Department of Mathematics

Academic Year 2021 - 2022

Efficient Classical Simulation of a Variant of Cluster State Quantum Computation

Sahar Atallah
1830214

A thesis submitted for the degree of Doctor of Philosophy

Brunel University London
Department of Mathematics
Uxbridge
Middlesex
UB8 3PH
United Kingdom
T: +44 1895 203397
F: +44 (0) 1895 251686

Acknowledgments

This thesis would not have been possible unless the help of many people for whom I will be thankful.

I owe my deepest gratitude to my supervisor Dr Shashank Virmani, who showed me the significance of understanding every idea from all its perspectives, who helped me passing every one of the obstacles I went up against amid the entire procedure, and who improved my work with his insightful guidance. It has been such a benefit for me to be supervised by a man with endless experience and boundless kindness and patience. I might want to express appreciation toward him for acquainting me with the subject of efficient classical simulation of a variant of cluster state quantum computation.

I would like to thank Dr Anne-Sophie Kaloghiros, for her help, time and guidance through the entire procedure.

I am grateful toward Dr Dmitry Savin and Dr Ilia Krasikov for their help and kind support.

I am indebted to my friends, Michael Garn and Yukuan Tao who provided the help I needed in my research. I am thankful to many of my friends for their help and ethical support. My unique appreciation to Rodrigue Kazzi for backing me in any decision I made. I find myself fortunate to have companions like them in my life.

Finally, I cannot disregard my family who encouraged me in all circumstances. It is an honor for me to thank my father and my mother for their unconditional love and support.

Abstract

Quantum computers are known for their ability to solve some computational problems faster than classical computers. There is a race to build quantum computers because it is believed they might be better than classical; but it remains unknown whether quantum computers are in fact better than conventional computers. To understand this problem, we develop a new method of classically simulating certain types of quantum system that are previously unknown to be efficiently simulatable on classical computers.

We adjust a part of cluster state quantum computation to study the computational power and we demonstrate that there is a finite region of pure states $|\psi\rangle$ around the Z-eigenstates for which the setup can be efficiently simulated classically, given that the measurements are limited to Z and $X-Y$ plane measurements. This classical simulation works by considering alternative local state spaces that we called "cylinders" and different notion of entanglement to normal quantum entanglement.

Then, we work out similar regions for states created using other diagonal gates instead of the CZ. These diagonal gates are represented by $V(\theta) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z_\theta$ where $Z_\theta = |0\rangle\langle 0| + e^{i\theta}|1\rangle\langle 1|$. It turns out that almost all inputs are classically simulatable when θ is small.

In addition, we find that classical simulation also works by considering new type of non-quantum state spaces other than cylinders and maintaining non-entangled representation by growing the size of these state spaces. We search over some state spaces to try optimize our classical simulation and it turns out that, among the state spaces that we searched through, the cylinder is the most optimal state space.

And finally, we will look at a coarse graining version of construction which increases the efficiently simulatable region.

Contents

Chapter 1

Introduction

Quantum computing is the use of quantum phenomena, such as the so-called entanglement and superposition, to perform computation. Computers that performs quantum computations are known as quantum computers. These computers are able to solve some computational problems, such as factorization in polynomial time, exponentially faster than the best known algorithm on classical computers. Algorithms that are performed on quantum computers, are called quantum algorithm [22, 23, 44, 42, 56]. For example, Shor [57] who works on factoring integers; this problem is believed to be hard on a classical computer but efficient algorithms are given for this problem on a quantum computer. An additional example is that Grover [30] discovers an algorithm that can solve a problem on a quantum computer using quadratic speedup operations.

Before mentioning the goal of the research, let's start by revealing why we are interested in simulating quantum systems on a classical computer. The first reason is that it is unknown if quantum computers are better than conventional computers. To understand this problem, we can proceed one of two ways. We can try to find that quantum computers can't be efficiently simulated classically or we can try to develop an algorithm that can classically simulate a quantum system.

The motivation of simulating a complex quantum system efficiently classically is that people think that quantum computers, built with quantum systems, can solve problems such as factorization that can't be done classically; but if it is found that simulating quantum computer can be done classically, then it means that any algorithm can be done on quantum computers, it can be efficiently simulated classically. For instance, there would be an efficient algorithm for factorization on a classical computer.

Progress has been made in efficiently simulating a quantum system on a classical computer. For example, *Gottesman-Knill* theorem [27] shows that a quantum computation, that involves only the state preparations in the computational basis,

Hadamard gates, Phase gates, Controlled-NOT gates, Pauli gates and measurements in the Pauli group, may be efficiently simulated on a classical computer. So, it is an algorithm that only simulate certain type of quantum system. Gottesman-Knill theorem displays that some quantum computations, including entangled states (but not all types of entanglement) may be efficiently simulated classically. Hence, Gottesman-Knill theorem is very interesting because there are connections to error correction and only single particle operations are needed to turn stabilizer circuits into a form of quantum computation.

Most people in this field believe that it is not possible to efficiently simulate quantum systems classically. One reason is that factorization can be done efficiently on quantum computer, in spite of the fact that over many years people have failed to find an efficient factorization algorithm on a classical computer. But there is no proof of this fact nor the fact that quantum system cannot be efficiently simulated classically.

In this thesis, we aim to develop a new method of classically simulating certain types of quantum system that are previously unknown to be efficiently simulatable on classical computers.

1.1 Basic mathematics

In this section, we are going to explain the definitions and the formulas of all the mathematical terms used in this research.

1.1.1 Hilbert space

We begin with the definition of the fundamental mathematical concept, *"the Hilbert Space"* which is denoted by H.

The Hilbert spaces permit generalising the linear algebra and calculus methods from the two and three dimensional Euclidean spaces to other dimensions spaces and to spaces that have an infinite dimension, but we will only be concerned with finite dimensional systems. Hilbert space is a complex inner product space which means that it is a complex *vector space* endowed with an *inner product* operation, we will shortly explain the meaning of these terms.

The vector state space is a set of elements, called vectors such that the set must satisfies the following properties:

1. Any vector $|v\rangle$ can be multiplied by any number called scalar. Then the vector is denoted by $\alpha|v\rangle$, where α can be a real or complex scalar depending upon the vector space. If the scalar is a real number, then the space is called a *real vector space*, and if the scalar is a complex number, then the space is a *complex vector space*.

2. For any vector $|v_1\rangle$ and any two scalars α, β, it holds that $(\alpha + \beta)|v_1\rangle = \alpha|v_1\rangle + \beta|v_1\rangle$.

3. For any vector $\alpha|v\rangle$ and any scalar β, there is a vector denoted by $\beta(\alpha|v\rangle) = (\beta\alpha)|v\rangle = (\alpha\beta)|v\rangle$.

4. The vectors can be added together. For instance, for any two vectors $|v_1\rangle$ and $|v_2\rangle$, there is a vector $|w\rangle$ denoted by $|w\rangle = |v_1\rangle + |v_2\rangle$.

5. The sum of vectors must be associative and commutative. For instance, by associative and commutative, we mean $(|v_1\rangle + |v_2\rangle) + |v_3\rangle = |v_1\rangle + (|v_2\rangle + |v_3\rangle)$ and $|v_1\rangle + |v_2\rangle = |v_2\rangle + |v_1\rangle$ respectively.

6. The multiplication by α is distributive such that $\alpha(|v_1\rangle + |v_2\rangle) = \alpha|v_1\rangle + \alpha|v_2\rangle$.

7. By multiplying any vector $|v\rangle$ by 1, $|v\rangle$ remains the same.

8. There exists a zero vector 0 such that by adding any $|v\rangle$ to 0, the same $|v\rangle$ will be given, i.e. $|v\rangle + 0 = |v\rangle$.

9. For any vector $|v\rangle$, there is an inverse for this vector denoted $-|v\rangle$, such that $|v\rangle + (-|v\rangle) = 0$.

A set of vectors $|a_1\rangle, |a_2\rangle, \ldots, |a_d\rangle$ is a *spanning set* of a vector space E if $\forall u \in E, \exists \beta_1, \beta_2, \ldots, \beta_d \in \mathbb{K}$ such that $u = \beta_1|a_1\rangle + \beta_2|a_2\rangle + \ldots + \beta_d|a_d\rangle$. A set of vectors is said to be *Linearly Independent* if one of the following two equivalent cases holds:

- if the basis vectors give a unique expansion,
- if the zero vector 0 has a unique expansion, $0 = m|a_1\rangle + n|a_2\rangle + p|a_3\rangle + \ldots + q|a_d\rangle$, where the coefficients, $m, n, p, \ldots, q$ must be zeros.

Hence, a set of vectors is a *basis* if the set is a linearly independent spanning set.

The total number of vectors, which is d in the above basis set, represents the *dimension* of a vector space.

Some vector spaces can be given an inner product which is a map of two input vectors to a complex output number. Using the *Dirac notation* or *bra-ket notation*, which is a language that match the needs of expressing states in quantum mechanics, an inner product is obtained by combining two vectors, the *bra* and the *ket* vectors, denoted by $\langle g|$ and $|h\rangle$ respectively. The bra vector $\langle g|$, is a linear map that takes a vector $|h\rangle$ and gives $\langle g|h\rangle$ which represents the inner product.

The inner product is defined to satisfy the following axioms:

1. The inner product $\langle g|h\rangle$ must be equal to its *adjoint* $\langle g|h\rangle^* = \langle h|g\rangle$. This equality is called the *conjugate symmetry*. By adjoint, we mean that it is a linear transformation from a vector space to itself and this linear map leads to the transpose conjugate.

2. When a vector $\langle v|$ acts on a sum of two vectors $(|h\rangle + |w\rangle)$, $\langle v|h\rangle + \langle v|w\rangle$ will be given.

3. When a vector $|h\rangle$ is multiplied by a scalar t, then $\langle g|\ (t|h\rangle) = t\langle g|h\rangle$.

4. For all vectors $|h\rangle$, $\langle h|h\rangle \geq 0$.

5. If $|h\rangle$ is a zero vector, then $\langle h|h\rangle = 0$.

6. If $\langle h|$ acts on a zero vector $|0\rangle$, then $\langle h|0\rangle = \langle 0|h\rangle = 0$.

If the inner product of two vectors, $|g\rangle$ and $|h\rangle$ is 0, which means that $\langle g|h\rangle = \langle g|h\rangle^* = \langle h|g\rangle = 0$, then these two vectors are defined to be *orthogonal*.

In vector spaces, the norm, which is a non-negative real valued function, is used to define lengths of vectors. The norm which is denoted by $||\ |v\rangle\ ||$, satisfies three conditions; for instance, by taking a vector $|x\rangle$ in H:

1. For all vectors $|x\rangle$, $||\ |x\rangle\ || \geq 0$ which means $||\ -|x\rangle\ || = ||\ |x\rangle\ ||$ with $||\ |x\rangle\ || = 0$ if $|x\rangle = 0$.

2. By multiplying the vector $|x\rangle$ by a scalar w, then the norm is given by $||\ w|x\rangle|| = |w| \times ||\ |x\rangle\ ||$.

3. By taking two vectors $|x\rangle$ and $|y\rangle$, the norm of the sum of these two vectors is given by $||\ |x\rangle + |y\rangle\ || \leq ||\ |x\rangle\ || + ||\ |y\rangle\ ||$ which represents the triangle inequality.

The inner product automatically leads to a type of norm. Then the norm of a vector space that has an inner product is given by:

$$|| \ |x\rangle||_2 = \sqrt{\langle x|x\rangle} = \sqrt{\sum_{i=1} |x_i|^2}.$$

Let's consider a basis of vectors $|a_1\rangle, |a_2\rangle, |a_3\rangle, \ldots$. If the inner product of two vectors

$$\langle a_i|a_j\rangle = 0$$

for all $i \neq j$ and if the norm of all vectors

$$\sqrt{\langle a_i|a_i\rangle} = 1 = \langle a_i|a_i\rangle$$

then the set is an *orthonormal basis*.

Vectors can be described in terms of expansion coefficients in an orthonormal basis. Let's consider any two vectors $|h\rangle = \sum_i \alpha_i|h_i\rangle$ and $|g\rangle = \sum_i \beta_i|g_i\rangle$ where $|g_i\rangle = |h_i\rangle$, then the inner product between the two vectors is given by:

$$\begin{aligned}\langle g|h\rangle &= \sum_i \beta_i^* \alpha_i \langle g_i|h_i\rangle \\ &= \sum_i \beta_i^* \alpha_i \ ; \text{ where } \langle g_i|h_i\rangle = 1 \text{ because they are normalised.}\end{aligned}$$

The inner product can be written as a product of two matrices:

$$\langle g|h\rangle = \begin{pmatrix} \beta_1^* & \beta_2^* & \ldots \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \end{pmatrix}$$

Therefore, the two vectors $|h\rangle$ and $|g\rangle$, also can be written as matrices. $|h\rangle$ which is the ket vector, is a column vector in a complex vector space and the bra vector $\langle g|$ is the adjoint of $|g\rangle$, these vectors are given by

$$|h\rangle = \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ \end{pmatrix}$$

and

$$\langle g| = \begin{pmatrix} g_1^* & g_2^* & \dots \end{pmatrix}.$$

1.1.2 Operator

An operator O maps input vectors into output vectors that are in two different Hilbert spaces, O: $H_{in} \rightarrow H_{out}$. For instance, we denote this transformation by the following equation

$$|w\rangle = O|v\rangle.$$

- An operator is *linear*, if the output vector $|w\rangle$ is the sum of all the output vectors, if $|v\rangle = a|v_1\rangle + b|v_2\rangle + \ldots$, then

$$|w\rangle = O|v\rangle = a\,O|v_1\rangle + bO|v_2\rangle + \ldots$$

- By considering an operator A that maps two vectors in two different Hilbert spaces, from $H_1 \rightarrow H_2$, its *adjoint operator* denoted $A^\dagger$ maps from $H_2 \rightarrow H_1$ such that $\langle Av_1|v_2\rangle = \langle v_1|A^\dagger v_2\rangle$. We will only be concerned with an adjoint operator that maps from a Hibert space to itself.

- A *Hermitian operator* is an operator A that satisfies $A^\dagger = A$ where $A^\dagger$ is the adjoint operator of A which also can be called *Hermitian conjugate*.

- *Unitary operators* U are linear transformation that maps an input vector $|\phi\rangle$ in H_1 to an output vector $|\psi\rangle$ in H_2. The two Hilbert spaces H_1 and H_2, must be equivalent, which means $H_1 = H_2$ and U has an inverse $U^\dagger$ which is the Hermitian conjugate, then

$$U^\dagger U = I.$$

- A *projector* or a *projection operator* P is a linear operator that maps a vector in vector space V to itself such that $P^2 = P$ which means that if P^2 is applied to a vector, the same outcome is obtained as if only one P is applied.

Operators can be represented as matrices when their actions are expressed in terms of orthonormal basis. Let's consider a linear operator O that maps an input vector $|v\rangle$ to an output vector $|w\rangle$. The input vector $|v\rangle$ can be expanded in terms of

an orthonormal basis:

$$\begin{aligned}|v\rangle &= \sum_i \alpha_i |v_i\rangle \\ &= \alpha_1|v_1\rangle + \alpha_2|v_2\rangle + \ldots,\end{aligned}$$

The coefficients column vector is given by

$$\begin{pmatrix}\alpha_1 \\ \alpha_2 \\ \vdots \\ \end{pmatrix}$$

After applying the operator on $|v\rangle$, it is the same as applying O on each vector $|v_1\rangle, |v_2\rangle, \ldots$, these output vectors that also can be expanded as a sum of basis vectors, are presented by

$$\begin{aligned}|w\rangle &= O|v\rangle \\ &= \sum_i \alpha_i O|v_i\rangle \\ &= \alpha_1 O|v_1\rangle + \alpha_2 O|v_2\rangle + \ldots, \\ &= \alpha_1(\beta_{1,1}|w_1\rangle + \beta_{2,1}|w_2\rangle + \ldots) + \alpha_2(\beta_{1,2}|w_1\rangle + \beta_{2,2}|w_2\rangle + \ldots) + \ldots, \\ &= \alpha_1 \left(\sum_j \beta_{j,1}|w_j\rangle\right) + \alpha_2 \left(\sum_j \beta_{j,2}|w_j\rangle\right) + \ldots, \\ &= \sum_i \alpha_i \left(\sum_j \beta_{j,i}|w_j\rangle\right) \\ &= \sum_j \left(\sum_i \beta_{j,i}\alpha_i\right)|w_j\rangle \\ &= \sum_j \gamma_j |w_j\rangle\end{aligned}$$

where $\gamma_j = \sum_i \beta_{j,i}\alpha_i$, it can be written as

$$\begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \end{pmatrix} = \begin{pmatrix} \beta_{1,1} & \beta_{1,2} & \beta_{1,3} & \dots \\ \beta_{2,1} & \beta_{2,2} & \beta_{2,3} & \dots \\ \beta_{3,1} & \beta_{3,2} & \beta_{3,3} & \dots \\ \vdots & & & \\ & \dots & & \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \vdots \end{pmatrix},$$

Then $|w\rangle$ can be represented by

$$\begin{pmatrix} \\ |w\rangle \\ \\ \end{pmatrix} = \begin{pmatrix} & & & \\ |w_1\rangle & |w_2\rangle & |w_3\rangle & \dots \\ & & & \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \\ \vdots \end{pmatrix}$$

Hence, the linear operator O can be represented by a matrix that takes input column vectors and gives output column vectors.

1.1.3 Tensor product

The *tensor product* is a map that take a pair of vector spaces, V and W with dimensions c and d respectively to a single vector space $V \otimes W$ that has a dimension of cd. The sign "$\otimes$" is the tensor product symbol.

The tensor product satisfies these axioms:

1. By taking two vectors $|v\rangle \in V$ and $|w\rangle \in W$, it holds that: $a(|v\rangle \otimes |w\rangle) = a(|v\rangle) \otimes |w\rangle = |v\rangle \otimes a(|w\rangle)$ where a is a scalar.
2. By adding two vectors $|v_1\rangle + |v_2\rangle$, for a vector $|w\rangle \in W$, we have $(|v_1\rangle + |v_2\rangle) \otimes |w\rangle = |v_1\rangle \otimes |w\rangle + |v_2\rangle \otimes |w\rangle$.
3. If we consider a vector $|v\rangle \in V$ and $(|w_1\rangle + |w_2\rangle)$, then $|v\rangle \otimes (|w_1\rangle + |w_2\rangle) = |v\rangle \otimes |w_1\rangle + |v\rangle \otimes |w_2\rangle$.

Let's consider $|v\rangle$ and $|w\rangle$ two column vectors in vector space V and W. These

two vectors have x_i and y_i as components where $i = 1, 2, \ldots, n$,

$$|v\rangle = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \in \mathbb{R}^n \text{ and } |w\rangle = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \in \mathbb{R}^n$$

When a tensor product is applied between these two vectors, a joint vector has been formed:

$$|v\rangle \otimes |w\rangle = \begin{pmatrix} x_1 \times \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \\ \vdots \\ x_n \times \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} \end{pmatrix} = \begin{pmatrix} x_1 y_1 \\ x_1 y_2 \\ \vdots \\ x_1 y_n \\ \vdots \\ x_n y_1 \\ x_n y_2 \\ \vdots \\ x_n y_n \end{pmatrix}$$

In general, if we have $E \in M_1$ and $F \in M_2$, where M_1 and M_2 are the spaces of $c \times d$ and $m \times n$ matrices respectively, then $E \otimes F$ is a $cm \times dn$ matrix.

$$E \otimes F = \begin{pmatrix} E_{11}F & E_{12}F & \ldots & E_{1d}F \\ E_{21}F & E_{22}F & \ldots & E_{2d}F \\ \vdots & \vdots & \vdots & \vdots \\ E_{c1}F & E_{c2}F & \ldots & E_{cd}F \end{pmatrix}$$

For example, by taking $c = d = 3$ then

$$E \otimes F = \begin{pmatrix} E_{11}F & E_{12}F & E_{13}F \\ E_{21}F & E_{22}F & E_{23}F \\ E_{31}F & E_{32}F & E_{33}F \end{pmatrix}$$

To simplify many calculations, the fact about tensor product is that

$$(E \otimes F)(G \otimes H) = EG \otimes FH.$$

1.1.4 Quantum states

An *observable* is a physical property of a system that can be measured in the lab. In quantum mechanics, an observable is described by a Hermitian operator and the eigenvalues of this operator represent the possible outcomes of measurement that correspond to the vectors in the orthonormal basis. The observable is said to be *degenerate* when eigenvectors correspond to one eigenvalue and when there is one eigenvector for only one eigenvalue then the observable is *non-degenerate*.

A projective measurement can be constructed from an observable denoted M, which is a Hermitian operator. The observable M has a decomposition:

$$M = \sum_m mP_m \tag{1.1}$$

where P_m is the projector on the eigenspace of the observable that has an eigenvalue m. By eigenspace we mean the set of eigenvectors that correspond to the same eigenvalue.

In quantum information theory, we are only interested in labelling these m eigenvalues, then we are going to use P_i where i are integer numbers that represent the possible outcomes. P_i must satisfy the following properties:

- $P_i^\dagger = P_i$,
- $P_i P_{i'} = \delta_{i,i'} P_i$
- $\sum_i^d P_i = I$,

The probability of getting a measurement outcome i is given by the following *Born Rule*.

The Born Rule

The Born Rule or the Probability Rule is the rule that gives the probabilities of getting possible outcomes of measurement.

Let's consider an operator O that has real eigenvalues λ_i that correspond to eigenvectors $|\lambda_i\rangle$, then the operator O can be defined by

$$O = \sum_i \lambda_i P_i$$

where $P_i = \sum_k |\lambda_k\rangle\langle\lambda_k|$ is the projector. If a measurement acts on a quantum system that is in a state $|\psi\rangle$, the probability of getting an outcome i is presented by:

$$prob(i) = \langle\psi|P_i|\psi\rangle \tag{1.2}$$

Instead of taking a single state $|\psi\rangle$, we consider an ensemble of state $|\psi_j\rangle$ with probabilities p_j, then the probability of getting an outcome i is given by:

$$prob(i) = \sum_j p_j\langle\psi_j|P_i|\psi_j\rangle \tag{1.3}$$

We can derive from equation (1.3) the following:

$$\begin{aligned}
prob(i) &= \sum_j p_j\langle\psi_j|P_i|\psi_j\rangle \\
&= \sum_j p_j \sum_k \langle\psi_j|\lambda_k\rangle\langle\lambda_k|\psi_j\rangle \\
&= \sum_j p_j \sum_k \langle\lambda_k|\psi_j\rangle\langle\psi_j|\lambda_k\rangle \\
&= \sum_k \langle\lambda_k| \sum_j p_j|\psi_j\rangle\langle\psi_j|\lambda_k\rangle \\
&= \sum_k \langle\lambda_k|\rho|\lambda_k\rangle \\
&= tr(\rho \sum_k |\lambda_k\rangle\langle\lambda_k|) \\
&= tr(\rho P_i)
\end{aligned}$$

where "tr" is the trace of an operator matrix. $\rho = \sum_j p_j|\psi_j\rangle\langle\psi_j|$ is the *density matrix* (or a *density operator*) which describes the quantum state and it has the following properties:

- Hermitian, it means that $\rho = \rho^\dagger$,
- a *positive semi-definite* which means that the eigenvalues are non-negative. It can be denoted as $\rho \geq 0$,
- has trace one.

Using the Born's rule, a quantum state is a mathematical structure that allows us to calculate probabilities of the outcomes of each possible measurement on a system.

States that can't be written as a mixture of states are called *pure quantum states* but others are called *mixed quantum states*.

If $\rho = \sum_{j}^{n} p_j |\psi_j\rangle\langle\psi_j| = |\psi\rangle\langle\psi|$, where $n = 1$ and $p_1 = 1$, then ρ represents a pure quantum state; but if it is not the case then ρ is called a mixed quantum state. This is the first way in which mixed states can arise and we will shortly discuss the second way, which is the *partial trace* of a pure state, that can be used to obtain a mixed state.

Pure quantum state is described by a single vector $|k\rangle$ or by a superposition of basis states $|\Psi\rangle = \sum_i c_i |k_i\rangle$ where c_i are the probabilities corresponding to $|k_i\rangle$ and $\sum_i |c_i|^2 = 1$.

$$\text{If } \mathrm{tr}(\rho^2) \begin{cases} = 1, \text{ then } \rho \text{ is a pure state} \\ \\ < 1, \text{ then } \rho \text{ is a mixed state.} \end{cases}$$

The Qubit

The classic binary bit has a quantum version which is called quantum bit or qubit. Qubit is in a two dimensional Hilbert space and it can be in states labelled 0, 1, or a superposition of these. The superposition stated that any two or more quantum states can be added together, we end up with another quantum state, for example, $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are two complex numbers and $|0\rangle$ and $|1\rangle$ are the *ket-0* and *ket-1* respectively. So, The qubit is represented by a column vector of two elements, where these elements should satisfy the normalisation condition, from the previous example, $|\alpha|^2 + |\beta|^2 = 1$. By using the Dirac notation, where the ket is $|0\rangle$, $|1\rangle$ and the bra is the conjugate transpose, we have: $|0\rangle := \begin{pmatrix} 1 \\ 0 \end{pmatrix}; |1\rangle := \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and these two are the orthonormal basis states. $|0\rangle, |1\rangle$ together is called computational basis.

For instance, the qubit state $|+\rangle$ is a superposition of the basis state which means that it can be described by a linear combination of $|0\rangle$ and $|1\rangle$:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}.$$

Multi-particle system

A *multiparticle system* can be defined by two axioms:

1. Tensor product is crucial in the understanding of multiparticle systems. If there are two Hilbert space H_A and H_B on two different particles A and B respectively, then a Hilbert space H_{AB} is described by a tensor product between these two spaces. H_{AB} has two different types of states:

   - a product state which is given by,

     $$|\phi_1\rangle \otimes |\phi_2\rangle$$

     where $|\phi_1\rangle$ and $|\phi_2\rangle$ are two independent states.
   - a joint state that has the possibility of being an entangled state. This specific state cannot be written as a product state, as we will shortly explain.

2. If a measurement will be applied on one particle, then in the joint system, the measurement is extended by the identity operator which is applied on the second particle.

For instance, if a measurement P_j is applied on A, then we need to apply a tensor product between ${P_j}_A$ and the identity operator I_B which is applied on the second particle B. The probability of getting the measurement outcome j is given by

$$Pr(j) = tr((P_{j\,A} \otimes I_B)\rho_{AB}) = tr(P_{j\,A} tr_B(\rho_{AB})) = tr(P_{j\,A}\rho_A) \tag{1.4}$$

where ρ_A is defined as

$$\rho_A := tr_B(\rho_{AB}) := \sum_k^{d_B} (I_A \otimes \langle k_B|)\rho_{AB}(I_A \otimes |k_B\rangle)$$

d_B is the dimension of H_B, $|k_B\rangle$ is any orthonormal basis on B and I_A is the identity operator in H_A. ρ_A is called *The reduced state* or the *reduced density matrix* of A which is computed by taking the *partial trace* over B, denoted by tr_B.

POVM

As mentioned above, the Born's rule states that the probability of the outcome for measurement, is represented by

$$prob(i) = \sum_j p_j \langle \psi_j | P_i | \psi_j \rangle = tr(\rho P_i)$$

where $\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j|$ and $P_i = \sum_k |\lambda_k\rangle\langle\lambda_k|$. The projective measurements P_i have the axioms that we mentioned in the beginning of this part:

- $P_i^\dagger = P_i$,
- $\sum_i P_i = I$ where I is the identity operator because the sum of all probabilities must be equal to 1.
- $P_i P_{i'} = \delta_{i,i'} P_i$. This property together with the above two axioms, imply that the operator P_i is positive and therefore the probability will be non-negative.

If we add ancilla particle which is used in the measurement, to the probability equation, these axioms can be modified. Let's try to interact the quantum system with an external one, so we are going to add an ancilla $|0\rangle\langle 0|$. Then, we have

$$\text{Prob(i)} = tr[(\rho \otimes |0\rangle\langle 0|)P_i] \equiv tr(\rho M) \tag{1.5}$$

where $P_i = |\lambda_k\rangle\langle\lambda_k|$ and $M = \sum_{a,b} \lambda_{b0}\lambda^*_{a0}|b0\rangle\langle a0|$.

Proof. By taking $|\lambda_k\rangle = \sum_{k,j} \lambda_{kj}|kj\rangle$, then the probability of getting i is

$$\begin{aligned}
\text{Prob(i)} &= tr[(\rho \otimes |0\rangle\langle 0|)P_i] \\
&= tr[(\rho \otimes |0\rangle\langle 0|)|\lambda_k\rangle\langle\lambda_k|] \\
&= tr\left[\sum_{a,b,k,j,m,l} (\rho_{ab} \otimes |a0\rangle\langle b0|)\lambda_{kj}\lambda^*_{ml}|kj\rangle\langle ml|\right] \\
&= tr\left[\sum_{a,b} (\rho_{ab} \otimes |a0\rangle\langle b0|)\lambda_{b0}\lambda^*_{a0}|b0\rangle\langle a0|\right] \text{ by taking } k = b, m = a, j = l = 0; \\
&= tr\left[\sum_{a,b} \lambda_{b0}\lambda^*_{a0}|b0\rangle\langle a0|\rho_{ab}\right] \\
&= tr(\rho M); \text{where } M = \sum_{a,b} \lambda_{b0}\lambda^*_{a0}|b0\rangle\langle a0|
\end{aligned}$$

M is a *Positive Operator Valued Measure* element because when a projective measurement P_i is applied on a particle and an ancilla, we end up with a statistic defined by *POVM* where only the third axiom has been modified, which means that the orthogonality property is removed and the positivity is retained. □

Therefore Positive Operator Valued Measure or *Generalized measurement* is described by the set of positive operators P_i that sum to the identity matrix: $\sum_{i=1}^{n} P_i = I$.

Bloch sphere

Let's consider a system of a single qubit that has a 2×2 density matrix ρ. This matrix is Hermitian, has trace one and can be expanded in the basis $I, \sigma_1, \sigma_2, \sigma_3$ where I is the identity matrix, $\sigma_1 = X$, $\sigma_2 = Y$ and $\sigma_3 = Z$ are the Pauli matrices (as will be defined on page 19). The coefficient of I in the expansion of the ρ is $\frac{1}{2}$ to end up with $tr(\rho) = 1$. The expansion is given by:

$$\rho = \frac{1}{2}(I + xX + yY + zZ) = \frac{1}{2}\begin{pmatrix} 1+z & x-iy \\ x+iy & 1-z \end{pmatrix}$$

The coefficients x, y and z are found using the Hilbert Schmidt inner product which is presented by $tr(A^\dagger B)$. Hence, they are computed as follow

$$\frac{1}{2} tr(\rho \sigma_i).$$

Then the determinant of the ρ must be computed to find the eigenvalues λ_i. Hence, the eigenvalues are given by

$$\lambda_i = \frac{1}{2} \pm \frac{\sqrt{x^2 + y^2 + z^2}}{2}.$$

The non-negativity of the eigenvalues will be holding if $x^2 + y^2 + z^2 \leq 1$. Thus a quantum state can be represented by a vector (x, y, z), where these values are the coefficients of the above decomposition. These coefficients correspond to a point on the so-called a *Bloch sphere* which is considered as a unit 2-sphere and the standard basis vectors $|0\rangle$ and $|1\rangle$ are represented by the north and south poles of the Bloch sphere respectively [50].

The pure quantum states of the system are the points on the surface of the sphere where the norm of the Bloch vector must be equal to 1, while the internal points correspond to the mixed states where the norm of the vector must be less than 1.

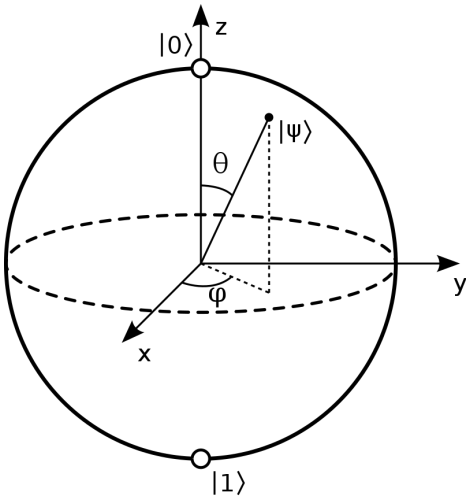


Figure 1.1: This image represents the Bloch sphere. It is taken from "Wikipedia", `https://commons.wikimedia.org/wiki/File:Bloch_sphere.svg`, February 2022.

Entangled and separable quantum states

Let's consider two quantum systems A and B in two different Hilbert spaces H_A and H_B. If the pure state $|\chi_{AB}\rangle \in H_A \otimes H_B$ can be written in the form $|\chi_{AB}\rangle = |\chi_A\rangle \otimes |\chi_B\rangle$ where $|\chi_j\rangle$ is a pure state of the j^{th} subsystem, then the state is separable, but if it can not be written in the showed form then the state is entangled. For instance, let's take a state

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle).$$

This state cannot be written as a product state because the state should have the following form

$$(a|0\rangle + b|1\rangle) \otimes (A|0\rangle + B|1\rangle)$$

where aA and bB should be equal to $\frac{1}{\sqrt{2}}$, aB and bA should be equal to 0,which means that two of these coefficients must be equal to 0 but this cannot be done, for instance, by taking a product between $aB = 0$ and $bA = 0$ and a second product between $aA = \frac{1}{\sqrt{2}}$ and $bB = \frac{1}{\sqrt{2}}$, then we have:

$$\left.\begin{array}{l} aB = 0 \\ \\ bA = 0 \end{array}\right\} abAB = 0 \; ; \qquad \left.\begin{array}{l} aA = \frac{1}{\sqrt{2}} \\ \\ bB = \frac{1}{\sqrt{2}} \end{array}\right\} abAB = \frac{1}{2}$$

we find that these products are inconsistent. Then

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \neq (a|0\rangle + b|1\rangle) \otimes (A|0\rangle + B|1\rangle)$$

Therefore, $|\psi\rangle$ is an entangled quantum state.

Let's consider the density matrix ρ, which corresponds to a mixed state, is separable if there is p_k is greater or equal to 0, ρ_1^k and ρ_2^k are mixed states of the subsystems:

$$\rho = \sum_k p_k \rho_1^k \otimes \rho_2^k$$

where

$$\sum_k p_k = 1,$$

The state is called *simply separable* or *product state* when the state can be expressed as $\rho = \rho_1 \otimes \rho_2$. If the two conditions, $p_k \geq 0$ and $\rho = \sum_k p_k \rho_1^k \otimes \rho_2^k$, are not attained then ρ is entangled mixed state. In the next chapter, we will define and use the notion of generalised separability.

Local hidden variable model

The original motivation for investigating separability [63] was to understand *local hidden variables* [26, 8]. Let's consider a source that creates particles λ. These particles are sent to two distant detectors D_1 and D_2 controlled by two people. By assuming that each person can choose his measurement, M_1 and M_2, the two detectors generate two outcomes A_1 and A_2 respectively. Hence, λ holds local hidden variables that have a probability distribution p_λ.

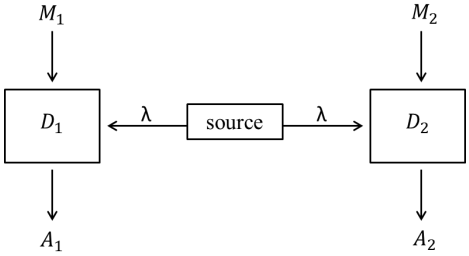


Figure 1.2: Local hidden variables

Local hidden variable models can be presented by the following equation,

$$\sum_\lambda p_\lambda p^\lambda(A_1|M_1) p^\lambda(A_2|M_2)$$

where $p^\lambda(A_i|M_i)$ is the probability of getting an outcome A_i given a measurement M_i. Bell [15] showed that local hidden variables are inconsistent with the statistical predictions of quantum mechanics. Experiments have been performed to demonstrate

non-locality [24, 21].

Positive partial transpose

In general, it is difficult to test whether a state ρ_{AB} is separable or not [25], in our case the so-called *PPT test* will be sufficient for this research. The *positive partial transpose test* or PPT test is one of the useful tests that works to find whether the state is separable [47]. For instance, if we have

$$X_{AB} = \sum_{i,j,k,l} c_{i,j,k,l} |i\rangle\langle j|_A \otimes |k\rangle\langle l|_B$$

then the *partial transpose* of X_{AB} is

$$X_{AB}^{T_A} = \sum_{i,j,k,l} c_{i,j,k,l} |i\rangle\langle j|_A^T \otimes |k\rangle\langle l|_B = \sum_{i,j,k,l} c_{i,j,k,l} |j\rangle\langle i|_A \otimes |k\rangle\langle l|_B$$

If we consider ρ is a separable state

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|_A \otimes |\phi_i\rangle\langle\phi_i|_B,$$

then

$$\rho^{T_A} = \sum_i p_i |\psi_i^*\rangle\langle\psi_i^*|_A \otimes |\phi_i\rangle\langle\phi_i|_B$$

where $|\psi_i^*\rangle$ are pure states and their coefficients, that are in the computational basis, are the complex conjugates of $|\psi_i\rangle$. After finding the partial transpose, if ρ^{T_A} is positive semi-definite then ρ is PPT. Therefore, ρ is separable which implies that $\rho \in$ PPT. If $\rho \notin PPT$ then ρ is an entangled quantum state but if $\rho \in PPT$ then ρ can be separable. However, 2×2 and 2×3 systems is known to be equivalent to PPT separability [34]. But in the case of 3×3 and 2×4 systems, the eigenvalues of the partial transpose of entangled mixed states are found to be positive [35].

1.1.5 Controlled gates

The manipulation of qubits can be done with the help of quantum logic gates. So quantum logic gates are basic quantum circuits that operate on a small number of qubits. Quantum logic gates are represented by unitary operators,

$$U^\dagger U = U U^\dagger = I$$

where $U^\dagger$ is the conjugate transpose matrix of U.

A gate acting on n qubits is represented by a $2^n \times 2^n$ unitary matrix. There exists an uncountable infinite number of gates but the most used gates are the *Identity* and the *Pauli gates*. The Identity gate is the identity matrix, and it is defined for a single qubit:

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

The Pauli gates are the Pauli matrices and act on a single qubit. The Pauli "X", "Y" and "Z" correspond to rotations around the x, y and z axes of the Bloch sphere.

1. The Pauli-X gate is sometimes called the bit flip as it maps $|0\rangle$ to $|1\rangle$ and $|1\rangle$ to $|0\rangle$.
$$X = \sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

2. The Pauli-Y gate maps $|0\rangle$ to $i|1\rangle$ and $|1\rangle$ to $-i|0\rangle$.
$$Y = \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

3. The Pauli-Z gate leaves the basis state $|0\rangle$ unchanged and maps $|1\rangle$ to $-|1\rangle$.
$$Z = \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

The square of a Pauli matrix is the identity matrix,

$$I^2 = X^2 = Y^2 = Z^2 = -iXYZ = I$$

Also, they anti-commute, for example, $ZX = iY = -XZ$.

There is also the Hadamard gate that maps the basis state $|0\rangle$ to $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$ and $|1\rangle$ to $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$,

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Controlled gates are the gates that act on two or more qubits, where one or more qubits act as a control. In general, let's consider U is a gate that acts on one single

qubit. The matrix of this gate is represented by

$$U = \begin{pmatrix} U_{00} & U_{01} \\ U_{10} & U_{11} \end{pmatrix}. \tag{1.6}$$

Then the controlled gate operates on two qubits where the first qubit acts as a control. So the matrix that represents CU is

$$CU = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & U_{00} & U_{01} \\ 0 & 0 & U_{10} & U_{11} \end{pmatrix}. \tag{1.7}$$

For example, the controlled-NOT gate or CX acts on two qubits and performs the X on the second qubit only when the first qubit is $|1\rangle$ and when the first qubit is $|0\rangle$ leaves it unchanged, which means

$$|00\rangle \rightarrow |00\rangle$$

$$|01\rangle \rightarrow |01\rangle$$

$$|10\rangle \rightarrow |11\rangle$$

$$|11\rangle \rightarrow |10\rangle$$

It is represented by the following matrix:

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

The Controlled-Z gate or CZ acts on two qubits and performs the Z operation on the second qubit only when the first qubit is $|1\rangle$ and when the first qubit is $|0\rangle$ leaves it unchanged, which means

$$|00\rangle \rightarrow |00\rangle$$

$$|01\rangle \rightarrow |01\rangle$$

$$|10\rangle \rightarrow |10\rangle$$

$$|11\rangle \to -|11\rangle$$

The matrix is given by:

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

1.1.6 Affine and convex sets

In this section, we are going to define the convex hull that we are going to use in following chapters.

Let's take two different points x_1 and x_2 in $\mathbb{R}^n$ and these points are of the form

$$y = \lambda x_1 + (1 - \lambda)x_2$$

where $\lambda \in \mathbb{R}$. The previous equation forms the *line* passing through x_1 and x_2. If the parameter $\lambda = 0$ then the equation $y = x_2$, and if $\lambda = 1$, then $y = x_1$. So the values of λ between 0 and 1 represent the *closed line segment* between the two points x_1 and x_2.

Now, let's consider a set $C \subseteq \mathbb{R}^n$. C is *affine* if the line between any two different points in C lies in C, which means that if for any $x_1, x_2 \in C$ and $\lambda \in \mathbb{R}$, there is $\lambda x_1 + (1-\lambda)x_2 \in C$. C has the linear combination of any two points in C, and all the coefficients in the linear combination sum to one. In general, by taking more than two points in the following form

$$\lambda_1 x_1 + \lambda_2 x_2 + \ldots + \lambda_k x_k,$$

where

$$\lambda_1 + \lambda_2 + \ldots + \lambda_k = 1,$$

the equation is defined as an *affine combination* of the points $x_1, x_2, \ldots, x_k$.

Moreover, if C is an affine set, $x_1, x_2, \ldots, x_k \in C$ and $\lambda_1 + \lambda_2 + \ldots + \lambda_k = 1$, then the point

$$\lambda_1 x_1 + \lambda_2 x_2 + \ldots + \lambda_k x_k \in C.$$

The affine hull (aff) is the set of all affine combinations of points in the set C.

$$affC = \{\lambda_1 x_1 + \lambda 2 x_2 + \ldots + \lambda_k x_k | x_1, x_2, \ldots, x_k \in C, \lambda_1 + \lambda_2 + \ldots + \lambda_k = 1\}$$

By taking a line segment between two points in C, C is *convex set* if this line segment lies in C, it means that for any $x_1, x_2 \in C$, any λ between 0 and 1, it hold that

$$\lambda x_1 + (1 - \lambda) x_2 \in C$$

Every affine set is a convex set because the set contains the whole line between two different points in the set.

A convex combination of the points $x_1, x_2, \ldots, x_k$ has a form

$$\lambda_1 x_1 + \lambda_2 x_2 + \ldots + \lambda_k x_k$$

where $\lambda_1 + \lambda_2 + \ldots + \lambda_k = 1$ and $\lambda_i \geq 0$ for $i = 1, 2, \ldots, k$.

The set of all convex combinations of points in a set C is called the *convex hull (conv)* of C,

$$convC = \{\lambda_1 x_1 + \lambda_2 x_2 + \ldots + \lambda_k x_k | x_i \in C, \lambda_i \geq 0, i = 1, 2, \ldots, k, \lambda_1 + \lambda_2 + \ldots + \lambda_k = 1\}$$

.

1.1.7 Cluster States

In quantum information and quantum conputation, a *cluster state* is not a single quantum state, it is a family of quantum state that can support quantum computation. It can be explained as follows: any graph that contains a number of vertices, n, we can set n-qubit cluster state, it means that each vertex corresponds to a qubit in the state $\frac{|0\rangle + |1\rangle}{\sqrt{2}}$. Each edge that represents the CZ gate, connect two vertices or two qubits. For two dimensions, the cluster state is considered as rectangular lattice and for n dimensions, it corresponds to a graph with n-dimensional lattice. Initially, the "cluster state" [53, 54, 45] is a type of quantum computation and it is made of one qubit measurements.

Cluster states can be defined as graph states which is a particular sort of multi-qubit state that can be shown as a graph.

In this thesis, we are going to consider what happens if we change the inputs and

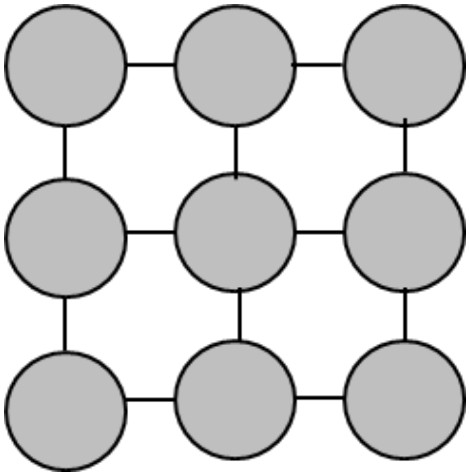


Figure 1.3: 9-qubit cluster state

the gates of the cluster state quantum computation and we will show that there are regions that can be efficiently simulated classically.

1.2 Classical simulation

In general, it is not believed that quantum system can be efficiently simulated classically [57]. But for several years great effort has been devoted to the study of efficient simulation on a classical computer. These studies have offered rich contributions to the field of research. In this section, we will review a non-exhaustive selection of quantum systems that have been shown to be classical simulatable.

1. **Stabilizer circuit**

   A *stabilizer formalism* can be described by an example. Let's consider a state of two qubits,

   $$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

   By applying two operators $X \otimes X$ and $Z \otimes Z$ on $|\psi\rangle$, we find that the state remains the same,

   $$(X \otimes X)|\psi\rangle = |\psi\rangle \text{ and } (Z \otimes Z)|\psi\rangle = |\psi\rangle$$

   then the state $|\psi\rangle$ is stabilized by these two operators [46]. The stabilizer formalism has an important concept which states that a quantum state can be presented by using Pauli operators that stabilize it. By applying unitary gates on the quantum state, it means that these gates act on the string of the Pauli operators. But if these unitary gates are made of $CNOT$, Hadamard and phase gates, these Pauli operators are transformed into new set of stabilizer Pauli operators. Then a circuit that consists only of these gates is called a *stabilizer circuit* [1].

   Gottesman-Knill theorem shows that a quantum computation, that consists of state preparations in the computational basis, gates in the so-called *Clifford group* [27], Pauli gates and measurements in the Pauli basis, may be efficiently simulated on a classical computer.

   This theorem has been extended in a variety of ways [64, 13, 14, 40]. It also has been extended to quantum circuits that are composed of Clifford and a single type of non-Clifford gates [19]. For instance, in [19], the classical algorithm that simulate the system is polynomial in the number of qubits and the number of Clifford gates but it is exponential in the limited number of the non-Clifford gates.

2. **Matrix product states and tensor networks**

The product state $|\psi\rangle_{1\ldots n}$ can be written in the following form,

$$|\psi\rangle_{1\ldots n} = |\alpha\rangle_1|\beta\rangle_2 \ldots |\kappa\rangle_n \tag{1.8}$$

where this state consists of n qubits. A matrix product state or *MPS* is a one dimension quantum state of n qubits and it is a generalisation of the above equation. Each state in equation (1.8) is replaced by a matrix of state; for instance, the state $|\alpha\rangle$ is changed into $|\alpha_{ij}\rangle$. Therefore, the matrix product of n matrices is given by the following form:

$$|\psi\rangle = \sum_{i,j,k,\ldots,m} |\alpha_{ij}\rangle|\beta_{jk}\rangle \ldots |\kappa_{mi}\rangle \tag{1.9}$$

If the matrix of states, $|\alpha_{ij}\rangle, \ldots, |\kappa_{mi}\rangle$, are written as $A, B \ldots, K$, then the state $|\psi\rangle = tr(AB \ldots K)$. In the standard basis, states can be written as,

$$|\alpha_{ij}\rangle = \sum_{i_1} A_{ij}^{(i_1)}|i_1\rangle, \ldots, |\kappa_{mi}\rangle = \sum_{i_n} K_{mi}^{(i_n)}|i_n\rangle,$$

then the matrix product state $|\psi\rangle$ will be given by

$$|\psi\rangle = \sum_{i,\ldots m,i_1\ldots i_n} A_{ij}^{(i_1)} B_{jk}^{(i_2)} \ldots K_{mi}^{(i_n)}|i_1\rangle|i_2\rangle \ldots |i_n\rangle. \tag{1.10}$$

As mentioned above, the MPS is a one dimension state [48] but there is another description of MPS, which is called *Projected entangled pairs state* or *PEPS*. $PEPS$ extends the MPS in two and higher dimensions.

Using MPS, Jozsa [37] considered any poly sized qauntum circuit of n qubits and has shown that this circuit can be efficiently simulated classically only if it is logarithmic in the number of qubit gates. Using the matrix product state representation, a classical methodology is presented to simulate cluster state quantum computation that has a polynomial number of qubits and an exponential number in the width of the cluster [66]; then it is shown that any logarithmic depth quantum computation with gates that relates only near qubits can be efficiently simulated classically.

The *tensor network* which is a generalisation of matrix product states, can be described by a graphical notation [16]. The matrix product state $|\psi\rangle$ which is presented by equation (1.10) can be illustrated by the following graph:

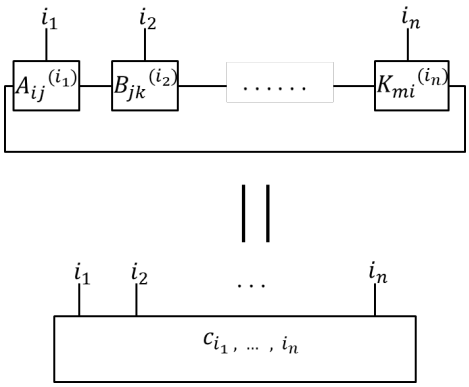


Figure 1.4: This image presents the tensor network. The *MPS* can also be written as: $\sum_{i_1,\ldots,i_n} c_{i_1,\ldots,i_n}|i_1,\ldots,i_n\rangle$.

Markov and Shi [41] proved that a quantum circuit with a non-Clifford gates whose graph is close to a tree graph as measured by the treewidth, can be simulated in an exponential time. In addition, any quantum circuit can be efficiently simulated using the tree tensor network contraction can only produce a limited amount of entanglement[65].

3. **Matchgates**

A *matchgate* can be defined as a gate G of two qubits that takes the following form in the computational basis,

$$G = \begin{pmatrix} m & 0 & 0 & n \\ 0 & r & s & 0 \\ 0 & t & w & 0 \\ p & 0 & 0 & q \end{pmatrix};$$

where

$$A = \begin{pmatrix} m & n \\ p & q \end{pmatrix} \text{ and } B = \begin{pmatrix} r & s \\ t & w \end{pmatrix}$$

are the inner and the outer block of G respectively and they are in the special unitary group or $SU(2)$ which is a 2×2 unitary matrix with a determinant equal to 1. Valiant [60, 59] defines the notion of matchgate and demonstrates that matchgate circuits can be classically simulated in polynomial time under specific conditions. Then Terhal and Divincenzo [58] analyze the class of quantum computations demonstrated by Valiant [60] and show that this class is related to a fermions model in one dimension. Matchgate circuits in which these gates are only applied to neighbour qubit lines can also be efficiently simulated classically [39]. Instead of taking a circuit that has product input and the output consists

of a single qubit measurement [39], Brod [20] displays that even if the input is product states and the output has measurements of many qubits, matchgates can be simulated on a classical computer.

1.3 Quantum supremacy

The term *quantum supremacy* describes the goal of building controllable quantum devices with well understood dynamics that cannot be efficiently simulated on classical computers [32, 17]. The Martinis-Google group claimed a demonstration of quantum supremacy by taking a random quantum circuit and seeing whether the outcome of measurements can be sampled on a classical computer [6]. However, there have been number of attempts to replicate their work on classical computers [9, 2], and so the debate is far from being conclusive.

1.4 Preview of the thesis results

In this thesis, we will begin by changing the inputs of cluster state quantum computation to end up with some inputs that allow the system to be efficiently simulated classically. If different pure state choices for $|\psi\rangle = |+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$ are made, two facts are well known already:

1. If $|\psi\rangle = |0\rangle$ or $|\psi\rangle = |1\rangle$ then the system is classically simulatable efficiently.
2. If $|\psi\rangle$ is a state from the XY plane of the Bloch sphere then the system has the power of quantum computation.

We explore questions related to this in the various chapters of the thesis:

- In chapter 2, we will show that if $|\psi\rangle$ has the (unormalised) form $|0\rangle + \epsilon|1\rangle$ where ϵ is a small non-zero number, the system can be efficiently simulated classically. We are going to use new state spaces that involve cylinders rather than Bloch sphere and these cylinders are parameterized by radius r. We must grow the radius in order to get separable decomposition. However, the radius cannot grow too much. Then instead of CZ gates, we will try to apply diagonal gates $V(\theta) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z_\theta$ on two cylindrical state spaces and we will find that almost all inputs can be simulated classically when θ is small. In addition, it turns out that CZ gate requires the most growth factor λ, which is

the ratio of the output radius to the input radius, to maintain separability. My contribution was the computation of the growth factors for the $V(\theta)$ gates.

- In chapter 3, by taking some examples of alternative local state spaces and by growing the size of these specific states, we will find that the setup can also be efficiently simulated on a classical computer. Then, we tried optimize our classical simulation over these state spaces that we tried and it turns out that the cylinder requires the least growth. My contribution was to perform all these calculations using linear programming.

- In chapter 4, we attempt to improve the radius of a system that can be efficiently simulated classically. By using the coarse graining, we will find that the value of the radius of four particles in a block is greater than the radius of two particles. We will discover that by adding more vertices in a block, the value of the radius, that can be efficiently simulated classically, increases slightly [7]. My contribution was to calculate the radius threshold for four particles, assuming a conjecture about which measurements determine when certain invalid negative probabilities arise.

Chapter 2

Efficient Classical Simulation of quantum circuits with alternative inputs

A typical quantum computer is a sequence of unitary quantum gates acting on qubits. Quantum gates are capable of generating entanglement between these qubits. But real quantum gates suffer from noise which can destroy the creation of entanglement.

This means that very noisy systems cannot be used for better than classical computation, because it is known that gate model quantum computers with gates that don't generate entanglement can be efficiently simulated classically [33] (this is a part of variety of works that investigate when classical simulation is possible if there is a limited number of entanglement [38, 62]). Here, by efficiently classical simulation, we mean sampling in polynomial time from a probability distribution that approximates the probability distribution of measurement outcomes to arbitrary accuracy.

When we say that a gate doesn't generate entanglement, we mean that a quantum gate $\mathcal{E}$ takes a product input, $\rho \otimes \sigma$, to a separable output state. We then say that $\mathcal{E}$ is a *separable gate*. So, it preserves the separability.

One of the motivations of the above classical algorithm of [33] was to obtain upper bounds on the value of the fault-tolerance threshold, which is the noise level below which quantum computation can be achieved, because once the noise level is high enough to make the gate separable the device can be efficiently simulated classically. If we assume that quantum computers cannot be efficiently simulated classically, the fault-tolerance threshold must be lower than this.

This algorithm cannot be used to simulate pure entangled quantum systems. However, we will see later in this chapter that by modifying the notion of separability, the algorithm in Harrow and Nielsen [33] can be used to simulate some entangled quantum systems. Specifically, we will consider particular entangled quantum systems that are variant of cluster state quantum computation. As discussed in chapter 1, in cluster state, we initialize in the state $|\psi\rangle = |+\rangle = \frac{|0\rangle+\epsilon|1\rangle}{\sqrt{2}}$; but we are going

to show that if we initialize in the state $|\psi\rangle = |0\rangle + \epsilon|1\rangle$ where ϵ is a small enough non-zero number, instead of the $|+\rangle$ state, the system can be efficiently simulated on a classical computer. It means that by providing a small enough ϵ, it turns out that we can give a classical simulation algorithm that uses cylindrical separability rather than normal separability. Then we will replace the CZ gates by other diagonal gates $V(\theta) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z_\theta$ where $Z_\theta = |0\rangle\langle 0| + e^{i\theta}|1\rangle\langle 1|$ and we will find similar results that hold for these diagonal gates.

First, we are going to describe the algorithm of Harrow and Nielsen [33] which we use in the research. Then we are going to modify the pure states of the cluster state quantum computation to end up with a system that can be efficiently simulated classically using the new notion of generalised separability. Finally, we will apply diagonal gates instead of CZ gates on the two inputs and it turns out that almost all inputs can be simulated classically.

2.1 Description of the algorithm in Harrow and Nielsen[33]

The algorithm of [33] starts with an assumption that the input is a product state,$\rho_A \otimes \rho_B \otimes \rho_C \ldots$, acted upon by separable quantum gates, $\mathcal{E}_1, \ldots, \mathcal{E}_{p(n)}$ where the polynomial $p(n)$ is the number of gates and n is the measure of problem size. For instance, if the first gate,$\mathcal{E}_1$, which is assumed to be separable, acts on two qubits A and B,

$$\mathcal{E}_1 \left(\rho_A \otimes \rho_B\right) = \sum_k p_k \rho_A^k \otimes \rho_B^k$$

The input is taken to be a separable output state. Then the algorithm samples the probability distribution, p_k, gets one outcome and stores the product state which corresponds to the obtained outcome, $\rho_A^k \otimes \rho_B^k$. This product state is considered as an input for the next separable gate, for instance, $\mathcal{E}_2$. These gates may act not only on the stored product state but also on different qubits. This process is repeated until it reaches the final outcome which is produced from the final separable gate $\mathcal{E}_{p(n)}$.

At the end of the process, qubits are measured at the output of the quantum computation, where the output is a string of product states. Using the trace, the probability distribution of each one of the products is computed. Because these computations involve a linear number of 2×2 matrices, they can be computed classically. The algorithm simulates the final measurement when it samples from those

probability distributions and produces a classical outcome for measurements of each qubit. It ends up with an equivalence between the classical simulation and certain measurement on quantum states.

Now let's describe in detail how the classical simulation is performed in the paper of [33]. The initial state of the computer is assumed to be in a computational basis state. The algorithm uses variables in the classical simulation, that are three dimensional real vectors $\overrightarrow{s}_j$ for each $j = 1, \ldots, q(n)$ where the polynomial $q(n)$ is the number of qubits. These vectors represent the bloch vectors of qubits in the quantum computers which the algorithm simulates. Each $\overrightarrow{s}_j$ is valid which means that it satisfies many properties. Two of these properties are: each component of the vector is in [-1, 1] and the norm of the vector is less than or equal to 1. $\rho(\overrightarrow{s})$ is defined to be the density matrix of $q(n)$ qubits when $\overrightarrow{s}$ is valid.

W.l.o.g. a single qubit gate can be considered as a two qubit product gate. Then the algorithm assumes that $\mathcal{E}_1, \ldots, \mathcal{E}_{p(n)}$ are two-qubit separable gates. It starts to simulate the first separate gate $\mathcal{E}_1$ that acts on two qubits A and B, using the input $\overrightarrow{s}$ in the gate simulation procedure.

The gate simulation procedure is described as followed. The input was represented by the three dimensional vector $\overrightarrow{s}$ as mentioned above. The body of the procedure consisted of finding valid vectors, $\overrightarrow{s_A}^j$ and $\overrightarrow{s_B}^j$ on the output. The equation can be written as:

$$\mathcal{E}_1 \left(\rho(\overrightarrow{s_A}) \otimes \rho(\overrightarrow{s_B})\right) = \sum_k p_k \rho(\overrightarrow{s_A}^j) \otimes \rho(\overrightarrow{s_B}^j)$$

.

These separable decompositions are computed to a certain accuracy, which is introduced to be less than or equal to 2^{-l} where l is the number of bits of precision. To find the probabilities and the vectors, it requires operations that are poly(2^l) which turns out to be sufficient (as explained in [33] in section 3.B) and l is picked to be logarithmic in the circuit; because if the circuit is large, more gates, that introduce more errors, are required; so l needs to be large.

By *Caratheodory's theorem*, there are at most 16 terms in the sum; we mean that if a point of $\mathbb{R}^d$ is in a convex hull of a set, this point can be written as the convex combination of at most $d + 1$ in the set. Then the algorithm samples the probability distribution,p_k, and stores the obtained product state, $\rho(\overrightarrow{s_A}^j) \otimes \rho(\overrightarrow{s_B}^j)$, and it is used as an input for the following gate. This process is repeated many times until the gate $\mathcal{E}_{p(n)}$ is reached and a final output is produced. At the end, this procedure produced a valid vector as an output. And this vector is the set of all three dimensional real vectors of n qubits.

Finally, the final measurement is simulated in the computational basis. The algorithm measures all qubits at the output of the quantum computation. The probability distribution is computed of each one of the products. The algorithm samples from those probability distributions and produces a classical outcome for measurements of each qubit.

As a conclusion, [33] prove that a quantum device built from noisy quantum gates, that don't generate entanglement, can be efficiently simulated classically.

2.2 Cylinder separability

Explicitly, as we mentioned in the introduction, we will be studying a variant of cluster state quantum computation and we will show that it could be efficiently simulated on a classical computer. The algorithm of [33] cannot be used because these cluster state circuits lead to pure entangled quantum states; and these quantum states cannot be approximated by a separable decomposition. But we develop a new method; we are going to use a different notion of separability to show that these states can be efficiently simulated classically.

Separable states are quantum states that can be decomposed into individual states which belong to separate particles [63]. In the multipartite case, a pure state $|\phi\rangle$ is separable if it has the form: $|\phi\rangle = |\phi_1\rangle \otimes \ldots \otimes |\phi_n\rangle$.

And a mixed state ρ is separable if it has the following form

$$\rho = \sum_i p_i \rho_A^i \otimes \rho_B^i \otimes \rho_C^i \otimes \ldots \tag{2.1}$$

where ρ_A^i, ρ_B^i, $\rho_C^i, \ldots$, are positive density matrices. In normal quantum state, these density matrices contain bloch vectors that come from the bloch sphere.

In generalised separability [52, 51],

$$\rho = \sum_j p_j \sigma_A^j \otimes \sigma_B^j \otimes \sigma_C^j \otimes \ldots$$

where σ_A^j, σ_B^j, $\sigma_C^j, \ldots$, are positive density matrices that are drawn from a set Q of quantum states. We relax this condition by allowing this set to be a new state space. For instance, instead of the Bloch sphere, we will consider Bloch vectors drawn from a cylindrical state space with a specific radius $r \leq 1$ and a height $h = 1$. This is the new notion of generalised separability, the so-called "*cylinder separability*", which is

going to lead to efficient classical simulation. Generalised separability has also been previously used to construct efficient simulation on a classical computer of certain type of $PEPS$ [3] where these states are in general, hard to simulate[55].

As we will see later in this chapter, by taking two of these cylindrical state spaces acting with the controlled-Z gate, we will end up with a separable output state with respect to two cylindrical state spaces with larger radii "R". Then the system could be efficiently simulated on a classical computer only when the output radii are large enough for the input radii, which means that the output radii should be less than or equal to 1 because of the negativity that always appears [5, 4].

Note that, even though we have systems that are close to product state, it does not mean that the setup can be efficiently simulated classically [61, 29, 28]. It has also been shown that variant of the systems that we are considering can enable quantum computation under the assumption that repeated measurements on any given qubit are allowed [43, 12], however no previous work has shown the efficient classical simulation results that we present. In addition, our framework does not fit into *Generalized probabilistic theories* or *GPTs* because if CZ gates are applied many times, the radius will grow and we will end up with negative probabilities. By GPT, we mean that it is a framework for generalizing quantum theory to explore the cause of its unusual features [49, 36, 31, 10, 11].

2.3 Cylinders and the restricted measurements

For a specific set of measurement operators, denoted by $\{M\}$, the *dual* is a set of operators ρ, that give positive numbers under the *Born rule*, $tr(M\rho)$. For instance, motivated by the cluster state quantum computation schemes, if we measure in the $X-Y$ plane and Z directions, the dual is a cylinder. To demonstrate how a cylinder is obtained, we start by measuring in the Z direction, we need to compute the trace, that should be positive,

$$tr\left(\frac{I \pm Z}{2}\rho\right) = tr\left\{\left(\frac{I \pm Z}{2}\right)\left(\frac{I + xX + yY + zZ}{2}\right)\right\} = \frac{1 \pm z}{2},$$

By adding a constraint that the operators are unit trace, we end up with two parallel horizontal planes that should be orthogonal to the Z axis where Z is in the range $[-1, 1]$, and all the operators of the bloch space between these planes give

positive probabilities. Then we add the $X - Y$ measurement to the Z measurement,

$$tr\left(\frac{I \pm X}{2}\rho\right) = tr\left\{\left(\frac{I \pm X}{2}\right)\left(\frac{I + xX + yY + zZ}{2}\right)\right\} = \frac{1 \pm x}{2},$$

and

$$tr\left(\frac{I \pm Y}{2}\rho\right) = tr\left\{\left(\frac{I \pm Y}{2}\right)\left(\frac{I + xX + yY + zZ}{2}\right)\right\} = \frac{1 \pm y}{2},$$

X and Y are in the range $[-1, 1]$. Therefore, the intersection of all these planes makes a cylindrical state space. We are using this specific non-physical input state space because we are only considering the restricted measurements, Z and XY measurements. So, we are interested in quantum and non-quantum operators that give valid and positive probabilities for these measurements under the born rule.

Thus, the same process of [33] algorithm is applied with this new type of state space, that allows us to express entangled pure states as separable states. Instead of the bloch sphere, we can use the cylinder because of the measurement restrictions, and those cylinder states still give us a valid probability distribution.

2.4 Bloch vectors of the cylinder

In this section, we discuss two ways in which we characterize the cylinder.

The Bloch vectors of the cylinder can be characterized by

$$\big\{(x, y, z) | z \in [-1, 1], x^2 + y^2 \leq 1\big\} \tag{2.2}$$

Also, the cylinder can be defined as:

$$Cyl(r) := \big\{\rho | \rho = \rho^\dagger, tr\rho = 1, ||\rho - D_Z(\rho)|| \leq r\big\} \tag{2.3}$$

where $D_Z(\rho) := \frac{1}{2}(\rho + Z\rho Z^\dagger)$ is the dephasing of ρ.

Equation (2.3) is a description of a cylinder with radius r; it doesn't consist of bloch vectors, it's about density matrices, that are a 2×2 matrices. Hence, equation (2.3) is equivalent to the equation (2.2), which is a description in terms of bloch vector X, Y and Z coordinates.

If we compute $\rho - D_Z(\rho)$ where

$$\rho = \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix}$$

so

$$\rho - D_Z(\rho) = \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} - \frac{1}{2}\left\{ \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \right\}$$

$$= \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} - \frac{1}{2}\left\{ \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} + \begin{pmatrix} \rho_{00} & -\rho_{01} \\ -\rho_{10} & \rho_{11} \end{pmatrix} \right\}$$

$$\rho - D_Z(\rho) = \begin{pmatrix} 0 & \rho_{01} \\ \rho_{10} & 0 \end{pmatrix}$$

By taking $A = \rho - D_Z(\rho)$, we have to compute the determinant of $A - \lambda I$ to find the eigenvalues:

$$A - \lambda I = \begin{pmatrix} 0 & \rho_{01} \\ \rho_{10} & 0 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} -\lambda & \rho_{01} \\ \rho_{10} & -\lambda \end{pmatrix}$$

Determinant $(A - \lambda I) = \lambda^2 - \rho_{01}\rho_{10}$, then

$$\lambda^2 - \rho_{01}\rho_{10} = 0$$

$$\lambda^2 = \rho_{01}\rho_{10}$$

$$\lambda = \pm\sqrt{|\rho_{01}||\rho_{10}|}$$

where, $\rho_{01} = \frac{x+iy}{2}$, $\rho_{10} = \frac{x-iy}{2}$ and $|\rho_{01}| = |\rho_{10}|$ because they are the complex conjugate of each other, so we have $|\rho_{01}|^2 = |\rho_{10}|^2 = \frac{x^2+y^2}{4}$,

$$\lambda = \pm\sqrt{|\rho_{01}|^2} = \pm\sqrt{\frac{x^2+y^2}{4}} = \pm\frac{1}{2}\sqrt{x^2+y^2}$$

so,

$$\lambda_1 = \frac{1}{2}\sqrt{x^2+y^2}; \lambda_2 = -\frac{1}{2}\sqrt{x^2+y^2}$$

The trace norm is the sum of the absolute values of these eigenvalues where $|\lambda| =$

$|\lambda_1| = |\lambda_2| = \frac{1}{2}\sqrt{x^2 + y^2}$

$$\text{Trace norm}(A - \lambda I) = |\lambda_1| + |\lambda_2| = 2|\lambda| = 2 \times \frac{1}{2} \times \sqrt{x^2 + y^2} = \sqrt{x^2 + y^2}$$

By replacing r in equation (2.3) with 1, so we end up with equation (2.2).

2.5 Cylinder state space with different radii

To develop efficient classical simulation, we have to demonstrate how the generalised separable decomposition can be obtained in terms of cylinder state spaces.

2.5.1 CZ gate acting on two input qubits

The important idea in this section is to show that if there are two inputs represented by two cylindrical state spaces with radii r to which the CZ gate has been applied, the output is separable with respect to two different cylindrical state spaces with larger radii R. It can be illustrated as follow:

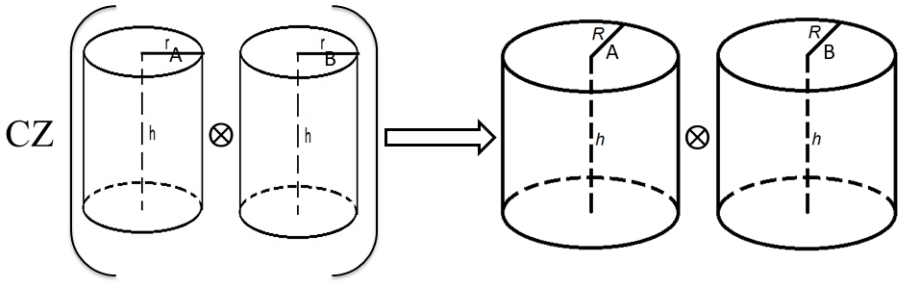


Figure 2.1: A CZ gate acts on two cylindrical state spaces with two radiir_A and r_B, the output is separable with respect to two cylindrical state spaces with larger radii R_A and R_B

Also, it can be demonstrated by the following Lemma:

Lemma 1. Consider a CZ gate acting on two cylindrical state spaces with two different radii. Let's us state this more precisely: let's take $CZ(cyl(r_A) \otimes cyl(r_B))$ and any operator, as being $cyl(r_A)$ and $cyl(r_B)$ separable, can be written in the generalised separable form if and only if (where we define $f_A := \frac{r_A}{R_A}$ and $f_B := \frac{r_B}{R_B}$)

$$(f_A + f_B)^2 + f_A^2 f_B^2 \leq 1 \tag{2.4}$$

Before starting with the proof, a question can be asked: when the CZ can be considered as a separable operation?
Let's define the growth factor as:

$$g_i := \frac{R_i}{r_i} \tag{2.5}$$

As we mentioned previously, if CZ gate acting on two cylindrical state spaces with different radii r, then the output that consists of two new cylindrical state spaces with larger radii R is separable. It means that g_i is large enough.

Let's consider the growth factors of the two states are equivalent. By applying the ratio to the determinant equation of the outer block we get:

$$1 - \frac{4}{g^2} - \frac{1}{g^4} \geq 0$$

The value of g is

$$g \geq \lambda := \sqrt{2 + \sqrt{5}} \approx 2.05817$$

so,

$$R \geq 2.05817r$$

This means that the radii of the output should be minimum twice the radii of the input, then the CZ can be considered a separable operation.

Proof. Let's start by writing the Pauli decomposition of two qubit operators that is represented by:

$$\rho_{AB} = \frac{1}{4}\sum_{i,j} \rho_{i,j}\sigma_i \otimes \sigma_j \tag{2.6}$$

where $\sigma_0, \sigma_1, \sigma_2$ and σ_3 are the 2×2 matrices representing I, X, Y and Z operators respectively. The above equation can also be represented as a 4×4 matrix with rows and columns numbered from $0, \ldots, 3$:

$$\begin{pmatrix} \rho_{00} & \rho_{01} & \rho_{02} & \rho_{03} \\ \rho_{10} & \rho_{11} & \rho_{12} & \rho_{13} \\ \rho_{20} & \rho_{21} & \rho_{22} & \rho_{23} \\ \rho_{30} & \rho_{31} & \rho_{32} & \rho_{33} \end{pmatrix}$$

By applying the CZ gate on the matrix we get:

$$\begin{pmatrix} \rho_{00} & \rho_{01} & \rho_{02} & \rho_{03} \\ \rho_{10} & \rho_{11} & \rho_{12} & \rho_{13} \\ \rho_{20} & \rho_{21} & \rho_{22} & \rho_{23} \\ \rho_{30} & \rho_{31} & \rho_{32} & \rho_{33} \end{pmatrix} \rightarrow \begin{pmatrix} \rho_{00} & \rho_{31} & \rho_{32} & \rho_{03} \\ \rho_{13} & \rho_{22} & -\rho_{21} & \rho_{10} \\ \rho_{23} & -\rho_{12} & \rho_{11} & \rho_{20} \\ \rho_{30} & \rho_{01} & \rho_{02} & \rho_{33} \end{pmatrix}$$

The product operator can be written as:

$$\frac{1}{2}(\sigma_0 + x_A\sigma_1 + y_A\sigma_2 + z_A\sigma_3) \otimes \frac{1}{2}(\sigma_0 + x_B\sigma_1 + y_B\sigma_2 + z_B\sigma_3),$$

for conciseness, we will represent it by the following vector notation

$$(1, x_A, y_A, z_A) \otimes (1, x_B, y_B, z_B)$$

To determine if the output of a CZ gate lead to cylindrical separable states that have different radii, we need to consider only the extremal input points from the top and the bottom faces of the cylinder. The reason for this consideration is that CZ gate is a linear transformation which is a mapping between two vector spaces that preserves the operations of vector addition; so a linear transformation commutes with addition. Furthermore, when CZ is applied on the the extremal input, the output is separable; then if a mixture of the extremal points is taken and CZ has been applied, the output is also separable because CZ is linear into a mixture of the original parts and those are separable. So the output for all the inputs will be separable.

We will take advantage of the symmetry around the Z axis. Let's consider a

cylinder separable decomposition:

$$CZ(\rho_A \otimes \rho_B) = \sum_i p_i \omega_A^i \otimes \omega_B^i$$

where $\omega_A^i \in Cyl(R_A)$ and $\omega_B^i \in Cyl(R_B)$. We know that CZ gate commutes with Z rotation U_z and cylinders are invariant under Z rotations, because of these properties, we have the following separable decomposition:

$$CZ(U_z^A(\rho_A) \otimes U_z^B(\rho_B)) = \sum_i p_i U_z^A(\omega_A) \otimes U_z^B(\omega_B)$$

This is used to reduce the number of inputs that we need to test. First, we set the Pauli operator $y = 0$ because if we take two inputs $(1, r_A, 0, \pm 1)$ and $(1, r_B, 0, \pm 1)$ and we apply the Z rotation, we notice that these inputs are similar to other inputs, such as $(1, r_A \cos\phi X, r_A \sin\phi Y, \pm 1)$ and $(1, r_B \cos\phi X, r_B \sin\phi Y, \pm 1)$ where $\phi \neq 0$. Then, if we obtain a separable decomposition for the first two inputs, we will end up with separable decompositions for the other inputs by applying the Z rotation.

So now, we can concentrate on the two inputs that have the form $(1, r_A, 0, \pm 1)$ and $(1, r_B, 0, \pm 1)$.

Let's see which value of z we need to take for the both inputs particles. If we take $z = 1$ for particle A, we will have:

$$CZ(1, r_A, 0, 1) \otimes (1, r_B, 0, \pm 1) = \sum_i p_i \omega_A^i \otimes \omega_B^i$$

But if we take $z = -1$ for particle A, the separable decomposition will be

$$CZ(1, r_A, 0, -1) \otimes (1, r_B, 0, \pm 1) = \sum_i p_i X \omega_A^i X^\dagger \otimes Z \omega_B^i Z^\dagger$$

So, we need to take $z = 1$ because changing z to -1 when $y = 0$, is the same as doing a Z rotation on the second particle. And because CZ is symmetric, the value of z in the second input will be equal to 1 as well.

The two input state particles are given by:

$$\frac{1}{2}(I + r_A X + 0Y + 1) \otimes \frac{1}{2}(I + r_B X + 0Y + 1) \tag{2.7}$$

After applying CZ on the input, the matrix will be represented by:

$$\begin{pmatrix} 1 & r_B & 0 & 1 \\ r_A & 0 & 0 & r_A \\ 0 & 0 & r_A r_B & 0 \\ 1 & r_B & 0 & 1 \end{pmatrix} \tag{2.8}$$

We need to show that the above matrix can be written as the outer product:

$$\sum_i p_i \begin{pmatrix} 1 \\ R_A \cos(\theta_i) \\ R_A \sin(\theta_i) \\ 1 \end{pmatrix} \begin{pmatrix} 1 & R_B \cos(\theta_i) & R_B \sin(\theta_i) & 1 \end{pmatrix}$$

If we take $\theta_i = 0$ and if we multiply the previous matrix by two 4×4 diagonal matrices:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{R_A} & 0 & 0 \\ 0 & 0 & \frac{1}{R_A} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{R_B} & 0 & 0 \\ 0 & 0 & \frac{1}{R_B} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

one on the left and the other on the right respectively, we end up with two cylinders with radii equal to 1, $Cyl(1)$.

By multiplying these two matrices by matrix (2.8), one on the right and the other on the left, we get the following matrix:

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 1 \\ \frac{r_A}{R_A} & 0 & 0 & \frac{r_A}{R_A} \\ 0 & 0 & \frac{r_A r_B}{R_A R_B} & 0 \\ 1 & \frac{r_B}{R_B} & 0 & 1 \end{pmatrix} \tag{2.9}$$

So, The matrix (2.8) is $Cyl(r_A)$, $Cyl(r_B)$ separable if and only if matrix (2.9) is $Cyl(1)$, $Cyl(1)$ separable.

If a cylinder decomposition exists for matrix (2.9), then all the z components in the matrix are replaced by 0 to obtain a quantum separable decomposition of a two qubit quantum operator, which means that by taking a vector in the $X - Y$ plane where $z = 0$, we end up with a bloch vector decomposition. On the contrary, if a

quantum separable decomposition exists,

$$\sum_i p_i(1, x_A^i, y_A^i, z_A^i) \otimes (1, x_B^i, y_B^i, z_B^i)$$

then all the z components are replaced by 1 to get a cylinder decomposition. So we know that determining the operator presented by matrix (2.9) is equivalent to determining a two qubit quantum operator.

When we replace all z components by 0, the above matrix can be seen as follows:

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 0 \\ \frac{r_A}{R_A} & 0 & 0 & 0 \\ 0 & 0 & \frac{r_A r_B}{R_A R_B} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{2.10}$$

This means that if matrix (2.10) has a quantum separable decomposition then (2.9) has cylinders separable decomposition with radii "1":

$$\sum_i p_i(1, x_A^i, y_A^i, 1) \otimes (1, x_B^i, y_B^i, 1)$$

Then (2.9) is $Cyl(1)$ separable which makes (2.8) is a $Cyl(r_A)$, $Cyl(r_B)$ separable if and only if (2.10) corresponds to a positive and PPT operator.

Now let's calculate and check the minimal eigenvalues of the operator represented by the matrix (2.10) are non-negative. In the form of Pauli operators, the equation is displayed as follows:

$$I + (\frac{r_A}{R_A} X \otimes I + \frac{r_B}{R_B} I \otimes X) + \frac{r_A r_B}{R_A R_B} Y \otimes Y$$

And its partial transpose

$$I + (\frac{r_A}{R_A} X \otimes I + \frac{r_B}{R_B} I \otimes X) - \frac{r_A r_B}{R_A R_B} Y \otimes Y$$

When the partial transpose has been applied, we noticed a changed in the third column which is the same as doing an X rotation. Now, we will work out the eigenvalues of one equation because we found that all the eigenvalues of these two operators are equivalent. Before computing the eigenvalues, we used the Hadamard gate on both qubits to get the following equation:

$$I + (\frac{r_A}{R_A} Z \otimes I + \frac{r_B}{R_B} I \otimes Z) - \frac{r_A r_B}{R_A R_B} Y \otimes Y \tag{2.11}$$

In computational basis, this is equal to

$$\begin{pmatrix} 1+f_A+f_B & 0 & 0 & f_A f_B \\ 0 & 1+f_A-f_B & -f_A f_B & 0 \\ 0 & -f_A f_B & 1-f_A+f_B & 0 \\ f_A f_B & 0 & 0 & 1-f_A-f_B \end{pmatrix} \tag{2.12}$$

Now, we can find the eigenvalues by dividing the matrix into two blocks: the inner block and the outer block. The determinant of the inner and the outer block are:

$$1-(f_A-f_B)^2-f_A^2 f_B^2$$
$$1-(f_A+f_B)^2-f_A^2 f_B^2$$

Between these two blocks, the determinant of the outer block is the lowest as f_A and f_B are positive. As we conclude, if the determinant of the outer block is non-negative then the output of the two cylindrical state spaces will be separable.

□

2.5.2 The diagonal gates acting on two input qubits

In this section, instead of applying the CZ gate, we will consider a diagonal gate that acts on two qubits. This specific gate, $V(\theta)$, is represented by the following matrix:

$$V(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix} \tag{2.13}$$

This matrix can be written in the form:

$$V(\theta) = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z_\theta$$

where θ goes from 0 to 2π and $Z_\theta = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}$.

As mentioned previously, to show that the output of this diagonal gate lead to cylindrical separable states that have different radii, we need to consider only extremal input points of the cylinder because if the output from the extremal inputs is separable then the output from all the inputs can be separable because the $V(\theta)$ gate is linear transformation.

To reduce the number of inputs, we take advantage of the symmetry around the z axis. By considering a cylinder separable decomposition,

$$V(\theta)(\rho_A \otimes \rho_B) = \sum_j p_j m_A^j \otimes m_B^j$$

where $m_A^j \in Cyl(R_A)$ and $m_B^j \in cyl(R_B)$, we know that $V(\theta)$ gate commutes with Z rotation U_z and cylinders are invariant under Z rotations; we have the decomposition:

$$V(\theta)(U_z^A(\rho_A) \otimes U_z^B(\rho_B)) = \sum_j p_j U_z^A(m_A) \otimes U_z^A(m_B)$$

If we take two inputs with $y = 0$ and we apply the Z rotation, we notice that these inputs can be taken to other inputs. Then, if we have a separable decomposition for the first two inputs, we can obtain separable decompositions for all other inputs by applying the Z rotation.

So, we will concentrate on the two inputs and we will consider all possibilities of Z, $(1, r_A, 0, \pm 1)$ and $(1, r_B, 0, \pm 1)$.

By taking the first possibility where $z = 1$ for both inputs,

$$V(\theta)((1, r_A, 0, 1) \otimes (1, r_B, 0, 1)) = \sum_j p_j m_A^j \otimes m_B^j$$

In pauli basis, these two inputs are given by

$$\frac{1}{2}(I + r_A X + 0Y + Z) \otimes \frac{1}{2}(I + r_B X + 0Y + Z) = \begin{pmatrix} 1 & r_B & 0 & 1 \\ r_A & r_A r_B & 0 & r_A \\ 0 & 0 & 0 & 0 \\ 1 & r_B & 0 & 1 \end{pmatrix}$$

In the previous part, we could reduce the number of inputs because when $z = -1$ in the first particle is the same as applying an X rotation and a Z rotation to the first and the second particle respectively. Then, the PPT criterion is used when CZ acted on the product term in which $z = 1$ in both particles.

Lemma 2. After applying the $V(\theta)$ gate on the matrix $(\rho_A \otimes \rho_B)$ i.e. this state is

represented by the following matrix in the pauli basis,

$$\begin{pmatrix} 1 & r_B & 0 & 1 \\ r_A & r_A r_B & 0 & r_A \\ 0 & 0 & 0 & 0 \\ 1 & r_B & 0 & 1 \end{pmatrix}$$

we end up with the following:

- the output top row is identical to the bottom row,
- the left column and the right column of the output are the same.

Hence, PPT criterion can be applied in the possibility where $z = 1$ for both inputs.

Proof. We consider $V(\theta)(\rho_A \otimes \rho_B)V(\theta)^\dagger = \rho_{out}$. We need to find if the top and the bottom row are the same. So, we subtract these two rows to see if we will end up with 0. The top row is given by $tr_A[(I_A \otimes I_B)\rho_{out}]$ and the bottom row is given by $tr_A[(Z_A \otimes I_B)\rho_{out}]$.

$$\begin{aligned}
\text{Top - bottom row} &= tr_A[(I_A \otimes I_B)\rho_{out} - (Z_A \otimes I_B)\rho_{out}] \\
&= tr_A[(I_A - Z_A) \otimes I_B \rho_{out}] \\
&= tr_A[(I - Z)_A \otimes I_B V(\theta)(\rho_A \otimes \rho_B)V(\theta)^\dagger] \\
&= tr_A[2|1\rangle\langle 1| \otimes I_B \left(|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes Z_\theta\right)(\rho_A \otimes \rho_B)V(\theta)^\dagger] \\
&= tr_A[2|1\rangle\langle 1| \otimes I_B \left(|1\rangle\langle 1| \otimes Z_\theta\right)(\rho_A \otimes \rho_B)V(\theta)^\dagger] \\
&= tr_A[2|1\rangle\langle 1| \otimes I_B(\rho_A \otimes Z_\theta \rho_B)V(\theta)^\dagger] \\
&= tr_A[2|1\rangle\langle 1|\rho_A \otimes Z_\theta \rho_B Z_\theta^\dagger] \\
&= 2\langle 1|\rho_A|1\rangle Z_\theta \rho_B Z_\theta^\dagger; \text{where } \langle 1|\rho_A|1\rangle = 0 \text{ because it is on the surface of the cylinder.} \\
&= 0
\end{aligned}$$

By obtaining that the top and the bottom row are the same, this argument is the main factor for the use of PPT criterion when $z = 1$ for both inputs. □

Explicitly, we can go through the same argument for all other possibilities $(1, r_A, 0, \pm 1) \otimes (1, r_B, 0, \pm 1)$, and we find that we end up with the same result. Therefore we know that PPT can be applied in this section.

The growth factors of different thetas

In this part, we will attempt to find the growth factors for each gate $V(\theta)$ where θ will change from 0 to 2π. We will first work out the action of $V(\theta)$ gate on all product of pauli. So, we have to apply a tensor product between each two pauli matrices from the first and the second inputs respectively. For example, we choose I from the first input and then we do a tensor product between the chosen pauli matrix and the I, X, Y, and Z from the second input. The same is applied for X, Y, and Z from the first input (i.e.$I \otimes I$, $I \otimes X$, $I \otimes Y$, $I \otimes Z$, $X \otimes I$, $X \otimes X$, etc). Then, we obtained a 4×4 matrix. Then, we multiply the matrix $V(\theta)$ with each of the calculated 4×4 pauli matrix and with the conjugate matrix of $V(\theta)$. That's how We end up with matrices of 4×4. For more details, see Appendix A.

After written out these matrices in terms of pauli operators, we note down all the coefficients in this 16×16 matrix, that we called "$VV(\theta)$"

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & a & b & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c & d & 0 \\
0 & d & a & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & b & c & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & e & 0 & 0 & j & p & 0 & 0 & q & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & e & p & 0 & 0 & p & j & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & q & e & 0 & 0 & k & p & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & j & 0 & 0 & e & q & 0 & 0 & p & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & q & 0 & 0 & p & e & 0 & 0 & j & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & q & k & 0 & 0 & e & p & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & j & q & 0 & 0 & q & e & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & p & 0 & 0 & q & j & 0 & 0 & e & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & c & d & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a & b & 0 \\
0 & b & c & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & d & a & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{matrix}
(I \otimes I) \\ (I \otimes X) \\ (I \otimes Y) \\ (I \otimes Z) \\ (X \otimes I) \\ (X \otimes X) \\ (X \otimes Y) \\ (X \otimes Z) \\ (Y \otimes I) \\ (Y \otimes X) \\ (Y \otimes Y) \\ (Y \otimes Z) \\ (Z \otimes I) \\ (Z \otimes X) \\ (Z \otimes Y) \\ (Z \otimes Z)
\end{matrix}$$

where

$$a = \frac{1}{2} + \frac{1}{2}\cos\theta; b = \frac{1}{2}\sin\theta; c = \frac{1}{2} - \frac{1}{2}\cos\theta; d = -\frac{1}{2}\sin\theta;$$

$$e = \frac{1}{2}+\frac{1}{4}\exp(-i\theta)+\frac{1}{4}\exp(i\theta); j = \frac{1}{2}-\frac{1}{4}\exp(-i\theta)-\frac{1}{4}\exp(i\theta); k = -\frac{1}{2}+\frac{1}{4}\exp(-i\theta)+\frac{1}{4}\exp(i\theta);$$

$$p = \frac{1}{4}i\exp(-i\theta) - \frac{1}{4}i\exp(i\theta); q = -\frac{1}{4}i\exp(-i\theta) + \frac{1}{4}i\exp(i\theta)$$

Our goal is to check whether the output is within the output state space which is the convex hull of all the points on the cylinder. So, we are going to theoretically and numerically work out the required growth factor g for different values of θ, where g is a function of $theta$, to make the output separable. We know two values of g:

- When θ is 0, the growth factor g is 1. Because when $\theta = 0$, $V(1)$ is the identity gate and there is no entangling.
- By taking $z = 1$ for both inputs $(1, r_{A/B}, 0, 1)$, we know from the previous section we got $g \geq 2.05817$ when θ is equal to π.

Product input with component z=1

We will start by computing the growth factor theoretically. As we showed previously, to determine if the output of $V(\theta)$ gate leads to cylindrical separable state, we only need to consider extremal input points of the cylinder. We know that $V(\theta)$ gate commutes with Z rotation U_z and cylinders are invariant under Z rotations; which means that we are dealing with the two inputs used in the previous section:

$$\frac{1}{2}(I + r_A X + 0Y + 1) \otimes \frac{1}{2}(I + r_B X + 0Y + 1)$$

When we apply $V(\theta)$ to the product input, the matrix is given by:

$$\begin{pmatrix} 1 \\ r_B \\ 0 \\ 1 \\ r_A \\ (\frac{1}{2}+\frac{1}{4}e^{-i\theta}+\frac{1}{4}e^{i\theta})r_Ar_B \\ (-\frac{1}{4}ie^{-i\theta}+\frac{1}{4}ie^{i\theta})r_Ar_B \\ r_A \\ 0 \\ (-\frac{1}{4}ie^{-i\theta}+\frac{1}{4}ie^{i\theta})r_Ar_B \\ (\frac{1}{2}-\frac{1}{4}e^{-i\theta}-\frac{1}{4}e^{i\theta})r_Ar_B \\ 0 \\ 1 \\ r_B \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} I\otimes I \\ I\otimes X \\ I\otimes Y \\ I\otimes Z \\ X\otimes I \\ X\otimes X \\ X\otimes Y \\ X\otimes Z \\ Y\otimes I \\ Y\otimes X \\ Y\otimes Y \\ Y\otimes Z \\ Z\otimes I \\ Z\otimes X \\ Z\otimes Y \\ Z\otimes Z \end{pmatrix}$$

If we represent the above 16×1 matrix by a 4×4 matrix, we have:

$$\begin{pmatrix} 1 & r_B & 0 & 1 \\ r_A & (\frac{1}{2}+\frac{1}{4}e^{-i\theta}+\frac{1}{4}e^{i\theta})r_Ar_B & (-\frac{1}{4}ie^{-i\theta}+\frac{1}{4}ie^{i\theta})r_Ar_B & r_A \\ 0 & (-\frac{1}{4}ie^{-i\theta}+\frac{1}{4}ie^{i\theta})r_Ar_B & (\frac{1}{2}-\frac{1}{4}e^{-i\theta}-\frac{1}{4}e^{i\theta})r_Ar_B & 0 \\ 1 & r_B & 0 & 1 \end{pmatrix} \tag{2.14}$$

As we mentioned in the previous section,we will show that the matrix can be written as the outer product:

$$\sum_i p_i \begin{pmatrix} 1 \\ R_A\cos(\theta_i) \\ R_A\sin(\theta_i) \\ 1 \end{pmatrix} \begin{pmatrix} 1 & R_B\cos(\theta_i) & R_B\sin(\theta_i) & 1 \end{pmatrix}$$

So, we multiply the previous matrix by two 4×4 diagonal matrices:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{R_A} & 0 & 0 \\ 0 & 0 & \frac{1}{R_A} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} and \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{1}{R_B} & 0 & 0 \\ 0 & 0 & \frac{1}{R_B} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

one on the left and the other on the right respectively, we end up with two cylinders with radii equal to 1, $Cyl(1)$.

We get the following matrix:

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 1 \\ \frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & \frac{r_A}{R_A} \\ 0 & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 1 & \frac{r_B}{R_B} & 0 & 1 \end{pmatrix} \tag{2.15}$$

So, The matrix (2.14) is $Cyl(r_A)$, $Cyl(r_B)$ separable if and only if matrix (2.15) is $Cyl(1)$, $Cyl(1)$ separable.

Then, we replace all z components by 0 to obtain a quantum separable decomposition of a two qubit quantum operators, the above matrix can be seen as follows:

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 0 \\ \frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{2.16}$$

The partial transpose of the above matrix will be equal to:

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 0 \\ \frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & -(-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & -(\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{2.17}$$

Using the the pauli operators, the matrix (2.16) can be written as :

$I + \frac{r_B}{R_B} I \otimes X + \frac{r_A}{R_A} X \otimes I + [(\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})X \otimes X + (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})Y \otimes Y]\frac{r_A r_B}{R_A R_B} + (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})(X \otimes Y + Y \otimes X)\frac{r_A r_B}{R_A R_B}$

And the partial transpose is given by:

$$I + \frac{r_B}{R_B} I \otimes X + \frac{r_A}{R_A} X \otimes I + [(\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})X \otimes X - (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})Y \otimes Y]\frac{r_A r_B}{R_A R_B} + (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})(-X \otimes Y + Y \otimes X)\frac{r_A r_B}{R_A R_B}$$

When the partial transpose has been applied, we noticed only a change in the third column which is the same as doing an X rotation. When the X rotation is applied, the eigenvalues of the two matrices (2.16) and (2.17) are the same. We checked numerically if the minimum eigenvalues don't change and we found that they are equivalent for different values of theta starting from 0 ending at 2π.

Now, by taking the symmetric case $R_A = R_B = R$ and for each value of R going from 1 to 2.1 where the input radius is 1, we store the value of R only when the minimum eigenvalue for each value of theta becomes negative, it means that when the minimum eigenvalue of a specific theta is negative, it requires a the growth R to be separable. We end up with a graph that represents the storing value of R for each theta.

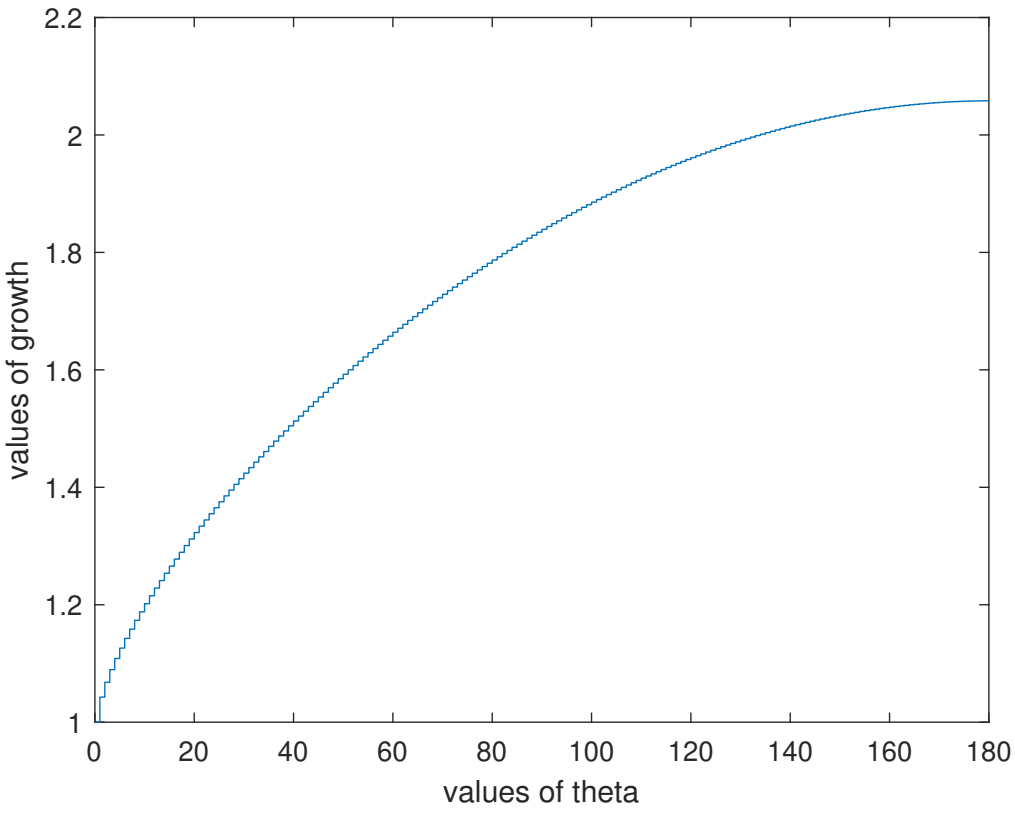


Figure 2.2: The required growth for each θ to end up with a separable gate.

It turns out that the CZ gate, where $\theta = \pi$, requires the most growth in R.

In these following parts, the same procedure will be repeated but for different product inputs.

Product input with component z=1 and z=-1

We will start by applying the diagonal $V(\theta)$ gate on the product input that can be written in the form:

$$\frac{1}{2}(I + r_A X + 0Y + 1) \otimes \frac{1}{2}(I + r_B + 0Y - 1)$$

So $V(\theta)(1, r_A, 0, 1) \otimes (1, r_B, 0, -1)V(\theta)^\dagger$ is given by a 4×4 matrix:

$$\begin{pmatrix} 1 & r_B & 0 & -1 \\ (\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})r_A & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})r_A r_B & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})r_A r_B & (-\frac{1}{2}e^{-i\theta} - \frac{1}{2}e^{i\theta})r_A \\ (-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})r_A & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})r_A r_B & (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})r_A r_B & (\frac{1}{2}ie^{-i\theta} - \frac{1}{2}ie^{i\theta})r_A \\ 1 & r_B & 0 & -1 \end{pmatrix}$$

This matrix is separable if and only if the following matrix is $cyl(1), cyl(1)$ separable

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & -1 \\ (\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})\frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{2}e^{-i\theta} - \frac{1}{2}e^{i\theta})\frac{r_A}{R_A} \\ (-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})\frac{r_A}{R_A} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_A}{R_B R_B} & (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (\frac{1}{2}ie^{-i\theta} - \frac{1}{2}ie^{i\theta})\frac{r_A}{R_A} \\ 1 & \frac{r_B}{R_B} & 0 & -1 \end{pmatrix}$$

By replacing all z component by 0, a quantum separable decomposition is obtained,

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 0 \\ (\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})\frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ (-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})\frac{r_A}{R_A} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_A}{R_B R_B} & (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

The matrix can be written as:

$I + \frac{r_B}{R_B} I \otimes X + [(\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})X \otimes I + (-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})Y \otimes I]\frac{r_A}{R_A} + [(\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})X \otimes X + (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})(X \otimes Y + Y \otimes X) + (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})Y \otimes Y]\frac{r_A r_B}{R_A R_B}$

The partial transpose of the matrix is given by

$$\begin{pmatrix} 1 & \frac{r_B}{R_B} & 0 & 0 \\ (\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})\frac{r_A}{R_A} & (\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ -(-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})\frac{r_A}{R_A} & -(-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})\frac{r_A r_A}{R_B R_B} & -(\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})\frac{r_A r_B}{R_A R_B} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

it is same as:

$I + \frac{r_B}{R_B} I \otimes X + [(\frac{1}{2}e^{-i\theta} + \frac{1}{2}e^{i\theta})X \otimes I - (-\frac{1}{2}ie^{-i\theta} + \frac{1}{2}ie^{i\theta})Y \otimes I]\frac{r_A}{R_A} + [(\frac{1}{2} + \frac{1}{4}e^{-i\theta} + \frac{1}{4}e^{i\theta})X \otimes X + (-\frac{1}{4}ie^{-i\theta} + \frac{1}{4}ie^{i\theta})(X \otimes Y - Y \otimes X) - (\frac{1}{2} - \frac{1}{4}e^{-i\theta} - \frac{1}{4}e^{i\theta})Y \otimes Y]\frac{r_A r_B}{R_A R_B}$

As before, when we apply PPT criterion to the matrix, we end up with negative signs in the third row or in the third column, which also can be the same as applying the X rotation. Then, we will check numerically the eigenvalues of these last two matrices if they are equal. We notice that the minimum eigenvalues are the same. In addition, we notice that the minimum eigenvalues of these matrices are identical to those of matrices (2.16) and (2.17).

As mentioned in the previous part, we are going to find the necessary value of growth for each theta to end up with separable gates.

By taking the symmetric case, $R_A = R_B = R$ and by changing the z component of the second particle in the product input, it turns out that the same graph (2.2) is obtained and the CZ gate requires the most growth in R.

For the two remaining cases where the product input has

- component $z = -1$ and $z = 1$ for the first and second particle respectively,
- and $z = -1$ for both particles,

the same procedure has been applied. A 4×4 matrix is obtained after $V(\theta)$ acting on the product input. This matrix is $Cyl(r_A), Cyl(r_B)$ separable if and only if $Cyl(1), Cyl(1)$ is separable. Then all the components of z in the matrix are replaced by 0, to have a quantum separable decomposition. PPT is applied, as before, we notice that all minimum eigenvalues are identical. We compare all the minimum eigenvalues with the previous two cases, where $z = 1$ for both particles and $z = -1$ for the second particle, we notice that these eigenvalues remain the same.

By considering the symmetric case where $R_A = R_B = R$, we produce the graph that represents the required value of growth or R for each θ going from 0 to 2π to have separable gates. The obtained graphs of these two cases are exactly the same as the graph (2.2).

2.6 Summary of chapter 2

We started by relaxing the notion of separability to get a new generalised notion that we called cylinder separability, where, instead of Bloch sphere, we had cylinders.

Then we applied this notion to systems where we changed the inputs $|\psi\rangle$ of the cluster state quantum computation, i.e. the modified input state has the unormalised form $|\psi\rangle = |0\rangle + \epsilon|1\rangle$. Then we find that, after applying CZ gates on two cylindrical state spaces with radii r, the output is separable with respect to two different cylindrical state spaces with larger radii $R = \lambda r$. Separability can be maintained by growing the cylinder by λ but we must not grow beyond radius 1, because this state space should be in the dual of the cylinder measurements. Then, instead of the CZ gates, we applied diagonal gates $V(\theta)$ on two cylindrical state spaces. We tried to find the minimum growth required for separability for each value of θ and it turned out that the CZ gate is the gate that requires the most growth factor λ to maintain separability. In the following chapter, we will find that, by growing the size of new state spaces, the system can be efficiently simulated classically.

Chapter 3

Different state spaces

To efficiently simulate a complex quantum system on a classical computer, we approximate a pure entangled quantum state by a separable decomposition. As we mentioned in the previous chapter, we use a state space as the cylinder instead of Bloch sphere to get a separable decomposition.

In this chapter, we will try to find a non-quantum state space that is better than the cylinder. So, we are going to try to find a state space that starts with greater quantum inputs than the cylinder, but grows more slowly each time CZ gate has been applied to maintain the separable decomposition, so that we do not exit the dual of the allowed measurement. However, it turns out that the cylinder is the most optimal state space among all the state spaces that we searched through, which means that the cylinder requires the lowest growth of $\sqrt{2+\sqrt{5}} \approx 2.05817$.

3.1 State spaces

We consider state spaces that are convex hull of $N(5-2)+2$ vertices where N is an integer number and these vertices are described by Bloch vectors. The considered state spaces are chosen because of the symmetry about the z axis and they are among the simplest we could try for relatively small number of extremal points. This study requires small number of extremal points because we are going to use Linear Programming on Matlab on a regular laptop to work out numerically the growth factors to see if there is a state space grows less than the cylinder. If our trial shapes have too many points, the algorithms run prohibitively slowly.

We will begin by describing the boundary of these state spaces in the positive x part of the $X-Z$ plane only. Then this cross-section will be rotated around the z axis N times to construct the whole shape.

The boundary of these state spaces in the positive x part of the $X-Z$ plane is defined by five points. These points are $Q=(0,1)$, $S=(0,-1)$, which represent

the top and the bottom of the state space respectively, $T = (1, 0)$ which represents a point at the "equator", and two other points $K = (w, h)$ and $L = (w, -h)$ where "h" represents a chosen height and "w" represents a related width. For instance, the following two dimensional graph illustrates the boundary we get in the $X - Z$ plane for $h = 0.5$ and $w = \sqrt{\frac{3}{4}}$.

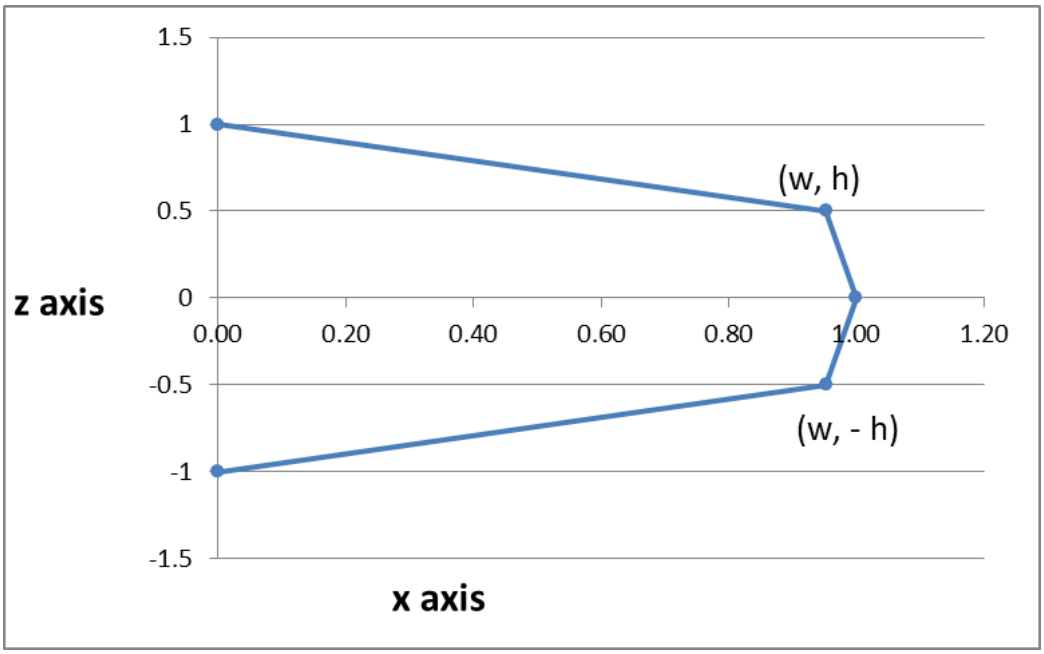


Figure 3.1: The boundary of the state space in the $X - Z$ plane for $h = 0.5$ and $w = \sqrt{\frac{3}{4}}$

We will repeat this boundary by rotating it N times about the z axis, at regular azimuthal angle intervals $\theta = n\frac{2\pi}{N}$, where $n = 0, \ldots, N-1$, so overall, the vertices of the state spaces will be given by:

- $(0, 0, 1)$ represents the top of the state space
- $(0, 0, -1)$ represents the bottom of the state space
- $(w\cos(\theta),\ w\sin(\theta),\ \pm h)$
- $(\cos\theta,\ \sin(\theta),\ 0)$ where these points are located on the horizontal plane

The following image illustrates the state space with $h = 0.5$, $w = \sqrt{\frac{3}{4}}$ and $N = 6$ which is the total number of angles.

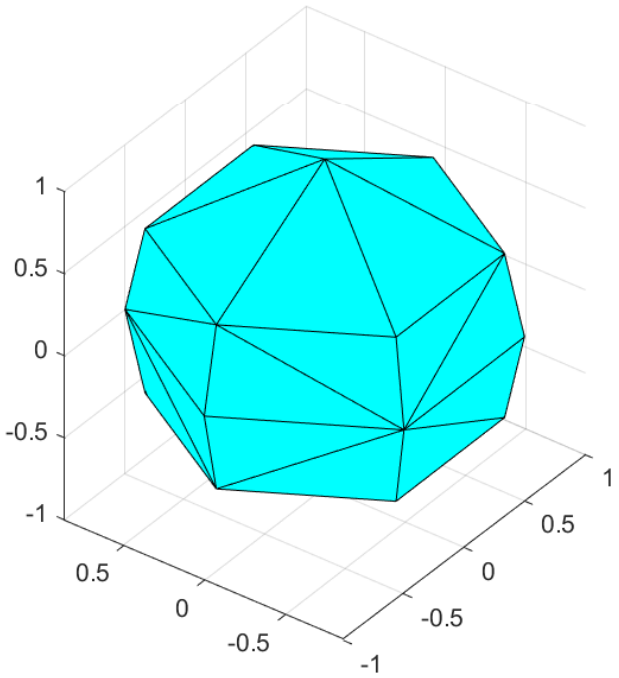


Figure 3.2: The state space for $N = 6$, $h = 0.5$ and $w = \sqrt{\frac{3}{4}}$

This means that the shape of the state space that we will consider, is parameterized by three numbers: N, h and w. We will change these values to find the growth factor needed to maintain separability.

But, to see the relationship between these shapes and the bloch sphere, we are going to define a new parameter ϵ to replace the parameter w, where we have:

$$w = (\sqrt{1-h^2})(1+\epsilon) \tag{3.1}$$

When $\epsilon = 0$ all the vertices are on the bloch sphere surface. When ϵ is bigger than 0, some of the vertices protrude outside the Bloch sphere. Then we will vary the value of ϵ, which does not need to be small, to see what happens to the growth factor when we step away from the Bloch sphere. Note however that when the value of ϵ is too high, where $w > 1$,

$$\epsilon > \frac{1}{\sqrt{1-h^2}} - 1 \tag{3.2}$$

the shape of these state spaces changes, and we end up with a non-convex shape. For instance, the following graph represents the obtained boundary in the $X - Z$ plane for $h = 0.5$ and $w = \sqrt{\frac{3}{4}}$ when $\epsilon = 1.1$; in theses cases, we are going to take the convex hull which reduce the number of extremal points and change the outermost boundary to a flat vertical face as illustrated in the figure,

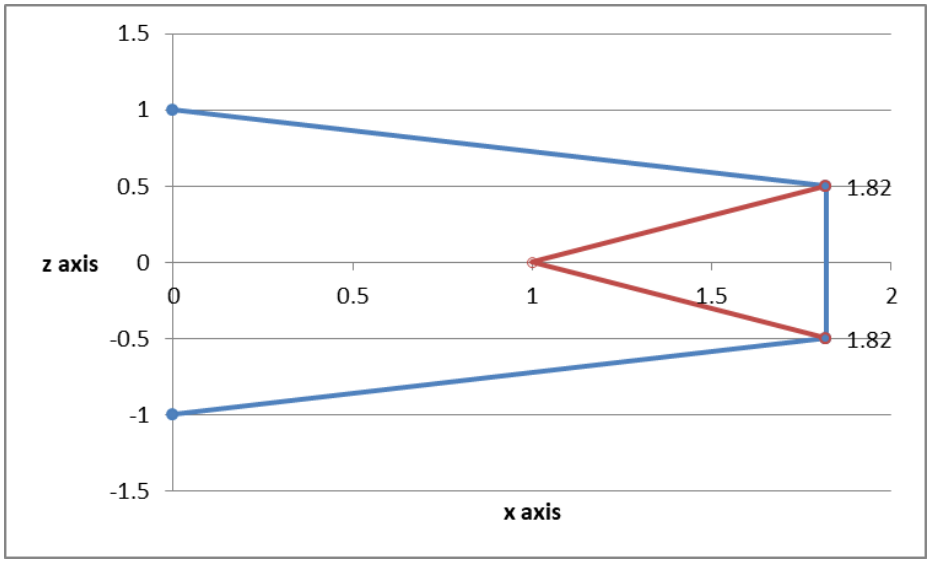


Figure 3.3: The boundary of the convex hull in the $X - Z$ plane for $h = 0.5$, $\epsilon = 1.1$ and $w = \sqrt{\frac{3}{4}}$

As a conclusion, we are going to use three parameters: N, h and ϵ. When $\epsilon \leq \frac{1}{\sqrt{1-h^2}} - 1$, we end up with cross-section which is illustrated by the following graph,

and when $\epsilon > \frac{1}{\sqrt{1-h^2}} - 1$, we end up with cross-section with a flat boundary as

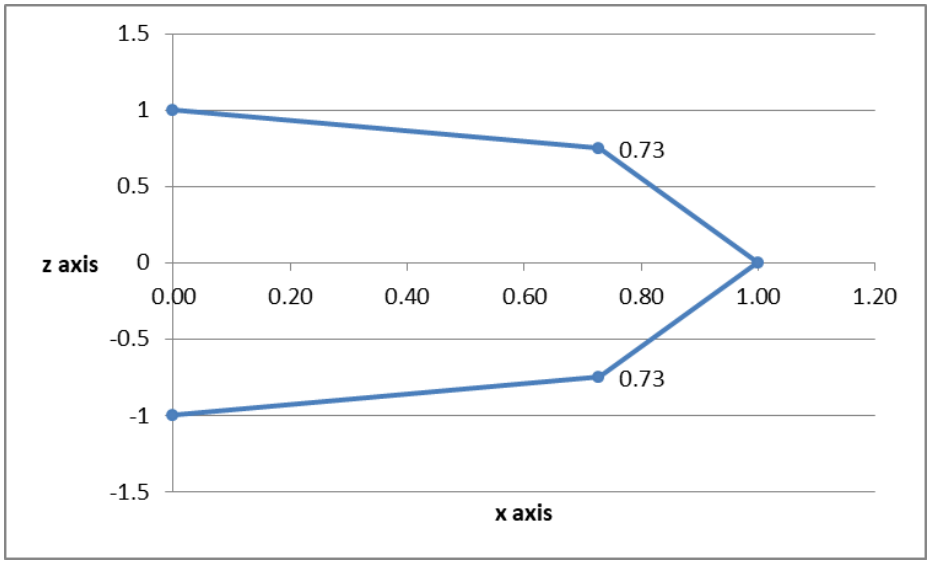


Figure 3.4: The boundary of the convex hull in the $X - Z$ plane for $h = 0.75$, $\epsilon = 0.1$ and $w = \sqrt{\frac{7}{16}}$

illustrated, for example, in the following figure,

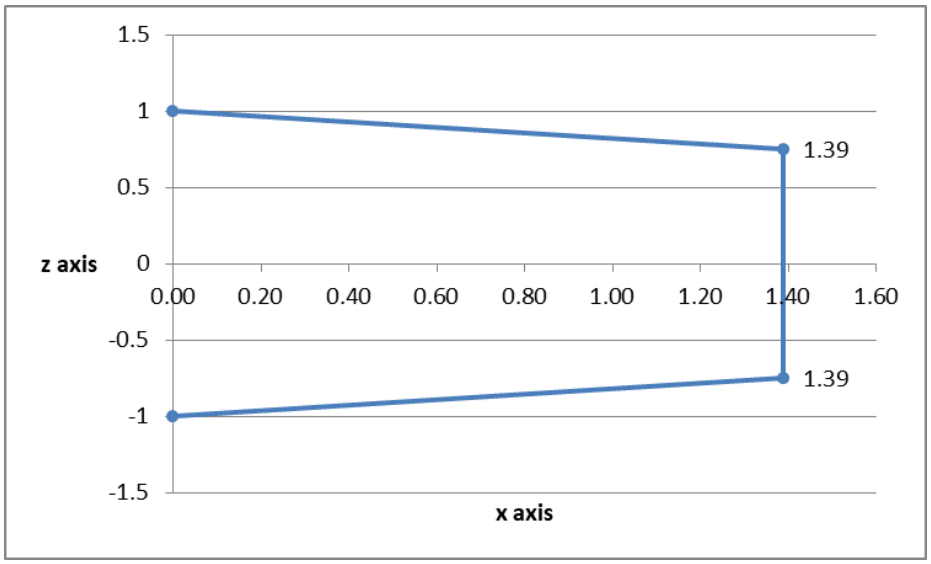


Figure 3.5: The boundary of the convex hull in the $X - Z$ plane for $h = 0.75$, $\epsilon = 1.1$ and $w = \sqrt{\frac{7}{16}}$

Our goal in this chapter is to use *Linear Programming* to work out numerically the growth factors for different values of N, h and ϵ to see whether any of these shapes may grow slower than the cylinder.

3.2 Using Linear Programming to decide the convex hull membership

The problem that we are working on is whether we end up with separability, it means that we need to work out whether a point is in the convex hull of a set of points, in our case we are going to use the Linear Programming [18].

In this section, we will describe a general method of using Linear Programming to decide if a point is inside the convex hull. We start by considering a set G of vectors where $G = \{g_1, \ldots, g_d\}$ where the $g_i \in \mathbb{R}^n$. The state space is the convex hull of G and we attempt to work out whether a vector v is in the convex hull of G. So we use the Linear Programming to check if the vector is inside the convex hull and if the problem is feasible.

The Linear Programming consists of minimizing $C^T x$ where C is *Linear objective function*. The minimisation is subject to a linear equality and linear inequality constraints where both of them consist of a matrix and a vector.

The equality constraint states that $\sum_i x_i g_i = v$. This constraint can be presented as:

$$\begin{pmatrix} & & & \\ g_1 & g_2 & \ldots & \\ & & & \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} = v$$

If x satisfies this equality constraint, it means that v can be written as a linear combination of g_i vectors.
In order to have v inside the convex hull of G, x should be positive numbers and $\sum_i x_i = 1$. Then in our case, the linear inequality constraint consists of an identity matrix I and a zero vector b, where we have,

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 \\ & & & & & \\ 0 & 0 & 0 & 0 & \ldots & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \geq b$$

So v is not just a linear combination but it is a convex combination of g_i vectors.

We are not interested in optimization, we only care whether these constraints can

be satisfied to get v in the convex hull of G. So we can choose any vector for the linear objective function but we pick the zero vector.

After applying the Linear Programming, if the problem is feasible then the vector v is inside the convex hull but if the problem is infeasible, then $v \notin G$.

3.3 Linear Programming

In this section, we will see how to cast our problem in the form that we discussed in the previous section. For a particular shape, the set G contains vectors g_i. g_1 and g_2 are the vectors that represent the top and the bottom of the state space,

$$g_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \text{ and } g_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix}$$

For g_j where $j \geq 3$, these vectors have two forms. The first form is given by

$$\begin{pmatrix} 1 \\ Rw(1+\epsilon)\cos(\theta) \\ Rw(1+\epsilon)\sin(\theta) \\ \pm h \end{pmatrix}$$

where R is the output radius, $\theta = n\frac{2\pi}{N}$ and $n = 0, \ldots, N-1$, and the second form represents the points located on the horizontal plane,

$$\begin{pmatrix} 1 \\ \cos(\theta) \\ \sin(\theta) \\ 0 \end{pmatrix}$$

Note that g_i begin always with 1 which is the coefficient of the pauli operator I because of normalisation.

After putting all these g_i together, we end up with a $4 \times (N(5-2)+2)$ output matrix, $m_{\text{output state space}}$. This matrix describes the shape, for instance by using Matlab, we take $R = 2.468$, $h = 0.5$, $w = \sqrt{\frac{3}{4}}$, $\epsilon = 0$ and $N = 6$, we obtain the following

output matrix,

$$m_{\text{output state space}} = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 2.14 & 1.07 & \dots & 1.07 \\ 0 & 0 & 0 & 1.85 & \dots & -1.85 \\ 1 & -1 & 0.5 & 0.5 & \dots & -0.5 \end{pmatrix}$$

Each of these columns represents a vertex in the state space.
In our case, G is going to be the product of $m_{\text{output state space}}$. So, the set G which is represented by a $(16 \times (N(5-2)+2)^2)$ matrix, is given by,

$$G = m_{\text{output state space}} \otimes m_{\text{output state space}}$$

.

The matrix $m_{\text{input state space}}$ has the same coefficients as $m_{\text{output state space}}$ but the X and Y coordinates are not multiplied by R. Each column in $m_{inputstatespace}$ represents a vector t_i where $i = 1, \dots, N(5-2)+2$. Then v is constructed from a particular pair of inputs $t_i \otimes t_j$ acting with the CZ gate. Explicitly, each v is is going to be in the following form,

$$v = CZ(t_i \otimes t_j)$$

We need to work out whether all the $t_i \otimes t_j$ inputs are taken to separable outputs. But we are not going to test the $(N \times (5-2)+2)^2$ inputs, we only need to take 5^2 inputs because the CZ commutes with the Z rotation; because of this property, we have the following separable decomposition:

$$CZ(U_{zi}(t_i) \otimes U_{zj}(t_j)) = \sum_k p_k U_{zi}(\lambda_i) \otimes U_{zj}(\lambda_j)$$

If CZ is applied on the measurement operators, X and Y measurements will be rotated. Then if we apply the Z rotation on a separable decomposition for inputs with $y = 0$, we find that the other inputs with $y = \sin(\theta)$ as a coefficient of Y will have separable decomposition.

So the inputs t_i are the following vectors where y has the coefficient 0,

$$t_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}, t_2 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix}, t_3 = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \text{ and for } t_j \geq 4 \; t_j = \begin{pmatrix} 1 \\ w(1+\epsilon) \\ 0 \\ \pm h \end{pmatrix}$$

where t_1, t_2 represent the top and the bottom of the state space respectively, t_3

represents the point on the horizontal line and the other vectors have the form t_j. For example, by using Matlab, we take $R = 2.468$, $h = 0.5$, $w = \sqrt{\frac{3}{4}}$, $\epsilon = 0$ and $N = 6$, we present one pair of input $m_{\text{input state space}}$,

$$m_{\text{input state space}} = \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 0 & 0.866 & 0.433 & \dots & 0.433 \\ 0 & 0 & 0 & 0.75 & \dots & -0.75 \\ 1 & -1 & 0.5 & 0.5 & \dots & -0.5 \end{pmatrix}$$

We will vary the output radius R across a range until the problem becomes feasible, then this specific R is taken as the minimum growth required to have separability.

We start by fixing the output radius on the first value of the chosen range for a specific value of N. We will fix the first input t_1 and apply the CZ gate on a tensor product between the fixed input and itself. Then we use the Linear Programming to check if it ends up with a separable decomposition. So we have two cases:

1. If it has such decomposition, we do the same process but we change the tensor product, which will occur between the fixed and the second input t_2. If we get a separable decomposition, the same procedure will be repeated until we end up with a separable decomposition for all the 5 inputs. Then, we fix the second input instead of the first and re-do the same technique until we obtain a separable decomposition for a tensor product between the final input and itself. If all these tensor products have separable decomposition then the output radius is considered as the optimal one. But if in the middle of this procedure, we don't reach a separable decomposition, the second case will be applied.

2. If we don't obtain a separable decomposition, then the same procedure will be replicated with the second value of output radius until it reaches the last required output radius in the range.

The same procedure is repeated but for different values of N. If the procedure ends up by giving us a minimum value of the growth factor (or a minimum output radius by taking input radius equal to 1), this value will be compared with the growth (or the output radius) of the cylinder to choose which state space is optimal. But if the process has gone through all the radii and we don't have a separable decomposition, we know that the cylinder is the optimal state space.

3.4 Results

Our state spaces are characterised by h, N and ϵ. We vary these parameters to get a minimum growth factor. The following table shows, for each state space, the minimum radius growth required for all the pairs of inputs to end up with a separable output.

height	ϵ	**N**	**Growth factor or R**
0.5	0	6	2.468
		10	2.442
		15	2.442
		20	2.415
	0.1	6	2.507
		10	2.434
		15	2.443
		20	2.415
	1.1	6	2.528
		10	2.434
		15	2.443
		20	2.415
0.75	0	6	2.514
		10	2.439
		15	2.443
		20	2.415
	0.1	6	2.528
		10	2.452
		15	2.443
		20	2.415
	1.1	6	2.528
		10	2.452
		15	2.443
		20	2.415

Table 3.1: The radius growth for different state spaces

h	ϵ	**N**	**Growth factor or R**
0.8	0	6	2.528
		10	2.434
		15	2.443
		20	2.415
	0.1	6	2.528
		10	2.452
		15	2.443
		20	2.415
	1.1	6	2.528
		10	2.452
		15	2.443
		20	2.415
0.6	0	6	2.484
		10	2.447
		15	2.439
		20	2.415
	0.1	6	2.515
		10	2.447
		15	2.441
		20	2.415
	1.1	6	2.528
		10	2.447
		15	2.441
		20	2.415
0.4	0	6	2.465
		10	2.436
		15	2.441
		20	2.415
	0.1	6	2.500
		10	2.448
		15	2.441
		20	2.415
	1.1	6	2.500
		10	2.448
		15	2.441
		20	2.415

Table 3.2: The radius growth for different state spaces

h	ϵ	N	Growth factor or R
0.1	0	6	2.488
		10	2.432
		15	2.432
		20	2.415
	0.1	6	2.484
		10	2.429
		15	2.429
		20	2.415
	1.1	6	2.484
		10	2.429
		15	2.429
		20	2.415
0.25	0	6	2.458
		10	2.431
		15	2.438
		20	2.415
	0.1	6	2.468
		10	2.444
		15	2.443
		20	2.415
	1.1	6	2.468
		10	2.444
		15	2.443
		20	2.415

Table 3.3: The radius growth for different state spaces

As a conclusion of this section, when we change the height for $N = 20$, which is the total number of points around the equator, we notice that the minimum value of growth factor is 2.145 for all the considered state spaces in both cases: within and outside the Bloch sphere. This means that the cylinder with a growth factor 2.05817 is the most optimal state space among all the state spaces that we studied.

3.5 Summary of chapter 3

In this chapter, we started by considering new state spaces that are the convex hull of $N(5-2)+2$ vertices. Then, we grew the size of these spaces to maintain the non-entangled representation. We also used the Linear programming to work out numerically the growth factors to see whether any of these state spaces may grow slower than the cylinder. It turned out that the minimum value of growth factor for the considered state spaces is 2.145 which is bigger than the cylinder growth factor.

Therefore, the cylinder that requires the least growth, is the most optimal state space among all the state spaces that we studied. In the following chapter, we will try to increase the region that can be efficiently simulated classically, using the method of the coarse graining.

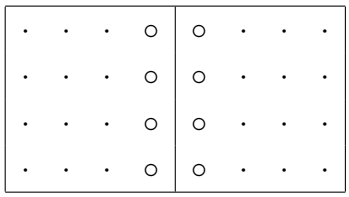
Chapter 4

Coarse Graining

When a CZ gate is applied on two cylindrical state spaces, we must grow the radius r in order to describe the system with separable decomposition. However, r cannot grow too much, i.e. r must not grow beyond 1, because the separable decomposition must use state spaces that are in the dual of the cylinder measurements. In this chapter, we are going to introduce a method where we put particles into blocks and these blocks are treated as single systems. This method is called coarse graining. So, we are going to try to use coarse graining to increase the size of the region that we can simulate efficiently, and we will find that this enables us to improve the value of the radius r by putting two particles and then four particles in a block.

4.1 Description of coarse graining

We start by taking a lattice similar to the cluster state and we divide it into blocks. Each block contains vertices that are connected by several CZ gates. The CZ gates that are within the block, are defined as "internal CZ gates". We also connect vertices from different blocks by CZ gates that we called "external CZ gate". For instance, we are considering a 4×4 lattice in each block:



In this diagram, the dots and the circles within each block are related by internal CZ gates. The circles in the first block are connected to those in the second block by the external CZ gates.

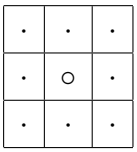
In each block that is considered as a single particle, we are going to grow the radii

of the cylindrical state spaces according to the applied external CZ gates to maintain separable decomposition. By taking the previous example, where we considered a 4×4 lattice in each block, $Cyl(r)$ are represented by the dots and $Cyl(r\lambda)$ are represented by the circles where $\lambda = \sqrt{2+\sqrt{5}} \simeq 2.05817$, which was found in the previous chapter, then $r \leq \frac{1}{\lambda}$.

Then the operators, that are within the block, are created from these cylindrical state spaces $Cyl(r\lambda^i)$, where i is the number of the external CZ gates, and the internal CZ gates that are applied on these state spaces. The set of these operators is defined as "block state space".

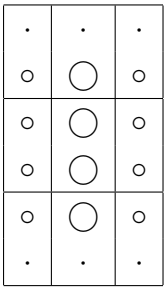
By growing the radii of the state spaces, we construct separable decomposition before applying the internal CZ gates. Now, we work out whether the separable decomposition has positive outcomes for allowed measurements.

We start by taking one particle in the block state space, it means that the internal CZ gate cannot be applied, only four external CZ gates are applied to this cylindrical state space.



$Cyl(r\lambda^4)$ are given by the circle and the dots may grow only by λ or they don't grow. But we won't consider cylinder presented by dots because if $Cyl(r\lambda^4)$ satisfies the dual constraint then the other cylinders that have r or $r\lambda$ will be in the dual. By taking the allowed measurements, we compute the trace that should be positive. Then $r\lambda^4$ must be less than 1 which means that $r \leq \frac{1}{r\lambda^4} \approx 0.0557$.

Now we consider the case where a block consists of two particles. Then, we have only one internal gate and three external gates.



$Cyl(r\lambda^3)$ are given by the big circles and the small circles may grow only by λ or λ^2 and the dots represent $Cyl(r)$. As mentioned above, we won't consider cylinders that are presented by dots and small circles.

All the extremal inputs of a cylindrical state space have the following form:

$$(1, r_{2cylinders}\cos\theta, r_{2cylinders}\sin\theta, \pm 1)$$

where $r_{2cylinders}$ is defined by $r_{2cylinders} := r\lambda^3$ and θ varies between 0 and 2π . We are going to work out the maximum value $r_{2cylinders}$ to see whether the separable decomposition has positive measurement outcomes.

Instead of taking all the possible inputs, we use the following to reduce the number of inputs:

- If we apply CZ on two Z measurements, there is no negativity because we end up with the same operators. However, if the CZ gate acts on a Z and an $X-Y$ measurements, we end up with a rotation of the $X-Y$ measurement. Then the product measurement on the two particles does not affect the positivity of the outcomes. Explicitly, we assume that $r_{2cylinders}$ should not grow beyond 1 in order to satisfy the dual constraint. Therefore we do not need to consider the Z measurement.
- We consider that y components of $Cyl(r_{2cylinders})$ are equal to 0. Hence, if we consider $Cyl(r_{2cylinders})$ with $z = -1$, we find that it is the same as applying an X rotation on the first particle with $z = 1$ and Z rotation on the second particle.
- CZ commutes with the Z rotation. If we take the projectors of $X-Y$ plane measurements and we apply the Z rotation, we end up with projectors that are in a different direction. Instead of taking $Cyl(r_{2cylinders})$ with component $y = 0$, we are going to consider the measurement projector $\left(\frac{I-X}{2}\right)$.

Now, we need to compute the $\mathrm{tr}(\rho \mathrm{P}) \geq 0$, where

$$\rho = CZ(1, r_{2cylinders}\cos\theta_A X, r_{2cylinders}\sin\theta_A Y, Z)\otimes(1, r_{2cylinders}\cos\theta_B X, r_{2cylinders}\sin\theta_B Y, Z),$$

where

$$z = 1 \text{ and } P = \left(\frac{I-X}{2}\right)^{\otimes 2}.$$

So we end up with the following equation,

$$\mathrm{tr}(\rho \mathrm{P}) = \mathrm{r}^2_{2\mathrm{cylinders}}\sin\theta_{\mathrm{A}}\sin\theta_{\mathrm{B}} - \mathrm{r}_{2\mathrm{cylinders}}(\cos\theta_{\mathrm{A}} + \cos\theta_{\mathrm{B}}) + 1 \geq 0.$$

Also, it can be written as

$$-r_{2cylinders}^2 \cos(\theta_A - \theta_B) + (1 - r_{2cylinders}\cos\theta_A)(1 - r_{2cylinders}\cos\theta_B) \geq 0$$

By taking $\theta_A = \theta_B = 0$, we have $r_{2cylinders} \leq \frac{1}{2}$. Therefore, all the allowed measurements and all the inputs have positive outcomes if and only if $r_{2cylinders} \leq \frac{1}{2}$.

By taking $r_{2cylinders} = r\lambda^3$, then r will be less than or equal to $\frac{1}{2\lambda^3} \simeq 0.0573$.

We find that r of the block that contains only one particle is less than r of the block of two particles, $\frac{1}{\lambda^4} \simeq 0.0557 \leq \frac{1}{2\lambda^3} \simeq 0.0573$. Therefore, the value of r increases as long as it satisfies the dual constraints.

We will show under certain assumptions which we believe to be true how we compute the maximum value of r for a block state space that contains only four state spaces. These assumptions are:

1. the measurement projector is $\left(\frac{I-X}{2}\right)$,
2. instead of considering all possible inputs, we take the input $(I, rX, 0Y, 1Z)$

4.2 Block state space of four cylindrical state spaces

In this section, we are going to combine four cylindrical state spaces into the block state space and apply four internal CZ gates, to work out the maximum radius of the block r_{pos}, such that the block state space belongs to the dual of cylinder measurement. We will find this maximum value by computing the trace(ρP), where ρ is the set of state spaces and P is the set of the allowed measurements.

To work out r_{pos}, we assume that the Bloch vector for each particle is $(r, 0, 1)$ and all the angles are equal to zero. Then, without applying CZ, $\rho^{(1,2,3,4)}$ is equal to:

$$\rho^{(1,2,3,4)} = \frac{1}{2}(I + rX + Z) \otimes \frac{1}{2}(I + rX + Z) \otimes \frac{1}{2}(I + rX + Z) \otimes \frac{1}{2}(I + rX + Z) \quad (4.1)$$

After applying the CZ_1 on the first and the second qubits, CZ_2 on the second and third qubits, CZ_3 on the third and the fourth qubits and finally CZ_4 on the fourth and first qubits, we obtained the following $\rho^{(1,2,3,4)'}$:

$$\rho^{(1,2,3,4)'} = \frac{1}{16}[I\otimes I\otimes I\otimes I + rZ\otimes I\otimes Z\otimes X + I\otimes I\otimes I\otimes Z + rI\otimes Z\otimes X\otimes Z + r^2 Z\otimes Z\otimes$$

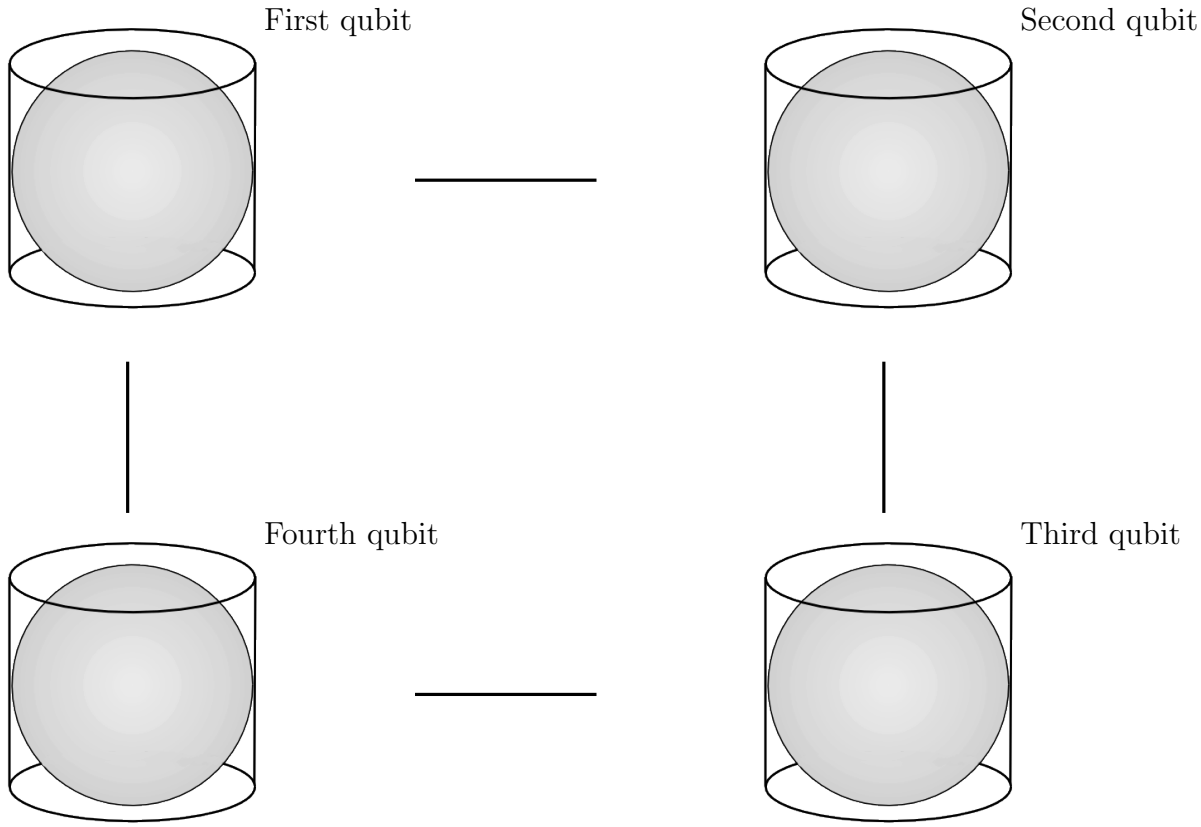


Figure 4.1: The block state space that consists of four cylinder state spaces with four CZ gates

$Y\otimes Y+rI\otimes Z\otimes X\otimes I+I\otimes I\otimes Z\otimes I+rZ\otimes I\otimes I\otimes X+I\otimes I\otimes Z\otimes Z+rZ\otimes X\otimes Z\otimes I+r^2I\otimes X\otimes I\otimes X+rZ\otimes X\otimes Z\otimes Z+r^2Z\otimes Y\otimes Y\otimes Z+r^3I\otimes Y\otimes X\otimes Y+r^2Z\otimes Y\otimes Y\otimes I+rZ\otimes X\otimes I\otimes I+r^2I\otimes X\otimes Z\otimes X+rZ\otimes X\otimes I\otimes Z+I\otimes Z\otimes I\otimes I+rZ\otimes Z\otimes Z\otimes X+I\otimes Z\otimes I\otimes Z+rI\otimes I\otimes X\otimes Z+r^2Z\otimes I\otimes Y\otimes Y+rI\otimes I\otimes X\otimes I+I\otimes Z\otimes Z\otimes I+rZ\otimes Z\otimes I\otimes X+I\otimes Z\otimes Z\otimes Z+rX\otimes Z\otimes I\otimes Z+r^2Y\otimes Z\otimes Z\otimes Y+rX\otimes Z\otimes I\otimes I+r^2X\otimes I\otimes X\otimes I+r^3Y\otimes I\otimes Y\otimes X+r^2X\otimes I\otimes X\otimes Z+rX\otimes Z\otimes Z\otimes Z+r^2Y\otimes Z\otimes I\otimes Y+rX\otimes Z\otimes Z\otimes I+r^2Y\otimes Y\otimes Z\otimes Z+r^3X\otimes Y\otimes I\otimes Y+r^2Y\otimes Y\otimes Z\otimes I+r^3Y\otimes X\otimes Y\otimes I+r^4X\otimes X\otimes X\otimes X+r^3Y\otimes X\otimes Y\otimes Z+r^2Y\otimes Y\otimes I\otimes Z+r^3X\otimes Y\otimes Z\otimes Y+r^2Y\otimes Y\otimes I\otimes I+rX\otimes I\otimes I\otimes Z+r^2Y\otimes I\otimes Z\otimes Y+rX\otimes I\otimes I\otimes I+r^2X\otimes Z\otimes X\otimes I+r^3Y\otimes Z\otimes Y\otimes X+r^2X\otimes Z\otimes X\otimes Z+rX\otimes I\otimes Z\otimes Z+r^2Y\otimes I\otimes I\otimes Y+rX\otimes I\otimes Z\otimes I+Z\otimes I\otimes I\otimes I+rI\otimes I\otimes Z\otimes X+Z\otimes I\otimes I\otimes Z+rZ\otimes Z\otimes X\otimes Z+r^2I\otimes Z\otimes Y\otimes Y+rZ\otimes Z\otimes X\otimes I+Z\otimes I\otimes Z\otimes I+rI\otimes I\otimes I\otimes X+Z\otimes I\otimes Z\otimes Z+rI\otimes X\otimes Z\otimes I+r^2Z\otimes X\otimes I\otimes X+rI\otimes X\otimes Z\otimes Z+r^2I\otimes Y\otimes Y\otimes Z+r^3Z\otimes Y\otimes X\otimes Y+r^2I\otimes Y\otimes Y\otimes I+rI\otimes X\otimes I\otimes I+r^2Z\otimes X\otimes Z\otimes X+rI\otimes X\otimes I\otimes Z+Z\otimes Z\otimes I\otimes I+rI\otimes Z\otimes Z\otimes X+Z\otimes Z\otimes I\otimes Z+rZ\otimes I\otimes X\otimes Z+r^2I\otimes I\otimes Y\otimes Y+rZ\otimes I\otimes X\otimes I+Z\otimes Z\otimes Z\otimes I+rI\otimes Z\otimes I\otimes X+Z\otimes Z\otimes Z\otimes Z]$.

r_{pos} can be calculated by studying the trace, that should be positive, of $\rho^{(1,2,3,4)'}$

and the measurement projector P:

$$tr(\rho^{(1,2,3,4)'}P) \geq 0 \tag{4.2}$$

where

$$P = [\frac{1}{2}(I - X)]^{\otimes 4} \tag{4.3}$$

the computation of trace is given by

$$tr(\rho^{(1,2,3,4)'}P) = \frac{1}{16}(r_{pos}^4 + 2r_{pos}^2 - 4r_{pos} + 1) \geq 0 \tag{4.4}$$

The maximum value of r_{pos} satisfying the equation of trace is:

$$r_{pos} \leq 0.295597743 \simeq 0.2956$$

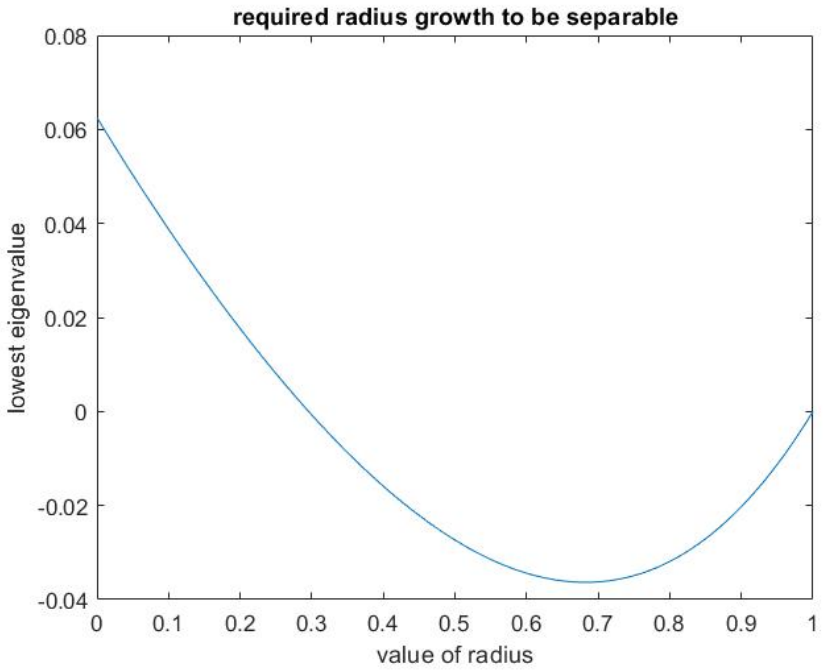


Figure 4.2: maximum radius required in the block state space to be separable.

In this block, we can only apply two external CZ gates which means that r will be taken to $r\lambda^2$ to maintain the separable decomposition. Hence, $r\lambda^2 \leq 0.2956$, then r is given by

$$r \leq \frac{0.2956}{\lambda^2} = \frac{0.2956}{2.05817^2} \approx 0.0698.$$

The value 0.0698 is computed based on the assumptions that the negativity will first appear for measurement $\frac{(I-X)}{2}$.

It turns out that when we add more vertices in the block state space, the value of r increases as long as it satisfies the dual constraints to maintain separability [7].

4.3 Block state space of many cylindrical state spaces

This section is a review of further progress which was made by my co-authors and described in article [7]. We are going to increase the size of the block state space and we will obtain two sequences (upper and lower) of optimization problems. The lower sequence is non-decreasing while the upper sequence is non-increasing. The lower sequence represents the values of the input radius where the system can be efficiently simulated classically and the upper sequence represents the input radius where the system gives negative probabilities. In the case of $2D$ lattice, these two sequences converge to limits that are close but we do not know if both sequences have the same limits, i.e. if they have the same limit then below this specific value the system can be efficiently simulated classically and above it the system has negative probabilities.

We consider a block B that has $H \times W$ qubits, where H and W are greater than 2. To initialise the qubits, we consider two ways:

1. All cylinders have the same state $(1, r, 0, 1)$. After applying internal CZ,the resulting operator is denoted by $\rho(B, r)$.

2. All cylinders that are in the corner of the block, are prepared in a state with $\lambda^2 r$ and the boundary cylinders are in a state with λr, while the remaining cylinders are in a state with input radius r. All these states have $z = 1$. After applying the internal CZ, the resulting operator is denoted by $\rho_\lambda(B, r)$.

We will also define the following:

$$s(B) := max\{r|\rho(B, r) \geq_M 0\}$$

$$s_\lambda(B) := max\{r|\rho_\lambda(B, r) \geq_M 0\}$$

where $\rho(B, r) \geq_M 0$ and $\rho_\lambda(B, r) \geq_M 0$ means that these operators are in the dual of the set M of the allowed measurements.

We are interested in $s_\lambda(B)$ to implement the coarse graining scheme. If $r \leq s_\lambda(B)$ in a block B, then cylinders with radius r can be efficiently simulated classically because they lead to positive probabilities. But if $r \geq s(B)$ in a block B, then cylinders with radius r lead to negative probabilities which mean that we won't end up with separable decomposition. So, we will try to find blocks where $s_\lambda(B)$ is large.

Lemma 3. Let's consider a region KL of qubits in a lattice and we divide this region into two disjoint subregions K and L. By disjoints we mean that we remove all the CZ gates that relate these two regions. For any region F, we have

$$s(F) \geq s_\lambda(F) \tag{4.5}$$

Then for the region KL, we have:

$$s(KL) \leq \min\{s(K), s(L)\} \tag{4.6}$$

$$s_\lambda(KL) \geq \min\{s_\lambda(K), s_\lambda(L)\} \tag{4.7}$$

If we consider that the two regions are identical, $K = L$, then

$$s(LL) \leq s(L) \tag{4.8}$$

$$s_\lambda(LL) \geq s_\lambda(L) \tag{4.9}$$

This tells us that every time we join the subregions together, we increase the size of a region, then s will only decrease and s_λ will only increase.

Proof. To end up with equation (4.5), we start by obtaining $\rho(F, r)$ from $\rho_\lambda(F, r)$ by dephasing the edges of the corner of qubits, i.e. we need to remove λ^2 and λ. Hence, if $\rho_\lambda(F, r) \geq 0$ then $\rho(F, r) \geq 0$. When $\rho(F, r)$ becomes negative, the value of r has to be greater than the value of r when $\rho_\lambda(F, r)$ becomes negative. So, as dephasing maintains positivity, $s(F) \geq s_\lambda(F)$.

To end up with equation (4.6), we begin by considering a cylinder with $z = 1$ that has the form $|0\rangle\langle 0| + a|0\rangle\langle 1| + b|1\rangle\langle 0|$ where $a, b \in \mathbb{C}$. Now suppose we interact this cylinder with two other particles in an arbitrary state T using two CZ. Then we have

$$|0\rangle\langle 0| \otimes T + a|0\rangle\langle 1| \otimes (TZ^{\otimes 2}) + b|1\rangle\langle 0| \otimes (Z^{\otimes 2}T)$$

If we trace out the first particle, then the remaining particles are in their original marginal state T. So, cutting the external CZ gates that join the two regions K and L, does not change the marginal operators on K or L, $tr_K\{\rho(KL, r)\} = \rho(L, r)$. If...$\rho(KL, r) \geq_M 0$ then we have $tr_K\{\rho(KL, r)\} \geq_M 0$, but as $tr_K\{\rho(KL, r)\} = \rho(L, r)$ this means that $\rho(KL, r) \geq_M 0$ implies $\rho(L, r) \geq_M 0$ and $\rho(K, r) \geq_M 0$. So by taking the minimum between $s(L)$ and $s(K)$, $s(KL)$ must be less than or equal to the minimum to have $\rho(KL, r) \geq_M 0$.

To end up with equation (4.7), we have $\rho_{lambda}(KL, r)$ is in the convex hull of

$\rho_\lambda(K,r) \otimes \rho_\lambda(L,r)$. Hence $\rho_\lambda(KL,r)$ on the block must be positive if $\rho_\lambda(K,r) \geq_M 0$ and $\rho_\lambda(L,r) \geq_M 0$ and so the equation $s_\lambda(KL) \geq \min\{s_\lambda(K), s_\lambda(L)\}$ must hold. $\square$

This lemma allows us to define sequences that capture when r is classically simulated classically using the coarse graining approach. Let's consider that we construct a sequence of blocks and we start with a 2×2 block B_1 and then construct larger blocks by joining two copies of B_{n-1} to make B_n. We define the sequences as follows

$$u_n := s(B_n) \tag{4.10}$$

$$l_n := s_\lambda(B_n) \tag{4.11}$$

From equation (4.5), we have $u_n \geq l_n$ where l_n is non-decreasing and u_n is non-increasing and hence both sequences converge. We denote the limits as:

$$u := \lim u_n$$

$$l := \lim l_n$$

If $r < l$ then r is classically simulatable efficiently but if $r > u$ then we end up with negative probabilities. If it turns out that $u = l$ then for $r < l$ the system is classically simulated and for $r > l$ the system has negative probabilities. But we do not know whether or not u is equal to l for any system. For a $2D$ lattice with the interaction of CZ gates, we calculated the lower and the upper bounds using trial measurements and inputs on rectangles of size 6×7. These numbers indicates that $0.0698 \leq l \leq u \leq 0.139$ for a $2D$ square lattice. For this specific lattice, the numerical experiments seem to suggest these two assumptions: the upper and lower sequences are determined by taking a measurement projector that has the form $\frac{I-X}{2}$ and the input that has the form $(1, \alpha, 0, 1)$ in the Pauli basis where α includes r and any growth factors λ applied in the coarse graining approach. When these inputs and measurements give positive probabilities, the maximum r appears to be the maximum in the following equations $s(B) = \max\{r|\rho(B,r) \geq_M 0\}$.

4.4 Summary of chapter 4

By using the method of coarse graining, we tried to expand the size of the region of the system that can be efficiently simulated classically. It turns out that the value of the radius increases slightly for two particles. But for four particles in a block, the system can be efficiently classically simulated under the following two assumptions:

the measurement projector is $\left(\frac{I-X}{2}\right)^{\otimes 4}$ and the input is $(1, r, 0, 1)$ in the Pauli basis. We use this measurement projector because it seems for numerical experiments that the negativity appears first for this measurement, i.e. if, by applying this measurement, we end up with positivity then the remaining allowed measurements will give positive values. In [7], we found that by adding more vertices in a block, the value of the radius increases, however it doesn't exceed 1.

Chapter 5

Summary

It is unknown if quantum computers are better than classical computers. We can proceed one of two ways to understand this problem. The first way is that we can try to find that quantum computers cannot be efficiently simulated classically and the second is that we can try to develop an algorithm that can classically simulate a quantum system.

It is believed that quantum computers, built with quantum system, can efficiently solve hard problems that are unknown whether they can be efficiently solved on classical computers. If it is found that simulating a quantum computing can be done classically, then any algorithm that can be done in quantum computers can be efficiently simulated on a classical computer.

In this thesis, our goal is to develop a new method of classically simulating certain types of quantum system that are previously unknown to be efficiently simulatable on classical computers.

We started by defining cylinder separability which is a new generalised notion of separability. Then we applied this notion to systems where we vary the inputs of cluster state quantum computation and we end up with some inputs that allow the system to be efficiently simulated classically. As we mentioned previously, the cluster state has input states $|\psi\rangle = |+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$, but we have shown that if we change $|\psi\rangle$ into the unormalised form $|0\rangle + \epsilon|1\rangle$ where ϵ is a small non-zero number, the system can be efficiently simulated classically.

The classical simulation is achieved by demonstrating that if we take CZ gate acting on two inputs that are presented by two cylindrical state spaces with radii r, the output is separable with respect to two different cylindrical state spaces with larger radii $R = \lambda r$ where $\lambda = \sqrt{2+\sqrt{5}}$. We can maintain a separable decomposition by growing the cylinder by λ each time but we must not grow beyond radius 1, because this state space should be in the dual of the cylinder measurements. This work forms the basis of the classical simulation.

We also considered, instead of CZ gates, diagonal gates $V(\theta)$ that acts on two cylindrical state spaces. Our goal is to try to find the minimum growth required for separability. After working out the required growth factor as a function of θ for all the diagonal gates, we discover that the CZ gate is the gate that requires the most growth factor λ to maintain separability. This work can be taken further. Instead of using $V(\theta)$ gates of two qubits, we could try to use different diagonal gates such as $|0\rangle\langle 0| \otimes z_\theta + |1\rangle\langle 1| \otimes z_\theta$ where $z_\theta = |0\rangle\langle 0| + e^{i\theta}|1\rangle\langle 1|$, or we could try other gates with number of qudits to see whether or not it is possible to efficiently simulated on a classical computer.

Then, we tried to find if there are other state spaces require a lower growth factor. So, we considered new state spaces that are the convex hull of $N(5-2)+2$ vertices. We chose these state spaces because of the symmetry about the z axis and they are among the simplest we could try for small number of extremal points. Then we grew the size of these state spaces to maintain separability and we used Linear programming to compute the required growth factors to see if any of the state spaces that we studied can grow slower than the cylinder. But it turns out that, among all these state spaces that we tried, the cylinder is the most optimal state space that needs the least growth. In further work, we could try different state spaces rather than the ones we studied but these state spaces have to be also symmetric about the z axis.

Finally, we introduce the method of coarse graining and we try to improve the value of the radius of a system that can be efficiently simulated on a classical computer using this specific method. Then we find that the radius that can be simulated efficiently, increases slightly for two particles and then even further subject to the following two assumptions for four particles in a block: the measurement projector has the form $\frac{I-X}{2}$ and instead of taking all the possible inputs, we consider the input $(I, rX, 0Y, Z)$. If we end up with positivity using this measurement projector then the remaining allowed measurements will give positive values. We will also find that by adding more vertices in a block, the value of the radius will increase slightly [7]. Two open problems are whether the upper and lower sequences converge and whether there are other ways of coarse graining that can improve the classical simulation. These open problems need further investigations.

Appendix A

Representation of V(θ) in the Pauli basis

In this part, we will compute the representation of $V(\theta)$ in Pauli basis that we called $VV(\theta)$. It is a 16×16 matrix and is used in chapter 2.

First, we take two inputs in the following Pauli basis:

$$(I, X, Y, Z)$$

where we have:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}; Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} I \text{ and } Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Then, we act the Controlled gate $V(\theta)$ on these two qubits by taking one Pauli operator from the first and second input; where $V(\theta)$ and $V^\dagger(\theta)$ are represented as the following:

$$V(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\theta} \end{pmatrix} \text{ and } V^\dagger(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-i\theta} \end{pmatrix}$$

- Let's start by taking I from the first input and changing the Pauli operators of the second input.

$$
\begin{aligned}
V(\theta)(I \otimes I)V^{\dagger}(\theta) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\
&= I \otimes I.
\end{aligned}
$$

- $V(\theta)$ is acting on I and X where X is the second input considered.

$$
\begin{aligned}
V(\theta)(I \otimes X)V^{\dagger}(\theta) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix} \\
&= \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & e^{-i\theta} \\ 0 & 0 & e^{i\theta} & 0 \end{pmatrix} \\
&= (\frac{I+Z}{2}) \otimes X + (\frac{I-Z}{2}) \otimes (\cos\theta X + \sin\theta Y) \\
&= aI \otimes X + bI \otimes Y + cZ \otimes X + dZ \otimes Y
\end{aligned}
$$

- Let's take Y the second input.

$$
\begin{aligned}
V(\theta)(I \otimes Y)V^{\dagger}(\theta) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix} \\
&= \begin{pmatrix} 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \\ 0 & 0 & 0 & -ie^{-i\theta} \\ 0 & 0 & ie^{i\theta} & 0 \end{pmatrix} \\
&= (\frac{I+Z}{2}) \otimes Y + (\frac{I-Z}{2}) \otimes (\cos\theta Y - \sin\theta X) \\
&= aI \otimes Y + dI \otimes X + bZ \otimes X + cZ \otimes Y
\end{aligned}
$$

- The second input considered is Z.

$$
\begin{aligned}
V(\theta)(I \otimes Z)V^{\dagger}(\theta) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix} \\
&= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -e^{-i\theta}e^{i\theta} \end{pmatrix} \\
&= I \otimes Z
\end{aligned}
$$

Instead of I, we consider X as the first input and change the second input.

- The second input is I.

$$V(\theta)(X \otimes I)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-i\theta} \\ 1 & 0 & 0 & 0 \\ 0 & e^{i\theta} & 0 & 0 \end{pmatrix}$$

$$= (\frac{X+iY}{2}) \otimes (\frac{I+Z}{2} + \frac{I-Z}{2}e^{-i\theta}) + (\frac{X-iY}{2}) \otimes (\frac{I+Z}{2} + \frac{I-Z}{2}e^{i\theta})$$

$$= eX \otimes I + jX \otimes Z + pY \otimes I + qY \otimes Z$$

- X is considered as the second input.

$$V(\theta)(X \otimes X)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 0 & e^{-i\theta} \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ e^{i\theta} & 0 & 0 & 0 \end{pmatrix}$$

$$= \frac{X+iY}{2} \otimes (\frac{X-iY}{2} + \frac{X+iY}{2}e^{-i\theta}) + \frac{X-iY}{2} \otimes (\frac{X+iY}{2} + \frac{X-iY}{2}e^{i\theta})$$

$$= eX \otimes X + pX \otimes Y + pY \otimes X + jY \otimes Y$$

- The second input taken is Y.

$$V(\theta)(X\otimes Y)V^{\dagger}(\theta) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(i\theta)}\end{pmatrix}\begin{pmatrix}0&0&0&-i\\0&0&i&0\\0&-i&0&0\\i&0&0&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(-i\theta)}\end{pmatrix}$$

$$= \begin{pmatrix}0&0&0&-ie^{-i\theta}\\0&0&i&0\\0&-i&0&0\\ie^{i\theta}&0&0&0\end{pmatrix}$$

$$= \frac{X+iY}{2}\otimes(\frac{iX+Y}{2}+\frac{-iX+Y}{2}e^{-i\theta})+\frac{X-iY}{2}\otimes(\frac{-iX+Y}{2}+\frac{iX+Y}{2}e^{i\theta}$$

$$= qX\otimes X + eX\otimes Y + kY\otimes X + pY\otimes Y$$

- The last operator is Z which is considered for the second input.

$$V(\theta)(X\otimes Z)V^{\dagger}(\theta) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(i\theta)}\end{pmatrix}\begin{pmatrix}0&0&1&0\\0&0&0&-1\\1&0&0&0\\0&-1&0&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(-i\theta)}\end{pmatrix}$$

$$= \begin{pmatrix}0&0&1&0\\0&0&0&-e^{-i\theta}\\1&0&0&0\\0&-e^{i\theta}&0&0\end{pmatrix}$$

$$= (\frac{X+iY}{2})\otimes(\frac{I+Z}{2}-\frac{I-Z}{2}e^{-i\theta})+(\frac{X-iY}{2})\otimes(\frac{I+Z}{2}-\frac{I-Z}{2}e^{i\theta})$$

$$= jX\otimes I + qY\otimes I + eX\otimes Z + pY\otimes Z$$

The third operator is Y which is taken as the first unchangeable input.

- Let's start by taking I the second input.

$$V(\theta)(Y \otimes I)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & -i \\ i & 0 & 0 & 0 \\ 0 & i & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & -ie^{-i\theta} \\ i & 0 & 0 & 0 \\ 0 & ie^{i\theta} & 0 & 0 \end{pmatrix}$$

$$= qX \otimes I + pX \otimes Z + eY \otimes I + jY \otimes Z$$

- The second input is X.

$$V(\theta)(Y \otimes X)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 0 & -ie^{-i\theta} \\ 0 & 0 & -i & 0 \\ 0 & i & 0 & 0 \\ ie^{i\theta} & 0 & 0 & 0 \end{pmatrix}$$

$$= qX \otimes X + kX \otimes Y + eY \otimes X + pY \otimes Y$$

- Let's take Y the second input.

$$V(\theta)(Y \otimes Y)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & 0 & -e^{-i\theta} \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -e^{i\theta} & 0 & 0 & 0 \end{pmatrix}$$

$$= jX \otimes X + qX \otimes Y + qY \otimes X + eY \otimes Y$$

- Z is considered as the second input.

$$V(\theta)(Y \otimes Z)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & i \\ i & 0 & 0 & 0 \\ 0 & -i & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 0 & -i & 0 \\ 0 & 0 & 0 & ie^{-i\theta} \\ i & 0 & 0 & 0 \\ 0 & -ie^{i\theta} & 0 & 0 \end{pmatrix}$$

$$= pX \otimes I + qX \otimes Z + jY \otimes I + eY \otimes Z$$

And finally, $V(\theta)$ is acting on the last operator Z which is taken as the first input.

- The second input considered is I.

$$V(\theta)(Z \otimes I)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -e^{i\theta}e^{-i\theta} \end{pmatrix}$$

$$= Z \otimes I$$

- X is taken as the second input.

$$V(\theta)(Z \otimes X)V^{\dagger}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(i\theta)} \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{(-i\theta)} \end{pmatrix}$$

$$= \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -e^{-i\theta} \\ 0 & 0 & -e^{i\theta} & 0 \end{pmatrix}$$

$$= cI \otimes X + aZ \otimes X + dI \otimes Y + bZ \otimes Y$$

- The second changeable input is Y.

$$V(\theta)(Z\otimes Y)V^{\dagger}(\theta) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(i\theta)}\end{pmatrix}\begin{pmatrix}0&-i&0&0\\i&0&0&0\\0&0&0&i\\0&0&-i&0\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(-i\theta)}\end{pmatrix}$$
$$= \begin{pmatrix}0&-i&0&0\\i&0&0&0\\0&0&0&ie^{-i\theta}\\0&0&-ie^{i\theta}&0\end{pmatrix}$$
$$= cI\otimes Y + aZ\otimes Y + bI\otimes X + dZ\otimes X$$

- The last operator Z is taken as the second input.

$$V(\theta)(Z\otimes Z)V^{\dagger}(\theta) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(i\theta)}\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&-1&0&0\\0&0&-1&0\\0&0&0&1\end{pmatrix}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&e^{(-i\theta)}\end{pmatrix}$$
$$= \begin{pmatrix}1&0&0&0\\0&-1&0&0\\0&0&-1&0\\0&0&0&e^{i\theta}e^{-i\theta}\end{pmatrix}$$
$$= Z\otimes Z$$

After applying $V(\theta)$ to these inputs, we get the 16×16 matrix, that we called $VV(\theta)$.

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \mathrm{a} & \mathrm{b} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{c} & \mathrm{d} & 0 \\
0 & \mathrm{d} & \mathrm{a} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{b} & \mathrm{c} & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathrm{e} & 0 & 0 & \mathrm{j} & \mathrm{p} & 0 & 0 & \mathrm{q} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathrm{e} & \mathrm{p} & 0 & 0 & \mathrm{p} & \mathrm{j} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathrm{q} & \mathrm{e} & 0 & 0 & \mathrm{k} & \mathrm{p} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathrm{j} & 0 & 0 & \mathrm{e} & \mathrm{q} & 0 & 0 & \mathrm{p} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathrm{q} & 0 & 0 & \mathrm{p} & \mathrm{e} & 0 & 0 & \mathrm{j} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathrm{q} & \mathrm{k} & 0 & 0 & \mathrm{e} & \mathrm{p} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathrm{j} & \mathrm{q} & 0 & 0 & \mathrm{q} & \mathrm{e} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathrm{p} & 0 & 0 & \mathrm{q} & \mathrm{j} & 0 & 0 & \mathrm{e} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & \mathrm{c} & \mathrm{d} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{a} & \mathrm{b} & 0 \\
0 & \mathrm{b} & \mathrm{c} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathrm{d} & \mathrm{a} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{matrix}
(I \otimes I) \\ (I \otimes X) \\ (I \otimes Y) \\ (I \otimes Z) \\ (X \otimes I) \\ (X \otimes X) \\ (X \otimes Y) \\ (X \otimes Z) \\ (Y \otimes I) \\ (Y \otimes X) \\ (Y \otimes Y) \\ (Y \otimes Z) \\ (Z \otimes I) \\ (Z \otimes X) \\ (Z \otimes Y) \\ (Z \otimes Z)
\end{matrix}$$

where

$$a = \frac{1}{2} + \frac{1}{2}\cos\theta; b = \frac{1}{2}\sin\theta; c = \frac{1}{2} - \frac{1}{2}\cos\theta; d = -\frac{1}{2}\sin\theta;$$

$$e = \frac{1}{2} + \frac{1}{4}\exp(-i\theta) + \frac{1}{4}\exp(i\theta); j = \frac{1}{2} - \frac{1}{4}\exp(-i\theta) - \frac{1}{4}\exp(i\theta); k = -\frac{1}{2} + \frac{1}{4}\exp(-i\theta) + \frac{1}{4}\exp(i\theta);$$

$$p = \frac{1}{4}i\exp(-i\theta) - \frac{1}{4}i\exp(i\theta); q = -\frac{1}{4}i\exp(-i\theta) + \frac{1}{4}i\exp(i\theta)$$

Appendix B

Matlab code

B.1 Matlab code for Chapter 2

When the gate $V(\theta)$ acts on cylindrical state spaces in chapter 2, we end up with identical minimum eigenvalues. We check this result using Matlab code:

```
    clear
%for the two inputs (1,rA,0,1)tensor (1,rB,0,1)
% pauli matrices
i = complex(0,1);
I = [1 0; 0 1];
X = [0 1; 1 0];
Y = [0 -i; i 0];
Z = [1 0; 0 -1];
rA = 1;
rB = 1;

j = 1;
array_min = zeros(5,361*numel(0:0.1:1.5));

%cs = cosd(theta)+ i*sind(theta);
%cs_m = cosd(theta) - i* sind(theta);

II = kron(I,I);%II
IX = kron(I,X);%IX
IY = kron(I,Y);%IY
IZ = kron(I,Z);%IZ
XI = kron(X,I);%XI
XX = kron(X,X);%XX
XY = kron(X,Y);%XY
```

```
XZ = kron(X,Z);%XZ
YI = kron(Y,I);%YI
YX = kron(Y,X);%YX
YY = kron(Y,Y);%YY
YZ = kron(Y,Z);%YZ
ZI = kron(Z,I);%ZI
ZX = kron(Z,X);%ZX
ZY = kron(Z,Y);%ZY
ZZ = kron(Z,Z);%ZZ

for K = 0:0.1:1.5
%add 1 to RA: going from 1 to 2.5 by 0.1
RA = 1+K;
RB = 1+K;

for theta = 0:360

cs = cosd(theta)+ i*sind(theta);
cs_m = cosd(theta) - i* sind(theta);

%this matrix is similar to equation 8 in the paper, where all the Z components are 0:
matrix_1 = II + (rB/RB)*IX + (rA/RA)* XI +
((0.5+0.25*cs_m+0.25*cs)*((rA*rB)/(RA*RB)))* XX + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*XY + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*YX
+((0.5-0.25*cs_m-0.25*cs)*((rA*rB)/(RA*RB)))*YY ;

%this matrix is the partial transpose:
%3rd column is minus
matrix_2 = II + (rB/RB)*IX + (rA/RA)* XI +
((0.5+0.25*cs_m+0.25*cs)*((rA*rB)/(RA*RB)))* XX - ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*XY + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*YX
-((0.5-0.25*cs_m-0.25*cs)*((rA*rB)/(RA*RB)))*YY ;

%or 3rd row is minus
matrix_3 = II + (rB/RB)*IX + (rA/RA)* XI +
((0.5+0.25*cs_m+0.25*cs)*((rA*rB)/(RA*RB)))* XX + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*XY - ((-0.25*i*cs_m
```

```
+0.25*i*cs)*((rA*rB)/(RA*RB)))*YX
-((0.5-0.25*cs_m-0.25*cs)*((rA*rB)/(RA*RB)))*YY ;

%min eigenvalues of matrix_1

eigenvalues_1 = eig(matrix_1);
min_eig_1 = min(eigenvalues_1);

eigenvalues_2 = eig(matrix_2);
min_eig_2 = min(eigenvalues_2);

eigenvalues_3 = eig(matrix_3);
min_eig_3 = min(eigenvalues_3);

array_min(1,j) = theta;
array_min(2,j) = min_eig_1;
array_min(3,j) = min_eig_2;
array_min(4,j) = min_eig_3;
array_min(5,j) = RA;

j = j+1;

end

end
```

The graph (2.2) which represents the required growth for each theta, is given by the following code:

```
clear
%for the two inputs (1,rA,0,1)tensor (1,rB,0,1)
% pauli matrices
i = complex(0,1);
I = [1 0; 0 1];
X = [0 1; 1 0];
Y = [0 -i; i 0];
Z = [1 0; 0 -1];
rA = 1;
rB = 1;

j = 1;

array_R = zeros(1,numel(0:0.1:1.1 ));
array_theta = zeros(1,numel(0:0.1:1.1 ));

%cs = cosd(theta)+ i*sind(theta);
%cs_m = cosd(theta) - i* sind(theta);

II = kron(I,I);%II
IX = kron(I,X);%IX
IY = kron(I,Y);%IY
IZ = kron(I,Z);%IZ
XI = kron(X,I);%XI
XX = kron(X,X);%XX
XY = kron(X,Y);%XY
XZ = kron(X,Z);%XZ
YI = kron(Y,I);%YI
YX = kron(Y,X);%YX
YY = kron(Y,Y);%YY
YZ = kron(Y,Z);%YZ
ZI = kron(Z,I);%ZI
ZX = kron(Z,X);%ZX
ZY = kron(Z,Y);%ZY
ZZ = kron(Z,Z);%ZZ
```

```
for K = 0:0.00001:1.1
%add 1 to RA: going from 1 to 2.1 by 0.00001
RA = 1+K;
RB = 1+K;

for theta = 0:360

cs = cosd(theta)+ i*sind(theta);
cs_m = cosd(theta) - i* sind(theta);

%this matrix is similar to equation 8 in the paper, where all the Z components are 0:
matrix_1 = II + (rB/RB)*IX + (rA/RA)* XI +
((0.5+0.25*cs_m+0.25*cs)*((rA*rB)/(RA*RB)))* XX + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*XY + ((-0.25*i*cs_m
+0.25*i*cs)*((rA*rB)/(RA*RB)))*YX
+((0.5-0.25*cs_m-0.25*cs)*((rA*rB)/(RA*RB)))*YY ;

%min eigenvalues of matrix_1

eigenvalues_1 = eig(matrix_1);
min_eig_1 = min(eigenvalues_1);

if min_eig_1 < 0
    array_R(1,j) = RA;
    array_theta(1,j) = theta;
else
    continue
end



if min_eig_1 < 0
    break
else
    continue
end

end
j = j + 1;
end
```

```
plot(array_theta, array_R)
%title('Required growth to be separable')
xlabel('values of theta')
ylabel('values of growth')
```

B.2 Matlab code for Chapter 3

The Matlab code, which is shown below, is the same code applied to the section of *Different state spaces*. The difference is that we are changing values of "N", epsilon, heights and radii:

```
r = 1; %(radius of the input)
h = 1; %(height of the input)
M = 5; %(number of points going down the Z axis (with (1,0,0,1) and (1,0,0,-1)))
N = 15; %(number of angles going around the equator)
epsilon = 0.1;
theta = 360/N;
height = [1 0.75 0 -0.75 -1]; %(heights of the five inputs)
width = [0, (\sqrt(7)/4)*(1+epsilon), 1, (\sqrt(7)/4)*(1+epsilon), 0];  %(Radii of th


CZ = \begin{blockarray}{*{17}{c}}
\begin{block}{(*{16}{c})c}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($I\otimes I$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & ($I\otimes X$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & ($I\otimes Y$)
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($I\otimes Z$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($X\otimes I$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & ($X\otimes X$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & ($X\otimes Y$)
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($X\otimes Z$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & ($Y\otimes I$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($Y\otimes X$)
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($Y\otimes Y$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($Y\otimes Z$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & ($Z\otimes I$)
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($Z\otimes X$)
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & ($Z\otimes Y$)
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & ($Z\otimes Z$)
\end{block}
\end{blockarray}

output_shape = zeros(4,N(M-2)+2); % a (4\times N(M-2)+2) matrix
```

```
% Linear programming in Matlab: linprog(f, A, b, A', b')
C = zeros((N(M-2)+2)^2,1); % the objective function
I = - eye((N(M-2)+2)^2) ; % A inequality
O = zeros((N(M-2)+2)^2,1) ; % b inequality

for R = 2.4:0.001:2.46

% Output

init = 3;
for k = 2: M-1
for phi = 0:360/N:360-360/N
output_shape(:, init) = [1; R*width(k)*cos(phi); R*width(k)*sin(phi); height(k)];
init = init+1;
end
end

output_shape(:,1) = [1; 0; 0;1]; %((:,1) means rows of a specific col (in this case i
output_shape(:,2) = [1; 0; 0;-1];

out_shape = kron(output-shape,output-shape); %(out_shape is A equality in linear prog

% Input

number_R = 0;
optimal_R = 0;

for ll = 1:M
for mm = 1: M
input = kron([1; width(ll); 0; height(ll)],[1; width(mm); 0; height(mm)]);
input_CZ = CZ*input; %(input_CZ is b equality in linear programming)

Xopt = linprog(C, I, O, out_shape, input_CZ);
if Xopt >= 0
number_R = number_R + 1;
    if mm == M and ll == M and number_R == M^2
        optimal_R = R;
    end
```

```
else
        break
end
end

if Xopt >= 0
    continue
else
    break
end
end

if number_R == M^2
    break
else
    continue
end
end

% For the linear programming, the formula that we need to use is as follow: linprog(f
where
% C = zeros((N(M-2)+2)^2,1); % which is the objective function,
% I = - eye((N*(M-2)+2)^2); % which is the A inequality,
% O = zeros((N*(M-2)+2)^2,1); % which is the b inequality,
% The out_shape is the A equality,
% The input_CZ is b equality,
% lb=ub=0; that are the lower and the upper bound respectively.

% So by replacing the above formula by the equivalent notation, we have
Xopt = linprog(C, I, O, out-shape, input-CZ).
```

List of Figures

List of Tables

Bibliography

[1] Scott Aaronson and Daniel Gottesman. "Improved simulation of stabilizer circuits". In: *Physical Review A* 70.5 (2004), p. 052328.

[2] Scott Aaronson and Sam Gunn. "On the classical hardness of spoofing linear cross-entropy benchmarking". In: *arXiv preprint arXiv:1910.12085* (2019).

[3] Hussain Anwar et al. "Families of pure PEPS with efficiently simulatable local hidden variable models for most measurements". In: *arXiv preprint arXiv:1412.3780* (2014).

[4] Hussain Anwar et al. "Generalised versions of separable decompositions applicable to bipartite entangled quantum states". In: *New Journal of Physics* 21.9 (2019), p. 093031.

[5] Hussain Anwar et al. "Smallest state spaces for which bipartite entangled quantum states are separable". In: *New Journal of Physics* 17.9 (2015), p. 093047.

[6] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (2019), pp. 505–510.

[7] Sahar Atallah et al. "Efficient classical simulation of cluster state quantum circuits with alternative inputs". In: *arXiv preprint arXiv:2201.07655* (2022).

[8] Remigiusz Augusiak, Maciej Demianowicz, and Antonio Acın. "Local hidden–variable models for entangled quantum states". In: *Journal of Physics A: Mathematical and Theoretical* 47.42 (2014), p. 424002.

[9] Boaz Barak, Chi-Ning Chou, and Xun Gao. "Spoofing linear cross-entropy benchmarking in shallow quantum circuits". In: *arXiv preprint arXiv:2005.02421* (2020).

[10] Jonathan Barrett. "Information processing in generalized probabilistic theories". In: *Physical Review A* 75.3 (2007), p. 032304.

[11] Jonathan Barrett et al. "The computational landscape of general physical theories". In: *npj Quantum Information* 5.1 (2019), pp. 1–10.

[12] Sean D Barrett et al. "Transitions in the computational power of thermal states for measurement-based quantum computation". In: *Physical Review A* 80.6 (2009), p. 062328.

[13] Stephen D Bartlett and Barry C Sanders. "Efficient classical simulation of optical quantum information circuits". In: *Physical review letters* 89.20 (2002), p. 207903.

[14] Stephen D Bartlett et al. "Efficient classical simulation of continuous variable quantum information processes". In: *Physical Review Letters* 88.9 (2002), p. 097904.

[15] John S Bell. "On the einstein podolsky rosen paradox". In: *Physics Physique Fizika* 1.3 (1964), p. 195.

[16] Jacob Biamonte and Ville Bergholm. "Tensor networks in a nutshell". In: *arXiv preprint arXiv:1708.00006* (2017).

[17] Sergio Boixo et al. "Characterizing quantum supremacy in near-term devices". In: *Nature Physics* 14.6 (2018), pp. 595–600.

[18] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[19] Sergey Bravyi and David Gosset. "Improved classical simulation of quantum circuits dominated by Clifford gates". In: *Physical review letters* 116.25 (2016), p. 250501.

[20] Daniel J Brod. "Efficient classical simulation of matchgate circuits with generalized inputs and measurements". In: *Physical Review A* 93.6 (2016), p. 062332.

[21] Nicolas Brunner et al. "Bell nonlocality". In: *Reviews of Modern Physics* 86.2 (2014), p. 419.

[22] Andrew M Childs. "Lecture notes on quantum algorithms". In: *Lecture notes at University of Maryland* (2017).

[23] Andrew M Childs and Wim Van Dam. "Quantum algorithms for algebraic problems". In: *Reviews of Modern Physics* 82.1 (2010), p. 1.

[24] John F Clauser et al. "Proposed experiment to test local hidden-variable theories". In: *Physical review letters* 23.15 (1969), p. 880.

[25] Andrew C Doherty, Pablo A Parrilo, and Federico M Spedalieri. "Distinguishing separable and entangled states". In: *Physical Review Letters* 88.18 (2002), p. 187904.

[26] Albert Einstein, Boris Podolsky, and Nathan Rosen. "Can quantum-mechanical description of physical reality be considered complete?" In: *Physical review* 47.10 (1935), p. 777.

[27] Daniel Gottesman. "The Heisenberg representation of quantum computers". In: *arXiv preprint quant-ph/9807006* (1998).

[28] D Gross and J Eisert. "Novel schemes for measurement-based quantum computation". In: *Physical review letters* 98.22 (2007), p. 220503.

[29] Daivd Gross et al. "Measurement-based quantum computation beyond the one-way model". In: *Physical Review A* 76.5 (2007), p. 052315.

[30] Lov K Grover. "A fast quantum mechanical algorithm for database search". In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 1996, pp. 212–219.

[31] Lucien Hardy. "Quantum theory from five reasonable axioms". In: *arXiv preprint quant-ph/0101012* (2001).

[32] Aram W Harrow and Ashley Montanaro. "Quantum computational supremacy". In: *Nature* 549.7671 (2017), pp. 203–209.

[33] Aram W Harrow and Michael A Nielsen. "Robustness of quantum gates in the presence of noise". In: *Physical Review A* 68.1 (2003), p. 012308.

[34] Michał Horodecki, Paweł Horodecki, and Ryszard Horodecki. "Separability of mixed states: necessary and sufficient conditions". In: *Physics Letters A* 223.1 (1996), pp. 1–8. ISSN: 0375-9601. DOI: https://doi.org/10.1016/S0375-9601(96)00706-2. URL: https://www.sciencedirect.com/science/article/pii/S0375960196007062.

[35] Pawel Horodecki. "Separability criterion and inseparable mixed states with positive partial transposition". In: *Physics Letters A* 232.5 (1997), pp. 333–339.

[36] Peter Janotta and Haye Hinrichsen. "Generalized probability theories: what determines the structure of quantum theory?" In: *Journal of Physics A: Mathematical and Theoretical* 47.32 (2014), p. 323001.

[37] Richard Jozsa. "On the simulation of quantum circuits". In: *arXiv preprint quant-ph/0603163* (2006).

[38] Richard Jozsa and Noah Linden. "On the role of entanglement in quantum-computational speed-up". In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 459.2036 (2003), pp. 2011–2032.

[39] Richard Jozsa and Akimasa Miyake. "Matchgates and classical simulation of quantum circuits". In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 464.2100 (2008), pp. 3089–3106.

[40] Richard Jozsa and Maarten Van den Nest. "Classical simulation complexity of extended Clifford circuits". In: *arXiv preprint arXiv:1305.6190* (2013).

[41] Igor L Markov and Yaoyun Shi. "Simulating quantum computation by contracting tensor networks". In: *SIAM Journal on Computing* 38.3 (2008), pp. 963–981.

[42] Ashley Montanaro. "Quantum algorithms: an overview". In: *npj Quantum Information* 2.1 (2016), pp. 1–8.

[43] Caterina E Mora et al. "Universal resources for approximate and stochastic measurement-based quantum computation". In: *Physical Review A* 81.4 (2010), p. 042315.

[44] Michele Mosca. "Quantum algorithms". In: *arXiv preprint arXiv:0808.0369* (2008).

[45] Michael A Nielsen. "Cluster-state quantum computation". In: *Reports on Mathematical Physics* 57.1 (2006), pp. 147–161.

[46] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. 2002.

[47] Asher Peres. "Separability criterion for density matrices". In: *Physical Review Letters* 77.8 (1996), p. 1413.

[48] David Perez-Garcia et al. “Matrix product state representations”. In: *arXiv preprint quant-ph/0608197* (2006).

[49] Martin Plávala. “General probabilistic theories: An introduction”. In: *arXiv preprint arXiv:2103.07469* (2021).

[50] John Preskill. “Lecture notes for physics 229: Quantum information and computation”. In: *California Institute of Technology* 16.1 (1998), pp. 1–8.

[51] N Ratanje and S Virmani. “Exploiting non-quantum entanglement to widen applicability of limited-entanglement classical simulations of quantum systems”. In: *arXiv preprint arXiv:1201.0613* (2012).

[52] N Ratanje and Shashank Virmani. “Generalized state spaces and nonlocality in fault-tolerant quantum-computing schemes”. In: *Physical Review A* 83.3 (2011), p. 032309.

[53] Robert Raussendorf and Hans J Briegel. “A one-way quantum computer”. In: *Physical Review Letters* 86.22 (2001), p. 5188.

[54] Robert Raussendorf, Daniel E Browne, and Hans J Briegel. “Measurement-based quantum computation on cluster states”. In: *Physical review A* 68.2 (2003), p. 022312.

[55] Norbert Schuch et al. “Computational complexity of projected entangled pair states”. In: *Physical review letters* 98.14 (2007), p. 140506.

[56] Peter W Shor. “Introduction to quantum algorithms”. In: *Proceedings of Symposia in Applied Mathematics*. Vol. 58. 2002, pp. 143–160.

[57] Peter W Shor. “Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer”. In: *SIAM review* 41.2 (1999), pp. 303–332.

[58] Barbara M Terhal and David P DiVincenzo. “Classical simulation of noninteracting-fermion quantum circuits”. In: *Physical Review A* 65.3 (2002), p. 032325.

[59] Leslie G Valiant. “Expressiveness of matchgates”. In: *Theoretical Computer Science* 289.1 (2002), pp. 457–471.

[60] Leslie G Valiant. “Quantum circuits that can be simulated classically in polynomial time”. In: *SIAM Journal on Computing* 31.4 (2002), pp. 1229–1254.

[61] Maarten Van den Nest. “Universal quantum computation with little entanglement”. In: *Physical review letters* 110.6 (2013), p. 060504.

[62] Guifré Vidal. “Efficient classical simulation of slightly entangled quantum computations”. In: *Physical review letters* 91.14 (2003), p. 147902.

[63] Reinhard F Werner. “Quantum states with Einstein-Podolsky-Rosen correlations admitting a hidden-variable model”. In: *Physical Review A* 40.8 (1989), p. 4277.

[64] Mithuna Yoganathan, Richard Jozsa, and Sergii Strelchuk. “Quantum advantage of unitary Clifford circuits with magic state inputs”. In: *Proceedings of the Royal Society A* 475.2225 (2019), p. 20180427.

[65] Nadav Yoran. “Efficiently contractable quantum circuits cannot produce much entanglement”. In: *arXiv preprint arXiv:0802.1156* (2008).

[66] Nadav Yoran and Anthony J Short. “Classical simulation of limited-width cluster-state quantum computation”. In: *Physical review letters* 96.17 (2006), p. 170503.