Introduction to Quantum Error Correction

Roberto Bondesan[1]

[1]Department of Computing, Imperial College London

1 Course info

Outline

Date	Topics
June 12	Elements of quantum computing; noise in quantum computing; Classical error correction;
June 13	Quantum stabiliser codes; Decoding problem
June 14	CSS codes; topological codes; statistical mechanical mapping

2 From classical to quantum codes

2.1 Elements of quantum computing

To motivate quantum error correction we first briefly review the basic idea of quantum computing. Quantum computers manipulate quantum information by applying a sequence of operations called gates. The quantum information is encoded in the state of n qubits. A qubit is a two-level quantum system, a superposition of bit 0 and 1:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\ . \tag{1}$$

Single qubits can be manipulated with unitary and measurement gates, graphically in figure 1. Im-

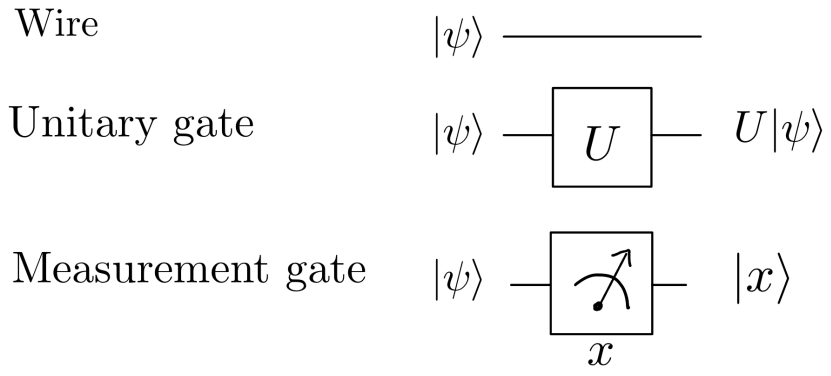


Figure 1:

portant single qubit unitary gates are Pauli matrices and Hadamard gate

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\ , \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\ , \quad Y = iXZ = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{2}$$

$$H = \frac{1}{\sqrt{2}}(X+Z) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{3}$$

Important to remember that Pauli matrices are Hermitian, square to 1 and we have relations

$$XZ = -ZX\,, \quad H^2 = \mathbf{1}_2\,, \quad HXH = Z\,, \quad HZH = X\,. \tag{4}$$

H thus maps between eigenstates of Z, $|0\rangle$, $|1\rangle$ and those of X, $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$:

$$H\,|0\rangle = |+\rangle\ , \quad H\,|1\rangle = |-\rangle\ . \tag{5}$$

Measurement gate is the only way to get classical information according to Born rule (readout bit x) from a qubit and while unitary are reversible, measurement is irreversible:

$$\alpha\,|0\rangle + \beta\,|1\rangle \to |x\rangle = \begin{cases} |0\rangle & \text{prob } |\alpha|^2 \\ |1\rangle & \text{prob } |\beta|^2 \end{cases}\ . \tag{6}$$

Measurement is a measurement of the Z observable.

A system of n qubits is a superposition of all possible bit strings of n bits:

$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x\,|x\rangle\ . \tag{7}$$

We draw the n qubits as wires and unitaries as boxes enclosing the wires they act non-trivially on, see figure 2. An important set of gates is the controlled operation, see figure 3. In particular is $U = X$

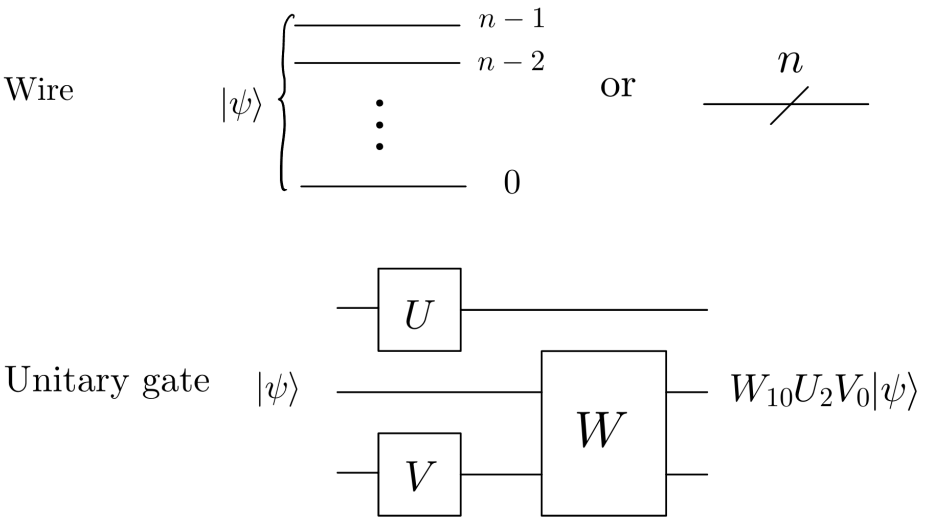


Figure 2:

the gate is called CNOT and plays a critical role in quantum computing. In a system of n qubits we can do partial measurement: Then system

$$|\psi\rangle = |0\rangle \otimes |\phi_0\rangle + |1\rangle \otimes |\phi_1\rangle\ , \tag{8}$$

gets projected onto $|\phi_x\rangle$ and properly normalised.

An important circuit in quantum error correction is that of figure 4. We say that the circuit 4 allows us to measure the operator U. Assume that U has eigenvalues ± 1, since it projects onto an eigenspace of U depending on the measurement outcome x:

$$|0\rangle\,|v\rangle \mapsto \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|v\rangle \tag{9}$$

$$\mapsto \frac{1}{\sqrt{2}}(|0\rangle\,\mathbf{1} + |1\rangle\,U)\,|v\rangle \tag{10}$$

$$\mapsto \frac{1}{2}((|0\rangle + |1\rangle)\,\mathbf{1} + (|0\rangle - |1\rangle)U)\,|v\rangle = (|0\rangle\,P_U^1 + |1\rangle\,P_U^{-1})\,|v\rangle \tag{11}$$

$$\mapsto |x\rangle\,P_U^{1-2x}\,|v\rangle \tag{12}$$

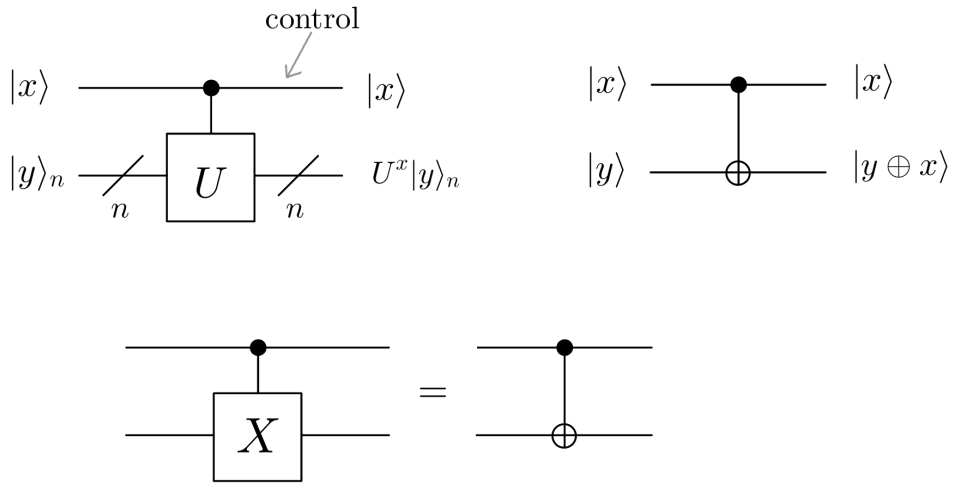
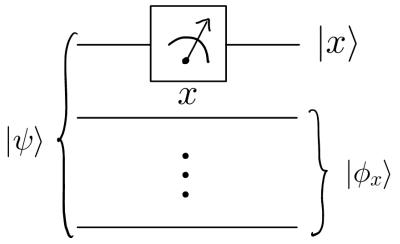


Figure 3:



with $P_U^{\pm 1}$ the projector onto the eigenspace of U with eigenvalue ± 1.

Quantum algorithm design aims at choosing quantum gates so that a given computational problem is solved more efficiently than with a classical computer. We will now delve here on any quantum algorithms but mention that there are rigorous speedups for problems such as quantum physics simulation, cryptography and searching among others.

2.2 Noise in quantum computing

Qubits are realised at the atomic scale, e.g. polarisation of photons, spin of an electrons, states of an atom, etc. This makes controlling qubits an extremely challenging problem and it is practically impossible to avoid errors. We call decoherence any noise that affects the system and leads to errors. Sources of noise are of two types:

1. Unwanted interactions among qubits and the environment. By environment we mean the system surrounding the quantum computer over which we have no control. This type of noise includes also unwanted interactions among qubits.

2. Imprecision in the experimental control, e.g. a wrong angle in the rotation of a qubit.

The simplest setting to study is that of a quantum memory when the goal is then to preserve a quantum state from being corrupted by noise for a certain amount of time t:

$$|\psi_0\rangle \otimes |\phi_E\rangle \to N\,|\psi_0\rangle \otimes |\phi_E\rangle\ . \tag{13}$$

In this case only the first source of errors is present. N is the unitary evolution to time t.

We can discuss this evolution as an operator on the qubits of the quantum computer by using the density matrix and partial trace. Recall that the density matrix of an ensemble of quantum states $\{|\psi_i\rangle, p_i\}$ is the positive semidefinite operator $\rho = \sum_i p_i\,|\psi_i\rangle\,\langle\psi_i|$ with trace 1. The density matrix of the system plus environment is

$$\rho_{SE} = N\rho_S(0) \otimes \rho_E(0)N^\dagger \tag{14}$$

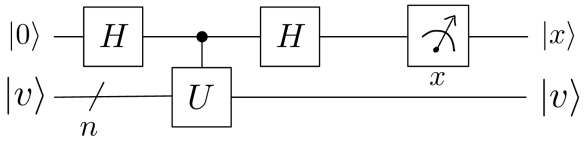


Figure 4:

where we have assumed more generally than both the system and the environment can be in a mixed state, but however at time zero they are not entangled. Let us denote by $\rho_E(0) = \sum_\nu \lambda_\nu |\nu\rangle\langle\nu|$, with $|\nu\rangle$ a basis of V_E the Hilbert space of the environment. Then we can trace out the environment to get

$$\rho_S = \mathrm{Tr}_E(N\rho_S(0)\otimes\rho_E(0)N^\dagger) \tag{15}$$

$$= \sum_{\mu,\nu} \lambda_\nu \langle\mu| N |\nu\rangle \rho_S(0) \langle\nu| N^\dagger |\mu\rangle \tag{16}$$

$$= \sum_a A_a \rho_S(0) A_a^\dagger\,, \quad A_{a=\mu\nu} = \sqrt{\lambda_\nu}\langle\mu| N |\nu\rangle\ . \tag{17}$$

The operators A_a are called Kraus operators and satisfy

$$\sum_a A_a^\dagger A_a = \sum_{\mu\nu} \lambda_\nu \langle\nu| N^\dagger |\mu\rangle \langle\mu| N |\nu\rangle = \sum_\nu \lambda_\nu = 1\,. \tag{18}$$

So we can model this type of noise by the map

$$\rho \mapsto \mathcal{E}(\rho) := \sum_a A_a \rho A_a^\dagger \tag{19}$$

which is completely positive (CP), in particular the output state is a valid density matrix as can be easily verified.

Now we can interpret $\mathcal{E}$ in terms of measurements as follows. Let us assume that the environment is in state $|0\rangle\langle 0|$ and suppose that after the unitary evolution N we perform a measurement of environment in the basis $|\mu\rangle$. Then the state of the system if outcome μ occurs is

$$\rho_\mu \propto \langle\mu| N(\rho\otimes|0\rangle\langle 0|)N^\dagger |\mu\rangle = \langle\mu| N |0\rangle \rho \langle 0| N^\dagger |\mu\rangle = A_\mu \rho A_\mu^\dagger \tag{20}$$

The normalised state is

$$\rho_\mu = \frac{A_\mu \rho A_\mu^\dagger}{p_\mu}\,, \quad p_\mu = \mathrm{Tr}\big(A_\mu \rho A_\mu^\dagger\big)\,. \tag{21}$$

Thus the quantum channel

$$\mathcal{E}(\rho) = \sum_\nu p_\mu \rho_\mu = \sum_\mu A_\mu \rho A_\mu^\dagger \tag{22}$$

is equivalent to randomly replacing ρ with ρ_ν with probability p_μ. We will also call this map a noisy quantum channel. The random evolution is thus a stochastic error model with error set $\{A_a\}$ and where errors occur $|\psi\rangle \mapsto A_a|\psi\rangle/\sqrt{p_a}$ with probability p_a.

2.2.1 Examples of noise channels

Some examples of single qubit noisy quantum channels important for the theory of quantum error correction are

- Bit flip channel: $A_0 = \sqrt{\epsilon}\,\mathbf{1}$, $A_1 = \sqrt{1-\epsilon}X$,

$$\mathcal{E}(\rho) = \epsilon\rho + (1-\epsilon)X\rho X\,. \tag{23}$$

  Here the qubit is flipped with probability $1-\epsilon$ and with probability ϵ nothing happens.

- Phase flip channel: $A_0 = \sqrt{\epsilon}\,\mathbf{1}$, $A_1 = \sqrt{1-\epsilon}Z$,

$$\mathcal{E}(\rho) = \epsilon\rho + (1-\epsilon)Z\rho Z\,. \tag{24}$$

  Here the qubit has its phase flipped with probability $1-\epsilon$ and with probability ϵ nothing happens.

- Depolarizing channel: $A_0 = \sqrt{1-\epsilon}\,\mathbf{1}$, $A_1 = \sqrt{\frac{\epsilon}{3}}X$, $A_2 = \sqrt{\frac{\epsilon}{3}}Y$, $A_3 = \sqrt{\frac{\epsilon}{3}}Z$:

$$\mathcal{E}(\rho) = (1-\epsilon)\rho + \frac{\epsilon}{3}(X\rho X + Y\rho Y + Z\rho Z)\,. \tag{25}$$

  Here X, Y, Z are applied with probability $\epsilon/3$, and with probability $1-\epsilon$ nothing happens.

All these noise channels are of the form Pauli channels:

$$\mathcal{E}(\rho) = \sum_{E\in\mathcal{P}} p(E)E\rho E^\dagger \tag{26}$$

where $\mathcal{P}$ is the set of all tensor products of Pauli matrices.

2.3 Classical error correction

We start to discuss how to solve the error correction problem for classical systems. Our approach would be then to generalise classical error correction to the quantum case, which is what happened historically as well. The quantum formalism reduces to the classical one by considering a diagonal density matrix in the computational basis. In this case, the diagonal element $p_X(x)$ is the probability distribution of the random variable X to take on the value x. x is a bit-string and is convenient to see it as a vector in $\mathbb{F}_2^n$. $\mathbb{F}_2$ is the binary field of two elements, $0, 1$ with addition modulo 2: $0+0 = 1+1 = 0$, $1+0 = 0+1 = 1$. The vector space $\mathbb{F}_2^n$ is thus the space of all tuples $x = (x_1, \ldots, x_n)$ with $x_i \in \mathbb{F}_2$ and addition of two vectors is done modulo 2. The theory of error correcting codes can be developed for general fields, but for simplicity and for our purposes of inspiring the theory of quantum error correction for qubits we will stick to the binary field.

The quantum channels that map $p_X(x)$ to $p_Y(y)$, with y a corrupted bit-string, are the transition matrices or conditional distributions $p_{Y|X}(y|x)$. Classical computers are practically very resilient to noise since bits are encoded in the collective state of of thousands of atoms where interactions with the environment play a much less important role. Classical error correction is instead central to communication theory. The channel then represents a noisy communication channel, e.g. a phone line. The sender uses this channel to communicate a message x to the receiver y.

2.3.1 Classical error correcting codes

The basic idea behind solving this problem is to send many copies of the message so that the receiver has a higher chance of identifying the message despite the noise that corrupts it. This is done using an error correction code that encodes the original message using redundancy. This encoded message is sent through the noisy channel and then decoded at the receiver side. An error correcting code $\mathcal{C}(n, M)$ encodes the messages $\{1, \ldots, M\}$ as elements $x^{[1]}, \ldots, x^{[M]}$ of $\mathbb{F}_2^n$. The number of bits used n is called the block length. Important properties of a code are the rate

$$r = \frac{1}{n}\log_2(M)\,, \tag{27}$$

which is the ratio between the number of encoded bits and the block length, and the distance:

$$d = \min\{d(u,v)\,|\,u, v \in \mathcal{C}(n,M)\,, u \neq v\}\,, \tag{28}$$

where $d(u,v)$ is the Hamming distance of the two codewords u, v, namely it counts the number of non-zero elements of $u+v$.

For a code of distance d, if $d(x,y)$, the distance between the sent message x and the received one y, is smaller than t with $t = \lfloor\frac{d-1}{2}\rfloor$, the receiver would be able to decode the message as we now show. Let us define a sphere of radius t around a bitstring u to be the set of all points with distance at most t from u. Then if the code has distance d, the spheres of radius $t = \lfloor\frac{d-1}{2}\rfloor$ around codewords are

disjoint. This can be seen by noting that if x in the sphere of radius t centered at u, then for another codeword v, by triangle inequality

$$d \leq d(u,v) \leq d(u,x) + d(x,v) \tag{29}$$

and so

$$d(x,v) \geq d - d(u,x) \geq \frac{d+1}{2}\,, \tag{30}$$

so x cannot be in the sphere centered at v of radius t and such spheres are disjoing. But now if we are guaranteed that the distance between the sent x and the received message y is smaller than t, the receiver can simply return the unique codeword $x \in \mathcal{C}$ such that $d(x,y) \leq t$ and succesfully correct for the error. This motivates constructing codes with large distance.

2.3.2 MAP decoding

In practice it is difficult to guarantee that the distance between sent and received message is $\leq t$. The receiver will then use a decoder that returns $\hat{x}(y) \in \mathcal{C}$ given a received message y. The probability of making an error in decoding is

$$\mathbb{P}(\hat{x}(y) \neq x) = 1 - p_{X|Y}(\hat{x}(y)|y) \tag{31}$$

The optimal decoding strategy would be then to minimise the error probability:

$$\hat{x}^{\mathrm{MAP}}(y) = \mathrm{argmax}_{x\in\mathcal{C}} p_{X|Y}(x|y) = \mathrm{argmax}_{x\in\mathcal{C}} p_{Y|X}(y|x) p_X(x) \tag{32}$$

by Bayes rule. MAP stands for maximum a posteriori since the probability maximised is the posterior given a prior p_X. The maximum likelihood decoder

$$\hat{x}^{\mathrm{ML}}(y) = \mathrm{argmax}_{x\in\mathcal{C}} p_{Y|X}(y|x) \tag{33}$$

coincides with the MAP decoder if p_X is uniform.

2.3.3 Binary symmetric channel and repetition code

Let us now see an example to illustrate these ideas. Let us fix the binary symmetric channel as our noise, BSC(ϵ). This flips a single bit x with probability ϵ and leaves it unchanged with probability $1-\epsilon$. Let us assume uniform prior p_X. If uncoded, the MAP or ML decoder will maximise the likelihood over x:

$$p_{Y|X}(y|x) = \epsilon^{1-\delta_{y,x}} (1-\epsilon)^{\delta_{y,x}}\,. \tag{34}$$

We can table this probability as function of $x, y \in \{0,1\}$:

$$\begin{pmatrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{pmatrix}. \tag{35}$$

If $\epsilon < \frac{1}{2}$, $1-\epsilon > \epsilon$ so $\hat{x}^{\mathrm{MAP}}(y) = y$ is the optimal decoder and the probability of an error is ϵ. Now let us consider the repetition code that repeats the input bit n times. So the input to the BSC(ϵ) is the tuple $(x,\ldots,x)$ and if the channel acts independently on each copy, the MAP decoder maximises, noting that the number of flips is $d(x,y)$,

$$p_{Y|X}(y|x) = \epsilon^{d(x,y)} (1-\epsilon)^{n-d(x,y)}\,, \tag{36}$$

Since $\epsilon < \frac{1}{2}$, $\log(\epsilon/(1-\epsilon)) < 0$ and by taking the log

$$\mathrm{argmax}_x d(x,y)\log(\epsilon) + (n-d(x,y))\log(1-\epsilon) = \mathrm{argmax}_x d(x,y)\log(\epsilon/(1-\epsilon)) = \mathrm{argmin}_x d(x,y)\,, \tag{37}$$

so finds minimum distance. Now

$$d(0^n, y) = \#(1s \text{ in } y)\,, \quad d(1^n, y) = \#(0s \text{ in } y) \tag{38}$$

so if there are more 0s in y, then we should decode as 0^n, and similarly if more 1s as 1^n, so that $\hat{x}^{\mathrm{MAP}}(y)$ is the majority vote of $y_1, \ldots, y_n$. Assuming n odd, the error probability is the probability that at least $\lceil n/2 \rceil$ flips occur, which is

$$P_{\mathrm{err}} = \sum_{i > n/2} \binom{n}{i} \epsilon^i (1-\epsilon)^{n-i} \,. \tag{39}$$

For example, if $n = 3$, this is

$$3\epsilon^2(1-\epsilon) + \epsilon^3 = 3\epsilon^2 - 2\epsilon^3 \tag{40}$$

which is less than ϵ if $\epsilon < 1/2$. As $n \to \infty$, the error probability tends to 0 but so does the rate which is $1/n$.

2.3.4 Linear codes

Now we discuss the framework of linear codes which heavily generalises the repetition code and allow us to construct codes with better properties. A classical linear code is a classical code that is a closed linear subspace of $\mathbb{F}_2^n$. This means that is if $u, v \in \mathcal{C}$, then $u + v \in \mathcal{C}$. If we denote the dimension of the subspace k then $\mathcal{C} \simeq \mathbb{F}_2^k$. We can arrange a basis of $\mathcal{C}$ in a generator matrix of size $k \times n$ so that we can span all the codewords by taking $u \in \mathbb{F}_2^k$ and encode it as $x = uG$. Note convention of x as row vector. The distance of a linear code is the Hamming weight $w(u)$ of the smallest non-zero codeword u, since

$$\min\{d(u,v)\,, u, v \in \mathcal{C}\,, u \neq v\} = \min\{d(u-v,0)\,, u, v \in \mathcal{C}\,, u \neq v\} = \min\{w(u)\,, u \in \mathcal{C}\,, u \neq 0\} \tag{41}$$

We denote by $[n, k, d]$ a classical linear code with block length n, k logical bits and distance d.

Dual code We call the dual code of $\mathcal{C}$ the set of bitstrings orthogonal to the codewords of $\mathcal{C}$. This means

$$\mathcal{C}^\perp = \{x \in \mathbb{F}_2^n \,|\, Gx^T = 0^T\} \,. \tag{42}$$

Note that if $u, v \in \mathcal{C}^\perp$, then $u + v \in \mathcal{C}^\perp$ so that $\mathcal{C}^\perp$ is a linear code as well. We call its generator matrix the parity check matrix H. Since G has k linearly independent rows, by the rank-nullity theorem we know that the kernel of G has dimension $n - k$ so that H is an $n - k \times n$ matrix with $n - k$ linearly independent rows. Therefore we can encode $n - k$ bits in the dual code as $x = uH$, $u \in \mathbb{F}_2^{n-k}$ and

$$GH^T = 0_{k \times n-k} \,, \tag{43}$$

or $HG^T = 0$, and we can characterise $\mathcal{C}$ as

$$\mathcal{C} = \{x \in \mathbb{F}_2^n \,|\, Hx^T = 0^T\} \,, \tag{44}$$

since if $x = uG$ then $Hx^T = HG^T u^T = 0$.

Decoding Assuming bit flip noise, MAP decoding in linear codes can be written as

$$\begin{aligned} \hat{x}^{\mathrm{MAP}}(y) &= \mathrm{argmax}_{x \,|\, Hx^T = 0^T} p_{Y|X}(y|x) && (45)\\ &= \mathrm{argmax}_{e \,|\, He^T = Hy^T} p_{Y|X}(y|x) + y && (46)\\ &= \mathrm{argmax}_{e \,|\, He^T = \sigma} p_{Y|X}(y|x) + y && (47) \end{aligned}$$

where we denoted $e = x + y$ which represent the number of bit flips. We call $\sigma = Hy^T \in \mathbb{F}_2^{n-k}$ the error syndrome, which is known to the receiver.

In the case of the BSC(ϵ), where $p_{Y|X}(y|x) = \epsilon^{d(x,y)}(1-\epsilon)^{n-d(x,y)}$, $\epsilon < \frac{1}{2}$, this can be further simplified as

$$\hat{x}^{\mathrm{MAP}}(y) = \mathrm{argmin}_{e \,|\, He^T = \sigma} w(e) + y \,, \tag{48}$$

namely the task is to find the minimum weight bitstring with the syndrome σ.

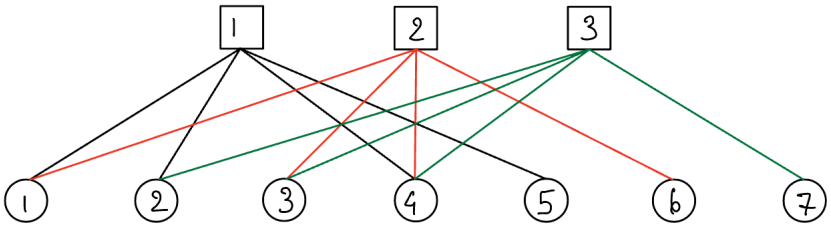


Figure 5:

Examples We have already seen an example of linear code in the repetition code. Its generator and parity check matrices are

$$G = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \end{pmatrix} \tag{49}$$

In fact we can see that $Hx^T = 0^T$ means that $x_1 + x_2 = 0, \ldots, x_{n-1} + x_n = 0$ so that $x_1 = \cdots = x_n$. The repetition code is a $[n, 1, n]$ code.

Another example of linear code is the $[7, 4, 3]$ Hamming code. Its generator and parity check matrix are

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{50}$$

Note that for the Hamming code, $\mathcal{C}^\perp \subset \mathcal{C}$. This can be seen by noting that here $k = 4, n - k = 3$ and if $x = uH$ for $u \in \mathbb{F}_2^3$ is in $\mathcal{C}^\perp$ then $Hx^T = 0^T$ since $HH^T = 0$ as can be easily verified. Also, the Hamming code can correct perfectly single bit flip errors. This is because if we apply He with e that has a single non-zero entry, one gets one of the columns of H. Noting that all the columns are distinct, if we know the syndrome associated to a single bit flip, we can uniquely determine which error occurred. From this we deduce that $t = 1$ so that $1 = (3 - 1)/2$ and $d = 3$.

Tanner graph A useful representation of a linear code is in terms of the so-called Tanner graph which is a bipartite graph with two sets of vertices: C check vertices associated to each row of H and variable vertices associated with columns of H. The Tanner graph has an edge between cv, $c \in C, v \in V$ if $H_{cv} = 1$. Figure 5 shows an example for the Hamming code.

2.4 Quantum codes

2.4.1 Challenges of quantum error correction

As first attempt at constructing a quantum error correction procedure we can try to use the same idea behind the simplest classical code, the repetition code. Suppose we have a state $|\psi\rangle$ that we want to send through a quantum channel that describes a noise process. The first step we can try is copy this state k times. It is however a fundamental fact in quantum computing that we cannot clone an arbitrary quantum state. The no cloning theorem can be proved by contradiction. Consider two arbitrary quantum states $|\psi\rangle, |\phi\rangle$ of n qubits. The existence of a cloning map C means that

$$C|\psi\rangle|0^n\rangle = |\psi\rangle|\psi\rangle, \quad C|\phi\rangle|0^n\rangle = |\phi\rangle|\phi\rangle. \tag{51}$$

Since C is a quantum map, it is linear, so that for any $\alpha, \beta \in \mathbb{C}$:

$$C(\alpha|\psi\rangle + \beta|\phi\rangle)|0^n\rangle = \alpha C|\psi\rangle|0^n\rangle + \beta C|\phi\rangle|0^n\rangle = \alpha|\psi\rangle|\psi\rangle + \beta|\phi\rangle|\phi\rangle. \tag{52}$$

On the other hand,

$$C(\alpha|\psi\rangle + \beta|\phi\rangle)|0^n\rangle = (\alpha|\psi\rangle + \beta|\phi\rangle)(\alpha|\psi\rangle + \beta|\phi\rangle) = \alpha^2|\psi\rangle|\psi\rangle + \alpha\beta(|\psi\rangle|\phi\rangle + |\phi\rangle|\psi\rangle) + \beta^2|\phi\rangle|\phi\rangle \tag{53}$$

These two expressions are different unless $\alpha\beta = 0$, so either $\alpha = 0$ or $\beta = 0$ and we have proved that we cannot close an arbitrary state. Of course, this does note mean that if we know what state we have prepared, we cannot prepare it again on another register.

Another conceptual challenge in quantum error correction is that we cannot inspect what state the receiver gets since any measurement would destroy the coherent superposition. Knowledge of the received bit string y was crucial classically for example to compute the error syndrome and this seems also another insurmountable problem.

A final challenge is that classically errors are essentially only bit flips if we assume that the receiver receives a bit string, which form a discrete set of possible errors. Quantumly there is instead a continuous set of possible quantum operations we can perform on a qubit, seemingly leading to needing infinite resources to correct those.

Fortunately, all these challenges can be solved. We here present explicit constructions of the 3 and the 9 qubit codes. The former corrects only bit flip errors but shows how we can effectively solve all the other problems outlined above. Shor's 9 qubit error correction code will allow us to correct arbitrary single qubit errors. We will then generalise these ideas to a large class of quantum codes.

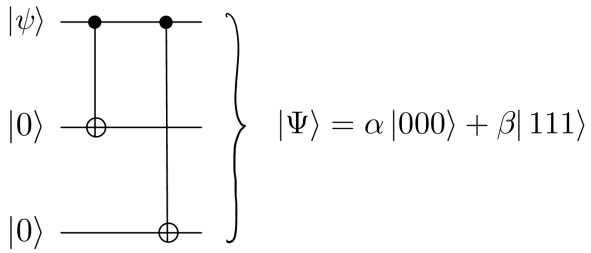
3 qubit code Let us assume that we have a system of 3 qubits and our noise model acts on at most a single qubit with bit flip X. The most general CP map describing this noise is

$$\mathcal{E}(\rho) = \sum_a A_a \rho A_a^\dagger\,, \quad A_a = \alpha_a^0 \mathbf{1} + \sum_{i=1}^n \alpha_a^i X_i\,, \tag{54}$$

for some $\alpha_a^i \in \mathbb{C}$. The 3 qubit code encodes a logical qubit into the Hilbert space of 3 qubits as follows:

$$|\bar{0}\rangle = |000\rangle\,, \quad |\bar{1}\rangle = |111\rangle\,. \tag{55}$$

Here we denote $\bar{0}, \bar{1}$ the logical states as opposed to physical states. We imagine that we have encoded an arbitrary single qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ onto $|\Psi\rangle = \alpha|\bar{0}\rangle + \beta|\bar{1}\rangle$. This can be accomplished by the encoding circuit in the figure 2.4.1.



Then we assume that noise of (54) acts on the system, leading to a corrupt state ρ. We now claim that performing the following operations of 6 allows us to recover the original state.

We see that we repeat the measurement circuit 4 twice, once with $U = Z_1Z_2$ and once with $U = Z_2Z_3$. Recall that this circuit projects onto the ± 1 eigenstate of U depending on the measurement outcome. Then the circuit of figure 2.4.1 has the following action:

$$\mathcal{E}(\rho) \otimes |0\rangle\langle 0| \otimes |0\rangle\langle 0| \mapsto \Pi_{xy}\mathcal{E}(\rho)\Pi_{xy} \otimes |x\rangle\langle x| \otimes |y\rangle\langle y|\,, \quad \Pi_{xy} = P_{Z_1Z_2}^{1-2x} P_{Z_2Z_3}^{1-2y} \tag{56}$$

Note that Z_1Z_2 commutes with Z_2Z_3 so they can be simultaneously diagonalised. What are the eigenspaces of Z_1Z_2, Z_2Z_3? First we note that the codes space is the $+1$ eigenspace:

$$Z_1Z_2|000\rangle = Z_2Z_3|000\rangle = |000\rangle \tag{57}$$

and similarly for $|111\rangle$. Single qubit bit flips can be distinguished by the eigenvalues of them:

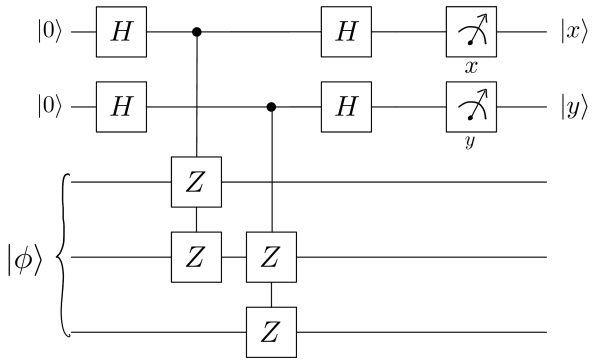


Figure 6:

	$\|\Psi\rangle$	$X_1\|\Psi\rangle$	$X_2\|\Psi\rangle$	$X_3\|\Psi\rangle$
Z_1Z_2	$+1$	-1	-1	$+1$
Z_2Z_3	$+1$	$+1$	-1	-1

This shows that the Hilbert space of 3 qubits decomposes into 2 dimensional spaces distinguished by the measurement outcomes:

$$(\mathbb{C}^2)^{\otimes 3} = \bigoplus_{x,y=0}^{1} \mathcal{H}_{xy}\,, \quad \mathcal{H}_{xy} \simeq \mathbb{C}^2\,. \tag{58}$$

We call the operators Z_1Z_2, Z_2Z_3 stabilisers since they stabilise the code space and the measurement outcome x, y the measurement syndrome since by analogy of the classical syndrome allows us to detect the error.

Now let us suppose that we measure $x = y = 0$. Noting that

$$\Pi_{00}X_i\,|\Psi\rangle = 0 \tag{59}$$

we see that

$$\Pi_{00}\mathcal{E}(|\Psi\rangle\langle\Psi|)\Pi_{00} = |\Psi\rangle\langle\Psi|\,, \tag{60}$$

so only the terms corresponding to the identity, ie no noise, are retained. Similary, since

$$\Pi_{01}X_1\,|\Psi\rangle = \Pi_{01}X_2\,|\Psi\rangle = 0\,, \quad \Pi_{01}X_3\,|\Psi\rangle = X_3\,|\Psi\rangle \tag{61}$$

if we measure $xy = 01$ only the terms with X_3 error survive.

$$\Pi_{01}\mathcal{E}(|\Psi\rangle\langle\Psi|)\Pi_{01} = X_3\,|\Psi\rangle\langle\Psi|\,X_3\,, \tag{62}$$

We can proceed similarly for $10, 11$ syndromes:

$$\Pi_{10}\mathcal{E}(|\Psi\rangle\langle\Psi|)\Pi_{10} = X_1\,|\Psi\rangle\langle\Psi|\,X_1\,, \tag{63}$$
$$\Pi_{11}\mathcal{E}(|\Psi\rangle\langle\Psi|)\Pi_{11} = X_2\,|\Psi\rangle\langle\Psi|\,X_2\,, \tag{64}$$

so that the procedure described allow us to detect what is the final state and what error occurred. If we wanted to recover the original state we can undo the bit flip by applying the appropriate bit flip operator. Application of this operator will however come with its faults and thus in practice it is better to account for the noise as postprocessing rather than active recovery.

This simple example of the 3 qubit code has shown us that we can actually solve the challenges of quantum error correction. We can circumvent no cloning by employing the encoding circuit described. We can avoid measuring the state to compute the syndrome by measuring ancilla qubits that allow us to perform projective measurements that do not destroy the coherence of the state. And finally, we do not need to worry about the fact that quantum errors form a continuous space. Performing the

projective measurement of the stabilisers projected us into a single summand of the quantum channel, which means that in practice we can consider the errors as if they were discrete. This important result is called the discretisation of errors and more generally means that if we can correct a discrete set of errors $\{E_i\}$ we will be able to correct any error formed by linear superpositions of those. We will discuss this further below. Note that the logical Pauli operators are

$$\bar{X} = X_1 X_2 X_3 \,, \quad \bar{Z} = Z_1 Z_2 Z_3 \tag{65}$$

since they act as X, Z on the logical qubits. These are not unique. For example $\bar{Z} = Z_1$ would also be a valid logical Z since it is related to the one above by the action of the stabiliser $Z_2 Z_3$ that acts trivially on the codes space.

9 qubit code We now drop the assumption that only bit flips are allowed, and consider a quantum error correction code that corrects arbitrary single qubit errors. The noise can be described as

$$\mathcal{E}(\rho) = \sum_a A_a \rho A_a^\dagger \,, \quad A_a = \alpha_a^0 \, \mathbf{1} + \sum_{i=1}^n \alpha_a^i X_i + \beta_a^i Y_i + \gamma_a^i Z_i \,, \tag{66}$$

We encode a logical qubit in the two states of 9 qubits

$$|\bar{0}\rangle = 2^{-3/2}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \,, \tag{67}$$

$$|\bar{1}\rangle = 2^{-3/2}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle) \,, \tag{68}$$

We can see this code as the concatenation of two codes. An inner code that is the 3 qubit bit flip code to protect from bit flip errors, which is built by the codewords $|000\rangle, |111\rangle$. An outer code that repeats the logical $\pm$ states of the inner code $|000\rangle \pm |111\rangle$. This outer code protects from phase flip errors since bit and phase flip errors are related by the Hadamard transform and so are the $0, 1$ and $\pm$ states. We now show that this intuition is correct. The stabilisers of this code are

$$S_1^Z = Z_1 Z_2 \quad S_2^Z = Z_2 Z_3 \tag{69}$$

$$S_3^Z = Z_4 Z_5 \quad S_4^Z = Z_5 Z_6 \tag{70}$$

$$S_5^Z = Z_7 Z_8 \quad S_6^Z = Z_8 Z_9 \tag{71}$$

$$S_1^X = X_1 X_2 X_3 X_4 X_5 X_6 \quad S_2^X = X_4 X_5 X_6 X_7 X_8 X_9 \,. \tag{72}$$

The Z ones are clear from the construction using three 3 qubit codes. The X ones are $\bar{X}_1 \bar{X}_2$ and $\bar{X}_2 \bar{X}_2$, the stabiliser of the outer code if $\bar{X}_i$ is the logical X of the i-th inner code.

Note that they all commute: recalling $XZ = -ZX$, each Z stabiliser and X stabiliser act on a common set of qubits that is even.

Now we show that arbitrary single qubit errors acting on an encoded state $|\Psi\rangle$ are also eigenstates of the stabilisers. First note that

$$Z_1 |\Psi\rangle = Z_2 |\Psi\rangle = Z_3 |\Psi\rangle \tag{73}$$

$$Z_4 |\Psi\rangle = Z_5 |\Psi\rangle = Z_6 |\Psi\rangle \tag{74}$$

$$Z_7 |\Psi\rangle = Z_8 |\Psi\rangle = Z_9 |\Psi\rangle \,. \tag{75}$$

so we can consider only the effect of Z_1, Z_4, Z_7 instead of all Z errors. Z errors can be distinguished from X, Y errors since they commute with all S_i^Z. Further $\{Z_1, S_1^X\} = [Z_1, S_2^X] = 0$, $\{Z_4, S_1^X\} = \{Z_4, S_2^X\} = 0$, $[Z_7, S_1^X] = \{Z_7, S_2^X\} = 0$, so they can be all distinguished. X errors can be distinguished from Z, Y error since they commute with S_i^X. $X_1, X_3, X_4, X_6, X_7, X_9$ anticommute with a single and different S_i^Z, while X_2, X_5, X_8 anticommute with two distinct S_i^Z. Finally Y errors have same anticommutation with S_i^Z so they can be distinguished. They can be distinguished from X errors because they anticommute with at least one S_i^X.

If we now measure the stabilisers - using a circuit similar to that of the 3 qubit code - the measurement projects onto a different single qubit error subspace which can be unequivocally identified and recovered from.

2.4.2 Error correction criteria

What errors can we correct with a quantum code? This question can be answered by using the Knill-Laflamme quantum error correction criterion which state that a set of errors $\{E_a\}$ is correctable if and only if

$$\langle\phi_i| E_a^\dagger E_b |\phi_j\rangle = C_{ab}\delta_{ij} \tag{76}$$

for all a, b and i, j where $|\phi_i\rangle$ is a basis of the code space and C_{ab} is Hermitian. Intuitively, for $i \neq j$ this means that errors should not distort codewords so that they cannot be distinguished anymore after the error occurs. Define the weight of a Pauli operator as the number of qubits on which a Pauli operator acts non-trivially. We say that a quantum error correcting code corrects t errors if the set of E_a's that allow recovery has weight t or less. Let us define now the distance of a code by the minimum weight of a non-trivial logical operator. This means that codewords are not orthogonal anymore and the Pauli error E_a is such that $\langle\phi_i| E_a |\phi_j\rangle \neq C_a\delta_{ij}$.

If the distance is $d = 2t + 1$ then it is clear that the code can correct t errors since if E_a, E_b have weight $\leq t$, then $\langle\phi_i| E_a^\dagger E_b |\phi_j\rangle = C_a\delta_{ij}$, since if it was different then then $E_a^\dagger E_b$ would be a logical operator of weight smaller than d and contradicts the assumption. Thus the distance plays a similar role to the distance in classical code. A quantum code with n qubits, k logical operators and distance d is denoted by $[[n, k, d]]$.

Let us now prove this condition. Error recovery means that there exists a recovery operator $\mathcal{R}$ such that

$$\mathcal{R} \circ \mathcal{E}(\rho_C) \propto \rho_C$$

where ρ_C is a code state and proportionality accounts for $\mathcal{E}(\rho) = \sum_a E_a \rho E_b^\dagger$ being non-trace preserving. We prove first the sufficiency: assume that the condition holds. Then let $F_a = \sum_b U_{ab} E_b$ be an equivalent set of Kraus operators for $\mathcal{E}$ such that U diagonalises C. Then

$$\langle\phi_i| F_a^\dagger F_b |\phi_j\rangle = \sum_{cd} U_{cb}\overline{U_{da}} \langle\phi_i| E_a^\dagger E_b |\phi_j\rangle = \sum_{cd} \overline{U_{da}} C_{ab} U_{cb}\delta_{ij} = d_a\delta_{ab}\delta_{ij} \,. \tag{77}$$

Now define for $d_a \neq 0$,

$$R_a = \frac{1}{\sqrt{d_a}} \Pi_C F_a^\dagger \,, \tag{78}$$

where $\Pi_C = \sum_j |\phi_j\rangle \langle\phi_j|$ is the projector onto the code and otherwise $R_a = 0$. The superoperator with Kraus operators R_k satisfies:

$$\sum_a R_a \sum_b F_b \rho_C F_b^\dagger R_a^\dagger = \sum_{a|d_a\neq 0} \frac{1}{d_a} \Pi_C F_a^\dagger \sum_b F_b \rho_C F_b^\dagger F_a \Pi_C \tag{79}$$

We show that this produces something proportional to ρ_C for a basis $\rho_C = |\phi_m\rangle \langle\phi_n|$. Plugging in, we find

$$\sum_{a|d_a\neq 0} \frac{1}{d_a} \sum_{jbk} |\phi_j\rangle \langle\phi_j| F_a^\dagger F_b |\phi_m\rangle \langle\phi_n| F_b^\dagger F_a |\phi_k\rangle \langle\phi_k| = \sum_{a|d_a\neq 0} \frac{1}{d_a} \sum_{jbk} \delta_{ab}\delta_{jm} d_a \delta_{ab}\delta_{kn} d_a |\phi_j\rangle \langle\phi_k| = \sum_a d_a \rho_C \tag{80}$$

which proves that we can recover from the errors. Now we want to show that the existence of R_a such that

$$\sum_{ab} R_a E_b \rho_C E_b^\dagger R_a^\dagger = c\rho_C \tag{81}$$

implies the condition. We can write this for all ρ as

$$\sum_{ab} R_a E_b \Pi_C \rho \Pi_C E_b^\dagger R_a^\dagger = c\Pi_C \rho \Pi_C \,. \tag{82}$$

This means that the two superoperators on the left and right hand side are the same. So there exists complex numbers α_{ab} such that

$$R_a E_b \Pi_C = \alpha_{ab} \sqrt{c} \Pi_C \,, \quad \text{or } \Pi_C E_c^\dagger R_a^\dagger = \Pi_C \overline{\alpha_{ac}} \sqrt{c} \,, \tag{83}$$

which multiplied give

$$\Pi_C E_c^\dagger R_a^\dagger R_a E_b \Pi_C = \overline{\alpha_{ac}} \alpha_{ab} c \Pi_C \tag{84}$$

and summing over a using that $\mathcal{R}$ is trace preserving, gives

$$\Pi_C E_c^\dagger E_b \Pi_C = C_{cb} \Pi_C \quad C_{cb} = \sum_a \overline{\alpha_{ac}} \alpha_{ab} c \tag{85}$$

which is the error correcting condition.

Another consequence of the quantum error correction criterion is the discretisation of errors. We assume that $\{E_a\}$ is a set of correctable errors and want to show that linear combinations of those errors are also correctable. We write $G_a = \sum_b f_{ab} F_b$ where F_a is the basis where the matrix C is diagonal. Using the R_a in the sufficiency proof we get for $\rho_C = |\phi_m\rangle \langle\phi_n|$

$$\sum_a R_a \sum_b G_b \rho_C G_b^\dagger R_a^\dagger = \sum_{abijpq} f_{bp} \overline{f_{bq}} \frac{1}{d_a} |\phi_i\rangle \langle\phi_i| F_a^\dagger F_p |\phi_m\rangle \langle\phi_n| F_q^\dagger F_a |\phi_j\rangle \langle\phi_j| \tag{86}$$

$$= \sum_{ab} f_{ba} \overline{f_{ba}} \frac{1}{d_a} |\phi_m\rangle d_a d_a \langle\phi_n| = \sum_{ab} |f_{ba}|^2 d_a \rho_C \tag{87}$$

which shows that the error correcting conditions are satisfied. We have shown that we only need to care about designing codes which correct discrete errors since linear combinations of those will automatically be corrected.

2.4.3 stabiliser codes

We now define stabiliser codes which are the most popular class of quantum codes considered to date and are based on the element mathematical framework of group theory and provide a generalisation of the classical linear codes we have already discussed.

We first define the Pauli group $\mathcal{P}$ of n qubits. The Pauli group for a single qubit is the following set of Pauli matrices times the fourth roots of unity:

$$\mathcal{P} = \{\omega \mathbf{1}, \omega X, \omega Y, \omega Z\}_{\omega \in \{\pm 1, \pm i\}} \,. \tag{88}$$

We need to include the phases so that it is closed under multiplication. $\mathcal{P}$ is the set of all n-fold tensor products of Pauli matrices with coefficients $\{\pm 1, \pm i\}$:

$$\mathcal{P} = \{\omega P_1 \otimes P_2 \otimes \cdots \otimes P_n \,, \quad \omega \in \{\pm 1, \pm i\} \,, P_i \in \{\mathbf{1}, X, Y, Z\}\} \,. \tag{89}$$

A stabiliser group is an Abelian subgroup of the Pauli group. We denote the stabiliser by $\mathcal{S}$. We want to define the stabiliser code $\mathcal{C}(\mathcal{S})$ as the space of states $|\psi\rangle$ such that $S|\psi\rangle = +|\psi\rangle$ for all $S \in \mathcal{S}$. To make this non-trivial we need to exclude $-\mathbf{1}$ from $\mathcal{S}$, otherwise $-\mathbf{1}|\psi\rangle = |\psi\rangle$ would imply that the subspace is trivial. Note that this implies that the elements of the stabiliser can be simultaneously diagonalised (since it is Abelian) and that they are Hermitian with eigenvalues ± 1. Indeed elements of the Pauli group are either Hermitian, in which case they have eigenvalues ± 1 or anti Hermitian, in which case they have eigenvalues $\pm i$. If S had eigenvalue $\pm i$, then S^2 would have eigenvalue -1 on both the $\pm i$ eigenspaces and $-\mathbf{1}$ would be an element of the stabiliser. We write $\mathcal{S} = \langle S_1, \ldots, S_\ell \rangle$ for the stabiliser generated by $S_1, \ldots, S_\ell$. This is the group obtained by all possible products of the S_i's.

We have already seen examples of stabiliser groups for the 3 and 9 qubit codes. The operators we measure to compute the syndromes are the generators of the respective groups. For the 3 qubit code for example, $\mathcal{S} = \langle Z_1 Z_2, Z_2 Z_3 \rangle = \{\mathbf{1}, Z_1 Z_2, Z_2 Z_3, Z_1 Z_3\}$ and the states stabilised by it are $|000\rangle$ and $|111\rangle$.

Now assume that our generators are independent – namely we cannot obtain one of them by multiplying the others – so that the set is the smallest. We denote the number of such independent

generators $n-k$. We now claim that the stabiliser code associated to it encodes k logical qubits. To see this note that each stabiliser has two eigenspaces of size 2^{n-1}, one associated to the $+1$ eigenvalue and one to the -1. Thus the condition $S_1 |\psi\rangle = |\psi\rangle$ singles out a subspace of dimension 2^{n-1} out of the space of the qubits. Imposing now $S_2 |\psi\rangle = |\psi\rangle$ further restricts the subspace to be of size $2^{n-1-1} = 2^{n-2}$. Proceeding this way until $S_{n-k} |\psi\rangle = |\psi\rangle$ we are left with a subspace of dimension $2^{n-(n-k)} = 2^k$ as claimed. From this discussion we have also seen that we can decompose the Hilbert space in 2^{n-k} subspaces each corresponding to a different choice of eigenvalues of the stabiliser generators, or syndromes $\sigma \in \mathbb{F}_2^{n-k}$ and each of dimension 2^k:

$$(\mathbb{C}^2)^{\otimes n} = \bigoplus_{\sigma \in \mathbb{F}_2^{n-k}} \mathcal{H}_\sigma \,, \quad \mathcal{H}_\sigma \simeq (\mathbb{C}^2)^{\otimes k} \,. \tag{90}$$

We now discuss how Pauli matrices are realised on the code space, ie the logical operators. An element of the Pauli group implementing a logical operator should be commuting with the stabiliser, so that it does not leave the code subspace. Indeed suppose that $|\psi\rangle$ is in the quantum code, and $\bar{A} S_i = -S_i \bar{A}$. Then $S_i \bar{A} |\psi\rangle = S_i \bar{A} |\psi\rangle = -\bar{A} |\psi\rangle$, so that $\bar{A} |\psi\rangle$ is not in the code anymore. However, we need the logical operators to be independent from the stabiliser, otherwise they would be part of it. Let us consider the centraliser of $\mathcal{S}$ in the $\mathcal{P}$,

$$\mathcal{Z}_{\mathcal{P}}(\mathcal{S}) = \{ g \in \mathcal{P} \,|\, gS = Sg \,, \forall S \in \mathcal{S} \} \,. \tag{91}$$

Now the normaliser of the stabiliser is

$$\mathcal{N}_{\mathcal{P}}(\mathcal{S}) = \{ g \in \mathcal{P} \,|\, g S_i g^{-1} \in \mathcal{S} \forall i \in \{1, \ldots, \ell\} \} \,, \tag{92}$$

is actually equivalent to the centraliser. This follows because $g S_i g^{-1} = S_i (-1)^c$ for some c that depends on the commutation of g, S_i. But this is in $\mathcal{S}$ only if $c = 0$ since there is only S_i and not $-S_i$ in $\mathcal{S}$, which shows that the centraliser is the same as the normaliser.

We can then consider the quotient group

$$\mathcal{L} = \mathcal{N}_{\mathcal{P}}(\mathcal{S}) / \mathcal{S} \tag{93}$$

which corresponds to the logical Pauli operators. Recall that the quotient group is defined as the set of left cosets: $G/N = \{aN \,, a \in G\}$.

In practice we will pick a representative of each of the equivalence class $A\mathcal{S}$ for $A \in \mathcal{N}_{\mathcal{P}}(\mathcal{S})$ and identify $\mathcal{L}$ with the set of these canonical representatives. A set of generators of $\mathcal{L}$ provides the logical X and Z, $\bar{X}_i, \bar{Z}_i$, $i \in \{1, \ldots, k\}$ since there are k logical qubits. They satisfy:

$$\bar{X}_i \bar{Z}_j = \bar{Z}_j \bar{X}_i \,, \quad i \neq j \,, \quad \bar{X}_i \bar{Z}_i = -\bar{Z}_i \bar{X}_i \,. \tag{94}$$

For the 3 qubit code, the centraliser of the stabiliser is $16 = 2^{3+1}$-dimensional. It is given by all the monomials in Z_i and those times $X_1 X_2 X_3$:

$$\mathcal{N}_{\mathcal{P}}(\mathcal{S}) = \{ g, gZ_1, gZ_2, gZ_3, gZ_1Z_2, gZ_2Z_3, gZ_1Z_3, gZ_1Z_2Z_3, \,|\quad g \in \{\mathbf{1}, X_1X_2X_3\} \} \tag{95}$$

We can pick the logical operators as

$$\bar{Z} = Z_1 Z_2 Z_3 \,, \quad \bar{X} = X_1 X_2 X_3 \,, \quad \mathcal{L} = \langle \bar{Z}, \bar{X} \rangle \,. \tag{96}$$

Equivalently we could take $\bar{Z} = Z_1$ for example. The dimension of $\mathcal{L}$ is thus 2^{2k} and that of $\mathcal{N}_{\widetilde{\mathcal{P}}}(\mathcal{S})$ is 2^{n+k}. Thus we cannot generate the whole $\widetilde{\mathcal{P}}$ with only elements of $\mathcal{S}, \mathcal{L}$. To understand the missing elements it is convenient to introduce a representation of elements in $\widetilde{\mathcal{P}}$ as binary vectors as follows.

Binary string representation Without loss of generality we can write any quantum channel modelling a noise process as

$$\mathcal{E}(\rho) = \sum_a A_a \rho A_a^\dagger \,, \quad A_a = \sum_{E \in \widetilde{\mathcal{P}}} \alpha_a^E E \tag{97}$$

where we defined

$$\widetilde{\mathcal{P}} = \mathcal{P}/\{\pm 1, \pm i\} \tag{98}$$

namely we identify in the Pauli group elements that differ by a phase since they can be clearly combined by a redefinition of the coefficients α_a^E and $E \in \widetilde{\mathcal{P}}$ provide a basis of the space of linear operators of n qubits.

Elements of $\widetilde{\mathcal{P}}$ can be identified by a binary string string $\alpha = (x|z) \in \mathbb{F}_2^{2n}$ by writing the element as

$$P(\alpha) = X(x)Z(z) = X_1^{x_1} \cdots X_n^{x_n} Z_1^{z_1} \cdots Z_n^{z_n} \,. \tag{99}$$

Commutation relations can be expressed as

$$P(\alpha)P(\beta) = X(x)Z(z)X(v)Z(w) = (-1)^{z\cdot v + x\cdot w} X(v)Z(w)X(x)Z(z) = (-1)^{\omega(\alpha,\beta)} P(\beta)P(\alpha) \tag{100}$$

where we introduced the bilinear form

$$\omega(\alpha,\beta) = \begin{pmatrix} x & z \end{pmatrix} \Lambda \begin{pmatrix} v \\ w \end{pmatrix} \in \mathbb{F}_2 \,, \quad \Lambda = \begin{pmatrix} 0 & \mathbf{1}_n \\ \mathbf{1}_n & 0 \end{pmatrix} . \tag{101}$$

ω is a symplectic or alternating form since $\omega(\alpha,\alpha) = 0$. The space of binary strings $\mathbb{F}_2^{2n}$ equipped with ω is a symplectic vector space. We denote by $r(g)$ the binary representation of $g \in \widetilde{\mathcal{P}}$.

Now we introduce the parity check matrix H of the stabiliser code as the $n-k \times 2n$ matrix obtained by concatenating the binary representations of the generators of the stabiliser group. If $\mathcal{S} = \langle S_1, \ldots, S_\ell \rangle$, its parity check matrix has rows $r(S_1), \ldots, r(S_\ell)$. It is easy to see that the generators of $\mathcal{S}$ are multiplicatively independent if and only if the rows of H are linearly independent. This follows from the fact that r is a group homomorphism: $r(g) + r(g') = r(gg')$. We can similarly associate a $2k \times 2n$ matrix L to the generators of the logical operators $\mathcal{L} = \langle \bar{Z}_1, \bar{X}_1, \ldots, \bar{Z}_k, \bar{X}_k \rangle$.

A useful result is that given the stabiliser $\mathcal{S} = \langle S_1, \ldots, S_\ell \rangle$ with independent generators, then there exist $T_1, \ldots, T_\ell$ such that

$$T_i T_j = T_j T_i \,, \quad T_i S_i = -S_i T_i \,, \quad T_i S_j = S_j T_i \,, j \neq i \,, \quad T_i L = L T_i \,, \forall L \in \mathcal{L} \,. \tag{102}$$

To show this let us fix $i \in \{1, \ldots, \ell\}$. In the binary representation, since the rows of H and L are linearly independent, we can find $\alpha \in \mathbb{F}_2^{2n}$ such that

$$\begin{pmatrix} H \\ L \end{pmatrix} \Lambda \alpha = e_i \,, \tag{103}$$

where e_i is a $n+k$ binary vector with 1 in the i-th entry and the rest 0. Note that the solution is not unique. Any $\alpha + \sum_{i=1}^{\ell} c_i H_{i,\cdot}$ would satisfy the same equation. We choose one solution and let $T_i = P(\alpha)$, which satisfies the required commutation relations.

We can repeat the same construction for all $i = 1, \ldots, \ell$ and impose that T_i commutes with $T_1, \ldots, T_{i-1}$ by adding linearly independent equations that impose that. We thus define the group of destabilisers or pure errors:

$$\mathcal{T} = \langle T_1, \ldots, T_\ell \rangle \,. \tag{104}$$

The operators T_i's are independent from $\mathcal{S}$ and $\mathcal{L}$ – otherwise they would be in $\mathcal{N}(\mathcal{S})$ and commute with $\mathcal{S}$. $\mathcal{T}$ is also Abelian. For example, for the 3 qubit code $\mathcal{T}$ has dimension 2^2 is the Abelian group

$$\mathcal{T} = \langle X_1 X_2, X_2 X_3 \rangle \tag{105}$$

The dimension of $\mathcal{T}$ is 2^ℓ. Setting $\ell = n - k$, we see that

$$\langle S_1, \ldots, S_{n-k}, \bar{X}_1, \bar{Z}_1, \ldots, \bar{X}_{2k}, \bar{Z}_{2k}, T_1, \ldots, T_{n-k} \rangle \tag{106}$$

has dimension $2^{n-k+2k+n-k} = 2^{2n}$ and is thus a set of generators for $\widetilde{\mathcal{P}}$.

We can expand any given element $E \in \widetilde{\mathcal{P}}$ as

$$E = TSL \,, \quad T \in \mathcal{T}, S \in \mathcal{S}, L \in \mathcal{L} \,. \tag{107}$$

This set is convenient to discuss error correction as we will show.

2.5 References

A more detailed discussion of noise in quantum computing can be found in chapters 1 and 5 of [LB13]. See [Mer07] for an introduction to quantum computing. For classical error correction, you can refer to [NC01] for a review of the basic concepts and [RU08] for an advanced discussion focused on iterative decoding. The stabiliser formalism is well explained in [NC01]. A proof of the decomposition into pure errors, stabiliser and logical operators using the formalism of symplectic vector spaces can be found in [Haa17].

3 CSS codes and the toric code

3.1 CSS codes

3.1.1 General construction

In the last lecture we have seen some ad hoc examples of stabiliser codes such as the 9 qubit code. But how do we construct a stabiliser code? We now present a general class of stabiliser codes called CSS from the initials of their inventors (Calderbank, Shor, Steane) that can be described in terms of two classical linear codes. Recall that a classical linear code $\mathcal{C}$ with parameters $[n,k,d]$ is the set of codewords such that $Hx^T=0$ where H is the $n-k\times n$ parity check matrix. The distance d of the code is the weight of the smallest non-zero codeword. Recall also that the dual code $\mathcal{C}^\perp$ encodes $n-k$ bits and is defined as by the condition $Gx^T=0$ where G is the generator matrix of $\mathcal{C}$ of size $k\times n$ that satisfies $HG^T=0$. $\mathcal{C}^\perp$ consists of all the codewords orthogonal to those of $\mathcal{C}$ and $G^\perp=H$.

Let now $(\mathcal{C}_X,\mathcal{C}_Z)$ be a pair of classical linear codes of length n such that $\mathcal{C}_Z^\perp\subseteq\mathcal{C}_X$. We now define a stabiliser code via its parity check matrix as follows

$$H=\begin{pmatrix}H_X & 0\\ 0 & H_Z\end{pmatrix} \tag{108}$$

where H_X (H_Z) is the parity check matrix of $\mathcal{C}_X$ ($\mathcal{C}_Z$). The peculiarity of CSS codes is thus to have X type and Z type checks. This is a valid parity check matrix if its rows are orthogonal according to the symplectic product:

$$H\Lambda H^T=\begin{pmatrix}H_X & 0\\ 0 & H_Z\end{pmatrix}\begin{pmatrix}0 & \mathbf{1}_n\\ \mathbf{1}_n & 0\end{pmatrix}\begin{pmatrix}H_X^T & 0\\ 0 & H_Z^T\end{pmatrix}=\begin{pmatrix}0 & H_XH_Z^T\\ H_ZH_X^T & 0\end{pmatrix}. \tag{109}$$

This is zero since by definition $\mathcal{C}_Z^\perp\subseteq\mathcal{C}_X$ implies that $H_Xx^T=0$ for $x\in\mathcal{C}_Z^\perp$, namely $x=uG_Z^\perp=uH_Z$, so that $H_XH_Z^T=0$ as required. Note that equivalently we could have asked that $\mathcal{C}_X^\perp\subseteq\mathcal{C}_Z$ since then $H_Zx^T=0$ if $x=uG_X^\perp=uH_X$, which implies $H_ZH_X^T=0$.

Let us denote by $dim(\mathcal{C})$ the number of logical bits of $\mathcal{C}$. Then the number of independent rows of H_X is $dim(\mathcal{C}_X^\perp)=n-dim(\mathcal{C}_X)$. The number of rows of H is $dim(\mathcal{C}_X^\perp)+dim(\mathcal{C}_Z^\perp)$ so that the number of logical qubits is

$$k=n-dim(\mathcal{C}_X^\perp)-dim(\mathcal{C}_Z^\perp)=dim(\mathcal{C}_X)-dim(\mathcal{C}_Z^\perp)=dim(\mathcal{C}_X/\mathcal{C}_Z^\perp)=dim(\mathcal{C}_Z/\mathcal{C}_X^\perp). \tag{110}$$

Logical X and Z operators are also given by products of X and Z operators only. Logical X need to commute with the Z checks and be independent from the X checks. A $2n$ bitstring $(x|0)$ represents an operator that commutes with the Z stabilisers if $H_Zx^T=0$ namely $x\in\mathcal{C}_Z$. The rows of H_X describe the codewords of $\mathcal{C}_X^\perp$ and we mod out their action as we did above for the general stabiliser formalism. Similarly for Z type logical operators. We can define the distances

$$d_X=\min\{|x|\,,x\in\mathcal{C}_Z/\mathcal{C}_X^\perp\},\quad d_Z=\min\{|x|\,,x\in\mathcal{C}_X/\mathcal{C}_Z^\perp\}. \tag{111}$$

so that the distance of the CSS code is

$$d=\min\{d_X,d_Z\}. \tag{112}$$

3.1.2 Example: Steane's code

Recall that the classical Hamming code $[7,4,2]$ is such that $\mathcal{C}^\perp\subseteq\mathcal{C}$. Thus we can take $\mathcal{C}_X=\mathcal{C}_Z=\mathcal{C}$ and $H_X=H_Z$ being the parity check matrix of the Hamming code. The number of logical qubits is $k=4-(7-4)=1$. The distance is 3 so that the Steane code is a $[[7,1,3]]$ code as can be verified explicitly.

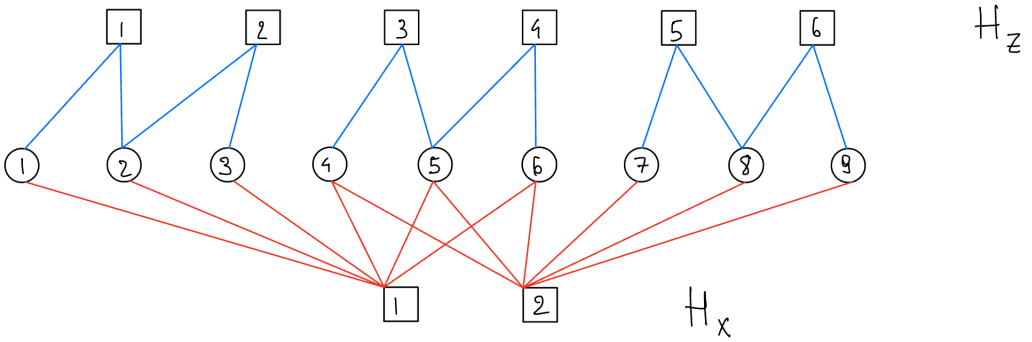


Figure 7:

3.1.3 Other representations of CSS codes

Tanner graph Similarly to classical linear codes, we can define the Tanner graph of a CSS code by stacking the Tanner graphs of the classical codes $\mathcal{C}_X, \mathcal{C}_Z$ which now share the same variables, associated to the qubits but have two types of checks, X type and Z type. We illustrate this for the 9 qubits code in Figure 7 shows an example for the Hamming code.

Chain complex Another particularly useful way to think about CSS codes is in terms of chain complexes.

We define a chain complex C over $\mathbb{F}_2$ of length 2 as the sequence

$$C = (C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0)\,. \tag{113}$$

Here C_i are vector spaces over $\mathbb{F}_2$ and $\partial_i : C_i \to C_{i-1}$ are linear operators fulfilling

$$\partial_1 \partial_2 = 0\,. \tag{114}$$

We can interpret this as the defining condition of a CSS code, $H_X H_Z^T = 0$ if we identify

$$\partial_2^T = H_Z\,, \quad \partial_1 = H_X\,. \tag{115}$$

So if we are given a chain complex we can construct a CSS code by defining the X, Z checks using the boundary operators. The elements of C_i are called i-chains and we introduce the spaces

$$Z_1(C) = \ker(\partial_1) \in C_1 \tag{116}$$
$$B_1(C) = \mathrm{im}(\partial_2) \in C_1 \tag{117}$$
$$H_1(C) = Z_1(C)/B_1(C)\,. \tag{118}$$

The elements of Z_1, B_1 spaces are called 1-cycles and 1-boundaries and H_1 is the 1st homology group, which contains the non-trivial cycles, ie those that are not boundaries.

Basis elements of C_i are called i-cells. We assume that C_i have a scalar product, which allow us to define the dual space C^i of co-chains, and similarly define:

$$Z^1(C) = \ker(\partial_2^T) \in C^1 \tag{119}$$
$$B^1(C) = \mathrm{im}(\partial_1^T) \in C^1 \tag{120}$$
$$H^1(C) = Z^1(C)/B^1(C)\,. \tag{121}$$

Note that we used ∂_2^T which is a linear map from $C^1 \to C^2$. The elements of Z^1, B^1 spaces are called 1-cocycles and 1-coboundaries and H^1 is the first cohomology.

In the canonical basis, we can identify $C_i = C^i$ by identifying the linear functional $c^* \in C^i$ that associates $1 \in \mathbb{F}_2$ to c and 0 to the other chains, with c. In the interpretation as a CSS code, where $\partial_2 = H_Z^T$ and $\partial_1 = H_X$, X checks correspond to 0 cells, physical qubits are 1 cells, and Z checks correspond to 2 cells. We can then associate to a 1-chain a string of Z Pauli operators

acting on the corresponding qubits, and to a i-cochain a string of X Pauli operators acting on the corresponding qubits. Z logicals are 1 chains that commute with H_X, namely they are $\in \ker(\partial_1)$ and we identify logical operators that differ by combinations of Z checks, so by $\mathrm{im}(\partial_2)$. Thus Z logicals are in $\ker(\partial_1)/\mathrm{im}(\partial_2) = H_1$ while X logicals are in $\ker(\partial_2^T)/\mathrm{im}(\partial_1^T) = H^1$.

3.2 Topological codes

We consider now a family of codes parametrised by the number of physical qubits n. We call a code Low Density Parity Check if the weight of each row and column of the parity check matrix is $O(1)$. This means that each stabiliser code acts only on a bounded number of qubits and that each qubit is acted on by only a bounded number of stabilisers. LDPC codes play an important role in both classical and quantum error correction. We will now consider a family of codes called topological codes. In these codes qubits are placed on the edges of graphs embedded in surfaces, and the topology of the surface will control the number of logical qubits. We will use the formalism of chain complexes introduced above specified to this setting. Thus we start to consider a 2 chain complex

$$C = (C_2 \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0)\,, \tag{122}$$

where we call 0-cells, vertices, 1-cells, edges and 2-cells, faces. If a face f has boundary $(e_1, \ldots, e_k)$ and an edge e has endpoints (v_1, v_2) then

$$\partial_2 f = e_1 + \cdots + e_k\,, \quad \partial_1 e = v_1 + v_2\,, \tag{123}$$

Cycles are collections of closed curves on the lattice, and the condition $\partial_1\partial_2 = 0$ simply means that the boundary of a face is closed. The homology group

$$H_1 = Z_1/B_1 \tag{124}$$

has elements given by cosets $\bar{z} = \{z + b \,|\, b \in B_1\}$. For a surface with genus g the first homology group is

$$H_1 = \mathbb{Z}_2^{2g}\,. \tag{125}$$

This is because boundaries are contractible cycles and H_1 counts the equivalence class of non-contractible cycles, which can be distinguished by how many times they wind around the cycles of the surface, which is a topological invariant of the surface.

The first non-trivial case is that of the torus $g = 1$. H_1 has two generators, corresponding to the two non-contractible cycles of a torus. See blue non-contractible cycles in 8.

We call the code obtained by embedding the square lattice on the torus, the toric code.

The dual boundary operators $\partial_1^T : C^0 \to C^1, \partial_2^T : C^1 \to C^2$, which after identifying $C^i \equiv C_i$ map vertices to edges emanating from it and edges to faces around it. We also have

$$H^1 \simeq H_1 \simeq \mathbb{Z}_2^{2g}\,. \tag{126}$$

It is also useful to consider the dual lattice which has dual vertices v^*, edges e^*, and faces f^* associated to faces, edges and vertices of the original lattice: $C_0^* \simeq C_2, C_1^* \simeq C_1, C_2^* \simeq C_0$ with boundary operators $\partial_1^* : C_0^* \to C_1^*, \partial_2^* : C_1^* \to C_2^*$ and

$$H_1^* = \frac{Z_1^*}{B_1^*} \simeq H_1\,. \tag{127}$$

We will draw Z operators as path on the lattice, and X operators as paths on the dual lattice as in figure 8. This is because ∂_1 maps primal edges to vertices and checks Z errors, while X errors are checked similarly by dual vertices and so X and Z errors are related by duality.

The stabiliser operators are defined in terms of the parity check matrices. Z checks are associated with $H_Z = \partial_2^T : C_1 \to C_2$ and is a $|F| \times |E|$ matrix. To each row is associated a plaquette operator:

$$B_f = \prod_{e \in \partial_2 f} Z_e \tag{128}$$

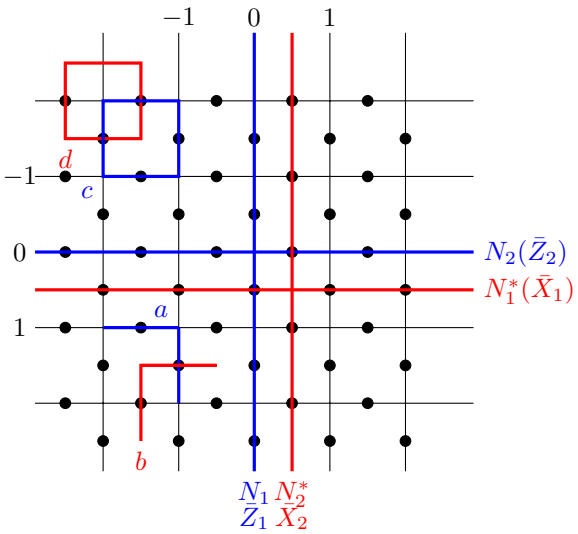


Figure 8: Toric code square lattice with periodic boundary conditions. Qubits are placed on the edges and represented by black dots. Blue paths are Z errors, red paths on the dual lattice are X errors. c, d are X- and Z-stabilisers, while N_i, N_i^* are logical operators corresponding to non-contracbtle loops around the torus.

X checks are associated with the parity check $H_X = \partial_1 : C_1 \to C_0$ which is a $|V| \times |E|$ matrix, whose rows define the vertex operators:

$$A_v = \prod_{e|v\in\partial_1 e} X_e \tag{129}$$

They are represented in figure 8. It is straightforward to check that indeed vertex and plaquette operators commute with each other since they will either share zero or two edges and this means an even number of minus signs when we consider the commutation of X and Z operators, leading to the commutation property.

Note that the Hadamard transform implements duality on the lattice, i.e. maps plaquette to vertex operators:

$$H^{\otimes n} A_v H^{\otimes n} = \prod_{e^*|f^*\in\partial_2^* e^*} Z_e \equiv B_{f^*}\,, \quad H^{\otimes n} B_f H^{\otimes n} = \prod_{e\in\partial_2 f} X_e = \prod_{e^*|v^*\in\partial_1^* e^*} X_e \equiv A_{v^*}\,, \tag{130}$$

We can check that (125) is consistent with what we know from the theory of stabiliser codes that the number of logical qubits is n minus the number of independent rows of the parity check matrix. For a torus, if we sum all the elementary contractible cycles around faces of the torus, we get 0. Similarly for the sum of cycles around dual faces or vertices. This can also be verified by noting that

$$\prod_{f\in F} B_f = 1\,, \quad \prod_{v\in V} A_v = 1 \tag{131}$$

because every edge is counted twice and we use that $Z_e^2 = X_e^2 = 1$. This means that the number of logical qubits is

$$E - k = V + F - 2 \to k = 2 \tag{132}$$

This is also consistent with Euler relations for a graph embedded on the torus:

$$\chi = V - E + F = 2(1 - g) = 0\,. \tag{133}$$

The logical operators are associated with non-contractible cycles $\bar{Z}_1, \bar{Z}_2$ on the lattice and its dual $\bar{X}_1, \bar{X}_2$, see figure 8 for definition of cycles N_a, N_a^*:

$$\bar{X}_a = \prod_{e^*\in N_a^*} X_e\,, \quad \bar{Z}_a = \prod_{e\in N_a} Z_e\,. \tag{134}$$

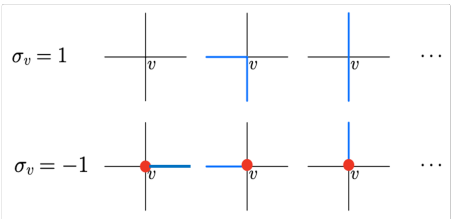


Figure 9:

This is a choice of the representative of the generators of the quotient groups H_1, H^1 of non-contractible paths. If we call n the number of qubits, we see that the smallest weight of the logical operators is the height or width of the square lattice so that the distance of the toric code is $d = \sqrt{n/2}$. The toric code is thus a

$$[[2d^2, 2, d]] \tag{135}$$

stabiliser code. Note that the toric code seems to have pretty poor parameters since the rate $2/d^2 \to 0$ as $d \to \infty$. However, it has nice properties that make it one of the best codes known (with its related code, the surface code discussed below). These properties are: local checks that make it easier to implement in hardware, and as we shall we see high error threshold.

Now we discuss the code space. Consider the projector onto the code space:

$$\prod_v \frac{\mathbf{1} + A_v}{2} \prod_f \frac{\mathbf{1} + B_f}{2} \tag{136}$$

A basis of states is naturally associated with paths on the dual lattice $|c\rangle$, $c \in C_1^*$, since it can be obtained from the $|0^n\rangle$ state (0 path) by acting with X operators that are associated with dual paths. Notice that $(\mathbf{1} + B_f)|c\rangle = 0$ if c is a path that visits an odd number of edges around f. Imposing this for all faces implies that c has to visit every face an even number of times, which means that c has to be a dual cycle: $c \in Z_1^*$. So the projector on the $+1$ eigenspace of B_f restricts the space to dual cycles. The vertex operators can be expanded as

$$\prod_v (\mathbf{1} + A_v) = \prod_{f^*} (\mathbf{1} + A_{f^*}) = \sum_{\{f_i^*\}} \prod_j A_{f_j^*} = \sum_{\{f_i^*\}} \prod_j \prod_{e^* \in \partial_2^* e^*} X_e = \sum_{b^* \in B_1^*} \prod_{e^* \in b} X_e \,, \tag{137}$$

where B_1^* is the space of boundaries consisting of contractible dual cycles. Acting with this projector onto a dual cycle $z \in Z_1^*$,

$$\prod_v \frac{\mathbf{1} + A_v}{2} |z\rangle = |\bar{z}\rangle \tag{138}$$

gives an element $\bar{z} \in Z_1^*/B_1^* = H_1^* \simeq H_1$. So we see that the codes space is indeed spanned by elements of the first homology group.

Pauli operators in the toric code can be associated to paths on the lattice (phase flips) or the dual lattice (bit flips): $(c, c^*) \in C_1 \times C_1^*$. Given an error (c, c^*), we can compute its syndrome by checking the commutation relations with A_v, B_f. A_v will catch the phase flip errors that correspond to paths with an odd number of edges at v. B_f will catch bit flip errors that correspond to dual paths with an odd number of edges at f. See figure 9.

We call violations of the cycle condition defects. Because of (131) defects occur in pairs. Figure 10 shows examples of error paths and show that the same syndrome can be associated to multiple paths. We will discuss this further when looking at decoding later.

A few comments are in order before about the toric code. The code space can be also described as the ground state of the Hamiltonian

$$H = -\sum_{f \in F} B_f - \sum_{v \in V} A_v \,. \tag{139}$$

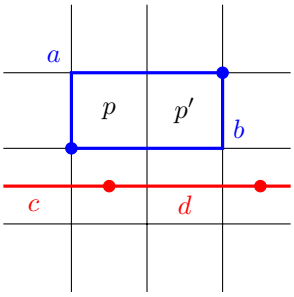


Figure 10: a and b are two possible errors (phase flip paths) that give rise to the same syndrome, here represented by blue dots. Similarly, c, d are two possible errors (bit flip paths) with the same syndrome, represented by red dots.

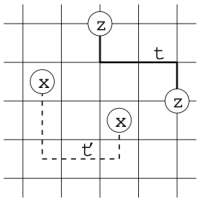


Figure 11:

This Hamiltonian is a special case of the Hamiltonian for the $\mathbb{Z}_2$ lattice gauge theory:

$$H = -\sum_{f \in F} B_f - \lambda \sum_e X_e \tag{140}$$

at $\lambda = 0$ where the gauge transformations are generated by A_v and the invariant sector is defined by $A_v |\psi\rangle = |\psi\rangle$. Since all terms commute it is straightorward to diagonalise the toric code Hamiltonian. Excitations are separated by a gap $\Delta = 2$. This model has been proposed to perform error correction at the physical level, namely by realising it and having excitations induced by noise to die out because of relaxation in the system occurring for example by cooling. The ground state of the toric code is highly entangled and is topologically ordered, since the ground state degeneracy depends on the topology of the surface only. Now consider two paths c, c^* that begin and end at given points. The states

$$\prod_{e \in c} Z_e |\psi\rangle \ , \quad \prod_{e^* \in c^*} X_e |\psi\rangle \tag{141}$$

for $|\psi\rangle$ in the toric code can be interpreted as having particles at the endpoints of the paths. Indeed A_v, B_f will have non-trivial action on them only at the endpoints. Due to the relation with the gauge theory we can think of the Z particles as electric charges and the X particles as magnetic vortices. See figure 11.

Suppose that we start with a Z particle at a given site of the lattice. This means that there is a string attached to it. If we now move a X particle around the Z particle, we will move the X string around the Z string and thus pick up a minus sign. See figure 12. This property is characteristic of Abelian anyons. The theory of topological phases and anyons is a deep and beautiful topic of which the toric code is just a starting point. I will now delve further into this but see the references at the end of this lecture for more.

3.2.1 Surface code

Implementing the periodic boundary conditions of the toric code requires coupling qubits that are on opposite edges of the square lattice. This prompts looking at a version of the toric code with open boundaries, called the surface code. The boundaries are chosen as shown in figure 13.

At the boundaries, the parity check operators act on 3 instead of 4 qubits. For a $d \times d$ lattice, there are d^2 qubits on horizontal edges, $(d-1)^2$ on vertical edges, $d(d-1)$ plaquette operators and $d(d-1)$

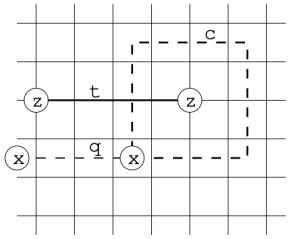


Figure 12:

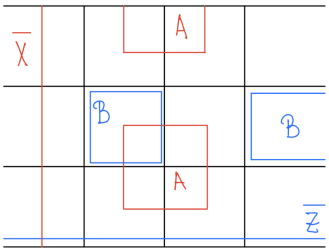


Figure 13: Show straight lines not zig zag like here

vertex operators. Note that now all stabiliser are independent since their product is not 1 anymore. The number of logical qubits is thus one:

$$k = d^2 + (d-1)^2 - 2d(d-1) = 2d^2 + 1 - 2d - 2d(d-1) = 1\,. \tag{142}$$

Logical operator $\bar{Z}$ connects the rough and logical $\bar{X}$ connects the smooth boundary. All other (equivalent) logical operators can be obtained by multiplying these by plaquette and vertex operators.

3.3 Decoding problem

3.3.1 Ideal syndrome measurements

We now discuss the decoding problem in some detail. We assume that the noise is described by a Pauli channel:

$$\mathcal{E}(\rho) = \sum_{E \in \widetilde{\mathcal{P}}} p(E) E \rho E\,. \tag{143}$$

The channel is specified by a probability distribution over the Pauli group (up to phases).

We assume that we can prepare the initial state $\rho = |\Psi\rangle\langle\Psi|$ in a quantum stabiliser code. Then noise acts and we can perform noiseless syndrome measurement by adding ancillas and using the measurement circuit discussed in the previous lecture. If the outcome of the measurement is σ, the resulting state is:

$$\rho' \propto \sum_{E} p(E) \Pi_\sigma E \rho E \Pi_\sigma\,, \tag{144}$$

Let us denote by E_σ an error with the syndrome σ. Recall now the groups $\mathcal{S}, \mathcal{L}$ and $\mathcal{T}$ from the theory of stabiliser codes. We know that all the errors obtained by multiplying E_σ by an element of $\mathcal{N}(\mathcal{S})$ have the same syndrome. We can pick in a canonical way

$$E_\sigma = T_\sigma \equiv T_1^{\sigma_1} \cdots T_\ell^{\sigma_\ell} \tag{145}$$

where T_i are the generators of the pure errors or destabilisers, which has the convenient property that they have trivial logical part. But any other prescription to compute an error with the right syndrome will do. We stick with T_σ here.

Any element in $\mathcal{S}$ will act trivially on the code state and therefore does not cause any problem. Instead a non-trivial element in $\mathcal{L}$ will change the logical state but goes undetected. In formulas, writing $E = TSL$,

$$\rho' \propto \sum_{S\in\mathcal{S}} \sum_{L\in\mathcal{L}} p(T_\sigma SL) LT_\sigma \rho T_\sigma L = \sum_{L\in\mathcal{L}} p(L|\sigma) LT_\sigma \rho T_\sigma L\,, \tag{146}$$

$$p(L|\sigma) = \sum_{S\in\mathcal{S}} p(T_\sigma SL)\,. \tag{147}$$

We see that the resulting state is an ensemble $(LT_\sigma |\Psi\rangle, p(L|\sigma))_{L\in\mathcal{L}}$. The best decoding strategy now is to compute the state in this ensemble with highest probability:

$$\max_{L\in\mathcal{L}} p(L|\sigma)\,. \tag{148}$$

This is called maximum likelihood decoding because in analogy to the classical decoding problem, it computes the output that maximise the probability of the channel – here this channel includes the syndrome measurement circuit. The morale is that logical errors are those that stabiliser codes cannot detect, and so the best we can do is to compute the most likely logical error to have occurred.

The main concern then is how to perform the computation required by maximum likelihood decoding efficiently. Naively in fact the sum involved in computing $p(L|\sigma)$ is exponentially large in the number of qubits. Due to the arbitrariness of $p(E)$ it seems that in the worst case this is unavoidable. In fact this intuition can be made precise. When looking at the runtime of algorithms which take inputs of size n, P denotes the set of problems which can be solved with runtime polynomial in n – which from a theoretical perspective are called efficient. NP denotes the class of problems for which a solution can be verified in polynomial time. A classic example is the SAT problem, where the problem is compute a bit string that satisfies a Boolean formula, ie a sequence of AND and ORs of bits, for example $(x_1 \vee x_2) \wedge (x_2 \vee x_3)$. Note that verifying that a bit string solves the binary formula is easy, just plug in and see if you get true. However, coming up with a solution to the problem seems much more complicated. In fact, SAT is NP-complete: this means that there exist a reduction of every problem in NP to SAT so that if we can solve SAT we can solve all the problems in NP. A long-standing open question in mathematics is P vs NP, for which it is fair to say that everyone thinks are distinct but a mathematical proof is lacking. A harder problem is to count the number of solutions to NP problems. The class of these problems is called $\#P$. Classical decoding amounts to a combinatorial optimisation problem and is NP-complete. Quantum decoding instead is $\#P$-complete [IP13]. These theoretical conclusions mean that there is no hope to find an efficient algorithm to solve any decoding problem, either classical or quantum, and quantum decoding is in the worst case harder than classical decoding. However, this does not prevent efficient algorithms to exist, which can compute approximately the logical probabilities for certain settings of practical relevance.

Another decoding strategy could have been to output the error compatible with the syndrome which has the highest probability. We can obtain this as as a zero temperature limit of our formalism. Let p^β be a deformation of the error model by raising each term by the positive number β. Then we can define

$$\max_{L\in\mathcal{L}} \lim_{\beta\to\infty} \sum_{S\in\mathcal{S}} p(T_\sigma SL)^\beta = \max_{L\in\mathcal{L}} \max_{S\in\mathcal{S}} p(T_\sigma SL)\,, \tag{149}$$

which indeed returns the most likely error. This decoder is suboptimal since some errors could be more likely but at the same time be in an equivalence class that has a lower probability and thus not being the optimal recovery.

We define the success probability of a decoder. Let us denote by $L_*(\sigma)$ the logical class the decoder returns for a given syndrome σ. Then the success probability is

$$P_{\text{succ}} = \sum_{\sigma\in\mathbb{F}_2^{n-k}} p(L_*(\sigma)|\sigma)\,. \tag{150}$$

3.3.2 Statistical mechanics interpretation

To understand better what problem we need to solve for decoding, we establish a connection between computing logical probabilities and statistical mechanics. This will show that we can use methods from

statistical mechanics to implement approximate decoding. We will derive the stat mech mapping for a generalisation of the logical probability, that associates to $E \in \widetilde{\mathcal{P}}$ the partition function corresponding to the marginal probability over the stabiliser:

$$Z_E \equiv p([E]) = \sum_{S \in \mathcal{S}} p(SE)\,. \tag{151}$$

The objects required for the logical probabilities above are $p(L|\sigma) = p([T_\sigma L])$. Writing a generic stabiliser in terms of a set of generators $S_1, \ldots, S_\ell$:

$$S = S_1^{c_1} \cdots S_\ell^{c_\ell}\,. \tag{152}$$

this leads to a statistical mechanical models with binary variables c_i:

$$Z_E = \sum_{c \in \mathbb{F}_2^\ell} p(S_1^{c_1} \cdots S_\ell^{c_\ell} E)\,. \tag{153}$$

To get some intuition we start by looking at iid bit flip noise

$$p(E) = \prod_{i=1}^n p_1(E_i)\,, \quad p_1(E) = \begin{cases} 1-\epsilon & \text{if } E = \mathbf{1} \\ \epsilon & \text{if } E = X \\ 0 & \text{if } E = Y, Z \end{cases}, \tag{154}$$

and consider the toric code and further set $E = \mathbf{1}$. Then only the sum over vertex stabilisers survives and we get denoting vertices and edges by V, E:

$$Z_{\mathbf{1}} = \sum_{c \in \mathbb{F}_2^V} \prod_{(vw) \in E} p_1(X^{c_v + c_w})\,, \tag{155}$$

and introducing $s = (-1)^c$:

$$p_1(X^{c+c'}) = \begin{cases} 1-\epsilon & \text{if } c = c' \\ \epsilon & \text{if } c \neq c' \end{cases} = e^{Jss' + h}\,, \quad J = \log\left(\frac{1-\epsilon}{\epsilon}\right), \quad h = \log(\epsilon(1-\epsilon))\,. \tag{156}$$

Discarding the constant term e^h, we can write

$$Z_{\mathbf{1}} \propto \sum_{s \in \{\pm 1\}^V} \prod_{(vw) \in E} e^{J s_v s_w}\,. \tag{157}$$

This is the partition function of the Ising model on the square lattice – the same derivation holds for arbitrary graphs. If $\epsilon < 1/2$, $J > 0$ (ferromagnetic) and if $\epsilon > 1/2$, $J < 0$ (antiferrogmanetic). If E is non-trivial, this will be modified to

$$Z_E = \sum_{c \in \mathbb{F}_2^V} \prod_{(vw) \in E} p_1(X^{c_v + c_w} E_{vw})\,, \tag{158}$$

so that now the coupling J between spins s_v, s_w depends on E_{vw}: if $E_{vw} = \mathbf{1}$ it is J, if $E_{vw} = X$, then ϵ is exchanged with $1-\epsilon$, which means $J \leftrightarrow -J$. If we introduce the variable $\eta_e = 1$ if $E_e = \mathbf{1}$ and $\eta_e = -1$ if $E_e = X$, we can write

$$Z_E \propto \sum_{s \in \{\pm 1\}^V} \prod_{(vw) \in E} e^{J \eta_{vw} s_v s_w}\,, \tag{159}$$

which is the partition function of the disordered Ising model. The spins tend to align when $\eta_{uw} = 1$ and anti-align when $\eta_{uw} = -1$. Recall that E represents a path on the dual lattice. The configurations of the Ising spins can be thought of as domain walls connecting Ising vortices as the boundary of E, see figure 14.

At this point we can use techniques from statistical mechanics to compute Z_E. Using the transfer matrix method and rewriting it as the expectation of a Gaussian fermionic state one can compute Z_E exactly for the surface in $O(n^2)$ time [BSV14].

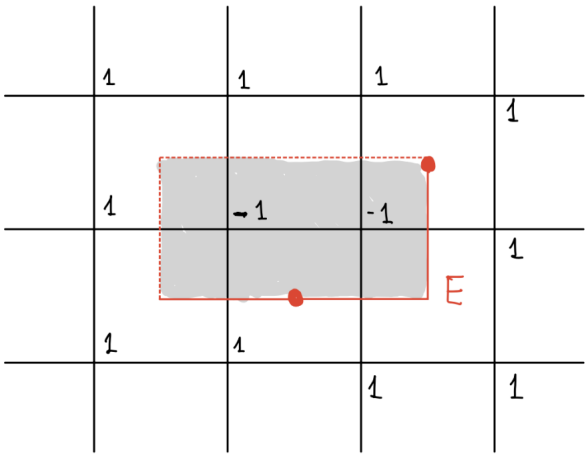


Figure 14:

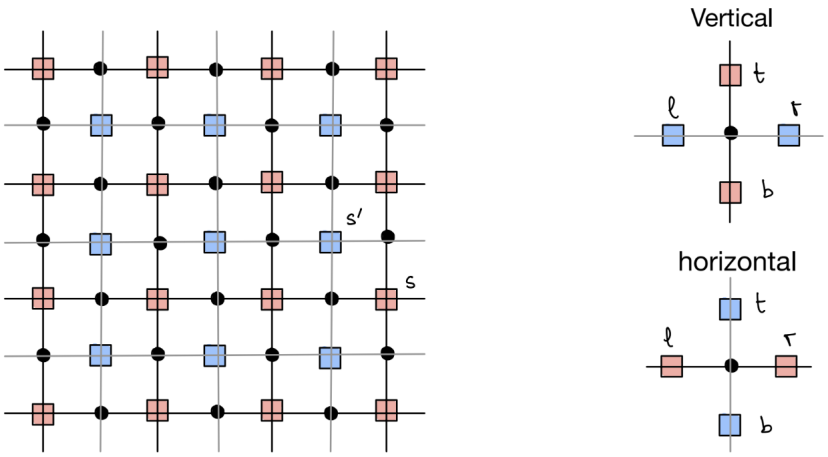


Figure 15:

In a similar way we can associate to a more general noise distribution a statistical mechanical model. Let us consider now a generic iid noise on each qubit:

$$p(E) = \prod_{i=1}^{n} p_1(E) \tag{160}$$

Now Z errors can occur. We consider again the toric code and the union of the lattice and its dual, which corresponds to the Tanner graph of the CSS code, see 15.

Now since the dual graph is isomorphic to the original graph, we introduce the variables c_u at vertex u and c'_u at dual vertex (plaquette) u, and $s = (-1)^c, s' = (-1)^{c'}$. Assuming again to start that $E = \mathbf{1}$, on vertical edges, the interaction is

$$p_1(X^{c_t+c_b} Z^{c'_l+c'_r}) = \begin{cases} p_I & \text{if } c_t = c_b, c'_l = c'_r \\ p_X & \text{if } c_t \neq c_b, c'_l = c'_r \\ p_Y & \text{if } c_t \neq c_b, c'_l \neq c'_r \\ p_Z & \text{if } c_t = c_b, c'_l \neq c'_r \,. \end{cases} = \exp(h + J s_t s_b + J' s'_r s'_l + U s_t s_b s'_r s'_l) \tag{161}$$

$$h = \frac{1}{4}\log(P_I P_X P_Z P_Y)\,, \quad J = \frac{1}{4}\log\left(\frac{P_I P_Z}{P_X P_Y}\right), \quad J' = \frac{1}{4}\log\left(\frac{P_I P_X}{P_Z P_Y}\right), \quad U = \frac{1}{4}\log\left(\frac{P_I P_Y}{P_X P_Z}\right). \tag{162}$$

On horizontal ones:

$$p_1(X^{c_l+c_r}Z^{c'_t+c'_b}) = \exp(h + Js_ls_r + J's'_ts'_b + Us_ls_rs'_ts'_b) \tag{163}$$

This corresponds to the Ashkin-Teller model, which is a model of 2 coupled Ising models. However the couplings are not integrable.

More generally, one can derive more complicated models when the noise probability is factorised in more complicated ways than independent noise:

$$p(E) = \prod_f p_f(E_f)\,. \tag{164}$$

Now we consider a noise model with a single parameter ϵ, for example the depolarizing noise rate. We define the threshold of a family of codes ϵ_t as

$$\lim_{n\to\infty} P_{\text{succ}} = \begin{cases} 1 & \text{if } \epsilon < \epsilon_t \\ 2^{-2k} & \text{if } \epsilon > \epsilon_t \end{cases}. \tag{165}$$

This is defined wrt the maximum likelihood decoder. This means that there exists a critical value ϵ_t such that below threshold the decoder always succeed. Above the threshold the noise is too high and the decoder fails, giving the worst case performance of a uniform distribution over the logical classes.

We want to relate the threshold to a phase transition in the statistical mechanical model. We consider the following disorder parameter given by the free energy cost of a non-trivial logical operator:

$$\Delta_L(E) = -\log Z_{EL} + \log Z_E \tag{166}$$

with L is one of the 2^{2k} logical operators. For topological codes where the logical operators correspond to string-like defects, this corresponds to the free energy of a domain wall. We consider now ϵ_c the critical point where this disorder parameter exhibits a phase transition, and we want to show that $\epsilon_t = \epsilon_c$. We can indeed easily show that

$$\Delta_L = \mathbb{E}(\Delta_L(E)) \tag{167}$$

goes to ∞ below threshold, which corresponds to an ordered or ferromagnetic phase where it is infinitely costly to create a domain wall. We note that E plays the role of quenched disorder.

To show that $\Delta_L \to \infty$ for $\theta < \theta_t$ and non-trivial L we note that the average over E is over functions that depend only on the equivalence classes $[E]$ over the stabiliser. We can pick representatives of those classes as $\{D_\sigma L\}_{\sigma,L}$ where $D_\sigma = T_\sigma L_*(\sigma)$ is the optimal recovery of the maximum likelihood decoder. Then call $p(L,\sigma) = Z_{D_\sigma L}$ so that

$$\Delta_L = -\sum_E p(E)\log\frac{Z_{EL}}{Z_E} = \sum_\sigma\sum_{L'} p(L',\sigma)\log\frac{p(L,\sigma)}{p(LL',\sigma)} \tag{168}$$

Now use the inequality

$$\sum_i a_i \log\frac{a_i}{b_i} \geq a\log\frac{a}{b}\,, \quad a = \sum_i a_i\,, \quad b = \sum_i b_i\,, \tag{169}$$

so that applied to $\sum_\sigma$:

$$\Delta_L \geq \sum_{L'} p(L')\log\frac{p(L)}{p(LL')} \tag{170}$$

Now recall that for $\theta < \theta_t$, $p(\mathbf{1}) \to 1$ since it is the maximum likelihood decoder and $p(L) \to 0$ for $L \neq \mathbf{1}$. Then we have

$$\Delta_L \geq \sum_{L'} p(L')\log p(L) - \sum_{L'} p(L')\log p(LL') = 0 - \log p(L) = +\infty\,. \tag{171}$$

A more precise discussion of the above threshold regime can be found in [CF21].

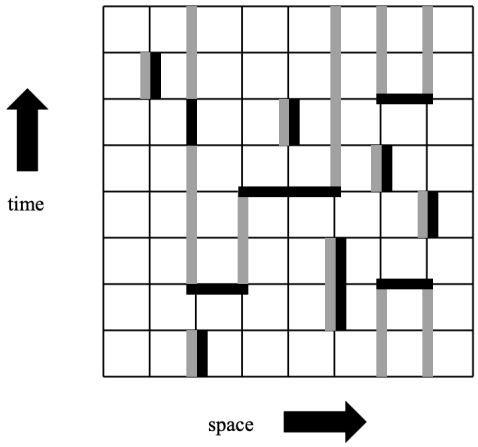


Figure 16:

3.4 Noisy syndrome measurements

In practice the syndromes we measure are noisy. The basic idea to deal with noisy syndromes is to measure the syndrome many times, for say T cycles and use the collected syndromes at different times to decode. We can again map the probabilities of different logical errors to statistical mechanical models, this time in one dimension higher, where the extra dimension corresponds to the discrete time of different syndrome measurement. We illustrate now this mapping using the example of the toric code. In practice one needs to take into account the measurement circuits of the syndrome which can all be faulty. To simplify the discussion we assume here a simple error model for the syndrome where the measurement outcome gets flipped with some probability p. We can think about this as an ideal measurement of the ancilla qubit followed by a random bit flip with probability p. We shall also assume phase flip noise on the data qubits that is distributed iid in both space and time, also with error probability p. What are the configurations of data errors that are compatible with the noisy syndromes? To answer this questions it is useful to think about the system in space-time. We stack the toric code lattice T times and a time slice t holds a configuration of the data errors E_t that can be represented as error chains on the lattice. These are the horizontal links of the spacetime 3D lattice. Vertical links of the lattice hold syndrome measurement errors F_i and are attached to vertices of the lattice. See figure 16 which shows a picture for the 1D version of the toric code, the repetition code.

An error configuration E now has both a horizontal component and a vertical component and thus can be associated with chains in the 3D lattice. The decoder's task is to identify an error in space-time. Let us now define the syndrome chain S by marking all the vertical links where the syndrome is non-trivial (grey in figure 16). The boundaries of the syndrome chains are those where the syndrome measurement changes and clearly the error given by S has the right syndrome. Basically it explains σ by saying that all non-trivial syndromes are due to measurement errors. Let us assume that the number T of measurement cycles is very large so that we can imagine that wordlines of defects are closed loops. Any other error E compatible with the syndrome measurements (black in figure 16) have necessarily the same boundary as the syndrome chain S and therefore we can write $E = S+C$ with C a cycle. This falls into the pattern of our previous discussion where S is a representative of the stabiliser equivalence class in space time and any other error can be obtained from S by adding elements which commute with the parity check matrices, ie cycles. We can divide these error cycles into equivalence classes based on their homology, ie whether they wrap around the torus or not, which as we know correspond to logical errors. Denoted σ the sequence of all syndrome measurements, we can compute the logical probabilities as before by summing over all cycles with homology h_L determined by the logical operator L:

$$p(L|\sigma) = \frac{\sum_{C\in h_L} p(S+C)}{\sum_C p(S+C)}\,. \tag{172}$$

Now maximum likelihood decoding will correspond to maximise the homology class of the errors where

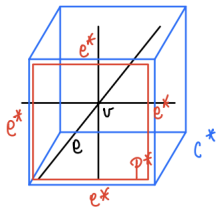


Figure 17:

we sum over cycles over the 3D lattice.

We can map these logical probabilities to a statistical mechanical model as in the case of noiseless syndrome as we now describe. Due to our choice of noise, an error chain E' has probability:

$$\prod_e (1-p)^{1-n_{E'}(e)} p^{n_{E'}(e)} \tag{173}$$

where e is a link of the 3D lattice, both horizontal and vertical. $n_E(e) = 1$ if $e \in E$ and $n_E(e) = 0$ if $e \notin E$. Let E be an error compatible with the syndrome and let us compute the sum over cycles that have the same homology class of E. We consider the chain $E' = E + C$ with C a homologically trivial loop in 3D, and note that $n_{E'}(e) = n_E(e) + n_C(e) \mod 2$. If $n_E(e) = 0$ the probability at edge e is proportional to

$$\left(\frac{p}{1-p}\right)^{n_C(e)} \tag{174}$$

instead if $n_E(e) = 1$, the probability at edge e is proportional to

$$\left(\frac{1-p}{p}\right)^{n_C(e)} \tag{175}$$

Thus we can write

$$Z_E \propto \sum_C \prod_e \exp(J_e u_e)\,, \quad u_e = 1 - 2n_C(e)\,, \quad \exp(-2J_e) = \begin{cases} p/(1-p) & e \in E \\ (1-p)/p & e \notin E \end{cases}. \tag{176}$$

Since C is a homologically trivial cycle, u_e satisfies

$$\prod_{e \ni v} u_e = 1 \tag{177}$$

which means that the number of edges around site v where $u_e = -1$ is even. We can solve this condition by going to the dual lattice where we associate to each edge a dual plaquette P^* and to the vertex v a cube C^* so that the condition becomes

$$\prod_{P^* \in C^*} u_{P^*} = 1 \tag{178}$$

which can be solved by

$$u_{P^*} = \prod_{e^* \in P^*} s_{e^*}\,, \tag{179}$$

where s_{e^*} is an Ising variable on the dual edges around the plaquette P^*, see figure 17.

Thus the statistical mechanical model is a random plaquette $\mathbb{Z}_2$ gauge theory in 3D:

$$Z_E \propto \sum_{\sigma_{e^*}} \exp\left(J \sum_{P^*} \eta_{P^*} u_{P^*}\right), \quad \eta_{P^*} = \begin{cases} 1 & P^* \in E \\ -1 & P^* \notin E \end{cases}, \quad e^{-2J} = p/(1-p)\,. \tag{180}$$

Here P are plaquettes and u_p is the product of the four Ising variables on the links around p. Again, studying the phase transition of the statistical mechanical model allows us to determine the threshold of the optimal decoder associated to this code and noise.

3.5 References

CSS codes are described in [NC01]. Representations of CSS codes in terms of chain complexes are discussed for example in [BE21]. Topological codes are discussed in chapter 19 of [LB13]. See also the original paper [Kit03]. More on topological phase and anyons can be found in [Kit06]. The toric code in terms of hypergraph product can be found in [TZ14]. For the stat mech mapping see [DKLP02, CF21, BAO+12]. For recent development in LPDC codes see [BCG+24].

4 Decoding algorithms

The maximum likelihood decoder is computationally intractable. Therefore we need to devise approximate and fast decoders for practical purposes. We shall see later the requirements in terms of latency when we consider real time decoding.

4.1 MWPM

The simplest decoder for the toric or surface code is the minimum weight perfect matching (MWPM) decoder. The basic idea is that chains with low weight – ie that affect a lower number of edges – are more likely and the MWPM decoder tries to find those by pairing syndromes together in order to achieve the pairings that result in the lowest weight. Let us assume that we have only Z errors. The MWPM decoder assumes that each single Z error creates two defects, ie neighbouring vertices that have a non-trivial syndrome. While this is true for the toric, for the surface code a single Z error at the rough boundary anticommutes with a single vertex operator, and thus we need to add boundary nodes that are all identified and which will produce a defect for single Z nodes at the boundary. Given a syndrome, MWPM constructs the syndrome graph: it is a complete graph with a node per defect and each edge has weight corresponding to the shortest path on the original graph (square lattice) between the two nodes. Note that finding the shortest path between two nodes can be done in time $O(N\log(N)+M)$ for a graph with N nodes and M vertices, so it is efficient. Now we call a perfect matching, that is a matching which is a set of edges such that no two edges share a common vertex, and that involves all vertices. Then we find the minimum weight perfect matching. This problem has an efficient algorithm with time complexity $O(NM\log(N))$. For each matching (u,v) the shortest path between u and v is then the error that our decoder outputs. The total runtime of the algorithm is $O(n^3\log(n))$ which is the runtime of the perfect matching algorithm for a syndrome of size $O(n)$. In practice this can runs in time almost linear in the number of qubits. In the general case of both X and Z errors, MWPM simply treats both errors independently and applies the matching procedure to both syndrome graphs.

MWPM has threshold $p_t = 0.15$ for depolarising noise with parameter p, which is pretty good performance. However this is not optimal for two reasons. First, it treats X and Z errors independently since it is agnostic to the error model. This can lead to the problem of misidentifying a Y error as illustrated in figure **??**. Second, as already discussed minimum weight as well as minimum energy decoders fail to account for the equivalence of errors and use the wrong probability to identify the most likely errors: most probable errors instead of most probable logical class, as done by the maximum likelihood decoder.

figure of threshold computation as critical phenomena with collapse of curves.

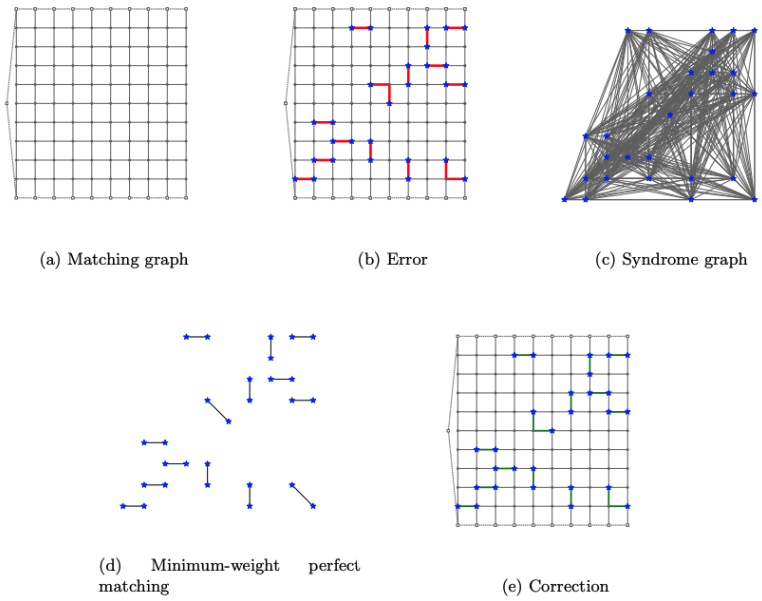
4.2 Tensor network

The tensor network decoder tries to remedy these issues by performing approximate computation of the logical class probabilities by using approximate tensor network algorithms which we now briefly discuss.

To start we introduce the notion of matrix product state. A matrix product state is a tensor $|\psi\rangle \in (\mathbb{C}^2)^{\otimes n}$ such that

$$\langle s_1,\dots,s_n|\psi\rangle = \sum A^{s_1}_{\alpha_1}[1]A^{s_2}_{\alpha_1\alpha_2}[2]\cdots A^{s_{n-1}}_{\alpha_{n-2}\alpha_{n-1}}[n-1]A^{s_n}_{\alpha_{n-1}}[n]\,. \tag{181}$$

Here the tensor $A^s_{\alpha\alpha'}[i]$ have one physical index $s\in\{0,1\}$ and two auxiliary indices $\alpha,\alpha\in\{1,\dots,\chi\}$ where χ is called the bond dimension of the MPS. MPS can represent thus high dimensional tensor



(a) Matching graph (b) Error (c) Syndrome graph

(d) Minimum-weight perfect matching (e) Correction

Figure 1: Stages of the minimum-weight perfect matching decoder for a distance 10 surface code.

Figure 18: From [Hig21].

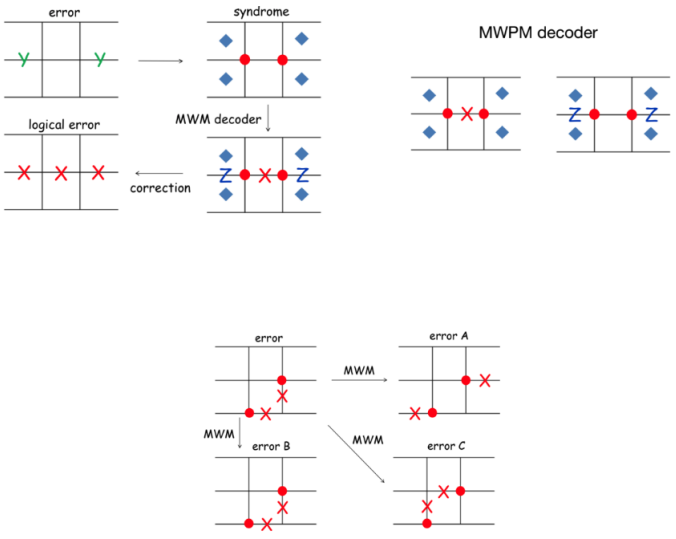


Figure 19: From [qec]. Top: The MWPM does not identify two Y errors and leads to a logical X error – note that X and Z operators are inverted wrt our conventions. Bottom: Three chains with the same weight and MWPM returns one of the three breaking ties at random. However B and C are related by a stabiliser and therefore the right probability to account is $P(BorC) = P(A)$ so that returning B or C should be the right choice instead of A.

with only a number of parameters linear in the size of the number of spins or qubits and are routinely applied to model wavefunctions of quantum many body systems. The bond dimension is directly related to the entanglement of the state, so that states with low entanglement can be approximated efficiently with MPS. A remarkable property of MPS is that their inner product can be computed in time linear in n using the transfer matrix method. That is, first we perform the sum over physical variables. This produces in the bulk matrices of size $\chi^2 \times \chi^2$. The boundaries are vectors of size χ^2 and therefore the inner product is simply a chain of matrix vector products. An illustration of an MPS is in figure **??**.

An important routine that one can do to a tensor network is compression, which means reduction of the bond dimension. We illustrate this important routine in the case of two tensors (matrices) of shape (m, χ) and (χ, m') which we call A, B. These are contracted along the bond dimension χ to represent the tensor

$$\psi_{s,s'} = \sum_{\alpha=1}^{\chi} A_{s\alpha} B_{\alpha s'} \,. \tag{182}$$

To perform compression we perform a singular value decomposition

$$\psi_{s,s'} = \sum_{\alpha=1}^{\min(D,D')} U_{s\alpha} \lambda_\alpha V^\dagger_{\alpha s'} \,. \tag{183}$$

Here the columns of U and V are orthonormal and λ_α is non-negative and sorted in their magnitude. If we truncate now the sum to $\chi' < \chi$ by removing the smallest singular values λ_α we have obtained an approximation

$$\psi'_{s,s'} = \sum_{\alpha=1}^{\chi'} U_{s\alpha} \lambda_\alpha V_{\alpha s'} \,. \tag{184}$$

The error is

$$\|\psi - \psi'\|_2^2 = \sum_{s,s'} \sum_{\alpha=\chi'+1}^{\min(D,D')} \lambda_\alpha^2 |U_{s\alpha}|^2 |V_{s'\alpha}|^2 = \sum_{\alpha=\chi'+1}^{\min(D,D')} \lambda_\alpha^2 \,. \tag{185}$$

The compression procedure can applied to reduce the dimensionality of a bond between two arbitrary tensors by first fusing all the other indices of the tensors to s, s, then applying SVD and then unfusing the indices.

With this background we can do describe the boundary MPS algorithm to approximate the logical probabilities. We fix the depolarising noise for the surface code and consider evaluating $Z_{\mathbf{1}}$ as an illustrative example. The other classes can be obtained by modifying the weights of the tensors as explained above in the derivation of the statistical mechanical models.

First we write the Tanner graph of the code associate tensors to both check nodes and variables nodes as in the figure 20. This is a tensor network with tensors the nodes of the Tanner graph and edges of the Tanner graph host tensor indices 0 and 1. To contract it we can see this partition function as the time evolution of an MPS at the boundary. When the bond dimension gets too large, we can truncate using the compression routine explained above. These steps are illustrated in figure 20.

This method can produce very accurate results: for the surface code it gave threshold $p_t = .189$ very close to the theoretical threshold of the maximum likelihood decoder estimated from the disordered statistical mechanical model. Its drawback is that despite being linear in the number of qubits, the computation is expensive and in practice it is too slow to be used for a real time decoder.

4.3 Outlook

So far we have looked at the case of quantum memories where the goal is to protect quantum information for a given time. In quantum computing we are also interested in performing computation, namely acting with logical gates on the encoded qubits. Using quantum codes and performing logical operations and error correction it is possible to perform fault tolerant if the noise is smaller than

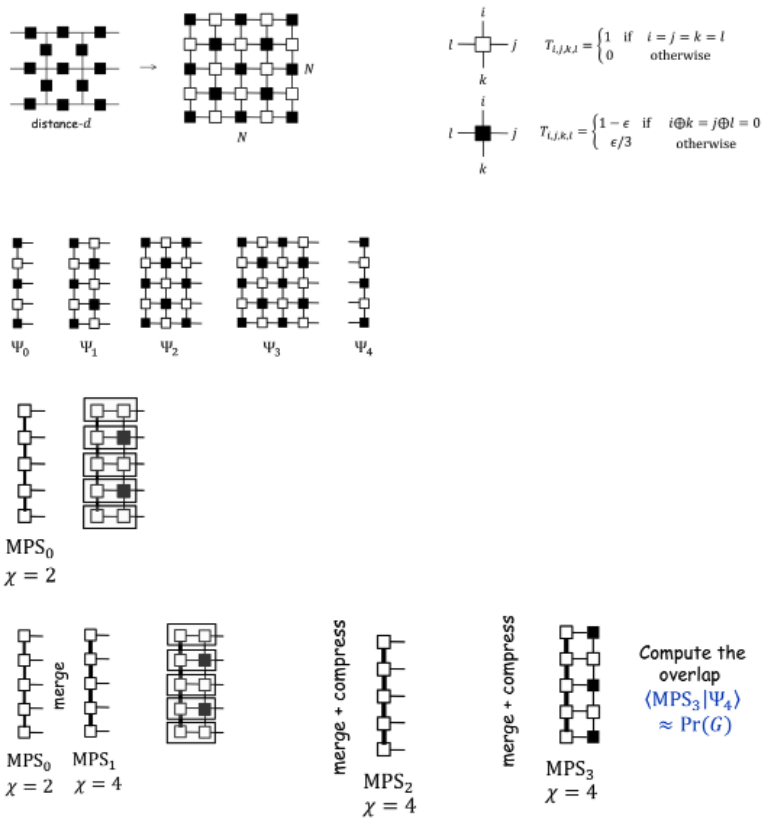


Figure 20: From [qec]. Top: tensor network. Bottom: Boundary MPS contraction of the tensor network representing a logical probability.

some threshold value. This also requires we actively perform error correction and our decoder needs to satisfy stringent latency requirements in order for errors to not backlog. A promising direction to perform efficient decoding is to use neural networks to learn to solve the decoding problem, which can adapt automatically the decoders to the experimental settings.

4.4 References

MWPM is discussed in [FMMC12], tensor network decoders in [BSV14, CF21] and for neural network decoder in [EBW23]. Another popular decoder for the surface code we did not discuss here is the union find decoder [DN21].

References

[BAO+12] H. Bombin, Ruben S. Andrist, Masayuki Ohzeki, Helmut G. Katzgraber, and M. A. Martin-Delgado. Strong resilience of topological codes to depolarization. *Physical Review X*, 2(2), April 2012.

[BCG+24] Sergey Bravyi, Andrew W. Cross, Jay M. Gambetta, Dmitri Maslov, Patrick Rall, and Theodore J. Yoder. High-threshold and low-overhead fault-tolerant quantum memory. *Nature*, 627(8005):778–782, March 2024.

[BE21] Nikolas P. Breuckmann and Jens Niklas Eberhardt. Quantum low-density parity-check codes. *PRX Quantum*, 2(4), October 2021.

[BSV14] Sergey Bravyi, Martin Suchara, and Alexander Vargo. Efficient algorithms for maximum likelihood decoding in the surface code. *Physical Review A*, 90(3), September 2014.

[CF21] Christopher T. Chubb and Steven T. Flammia. Statistical mechanical models for quantum codes with correlated noise. *Annales de l'Institut Henri Poincaré D, Combinatorics, Physics and their Interactions*, 8(2):269–321, May 2021.

[DKLP02] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505, September 2002.

[DN21] Nicolas Delfosse and Naomi H. Nickerson. Almost-linear time decoding algorithm for topological codes. *Quantum*, 5:595, December 2021.

[EBW23] Evgenii Egorov, Roberto Bondesan, and Max Welling. The end: An equivariant neural decoder for quantum error correction, 2023.

[FMMC12] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3), September 2012.

[Haa17] Jeongwan Haah. Algebraic methods for quantum codes on lattices. *Revista Colombiana de Matemáticas*, 50(2):299, January 2017.

[Hig21] Oscar Higgott. Pymatching: A python package for decoding quantum codes with minimum-weight perfect matching, 2021.

[IP13] Pavithran Iyer and David Poulin. Hardness of decoding quantum stabilizer codes, 2013.

[Kit03] A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, January 2003.

[Kit06] Alexei Kitaev. Anyons in an exactly solved model and beyond. *Annals of Physics*, 321(1):2–111, January 2006.

[LB13] D.A. Lidar and T.A. Brun. *Quantum Error Correction*. Cambridge University Press, 2013.

[Mer07] N David Mermin. *Quantum computer science: an introduction*. Cambridge University Press, 2007.

[NC01] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*, volume 2. Cambridge university press Cambridge, 2001.

[qec] S. Bravyi, talk at QEC14. https://www.qec14.ethz.ch/slides/SergeyBravyi.pdf.

[RU08] Tom Richardson and Ruediger Urbanke. *Modern coding theory*. Cambridge university press, 2008.

[TZ14] Jean-Pierre Tillich and Gilles Zemor. Quantum ldpc codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Transactions on Information Theory*, 60(2):1193–1202, February 2014.