TENSOR NETWORK AND QUANTUM ERROR CORRECTING CODES

Thesis

Submitted in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

IN

PHYSICS

by

SOUMYA SARKAR

Reg No: 196016

Roll No: 196PH021

Under the Guidance of

Dr. Deepak Vaid

Department of Physics

National Institute of Technology Karnataka, Surathkal

2019-2021

DECLARATION

I hereby declare that the report for the post-graduate project work entitled "TENSOR NETWORK AND QUANTUM ERROR CORRECTING CODES" which is submitted to National Institute of Technology Karnataka Surathkal, in partial fulfillment of the requirements for the award of the Degree of Master of Science in the Department of Physics, is a bonafide report of the work carried out by me. The material contained in this report has not been submitted to any University or Institution for the award of any degree.

Place: Surathkal
Date: August 2021

SOUMYA SARKAR
Roll No.: 196PH021
Department of Physics
NITK, Surathkal

CERTIFICATE

This is to certify that the project entitled "TENSOR NETWORK AND QUANTUM ERROR CORRECTING CODES" is an authenticated record of work carried out by SOUMYA SARKAR, Roll No.: 196PH021 in partial fulfillment of the requirement for the award of the Degree of Master of Science in Physics which is submitted to Department of Physics, National Institute of Technology Karnataka, Surathkal, during the period 2020-2021.

Dr. DEEPAK VAID
Project Advisor
Assistant Professor
Department of Physics
NITK, Surathkal

ACKNOWLEDGEMENT

I would like to express my sincere thanks to my project advisor Dr. Deepak Vaid, Assistant Professor, Department of Physics, National Institute of Technology Karnataka, Surathkal, for his timely guidance and support. I would like to thank our project coordinator Dr. N. K. Udayashankar, Department of Physics, National Institute of Technology Karnataka, Surathkal for the valuable project opportunity that he provided for me. I thank all those who have helped in the course of this project.

Place: Surathkal
Date: August 2021

Soumya Sarkar
Roll No.: 196PH021
Department of Physics
NITK, Surathkal

ABSTRACT

This project report contains a detailed description of the basics of Quantum Error-Correcting Codes. Quantum computation and technology is something that is believed to be the future of the human lifestyle. The main problem of this technology is its delicacy. In physical world, it's hard to maintain the quantum information error-free for a long time. So the need for error-correction is enormous. So first we will see what is the dependencies of quantum error correction on classical world. Then we see some of the basic error correcting codes, it's mathematical backgrounds, and applications. Then I study the application of Tensor Network which is, now, one of the useful tools to study the Quantum many Body Physics, to deal the error correcting code, especially the Surface code which is one of the most promising quantum error correcting codes towards large-scale Quantum Computer.

Contents

1 Introduction

When we talk about error-correction [1], we, in general, talk about open systems. In all the algorithms or theories when one uses qubits for building states of superposition - the qubits are assumed to be perfect throughout the process and have no interaction with the surroundings - a closed system. But in the context of error-correction, we consider the errors that are produced in a state as they interact with the environment, and sometimes some unnecessary interactions that create those errors or noise. The error is either a bit-flip, a phase-flip, or any arbitrary error (i.e. the linear combination of bit-flip and phase-flip).

Schrödinger equation is something that is used to study any quantum system under consideration. In most cases, we are not concerned with the time-dependent interactions, so we use the time-independent part of it. Using Born-Oppenheimer non-relativistic approximation we have the Schrödinger equation for an isolated N-electron system as

$$H\psi = E\psi, \tag{1.1}$$

Where, E and $\psi = \psi(x_i)$ are the energy and wavefunction of the system respectively. The hamiltonian operator H is given by

$$H = -\sum_{i=1}^{N} \frac{1}{2}\nabla_i^2 + \sum_{i=1}^{N} v(\vec{r_i}) + \sum_{i<j}^{N} \frac{1}{r_{ij}}, \tag{1.2}$$

where,

$$v(\vec{r_i}) = -\sum_{k} \frac{Z_k}{r_{ik}} \tag{1.3}$$

is the external potential due to nuclei of charge Z_k acting on ith electron. The equation can be solved following certain boundary condition which makes the wave-function well behaved. Condensed Matter physics tries to describe macroscopic properties of a quantum system. These properties arise from their macroscopic structures. However due to exponential scaling it becomes very difficult to solve.

We have hydrogen system that can be solved exactly. This is a one-particle system and we know the calculation is huge. Whereas, when we have a quantum system with more than one particle the situation becomes harder to solve. Helium molecules with only two particles cannot be solved exactly. We need to have approximations, though we use educated guesses in considering approximations. But things get impossible to be solved analytically when the number of the interacting particles is in the order of Avogadro's number. Then we use numerical or use other techniques to solve the Hamiltonian of that system. As the number of particles in a system increases, the dimension of the Hilbert space increases exponentially.

This mean-field theory generally avoids the correlations among the particles. To solve this we often use simplified and effective Hamiltonian, especially in low-energy regimes. As the structure of a solid is very regular, the Hamiltonian can be assumed as particles on a lattice where the positions are fixed and the only degree of freedom for them is their spin. For example, the Hubbard model or Heisenberg model takes only local interactions into account. There are some renowned techniques like Quantum Monte Carlo methods, Hartree-Fock method which is very effective in quantum chemistry. To explore the systems with strong interaction or Entanglement, the Tensor Network technique becomes so handy for a many-body quantum system. In the later part of this thesis, I study this technique and using this calculated the Ground State Energy of a system of a 2D lattice of Rydberg atoms and see how this technique could be helpful in error correction considering error models in surface-17 code.

Here, I will explore the errors that we face while computing and the ways to correct those. To do that, we take the error-correcting ideas from the classical world [2]. And then we try to implement those to correct the quantum errors. And then

I calculate the Ground state energy of a system using a technique called Tensor Network and the relation of this in the context of Quantum Error-Correction. We will first see the basic differences between classical computers and quantum computers in section (3). In section (4), we see the classical way of correcting errors starting with Majority voting and its quantum counterpart including the syndrome measurements and phase flip error. Then we see Shor's 9-qubit code and the discretization of errors in section (5). Next, we explore the visualization of different errors through quantum channels in section (7). In section (8), we see the mathematical conditions for error-correcting codes to be efficient. In section (9), we prove the classical hamming bound and related terms and their quantum counterpart. We see one of the large classes of Quantum error-correcting codes - the CSS (Calderbank-Shor-Stean)[13, 14] code based on the features of Classical linear codes. After this, in the second half, I study the Tensor Network, starting with Spin lattice models in section (10). In section (11), a review of introduction in Tensor Network. Section (12) and (13) review the DMRG process and Rydberg atoms respectively. Then in section (14), I applied the DMRG calculation to find the Ground state of the Rydberg atoms system and provide the energy trend. Then I review the application of Tensor Network contraction in the context of Quantum error-correction, basically fusing the first and second part of this thesis.

2 Scope and motivation

Today's technology is dependent on classical principles. But quantum technology is dependent on quantum principles namely superposition, quantum entanglement, and interference. And quantum technology is very new with good exposure of 15 to 20 years and it is being developed exponentially. So the scope of this technology is huge. Agriculture, communication, security, finance, banking system, etc every aspect of life will be based on quantum technology within the next two decades or so hopefully. And it has the potential to revolutionize our daily life.

Now the motivation of doing the study of error correction is that the quantum bits are very delicate. It gets corrupted very easily while interacting with environment. In fact, this is the main bar that stops us from achieving the *Quantum Supremacy*. So, it's important to know about the types of errors to fix those as much as possible.

On the other hand, to study the many-body system with strong interaction, *Tensor Network technique* is becoming popular nowadays. We also see a review of this and use the DMRG algorithm to see the Ground state energy of a system like mentioned above. It's found that in most of the cases, this technique is useful and efficient in simulating error models in a Quantum Circuits.

3 Literature Survey

Classical bits can be copied, and that makes the major voting possible for basic error correction. But in quantum mechanics, copying a qubit is prohibited according to the No-Cloning theorem [3]. For this, the quantum error correction may look impossible. But Peter Shor showed that with a highly entangled state of ancillary qubits one can perform error correction, in his 9-qubit error-correcting code [4]. In the case of syndrome measurement, one has to measure the bit to diagnose the corrupted bit, this also seems to put an obstacle to correcting errors for quantum data as we know measuring a quantum state makes the state change all the information inside it. But later, it was shown that using CNOT gates one can easily have the syndrome measurement by not disturbing the actual quantum data. The first part of Quantum Error Correction is reviewed based on the book written by M.A. Nielson and L. Chuang [5].

Roger Penrose developed these graphical notations [6] which describe the Einstein notations for tensor operations. The DMRG calculation was first invented by Steven R. White in 1992 [7] to calculate the low energy of a 1D system. Here, I have chosen a particular system of Ryderg atoms whose Hamiltonian is mentioned in this paper [8] and calculated it's Ground State Energy. Then I reviewed the Tensor Network contraction application which was performed by Fang Zhang [9].

4 Classical and quantum computation

In this section, we will see the basic working of Classical and Quantum Computation in a nutshell. Eventually, we can see the differences between their working principles.

4.1 Classical computation:

1. In classical computers or machines we use voltage as our signal. '0' means off and '1' means on. We call it 'Bits', which are the basics of classical computation and information.

2. In this mode of computation, we get only one bit at a time as an output. For example for one-bit system, we get either '0' or '1' at a time.

3. Classical technology follows Moore's law[10] which says that the processing power doubles every 3 or 4 years.

   Growth: 2^1, 2^2, 2^3,. . . .

4. As the number of bits N goes to infinity the computation becomes somewhat impossible. According to thermodynamics, every computation is analogous to an engine cycle.

   The efficiency is $\eta = \frac{W}{Q_{in}} = \frac{Q_{in} \check{} Q_{out}}{Q_{in}}$.

   Let's assume if δQ is the heat dissipated by an element of the computer per unit time.

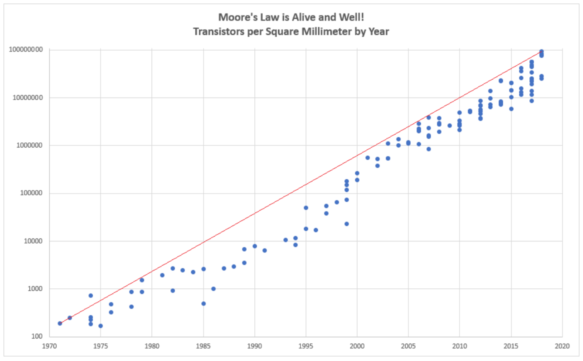


Figure 1: Plotting of Moore's law

$\therefore$ total waste $\rightarrow \Delta Q_{waste} \rightarrow N\delta Q$

So, we've to have a good cooling system to make the extra heat go away from the machine. Otherwise, it affects the other components. So the restriction in mathematical term is $\frac{\Delta Q_{out}}{\Delta t} \leq \frac{\Delta Q_{in}}{\Delta t}$. One doesn't follow the restriction and makes his or her machine crash!!!

5. If the order of a problem increases like $\mathcal{O}(N)$ or $\mathcal{O}(lnN)$ or $\mathcal{O}(NlnN)$ i.e. polynomial – it can be solved using classical computers. But, if the order increases exponentially i.e. $\mathcal{O}(exp(N))$, it cannot be solved by classical computers efficiently.

6. From an N bit classical system one can have N bit of classical information.

4.2 Quantum computation:

1. In Quantum Computers we use any two-dimensional quantum system like photons, electron spin, NMR [11], trapped ions, etc. $|0\rangle$, $|1\rangle$ are what we call the qubits (quantum counterparts of bits) – the building blocks of quantum technology.

2. In this mode of computation, we get all the qubits of a system at once in

superposition manner. For example, for a one qubit system we get $|0\rangle$, $|1\rangle$ at the same time and can be written as

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \tag{4.1}$$

3. Quantum technology follows Nevan's law[12] which says that the advancement of the technology follows doubly exponential growth like 2^2, 2^4, 2^6,.... With this, we would have our today's laptops or smartphones by 1975.

4. In polar coordinate, $C_n = r_n e^{i\phi_n}$, $C_m = r_m e^{i\phi_m}$ $\therefore \rho = |C_n\rangle \langle C_m| = r_n r_m^* exp[i(\phi_n - \phi_m)]$ Here, $\Delta\phi_{nm} = (\phi_n - \phi_m)$ is called *Decoherence*.

   This term is important as the largest problem of the construction of a scalable Quantum Computer is our inability to maintain coherence. Thus the error of a qubit after going through a channel is very high and increases with the number of the quantum gates used.

5. As already shown for a one qubit system, we can also see that for a two-qubit quantum system a state ϕ can be written as

$$|\phi\rangle = \alpha |00\rangle + \beta |01\rangle + \gamma |10\rangle + \delta |11\rangle , \tag{4.2}$$

   Where, α, β, γ and δ are complex numbers in general.

   So, as one can see from the one-qubit quantum system, we get two bits of classical information and from a two-qubit system, we get four bits of classical information. Likewise, in general, from an N qubit quantum system, we get 2^N bits of classical information.

6. According to the previous point, one can easily simulate a system with exponential order, unlike classical ones. So the computation becomes more effective.

One needs exponentially fewer amount of components to do the same calculation done by classical computers. Hence, the speed also increases.

Quantum computation is a different technology as a whole with a different approach to solving problems using the weirdness of nature, hence having the quantum advantages (Superposition, Entanglement, Interference) over the classical world.

5 Basic error correction

The fact that we have started to apply it to solve the real problems so recently and scientists are hoping to get Quantum Supremacy in the very near future. Though Google has done some benchmarking on Quantum Supremacy in 2019 [15]. So this technology is very dynamic and progressing real fast. Now as technology develops we have to be concerned about the errors that arise due to the interactions of the system with its surroundings.

There is a whole different part which deals with this problem and tries to solve it and have the perfect qubits as much as possible using different algorithms. In the first half of the thesis, we will be introduced to such different algorithms and will try to understand the errors first and then solve them. The advancement of quantum technology demands a delicate study of error-correcting codes making it one of the most important parts of quantum technology.

5.1 Majority voting (Bit flip):

Let's think of a bit going through a classical noisy channel with $P > 0$ where P are the probability of bit-flip and $(1 - P)$ is the probability of no error, and it's also given that $P < \frac{1}{2}$.

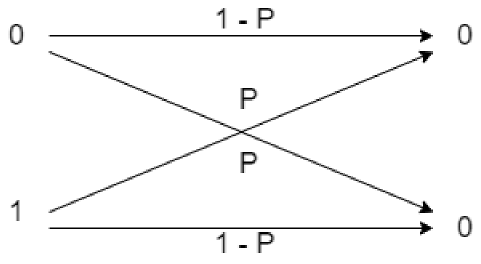


Figure 2: Transition of qubits with their probabilities

Now to detect the error, what we do is we take the bit, make copies of it and map it to three-bit codewords

$$\begin{aligned} 0 &\rightarrow 000 \\ 1 &\rightarrow 111 \end{aligned} \tag{5.1}$$

After going through the noisy channel if e.g. the output is 001, given P is not high, so the input was 000 i.e 0. Likewise, with the same condition, if the output is 011, the input was 111 i.e. 1. If we get the output as 000 or 111, then we conclude that no error has occurred or the channel is too noisy to detect the error.

Probability that two or more bits are flipped is $\binom{3}{2}P^2(1-P) + \binom{3}{3}P^3 = 3P^2(1-P) + P^3 = 3P^2 - 2P^3$ (Given $P > \frac{1}{2}$)

As the method, we discussed above is a classical one, so we could make a copy of the input state. In the quantum case, we take ancillary qubits and make the syndrome measurement to detect the error. We map the qubits $|0\rangle$ and $|1\rangle$ as follows

$$\begin{aligned} |0\rangle &\rightarrow |000\rangle \\ |1\rangle &\rightarrow |111\rangle \end{aligned} \tag{5.2}$$

The circuit for the mapping is

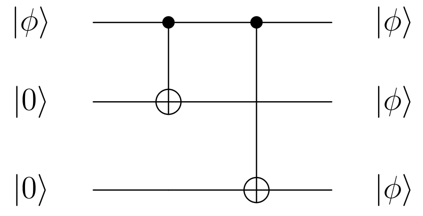


Figure 3: Mapping of quantum state in repetition code for phase flip error

Now let's calculate the probability of bit flip for one or fewer qubit is $\binom{3}{0}(1-P)^3 + \binom{3}{1}P(1-P)^2 = 1 - 3P^2 + 2P^3$ (Given $P > \frac{1}{2}$)

$\therefore$ Error remains with probability $1 - (1 - 3P^2 + 2P^3) = 3P^2 - 2P^3$, which is same as for classical case.

5.2 Error detection (Syndrome measurement)

Pauli-Z gate is used to measure the syndrome and compare the states pairwise. For repetition code we first apply $Z_1 \otimes Z_2 \otimes I$ and then we apply $I \otimes Z_2 \otimes Z_3$, if we get the same eigenvalue for both measurements, we take it as $+1$ and if we get different values, we take it as -1. Different measurement values imply that some error has occurred. With the simultaneous measurements of these two operators, the error on a particular qubit is detected. Then we apply the X-gate on that particulate qubit to get the actual input system.

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{5.3}$$

This gate has the eigenvalues $+1$ and -1 and acts on the qubits as follows $Z|0\rangle = |0\rangle$; $Z|1\rangle = -|1\rangle$

- **Steps:**

  1. Z_1, Z_2, Z_3 gates are connected with each qubit respectively.
  2. Z_1Z_2 and Z_2Z_3 measurements are taken respectively.
  3. If for each pair both measurements are same then we call it 0 and 1 if different.

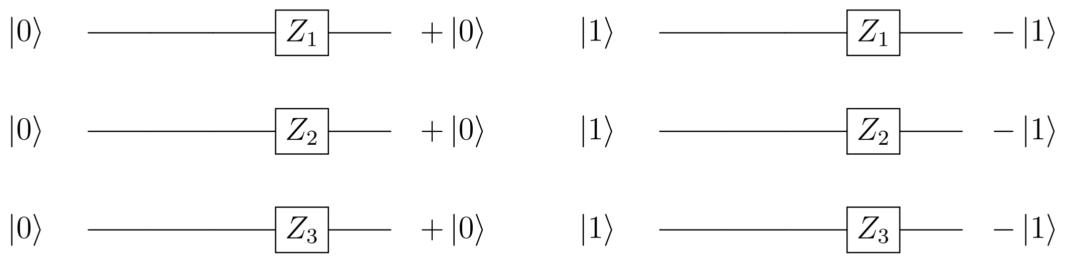

$|0\rangle$ Z_1 $+|0\rangle$
$|0\rangle$ Z_2 $+|0\rangle$
$|0\rangle$ Z_3 $+|0\rangle$
$|1\rangle$ Z_1 $-|1\rangle$
$|1\rangle$ Z_2 $-|1\rangle$
$|1\rangle$ Z_3 $-|1\rangle$
(a) When initial state is $|000\rangle$ (b) When initial state is $|111\rangle$

Figure 4: No error

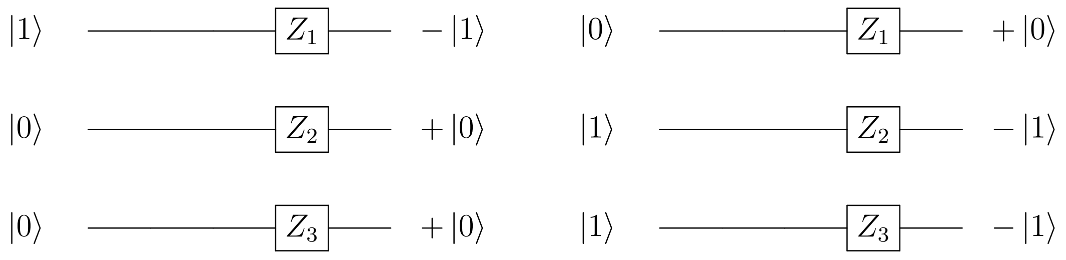
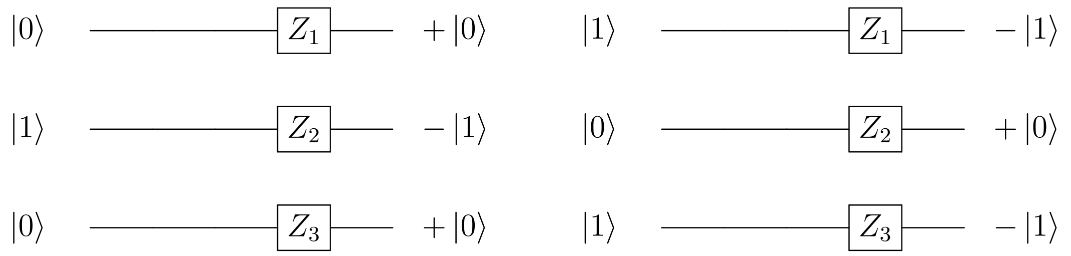
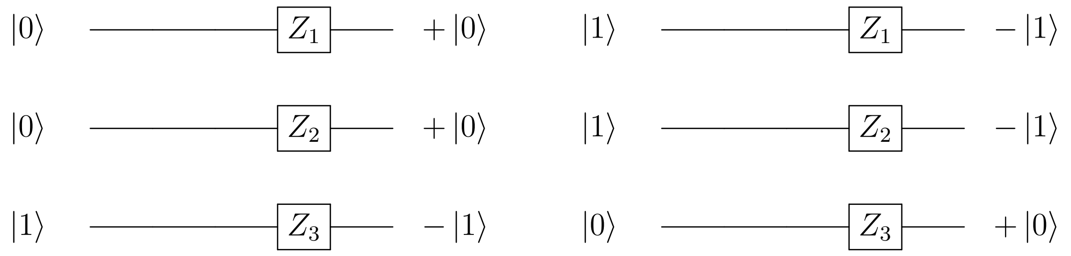

$|1\rangle$ Z_1 $-|1\rangle$
$|0\rangle$ Z_2 $+|0\rangle$
$|0\rangle$ Z_3 $+|0\rangle$
$|0\rangle$ Z_1 $+|0\rangle$
$|1\rangle$ Z_2 $-|1\rangle$
$|1\rangle$ Z_3 $-|1\rangle$
(a) When initial state is $|000\rangle$ (b) When initial state is $|111\rangle$

Figure 5: Error is on 1st qubit



(a) When initial state is $|000\rangle$ (b) When initial state is $|111\rangle$

Figure 6: Error is on 2nd qubit



(a) When initial state is $|000\rangle$ (b) When initial state is $|111\rangle$

Figure 7: Error is on 3rd qubit

The circuits in Fig. (4), (5), (6), (7) are little misleading in the sense we don't measure it in this way. It's just a representation. We measure those qubits separately pairwise. The exact implementation of the syndrome measurement is shown in Fig. (8) where we use controlled-X gates conjugated by H-gates ($HXH = Z$).

If we tabulate the result we would have a better overview of the syndrome measurements

Error	Z_1Z_2	Z_2Z_3	Z_1Z_2	Z_2Z_3
	Initial state $\|000\rangle$		Initial state $\|111\rangle$	
No error	+ + (0)	+ + (0)	- - (0)	- - (0)
1st qubit error	-+ (1)	+ + (0)	+ - (1)	- - (0)
2nd qubit error	+- (1)	- + (1)	- + (1)	+ - (1)
3rd qubit error	++ (0)	+ - (1)	- - (0)	- + (1)

Table 1: All measurement eigenvalues and corresponding syndromes

So what we can see is as follows

1. When both measurements Z_1Z_2 and Z_2Z_3 are zero, there is no error.
2. When $Z_1Z_2 = 1$ and $Z_2Z_3 = 0$ then the error is on 1st qubit.
3. When $Z_1Z_2 = 0$ and $Z_2Z_3 = 1$ then the error is on 3rd qubit.
4. When both measurements Z_1Z_2 and Z_2Z_3 are one, then the error is on 2nd qubit.

The whole circuit for repetition code will look like as follows if we use CNOT gates

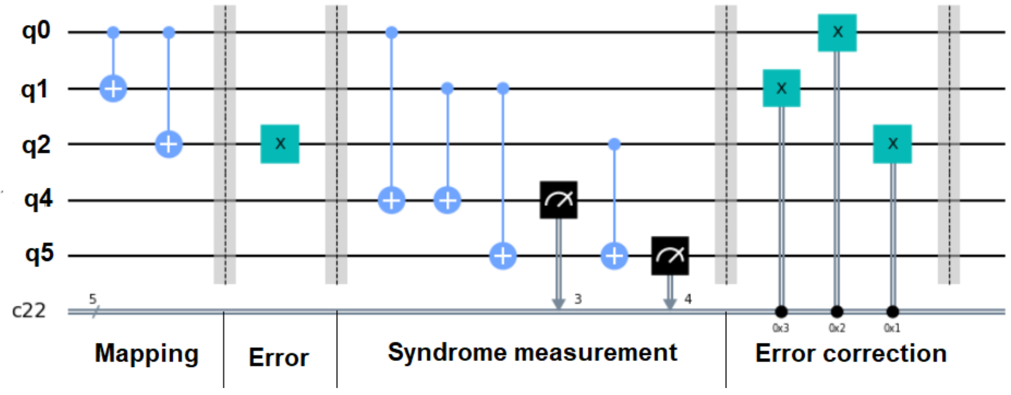


Figure 8: The whole circuit of repetition code in IBM Quantum Experience

Here in this figure, I have taken a particular example of a bit flip on 3rd qubit. $q4$ and $q5$ are called the syndrome qubits. Syndrome qubits are used to measure the syndrome. As we know we can't measure a qubit directly as it changes the state of that qubit, so we take the help of ancillary qubits, in this case, $q4$ and $q5$.

The input state is $|\phi\rangle_1 = |000\rangle$, now we introduce the bit flip on 3rd qubit using X-gate. After being introduced to bit flip the state becomes $|\phi\rangle_2 = |001\rangle$. Then we introduce the syndrome qubit and the state now becomes $|\phi\rangle_3 = |00100\rangle = |001\rangle \otimes |00\rangle$, where the last two qubits are for syndrome measurement. After applying the CNOT gates the final state becomes $|\phi\rangle_4 = |00101\rangle = |001\rangle \otimes |01\rangle$. So at this stage when we measure the two syndrome qubits we get the state $|01\rangle$. To have a better understanding we start with a known initial state. The initial state can be $|111\rangle$ too, then the final state would be $|\phi\rangle_4 = |11001\rangle = |110\rangle \otimes |01\rangle$. The syndrome qubit is independent of the initial state. So this is a general idea irrespective of the initial state. Now, looking at the syndrome state, we know that the error is on the 3rd qubit. we apply X-gate on the same to get the actual initial state.

Syndrome state $|00\rangle$ implies there is no error, $|01\rangle$ implies the error is on 3rd qubit (as shown in the above example), $|10\rangle$ implies the error is on 1st qubit and $|11\rangle$

implies the error is on 2nd qubit. By measuring the syndrome state we detect and eventually correct the particular qubit.

5.3 Phase flip

The bit-flip error also happens in classical computation and we develop the error correction of that particular type of error in quantum computation based on the classical framework. Whereas, the classical world has no analogy of the quantum property of phase. Hence, phase flip error does not happen in a classical case. When we apply Z-gate on $|1\rangle$, it becomes $-|1\rangle$. Hence we say phase is flipped. This type of error shows up in quantum channels. However, phase flip error can easily be turned into a bit-flip error if we change the basis from $|0\rangle$ and $|1\rangle$ to $|+\rangle$ and $|-\rangle$ respectively. We know $|+\rangle$ and $|1\rangle$ can be written as

$$\begin{aligned} |+\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |-\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned} \tag{5.4}$$

With respect to the Z-gate, $|+\rangle$ converts into $|-\rangle$ and vice-versa. So on this basis, the phase-flip error acts like a bit-flip error.

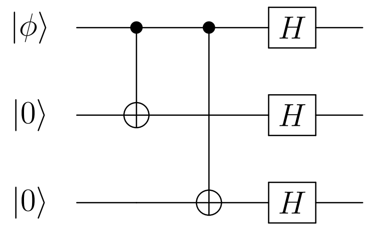


Figure 9: Mapping of quantum state in repetition code for phase flip error

- **Steps:**

    1. First we encode the input state into Hadamard basis, if $|\phi_1\rangle$ is the input state, then after mapping the state becomes $|\phi_2\rangle = H^{\otimes 3}|\phi_1\rangle = \alpha|+++\rangle + \beta|---\rangle$.
    2. Let's assume the error is on the first qubit, the state becomes $|\phi_3\rangle = \alpha|-++\rangle + \beta|+--\rangle$.
    3. Then we decode the state by applying hadamard gate on each qubit and the state becomes $|\phi_4\rangle = H^{\otimes 3}|\phi_3\rangle = \alpha|100\rangle + \beta|011\rangle$.

As one can see, the error now becomes the same as the bit-flip error. Now the same procedure of bit-flip error can be followed to detect and correct the error.

6 Shor's code

Shor's 9 qubit code is a quantum code that protects the qubit from not only bit-flip and phase flip error but also from arbitrary error which is basically a linear combination of bit-flip and phase flip errors.

- **Used for correcting:**

  1. Bit-flip, 2. Phase-flip, 3. Arbitrary error

- **Limitation:**

  It corrects errors only on one qubit.

- **Encoding:**

$$
\begin{aligned}
|0\rangle \to |{+}{+}{+}\rangle &\equiv \frac{(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)}{2\sqrt{2}} \\
|1\rangle \to |{-}{-}{-}\rangle &\equiv \frac{(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)}{2\sqrt{2}}.
\end{aligned}
\tag{6.1}
$$

- **Circuit:**

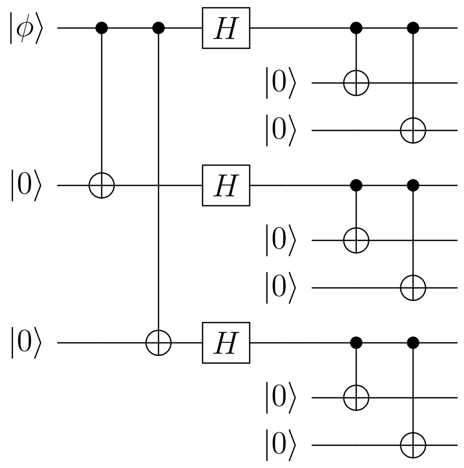


Figure 10: Circuit diagram for encoding in 9-qubit Shor's Code

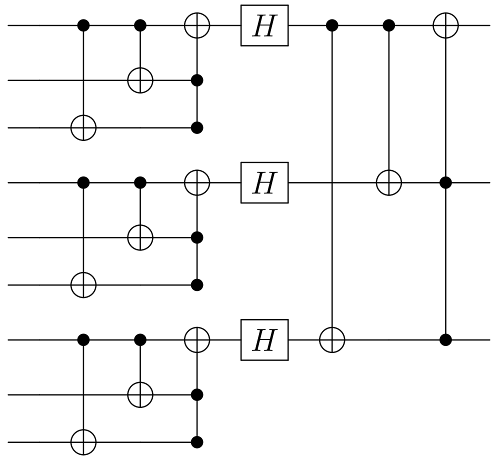


Figure 11: Circuit diagram for decoding in 9-qubit Shor's Code

- **Measurement:**

  1. **Bit flip**

     Like the bit-flip error correction here we take six measurements taking the pairs of Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9 and do the same i.e. 0 for same sign and 1 for different signs. After knowing all bit flip syndromes we inverse them accordingly to correct the error.

  2. **Phase flip**

     Now, if phase flip happens then instead of taking each qubit, we consider a block of qubits because if, say, there's a bit flip on the first qubit then the sign will change for all of the qubits in the first block.

     The phase flip will change a state from $|000\rangle + |111\rangle$ to $|000\rangle - |111\rangle$. Now we compare the sign of the blocks

$$\begin{aligned} &|000\rangle + |111\rangle\,;|000\rangle + |111\rangle \\ &|000\rangle - |111\rangle\,;|000\rangle - |111\rangle \\ \\ &|000\rangle + |111\rangle\,;|000\rangle - |111\rangle \\ &|000\rangle - |111\rangle\,;|000\rangle + |111\rangle\,. \end{aligned} \tag{6.2}$$

The first two equations, as having the same signs, give the syndrome as $|0\rangle$ and the last two give the syndrome $|1\rangle$.

Example: Let's look at an example in which we consider the bit and phase flip on the first bit in the input state a $|0\rangle$ + b $|1\rangle$ and see how the error gets removed using Shor's 9-qubit code step by step.

ENCODING:

$$\mathrm{a}\,|0\rangle + \mathrm{b}\,|1\rangle \xrightarrow{\text{CNOTs}} \mathrm{a}\,|000\rangle + \mathrm{b}\,|111\rangle$$

$$\xrightarrow{H^{\otimes 3}} \mathrm{a}\Big[(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)\Big] + b\Big[(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)\Big]$$

$$\xrightarrow{\text{Ancillary states}} \mathrm{a}\Big[(|000\rangle + |100\rangle)(|000\rangle + |100\rangle)(|000\rangle + |100\rangle)\Big]$$
$$+\,b\Big[(|000\rangle - |100\rangle)(|000\rangle - |100\rangle)(|000\rangle - |100\rangle)\Big]$$

$$\xrightarrow{\text{CNOTs}} \mathrm{a}\Big[(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)\Big]$$
$$+\,b\Big[(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)\Big]$$

INTRODUCING ERROR:

$$\xrightarrow{\sigma_y \text{ error on 1st qubit}} \text{a}\Big[(|100\rangle - |011\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)\Big]$$
$$+ b\Big[(|100\rangle + |011\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)\Big]$$

DECODING:

$$\xrightarrow{\text{CNOTs}} \text{a}\Big[(|111\rangle - |011\rangle)(|000\rangle + |100\rangle)(|000\rangle + |100\rangle)\Big]$$
$$+ b\Big[(|111\rangle + |011\rangle)(|000\rangle - |100\rangle)(|000\rangle - |100\rangle)\Big]$$

$$\xrightarrow{\text{CCNOT}} \text{a}\Big[(|011\rangle - |111\rangle)(|000\rangle + |100\rangle)(|000\rangle + |100\rangle)\Big]$$
$$+ b\Big[(|011\rangle + |111\rangle)(|000\rangle - |100\rangle)(|000\rangle - |100\rangle)\Big]$$

$$\xrightarrow{H^{\otimes(1,4,7)}} \text{a}\Big[\,|111\rangle\,|000\rangle\,|000\rangle\,\Big] + b\Big[\,|111\rangle\,|100\rangle\,|100\rangle\,\Big]$$

$$\xrightarrow{\text{1st,4th,7th qubits}} \text{a}\Big[\,|1\rangle_1\,|0\rangle_4\,|0\rangle_7\,\Big] + b\Big[\,|1\rangle_1\,|1\rangle_4\,|1\rangle_7\,\Big]$$
$$\xrightarrow{CNOTs} \text{a}\Big[\,|1\rangle_1\,|1\rangle_4\,|1\rangle_7\,\Big] + b\Big[\,|1\rangle_1\,|0\rangle_4\,|0\rangle_7\,\Big]$$

$$\xrightarrow{\text{initial state}} \text{a}\,|0\rangle + \text{b}\,|1\rangle$$

So, we reached the same state which we started with despite the introduction of σ_y error in between.

7 Arbitrary error is discrete:

We know that any arbitrary state can be represented as a linear combination of phase flip and bit flip i.e. Arbitrary flip = α(Phase-flip) + β(Bit-flip) The error operations can be described by the Pauli matrices as follows

Bit flip error

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{7.1}$$

Phase flip error

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{7.2}$$

Phase-bit flip error

$$ZX = iY = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{7.3}$$

No error

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{7.4}$$

Any arbitrary single qubit E can be expressed as a linear combinations of no error (I), bit flip error (X), phase flip error (Z) and phase flip error (iY) – the Pauli Matrices. To understand this let's see the following.

$$\begin{aligned} E &= \begin{pmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{pmatrix} \\ &= a\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + b\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + c\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + d\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \\ &= \begin{pmatrix} a+c & b+d \\ b-d & a-c \end{pmatrix} \end{aligned} \tag{7.5}$$

So, the elements of arbitrary single-qubit E are expressed as

$$\begin{aligned} E_{11} &= a+c \\ E_{12} &= b+d \\ E_{21} &= b-d \\ E_{22} &= a-c \end{aligned} \tag{7.6}$$

Henceforth,

$$\begin{aligned} a &= \tfrac{E_{11}+E_{22}}{2} \\ b &= \tfrac{E_{12}+E_{21}}{2} \\ c &= \tfrac{E_{11}-E_{22}}{2} \\ d &= \tfrac{E_{12}-E_{21}}{2} \end{aligned} \tag{7.7}$$

Therefore, a single qubit error can be written as

$$\begin{aligned} E &= \begin{pmatrix} E_{11} & E_{12} \\ E_{21} & E_{22} \end{pmatrix} \\ &= \frac{E_{11}+E_{22}}{2} I + \frac{E_{12+E_{21}}}{2} X + \frac{E_{11}-E_{22}}{2} Z + \frac{E_{12}-E_{21}}{2} ZX. \end{aligned} \tag{7.8}$$

So, an unnormalised state $|\phi'\rangle$ is a superposition of above four terms.

$$\begin{aligned} |\phi'\rangle = E|\phi\rangle &= \frac{E_{11}+E_{22}}{2} I|\phi\rangle + \frac{E_{12}+E_{21}}{2} X|\phi\rangle \\ &+ \frac{E_{11}-E_{22}}{2} Z|\psi\rangle + \frac{E_{12}-E_{21}}{2} ZX|\phi\rangle. \end{aligned} \tag{7.9}$$

Thus after measuring the error syndromes, $|\phi'\rangle$ collapses into one of the above states, and by operating proper inversion operator we can recover the actual input state $|\phi\rangle$. The above result is very important as it shows that the ***quantum errors are discrete***.

8 Visualisation of different types of errors

To understand the physical interpretation of different errors better, the following geometric figures help to visualize a single bit error. Operation elements: E_0, E_1, where E_0 represents that there is no error with probability P and E_1 represents the particular error with probability $(1 - P)$.

- **Bit flip:**

  For bit flip error

$$E_0 = \sqrt{p}I = \sqrt{p}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{8.1}$$

$$E_1 = \sqrt{1-P}X = \sqrt{1-P}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{8.2}$$

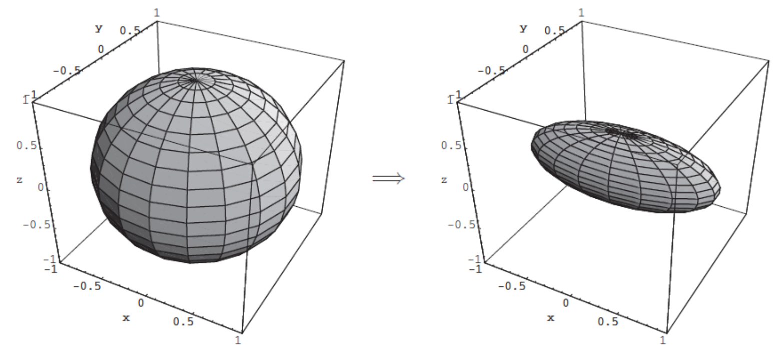


Figure 12: Geometric representation of bit flip error: The contraction is only along Y- and Z-axes for bit flip error[5]

  A point on a Bloch Sphere represents the pure states and any point inside the sphere represents a mixed state.

  So any mixed state is a convex combination of pure states and any state in the interior is a mixed state. Now here we are applying X-gate, so the error operation takes the points from outside in any direction except along the X-axis and maps those insides. Thus this contraction gives a visualisation of this error.

- **Phase flip error:**

  The phenomena is same as before but this time the contraction of the sphere happens except along Z-axis.

$$E_0 = \sqrt{p}I = \sqrt{p}\begin{bmatrix} 1 & 0 \\ 0 & 1. \end{bmatrix} \tag{8.3}$$

$$E_1 = \sqrt{1-P}Z = \sqrt{1-P}\begin{bmatrix} 1 & 0 \\ 0 & -1. \end{bmatrix} \tag{8.4}$$

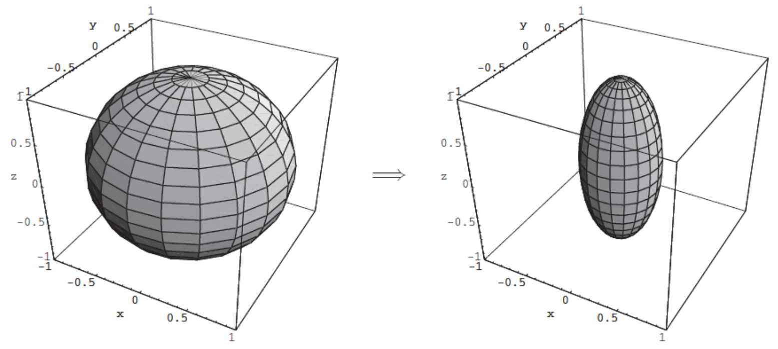


Figure 13: Geometric representation of phase flip error: The contraction is only along X- and Y-axes for phase flip error[5]

- **Bit-phase flip:**

  The contraction happens except along Y-axis.

$$E_0 = \sqrt{p}I = \sqrt{p}\begin{bmatrix} 1 & 0 \\ 0 & 1. \end{bmatrix} \tag{8.5}$$

$$E_1 = \sqrt{1-P}Y = \sqrt{1-P}\begin{bmatrix} 0 & -i \\ i & 0. \end{bmatrix} \tag{8.6}$$

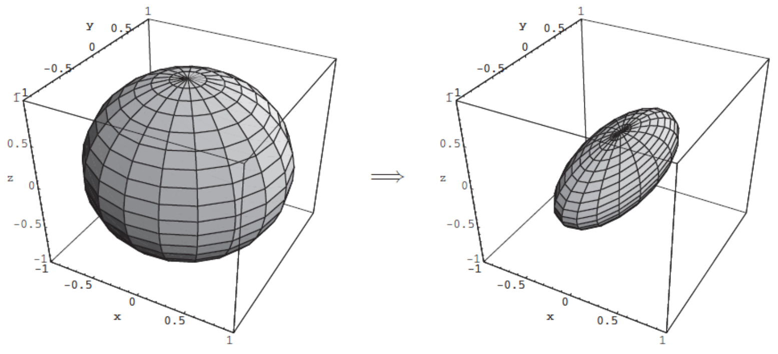


Figure 14: Geometric representation of bit-phase flip error: The contraction is only along X- and Z-axes for bit-phase flip error[5]

- **Depolarizing channel:**

  This is an important quantum noise with probability P that the qubit is depolarised and with probability $(1-P)$ of no error. We see that for this error the sphere contracts with the same amount along all axes.

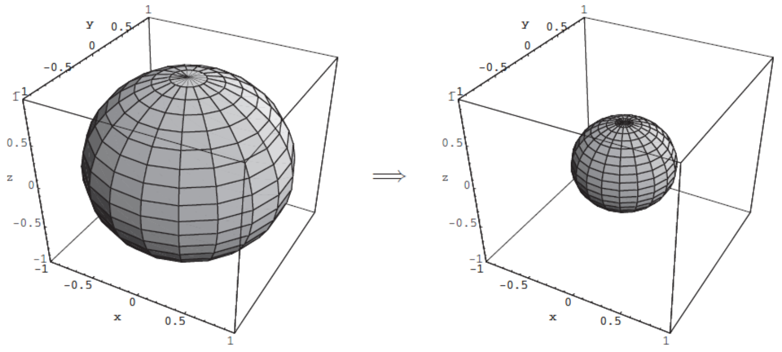


Figure 15: Geometric representation of depolarizing channel: The contraction is along all axes[5]

A depolarizing channel has the operation elements $\left(\sqrt{1-\frac{3P}{4}}, \frac{\sqrt{P}}{2}X, \frac{\sqrt{P}}{2}Y, \frac{\sqrt{P}}{2}Z\right)$. Now if we parameterize the channel conveniently and replace $3P/4$ by P, then the state becomes

$$\epsilon(\rho) = (1-P)\rho + \frac{P}{3}(X\rho X + Y\rho Y + Z\rho Z). \tag{8.7}$$

9 Mathematics behind quantum error-correction

9.1 The conditions

We need a general framework on what we set up the error-correcting codes, one of which is Quantum Error Correcting Conditions. To make error-correction possible one has to satisfy this set of equations.

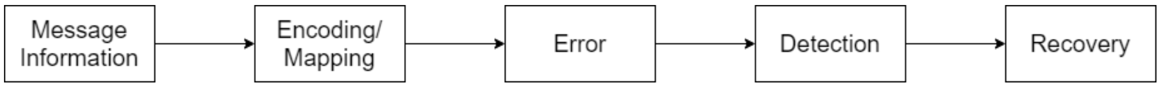


Figure 16: Block diagram of error detection and correction

- **Assumptions:**

  1. Noise is described by ϵ
  2. Error correction procedure is described by trace-preserving error-correction operator R

  So, for error-correction to be successful

  $$(R \circ \epsilon)(\rho) \propto \rho, \tag{9.1}$$

  Where ρ belongs to codeword C. And the sign '$\propto$' would be '$=$' if 'ϵ' is also trace-preserving along with the R operator.

- **<u>The conditions:</u>**

  1. ***Orthogonality:***

     If there is a codeword $|\phi_l\rangle$ which is applied by an error operator E_l and the same codeword applied by another error operator E_k, then $\langle\phi_l|E_l^\dagger E_k|\phi_l\rangle = 0; \forall\ l \neq k; \forall\ l$

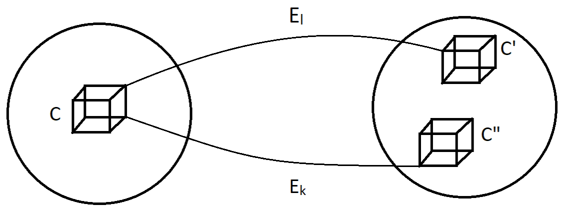


Figure 17: The transformed codewords don't overlap

If I want to take a subspace of codeword C in a Hilbert space and want to change it by applying error operators and get codewords C' and C'', then C' and C'' never overlaps because if they overlap then we can't distinguish the error from each other and can't correct it.

2. ***Non-deformation:***

If an error operator acts on a codeword and transforms it to another codeword then all the vectors inside that codeword would contract with same probability i.e. E_k, then $\langle\phi_l|E_l^\dagger E_l|\phi_l\rangle = d_{kk}\ \forall\ l$.

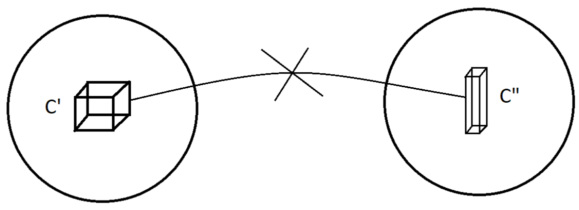


Figure 18: The deformation of transformed codeword is not possible

If the error operator does not act on all the vectors with equal probability in that codeword then there would be a relative shrinking of the subspace and the shape of the codeword gets deformed. Hence we can't have a squared-up Hilbert space, we can't correct the error.

These two conditions can be written in one equation as

$$PE_l^\dagger E_k P = d_k \delta_{lk} P, \tag{9.2}$$

where P are the probability of a qubit to be flipped.

9.2 Improvement in fidelity (Why QECC is important)

In this section, we take the most general error channel i.e. depolarizing channel to show that if we apply error-correction code the fidelity improves. It is a mathematical approach to show that quantum error-correction code works. Before going ahead let's make some assumptions, one is that there are independent errors on different qubits and secondly the noise is sufficiently weak.

Now, we take this example where we see a noisy channel that can be written as the linear combination of I, X, Z, and iY.

Recall the depolarizing channel on a single qubit,

$$\epsilon(\rho) = (1-P)\rho + \frac{P}{3}\left(X\rho X + Y\rho Y + Z\rho Z\right). \tag{9.3}$$

$$\begin{aligned} \text{Fidelity} &= \sqrt{\langle\psi|\rho|\psi\rangle} \\ &= \sqrt{(1-P) + \frac{P}{3}\langle\phi|(X|\phi\rangle\langle\phi|X + Y|\phi\rangle\langle\phi|Y + Z|\phi\rangle\langle\phi|Z)|\phi\rangle} \\ &= \sqrt{(1-P) + \frac{P}{3}(\langle\phi|X|\phi\rangle^2 + \langle\phi|Y|\phi\rangle^2 + \langle\phi|Z|\phi\rangle^2}). \end{aligned} \tag{9.4}$$

If $|\phi\rangle = |0\rangle$

$$\begin{aligned} \text{Fidelity } &= \sqrt{(1-P) + \frac{P}{3}(0+0+1)} \\ &= \sqrt{1 - \frac{2P}{3}}. \end{aligned} \tag{9.5}$$

If $|\phi\rangle = |1\rangle$

$$\begin{aligned}\text{Fidelity} &= \sqrt{(1-P)+\frac{P}{3}(0+0+1)} \qquad (9.6)\\ &= \sqrt{1-\frac{2P}{3}}.\end{aligned}$$

Therefore,

$$F_{\min(E)} = \sqrt{1-\frac{2P}{3}} = 1-\frac{P}{3}+O(P^2). \qquad (9.7)$$

Now, let's encode a single qubit into n-qubit quantum code. Suppose the depolarizing channel acts independently on each of the qubit

$$\mathcal{E}^{\otimes n}(\rho) = (1-p)^n\rho + \sum_{j=1}^{n}\sum_{k=1}^{3}(1-p)^{n-1}\frac{p}{3}\sigma_k^j\rho\sigma_k^j + \cdots . \qquad (9.8)$$

where 'J' is the number of qubits the gate is acting on and 'σ_i' are the Pauli matrices X, Y, Z. For 3-qubit, the quantum state after error correction is $\rho = [(1-p)^3 + 3p(1-p)^2]\,|\phi\rangle\langle\phi| + \cdots$

For n-qubit,

$$\begin{aligned}\rho &= [\binom{n}{0}(1-P)^nP^0 + \binom{n}{1}(1-P)^{n-1}P^1]|\phi\rangle\langle\phi| + \cdots \qquad (9.9)\\ &= [(1-p)^n + np(1-p)^{n-1}]|\phi\rangle\langle\phi| + \cdots .\end{aligned}$$

Therefore fidelity,

$$\begin{aligned} F_R &\geq \sqrt{(1-P)^n + np(1-P)^{n-1}} \\ &= \sqrt{(1-P)^{n-1}(1-P+np)} \\ &= 1 - \frac{\binom{n}{2}}{2} P^2 + O(P^3). \end{aligned} \tag{9.10}$$

Now as we can see from the Eq. (9.7) and Eq. (9.10), it's clear that after we apply quantum error codes on a corrupted state, the fidelity improves. Thus QECC is important [19].

10 Classes of quantum error correction codes

To have an error-corrected qubit, one cannot just have any error-correcting code to correct a message of a particular dimension by encoding it to a higher dimension. So, one needs some limitations on the parameters of a code to correct errors efficiently. A error-correction process can be written as $(Z\,|\phi_i\rangle \otimes |A\rangle \to |\phi_i\rangle \otimes |A_Z\rangle)$, where $Z\,|\phi_i\rangle$ is altered coherent error state, Z is a linear transformation and needs not to be unitary and A is the ancillary qubit. The same transformation can be written for superposition states too,

$$\left(Z \sum_{i=1}^{2^k} c_i\,|\psi_i\rangle \right) \otimes |A\rangle \longmapsto \left(\sum_{i=1}^{2^k} c_i\,|\psi_i\rangle \right) \otimes |A_Z\rangle\,. \tag{10.1}$$

But the transformation $Z \to |A_Z\rangle$ is to be linear. If the mapping is one-to-one, then it's called **non-degenerate code**, if not then it's called **degenerate code** which is not known in classical codes. Let's take the 9-qubit Shor's code for example. In Shor's code, we see that the phase errors within a block of 3 qubits act indifferently whereas in classical cases, errors on different bits arise from different codewords.

10.1 Classical Hamming Bound

Hamming bound is a type of non-degenerate bound and before looking at quantum hamming bound, let's understand it classically first. To do that one must know some terms regarding this.

- **Hamming distance**

  The number of mismatches between two strings at the same position is called the Hamming distance.

  Let's take an example of two words 'work' and 'walk', so at the first and last positions, both letters are the same i.e. 'w' and 'k' respectively. In the second and third positions, the letters are different for the words. So the Hamming distance between the two words is 2. Likewise, for two strings e.g '1001101' and '1111001' – the Hamming distance is 3 as the binary numbers of the codewords are different at second, third, and fifth positions. Now, we know that an XOR gate gives an output '1' if two inputs are different, so we can use an XOR gate two calculate the Hamming Distance between two codewords – the number of '1's in the output tells you the Hamming Distance.

$$\begin{array}{cccccccc} & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ \oplus & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ \hline & 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{array} \tag{10.2}$$

  In quantum computation, CNOT gate acts as a classical XOR gate. If the Hamming distance is zero, then there is no impurity.

- **Hamming Weight**

  The number of non-zero (i.e. 1 for a binary codeword) digits in a codeword is called the Hamming Weight of that codeword.

  W(111) = 3

  W(101101) = 4

- **Hamming Bound**

  Let's consider, as mentioned before, to detect or correct a code, we generally map the dimension of a message bit to a higher dimension. A message of 'k'-bit if encoded to n-bit, where 'k' is less than 'n', then the space of the message bit is a 2^k-dimensional subspace of a Hilbert Space of 2^n-dimension and we take the minimum hamming distance between two codewords is 2. The code is thus called a $[n, k, d]$ code.

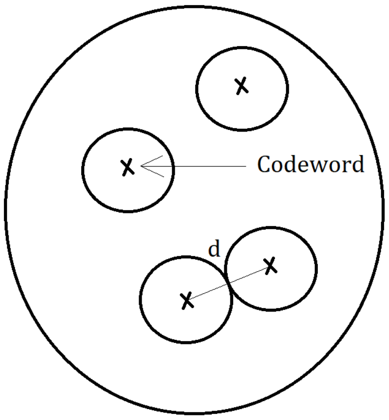


Figure 19: Pictorial illustration of Hamming Bound

So the picture above gives a very good illustration of Hamming Bound. The bigger sphere is a 2^n space in which I have my valid codewords. Now I can make a circle or a sphere around each codeword with radius 't'. The physical significance of this is 't' is the number of maximum errors inside the Hamming spheres (The smaller spheres). As 'd' be the minimum distance between two codewords, we can choose $2t < d$ as no two codewords overlap with each other for being non-degenerate. But, later, we will see that we choose the 't' as

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor . \tag{10.3}$$

Note that, the operator is a 'Floor function'. Now, the number of ways we can choose a vector with 't' or less error from the n-dimensional space is

$$\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + ... + \binom{n}{t}, \tag{10.4}$$

where the first term is the codeword itself and $\binom{n}{i}$ $(i \leq t)$ represents the number of ways to choose the vectors which differ from the codeword by 'i'.

We have total of 2^k codewords inside the hamming spheres with error 't' or less from a 2^n Hilbert Space, so the condition is

$$2^k \left[\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + ... + \binom{n}{t} \right] \leq 2^n. \tag{10.5}$$

For a (7, 4, 3) Hamming code, the generalised set of parameters is $(2^r - 1, 2^r - r - 1, 3)$. For $r \geq 2$ this code satisfies the hamming bounce, hence can detect or correct errors efficiently. This is the insight of Hamming bound.

- **Perfect Codes**

  When the number of vectors inside the hamming spheres is equal to the total number of vectors in the Hilbert space, then it's called a Perfect Code [20].

$$2^k \left[\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + ... + \binom{n}{t} \right] = 2^n. \tag{10.6}$$

  - **Ex 1.**

    Let's take, $n = 3$, $k = 1$, $d = 3$
    $\therefore t = \left\lfloor \frac{3-1}{2} \right\rfloor = 1$

LHS: $2^1\left(\binom{n}{0}+\binom{n}{1}\right)=2(1+3)8$

RHS: $2^3=8$

So, LHS $=$ RHS and this is a ***Perfect Code***.

- **Ex 2.**

  Let's take, $n=4$, $k=1$, $d=4$

  $\therefore t=\left\lfloor\dfrac{4-1}{2}\right\rfloor=\lfloor 1.5\rfloor=1$

  LHS: $2^1\left(\binom{n}{0}+\binom{n}{1}\right)=2(1+3)=8$

  RHS: $2^4=16$

So, LHS$\leq$ RHS and this is ***not a Perfect code*** despite following Hamming bound.

10.2 Quantum Hamming Bound

Depending on the framework of classical Hamming Bound, quantum Hamming Bound was explored. This is a general property of the set of parameters of a quantum error-correcting code which it has to follow to be efficient for correcting errors. Suppose, 'n' is the number of total qubits, 'k' is the number of qubits to be encoded, 't' is the number of maximum qubits with errors that the code can correct up to and 'j' is the number of errors ($j\leq t$). So, from the total number of qubits, there are $\binom{n}{j}$ sets of locations where the errors can occur. Now, there are three types of basic errors that can happen on quantum bits i.e. bit-flip, phase-flip, and bit-phase flip. The total number of error on 't' or fewer qubits

$$\sum_{j=0}^{t}\left(\begin{array}{c} n \\ j \end{array}\right)3^j. \tag{10.7}$$

As seen before, the message is in the 2^k-dimensional subspace of a 2^n-dimensional Hilbert Space. So the condition must be

$$\sum_{j=0}^{t} \binom{n}{j} 3^j 2^k \leq 2^n. \tag{10.8}$$

This is called Quantum Hamming Bound. So there is no non-degenerate code that encodes 'k' qubits in fewer than 'n' qubits to protect all errors on a 't' number of qubits.

Now like the Hamming bound we have some other bounds like Singleton bound, Gilbert-Varshamov bound, etc. that basically completes the theoretical base depending upon which we can go forward to some actual codes in order to detect and correct errors. Here in this section we will see how to construct a quantum error-correction code called Calderbank-Shor-Steane (CSS) codes but on the classical framework. For that we see the construction of Classical linear code [16].

10.3 Classical linear codes

- **Generator matrix:**

  This matrix is an operator that projects binary message bits to higher-dimensional space. If that projects a k-dimensional binary message bit to an n-dimensional space then the matrix will be $(k \times n)$-dimensional. A couple of examples will make this thing clearer.

  For repetition code, we project 1-dimensional message bit (0 or 1) to a 3-dimensional space (000 or 111). So the generator matrix will be (1×3)-dimensional. And in this case the generator matrix is $G = [111]$. So the transformation follows

$$[0][111] = [000] \tag{10.9}$$
$$[1][111] = [111].$$

Now, another popular quantum code is (7, 4) Hamming code, where the generator matrix projects a 4-dimensional message bit to a 7-dimensional encoded bit. The (n - k) bits are made following the conditions

$$x = a \oplus b \oplus d \tag{10.10}$$
$$y = a \oplus c \oplus d$$
$$z = b \oplus c \oplus d.$$

Now from 7 bits, we can have $2^7 = 128$ total number of codewords among which only 16 are valid codewords that follow the above equations. So the generator matrix for a (7, 4) Hamming code can be constructed as

$$\begin{bmatrix} a & b & c & d \end{bmatrix} \overbrace{\left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]}^{G}_{\begin{array}{cc} \text{Identity matrix} & x \quad y \quad z \end{array}} = \begin{bmatrix} a & b & c & d & x & y & z \end{bmatrix}$$

Generator matrix of Hamming code

'G' is called the *Generator Matrix.*

Now, let's have a look at some of the properties of linear codes

1. All zeroes are always a valid codeword

2. The sum of two valid codewords is also a valid codeword.

   If $C = C_1 + C_2$, and C_1, C_2 are two valid codewords, then C is also a valid codeword.

3. Hamming weight of a codeword C is basically the Hamming distance of all zeroes and the codeword C

   Ex. $D(C, 00000) = W(C)$, where C is a 5-dimensional codeword.

4. Smallest Hamming distance is the minimum weight of the valid codewords among all valid codewords in the code

Now, why do we need those properties? To understand that first, let's have a look at the pictorial representation of minimum hamming distances for different codes.

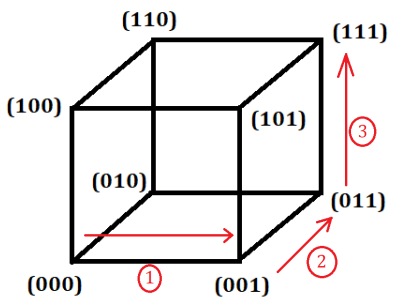


Figure 20: Geometrical representation of (3,1) repetition code

This is for (1, 3) repetition code. Where the two valid codewords are (000) and (111), and how we can go from one valid codeword to another valid codeword – that is by 3 steps. So the hamming distance of (1, 3) repetition code is 3.

For a 4-dimensional code, assume that we have a generator matrix

$$G = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}. \tag{10.11}$$

So, we have the codewords of the message bits as $00 \rightarrow 0000$, $01 \rightarrow 0011$, $10 \rightarrow 1100$ and $11 \rightarrow 1111$. The pictorial representation of this is as follows

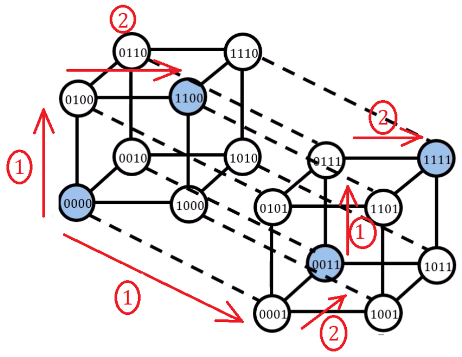


Figure 21: Geometrical representation of a code with Hamming distance 2

So, again, here to go from one valid codeword to another we have to go two steps. So the hamming distance of this code is 2.

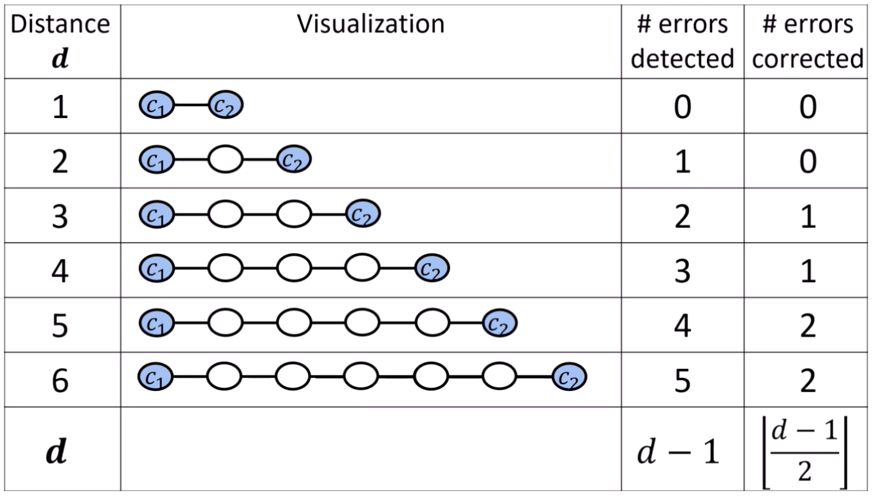
But, for (7, 4) hamming code or any other higher dimensional code, the drawings become so hard to digest and also comparing each codeword to calculate minimum Hamming distance becomes tedious. So here comes the importance of the properties of Linear codes. As the 4th property tells that the hamming distance is just the minimum weight of all the valid codewords in the code, then the hamming distance of the (7, 4) hamming code is 3.

Message	Codeword
0000	0000000
0001	0001111
0010	▨ 0010011
0011	▨ 0011100
0100	▨ 0100101
0101	▨ 0101010
0110	0110110
0111	0111001
1000	▨ 1000110
1001	▨ 1001001
1010	1010101
1011	1011010
1100	1100011
1101	1101100
1110	▨ 1110000
1111	1111111

Table 2: Codewords with minimum Hamming weight 3 in (7, 3) Hamming Code

The grey-colored cells are the ones that have the codewords with a minimum weight of 3, so the minimum distance of the code is 3.

Now, why are we so obsessed with minimum Hamming distance? Because this is needed to calculate the number of errors a code can detect or correct. The following table makes this clear.



Distance d	Visualization	# errors detected	# errors corrected
1	c_1 c_2	0	0
2	c_1 c_2	1	0
3	c_1 c_2	2	1
4	c_1 c_2	3	1
5	c_1 c_2	4	2
6	c_1 c_2	5	2
d		$d-1$	$\left\lfloor \frac{d-1}{2} \right\rfloor$

Figure 22: Pictorial illustration of Hamming Bound https://www.youtube.com/watch?v=as_mNSx6OG8&t=335s

As we can see above, the blue circles depict the valid codewords ad the white ones replicate the codewords with errors. For minimum distance 1, there is no way to detect or correct any error because any error to this moves the information from one valid codeword to another. For minimum distance 2, we can detect one error but as we can tell from which way it comes from i.e. we can't tell from which valid codeword the error comes into existence, so we can't correct it. For minimum distance 3, we can detect two errors and correct 1 error as this time we can tell which error has occurred from which valid codeword, we just inverse that to correct. For minimum distance 4, obviously, we can detect three errors but correct only 1 as we can't tell from which valid codeword the error of the middle white circle comes and so on. In general for minimum distance 'd', we can detect $(d-1)$ number of errors and floored value of $\frac{d-1}{2}$.

- **Transmission efficiency**

  Transmission efficiency of a code is defined by $\frac{\text{Message bits(k)}}{\text{Total bits(n)}}$.

  For (1,3) repetition code, minimum distance d is 3, so it can correct $\lfloor \frac{3-1}{2} \rfloor = 1$.

Likewise, for (3,5) and (4,7) repetition codes one can correct 2 and 3 errors but as one increases the encoded codes the transmission efficiency decreases.

$$\begin{aligned} \text{(1,3) repetition code} &\rightarrow \frac{1}{3} = 0.33 \\ \text{(3,5) repetition code} &\rightarrow \frac{1}{5} = 0.20 \\ \text{(4,7) repetition code} &\rightarrow \frac{1}{7} = 0.14. \end{aligned} \tag{10.12}$$

So, in general, repetition code is not a very good code for correcting errors. On the other hand, for (4,7) hamming code, one can correct only one error but with a transmission efficiency of 0.57, which is much more efficient.

- **Parity check matrix**

The linear code is more efficient and lesser time-consuming in case of error correction. In this code, we use a matrix called 'Parity Check Matrix' H, which is a $(n-k) \times n$ dimension matrix. Every code has a parity check matrix H, for which the valid codewords in that code follow the condition $HC^T = 0$, otherwise, that will be an invalid codeword.

To construct a parity check matrix (say) for (7,4) Hamming code, we follow Eq. (10.10), if we take the LHS to RHS we have

$$\begin{aligned} x \oplus a \oplus b \oplus d &= 0 \\ y \oplus a \oplus c \oplus d &= 0 \\ z \oplus b \oplus c \oplus d &= 0. \end{aligned} \tag{10.13}$$

Now taking all the coefficients we can make a matrix

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}. \tag{10.14}$$

This is the Parity Check matrix for (7,4) Hamming code. To check if a codeword is valid or not, we just have to take the transpose of the codeword and operate the H on the codeword. If we get zero then the codeword is a valid codeword for Hamming code.

Let, $\vec{e}$ is an error on the codeword $\vec{C}$. Now, $H(\vec{C}+\vec{e})^T = H\vec{C}^T + H\vec{e}^T = H\vec{e}^T$. So we get the syndrome vector for the corrupted bit. This syndrome vector, only, depends on the error, not on the codeword. So, we exactly can know what we need to fix irrespective of codeword. In the above methods, we have to look at all the code vectors to check where the error has occurred and that is time-consuming as $(\vec{C}+\vec{e})^T$ are 2^n numbers of vectors. Instead here, as this depends only on the error vector, we have only 2^{n-k} error vectors to look at. On the other hand, it can only correct one error in a codeword. We can also get the minimum Hamming distance and generator matrix for a code. So the concept of the Parity check matrix is very important.

10.4 CSS code (Calderbank-shor-Stean codes)

Based on some properties and results of Classical Linear Codes CSS codes is one of the first large class quantum error correction code. Suppose, C is an [n,k] code which has the generator matrix G and parity check matrix H, then its dual code $C^{\perp}$ will have the generator matrix G^T and parity check matrix H^T. If x are the codewords of code C and y are the codewords of code $C^{\perp}$, then x and y are orthogonal to each other. Dual code is the main feature in the construction of CSS code.

To see how CSS codes work, we need to know two facts. First, $\sum_{b\in C}(-1)^{a\cdot b} = |C|$ if $a \in C^{\perp}$ and $\sum_{b\in C}(-1)^{a\cdot b} = 0$ if $a \notin C^{\perp}$ and secondly, phase flip error in Z basis is just analogous to bit flip in Hadamard basis. Suppose C_1 is [n, k_1] code, C_2 is [n, k_2] code and $C_2 \subset C_1$. C_1 and $C_2^{\perp}$ both correct 't' errors. Now assume $x \in C_1$ and $a' \in C_1$ such that $a - a' \in C_2$. So, $|a + C_2\rangle = |a' + C_2\rangle$. Quantum code CSS ($C_1$, C_2)

is defined to be a vector space spanned by the states $|a + C_2\rangle \, \forall a \in C_1$. The number of cosets of C_2 in C_1 is $|C_1|/|C_2|$, whose dimension is $2^{k_1 - k_2}$. So CSS is a [n, $k1 - k2$] quantum code.

We define the quantum state by

$$|a + C_2\rangle \equiv \frac{1}{\sqrt{|C_2|}} \sum_{b \in C_2} |a + b\rangle. \tag{10.15}$$

After e_1 it flip error and e_2 phase flip error the quantum state becomes

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(a+b) \cdot e_2} \left|a + b + e_1\right\rangle. \tag{10.16}$$

Now, to detect the bit flip error e_1 we use an ancillary state $|0\rangle$ and apply parity check matrix H_1

$$\begin{aligned}
&\left|a + b + e_1\right\rangle |0\rangle \qquad (10.17)\\
&= \left|a + b + e_1\right\rangle \left|H_1 \left(a + b + e_1\right)\right\rangle \\
&= \left|a + b + e\right\rangle \left|H_1 e_1\right\rangle [\because x \in C_1, y \in C_2, H_1 a = H_1 b = 0].
\end{aligned}$$

Where $|H_1 e_1\rangle$ is the syndrome qubit, And the final state will be

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(a+b) \cdot e_2} \left|a + b + e_1\right\rangle \left|H_1 e_1\right\rangle. \tag{10.18}$$

Now, measuring the syndrome qubit, from the syndrome vector we can know the error and then using NOT gate we correct the error.

$$\frac{1}{\sqrt{|C_2|}} \sum_{y \in C_2} (-1)^{(a+b) \cdot e_2} \left|a + b + e_1\right\rangle. \tag{10.19}$$

To detect phase flip error we recall the second fact and apply Hadamard gate to each qubit

$$H^{\otimes}\Big(\frac{1}{\sqrt{|C_2|}}\sum_{y\in C_2}(-1)^{(a+b)\cdot e_2}\left|a+b+e_1\right\rangle\Big) \tag{10.20}$$

$$= \frac{1}{\sqrt{|C_2|\,2^n}}\sum_{z}\sum_{y\in C_2}(-1)^{(a+b)\cdot(e_2+z)}|z\rangle. \tag{10.21}$$

where, $z = z + e_2$

$$\frac{1}{\sqrt{|C_2|\,2^n}}\sum_{z'}\sum_{b\in C_2}(-1)^{(a+b)\cdot z'}\left|z'+e_2\right\rangle. \tag{10.22}$$

e_2 comes inside the ket from the phase factor, so the phase flip error now becomes a bit flip error. From the first fact we know $\sum_{z'\in C_2}(-1)^{y\cdot z'} = |C_2| = 2^{k_2}$ if $z' \in C_2^{\perp}$ and $\sum_{z'\in C_2}(-1)^{y\cdot z'} = 0$ if $x \notin C^{\perp}$.

$$\sqrt{\frac{|C_2|}{2^n}}\sum_{z'\in C_2^{\perp}}(-1)^{a\cdot z'}\left|z'+e_2\right\rangle. \tag{10.23}$$

Now doing the same as detecting the bit flip error (i.e. using an ancillary qubit and applying parity check matrix H_2)

$$\frac{|C_2|}{\sqrt{2^n/}}\sum_{z'\in C_2^{\perp}}(-1)^{a\cdot z'}\left|z'+e_2\right\rangle\left|H_2e_2\right\rangle. \tag{10.24}$$

Applying NOT gate we get

$$\frac{|C_2|}{\sqrt{2^n/}}\sum_{z'\in C_2^{\perp}}(-1)^{a\cdot z'}\left|z'\right\rangle. \tag{10.25}$$

For correcting the error we apply the Hadamard gate to each gate again

$$\frac{1}{\sqrt{|C_2|}} \sum_{b \in C_2} |a + b\rangle. \tag{10.26}$$

So, using CSS code we can correct the phase flip and bit-flip error.

11 Spin lattice models

After some basics of Quantum Error-Correcting Codes, now the approaches to studying these errors are needed. The tensor Network-based approach is seen to be more efficient to study the real error models in quantum circuits. Before starting with the Tensor Networks, let's first visualize the model on which a tensor can be defined.

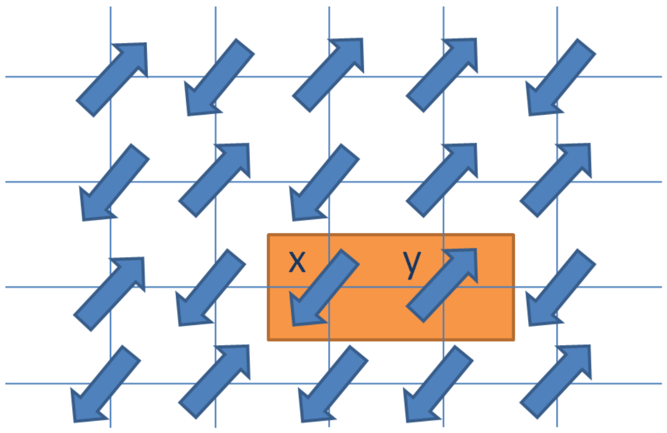


Figure 23: Spin lattice model on 2D grid

Let's consider a lattice with spins on its vertices. The state of those N-dimensional particles is described on the Hilbert space $(\mathbb{C}^d)^{\otimes N}$. For a two dimensional lattice where the Hamiltonians are local, the energies of the particles can be written as

$$E(S_1, S_2, ...) = \sum_{<xy>} S_x.S_y. \tag{11.1}$$

where x and y are the labels of the sites where the particles reside.$S_x.S_y$ is the energy of a particle on 'xy' site and E is the total energy of all the particles. Now we can define each vertex as a Tensor T with four legs S_a, S_b, S_c, S_d. And the interpretation of the tensor is that it computes the local energies associated with the legs living on the vertices. The same model can be made in case of a 1D chain where each vertex will have two legs instead of four.

12 Tensor Networks

There are a few concepts and terms which we have to understand before we dig into the concept of TNs and DMRG calculation for the ground state of a many-body quantum system.

12.1 Single Value Decomposition (SVD)

This is a matrix factorising method. Let, M be any complex $m \times n$ matrix with $n \geq m$ (If $n < m$, then SVD will be done on $M^{\dagger}$).

Then there exists $u(m \times m), D(m \times n)$ and $v(m \times n)$ such that $M = uDv$, where D is a diagonal matrix with $D_{ii} \geq 0, u$ is a unitary matrix and v is orthonormal.

12.2 Smidth Decomposition

The Schmidt decomposition is a way of expressing a vector as the tensor product of two inner product spaces. If there is a two-qubit pure state $|\psi\rangle$ such that $|\psi\rangle \in H_A \otimes H_B$, then $\exists\ |a_i\rangle \in H_A$ and $|b_i\rangle \in H_B$ such that

$$|\psi\rangle_{AB} = \sum_{i=1}^{N} \lambda_i \, |a_i\rangle \otimes |b_i\rangle \, , \tag{12.1}$$

where

$$\sum_{i=1}^{N} \lambda_i^2 = 1. \tag{12.2}$$

λ is called the Schmidth coefficient and $|a_i\rangle$, $|b_i\rangle$ are the Schmidth bases. Schmidth Decomposition can be used to calculate the entaglement which is very crucial in DMRG calculation for many-body system.

A density matrix of the state ψ, thus, can be written as

$$\begin{aligned} \rho = |\psi\rangle_{AB} \langle\psi|_{AB} &= \Big(\sum_{i=1}^{N} \lambda_i \, |a_i\rangle \otimes |b_i\rangle \Big) \Big(\sum_{j=1}^{N} \lambda_j \, \langle a_j| \otimes \langle b_j| \Big) \\ &= \sum_{i,j} \lambda_i \lambda_j \Big(|a_i\rangle \otimes |b_i\rangle \Big) \Big(\langle a_j| \otimes \langle b_j| \Big). \end{aligned} \tag{12.3}$$

And the Partial Trace is the process to extract the information of a subsystem from a multipartite quantum state and for a bipartite system it can be written as $\rho^A = Tr_B(\rho)$ and $\rho^B = Tr_A(\rho)$.

$$\begin{aligned}
\rho^A &= Tr_B(\rho) \\
&= \sum_{i,j} \lambda_i\lambda_j \Big(|a_i\rangle \langle a_j| \Big) Tr \Big(|b_j\rangle \langle b_j| \Big) \\
&= \sum_{i,j} \lambda_i\lambda_j \Big(|a_i\rangle \langle a_j| \Big) \Big(\langle b_j| \, |b_j\rangle \Big) \\
&= \sum_{i} \lambda_i^2 \, |a_i\rangle \langle a_i| \, .
\end{aligned} \tag{12.4}$$

So, Partial trace of subsystem A of the state $|\psi\rangle_{AB}$ is defined as $\rho^A = \sum_i \lambda_i^2 |a_i\rangle \langle a_i|$ and similarly Partial Trace of subsystem B can be written as $\rho^B = \sum_i \lambda_i^2 |b_i\rangle \langle b_i|$.

If, $\lambda_i \neq 0$, then it'll be a Separable state if $\lambda_i = 1$ and an Entangled state if $\lambda_i > 1$. So, Schmidt Decomposition can be useful to know the entanglement of a state.

12.3 Tensor Network Notations

Tensor Networks are a very useful technique to work with tensors with many indices. It is a visualised representation of multi-linear mapping between two vector spaces. These tensor networks contract high-order tensors into a low-order tensor and thus making the computations like summation, inner and outer products of two N-indices tensors.

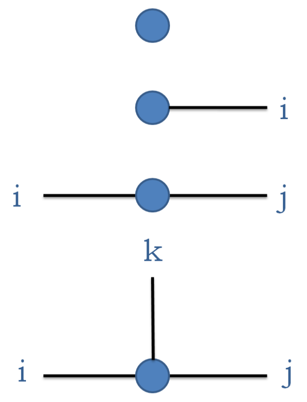


Figure 24: Basic notations for Tensor Networks

Here, the first figure having no leg (or index) is the visual representation of a Scalar quantity, the second figure has one index and is the representation of a Vector (v_i), the third figure has two indices and is known as Matrix (M_{ij}) and the fourth figure has three indices and is known as Tensor (T_{ijk}).

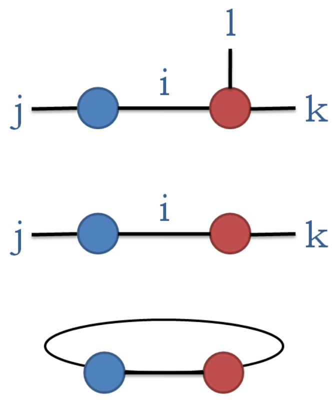


Figure 25: Some other notations for Tensor Networks

Here the first figure is representing a tensor contraction of two tensors with two indices (Vector) and three indices (Matrix) respectively (The operation can be realized as $M_{ij}N_{ikl} = T_{jkl}$, 'i' index is contracted here). In the next figure matrix multiplication is depicted ($u_{ij}v_{ik} = U_{jk}$) and the last figure is the representation of trace of a matrix i.e. if two matrices are assumed to be A and B then this represents $Tr(AB)$.

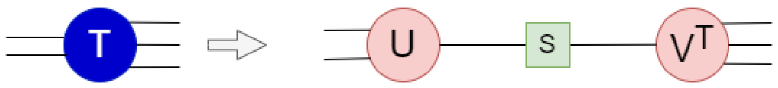


Figure 26: Singular Value Decomposition

12.4 Tensor Networks in Many-Body Quantum System

In Mean Field Theory, we neglect the concept of Entanglement. That's the old way of Condensed Matter Physics being able to describe most of the systems as they are in high temperature where entanglement becomes less important. But we are very much interested in lower temperature because we are to calculate the the ground state energy of a quantum system. And at lower temperature entanglement plays a very crucial role.

$$\underbrace{\bullet\ \bullet\ \bullet\ \bullet\ \bullet\ \bullet \ldots \bullet\ \bullet\ \bullet}_{(\mathbb{C}^d)^{\otimes N}}$$

The basis state of a system can be written as $|i_1\rangle$, where $i_1 = 0, 1, 2, ..., d-1$. The basis set of the total system is $|i_1, i_2, i_3, ..., i_N\rangle$ and $i_k = 0, 1, 2, ..., d-1$. In general, one can write the state as

$$|\psi\rangle = \sum C_{i_1,i_2,...,i_N} |i_1, i_2, ..., i_N\rangle \, . \tag{12.5}$$

The interaction can be between any two spins but we are more interested in local and strong interactions rather than long and weak interactions. This allows us to study the system from the point of view of entanglement.

Now, we will talk about Quantum matter. But the behavior of quantum matter is not the same as that of conventional matter, as discussed, from the angle of entanglement and interaction. For example, the Hamiltonian of an Ising model in a magnetic field is

$$H = -\sum_i \sigma_i^z \sigma_{i+1}^z + h \sum_i \sigma_i^x, \tag{12.6}$$

where the first term indicates the Ising interaction (All spins up or all spins down) h is the strength of the magnetic field and second term aligns them along x axis. When $h << 1$, it breaks symmetry and it respects symmetry when $h >> 1$. For this, we measure local order parameter $\frac{1}{N}\sum_i \sigma_i^z$ to know which phase we are in.

In case of Quantum matter, it doesn't follow the framework of the conventional matter as it includes entanglement. We are interested in the ground state as from this we can get the information of the excited states from the ground state as at low-temperature Ground State is most quantum in nature and gapped. The most significant feature of entanglement is that the entropy depends on the dimension of the boundary of the system rather than the volume. This feature enables us to perform DMRG calculations to calculate the energy of the ground state. We will discuss this feature called 'Area law' and DMRG calculation in detail.

Talking about the symmetry, due to the **Monogamy of Entanglement** we can't have the same entanglement between any two sites, in the picture above, the entanglements between sites 1, 2 and sites 3, 4 are not the same. So, it does not respect the translation symmetry. But we want every cut to behave in same manner. To avoid the asymmetry, let's represent it in such a way that one particle is divided into two sub-particles as below.

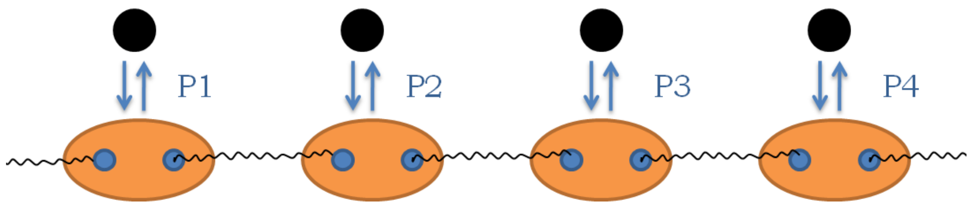


Figure 27: Mapping of two particles to one particle to keep the translation symmetry

If we consider the black particles with dimension d in the place of two blue particles with dimension D such that $PS : \mathbb{C}^D \otimes \mathbb{C}^D \approx \mathbb{C}^d; (S = 0, 1, 2, 3, ...)$, where P are the linear map that ensures the effective particle (The black particles) is our actual particle, then if we, now, cut anywhere the symmetry remains same. If the sites are the same then for each site the linear map P is the same, otherwise, it'll be site-dependent.

To maintain translation-invariant property, we can follow two conditions -

1. **Open Boundary Condition:**

   Here, the boundaries are open, where the map P acts as $P : \mathbb{C}^D \approx \mathbb{C}^d$ and the maps inside act as mentioned above.

2. **Closed Boundary Condition:**

   Here the particles in the edges are entangled with each other making a closed chain-like structure.

$$|\psi\rangle = \left[P^{(1)}_{1a1b} \otimes P^{(2)}_{2a2b} \otimes P^{(1)}_{3a3b} .... \otimes P^{(n)}\right] |\omega\rangle^{\otimes N} , \tag{12.7}$$

where

$$|\omega\rangle = \frac{1}{\sqrt{D}} \sum_{i=1}^{D} |i, i\rangle . \tag{12.8}$$

Here, $P^{i}_{i_a i_b}$ is the map that takes two auxiliary particles i_a and i_b into ith particle and ω is called the 'Matrix Product State' which is a maximally entangled state. Now we can write the state in two following ways

$$|\psi\rangle = \left[P^{(1)} \otimes P^{(2)} \otimes P^{(1)} .... \otimes P^{(n)}\right] |\omega\rangle^{\otimes N} . \tag{12.9}$$

$$|\psi\rangle = \sum C_{i_1, i_2.., i_N} |i_1, i_2.., i_N\rangle . \tag{12.10}$$

And we have to see what is the interpretation of the coefficient $C_{i_1,i_2..,i_N}$.
Let's say, a projection operator $P^(S)$ which projects particles α, β to particle i can be written as below

$$P^{(S)} = \sum_{\alpha,\beta=1,..,D i=1,...,D} A^{(S),i}_{\alpha,\beta} |i\rangle \langle \alpha, \beta| . \tag{12.11}$$

Then for two sets of particles $\alpha, \beta, \gamma, \delta$ (β and γ particles are entangled) which are mapped to i_1 and i_2 pairwise can be written as follows

$$\begin{aligned} P^{(1)} \otimes P^{(2)} |\omega_{1,2}\rangle &= \Big(\sum_{\alpha,\beta,\gamma,\delta,i_1,i_2} A^{(1),i_1}_{\alpha,\beta} |i_1\rangle \langle \alpha, \beta| \otimes A^{(2),i_2}_{\gamma,\delta} |i_2\rangle \langle \gamma, \delta| \Big) \\ &\Big(\frac{1}{\sqrt{D}} \sum_{k=1}^{D} |k,k\rangle\Big) \\ &= \sum_{\alpha,\beta,\delta,i_1,i_2} A^{(1),i_1}_{\alpha,\beta} |i_1\rangle \langle \alpha| \otimes A^{(2),i_2}_{\beta,\delta} |i_2\rangle \langle \delta| \\ &= \sum_{\alpha,\delta,i_1,i_2} \sum_{\beta} A^{(1),i_1}_{\alpha,\beta} A^{(2),i_2}_{\beta,\delta} |i_1 i_2\rangle \langle \alpha\delta| \\ &= \sum_{\alpha,\delta,i_1,i_2} \Big(A^{(1),i_1} A^{(2),i_2}\Big)_{\alpha,\delta} |i_1 i_2\rangle \langle \alpha\delta| . \end{aligned} \tag{12.12}$$

Here, $\Big(A^{(1),i_1} A^{(2),i_2}\Big)$ is the matrix product of two operators. So, the coefficients can be interpreted as the ‘Matrix Product state’ (MPS) of the operators.

Similarly,

$$P^{(1)} \otimes P^{(2)} \otimes P^{(3)} |\omega_{1,2}\rangle = \sum_{\alpha,\delta,i_1,i_2} \left(A^{(1),i_1} A^{(2),i_2} A^{(3),i_3} \right)_{\alpha,\beta} |i_1 i_2 i_3\rangle \langle \alpha\beta| . \quad (12.13)$$

13 Density Matrix Renormalization Group (DMRG)

For many-body systems and weak interaction, we use DFT (Density Function Theory), DMFT (Dynamical Mean-Feild Theory), etc. On the other hand, for strongly interacting systems where entanglement takes an important role we use DMRG (Density Matrix Renormalization Group) calculation to find the ground state energy of a Hamiltonian.

13.1 Review of making the Hamiltonian matrix for few spin system

We know the Pauli matrices can be written as

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

In a two-spin system, the total spin can be written as $\vec{S}_{Total} = \vec{S}_1 + \vec{S}_2$, where S_1 is the spin angular momentum of the first particle and S_2 is the spin angular momentum of the second particle.

$$\begin{aligned} S^2_{Total} &= S_{Total}.S_{Total} \\ &= (\vec{S}_1 + \vec{S}_2).(\vec{S}_1 + \vec{S}_2) \\ &= S_1^2 + S_2^2 + 2\vec{S}_1.\vec{S}_2. \end{aligned} \tag{13.1}$$

With the properties of spin angular momentum we get the solution of the above equation as

$$\vec{S}_1.\vec{S}_2 = \frac{1}{2}\big[S(S+1) - \frac{3}{2}\big]. \tag{13.2}$$

Now, after some lines of algebra, we can write Eq. (13.2) in terms of the adder (S^+) and ladder (S^-) operators as

$$\vec{S}_1.\vec{S}_2 = \vec{S}_{1z}.\vec{S}_{2z} + \frac{1}{2}\big(S_1^+ S_2^- + S_1^- S_2^+\big). \tag{13.3}$$

From Heisenberg model $H = J\sum_{i\neq j} \vec{S}_i.\vec{S}_j$, we can write the Hamiltonian of the two-spin system as

$$H = J \begin{pmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & -\frac{1}{4} & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & -\frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{4} \end{pmatrix},$$

where J is called 'Exchange constant'.

Likewise, for a three-spin system, the hamiltonian will be an 8×8 matrix

$$H = J. \begin{pmatrix} 2*1/4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & -2*1/4 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & -2*1/4 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2*1/4 \end{pmatrix}.$$

So as we can see for n-qubit system, the hamiltonian matrix is of $n \times n$ dimension. If n is very large, then it becomes very hard to diagonalize and calculate the ground state energy. Instead, we take a block for which $[H, S^z_{Total}] = 0$. This is a huge help, making the problem of the order of $\mathcal{O}(2^N/N)$ from $\mathcal{O}(2^N)$. For example, in this case for $N = 3$, we can take the block,

$$H_{block} = J \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \\ 0 & \frac{1}{2} & 0 \end{pmatrix}.$$

We can now diagonalize it and minimum eigenvalue will be the the Ground state energy. Here, $E_{gr} = -J$ and corresponding eigenstate is

$$|\psi\rangle = \tfrac{1}{\sqrt{6}}\begin{pmatrix}1\\-2\\1\end{pmatrix}.$$

13.2 Area Law

Area law is something based on which DMRG calculation is possible. The main goal of the calculations in many-body physics is to optimally reduce unnecessary degrees of freedom so that the order of the problem gets reduced. For that, we don't need to look at the whole Hilbert space of the system at once. Instead, we work with the entanglements corresponding to the ground state.

Let's take a one-dimensional spin chain which is in $|\psi\rangle$ state. We have to make the Schmidth Decomposition of the system wavefunction by dividing it into two subsystems $|L\rangle$ and $|R\rangle$ i.e. $|\psi\rangle = \sum \lambda_i |L\rangle |R\rangle$.

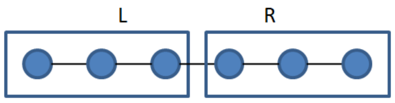
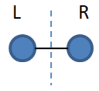


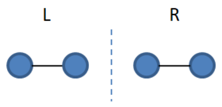
Figure 28: Schmidth decomposition of 1D spin-chain into left-right subsystems

With the number of particles, the entropy of the system should change, but in reality, it remains almost the same. Rather we see an alternation of the entropy value. The maximum entropy of a chain grows like $S_{max} = \frac{N}{2} ln2$. Except for the case N=2, where it has the maximum entropy i.e. 0.69. Why? The entanglement resides between a particle in left site $|L\rangle$ and one in right site $|R\rangle$. If the cut is through two entangled particles, then the entanglement of the chain becomes higher otherwise it remains low.

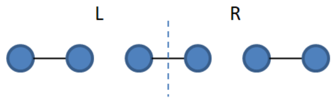
So, for N=2, the cut is through the pair, for the case N=4 also, the cut is through



(a) Higher Entanglement when N=2



(b) Lower Entanglement when N=4



(c) Higher Entanglement when N=6

Figure 29: Variation of entanglement depending on the number of particles

the pair, so for these two cases the entanglement entropy is high but as the cut is only one, not through each entangled pair, so for N=2 only, the entanglement entropy is maximum but that is not the case for N=4 or any other cases. On the other hand, for the case N=3, the cut is not through an entangled pair, the entropy is much lower. So, there's an alternation of high and low values. So as we can see here, the entropy is actually depending on the cut or in the broader sense of the boundary. As it turns out, this very special property of entanglement entropy comes from black hole physics. In black hole physics, the entropy is proportional to the surface of the black hole, not to the volume. In the context of many-body physics, the entropy is proportional to the boundary of the system, this is called the **Area Law**. For a 1D system, it's actually constant as the boundaries of the system are the points, for a 2D system the area is equivalent to its length, and for a 3D system, it's the surface area.

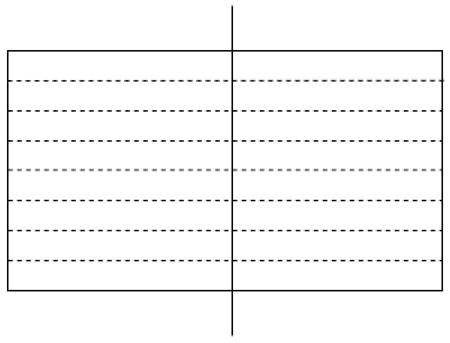


Figure 30: The cut through the entanglement between two subsystems

There is no entanglement between two sites unless a singlet bond is cut. The more the number of cuts, the higher the entropy. Area law makes DMRG work. it allows truncating states with lower probability

$$\sum_{i=1}^{2^{N/2}} \lambda_i^2 \left|i\right\rangle_L \left|i\right\rangle_R \rightarrow \sum_{i=1}^{m} \lambda_i^2 \left|i\right\rangle_L \left|i\right\rangle_R . \tag{13.4}$$

This is an exponential reduction. This is the same as we did for the Hamiltonian matrix in the previous section.

13.3 Procedure of DMRG

In the initial step, each site contains only one particle. This is solved by diagonalizing the Hamiltonian. Then one particle is added to each site in each iteration. So, each site grows like 2, 3, etc. and at each step the Hamiltonian is diagonalized. As there's exponential growth for each particle in a site, we truncate the block at D-dimension if the block is too big.

The set of left block+two particles+right block is called the **superblock**, which is in the Hilbert subspace of the whole system. The ground state energy candidate is found by calculating the reduced density matrix for one sub-block. At first, the approximated energy state is far away from the actual value. But with the number of sweeps, the values get improved and the site is updated. Once one block reaches the maximum size, the procedure goes with the second block. And after sweeps of calculations we get the approximated ground value depending on the error value we set.

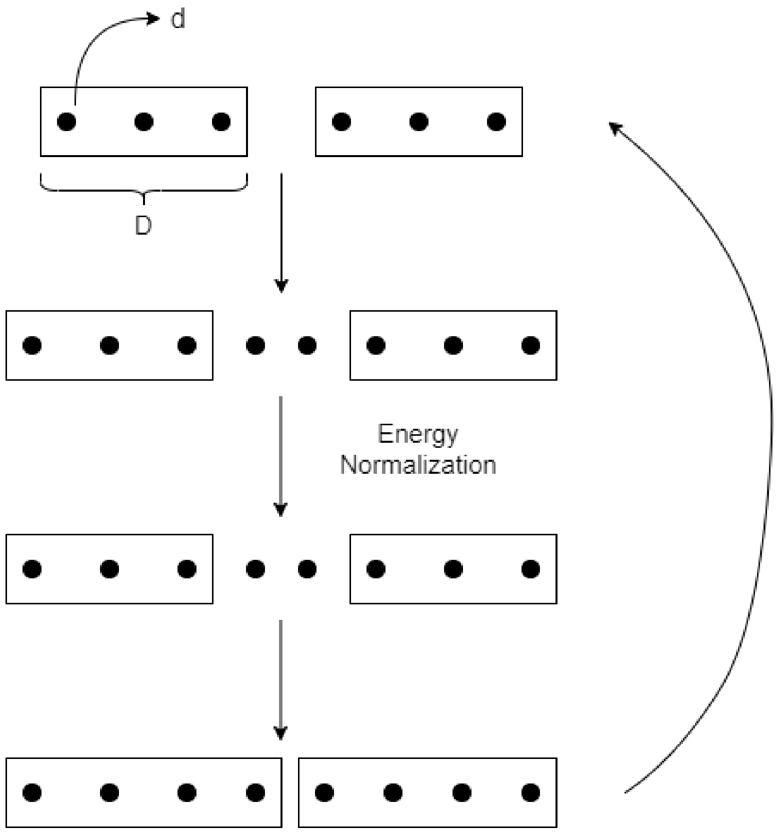


Figure 31: The energy trend of the Rydberg atoms lattice under dipole-dipole interaction

The above diagram represents the iDMRG algorithm (Infinite DMRG). Up to a nominal number of sites, we don't need to truncate any state as we can comfortably do the calculation when the system size is small. Instead looking at the whole Hilbert Space all together, the algorithm starts with very small sub-space with 4-6 sites and then divide it into two subsystems by performing Schmidth decomposition. Then we slowly grow the system by adding one site each to the subsystems and normalize the energy for each step and repeat this step until it gets bigger and then we perform DMRG algorithm.

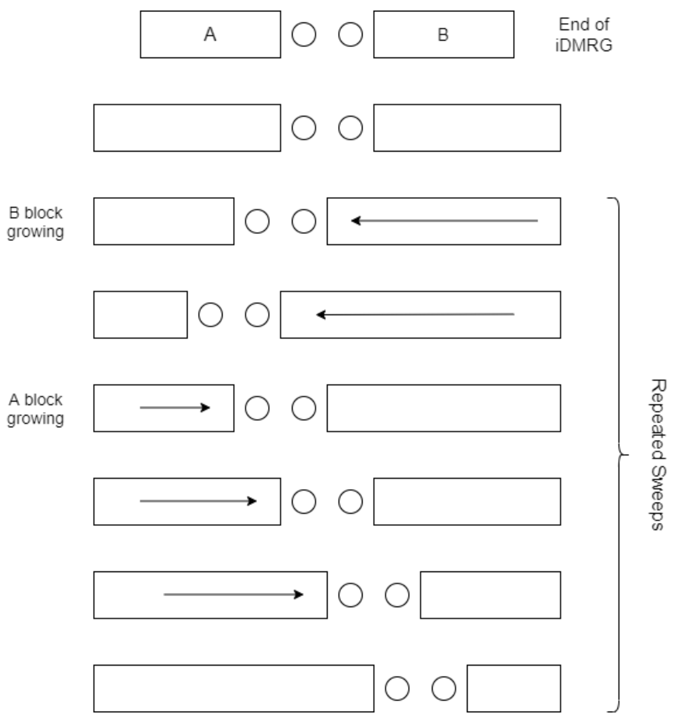


Figure 32: DMRG algorithm diagram

After iDMRG, then we start to truncate the states with lower probability or lower Schmidth eigenvalues. To do that we start right from the middle and then run SVD (Single Value Decomposition) which is equivalent to diagonalize the Hamiltonian for the considered system. We go left making the growth of right subsystem and with each shift we perform the SVD or the diagonalisation of the Hamiltonian and calculate the reduced density matrix of the subsystem. Same procedure will be followed for the left subsystem. The whole step completes one sweep and take out the minimum energy from this. With every sweep the minimum energy gets improved and after some sweeps it provides a well approximated energy value of the system.

14 Rydberg Atoms

The study of Rydberg atoms took a crucial role in developing quantum mechanics. To understand the atom-light interaction Rydberg atoms were uses as an ideal test-bed because of their strong interaction with electromagnetic fields and this inspired the birth of cavity quantum electrodynamics. In the stage of 1970s, the interactions among these atoms did not take any important rule. At the end of 1990, this becomes of huge interest when the progress of laser cooling of atoms allowed for the realisation of frozen Rydberg atoms. The strong interactions between these atoms can be used to make quantum gates. Rydberg atom is a very good candidate for the future quantum computer to establish a long-range entanglement due to strong interactions with highly coherent operations and flexible geometry and higher lifetime.

An atom can be regarded as a Rydberg atom if the outermost electron is in a very high principal quantum number. Using the Hydrogen model we can study the Rydberg state of any atom up to a good approximation as the principal quantum number is pretty high ($n \geq 10$).

The expectation value of radius of an electron is given by

$$< r >= n^2 a_0 \Big[1 + \frac{1}{2}\Big(1 - \frac{l(l+1)}{n^2}\Big)\Big], \tag{14.1}$$

where 'n' is the principal quantum number, 'a_0' is the Bohr radius and 'l' is the azimuthal quantum number.

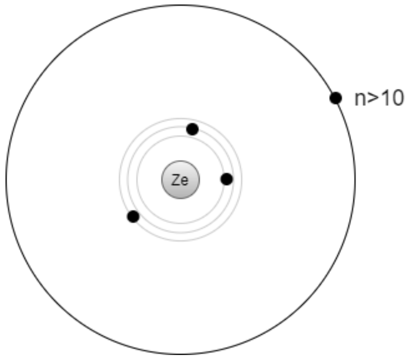


Figure 33: Rydberg state of an atom where the last principle quantum number is very high

Now we know the wavefunction of hydrogen atom i.e.

$$\Psi_{nlm} = \frac{1}{r}\rho^{l+1} \exp^{-\rho} v(\rho) Y_{lm}(\theta, \phi). \tag{14.2}$$

$v(\rho)$ =Polynomial of degree $n - l - 1$, $\rho = r/a_0 n$

The probability of finding the electron in the range r to $r+dr$ is $P(r) = |R_{nl}|^2 r^2$, where $R_{nl} = \frac{1}{r}\rho^{l+1} \exp^{-\rho} v(\rho)$

After some lines of substitutions we get

$$\begin{aligned} R_{nl} &\sim r^l(a_0 + a_1 r + ... + a_n r^{n-l-1}) \\ &= r^{n-1} e^{r/na_0}. \end{aligned} \tag{14.3}$$

As the radius is assumed to be large we ignore all terms of the polynomial ($v(\rho) \sim a_0 + a_1 r + ... + a_n r^{n-l-1}$) except the highest degree term.

In astrophysical objects in the recombination process, protons are seen to be capturing electrons from very high quantum numbers e.g. $n = 350$. For that, the radius becomes $r = 0.53 \times 10^{-10} \times 350^2 = 6.5\mu m$, which is not quantum anymore, rather it is in semi-classical regime. In normal atoms e.g. from $n_f = 5$ to $n_i = 1$, it takes $1/10^7$ seconds whereas Rydberg atoms take much more time to decay, sometimes 1 second, as there is an enormous number of states in between.

15 Application of DMRG calculation to Rydberg atoms interaction

In this section, we use the DMRG method to calculate the Ground State Energy of Rydberg atoms under strong dipole-dipole interactions for the XY spin model. The atoms excited to Rydberg atoms possess a large electric dipole moment which leads to strong dipole-dipole interactions between the atoms.

XY spin model is the limiting case of the Heisenberg spin model. In this model, the spins are less coupled in z-direction than in x and y -directions. In the other case, it's called the Ising model. Here, we consider the case where the atoms are in two different Rydberg states and are dipole coupled. In the spin-$\frac{1}{2}$ system, states $|\uparrow\rangle$ and $|\downarrow\rangle$ are separated by transition frequency typically in GHz range. The interaction potential scales like $\frac{C_3}{R^3}$. And the total Hamiltonian looks like [8]

$$H = \frac{\hbar\Omega_{xy}}{2}\sum_i \sigma_x^i - \frac{\hbar\delta_{xy}}{2}\sum_i \sigma_z^i + \sum_{i\neq j}\frac{C_3}{R_{ij}^3}\left(\sigma_+^i\sigma_-^j + \sigma_-^i\sigma_+^j\right), \tag{15.1}$$

where Ω_{xy} is the Rabi frequency and δ_{xy} is the detuning of microwave field and σs are the Pauli Matrices.

15.1 Application Process

Using the ITensor package in *Julia*, I have calculated the Ground State energy of this Hamiltonian in the 2D lattice of Rydberg atoms under this strong interaction. ITensor, short form of **intelligent tensor** package is a software library that is made to enable users to write code for the tensor notations without thinking much about the ordering of the indices. It also enables users to implement high-level algorithms like DMRG (The one I use here). In this process, I calculate the ground state energy for the 2-dimensional system consisting of 1 to 10 Rydberg atoms in each dimension (X and Y). I consider the system taking each and every combination of numbers of particles from 1 to 10 i.e. from 1×1 lattice to 10×10 lattice.

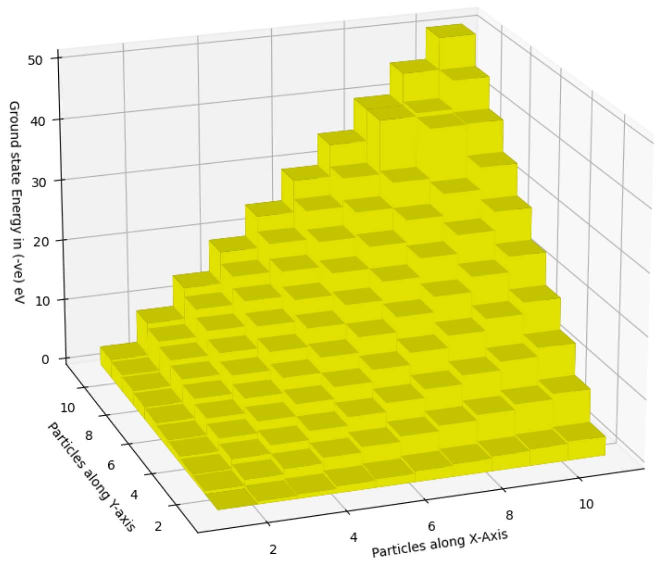


Figure 34: The energy trend of the rydberg atoms lattice under dipole-dipole interaction

The energy values are negative. As one can easily see the trend of the ground energy levels of the system. The more Rydberg particles in the system the lesser value of energy level and it's almost symmetric which is expected. The details of the code (and the explanation) are given in appendix A below.

16 Tensor Network Contraction Application to Error Correction

16.1 Hypergraphs

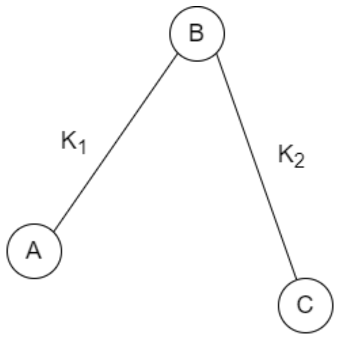


Figure 35: Graph

Here in this picture, A, B, C are called *Vertices* and K_1, K_2 are called *Edges.* This is a Normal Graph, where the edges are of size 2. This is denoted as $G(V, E)$, where $V = [A, B, C]$ are the vertices and $E = [K_1, K_2] = [(A, B), (B, C)]$ are the edges. Hypergraph is something where the size of the edges is equal or more than 2. It is denoted as $H(V, E)$

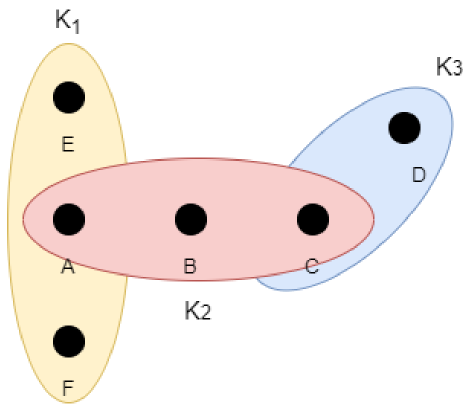


Figure 36: Hypergraph

Here, $V = [A, B, C, D, E, F]$, $E = [K_1, K_2, K_3] = [(A, B, C), (C, D), (E, A, F)]$. Tensor networks can be represented as hypergraphs where the vertex represents a tensor and each hyperedge is tensor-index. If not mentioned, we take it as 2. When the edges are closed, it's a scalar.

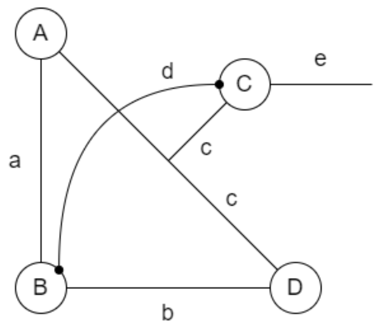


Figure 37: A representation of Tensor contraction

In the above figure, a tensor contraction is represented. Tensor $T_b = \sum_{a,b,c,d} A_{ac}B_{abd}C_{cde}D_{bc}$ is contracted by hyperedges a, b, c, d. Only one open hyperedge is e. So, $V = [A, B, C, D]$ and $E = [a, b, c, d, e] = [(A, B), (B, D), (A, C, D), (B, C), (C)]$ and this is a hypergraph $H = [V, E]$. In the context of quantum computation, it is a common occurrence that the dimension of each edge is 2 representing the two states of a qubit.

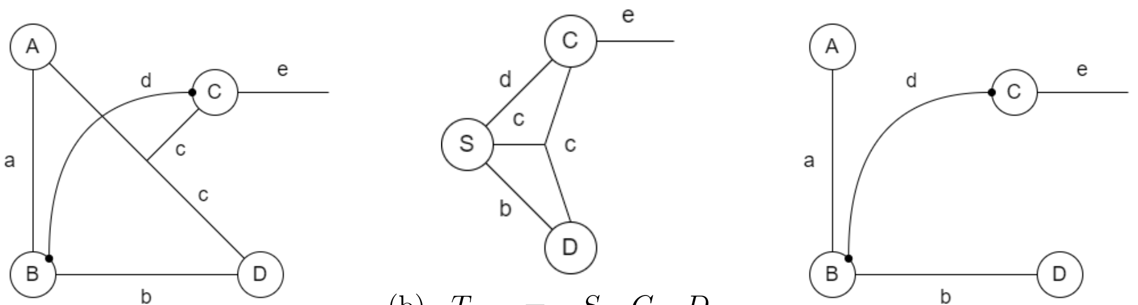
16.2 Parallel Tensor Network Contraction Algorithm

Tensor Network contraction is a common way to evaluate a tensor network which works by repeatedly eliminating closed edges and combining adjacent nodes until only a single node with open edges is left.

A tensor network $G' = (V', E')$ is a sub-network of another network $G(V, E)$ iff $V' \subseteq V$, $E' = [e \cap V' | e \in E]$ i.e. G' contains all edges of G associated with atleast one vertex in V' and all closed edges of G' are closed edges of G that falls inside V'.

In a Tensor Network G, any sub-network H with n vertices can be contracted to create a new Tensor Network G' with $(n-1)$ fewer vertices than G.

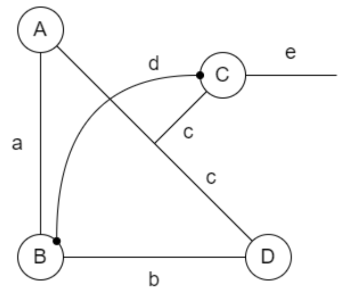
Now, we know if there are more than one same set of indices then we can contract those indices and in a different way to get the same result. Let's take the above example where the same set of indices are a, b, c, d, and the tensor can be contracted in different ways. The work is to find the optimized order of Tensor contraction for which the cost is minimum. [9]



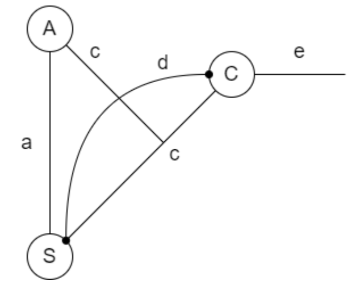
(a) $T_{be} = A_{ac}B_{abd}C_{cde}D_{bc}$ (b) $T_{be} = S_{bcd}C_{cde}D_{bc}$ (Contracting A and B) (c) $T_{be} = A_aB_{abd}C_{de}D_b$

Figure 38: Ordering of contraction with respect to index a

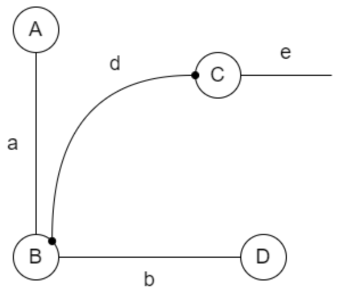
Here, A_{ac} and B_{abd} are contracted with respect to the index a. So, $S_{bcd} = A_{ac}B_{abd}$.



(a) $T_{be} = A_{ac}B_{abd}C_{cde}D_{bc}$



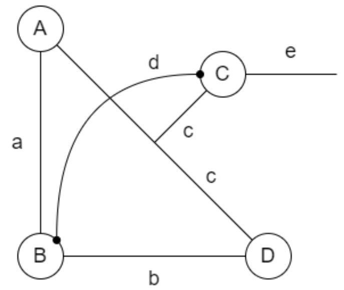
(b) $T_{be} = A_{ac}S_{acd}C_{cde}$ (Contracting B and D)



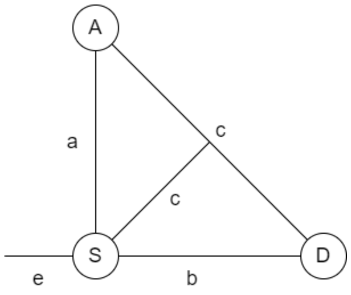
(c) $T_{be} = A_aB_{abd}C_{de}D_b$

Figure 39: Ordering of contraction with respect to index b

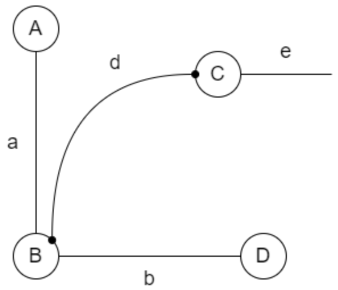
Here, B_{abd} and D_{bc} are contracted with respect to the index b. So, $S_{acd} = B_{abd}D_{bc}$.



(a) $T_{be} = A_{ac}B_{abd}C_{cde}D_{bc}$



(b) $T_{be} = A_{ac}S_{abce}D_{bc}$ (Contracting B and C)



(c) $T_{be} = A_{ac}B_{abd}C_{cde}D_{be}$

Figure 40: Ordering of contraction with respect to index d

Here, B_{abd} and C_{cde} are contracted with respect to the index d. So, $S_{abce} = B_{abd}C_{cde}$.

16.3 Application to Quantum Error Correction

In this section, we will study how Tensor Network is useful for studying Quantum Error Correction codes based on this paper. Quantum Error Correction is a very useful method for a scalable fault-tolerant quantum computer. Though it needs so many qubits to actually make one logical qubit and that is still some steps away in the future. But still, it's important to be studied theoretically and experimentally. This is the only way by which we can achieve the *Quantum Supremacy.* Now, existing error models provide only a rough approximation of the logical error rates. On the other hand, realistic error models would have a better approximation of the errors.

The study is done focusing the crosstalk error model on surface code, specifically the Surface-17 code.

16.3.1 Introduction

- **Fault Tolerant Quantum Computer**

  Even though the environment plays a great role to have errors on qubits, and that is one of the bars to achieve what we call the Quantum Supremacy, still we can think of a quantum computer that can perform pretty well. Arbitrary good computation can be done even with faulty logic gates given that the error probability is below a certain constant threshold.

  Performing error correction periodically is not sufficient to prevent the build-up of errors, because -

  1. The encoded gates can also cause error to propagate.

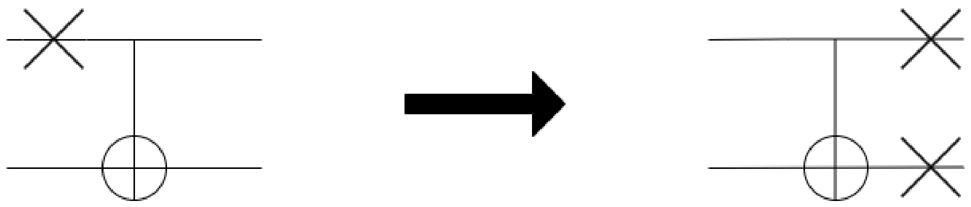


Figure 41: Propagation of error via a perfect CNOT gate

As you can see above though the gate is error-free, but the operation makes the error propagate. Let, $CNOT \sim U$ and $Error \sim X$, then the above operation will be like $UX_1 = UX_1U^\dagger U = X_1X_2U$ where the error propagates from X_1 to X_1X_2. So, encoded gates should be designed very carefully so that less amount of error is propagated. Suppose, the CNOT gate is itself noisy. Let's take operation implemented by noisy CNOT is E and the operation implemented by perfect CNOT is U. E can be then written as $E = EU^{-1}U$. So, the operation of noisy CNOT can be approximated as EU^{-1} on perfect CNOT.

2. Error Correction itself can produce errors on the encoded qubit.

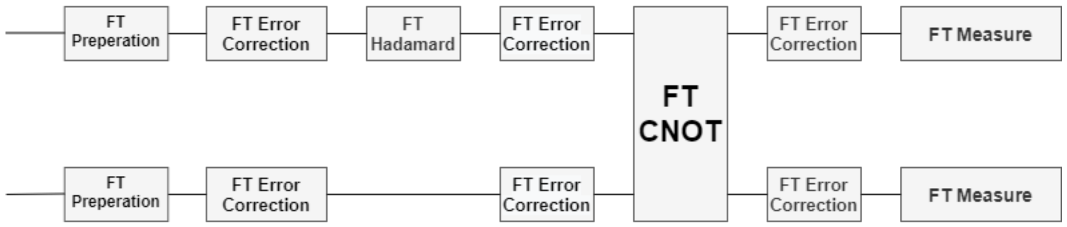


Figure 42: Block diagram of a Fault-Tolerant process

- **Concatenated Code and Threshold Theorem**

Suppose we have two levels of concatenation. If the failure probability of the components at a lower level is 'P'. After one level of encoding, it will be $P \rightarrow cP^2$. After two levels of encoding it will be $cP^2 \rightarrow c(cP^2)^2$. Now, if we concatenate it K times, the maximum failure probability is $(cP)^{2^K}/c$. Whereas the size of simulating circuit goes like d^K times the size of the actual circuit. But that's okay as $\mathcal{O}(d^K) << \mathcal{O}((cP)^{2^K})$. Now, if we want to simulate a circuit containing $P(n)$ gates, ($P(n)$ is polynomial of n) we want to achieve an accuracy of ϵ in our simulator. Each gate in the simulator must be accurate to $\epsilon/P(n)$. So, the failure probability is less than the accuracy of each gate. If we concatenate that K times then $\frac{(cP)^{2^K}}{c} \leq \frac{\epsilon}{P(n)}$, provided $P < P_{th} = 1/c$ [After one level of encoding $P \rightarrow cP^2$, if $P < 1/c$ then $cP^2 < P$ and that's an achievement]. This is known as **Threshold Condition** for Quantum

Computer. A quantum circuit containing $P(n)$ gates may be simulated with probability of error at most ϵ using $\mathcal{O}(Poly(logP(n)/\epsilon)P(n))$ gates on hardware whose components fail with probability at most P, given $P < P_{th}$.

Parallel operations are also very important. Without this threshold after some time errors accumulate in the circuit too quickly which we can't correct. So, a continuous fresh source of qubits is also needed.

- **Fault Tolerant Quantum Logic Gates**

  For Stean code $\bar{Z} = Z_1Z_2Z_3Z_4Z_5Z_6Z_7$ and $\bar{X} = X_1X_2X_3X_4X_5X_6X_7$. Now as $HZH^\dagger = X, HXH^\dagger = Z$, the encoded $\bar{H} = H_1H_2H_3H_4H_5H_6H_7$ must be conjugate with $\bar{Z}$ and $\bar{X}$ too. So, $\bar{H}\bar{Z}\bar{H}^\dagger = \bar{X}, \bar{H}\bar{X}\bar{H}^\dagger = \bar{Z}$.

  If Z error occurs on any one qubit i.e. $HZ = HZH^\dagger H = XH$, it's like applying Hadamard gate first and then the occurrence of X error. So, the error doesn't propagate. Error on one qubit remains on one qubit and that we can correct. So, Hadamard gate operation is automatically fault-tolerant. This bitwise operation is called the 'Transversality' property of an encoded quantum gate. Any bitwise operation by X and Z gate is also fault-tolerant automatically. For a phase gate S,

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}.$$

  Here, $\bar{S}\bar{Z}\bar{S}^\dagger = \bar{Z}, \bar{S}\bar{X}\bar{S}^\dagger = -\bar{Y}$ [The minus sign can be compensated by Z gate], where $\bar{S} = S_1S_2S_3S_4S_5S_6S_7$. So, for S-gate ZS operation to each qubit in the code effects an encoded phase gate which is transversal and thus fault-tolerant. Hence, we see H, X, Z, Y, S gates can be made fault tolerant. One more gate we require to complete the standard set of gates for universal quantum computation is $\pi/8$ gate.

Analog computers can't correct errors. Errors accumulate in continuum space and Quantum computer has the same feature that made people being skeptical about

Quantum Error Correction. But errors can be digitalised and there comes the power of Quantum Computer. To use concatenation code, we need a huge number of physical qubits to make one logical qubit but they stay across large distances from each other, so the operations are not local. That's where surface code comes in, to make the operations local, the 'Surface Code' approach for QEC is taken into account. For the concatenation code, the threshold is quite high, so the gates must be ridiculously good whereas, for Surface Code, the threshold is comparatively lower. So, we don't need to make the gates too perfect. By looking stabilizer we can tell what the state is, even if the state is not mentioned explicitly keeping in mind that the eigenvalue of an operation must be $+1$. For example, if we have the operation Z, the state must be $|0\rangle$. So Z is the stabilizer for $|0\rangle$. If M is a stabilizer of $|\psi\rangle$, then $M|\psi\rangle = |\psi\rangle$.

16.4 Surface Code

In this section, we will see a very brief introduction of *Surface Code*. There is more than one type of way of encoding physical qubits into a logical qubit. Surface code is one of the popular approaches to building a scalable quantum computer. It was evolved from a simple model of *toric code* developed by Alexei Kitaev [17].

One of the advantages of Surface Code, as described by Preskill and his team, is its tolerance to local errors. They used a stack of layered surfaces to implement the logical CNOT gate. Though the three-dimensional structure is making it complicated but they were able to show that the ability to handle the error of the code is almost 3% per clock cycle.

Then Raussendforf and his team showed that the logical CNOT gate can be implemented by braid operation on a single surface, which made it easier a simple. They also showed that using only one or two-qubit nearest-neighbor gates the tolerance of each cycle arrives at the threshold of 0.75%.

In Surface Code, physical qubits are connected by CNOT gates to create one logical qubit. Due to entanglement and measurement, this logical qubit has a far better performance than the physical qubit.

Single qubit error can be corrected by applying error-correcting code repeatedly. For example, an erroneous Z can not be corrected by Z but would only be affected by an X-measurement. Likewise, an X-error would only affect Z-measurement. So, in surface code, an error needs to be corrected if it affects the measurement. Thus the main focus of surface code is to detect error and not correct it.

$\hat{Z}_a\hat{Z}_b$	$\hat{X}_a\hat{X}_b$	$\vert\psi\rangle$
$+1$	$+1$	$(\vert 00\rangle + \vert 11\rangle)/\sqrt{2}$
$+1$	-1	$(\vert 00\rangle - \vert 11\rangle)/\sqrt{2}$
-1	$+1$	$(\vert 01\rangle + \vert 10\rangle)/\sqrt{2}$
-1	-1	$(\vert 01\rangle - \vert 10\rangle)/\sqrt{2}$

In this table, four eigenstates are given with their eigenvalues measured by $\hat{X}_a\hat{X}_b$ and $\hat{Z}_a\hat{Z}_b$. These are the bell states and form a complete set of two-qubit systems. For the state $|00\rangle + |11\rangle$ with eigenvalues $(+1, -1)$, if $\hat{X}_a$ error is applied then the state becomes $|10\rangle + |01\rangle$ with eigenvalues $(-1, +1)$. Now it can be seen that an $\hat{X}_b$ error would also give the same final state. The final state for $\hat{Z}_a$ and $\hat{Z}_b$ error is also the same. So the error can not be distinguished properly while they can be detected. That's why a more complex error-correcting code is needed.

The *Stabilizers* are very important in error correcting codes as they preserve the quantum state. If M is a stabilizer of $|\psi\rangle$ then $M|\psi\rangle = +|\psi\rangle$ with $+1$ eigenvalue. Here $\hat{X}_a\hat{X}_b$ and $\hat{Z}_a\hat{Z}_b$ are the stabilizers. These measures the states without changing the measurement outcome, if the eigenvalue changes to -1, error is assumed to be detected.

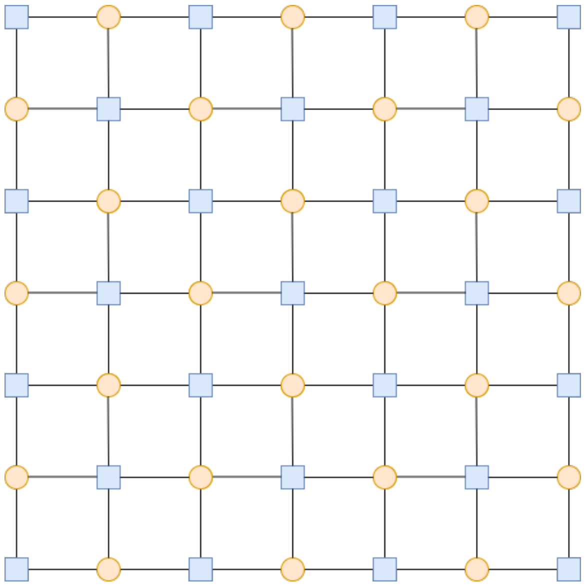


Figure 43: Grid of a surface code. The blue qubits represent ancilla qubits and the orange qubits represent data qubits

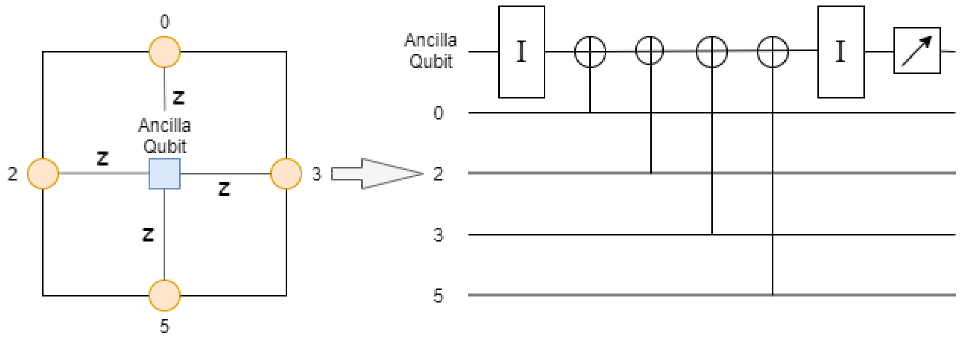


Figure 44: Z-stabilizer measurement

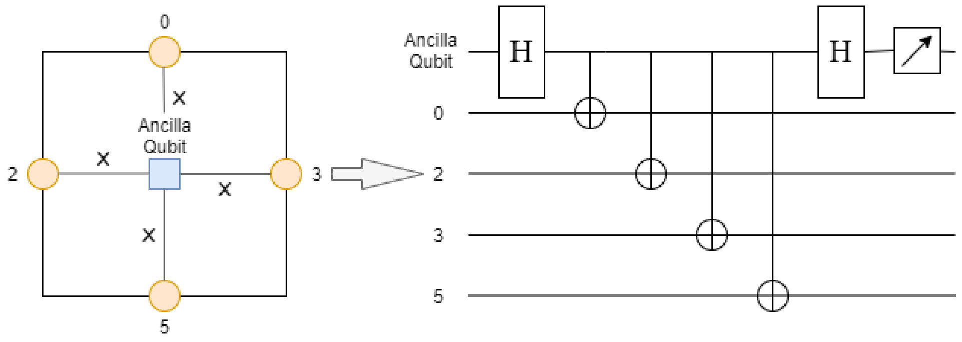


Figure 45: X-stabilizer measurement

Fig. (43) is a surface code grid where blue ones are the ancilla qubits and orange are the data qubits. The data qubits are connected with their neighboring ancilla qubit with CNOT gates. The picture is more clear in the next two figures. Fig. (44) shows the Z-measurement that stabilizes the surface code consists of CNOT gates that target the measurement qubit with four data qubits. Fig. (45) shows X-measurement that targets the nearest data qubits with ancilla qubit with Hadamard gates on both sides of the set of CNOTs. This measures qubit in lock-step and continues to each segment of the entire 2D structure.

16.5 Surface-17 code

In this study, the performance of the Tensor-Network-based approach is studied on the error model particularly on surface-17 code. The code looks as follows

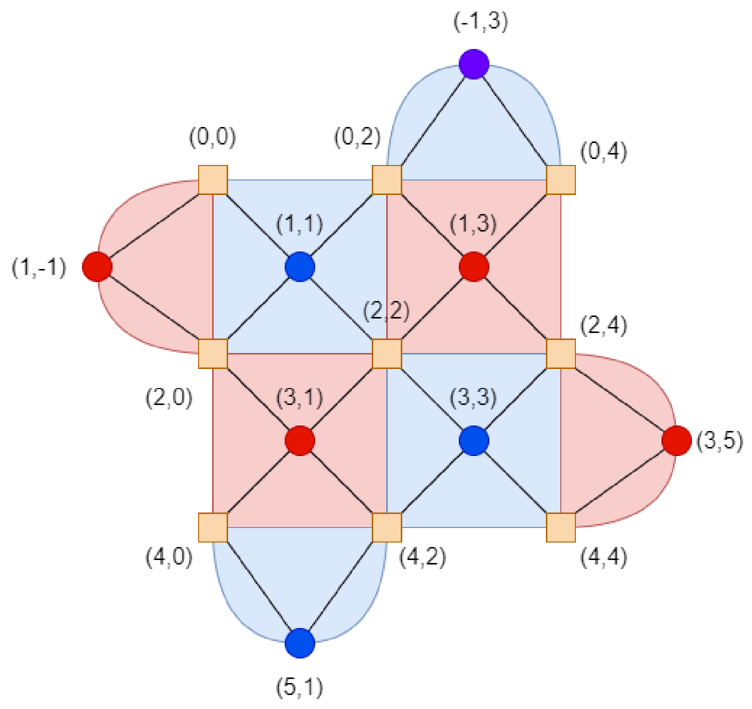


Figure 46: Design of surface-17 code

This is defined on a 3×3 grid with 9 physical qubits. There are four ancilla qubits for X-stabilizer and four for Z-stabilizer. In this design, red circles are X-qubits, blue circles are Z-qubits, square orange shapes are data qubits and red and blue shaded regions are X- and Z-stabilizers respectively. For example, X-ancilla at $(1, -3)$ measures $X_{(0,2)}X_{(0,4)}$ and Z-ancilla at $(3, 1)$ measures $Z_{(2,0)}Z_{(2,2)}Z_{(4,0)}Z_{(4,2)}$. The data stores info about logical qubits and ancilla qubits measures the error syndrome without disturbing the data qubit. And this is a [9,1,3] quantum code.

16.6 Experimental Error Models

We are to study the simulation of the error models on Surface-17 code that I have introduced earlier with tensor-network-based approaches. For this, we have to discretize the time evolution. This error would be introduced into the simulation. Though the errors are significant, but still sufficiently okay to understand the effects of crosstalk. We will first see the different error models.

16.6.1 Idle Error

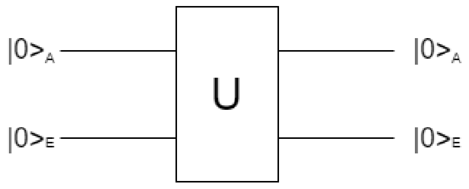
When no gate is applied on a qubit, that idling qubit can still undergo an error which is called *Idle error*. It consists two components:

- **Amplitude damping**

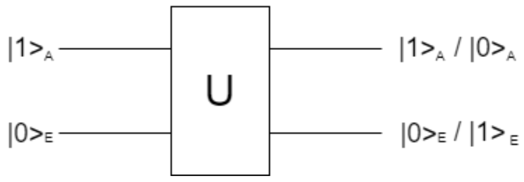
  A system A whose ground state is $|0\rangle_A$ & excited state is $|1\rangle_A$ which can emit a photon to environment and relax to ground state.

  1. The relaxation takes time 't' with probability p_d.
  2. Add an ancillary qubit $|0\rangle_E$ which is basically the environment E.
  3. Apply a unitary gate such that

$$U|00\rangle_{AE} = |00\rangle_{AE}, U|10\rangle_{AE} = \sqrt{1-p_d}\,|10\rangle_{AE} + \sqrt{p_d}\,|01\rangle_{AE} \tag{16.1}$$



(a) $U|00\rangle_{AE} = |00\rangle_{AE}$



(b) $U|10\rangle_{AE} = \sqrt{1-p_d}\,|10\rangle_{AE} + \sqrt{p_d}\,|01\rangle_{AE}$

4. Discard ancillary qubit of E.

- **Phase damping**

  This type of damping happens due to the energy which is weaker than the system energy but stronger than the environment energy. This causes the qubit to dephase into the computational basis $|0\rangle_A, |1\rangle_A$. This model can be simulated following the same steps above.

  1. The relaxation takes time 't' with probability p_ϕ.
  2. Add an ancillary qubit $|0\rangle_E$ which is basically the environment E.
  3. Apply a unitary gate such that

  $$U|00\rangle_{AE} = |00\rangle_{AE}, U|10\rangle_{AE} = \sqrt{1-p_\phi}|10\rangle_{AE} + \sqrt{p_\phi}|11\rangle_{AE}. \quad (16.2)$$

  4. Discard ancillary qubit of E.

As the excited states decay exponentially in time. So,

$$1 - p_d = e^{-t/T_d}, 1 - p_\phi = e^{-t/T_\phi}. \quad (16.3)$$

16.6.2 Unitary Gate Errors

There are two types of gates in the syndrome extraction circuit. The first one is $R_y(\pm\pi/2)$ which is modeled as depolarizing noise which means applying X, Y, and Z gates errors to the qubits with certain probabilities. The gate-specific error shrinks the Bloch sphere along the X- and Z-axis by a factor of $1-P_{XZ}$ and Y-axis by $1-P_Y$. The second one is the CZ gate, which is modeled as 'quasi-static' that means the phase error on each adjacent qubit pair is constant through time in a single run but random over multiple runs.

16.6.3 Measurement Error

In a superconducting quantum device, measurement is taken by introducing photons into readout resonators. The qubit is then dephased and measurement is taken. Then the photon is left sometimes, allowing it to deplete from the resonator.

In [18], this error is modeled as a 'butterfly gate'. They also note that the experimental parameters can be assumed as

- Amplitude -phase damping applies just before and after the measurement happens instantaneously at the halfway of the measurement.
- The classical output further faces the readout error ϵ_{RO} which is independent of the outcome.

During the photon depletion period, the amount of photons in the resonator decays over time but does not decrease to a completely negligible level. The mechanics of this process is a bit complex, so we just use the expression as follows

$$p_{\text{ photon}} = \exp\left(2\eta\alpha(0)\exp\left(s\left(t_m - t_r\right)\right)\left[\frac{e^{-st}}{4\eta^2+s^2}[-s\sin(2\eta t) - 2\eta\cos(2\eta t)]\right]_{t_1-t_r}^{t_2-t_r}\right). \tag{16.4}$$

Where, t_m is the starting of measurement period, t_g is the time taken to change the basis of the state, s and η are constant parameters.

16.7 Crosstalk modeling

In Quantum Computation, crosstalk or ZZ-coupling between two adjacent idling qubits is a parasitic error that we try to control at our will but in practice that is not the case. In this section, we will focus on the crosstalk and crosstalk in circuit models with different idle errors.

It is not surprise that on superconducting qubit, crosstalk error arises from CZ interaction and has the form

$$e^{ik(Z\otimes Z)} = e^{ik(|00\rangle\langle 00|-|01\rangle\langle 01|+|10\rangle\langle 10|-|11\rangle\langle 11|)} = e^{ik(-I\otimes I+Z\otimes I+I\otimes Z+4|11\rangle\langle 11|)}. \tag{16.5}$$

Here, the expression $e^{ik(Z\otimes Z)}$ is decomposed into four parts where e^{ik} is the global phase and thus we can ignore this. $e^{ik(Z\otimes I)}$ and $e^{ik(I\otimes Z)}$ are the one-qubit gates known as PHASE gates. The last term $e^{4ik(|11\rangle\langle 11|)}$ is the CPHASE gate of which the CZ gate is a special case.

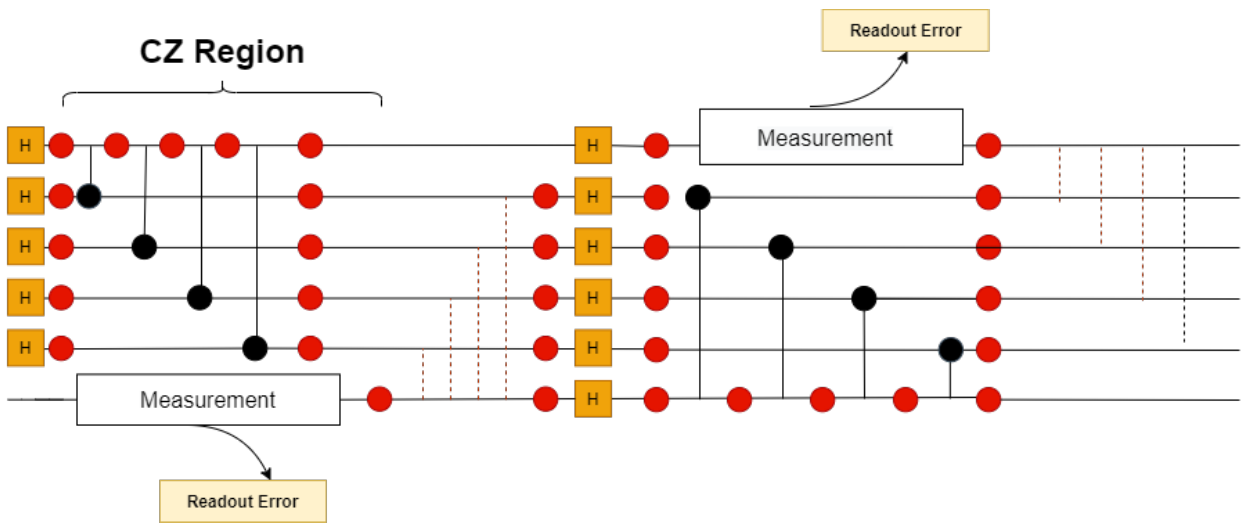


Figure 48: Error model of Surface-17 code implemented in Quantum Circuit

Here, the circuit model is not exact but this rough approximation is considered for the sake of simplicity. The strength of the crosstalk is taken to be constant irrespective of the time and qubit pair. In practice, this actually depends on the frequency of each qubit, but for an approximated model, it works well. To make the whole syndrome extraction simple the effect of crosstalk is discretized at the cost of simulation accuracy. The orange rectangular boxes represent noisy Hadamard gates, red circles represent idling errors, black diagrams represent gate specific errors and red dotted lines represent the crosstalk in the circuit.

In this circuit, data qubits are only adjacent to the ancilla qubit and vice versa. Now, as seen above, this circuit is divided into two regions by $R_y(\pm\pi/2)$ for each qubit pair.

- **CZ Regions:** In this region CZ gates are calibrated such that crosstalk does not present in this region while the effect of crosstalk is already taken into account for the CZ gate itself.
- **Measurement Regions:** In this region, all the crosstalks are moved to the same time point - the end of the measurement i.e. just before the next round. Thus the tensor network for the error model is simplified.

16.8 Simulating the noisy circuit with Tensor Network

In the case of error models, some unitary gates even ‘idle wires’ become non-unitary. So, it’s a bit different from simulating the ideal Quantum case. But as simulating all qubit channels is, kind of, same i.e adding ancilla qubit, applying unitary gate and discarding the ancilla qubit, so it still can be simulated but then we have so many quantum measurements for the environment and thus it’d take so many samples of those to get a reliable result.

Therefore the density approach is considered instead of state vector simulation. The density matrix still contains all the information about classical randomness for which it's not necessary to worry about sampling. This method takes the advantage of a parallel tensor network algorithm. Each input qubit becomes an order-2 tensor quantum channel is an order-4k tensor when applied to k-qubit. All the edges including closed and open edges are doubled for the orders of the tensors to be matched.

Some of the above-mentioned tensors represent only pure objects and unitary gates. However, if a tensor is decomposed into two tensors, one is the complex conjugate of the other and the connection of those two is the non-pure object in the circuit. When some tensors are contracted in a network, all closed indices are summed over one by one. For an open tensor network, some indices can not be summed over. And as we see in Sec. [16.2], one can contract a network in different ways. So a contraction order is to be optimized such that the time and space complexities are reduced. This tensor network-based approach is better than the state vector update approach regarding efficiency. Though finding the optimal contraction order is an NP-hard problem and takes exponential time.

16.9 Comparison

There have been a lot of works based on the real-life error models but most of those were done in sample-based approach where lots of samples are taken after which one can come to a reliable conclusion. Though every trial may take a relatively short time, in order to reduce the variance sampling-based approaches usually suffer from the large number of random trials needed. On the other hand tensor network-based approach is a single-trial method. Though a single trial takes much more time still it's more efficient.

Obviously, the method one should adopt depends on the experiment. There might be cases where a sample-based approach is much more feasible. But in this case for Surface-17 code with real-life error model, tensor network based approach is better.

17 Conclusion

This is an overview of basic error-correcting codes starting from [3,1] repetition code to the larger lass of error codes i.e. Quantum CSS code. This tells you how one can understand the syndrome measurement to detect an error and correct them, if possible, how fidelity gets improved when error-correcting code is applied, and also a general view on the condition and limitation that an error-correcting code has to follow to detect and correct error efficiently.

Then a review of tensor network and surface code is studied and a lattice of Rydberg atom under dipole-dipole interaction is studied by DMRG calculation in *JULIA* and it's seen that depending on the number of Rydberg atoms on the lattice a trend of energy values can be seen which is depicted in the Fig. (34). Then realistic approximated error model is realised in the quantum circuit and it's seen that the tensor-network-based approach is more efficient than the sample-based approach. The overview is important for the mutual relation between Tensor Network and Quantum Error-Correcting Codes which can make a large-scale Quantum Computer with abilities to perform calculations that a classical computer cannot perform.

References

[1] E. Knill and R. Laflamme. Theory of quantum errorcorrecting codes. Phys. Lett. A, 55:900, 1997

[2] F. J. MacWilliams and N. J. A. Sloane. The Theory of Error-correcting Codes. North-Holland, Amsterdam, 1977.

[3] W. K. Wootters and W. H. Zurek, A single quantum can- not be cloned, Nature (London) 299, 802 (1982).

[4] P. Shor. Scheme for reducing decoherence in quantum computer memory. Phys. Rev. A, 52:2493, 1995.

[5] M.A. Nielsen and I.L. Chuang, Quantum computation and quantum information, Cambridge University Press (Cambridge, 2000)

[6] R. Penrose, "Applications of Negative Dimensional Tensors," Academic Press Inc., Burlington, 1971.

[7] White, Steven R. (1992-11-09). "Density matrix formulation for quantum renormalization groups". Physical Review Letters. American Physical Society (APS). 69 (19): 2863–2866.

[8] Browaeys, A., Lahaye, T. Many-body physics with individually controlled Rydberg atoms. Nat. Phys. 16, 132–142 (2020). https://doi.org/10.1038/s41567-019-0733-z

[9] `https://deepblue.lib.umich.edu/bitstream/handle/2027.42/163098/fangzh_1.pdf?sequence=1&isAllowed=y`

[10] G. E. Moore,BCramming more componentsonto integrated circuits,[Electronics,vol. 38, pp. 114–117, 1965.

[11] Cory, D. G.; Price, M. D.; Maas, W.; Knill, E.; Laflamme, R.; Zurek, W. H.; Havel, T. F.; Somaroo, S. S. (1998). "Experimental Quantum Error Correction". Phys. Rev. Lett. 81 (10): 2152–2155

[12] https://www.quantamagazine.org/does-nevens-law-describe-quantum-computings-rise-20190618/

[13] P. W. Shor. Fault-tolerant quantum computation. In Proceedings, 37th Annual Symposium on Fundamentals of Computer Science, pages 56–65, IEEE Press, Los Alamitos, CA, 1996.

[14] A. M. Steane. Multiple particle interference and quantum error correction. Proc. R. Soc. London A, 452:2551–76, 1996.

[15] Arute, F., Arya, K., Babbush, R. et al. Quantum supremacy using a programmable superconducting processor. Nature 574, 505–510 (2019). https://doi.org/10.1038/s41586-019-1666-5

[16] William E. Ryan and Shu Lin (2009). Channel Codes: Classical and Modern. Cambridge University Press. p. 4. ISBN 978-0-521-84868-8.

[17] A.Yu. Kitaev, Fault-tolerant quantum computation by anyons, Annals of Physics, Volume 303, Issue 1, 2003, Pages 2-30, ISSN 0003-4916, https://doi.org/10.1016/S0003-4916(02)00018-0.

[18] TE O'Brien, B Tarasinski, and L DiCarlo. Density-matrix simulation of small surface codes under current and projected experimental noise. npj Quantum Information, 3(1):39, 2017.

[19] A.R. Calderbank and P.W. Shor. Good Quantum Error-Correcting codes exist. Phys. Rev. A., 54:1098, 1996

[20] R. Laflamme, C. Miquel, J.P. Paz, and W.H. Zurek. Perfect Quantum Error Correcting Code. Phys. Rev. Lett., 77:198, 1996.

[21] E. Knill. Non-binary Unitary Error Bases and Quantum Codes. LANL report LAUR-96-2717, quantph/9608048, 1996.

A Description of the *JULIA* code

A.1 DMRG calculation for 2×1 Rydberg atom lattice for calculating minimum energy value

```
Julia>  let
           Ny = 1
           Nx = 2

           N = Nx*Ny

           sites = siteinds("S=1/2", N;
                             conserve_qns = true)

           lattice = square_lattice(Nx, Ny; yperiodic = false)

           ampo = AutoMPO()
             for b in lattice
             ampo .+= (2.63*10^(-29)), "S+",b.s1
             ampo .+= (2.63*10^(-29)), "S-",b.s1
             ampo .+= 0.5, "S+", b.s1, "S-", b.s2
             ampo .+= 0.5, "S-", b.s1, "S+", b.s2
             ampo .+= -(2.63*10^(-34)),"Sz", b.s1
           end
           H = MPO(ampo,sites)

             state = [isodd(n) ? "Up" : "Dn" for n=1:N]

           psi0 = randomMPS(sites,state,20)
```

```
          sweeps = Sweeps(10)
            maxdim!(sweeps,20,60,100,100,200,400,800)
              cutoff!(sweeps,1E-8)
               @show sweeps

        energy,psi = dmrg(H,psi0,sweeps)

        return
      end
     sweeps = Sweeps
```

A.2 Output:

```
1 cutoff=1.0E-08, maxdim=20, mindim=1, noise=0.0E+00
2 cutoff=1.0E-08, maxdim=60, mindim=1, noise=0.0E+00
3 cutoff=1.0E-08, maxdim=100, mindim=1, noise=0.0E+00
4 cutoff=1.0E-08, maxdim=100, mindim=1, noise=0.0E+00
5 cutoff=1.0E-08, maxdim=200, mindim=1, noise=0.0E+00
6 cutoff=1.0E-08, maxdim=400, mindim=1, noise=0.0E+00
7 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
8 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
9 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
10 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00

After sweep 1 energy=-34.556755246332 maxlinkdim=20 maxerr=1.43E-03 time=1.640
After sweep 2 energy=-35.249039532643 maxlinkdim=60 maxerr=2.56E-05 time=3.202
After sweep 3 energy=-35.347114032405 maxlinkdim=100 maxerr=2.41E-05 time=3.582
After sweep 4 energy=-35.349310987151 maxlinkdim=100 maxerr=3.27E-05 time=4.128
After sweep 5 energy=-35.371386140634 maxlinkdim=200 maxerr=3.58E-06 time=15.958
After sweep 6 energy=-35.374719143140 maxlinkdim=400 maxerr=3.69E-07 time=47.179
After sweep 7 energy=-35.375102171408 maxlinkdim=800 maxerr=2.26E-08 time=86.461
After sweep 8 energy=-35.375110787349 maxlinkdim=800 maxerr=4.63E-08 time=114.327
After sweep 9 energy=-35.375111234217 maxlinkdim=800 maxerr=4.87E-08 time=6607.839
After sweep 10 energy=-35.375111292367 maxlinkdim=800 maxerr=4.93E-08 time=121.445
```

Here in this above code, I have calculated the Ground State Energy of the lattice of 2×1 (Effectively a chain) Rydberg atoms with dipole-dipole interaction. To do that, I use the *AutoMPO system* and *DMRG algorithm* of ITensor package in *JULIA* programming language. The Hamiltonian in Eq. (15.1) is made to undergo this calculation. At first, we need to write the σ_x^i matrix in terms of σ_+^i and σ_-^i to conserve the quantity called *'Quantum Numbers'* in the ITensor package. First I define the number of particles of spin-1/2 in each direction. Then under the loop, it's quite understandable that the Hamiltonian is reproduced. The *AutoMPO* system converts the construction of the Hamiltonian MPO. So the Hamiltonian H is defined.

Now moving forward to the DMRG calculation where the inputs are the Hamiltonian of the system and an initial guess of the ground state psi0. So I initialize the guess of my ground state. Then I set the number of sweeps as 10 and increase the maximum MPS bond-dimension gradually from 20 to 800. The DMRG algorithm iterations (sweeps) will run until it reaches the cutoff 10^{-8}. then we run the DMRG function which takes the Hamiltonian, initial guess state, and the iteration of the algorithm as its arguments and stores it in 'energy'. In the output, the last energy is the lowest energy of the system after the maximum number of sweeps.

In the next section, the same code is run but for the system of 20×10 lattice of the Rydberg system. And thus, all other energies are calculated and plotted in Fig. 34.

A.3 DMRG calculation for 20×10 Rydberg atom lattice for calculating minimum energy value

```
julia>  let
          Ny = 10
          Nx = 20

          N = Nx*Ny

          sites = siteinds("S=1/2", N;
                           conserve_qns = true)

          lattice = square_lattice(Nx, Ny; yperiodic = false)

          ampo = AutoMPO()
            for b in lattice
            ampo .+= (2.63*10^(-29)), "S+",b.s1
            ampo .+= (2.63*10^(-29)), "S-",b.s1
            ampo .+= 0.5, "S+", b.s1, "S-", b.s2
            ampo .+= 0.5, "S-", b.s1, "S+", b.s2
            ampo .+= -(2.63*10^(-34)),"Sz", b.s1
          end
          H = MPO(ampo,sites)

            state = [isodd(n) ? "Up" : "Dn" for n=1:N]

          psi0 = randomMPS(sites,state,20)

            sweeps = Sweeps(10)
              maxdim!(sweeps,20,60,100,100,200,400,800)
```

```
                cutoff!(sweeps,1E-8)
                 @show sweeps

          energy,psi = dmrg(H,psi0,sweeps)

          return
        end
      sweeps = Sweeps
```

A.4 Output

```
1 cutoff=1.0E-08, maxdim=20, mindim=1, noise=0.0E+00
2 cutoff=1.0E-08, maxdim=60, mindim=1, noise=0.0E+00
3 cutoff=1.0E-08, maxdim=100, mindim=1, noise=0.0E+00
4 cutoff=1.0E-08, maxdim=100, mindim=1, noise=0.0E+00
5 cutoff=1.0E-08, maxdim=200, mindim=1, noise=0.0E+00
6 cutoff=1.0E-08, maxdim=400, mindim=1, noise=0.0E+00
7 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
8 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
9 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00
10 cutoff=1.0E-08, maxdim=800, mindim=1, noise=0.0E+00

After sweep 1 energy=-97.230274525055 maxlinkdim=20 maxerr=2.64E-03 time=15.100
After sweep 2 energy=-100.359663404246 maxlinkdim=60 maxerr=1.03E-04 time=4.454
After sweep 3 energy=-101.477698578037 maxlinkdim=100 maxerr=2.56E-04 time=9.086
After sweep 4 energy=-101.727665081697 maxlinkdim=100 maxerr=3.29E-04 time=9.214
After sweep 5 energy=-102.364420408849 maxlinkdim=200 maxerr=7.43E-05 time=28.079
After sweep 6 energy=-102.631739108426 maxlinkdim=400 maxerr=2.53E-05 time=107.104
```