyons involves long paths.

As an example, consider the Fibonacci-anyon model. The fusion relations are based on the coupling relations of irreducible representations of the group $SU(2)_3$ up to an isomorphism [79]. The two anyon types in this model are the vacuum (1) and the Fibonacci anyon (τ). These can fuse according to $1 \otimes 1 = 1$, $\tau \otimes 1 = 1 \otimes \tau = 1$, and $\tau \otimes \tau = 1 \oplus \tau$. Starting from three τ anyons, one can fuse any two of them (the closest ones), to get either the vacuum or another τ, and then fuse this with the furthest anyon to get either τ (in two possible ways) or the vacuum (in one way). These states can be represented as $|(\tau,\tau);\tau\rangle \otimes |(\tau,\tau);\tau\rangle$, $|(\tau,\tau);1\rangle \otimes |(1,\tau);\tau\rangle$, and $|(\tau,\tau);\tau\rangle \otimes |(\tau,\tau);1\rangle$, in which the two kets represent the two consecutive fusions, from left to right, and the first two entries in each ket represent the fusing anyon pair, while the entry after the semicolon represents the fusion outcome (note that the fusion outcome in the left pair is a fusing anyon in the right ket). Selecting the histories that ultimately fuse to a single Fibonacci anyon, one has a bidimensional Hilbert subspace. Since these two fusion histories can be superposed with each other by braiding the furthest of the three starting anyons with any of the other two (i.e., quantum gates are implemented by braiding), the two histories ending with τ are two states of a qubit. Starting with more anyons, one ends up with even larger computational spaces.

Quantum information processing with anyons has already been implemented in an effective manner: anyon-like excitations have been created and manipulated using lattices of, e.g., trapped ions [77] or superconducting circuits [76]. Besides the experimental challenges in working with anyons, not so many schemes for quantum error correction with non-Abelian anyons are known (due, e.g., to difficulties like the multiple error mechanisms involving pair creation, hopping, braiding, and fusion of anyons, the non-trivial fusion of anyons that does not always annihilate undesired anyon pairs into the vacuum, and the challenge in simulating quantum systems with universal-computing capabilities). The most common approach is to encode the quantum information into the degenerate ground states of a system and use quantum error correction to recombine anyon pairs created by noise before they move through a non-trivial path. The outcome of fusing a pair of nearby anyons, however, is not necessarily the vacuum, and measurements within predefined regions of the lattice (e.g., a plaquette) should determine the outcome of the fusion, so that the decoder can coordinate the best way to bring together fusion outcomes from increasingly further separated regions until they finally fuse into the vacuum [80–82].

6 Advances and Perspectives

The field of quantum error correction has seen incredible experimental progress in recent years. Some key achievements are the implementation of sub-threshold quantum error

correction protocols, with demonstration of the improved error resilience when scaling the system size [43], the fault-tolerant implementation of a code-switching protocol (i.e., converting the logical information between two codes) [83], the parallel implementation and manipulation of multiple logical states [47], and the preparation of high-fidelity magical logical states (states that enable the implementation of non-Clifford logical gates, a feature essential to achieving universal quantum information processing) [84–86]. On the theoretical front, besides designing new codes and decoders, recent efforts on fault-tolerant and/or measurement-free protocols that avoid the long measurement times and noise [37, 70, 87] are paving the way toward quantum-error-correction strategies free from the quantum-to-classical interface that makes quantum information processors dependent on auxiliary classical computers.

7 Conclusions

The field of quantum error correction is rapidly expanding and plays an ever growing role in current and future developments in quantum information processing. The widely accepted view that advantageous and meaningful quantum computations will only be possible at the logical level, once quantum-error-correction has fully been implemented, makes it indispensable for students, researchers and professors interested and/or working in quantum information processing to understand the key ideas underlying quantum error correction. This work aims at providing such readers with a sufficiently concise, direct and reachable introduction to quantum error correction. For the interested reader, a (non-exhaustive) list of highly relevant papers in the field is provided.

Acknowledgements This work was supported by the Simons Foundation (grant number 1023171, RC), the Brazilian National Council for Scientific and Technological Development (CNPq, grant number 315081/2025-2) and the Financiadora de Estudos e Projetos (grant 1699/24 IIP-FINEP).

Author Contributions The manuscript was entirely written by the author.

Funding The Article Processing Charge (APC) for the publication of this research was funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (ROR identifier: 00x0ma614).

Data Availability No datasets were generated or analysed during the current study.

Declarations

Competing interests The authors declare no competing interests.

References

1. R.P. Feynman, Simulating physics with computers. Int. J. Theor. Phys. **21**, 467 (1982)
2. Ю.И. Манин' Вычислимое и невычислимое' Советское радио, Moscow (1980). https://ncatlab.org/nlab/files/Manin-1980.pdf
3. P. Benioff, Quantum mechanical hamiltonian models of discrete processes. J. Math. Phys. **22**, 495 (1981)
4. D. Deutsch, Quantum theory, the church–turing principle and the universal quantum computer. Proceedings of the Royal Society of London. A Math. Phys. Sci. **400**, 97 (1985)
5. D. Deutsch, R. Jozsa, Rapid solution of problems by quantum computation. Proceedings of the Royal Society of London. Series A Math. Phys. Sci. **439**, 553 (1992)
6. P.W. Shor, Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual Symposium on Foundations of Computer Science, p. 124 (1994). Ieee
7. P.W. Shor, Scheme for reducing decoherence in quantum computer memory. Phys. Rev. A **52**, 2493 (1995). https://doi.org/10.1103/PhysRevA.52.R2493
8. S. Lloyd, A potentially realizable quantum computer. Science **261**, 1569 (1993)
9. S. Lloyd, Almost any quantum logic gate is universal. Phys. Rev. Lett. **75**, 346 (1995). https://doi.org/10.1103/PhysRevLett.75.346
10. S. Lloyd, Universal quantum simulators. Science **273**, 1073 (1996)
11. B. Schumacher, Quantum coding. Physical Review A. **51**, 2738 (1995). https://doi.org/10.1103/PhysRevA.51.2738
12. C.H. Bennett, Quantum information and computation. Phys. Today **48**, 24 (1995)
13. D.P. DiVincenzo, Two-bit gates are universal for quantum computation. Phys. Rev. A **51**, 1015 (1995). https://doi.org/10.1103/PhysRevA.51.1015
14. D.P. DiVincenzo, Quantum computation. Science **270**, 255 (1995)
15. A.Y. Kitaev, Quantum measurements and the abelian stabilizer problem (1995). arXiv preprint quant-ph/9511026
16. R. Landauer, Information is physical. Phys. Today **44**, 23 (1991)
17. J. Preskill, Quantum computing and the entanglement frontier (2012). arXiv preprint arXiv:1203.5813
18. S. Aaronson, A. Arkhipov, The computational complexity of linear optics. In: Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing, p. 333 (2011). https://dl.acm.org/doi/abs/10.1145/1993636.1993682
19. F. Arute, K. Arya, R. Babbush, D. Bacon, J.C. Bardin, R. Barends, R. Biswas, S. Boixo, F.G. Brandao, D.A. Buell et al., Quantum supremacy using a programmable superconducting processor. Nature **574**, 505 (2019)

20. A. Morvan, B. Villalonga, X. Mi, S. Mandrà, A. Bengtsson, P. Klimov, Z. Chen, S. Hong, C. Erickson, I. Drozdov et al., Phase transitions in random circuit sampling. Nature **634**, 328 (2024)
21. A.M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C.T. Hann, M.J. Kastoryano, E.T. Khabiboulline, A. Kubica, et al. Quantum algorithms: A survey of applications and end-to-end complexities (2023). arXiv preprint arXiv:2310.03011
22. K. Kraus, General state changes in quantum theory. Ann. Phys. **64**, 311 (1971)
23. B. Schumacher, Sending entanglement through noisy quantum channels. Phys. Rev. A **54**, 2614 (1996). https://doi.org/10.1103/PhysRevA.54.2614
24. S.J. Devitt, W.J. Munro, K. Nemoto, Quantum error correction for beginners. Rep. Prog. Phys. **76**, 076001 (2013)
25. A.R. Calderbank, P.W. Shor, Good quantum error-correcting codes exist. Phys. Rev. A **54**, 1098 (1996). https://doi.org/10.1103/PhysRevA.54.1098
26. A. Steane, Multiple-particle interference and quantum error correction. Proceedings of the Royal Society of London. Series A Math. Phys. Eng. Sci. **452**, 2551 (1996)
27. D. Bacon, Operator quantum error-correcting subsystems for self-correcting quantum memories. Phys. Rev. A **73**, 012340 (2006). https://doi.org/10.1103/PhysRevA.73.012340
28. A.Y. Kitaev, Quantum computations: algorithms and error correction. Russ. Math. Surv. **52**, 1191 (1997)
29. A.Y. Kitaev, Quantum error correction with imperfect gates. In: Proceedings of the 3rd International Conference of Quantum Communication and Measurement, p. 181. Plenum Press, New York (1997)
30. A.Y. Kitaev, Fault-tolerant quantum computation by anyons. Ann. Phys. **303**, 2 (2003)
31. C.G. Brell, S. Burton, G. Dauphinais, S.T. Flammia, D. Poulin, Thermalization, Error Correction, and Memory Lifetime for Ising Anyon Systems. Phys. Rev. X **4**, 031058 (2014). https://doi.org/10.1103/PhysRevX.4.031058
32. C. Nayak, S.H. Simon, A. Stern, M. Freedman, S. Das Sarma, Non-abelian anyons and topological quantum computation. Rev. Mod. Phys. **80**, 1083 (2008). https://doi.org/10.1103/RevModPhys.80.1083
33. A.R. Calderbank, E.M. Rains, P.M. Shor, N.J. Sloane, Quantum error correction via codes over gf (4). IEEE Trans. Inf. Theory **44**, 1369 (1998)
34. D.J. MacKay, G. Mitchison, P.L. McFadden, Sparse-graph codes for quantum error correction. IEEE Trans. Inf. Theory **50**, 2315 (2004)
35. M.B. Plenio, V. Vedral, P.L. Knight, Quantum error correction in the presence of spontaneous emission. Phys. Rev. A **55**, 67 (1997). https://doi.org/10.1103/PhysRevA.55.67
36. M.B. Hastings, J. Haah, Dynamically generated logical qubits. Quantum **5**, 564 (2021)
37. T.L.M. Guedes, D. Winter, M. Müller, Quantum cellular automata for quantum error correction and density classification. Phys. Rev. Lett. **133**, 150601 (2024). https://doi.org/10.1103/PhysRevLett.133.150601
38. D.J. MacKay, Information Theory Inference and Learning Algorithms. Cambridge University Press, Cambridge, (2003)
39. W.K. Wootters, W.H. Zurek, A single quantum cannot be cloned. Nature **299**, 802 (1982)
40. V.B. Braginsky, F.Y. Khalili, Quantum Measurement Cambridge University Press, Cambridge, (1995)
41. E. Knill, R. Laflamme, Theory of quantum error-correcting codes. Phys. Rev. A **55**, 900–911 (1997). https://doi.org/10.1103/PhysRevA.55.900
42. A.I. Google Quantum, Exponential suppression of bit or phase errors with cyclic error correction. Nature **595**, 383 (2021). https://doi.org/10.1038/s41586-021-03588-y
43. A.I. Google Quantum, Quantum error correction below the surface code threshold. Nature **638**, 920 (2025)
44. D. Bluvstein, et al. A quantum processor based on coherent transport of entangled atom arrays. Nature (London). **604**, 451 (2022). https://doi.org/10.1038/s41586-022-04592-6
45. S.J. Evered et al., High-fidelity parallel entangling gates on a neutral atom quantum computer. Nature **622**, 268 (2023)
46. H. Bernien et al., Probing many-body dynamics on a 51-atom quantum simulator. Nature **551**, 579 (2017). https://doi.org/10.1038/nature24622
47. D. Bluvstein, S.J. Evered, A.A. Geim, S.H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter et al., Logical quantum processor based on reconfigurable atom arrays. Nature **626**, 58 (2024)
48. T. Graham et al., Multi-qubit entanglement and algorithms on a neutral-atom quantum computer. Nature **604**, 457 (2022)
49. E.B. Jones, L.E. Hillberry, M.T. Jones, M. Fasihi, P. Roushan, Z. Jiang, A. Ho, C. Neill, E. Ostby, P. Graf et al., Small-world complex network generation on a digital quantum processor. Nat. Commun. **13**, 4483 (2022)
50. L. Postler, S. Heuβen, I. Pogorelov, M. Rispler, T. Feldker, M. Meth, C.D. Marciniak, R. Stricker, M. Ringbauer, R. Blatt, et al. Demonstration of fault-tolerant universal quantum gate operations. Nature. **605**, 675 (2022)
51. L. Van Hove, Sur l'intégrale de configuration pour les systèmes de particules à une dimension. Physica **16**, 137–143 (1950)
52. L. Landau, R. Peierls, Erweiterung des unbestimmtheitsprinzips für die relativistische quantentheorie. Z. Phys. **69**, 56 (1931)
53. N.D. Mermin, H. Wagner, Absence of ferromagnetism or antiferromagnetism in one- or two-dimensional isotropic heisenberg models. Phys. Rev. Lett. **17**, 1133 (1966). https://doi.org/10.1103/PhysRevLett.17.1133
54. P.C. Hohenberg, Existence of long-range order in one and two dimensions. Phys. Rev. **158**, 383 (1967). https://doi.org/10.1103/PhysRev.158.383
55.] E.H. Lieb, D.W. Robinson, The finite group velocity of quantum spin systems. Commun. Math. Phys. **28**, 251 (1972)
56. D. Aharonov, M. Ben-Or, Fault-tolerant quantum computation with constant error. In: Proceedings of the Twenty-ninth Annual ACM Symposium on Theory of Computing, p. 176 (1997)
57. E. Knill, R. Laflamme, W.H. Zurek, Resilient quantum computation. Science **279**, 342 (1998)
58. M.A. Nielsen, I.L. Chuang, Quantum Computation and Quantum Information Cambridge University Press, Cambridge, (2010)
59. D. Gottesman, Class of quantum error-correcting codes saturating the quantum hamming bound. Phys. Rev. A **54**, 1862 (1996). https://doi.org/10.1103/PhysRevA.54.1862
60. H. Bombín, An introduction to topological quantum codes (2013). arXiv preprint arXiv:1311.0277
61. D.S. Wang, A.G. Fowler, A.M. Stephens, L.C. Hollenberg, Threshold error rates for the toric and surface codes (2009). arXiv preprint arXiv:0905.0531
62. S.B. Bravyi, A.Y. Kitaev, Quantum codes on a lattice with boundary. arXiv preprint quant-ph/9811052. (1998)
63. D. Horsman, A.G. Fowler, S. Devitt, R. Van Meter, Surface code quantum computing by lattice surgery. New J. Phys. **14**, 123011 (2012)
64. O. Higgott, T.C. Bohdanowicz, A. Kubica, S.T. Flammia, E.T. Campbell, Improved decoding of circuit noise and fragile boundaries of tailored surface codes. Phys. Rev. X **13**, 031007 (2023). https://doi.org/10.1103/PhysRevX.13.031007
65. A.M. Steane, Error correcting codes in quantum theory. Phys. Rev. Lett. **77**, 793 (1996). https://doi.org/10.1103/PhysRevLett.77.793
66. H. Bombin, M.A. Martin-Delgado, Topological quantum distillation. Phys. Rev. Lett. **97**, 180501 (2006). https://doi.org/10.1103/PhysRevLett.97.180501

67. H. Bombin, M.A. Martin-Delgado, Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates. Phys. Rev. B **75**, 075103 (2007). https://doi.org/10.1103/PhysRevB.75.075103
68. D. Gottesman, Theory of fault-tolerant quantum computation. Phys. Rev. A **57**, 127 (1998). https://doi.org/10.1103/PhysRevA.57.127
69. D. Poulin, Stabilizer formalism for operator quantum error correction. Phys. Rev. Lett. **95**, 230504 (2005). https://doi.org/10.1103/PhysRevLett.95.230504
70. K. Bolsmann, T.L. Guedes, W. Li, J.W. Wilkinson, I. Lesanovsky, M. Müller, Fast native three-qubit gates and fault-tolerant quantum error correction with trapped rydberg ions (2025). arXiv preprint arXiv:2512.16641
71. D. Aasen, J. Haah, Z. Li, R.S. Mong, Measurement quantum cellular automata and anomalies in floquet codes (2023). arXiv preprint arXiv:2304.01277
72. T.D. Ellison, J. Sullivan, A. Dua, Floquet codes with a twist (2023). arXiv preprint arXiv:2306.08027
73. J.C. Fuente, J. Old, A. Townsend-Teague, M. Rispler, J. Eisert, M. Müller, XYZ ruby code: Making a case for a three-colored graphical calculus for quantum error correction in spacetime. Physical Review X Quantum. **6**, 010360 (2025). https://doi.org/10.1103/PRXQuantum.6.010360
74. X. Fu, D. Gottesman, Error correction in dynamical codes (2024). arXiv preprint arXiv:2403.04163
75. R. Koenig, G. Kuperberg, B.W. Reichardt, Quantum computation with turaev-viro codes. Ann. Phys. **325**, 2707 (2010)
76. A.I. Google Quantum, Non-abelian braiding of graph vertices in a superconducting processor. Nature **618**, 264 (2023)
77. M. Iqbal, N. Tantivasadakarn, R. Verresen, S.L. Campbell, J.M. Dreiling, C. Figgatt, J.P. Gaebler, J. Johansen, M. Mills, S.A. Moses et al., Non-abelian topological order and anyons on a trapped-ion processor. Nature **626**, 505 (2024)
78. S.H. Simon, Topological Quantum Oxford University Press, Oxford, (2023)
79. S. Trebst, M. Troyer, Z. Wang, A.W. Ludwig, A short introduction to fibonacci anyon models. Prog. Theor. Phys. Suppl. **176**, 384 (2008)
80. S. Burton, C.G. Brell, S.T. Flammia, Classical simulation of quantum error correction in a fibonacci anyon code. Phys. Rev. A **95**, 022309 (2017). https://doi.org/10.1103/PhysRevA.95.022309
81. C.G. Brell, S. Burton, G. Dauphinais, S.T. Flammia, D. Poulin, Thermalization, error correction, and memory lifetime for ising anyon systems. Phys. Rev. X **4**, 031058 (2014). https://doi.org/10.1103/PhysRevX.4.031058
82. J.R. Wootton, J. Burri, S. Iblisdir, D. Loss, Error correction for non-abelian topological quantum computation. Phys. Rev. X **4**, 011051 (2014). https://doi.org/10.1103/PhysRevX.4.011051
83. I. Pogorelov, F. Butt, L. Postler, C.D. Marciniak, P. Schindler, M. Müller, T. Monz, Experimental fault-tolerant code switching. Nat. Phys. **21**, 298 (2025)
84. S. Bravyi, A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas. Phys. Rev. A **71**, 022316 (2005). https://doi.org/10.1103/PhysRevA.71.022316
85. L. Daguerre, R. Blume-Kohout, N.C. Brown, D. Hayes, I.H. Kim, Experimental demonstration of high-fidelity logical magic states from code switching. Phys. Rev. X **15**, 041008 (2025). https://doi.org/10.1103/dck4-x9c2
86. P. Sales Rodriguez, J.M. Robinson, P.N. Jepsen, Z. He, C. Duckering, C. Zhao, K.-H. Wu, J. Campo, K. Bagnall, M. Kwon, et al. Experimental demonstration of logical magic state distillation. Nature, pp. 1–3 (2025)
87. S. Heußen, D.F. Locher, M. Müller, Measurement-free fault-tolerant quantum error correction in near-term devices. Phys. Rev. X Quantum **5**, 010333 (2024). https://doi.org/10.1103/PRXQuantum.5.010333