Alma Mater Studiorum · Università di Bologna

Dipartimento di Fisica e Astronomia "Augusto Righi"
Corso di Laurea in Fisica

Digital approach to quantum adiabatic algorithms

Relatore:
Prof. Lorenzo Piroli

Presentata da:
Jacopo Sceusi

Anno Accademico 2023/2024

Contents

Abstract

Negli ultimi trent'anni si sono registrati grandi progressi nel campo del calcolo quantistico. Un approccio promettente al calcolo quantistico consiste nel calcolo quantistico adiabatico, o quantum annealing, che si basa sul teorema adiabatico quantistico. Sebbene quest'ultimo sia progettato per dispositivi quantistici analogici specifici, si può considerare una versione digitale del quantum annealing che può essere implementata su dispositivi quantistici di uso generale. Questo è l'argomento di interesse in questa tesi. Dopo aver introdotto i concetti di base dell'informazione quantistica, esamineremo gli aspetti del calcolo quantistico adiabatico. Discuteremo la versione digitale del quantum annealing e stabiliremo un collegamento con il cosiddetto Quantum Approximate Optimization Algorithm (QAOA), uno schema variazionale progettato per risolvere problemi di ottimizzazione combinatoria su dispositivi quantistici digitali. Un focus particolare è dato alla semplificazione di operatori di evoluzione con ordinamento temporale o con Hamiltoniane contenenti termini non commutanti, e alle tecniche di ottimizzazione per risolvere problemi computazionali complessi. L'obiettivo è fornire una base teorica solida per comprendere l'implementazione pratica di tali algoritmi.

Introduction

The modern science of computation owes much to the groundbreaking work of Alan Turing. In 1936, Turing introduced the concept of what we now recognize as a programmable computer — the Universal Turing Machine — a theoretical device capable of simulating any computation process, laying the foundation for classical computer science. This notion was encapsulated in the Church-Turing thesis [1], which asserts that any algorithmic process executable on a physical machine can be simulated by a Turing machine.
Decades later, the traditional framework of computation reached its limits as miniaturization pushed classical devices to encounter fundamental quantum effects. At the same time, classical intrinsic inability of simulating quantum systems, due to the exponential growth in the required resources as system size increases, challenged the Church-Turing thesis. In 1982, Richard Feynman argued that if one wants to simulate a quantum system accurately, one should use a quantum device — a suggestion that sparked the birth of quantum computing [11]. Feynman's insight laid the foundation for what is now known as quantum simulation, a powerful idea with the potential to surpass classical computational methods particularly in efficiently solving certain complex problems. This limitation was made explicit with the introduction of Peter Shor's algorithm in 1994, which demonstrated that quantum computers could factorize large integers exponentially faster than the best-known classical algorithms, in a time that grows (roughly) as the square of the number of digits. Shor's work provided a concrete challenge to the strong Church-Turing thesis, suggesting that there exist problems efficiently solvable by quantum machines that are beyond the reach of classical counterparts, even when enhanced with randomness [29].
Nowadays, hundreds of problems are known to be hard on classical devices, ranging from the practical (variants of the *"Traveling Salesman problem"*, see [15, 30]) to the whimsical (a problem derived from the game of Go). Further, a various range of important scientific problems from polymer folding [7], [4], to memory [14], to collective decision making in economics and social sciences [6], [23] can be ultimately resumed in a theory of computational hard problems, that may find solution within the quantum computing

[1] A more general version of this thesis is the Strong Church-Turing Thesis (or Extended Church-Turing Thesis) which posits that any computational problem that can be solved efficiently (i.e., in polynomial time) by a physically realizable machine can also be solved efficiently by a probabilistic Turing machine using a classical algorithm.

framework.
Despite this theoretical promise, we currently live in what John Preskill termed the Noisy Intermediate-Scale Quantum (NISQ) era [28]. In this era, quantum devices have a limited number of qubits (generally under a thousand), and they are prone to significant noise and limited coherence time, making it challenging to run deep or precise quantum algorithms. However, the potential advantages of quantum devices in optimization, cryptography, and simulation have motivated the search for algorithms capable of working under these constraints and became a central focus in the 1990s, with the development of the Quantum Approximate Optimization Algorithm (QAOA) and Digital Quantum Annealing (DQA), which have emerged as leading candidates for solving optimization problems — problems that are notoriously difficult for classical algorithms.
Both the DQA and QAOA are inspired by Quantum Annealing (QA)[2], that was originally conceived as a form of analog quantum computation: it seeks to navigate complex optimization landscapes by gradually evolving a quantum system from an easily prepared initial state to a final state that encodes the solution [17]. This method exploit the adiabatic theorem, which states that if the evolution is slow enough, the system will remain in its ground state. The Hamiltonian of the system starts with a "driver" term, which is easy to prepare, and ends with a "problem" Hamiltonian, whose ground state encodes the solution to the problem of interest. By adjusting the interpolation between these Hamiltonians, QA can leverage quantum effects like tunneling to explore the energy landscape and escape local minima — a task that classical optimization algorithms often struggle with.
However, QA's analog nature introduces practical limitations due to hardware constraints and noise. This has led to the exploration of Digital Quantum Annealing (DQA), a discretized version of QA designed to be implemented on gate-based quantum computers. The digital adaptation retains the core principles of QA while offering error correction and enhanced control over the quantum evolution process [22]. This flexibility makes DQA a promising candidate for NISQ devices.
Around the same time that QA was gaining traction, a more flexible approach was proposed: the Quantum Approximate Optimization Algorithm (QAOA). Introduced by E. Farhi and collaborators in 2014, QAOA is a hybrid algorithm designed to leverage both quantum and classical resources [10]. The algorithm iteratively applies a sequence of parameterized quantum gates that encode problem constraints, followed by a classical optimization step to adjust the parameters. This iterative refinement allows QAOA to approximate the solution to an optimization problem, with the quality of the approximation improving as the number of iterations—or "depth"—increases. QAOA shares conceptual similarities with QA; both attempt to navigate complex energy landscapes

[2]QA was heavily influenced by ideas from statistical mechanics as it leverages simulated quantum fluctuations, instead of thermal fluctuations, and tunneling offering a quantum-inspired variant of simulated annealing (SA) [20]

to identify optimal solutions. However, while QA relies on the continuous-time evolution of a quantum system, QAOA uses discrete, controlled operations. This distinction makes QAOA particularly well-suited for NISQ devices because gate operations can be tailored to specific problem instances and hardware architectures and long coherence times isn't required. Furthermore, QAOA's modular structure lends itself to potential improvements through hybrid techniques, combining the strengths of quantum search with classical refinement.

While QA, DQA, and QAOA each offer promising paths toward achieving quantum advantage, they are not without challenges. The accuracy of QA is contingent on the system's ability to remain adiabatic throughout the evolution — a condition that becomes harder to meet as system size grows and hardware noise increases. In QAOA, finding optimal parameters for deep circuits remains an open question, with classical optimizers often getting stuck in local minima. Despite these obstacles, both approaches are valuable as they push the boundaries of what can be accomplished with current and near-term quantum hardware. Taking these challenges into account, the question of whether it is possible to find efficient algorithms for solving hard computational problems on NISQ devices remains unresolved.

In the first chapter of the thesis, we provide a brief explanation of the necessary background knowledge needed for a thorough grasp of quantum computation and information. In the second chapter, we focus on the theory of quantum-system evolution. We state the adiabatic theorem and discuss the adiabatic limit of the time evolution of a quantum system. In Chapter 3, we analyze the digital approach to adiabatic evolution and explore how to approximate it effectively using discrete steps through unitary operators, which correspond to the logic gates of a quantum computer. We examine the Digital Quantum Annealing (DQA) algorithm as a method for solving classical computational problems using quantum resources. In Chapter 4, we present the Quantum Approximate Optimization Algorithm (QAOA) as an alternative digital approach to DQA, leveraging hybrid resources (classical and quantum) to address computational challenges.

Chapter 1

Quantum mechanics background

Quantum mechanics is the most precise and comprehensive description of the physical world we have. It also forms the foundation for understanding quantum computation and quantum information. This chapter offers the essential background in quantum mechanics required for this thesis.

1.1 Linear Algebra and Dirac notation

Since quantum mechanics is the primary reason for our exploration of linear algebra, we will introduce its standard notation, the *Dirac notation*, when discussing linear algebraic concepts. Dirac notation is a **compact** and **flexible** tool when working with states, operators and observables in QM. It eliminates the need to specify a vector's coordinates each time, clearly revealing the structure of Hilbert spaces and it facilitates the calculation of probabilities, the use of operators, and the manipulation of superpositions and state transformations. Although in QM Dirac notation is more concise and powerful than the matrix notation, we will use both in this first part, emphasizing the parallels between them. We will abandon the traditional notation and use only the Dirac formalism once we have provided the necessary foundations for an effective understanding of it.

Vectors and vector spaces Linear algebra studies *vector spaces* and *linear* operations on vector spaces. We are most interested in $\mathbb{C}^n$, the space of all n-tuples $(z_1, ..., z_n), z_i \in \mathbb{C}$. The elements of a vector space are called *vectors*, and they may be viewed either as *column* vectors $\mathbf{z}$ or as *row vectors* $\mathbf{z}^{+1}$,

$$\mathbf{z} = \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} \quad \mathbf{z}^+ = \begin{bmatrix} z_1^* & \dots & z_n^* \end{bmatrix}$$

[1] $\mathbf{z}^+ = z^{*T}$ represent the transposed conjugate vector of $\mathbf{z}$. This operation sends a vector space V into its conjugate dual $\tilde{V}^*$

The Dirac counterpart of a column (row) vector is the *ket* vector $|z\rangle$ (*bra* vector $\langle z|$). Bra/ket vectors represent quantum system states, which naturally live in an appropriate *Hilbert space* H. We will discuss a little further about Hilbert spaces in the dedicated paragrph; for now, we can simply think of a Hilbert space as a complex vector space $\mathbb{C}^n$ and use them as synonyms. As well as the matrix case, the bra and ket forms of a vector are related as *complex conjugates* (or *Hermitian conjugates*) of each other.

$$\mathbf{z} \leftrightarrow \mathbf{z}^+ \tag{1.1.1}$$

$$|z\rangle \leftrightarrow \langle z| \tag{1.1.2}$$

In $\mathbb{C}^n$ we define[2] an *addition* operation for vectors and a *multiplication by a scalar* operation

$$\mathbf{z}a + \mathbf{z}b = a \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} + b \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \equiv \begin{bmatrix} az_1 + bw_1 \\ \vdots \\ az_n + bw_n \end{bmatrix} \quad a, b \in \mathbb{C} \quad scalars \tag{1.1.3}$$

$$a\mathbf{z}^+ + b\mathbf{z}^+ = a \begin{bmatrix} z_1^* & \dots & z_n^* \end{bmatrix} + b \begin{bmatrix} w_1^* & \dots & w_n^* \end{bmatrix} \equiv \begin{bmatrix} az_1^* + bw_1^* & \dots & az_n^* + bw_n^* \end{bmatrix} \tag{1.1.4}$$

The correspondent bra/ket version is

$$|z\rangle\, a + |w\rangle\, b \tag{1.1.5}$$

$$a\,\langle z| + b\,\langle w| \tag{1.1.6}$$

Inner product Given a column vector $\mathbf{z}$ and a raw vector$\mathbf{w}^+$, we define their *inner product* $\mathbf{w}^+\mathbf{z}$ as

$$\mathbf{w}^+\mathbf{z} = \begin{bmatrix} w_1^* & \dots & w_n^* \end{bmatrix} \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} = \sum_i w_i^* z_i \quad . \tag{1.1.7}$$

In Dirac notation the inner product reads as the *bra-ket product*

$$\langle w|z\rangle = (\sum_i w_i^* \langle i|)(\sum_j z_j |j\rangle) = \sum_{ij} w_i^* z_j \delta_{ij} = \sum_i w_i^* z_i \quad . \tag{1.1.8}$$

It is easy to prove that the inner product is

- *Bilinear*

$$\mathbf{u}^+(a\mathbf{v} + b\mathbf{w}) = \mathbf{u}^+ a\mathbf{v}a + u^+ b\mathbf{w} \quad , \tag{1.1.9}$$

$$(a\mathbf{u}^+ + b\mathbf{v}^+)\mathbf{w} = a\mathbf{u}^+\mathbf{w} + b\mathbf{v}^+\mathbf{w} \quad , \tag{1.1.10}$$

[2]The formal definition of a vector space involves verifying numerous properties for the two operations. However, we will skip these formalities and take a more intuitive approach instead.

and analogously

$$\langle u| \left(|v\rangle \, a + |w\rangle \, b \right) = \langle u|v\rangle a + \langle u|w\rangle b \quad , \tag{1.1.11}$$

$$\left(a \, \langle u| + b \, \langle v| \right) |w\rangle = a\langle u|w\rangle + b\langle v|w\rangle \quad . \tag{1.1.12}$$

- *Hermitian*

$$\mathbf{u}^+\mathbf{v} = (\mathbf{v}^+\mathbf{u})^* \quad , \tag{1.1.13}$$

which means

$$\langle u|v\rangle = \langle v|u\rangle^* \quad . \tag{1.1.14}$$

Linear operators Linear mappings of a vector space (as $\mathbb{C}^n$) in itself are square $n \times n$ (complex) matrices as $\mathbf{A}$, representing what in QM we call *linear operators*, as $\hat{A}$, on a Hilbert space[3]

$$\mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \quad . \tag{1.1.15}$$

Given a column vector $\mathbf{v}$ and a matrix A, we can form the *column/row vector matrix multiplication*

$$\mathbf{Az} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} = \begin{bmatrix} \sum_i A_1 i z_i \\ \sum_i A_2 i z_i \\ \vdots \\ \sum_i A_n i z_i \\ , \end{bmatrix} \tag{1.1.16}$$

$$\mathbf{z}^+\mathbf{A} = \begin{bmatrix} z_1^* & z_2^* & \dots & z_n^* \end{bmatrix} \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} = \begin{bmatrix} \sum_i z_i^* A_{i1} & \sum_i z_i^* A_{i2} & \dots & \sum_i z_i^* A_{in} \end{bmatrix} \quad . \tag{1.1.17}$$

In analogous fashion, bra/ket operator multiplications are defined as

$$\mathbf{Az} \leftrightarrow A \, |z\rangle \quad , \quad \mathbf{z}^+\mathbf{A} \leftrightarrow \langle z| \, A \quad . \tag{1.1.18}$$

One can demonstrate that column/row vector matrix multiplications are

[3] we will omit the $\hat{}$ symbol to lighten the notation

- *Linear in the vector factors*

$$\mathbf{A}(\mathbf{u}a + \mathbf{v}b) = \mathbf{A}\mathbf{u}a + \mathbf{A}\mathbf{v}b \quad , \tag{1.1.19}$$
$$(a\mathbf{u}^+ + b\mathbf{v}^+)\mathbf{A} = a\mathbf{u}^+\mathbf{A} + b\mathbf{v}^+\mathbf{A} \quad , \tag{1.1.20}$$

which implies

$$A(|u\rangle\, a + |v\rangle\, b) = A\,|u\rangle\, a + A\,|v\rangle\, b \quad , \tag{1.1.21}$$
$$(a\,\langle u| + b\,\langle v|)A = a\,\langle u|\, A + b\,\langle v|\, A \tag{1.1.22}$$

- *Associative*

$$(\mathbf{w}^+\mathbf{A})\mathbf{z} = \mathbf{w} + (\mathbf{A}\mathbf{z}) = \mathbf{w}^+\mathbf{A}\mathbf{z} \quad , \tag{1.1.23}$$

so that

$$(\langle w|\, A)\,|z\rangle = \langle w|\,(A\,|z\rangle) = \langle w|\, A\,|z\rangle \quad . \tag{1.1.24}$$

Given two matrices $\mathbf{A}, \mathbf{B}$ and two scalars a, b we can define two operations:

- The *operator linear combination*

$$a\mathbf{A} + b\mathbf{B} = a\begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} + b\begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix} =$$
$$= \begin{bmatrix} aA_{11} + bB11 & A_{12} + bB12 & \dots & A_{1n} + bB1n \\ A_{21} + bB21 & A_{22} + bB22 & \dots & A_{2n} + bB2n \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} + bBn1 & A_{n2} + bBn2 & \dots & A_{nn} + bBnn \end{bmatrix} \quad , \tag{1.1.25}$$

which is fully specified by

$$(a\mathbf{A} + b\mathbf{B})\mathbf{z} = \mathbf{A}\mathbf{z}a + \mathbf{B}\mathbf{z}b \quad , \tag{1.1.26}$$
$$\mathbf{z}^*(a\mathbf{A} + b\mathbf{B}) = a\mathbf{z}^+\mathbf{A} + b\mathbf{z}^+\mathbf{B} \quad . \tag{1.1.27}$$

In Dirac notation we write a linear combination of operators $aA + bB$ as

$$(aA + bB)\,|z\rangle = A\,|z\rangle\, a + B\,|z\rangle\, b \quad , \tag{1.1.28}$$
$$\langle z|\,(aA + bB) = a\,\langle z|\, A + b\,\langle z|\, B \quad . \tag{1.1.29}$$

- *Operator product*

$$\mathbf{AB} = \begin{bmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} & \dots & B_{1n} \\ B_{21} & B_{22} & \dots & B_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ B_{n1} & B_{n2} & \dots & B_{nn} \end{bmatrix}$$
$$= \begin{bmatrix} \sum_i A_{1i}B_{i1} & \sum_i A_{1i}B_{i2} & \dots & \sum_i A_{1i}B_{in} \\ \sum_i A_{2i}B_{i1} & \sum_i A_{2i}B_{i2} & \dots & \sum_i A_{2i}B_{in} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_i A_{ni}B_{i1} & \sum_i A_{ni}B_{i2} & \dots & \sum_i A_{ni}B_{in} \end{bmatrix} \quad , \quad (1.1.30)$$

completely defined by

$$(\mathbf{AB})\mathbf{z} = \mathbf{A}(\mathbf{Bz}) = \mathbf{ABz} \quad , \quad (1.1.31)$$
$$\mathbf{z}^+(\mathbf{AB}) = (\mathbf{z}^+\mathbf{A})\mathbf{B} = \mathbf{z}^+\mathbf{AB} \quad , \quad (1.1.32)$$

in correspondence with

$$(AB)\,|z\rangle = A(B\,|z\rangle) = AB\,|z\rangle \quad , \quad (1.1.33)$$
$$\langle z|\,(AB) = (\langle z|\,A)B = \langle z|\,AB \quad (1.1.34)$$

Adjoints and Hermitian operators Let A be the matrix representation of a linear operator on a Hilbert space H, the *adjoint* of $\mathbf{A}$ is $\mathbf{A}^+ = \mathbf{A}^{*T}$.

$$\mathbf{A}^+ = \begin{bmatrix} A^*_{11} & A^*_{21} & \dots & A^*_{n1} \\ A^*_{12} & A^*_{22} & \dots & A^*_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ A^*_{1n} & A^*_{2n} & \dots & A^*_{nn} \end{bmatrix} \quad (1.1.35)$$

and it is completely defined by the relation

$$\mathbf{w}^+\mathbf{A}^+\mathbf{z} = (\mathbf{z}^+\mathbf{Aw})^* \quad . \quad (1.1.36)$$

In Dirac formalism it reads as the adjoint operator A^+ of an operator, characterized by

$$\langle w|\,A^+\,|z\rangle = \langle z|\,A\,|w\rangle^* \quad . \quad (1.1.37)$$

Outer product A useful way of representing linear operators is through the *outer product*. Given a column vector $\mathbf{v}$ and a row vector $\mathbf{w}^+$, we define the outer product $\mathbf{zw}^+$ as

$$\mathbf{vw}^+ = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \begin{bmatrix} w_1^* & w_2^* & \dots & w_n^* \end{bmatrix} = \begin{bmatrix} z_1w_1^* & z_1w_2^* & \dots & z_1w_n^* \\ z_2w_1^* & z_2w_2^* & \dots & z_2w_n^* \\ \vdots & \vdots & \ddots & \vdots \\ z_nw_1^* & z_nw_2^* & \dots & z_nw_n^* \end{bmatrix} \quad , \quad (1.1.38)$$

which has the property

$$(\mathbf{z}\mathbf{w}^+)\mathbf{u} = \mathbf{z}(\mathbf{w}^*\mathbf{u}) = \mathbf{z}\mathbf{w}^+\mathbf{u} \quad , \tag{1.1.39}$$
$$\mathbf{u}^+(\mathbf{z}\mathbf{w}^+) = (\mathbf{u}^+\mathbf{z})\mathbf{w}^* = \mathbf{u}^+\mathbf{z}\mathbf{w}^+ \quad . \tag{1.1.40}$$

In Dirac formalism, the outer product becomes the ket/bra product $|z\rangle\langle w|$ specified by

$$(|z\rangle\langle w|)|u\rangle = |z\rangle(\langle w|u\rangle) = |z\rangle\langle w|u\rangle \quad , \tag{1.1.41}$$
$$\langle u|(|z\rangle\langle w|) = (\langle u|z\rangle)\langle w| = \langle u|z\rangle\langle w| \quad , \tag{1.1.42}$$

with the properties of

- Linearity in both factors

$$\mathbf{v}(a\mathbf{z}^+ + b\mathbf{w}^+) = \mathbf{v}a\mathbf{z}^+ + \mathbf{v}b\mathbf{w}^+ \quad , \tag{1.1.43}$$
$$(\mathbf{z}a + \mathbf{w}b)\mathbf{v}^+ = \mathbf{z}a\mathbf{v}^+ + \mathbf{w}b\mathbf{v}^+ \quad . \tag{1.1.44}$$

- Adjunction

$$(\mathbf{z}\mathbf{w}^+)+ = \mathbf{w}\mathbf{z}^+ \quad . \tag{1.1.45}$$

Then, we also have

$$|v\rangle(a\langle z| + b\langle w|) = |v\rangle a\langle z| + |v\rangle b\langle w| \,, \tag{1.1.46}$$
$$(|z\rangle a + |w\rangle b)\langle v| = |z\rangle a\langle v| + |w\rangle b\langle v| \,, \tag{1.1.47}$$
$$(|z\rangle\langle w|)^+ = \langle w|z\rangle \quad . \tag{1.1.48}$$

Now, given two vector states $|v\rangle$ and $|w\rangle$ it is possible to define the linear operator $|w\rangle\langle v|$ whose action is characterized by

$$(|w\rangle\langle v|)(|v'\rangle) \equiv |w\rangle\langle v|v'\rangle = \langle v|v'\rangle|w\rangle \,, \qquad \langle v|v'\rangle \in \mathbb{C} \quad . \tag{1.1.49}$$

This equation implies that we can define the result of the operator $|w\rangle\langle v|$ acting on $|v'\rangle$, as the result of multiplying $|v'\rangle$ for the complex number $\langle v|v'\rangle$. We can extend this definition to linear combinations of operators. Thus, $\sum_i a_i |w_i\rangle\langle v_i|$ is thus the linear operator that acting on $|v'\rangle$ produces $\sum_i a_i |w_i\rangle\langle v_i|v'\rangle$ as result.

Hilbert space and orthonormal bases In QM vector spaces equipped with an inner product define *Hilbert spaces*. In infinite dimensions, Hilbert spaces have additional technical properties beyond those of general inner product spaces, but we will not need to focus on those here. In this thesis, we limit ourselves to considering systems with a finite number of elements. This means we deal with finite-dimensional Hilbert spaces, where all linear operators are bounded.

The Hilbert space is the natural space where the state of quantum systems represented by bra/ket vectors lies. We can define the *norm* of a vector $\mathbf{v}$ by mean of the inner product

$$\|\mathbf{v}\| \equiv \sqrt{\mathbf{v}^{+}\mathbf{v}} \quad . \tag{1.1.50}$$

Analogously, the norm of $|v\rangle$ is

$$\| |v\rangle \| \equiv \sqrt{\langle v|v\rangle} \quad . \tag{1.1.51}$$

Thus, every Hilbert space is also a *normed* space. Intuitively, we can say the norm is a measure of the "length" or "size" of vectors. A *unit* vector is a vector $\mathbf{v}$ such that $\|\mathbf{v}\| = 1$ ($\| |v\rangle \| = 1$). We also say that $|v\rangle$ is normalized. It is convenient to normalize a vector by dividing by its norm: $|v\rangle / \| |v\rangle \|$ is the normalized version of $|v\rangle, \forall |v\rangle \in H$. We can also define a *metric* induced by the norm:

$$d(\mathbf{v}, \mathbf{y}) \equiv \|\mathbf{v} - \mathbf{y}\| = \sqrt{(\mathbf{v} - \mathbf{y})^{+}(\mathbf{v} - \mathbf{y})} \quad . \tag{1.1.52}$$

Thus, every Hilbert space (normed space) is also a metric space. A metric is a way to measure distance between elements in the vector space[4]. Now that we have introduced all the fundamental elements of linear algebra in both Dirac notation and matrix notation (which helps us gain a better understanding of the operations expressed in Dirac notation), we are ready to proceed by using only the first formalism. When discussing finite-dimensional Hilbert spaces[5], say $\mathbb{C}^n$, it is useful to address them in terms of their *vector bases*, i.e., sets of vectors as $\{|v_i\rangle, i = 1, \ldots, n\}$ that *span* the whole vector space:

$$|v\rangle = \sum_i \alpha_i |v_i\rangle, \alpha_i \in \mathbb{C} \quad \forall |v\rangle \in \mathbb{C}^n \quad . \tag{1.1.53}$$

In particular, we are interested in *orthonormal bases* $\{|i\rangle, i = 1, \ldots, n\}$ of C^n because they possess the advantageous property that the inner products of the basis vectors take on a straightforward normalized diagonal form. This characteristic simplifies analyses and calculations that require the use of bases, making them clearer and more transparent when orthonormal bases are employed[6]. An orthonormal basis is one that satisfies the relation

$$\langle i|\, j\rangle = \delta_{ij} \quad . \tag{1.1.54}$$

[4]More formal definitions would require an Hilbert space H to be complete as a *metric space* with the metric *induced* by the *norm*, which in turn is induced by the inner product. In simple terms, saying that H is complete as a metric space means that every sequence of points that gets closer and closer to a certain limit actually has that limit within H.

[5]As we mentioned, we will deal with finite-dimensional Hilbert space only, so let us omit the discussion of infinite-dimensional spaces with orthonormal bases containing an infinite number of elements.

[6]In the infinite-dimensional case, the Hilbert space H of wave functions serves as a continuous-index configuration space analogous to C^n. Consequently, orthonormal bases of an appropriate form should also exist in H, fulfilling the same important role that they play in C^n.

We can expand vectors in the orthonormal basis

$$|v\rangle = \sum_i |i\rangle \langle i|\, v\rangle = \sum_i |i\rangle\, v_i \quad , \tag{1.1.55}$$

$$\langle v| = \sum_i \langle v|\, i\rangle \langle i| = \sum_i v_i \langle i| \quad , \tag{1.1.56}$$

where $v_i = \langle i|\, v\rangle = \langle v|\, i\rangle, v_i \in \mathbb{C}$. This is equivalent to say

$$\sum_i |i\rangle \langle i| = I \quad . \tag{1.1.57}$$

Equation (1.1.57) is known as *completeness relation*. One key application of the completeness relation is that it provides a method for representing any operator in terms of outer product notation. Let $A : V \to W$ be a linear operator, and consider $|v_i\rangle$ and $|w_j\rangle$ as orthonormal basis for V and W, respectively. Then, by applying the completeness relation twice, we arrive at:

$$A = I_W A I_V \tag{1.1.58}$$

$$= \sum_{ij} |w_j\rangle \langle w_j|\, A\, |v_i\rangle \langle v_i| \tag{1.1.59}$$

$$= \sum_{ij} \langle w_j|\, A\, |v_i\rangle\, |w_j\rangle \langle v_i| \quad , \tag{1.1.60}$$

where, in the first line, we used the identity operators I_W and I_V of W and V. Equation (1.1.60) is the outer product representation for A, which has *matrix element* $\langle w_j|\, A\, |v_i\rangle$ in the ith column and jth row, with respect to the input basis $|v_i\rangle$ and the output basis $|w_j\rangle$.

Tensor product The *tensor product* is a method for combining vector spaces into a larger vector space and is crucial for understanding quantum mechanics in multi-particle systems. Let V and W be vector spaces of dimensions m and n, respectively, and for convenience, let's assume V and W are Hilbert spaces. Then, the tensor product $V \otimes W$ is an mn-dimensional vector space. The elements of $V \otimes W$ are linear combinations of "tensor products" $|v\rangle \otimes |v\rangle$, where $|v\rangle \in V$ and $|w\rangle \in W$. In particular, if $\{|i\rangle\}$ and $\{|j\rangle\}$ are orthonormal bases for V and W respectively, then $\{|i\rangle \otimes |j\rangle\}$ forms an orthonormal basis for $V \otimes W$ For simplicity, we often use shorthand notations like $|v\rangle\, |w\rangle$, $|v, w\rangle$ or even $|vw\rangle$. Properties $(i) - (iii)$ are satisfied:

(i) $z(|v\rangle \otimes |w\rangle) = (z\, |v\rangle) \otimes |w\rangle = |v\rangle \otimes (z\, |w\rangle), \quad \forall z \in \mathcal{C}, |v\rangle \in V, |w\rangle \in W \quad .$

(ii) $(|v_1\rangle + |v_2\rangle) \otimes |w\rangle = |v_1\rangle \otimes |w\rangle + |v_2\rangle \otimes |w\rangle\,, \quad \forall\, |v_1\rangle\,, |v_2\rangle \in V, |w\rangle \in W \quad .$

(iii) $|v\rangle \otimes (|w_1\rangle + |w_2\rangle) = |v\rangle \otimes |w_1\rangle + |v\rangle \otimes |w_2\rangle\,, \quad \forall\, |v\rangle \in V, |w_1\rangle\,, |w_2\rangle \in W \quad .$

Given $V \otimes W$, the linear operators acting on such a space are of the form $A \otimes B$, where A, B are linear operators acting on V, W, respectively. We can define $A \otimes B$ by

$$(A \otimes B)(\sum_i a_i\, |v_i\rangle \otimes |w_i\rangle) \equiv \sum_i a_i A\, |v_i\rangle \otimes B\, |w_i\rangle \quad , \tag{1.1.61}$$

where we used linear combination $|v_i\rangle \otimes |w_i\rangle \in V \otimes W$ to represent a generic elements of the present space. More generically

$$(\sum_i c_i A_i \otimes B_i)\, |v\rangle \otimes |w\rangle \equiv \sum_i c_i A_i\, |v\rangle \otimes B_i\, |w\rangle \quad . \tag{1.1.62}$$

We can use the inner product of each vector space in V, W to define an inner product in $V \otimes W$

$$(\sum_i a_i\, |v_i\rangle \otimes |w_i\rangle\,, \sum_j b_j\, |v'_j\rangle \otimes |w'_j\rangle) \equiv \sum_{ij} a_i^* b_j\, \langle v_i|\, v_j^*\rangle\, \langle w_i|\, w_j^*\rangle \quad , \tag{1.1.63}$$

where we used the notation "(,)" for the inner product of two quantities. produce exponential of an operator, A *vector subspace* of a vector space V is a space $W \in V$ such that W is also a vector space, thus, W must be closed under linear combinations of vectors.

1.2 Elements of Quantum Computation

From the previous section we see that quantum mechanics does not specify what laws a physical system must obey, but rather provides a mathematical framework for the development of such laws. In the following, we provide a series of elements at the basis of quantum computation to define a connection between the physical world and the mathematical formalism of QM.

1.2.1 The Qubits

Single Qubit Classical computation and information are built upon the concept of *bit*. The quantum analogue for quantum computation and information is the *quantum bit*, or *qubit* for short. We describe qubits as mathematical entities which can be physically realized as a variety of systems that exhibit quantum mechanical properties: photons, ions trapped in electromagnetic fields, atoms trapped using optical tweezers or in optical lattices, etc. In the following sections of the thesis, we will use spin qubits organized in chains. The mathematical approach allows us to build a general quantum theory of information and computation regardless of the actual physical system used for implementation. The difference between a bit and a qubit can be attributed entirely to the

quantum mechanical nature of the latter. We can attribute a state to a bit out of the discrete set $\{0, 1\}$; a qubit is a 2-state system, that means it also have a state, but it lives in a 2D-Hilbert space C^2, with eigenstates $|0\rangle$ and $|1\rangle$. Thus, a qubit state is expressible as a linear combination

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \quad \alpha, \beta \in \mathbb{C} \quad , \tag{1.2.1}$$

called *superposition*. The state of the qubit is determined by the values of α and β, that are the *amplitudes* of the basis state $|0\rangle$ and $|1\rangle$, respectively. In other words, we could say that a classical bit is like a coin: either "heads" or "tails". By contrast, a qubit can exist in a continuum of states between "head" ($|0\rangle$) and "tail" ($|1\rangle$) until we measure it. When a qubit is measured, it only ever gives '0' or '1', that is a single-bit information, with a certain probability determined by $|\alpha|^2$ and $|\beta|^2$, respectively. Naturally, $|\alpha|^2 + |\beta|^2 = 1$, since probabilities must sum to one. Thus, a qubit's state is a unit vector in a two-dimensional complex vector space. As we anticipated, We can choose the computational basis of the vectors:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad . \tag{1.2.2}$$

We can also give a geometrical representation of the qubit state. Since the qubit is a 2-state system, its vector state can be expressed as

$$|\psi\rangle = e^{i\gamma}(\cos(\theta/2)\,|0\rangle + e^{i\phi}\sin(\theta/2)\,|1\rangle) \tag{1.2.3}$$

where $0 \leq \gamma \leq 2\pi, 0 \leq \theta \leq \pi, 0 \leq \phi \leq 2\pi$. Note that $e^{i\gamma}$ is a global phase and does not have observable effects, so it can be neglected. The angles θ and ϕ define a point on the Bloch sphere. All points on the surface of the unitary sphere represent a possible state for a qubit (*pure state*). The inside points correspond to *mixed states*, whose treatment through density matrices lies beyond the scope of this thesis. Now we want to recall an important difference between classical and quantum qubits. One can always know the state of a classical bit by measuring it once. The result can be 0 or 1 and it corresponds to the state the bit was in before the measurement. This is not the case for quantum bits: one can reconstruct the state of a quantum particle only by measuring it infinite times over an ensamble of the same particle. Thus, a single measurement is not sufficient to know the exact state of the system. Further, acting on the particle, will make its wavefunction collapse: once we observe a qubit, its state will change from being in a superposition to occupy the specific state we measured (the measure will provide a single bit of information, 0 or 1) even after the observation. Quantum mechanically, the measurement process changes the state of the system throughout is irreversible.

Multiple qubits It is straightforward to assemble multi-body systems of qubits. For a system of n-qubits, the computational basis states are of the form

$$|x_1\rangle \otimes |x_2\rangle \otimes ... \otimes |x_n\rangle = |x_1\rangle\, |x_2\rangle ... |x_n\rangle = |x_1, x_2, ..., x_n\rangle \tag{1.2.4}$$

(they are all equivalent notations), and live in the total 2^n dimensional Hilbert space $(\mathbb{C}^2)^{\otimes n}$ formed by the tensor product of each single-qubit Hilbert space $\mathbb{C}^2_i, i = 1, \ldots, n$

$$(\mathbb{C}^2)^{\otimes n} = \mathbb{C}^2_1 \otimes \mathbb{C}^2_2 \otimes \cdots \otimes \mathbb{C}^2_n \quad . \tag{1.2.5}$$

Since a single-qubit state is defined by 2 amplitudes α and β, an n-qubits state counts 2^n amplitudes. Thus, for $n = 500$ we need 2^{500} amplitudes to simulate the state of an n-qubits system over a classical computer, a number much bigger than the atoms in the observable universe. Quantum computers, instead, can store this amount of information in only 500 atoms.

1.2.2 Quantum Gates

As bits, wires and logic gates acting on the bits are the fundamental elements of classical circuits, qubits, wires (both classical and quantum communication channels) and quantum logic gates are the fundamental elements of quantum circuits.
The simplest quantum gates are those acting on a single qubit. As we will see shortly, they can be viewed as 2×2 unitary matrix, and since there are infinitely many unitary matrices, there are also infinitely many quantum gates they represent. However, it turns out that the properties of the whole set can be summarized in those of a smaller, limited, subset.

Theorem 1.2.1. ***(Single-qubit gates decomposition)*** *An arbitrary single qubit unitary gate (which is a 2×2 unitary matrix) can be decomposed as a product of rotations*

$$U = e^{i\alpha} \begin{pmatrix} e^{-i\beta/2} & 0 \\ 0 & e^{-i\beta/2} \end{pmatrix} \begin{pmatrix} \cos(\frac{\gamma}{2}) & -\sin(\frac{\gamma}{2}) \\ \sin(\frac{\gamma}{2}) & \cos(\frac{\gamma}{2}) \end{pmatrix}, \begin{pmatrix} e^{-i\delta/2} & 0 \\ 0 & e^{-i\delta/2} \end{pmatrix} \quad , \tag{1.2.6}$$

where α, β, γ, δ are all real-valued.

The proof to the theorem will be clear at the end of section 1.2.2. The first matrix is a *(global) phase shift* and can thus be neglected, while second matrix is just an ordinary *rotation*. The first and last matrices can be seen as rotations in a different plane.
In general, we don't need to implement these gates for arbitrary values of α, β, γ and δ. Instead, we can construct arbitrarily good approximations of such gates using only specific fixed values of the angles In this way, we can simulate any single-qubit gate using a finite set of quantum gates. More generally, any quantum computation, regardless of the number of qubits, can be generated using a finite set of gates, known as a universal set for quantum computation.

Theorem 1.2.2. ***(Universal decomposition)*** *Single-qubit gates, combined with the CNOT gate (which acts on two qubits), form a universal gate set. This means that they can approximate any quantum operation on an arbitrary number of qubits.*

Now, let us first illustrate the main features of a generic single-qubit gate.

Single-qubit gates

Formally, single-qubit gates are transformations belonging to the group $SU(2)$ of rotations in three-dimensional spin state space

$$SU(2)=\{2\times 2 \text{ complex matrix } U \mid U^\dagger = U^{-1},\ det(U)=1\}$$

(this is also called "*fundamental representation* of the transformation group). We require U to be *unitary* in order to preserve the *scalar product* between two generic vectors $|\psi\rangle$ and $|\phi\rangle$:

$$\langle\psi|\phi\rangle \to \langle U\psi|U\phi\rangle = \langle\psi|U^\dagger U|\phi\rangle = \langle\psi|I|\phi\rangle = \langle\psi|\phi\rangle \quad , \tag{1.2.7}$$

which implies the conservation of the *norm* the state belonging to a generic $|psi\rangle$

$$||\psi|| = \sqrt{\langle\psi|\psi\rangle} \quad . \tag{1.2.8}$$

Further, we require $det(U)=1$ to ensure that the transformations include the identity I and form a continuous group. Then, in "the vicinity of identity" we get

$$|\psi'\rangle = U|\psi\rangle \approx (I - i\epsilon^a T_a)|\psi\rangle \quad , \tag{1.2.9}$$

where ϵ^a represent a set of parameters, $\epsilon^a \ll 1, \quad a=0,1,2$, while T_a are a set of traceless anti-hermitian matrices called *generators* of the transformation

We can impose that the composition of different transformations satisfies the closure property of the group they belong to. Calculations show that this implies

$$[T_a, T_b] = i\epsilon^{abc} T_c \quad , \tag{1.2.10}$$

where ϵ^{abc} is the *Levi-Civita* tensor of a 3 dimensional space (also known as *total anti-symmetric tensor*) and it encodes the commutation coefficients called "*group structure constants*". The commutation relation (1.2.10) represent the *Lie algebra* of the group $SU(2)$ and it is at the basis of the properties and behaviour of the group transformations. It is evident that (1.2.10) coincides with the commutation relation of angular momenta in quantum mechanics. Based on this analogy, it is easy to see how, in this specific representation, the generators of $SU(2)$ transformations (hermitian and traceless) can be rewritten as

$$\begin{pmatrix} a & b-ic \\ b+ic & -a \end{pmatrix} = a\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + b\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + c\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad . \tag{1.2.11}$$

In the right hand side of of (1.2.11) we can recognize the matrix form of *Pauli operators* σ_a. The usual convention identifies the generators of $SU(2)$ as

$$T_a = \frac{1}{2}\sigma_a \quad . \tag{1.2.12}$$

When exponentiated, Pauli matrices give rise to three useful class of unitary matrices, the *rotation operators* about the $\hat{x}, \hat{y}$ and $\hat{z}$ axes, defined by:

$$\begin{aligned}
R_x(\theta) \equiv e^{-i\sigma_x/2} &= \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) \sigma_x = \begin{pmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \quad , \\
R_y(\theta) \equiv e^{-i\sigma_y/2} &= \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) \sigma_y = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \quad , \\
R_z(\theta) \equiv e^{-i\sigma_z/2} &= \cos\left(\frac{\theta}{2}\right) I - i \sin\left(\frac{\theta}{2}\right) \sigma_z = \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix} \quad .
\end{aligned} \tag{1.2.13}$$

These matrices are precisely the ones that appear in the "Decomposition Theorem" 1.2.1 Each single-qubit gate can be thus visualized as a rotation on the Bloch sphere, mathematically represented by *linear* operators, since they have to preserve states superposition and operate in *coherent* way.

We have seen that each Pauli operator is associated to a specific single-qubit gate.

X-gate

$$\sigma_x \to X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad . \tag{1.2.14}$$

The X-gate, or $bit - flip$ gate, is the quantum equivalent of a classical NOT-gate, since it swaps the state $|0\rangle$ with the state $|1\rangle$ and viceversa.

$$X|0\rangle = |1\rangle \quad , \quad X|1\rangle = |0\rangle \quad . \tag{1.2.15}$$

The eigenstate of X are $|+\rangle$ and $|-\rangle$ with eigenvalues $+1$ and -1, respectively.

Z-gate

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad . \tag{1.2.16}$$

The Z gate, also known as $phase - flip$, acts only on the qubit state $|1\rangle$ by adding a phase $\theta = \pi$. Thus

$$Z|0\rangle = |0\rangle \quad , \quad Z|1\rangle = -|1\rangle \quad . \tag{1.2.17}$$

Its eigenvectors are $|0\rangle$ and $|1\rangle$ with eigenvalues $+1$ and -1, respectively.

Y-gate

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad . \tag{1.2.18}$$

The Y gate, or *bit-phase-flip*, is a sort of combination of the X and Z-gate. More precisly

$$Y|0\rangle = i|1\rangle \quad , \quad Y|1\rangle = -i|0\rangle \quad . \tag{1.2.19}$$

The eigenstates of the Y-gate are $|\pm i\rangle$, hence $|+i\rangle = \frac{|0\rangle + i|1\rangle}{\sqrt{2}}$ and $|-i\rangle = \frac{|0\rangle - i|1\rangle}{\sqrt{2}}$.

H-gate

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad . \tag{1.2.20}$$

It can be decomposed as

$$H = \frac{X + Z}{\sqrt{2}} \quad . \tag{1.2.21}$$

The H-gate, or *Hadamard*-gate, allows us to put a set of qubits in a balanced superposition of $|0\rangle$ and $|1\rangle$ states,

$$H|0\rangle = |+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad , \quad H|1\rangle = |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \quad . \tag{1.2.22}$$

and viceversa,

$$H|+\rangle = |0\rangle = \frac{|+\rangle + |-\rangle}{\sqrt{2}} \quad , \quad H|-\rangle = |1\rangle = \frac{|+\rangle - |-\rangle}{\sqrt{2}} \quad . \tag{1.2.23}$$

Thus, H performs a change of computational basis. Its eigenstates are

$$|H_+\rangle = \cos\left(\frac{\pi}{8}\right)|0\rangle + \sin\left(\frac{\pi}{8}\right)|1\rangle \ , \tag{1.2.24}$$

$$|H_-\rangle = -\sin\left(\frac{\pi}{8}\right)|0\rangle + \cos\left(\frac{\pi}{8}\right)|1\rangle \ , \tag{1.2.25}$$

with eigenvalues $+1$ and -1, respectively.

CNOT-gate

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad . \tag{1.2.26}$$

The $CNOT$-gate, or *conditional flip*, is a two-qubit gate and operates on a control qubit and a target qubit. When the control qubit is in the state $|c\rangle$ and the target qubit in the state $|t\rangle$, the CNOT gate performs the following transformation:

$$|c, t\rangle \to |c, t \oplus c\rangle \quad . \tag{1.2.27}$$

This means that the target qubit is flipped with an X-gate if and only if the control qubit is in the state $|1\rangle$,while $|t\rangle$ is left unchanged when the control is in $|0\rangle$. We can resume the action of this gate in the following equations;

$$CNOT\,|00\rangle = |00\rangle \quad , \tag{1.2.28}$$
$$CNOT\,|01\rangle = |01\rangle \quad , \tag{1.2.29}$$
$$CNOT\,|10\rangle = |11\rangle \quad , \tag{1.2.30}$$
$$CNOT\,|11\rangle = |10\rangle \quad . \tag{1.2.31}$$

1.3 Quantum time evolution: the evolution operator

In classical physics, the state trajectory in the phase space represents the solution to a problem written in the form of a differential equation

$$\frac{dy}{dt} = f(y) \quad . \tag{1.3.1}$$

The solution to first order is

$$y(t + \Delta t) \approx y(t) + f(y)\Delta t \quad . \tag{1.3.2}$$

Similarly, in quantum physics we consider the differential equation

$$i\hbar \frac{d\,|\psi\rangle}{dt} = H\,|\psi\rangle \quad , \tag{1.3.3}$$

known as *Schrödinger equation.* For a time independent H, the solution to this equation is

$$|\psi(t)\rangle = e^{-i\hbar^{-1}Ht}\,|\psi(t)\rangle \quad . \tag{1.3.4}$$

However, in the general case the Hamiltonian $H(t)$ of a quantum system is time-dependent. We assume that the evolution of the state $|\psi(t)\rangle$ still obeys the Schrödinger equation:

$$i\hbar \frac{d\,|\psi(t)\rangle}{dt} = H(t)\,|\psi(t)\rangle \quad , \tag{1.3.5}$$

and satisfies the *initial condition i.c.*

$$|\psi(0)\rangle = \psi_0 \quad . \tag{1.3.6}$$

Equation (1.3.5) is a 1^{st}-order differential equation with an associated evolution trajectory in the quantum Hilbert space[7]. By linearity of (1.3.5), if $|\psi_1(t)\rangle$ and $|\psi_2(t)\rangle$

[7]The general idea recalls the evolution of the phase $P(t) \equiv (\mathbf{q}(t), \mathbf{p}(t))$ of a classical system in Hamilton mechanics. In this case, the evolution trajectory associated to the 1^{st}-order differential equation lies in the classical phase space

are the solutions of the equation such that $|\psi_1(0)\rangle = |\psi_{10}\rangle$ and $|\psi_2(0)\rangle = |\psi_{20}\rangle$, then $|\psi(t)\rangle = c_1 |\psi_1(t)\rangle + c_2 |\psi_2(t)\rangle$ is the solution such that $|\psi(0)\rangle = c_1 |\psi_{10}\rangle + c_2 |\psi_{20}\rangle$. Therefore, there exists a linear operator $U(t,0)$ such that the solution of the Schrödinger equation (1.3.5) with i.c. (1.3.5) can be written as:

$$|\psi(t)\rangle = U(t,0)\,|\psi_0\rangle \quad , \tag{1.3.7}$$

or more generally (since the initial time is arbitrary)

$$|\psi(t)\rangle = U(t,s)\,|\psi_s\rangle \quad , \tag{1.3.8}$$

where $U(t,s)$ is a linear operator called *evolution operator*. $U(t,s)$ respects the *evolution operator equations*

$$i\hbar\frac{\partial U(t,s)}{\partial t} = H(t)U(t,s) \quad , \quad i\hbar\frac{\partial U(t,s)}{\partial s} = -U(t,s)H(s) \quad , \tag{1.3.9}$$

with i.c.

$$U(s,s) = 1 \quad . \tag{1.3.10}$$

Note that the first equation in (1.3.9) guarantees the compatibility between the (1.3.8) and the Schrödinger equation (1.3.5), while the second one in (1.3.9) ensures that $|\psi(t)\rangle$ doesn't depend on the initial time s, that is, as we said, arbitrary. The i.c. makes $|\psi(t)\rangle$ reduce to $|\psi(s)\rangle$ for $t = s$[8]. From the equations (1.3.9) and (1.3.10) we can derive the following properties of the evolution operator $U(t,s)$:

- $U(t,s)$ is *unitary*[9] and *invertible*

$$U(t,s)^+ = U(t,s)^{-1} \quad . \tag{1.3.11}$$

  This is reflected in the conservation of probability

$$\langle\psi(t)|\psi(t)\rangle = 1 \quad . \tag{1.3.12}$$

- The *chain identity* is true:

$$U(t,u)U(u,s) = U(t,s) \quad . \tag{1.3.13}$$

[8]Note that we have not required $t > s$.

[9]Only *closed quantum systems* can be accurately described by unitary transformations. In reality, however, all systems interact to some extent with their environment. Despite this, certain systems can still be approximated as closed and described effectively by unitary evolution. Moreover, in principle, any open system can be viewed as part of a larger, closed system, which undergoes unitary evolution as a whole.

- The *flip-inverse identity* is also true:

$$U(t,s) = U(t,s)^{-1} \quad . \tag{1.3.14}$$

These properties are essential for us because they are the basis of the *compositional nature* and *reversibility* of the state $|\psi(t)\rangle$. In other words, it it is absolutely equivalent whether a state $|\psi(s)\rangle$ evolves into $|\psi(u)\rangle$ and then into $|\psi(t)\rangle$ or directly into $|\psi(t)\rangle$. This property is central to the whole theory of *state preparation* in QA and DQA. In the end, we can say that the solution of the Schrödinger equation (1.3.5) reduces to the computation of the associated evolution operator $U(t,s)$. However, in most cases, the latter is a problem as difficult as the former.

Time-independent Hamiltonian As we anticipated, for the time-independent Hamiltonian case, $H(t) = H_0$, it is possible to derive an explicit expression of the evolution operator $U(t,s)$

$$U(t,s) = e^{-i\hbar^{-1}(t-s)H_0} = U_0(t,s) \quad . \tag{1.3.15}$$

Time-dependent Hamiltonian In adiabatic quantum computation, we typically use time-varying Hamiltonians. These Hamiltonians are not fixed but evolve based on parameters that are controlled by the experimentalist and can be adjusted during the course of an experiment. While the system is not strictly closed, its evolution can still be described, to a good approximation, by the Schrödinger equation with a time-dependent Hamiltonian. The upshot is that in this thesis we will often describe the evolution of quantum systems - even systems which are not closed – using unitary operators.

It has been demonstrated that in the general case of a time-dependent Hamiltonian, the formal solution of (1.3.5) is

$$|\psi(t)\rangle = U_T(t,t_0)\,|\psi(t_0)\rangle \quad , \tag{1.3.16}$$

where the symbol $U_T(t,t_0)$ denotes the unitary *time-ordered* operator

$$U_T(t,t_0) = T\left(e^{\int_{t_0}^{t} H(t')\,dt'}\right) \quad , \tag{1.3.17}$$

which is the solution of (1.3.9) (1.3.10). If U is known, then the corresponding differential equation (1.3.9) can be solved for any initial condition. As we said, finding an exact analytical solution to (1.3.9) is generally not possible, making approximations necessary. Solving (1.3.9) by iteration yields the perturbation series

$$U_T(t,t_0) = 1 - i\int_{t_0}^{t} dt_1 H(t_1) + i^2 \int_{t_0}^{t} dt_1 \int_{t0}^{t_1} dt_2 H(t_1)H(t_2) + \ldots \quad . \tag{1.3.18}$$

For the purposes of this thesis, the series (1.3.18) is useless to us because truncating the series at some point would result in a non-unitary approximation to $U_T(t_f, t_i)$. From equation (1.3.18) the subscript "T" can be understood as indicating "time-ordering". However, since (1.3.18) is not used further, the exact meaning of T is not crucial. It is enough to know that $U_T(t_f, t_i)$ is the solution to (1.3.9) and possesses evolution property

$$U_T(t, t_0) = T(e^{\int_{t_0}^{t} H(t')\,dt'}) = T(e^{\int_{t_0}^{t_1} H(t')\,dt'})T(e^{\int_{t_1}^{t} H(t')\,dt'}) \\ = U_T(t, t_1)U_T t_1, t_0 = W(t, t_0), \quad t \geq t_1 \geq t_0 \quad . \quad (1.3.19)$$

which is the analog of (1.3.13). In order to prove it, we start from (1.3.9) for the time-ordered operator

$$i\hbar \frac{\partial U_T(t, t_0)}{\partial t} = H(t)U_T(t, t_0) \quad , \quad (1.3.20)$$

$$i\hbar \frac{\partial U_T(t, t_0)}{\partial t_0} = -U_T(t, t_0)H(t_0) \quad , \quad (1.3.21)$$

with initial conditions

$$U_T(t_0, t_0) = \mathbb{I} \quad . \quad (1.3.22)$$

Then, we compute

$$i\hbar \frac{\partial (U_T(t, t_1)U_T(t_1, t_0)}{\partial t_1} = i\hbar \frac{\partial U_T(t, t_1)}{\partial t_1} U_T(t_1, t_0) + U_T(t, t_1) i\hbar \frac{\partial U_T(t_1, t_0)}{\partial t_1} \quad (1.3.23)$$

$$= -U_T(t, t_1)H(t_1)U_T(t_1, t_0) + U_T(t, t_1)H(t_1)U_T(t_1, t_0) = 0 \quad . \quad (1.3.24)$$

As a consequence

$$U_T(t, t_1)U_T(t_1, t_0) = U_T(t, t_0)U_T(t_0, t_0) = U_T(t, t_0)\mathbb{I} = U_T(t, t_0) \quad . \quad (1.3.25)$$

More generally, given a long time interval Δt of evolution, we can decompose the time-ordered operator $U_T(t + \Delta t, t)$ as

$$U_T(\Delta t, t) = U_T(t_r, t_{r-1})U_T(t_{r-1}, t_{r-2}) \ldots U_T(t_2, t_1)U_T(t_1, t) \quad , \quad (1.3.26)$$

$$t_j = p_j \Delta t \quad , \quad j = 1, \ldots, r \quad , \quad (1.3.27)$$

for some appropriate positive integer r, where $p_1 + p_2 + \cdots + p_r = 1$.

Chapter 2

Time evolution and adiabatic theorem

In this Chapter we introduce the adiabatic algorithms of interest in this thesis. As mentioned, they are tailored to solve some mathematical optimization problems. We begin by briefly discussing them.

2.1 How to encode classical problems into quantum Hamiltonians?

Combinatorial optimization problems are specified by N bits and m clauses. Each clause is a constraint on a subset of the bits which is satisfied only for certain assignments of those bits. The objective function representing the number of satisfied clauses is

$$C(z) = \sum_{\alpha=1}^{m} C_\alpha(z) \quad . \tag{2.1.1}$$

It is defined on N-bit strings $z = (z_1, z_2, ..., z_N)$, where each $z_i \in \{0, 1\}$. Each $C_\alpha(z)$, which usually depends on a subset of the bits, is 1 if the clause α is satisfied and 0 otherwise. Thus, (2.1.1) identifies $C(z)$ as a **cost function** which evaluates the "cost" of a candidate solution z compared to the problem, determining how quantitatively good z is.
We aim to solve combinatorial optimization problems on quantum computers, in order to exploit quantum resources such as the quantum adiabatic theorem and quantum tunneling in QA and DQA, and the quantum superposition in QAOA.
This approach is realistic because we can encode the hard problem into one of the evolution operators and then recover the solution string z as the ground state of the final Hamiltonian. This means that we can represent the optimization problem in terms of qubits by mapping the N-bit strings z of the problem into spin chains. In this way, the parameter describing the spin at each site is promoted from a discrete variable $z_i \in \{+1, -1\}$ (representing 'spin up' and 'spin down' respectively) to states in a quantum vector space (typically the spin-$\frac{1}{2}$ or two-dimensional representation of $SU(2)$). Thus,

the quantum computer works on a 2^N-dimensional Hilbert space $\mathcal{H}$ with computational basis $|z\rangle = |z_1\rangle \dots |z_N\rangle = |z_1 \dots z_N\rangle$. In this framework, each discrete variable is associated with a qubit that is treated quantum-mechanically, and associated to the Pauli operators $\hat{\sigma}_j^x$, $\hat{\sigma}_j^y$, $\hat{\sigma}_j^z$. In particular, we recall that

$$\sigma_z |0\rangle = +|0\rangle \quad , \quad \sigma_z |1\rangle = -|1\rangle \quad . \tag{2.1.2}$$

Now, the correspondence between classical bit variables and quantum states identified by the assigned eigenvalue of σ^z is easily observed to be:

	Bit	Eigenvector	Eigenvalue
z_1	0	$\|0\rangle$	$+1$
z_2	1	$\|1\rangle$	-1

$$z_j = \frac{1 - \sigma_j^z}{2} \quad , \quad j = 1, 2 \quad . \tag{2.1.3}$$

This defines a map to a classical cost function $C(z)$, to a quantum Hamiltonian H, $C(z) \to H$. For instance, the cost function $C(z) = z_1 z_2$ corresponds to the quantum Hamiltonian operator $H = \frac{1}{4}(1 - \sigma_1^z)(1 - \sigma_2^z)$. Note that the σ_j^z associated to different qubits commute with each other, so they are measurable simultaneously.
Once $H(z)$ is defined for our problem, we can make the system evolve within a time interval $[t_i, t_f]$ from an initial state towards a final one using an evolution operator that is the exponential of a time-dependent Hamiltonian $H(t)$ of the form:

$$H(t) = (1 - \lambda(t))H_i + \lambda(t)H \quad , \tag{2.1.4}$$

$$\text{with} \quad \lambda(t_i) = 0 \quad , \quad \lambda(t_f) = 1 \quad . \tag{2.1.5}$$

As we mentioned, $H(z)$ is the problem Hamiltonian (also known as *cost Hamiltonian*) in which we have encoded the problem solution. H_i is the initial Hamiltonian, representing the system's starting state, and acts as a driving term to facilitate the evolution towards the problem Hamiltonian. In this way, at $t = t_i$ the system Hamiltonian is $H(t_i) = H_i$ and the system is in the initial state which coincides with the ground state of H_i. We want the evolution to bring the system to the ground state of the problem Hamiltonian (note that at $t = t_f$ we have $H(t_f) = H$). Finally, after measuring the system final state we can recover the desired classical solution of our problem (we recall that a measurement of a quantum state gives a classical sequence of bits of information). There are various methods to carry out the system evolution, ranging from those that implement a physical realization of the described process, such as *quantum annealers* QA, to methods that digitally simulate the process on a quantum computer, thus digitalized verions of QA (DQA) and variational algorithm, as QAOA.

To simplify our analysis, we define a custom Hamiltonian as

$$H = \sum_{j=0}^{N-1} H_{j,j+1} = \sum_{j=0}^{N-1} \Delta(j) \sigma_j^z \sigma_{j+1}^z \quad . \tag{2.1.6}$$

This is a straightforward Hamiltonian whose associated classical function can be easily derived using the mapping defined in (2.1.3). By doing so, we can focus our attention only the Digital Quantum Annealing (DQA) algorithm and on how to best approximate the adiabatic evolution operator using two-qubit unitary gates.
The system size is N, the σ_j^z are the Pauli-z matrices acting at position j, with $\sigma_{N+1}^z = \sigma_0^z$ (periodic condition); the parameter $\Delta(j)$ is the **anisotropy**, which measures the spins interactions along $\hat{z}$. Each $H_{j,j+1}$ represents the interactions between two consecutive spins. Note that (2.1.6) is diagonal on the computational basis $\{|z\rangle\}$.
We also define H_i as the *transverse-field Hamiltonian*

$$H_i = H_x = h_x \sum_{j=0}^{N-1} \sigma_j^x \quad . \tag{2.1.7}$$

Let us specify that h_x is the strength of the transverse field in the x-direction and σ_i^x is the x-th component of the spin operator on site i. This Hamiltonian tends to align all spins in the x-direction when h_x is large, producing a ground state that is a superposition of all basis states in the z-direction. This is a highly "delocalized" state, making it easy to initialize. Thus, we can write the time-dependent Hamiltonian (2.1.4) as

$$H(t) = (1 - \lambda(t))H_x + \lambda(t)H \quad . \tag{2.1.8}$$

Throughout this thesis, we will focus on closed spin chain systems with periodic boundary conditions, assuming they consist of an even number N of spins.

2.2 Continuous quantum annealing: the Adiabatic theorem

Quantum annealers take advantage of the shortcut provided by adiabatic quantum evolution in an analog quantum system. To illustrate this, consider a system initially prepared in an eigenstate of a time-dependent Hamiltonian. In our notation, let $|\psi_j(t)\rangle, j \in \{0, 1, 2, ...\}$, denote the instantaneous eigenstate of $H(t)$ with energy $E_j(t)$ such that $E_j(t) \leq E_{j+1}(t), \forall j, t$, i.e.,

$$H(t)\,|\psi_j(t)\rangle = E_j(t)\,|\psi_j(t)\rangle \quad , \tag{2.2.1}$$

and $j = 0$ denotes the (possibly degenerate) ground state. Assume that a system initial state is prepared in $|\psi_0(0)\rangle$, that is the ground state of the Hamiltonian $H(0) = H_x$. This means

$$|\psi_0(0)\rangle = |-\rangle^{\otimes N} = \left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right)^{\otimes N} \quad , \tag{2.2.2}$$

according to which each of the N qubits of the system can be found in both states $|0\rangle$ and $|1\rangle$. This initial state is easy to initialize as we can start with all qubits initialized

to $|1\rangle$ (the computational basis state) and then apply a Hadamard gate to each qubit in the state. We use the $H(t)$ (2.1.4) and initial state (2.2.2) both in the QA and DQA algorithms.
As the system evolves in time according to the Schrödinger equation (1.3.5) the state $|\psi(t)\rangle$ will continue to follow the corresponding instantaneous ground state of $H(t)$, namely $|\psi_0(t)\rangle$, provided that H(t) changes sufficiently slowly. The required slowness of this change is determined by the *adiabatic theorem* AT, which essentially quantifies how gradual the evolution must be to ensure that the system remains in its ground state. If this adiabatic condition is met, the system will evolve toward the ground state of the final Hamiltonian $H(t_f) = H$ (t_f is the total evolution time) which encodes the solution to the optimization problem. Thus, by controlling the evolution speed, we guide the system smoothly to the desired solution state. There are several formulation of AT [1], each providing slightly different results for the precision of bounds on the evolution time t_f and the error in state preparation, depending on the regularity properties of the time-dependent Hamiltonian and the size of the spectral gap. The "best" AT to use thus depends on the specifics of the quantum systems. Here, we will discuss only a few of the existing versions.
One of the simplest and oldest traditional AT [26] states that:

Theorem 2.2.1. ***(Adiabatic theorem)*** *A system initialized in $|\psi_0(0)\rangle$ remains in the same instantaneous eigenstate $|\psi_0(t)\rangle, \forall t \in [0, t_f]$ (up to a global phase), provided that*

$$\max_{t\in[0,t_f]} \frac{|\langle\psi_1|\partial_t\psi_0\rangle|}{|E_1 - E_0|} = \max_{t\in[0,t_f]} \frac{|\langle\psi_1|\partial_t H|\psi_0\rangle|}{|E_1 - E_0|^2} \ll 1 \quad . \tag{2.2.3}$$

However, this version of the theorem is not rigorous: if the Hamiltonian includes an oscillatory driving term, then the eigenstate population will oscillate with a time scale determined by this term, that is independent of t_f, even if the adiabatic criterion is satisfied.

Example 1 To see this consider an Hamiltonian of the form:

$$H(t) = a\sigma^z + b\sin(\omega t)\sigma^x \quad . \tag{2.2.4}$$

In this case, the adiabatic condition reduces to $|b\omega| \ll \alpha^2$. Even if this condition is satisfied, at resonance (when $\omega \approx 2\alpha$) the system undergoes Rabi oscillations with period T independent of t_f, $T = \pi|b|$.

A more precise version of AT that excludes such additional timescales was formulated by [2]. It holds only for Hamiltonians that can be written as $H_{t_f}(st_f) = H(s)$, where $s \equiv t/t_f \in [0, 1]$ is a dimensionless time and $H(s)$ is t_f-independent. Note that this definition includes the *"interpolating" Hamiltonians*, i.e., Hamiltonians of the form

$$H(s) = A(s)H_0 + B(s)H_1 \quad , \tag{2.2.5}$$

as (2.1.8), where $A(s)$ and $B(s)$ are monotonically decreasing and increasing, respectively, and excludes the multiple timescales cases. The Schrödinger equation then becomes

$$\frac{1}{t_f}\frac{\partial \psi_{t_f}(s)}{\partial s} = -iH(s)\,|\psi_{t_f}(s)\rangle \quad , \tag{2.2.6}$$

and the adiabatic condition used is reminiscent of (2.2.3)

$$\frac{1}{t_f}\max_{s\in[0,1]}\frac{|\langle\psi_1(s)|\partial_s H(s)|\psi_0(s)\rangle|}{|E_1(s)-E_0(s)|^2} \ll 1 \quad , \tag{2.2.7}$$

where the denominator is often addressed as the spectral gap $\Delta_{ij}(s) = E_1(s) - E_0(s)$. In particular, we are interested in $\Delta \equiv \min_{s\in[0,1]}\Delta(s) = \min_{s[0,1]} E_1(s) - E_0(s)$, since both (2.2.3) and (2.2.7) can be summarized by the condition that the total adiabatic evolution time t_f should be large on the timescale set by $\frac{1}{\Delta^2}$.

However, we have not seen any bounds on the closeness between the actual time-evolved state and the desired eigenstate, yet. To continue, following Kato's approach in [18], let us introduce two new operators. For all $s \in [0,1]$, we define:

- $P(s)$ as the eigenprojector of $H(s)$. It represent the ideal adiabatic projector onto the instantaneous ground state[1] $|\psi_0(s)\rangle$ of $H(s)$ with eigenenergy $E_0(s)$. It represents the state the system would ideally remain in if it followed a perfectly adiabatic path without any dynamic deviations.

- $P_{t_f}(s) = |\psi_{t_f}(s)\rangle\langle\psi_{t_f}(s)|$ as the projector onto the actual time-evolved state $|\psi_{t_f}(s)\rangle$, which is the solution of the time-dependent Schrödinger equation with Hamiltonian $H(s)$ as it evolves in time.

In real scenarios where the Hamiltonian evolution is not perfectly adiabatic, $|\psi_0(s)\rangle$ and $|\psi_{t_f}(s)\rangle$ may differ, particularly if there are dynamic effects (like non-adiabatic transitions) that arise due to a finite evolution rate. Therefore, the *istantaneus adiabatic distance* $\|P_{t_f}(s) - P(s)\|$ or the *final adiabatic distance* $\|P_{t_f}(1) - P(1)\|$ represents the deviation of the actual evolution from the ideal adiabatic behavior. We also assume that the spectral gap never vanishes, i.e. $\Delta > 0$[2].
Another version of the AT rigorously establishes the gap dependence of t_f , without any strong assumptions on the smoothness of $H(s)$ [16]

[1] We don't consider any restriction on the degeneracy of the ground state, and hence of the projector P(s), which can be (even infinitely) degenerate.

[2] In [3] it is presented a version of AT without gap conditions, where the estimate on the error term is o(1) as $t_f \to \infty$-

Theorem 2.2.2. ***(Adiabatic Theorem)*** *Suppose that the spectrum of $H(s)$ restricted to P has a non vanishing instantaneous gap $\Delta(s) = E_1(s) - E_0(s) > 0$ and that $H(s)$ is twice continuously differentiable. H, $H^{(1)}$ and $H^{(2)}$ are bounded operators, since we are in a finite-dimensional space*[3]*. Then for any $s \in [0,1]$ we have*

$$\|P_{t_f}(s) - P(s)\| \leq \frac{m(0)\|H^{(1)}(0)\|}{t_f \Delta^2(0)} + \frac{m(s)\|H^{(1)}(s)\|}{t_f \Delta^2(s)} + \frac{1}{t_f}\int_0^s \left(\frac{m\|H^{(2)}\|}{\Delta^2}\right) + \left(\frac{7m\sqrt{m}\|H^{(1)}\|^2}{\Delta^3}\right) \quad . \quad (2.2.8)$$

Equation (2.2.8) emphasizes how the deviation from the ideal adiabatic evolution depends on the norm of the 1^{st} or 2^{nd} time derivative of $H(s)$, rather than the matrix element that appears in (2.2.7). Theorem 2.2.2 shows that the adiabatic limit can be approached arbitrarily closely if (but not only if)

$$t_f \gg \max\{\ \max_{s\in[0,1]} \tfrac{\|H^{(2)}(s)\|}{\Delta^2(s)}, \max_{s\in[0,1]} \tfrac{\|H^{(1)}(s)\|^2}{\Delta^3(s)}, \max_{s\in[0,1]} \tfrac{\|H^{(1)}(s)\|}{\Delta^2(s)}\ \} \quad . \quad (2.2.9)$$

If $H(s)$ satisfies certain stricter assumptions, i.e., it belongs to the *Gevrey class* G^α, in [9] it is achieved a scaling of t_f with $1/\Delta^2$ (up to a logarithmic correction). This scaling is significantly better than that of every other rigorous AT formulations, which generally exhibit a worse gap dependence (cubic or higher).

Definition *(Gevrey class)*: $H(s) \in G^\alpha$ if $dH(s)/ds \neq 0, \forall s \in [0,1]$, and there exist constants $C, R > 0$, such that

$$\max_{s\in[0,1]} \|H^{(k)}(s)\| \leq CR^k k^{\alpha k}, \forall k \geq 1 \quad . \quad (2.2.10)$$

In simple terms, we want $H(s)$ to be bounded and infinitely differentiable, with limited value of magnitude for the higher derivatives.

Example 2 Consider the Hamiltonian $H(s) = [1 - \lambda(s)]H_0 + \lambda(s)H_1$, with

$$\lambda(s) = \begin{cases} c\int_{-\infty}^{s} e^{-\frac{1}{x-x^2}}\,dx & \text{if } s \in [0,1] \\ 0 & \text{if } s \notin [0,1] \end{cases} \quad .$$

From a rapid analysis of the function $\lambda(s)$ it emerges that since $e^{-\frac{1}{x-x^2}}$ is analytic in $(0,1)$, so is $\lambda(s)$. Further:

- Behavior outside $[0,1]$: for $s < 0$ or $s > 1$, $\lambda(s) = 0$, which ensures it doesn't affect the Hamiltonian outside this time interval.

[3] notation: $H^{(k)}(s) \equiv (\frac{\partial}{\partial t})^k H(t)|_s$

- Behavior within $[0,1]$: the term $e^{-\frac{1}{x-x^2}}$ has the following properties:
  - For $x \in (-\infty, 0)$ and $x \in (1, \infty)$, the expression $x - x^2$ is negative, leading to
    $$e^{-\frac{1}{x-x^2}} \to 0 \quad \text{as } x \text{ approaches either endpoint.}$$
  - For $x = 0$ and $x = 1$, the expression becomes $e^{-\infty}$, which is also 0. Therefore, the function $e^{-\frac{1}{x-x^2}}$ approaches 0 outside the interval $(0, 1)$.
- Integral behavior: the integral
  $$\int_{-\infty}^{s} e^{-\frac{1}{x-x^2}}\, dx$$
  will produce a finite value for $s \in [0, 1]$ due to the rapid decay of the integrand outside the interval where it contributes significantly. The integral is well-defined and smooth as it will only accumulate positive contributions as s increases from $-\infty$ to 1.
- Smoothness and derivatives: $\lambda(s)$ is continuous and differentiable within the interval $(0, 1)$ due to the properties of the integral and the fact that $e^{-\frac{1}{x-x^2}}$ is smooth for $x \in (0, 1)$.
  We can compute the first derivative of $\lambda(s)$:
  $$\lambda'(s) = c \cdot e^{-\frac{1}{s-s^2}} \quad \text{for } s \in [0, 1] \quad .$$
  Higher-ordser derivatives can be computed recursively applying the Leibniz rule, resulting in terms that grow in a factorial-like manner. Since $e^{-\frac{1}{x-x^2}}$ is a smooth function, all derivatives will also be smooth.

Figure 2.1 may help you to visualize the behavior of the function.

Going back to $H(t)$, its k-th derivative with respect to s is:

$$H^{(k)}(s) = \lambda^{(k)}(s)(H_1 - H_0) \quad .$$

Therefore, the norm $\|H^{(k)}(s)\|$ can be written as:

$$\|H^{(k)}(s)\| = |\lambda^{(k)}(s)| \cdot \|H_1 - H_0\| \quad .$$

The k-th derivative of $\lambda(s)$ involves combinations of derivatives of the integrand $e^{-\frac{1}{x-x^2}}$. Since $e^{-\frac{1}{x-x^2}}$ decays rapidly, the derivatives $\lambda^{(k)}(s)$ will also decay but with a rate that depends on k. Using the general behavior of derivatives of exponentially decaying integrals, we can approximate the growth rate of $\lambda^{(k)}(s)$ by:

$$|\lambda^{(k)}(s)| \leq C'(k!)^2 \quad ,$$

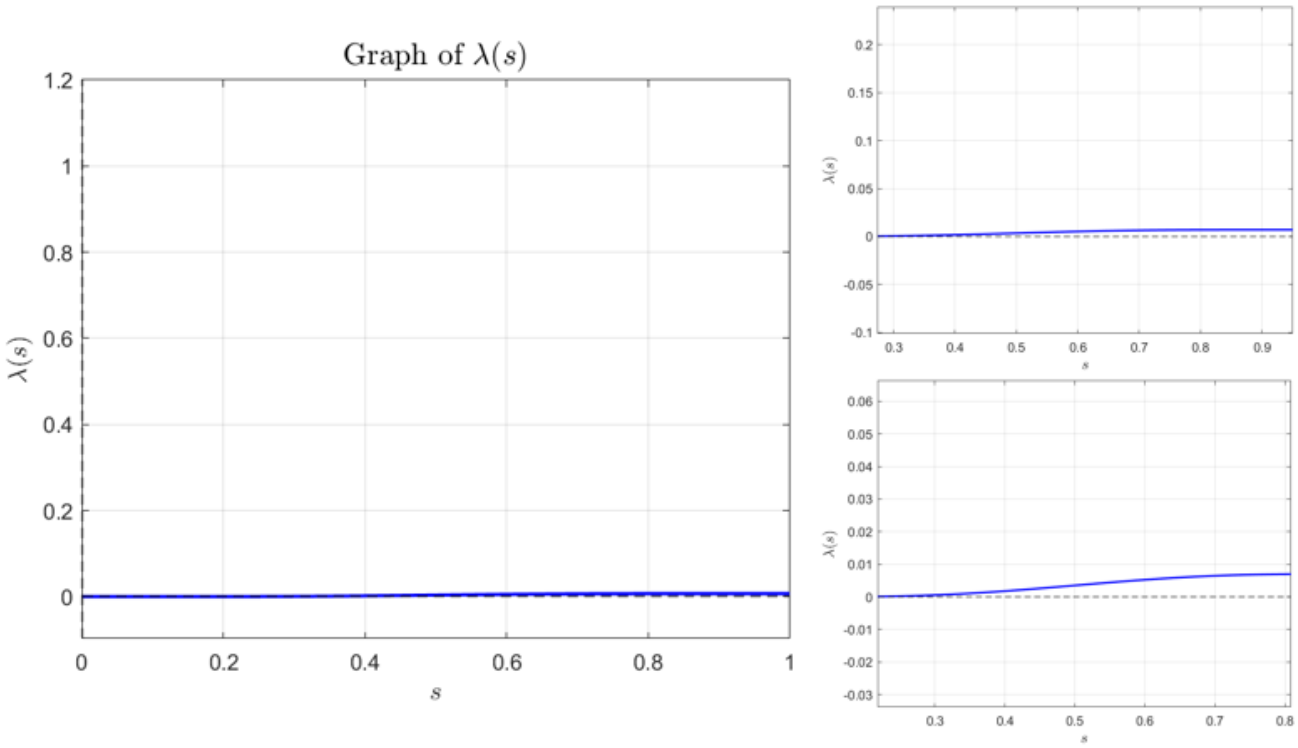

Figure 2.1: Function $\lambda(s)$ defined in Example 2 plotted with MatLab. On the right, zoomed-in views of the function are shown for different intervals on the axes.

where C' is a constant related to the decay properties of $e^{-\frac{1}{x-x^2}}$. By Stirling's approximation, $k! \approx \sqrt{2\pi k}\,(k/e)^k$, and thus we have:

$$(k!)^2 \approx (2\pi k) \cdot \left(\frac{k}{e}\right)^{2k} \quad .$$

So, we can bound $|\lambda^{(k)}(s)|$ by a term proportional to Ck^{2k} for some constant C and we can conclude that

$$\|H^{(k)}(s)\| \leq Ck^{2k} \cdot \|H_1 - H_0\| \quad ,$$

where C is a suitable constant that includes the norm $\|H_1 - H_0\|$. This bound shows that $H(s)$ satisfies the growth condition for the Gevrey class G^2.
Now, we are ready to state a new version of the AT.

Theorem 2.2.3. ***(Adiabatic Theorem)****: Assume that $H(s)$ is bounded and $H(s) \in G^\alpha$, with $\alpha > 1$. Define h as $h \equiv \|H(0)\| = \|H(1)\|$ and assume that $\Delta \ll h$. If*

$$t_f \geq \frac{K}{\Delta^2} |\ln(\Delta/h)|^{6\alpha} \quad , \tag{2.2.11}$$

for some Δ-independent constant $K > 0$ then $\|P_{t_f}(s) - P(s)\|$ is $o(1), \forall s \in [0,1]$.

As we said before, theorem 2.2.3 state the existence of a lower bound $t_f = \mathcal{O}(\Delta^{-2}/\ln(\Delta))$, but it does not provide any error bound. We can find an exponentially small error bound in t_f with a cubic dependence on $1/\Delta$ in [13]:

Theorem 2.2.4. *Assume that all derivatives of the Hamiltonian $H(s)$ vanish at $s = 0, 1$, and that the Gevrey condition*

$$\max_{s\in[0,1]} \|H^{(k)}(s)\| \leq CR^k \frac{(k!)^{1+\alpha}}{(k+1)^2}, \forall k \geq 1 \tag{2.2.12}$$

is satisfied, for three constants $C, R, \alpha > 0$. Then, the adiabatic error is bounded as

$$\min_{\theta} \|P_{t_f}(1) - e^{i\theta}P(1)\| \leq c_1 \frac{C}{\Delta} e^{-(c_2 \frac{\Delta^3}{C^2} t_f)^{\frac{1}{1+\alpha}}} \quad , \tag{2.2.13}$$

where $c_1 = eR(\frac{8\pi^2}{3}^3)$ and $c_2 = \frac{1}{4eR^2}(\frac{3}{4\pi^2})^5$.

Thus, for $t_f \gg \frac{c^2}{\Delta^3}$, the adiabatic error is exponentially small in t_f.

Chapter 3

Towards Digital Quantum Annealing

Circuit quantum computation is a complementary approach to AQC. It enables to construct arbitrary interactions and to implement error correction, which allows for *scalability*, which was problematic in AQC due to the presence of noise; this limited the procedure in the system size. A drawback of using quantum circuit algorithms is that they are tailored for specific problems. *Digitized adiabatic* quantum algorithm combine pros of both approaches. In this section, we will discuss how to encode the solution to a hard problem in the ground state of a quantum Hamiltonian, how to digitally simulate an adiabatic analog evolution, the size of the error introduced by this approximation, and how this error scales.

In Sec. 1.3, we saw that an exact analytical solution to either (1.3.5) or (1.3.9) is not possible in general, so approximations are needed. There exists many methods for decomposing and approximating solutions to these equations, such as Runge–Kutta methods, Magnus expansions [24], [5], and product formulae. Product formulae assume that we can decompose $H(t)$ as

$$H(t) = \sum_{j=0}^{m-1} H_j(t) \quad , \tag{3.0.1}$$

where m is the number of operators in the decomposition, and each $H_j(t)$ is such that it can be easily exponentiated for every $t \in [t_i, t_f]$. Thus, the operator U in (1.3.17) is approximated by a product of ordinary operator exponentials of $H_j(t)$:

$$U(t_i + \Delta t, t_i) \approx \prod_{p=0}^{N-1} e^{H_{j_p}(t_p)\Delta t_p} \quad , \tag{3.0.2}$$

with $\Delta t = t_f - t_i$ time interval, Δt_p a real number proportional to Δt and N is the number of terms used in the product. The reason we prefer the approach with product formulas is that it offers several advantages over alternative methods, while consistently

addressing the critical aspects of the problem: the t-dependence of each H_j and the non-commutativity between two any H_i, H_j (which makes it non-trivial to decompose U into a product of exponentials such that (3.0.2) holds). One key benefit is that product formulas can explicitly preserve certain symmetries of U, which means that the approximation in (3.0.2) remains unitary - unlike Runge–Kutta methods, which do not preserve unitarity in their approximations. Additionally, product formulas do not require the computation of commutators or integrals, as is necessary with the Magnus expansion. They often involve only sparse matrix-vector multiplications, making them well-suited for algorithms that are easily parallelized. Therefore, product formulae approximations generate a sequence of unitary operations that accurately represent the time-evolution operator, providing a set of instructions for a quantum computer to simulate a time-dependent quantum system.

3.1 Trotter and Suzuki formulae

3.1.1 Time-independent Hamiltonians

To study some of the most commonly used product formulas for approximating evolution operators, we begin with the time-independent Hamiltonian operator (see section 1.3) and consider a short time interval δt. At first order, we have

$$|\psi(t+\delta t)\rangle \approx (I - iH\delta t)\,|\psi(t)\rangle \ . \tag{3.1.1}$$

In many physically interesting situations, H can be written as a sum over local interactions. A typical example is that of spin chains such as the Ising model [12] [19] [25]. Specifically, for a system of N particles,

$$H = \sum_{j=0}^{L} H_j \quad , \tag{3.1.2}$$

where each H_j acts on at most a constant c number of qubits, and L is a polynomial in N. Such locality is physically resonable, since most interactions fall off while increasing distance or difference in energy. Equation (3.1.2) is important because $e^{-iH_j t}$ is much easier to approximate in quantum circuits than e^{iHt} since it involves a much smaller number of qubits. On the other side, the terms H_j often do not commute with each other, i.e.

$$[H_j, H_k] \neq 0 \implies e^{-iHt} = e^{-i\sum_{j=0}^{L} H_j} \neq \prod_{j=0}^{L} e^{-iH_j t} \quad . \tag{3.1.3}$$

This is where Trotter's theorem [34, 31] becomes useful.

Theorem 3.1.1. ***(Trotter formula)*** *Assume that $H = A + B$, where A and B are two Hermitian time-independent operators. Then,*

$$U(t_0 + \delta t, t_o) = e^{-i\delta t H} \approx e^{-i\delta t A} e^{-i\delta t B} + \mathcal{O}(\delta t^2) \quad . \tag{3.1.4}$$

This gives an accurate approximation for small δt. For large Δt, we could use equation (3.1.4) to derive the *Trotter formula*

$$\lim_{x\to\infty} (e^{iA\Delta t/n} e^{iB\Delta t/n})^n = e^{i(A+B)\Delta t} \quad , \tag{3.1.5}$$

or alternatively

$$e^{-i\Delta t H} = e^{-i\Delta t(A+B)} = (e^{-i\Delta t A/n} e^{-i\Delta t B/n})^n + \mathcal{O}(\Delta t^2/n) \quad , \tag{3.1.6}$$

which is valid even if A and B don't commute. Therefore, when the Hamiltonian H of a target system is made of two non-commuting terms A and B, we can approximate its continuous-time evolution by alternating the evolution under A and B, by a sequence of elementary steps of duration $\delta t = \Delta t/N$, which could be implemented as quantum "gates" acting on neighboring qubits. The smaller δt, the closer the discrete and continuous dynamics remain. A proof of the theorem 3.1.1 is given in [27]. More generally, for a sum of an arbitrary number of operators A_j, similar formulas yield the same error scaling. To achieve better scaling, one can use an alternative product of exponentials. For example, if H still consists of two non-commuting operators A and B, so that $H = A+B$, the following decomposition formula can be applied

$$e^{-itH} = e^{-it(A+B)} = e^{t_1 A} e^{t_2 B} e^{t_3 A} e^{t_4 B} \dots e^{t_m A} + \mathcal{O}(t^{m+1}) \quad , \tag{3.1.7}$$

for any positive integer m, with appropriately chosen parameters $\{t_j\}$ [32]. When H depends on time t explicitly, that is $H = H(t)$, we are dealing with the case of ordered exponentials. The situation becomes more complicated, but with a bit of work we can do similar considerations.

3.1.2 Time-dependent Hamiltonians

The purposes of this section is to give a general theory of decomposing ordered exponentials using the ordinary decomposition formulae of exponential operators. As in [33] we follow the strategy of reducing the decomposition problem of ordered exponentials to that of ordinary exponential operators.
Let us introduce the *super-operator*[1] $\mathcal{J}$:

$$F(t) e^{\Delta t \mathcal{J}} G(t) = F(t + \Delta t) G(t) \quad , \tag{3.1.8}$$

[1]While a typical operator acts on a vector within a vector space, a super-operator acts on other operators, which themselves may represent physical observables, density matrices, or state transformations. If one thinks of an operator as a matrix that transforms a vector, a super-operator transforms the operator (or matrix) itself. Thus, if ρ is a density matrix describing a quantum state, a super-operator S acts on ρ to change it in a specified way:$\rho \to S(\rho)$. A common example of a super-operator is the commutator of two operators A and B, $[A, B]$.

for any (even non-differentiable)[2] operators $F(t)$ and $G(t)$. If $F(t) = 1$ we have

$$1 \cdot e^{\Delta t \mathcal{J}} G(t) = e^{\Delta t \mathcal{J}} G(t) = G(t) \quad . \tag{3.1.10}$$

The approximation of a time-ordered exponential relies entirely on the following formula, which is the central point of this section. Any ordered exponential can be written as an ordinary exponential operator using the super-operator $\mathcal{J}$

$$T(e^{\int_t^{t+\Delta t} ds H(s)}) = e^{\Delta t (H(t)+\mathcal{J})} \quad . \tag{3.1.11}$$

Given the importance of this formula, we include its proof here for completeness, as illustrated in [33].

Proof: The right-hand side of (3.1.11) can be expressed as

$$\begin{aligned} e^{\Delta t (H(t)+\mathcal{J})} &= \lim_{n\to\infty} (e^{\frac{\Delta t}{n} H(t)} e^{\frac{\Delta t}{n}\mathcal{J}})^n && (3.1.12)\\ &= \lim_{n\to\infty} e^{\frac{\Delta t}{n} H(t+\Delta t)} \dots e^{\frac{\Delta t}{n} H(t+\frac{2\Delta t}{n})} e^{\frac{\Delta t}{n} H(t+\frac{\Delta t}{n})} && (3.1.13)\\ &= T(e^{\int_t^{t+\Delta t} ds H(s)}) \quad , && (3.1.14) \end{aligned}$$

where in (3.1.12) we have used the following relations recursively

$$\begin{aligned} e^{\frac{\Delta t}{n} H(t)} e^{\frac{\Delta t}{n}\mathcal{J}} &= e^{\frac{\Delta t}{n} H(t+\frac{\Delta t}{n})} \quad , && (3.1.15)\\ e^{\frac{\Delta t}{n} H(t+\frac{\Delta t}{n})} e^{\frac{\Delta t}{n} H(t)} e^{\frac{\Delta t}{n}\mathcal{J}} &= e^{\frac{\Delta t}{n} H(t+\frac{2\Delta t}{n})} e^{\frac{\Delta t}{n} H(t+\frac{\Delta t}{n})} \quad , \quad etc. && (3.1.16) \end{aligned}$$

Finally, in (3.1.14) we noticed that the product of the previous step is precisely the Riemann sum that approaches the time-ordered exponential in the limit $n \to \infty$. An alternative version of the proof can be obtained by noting that the left-hand side of (3.1.11) can be expanded as (1.3.18), and the right-hand side of (3.1.11) can be written

[2] When $F(t)$ is differentiable with respect to time t, we also use the notation

$$\mathcal{J} = \frac{\overleftarrow{\partial}}{\partial t} \quad , \tag{3.1.9}$$

where the arrow indicates differentiation of the operators that appear before this symbol.

as

$$
\begin{aligned}
&e^{\Delta t(H(t)+\mathcal{J})} = e^{\Delta t\mathcal{J}} T(e^{\int_0^{\Delta t} du H_t(u)}) \\
&= e^{\Delta t\mathcal{J}}(1 + \int_0^{\Delta t} du H_t(u) + int_0^{\Delta t} du 1 \int_0^{u_1} du_2 H_t(u_1) H_t(u_2) + \dots) \\
&= e^{\Delta t\mathcal{J}}(1 + \int_0^{\Delta t} du e^{-u\mathcal{J}} H(t) e^{u\mathcal{J}} + \int_0^{\Delta t} du 1 \int_0^{u_1} du_2 e^{-u_1\mathcal{J}} H(t) e^{(u_1-u_2)\mathcal{J}} H(t) e^{u_2)\mathcal{J}} + \dots \\
&= 1 + \int_0^{\Delta t} du H(t+u) + \int_0^{\Delta t} du_1 \int_0^{u_1} du_2 H(t+u_1) H(t+u_2) + \dots \\
&= 1 + \int_t^{t+\Delta t} ds H(s) \int_t^{t+\Delta t} ds_1 \int_t^{s_1} ds_2 H(s_1) H(s_2) + \dots \\
&= T(e^{\int_t^{t} + \Delta t\, ds H(s)}) \quad ,
\end{aligned}
$$

where we used the notation

$$
H_t(u) \equiv e^{-u\mathcal{J}} H(t) w + e^{u\mathcal{J}} \quad . \tag{3.1.17}
$$

Now that we know that any ordered exponential operator can be expressed by an ordinary exponential operator in terms of the super-operator $\mathcal{T}$, we can address the general case where $H(t)$ can be decomposed as $H(t) = A_1(t) + \cdots + A_q(t)$ [33]. Indeed, the corresponding ordered exponential is written as

$$
T(e^{\int_t^{t+\Delta t} ds H(s)}) = e^{\Delta t(A_1(t)+\cdots+A_q(t)+\mathcal{J})} \quad . \tag{3.1.18}
$$

Then, according to general decomposition theory of ordinary exponentials [32], the first-order decomposition of (3.1.18) is given by

$$
\begin{aligned}
U_1(t+\Delta t, t) &= e^{\frac{\Delta t}{2}\mathcal{J}} e^{\Delta t A_1(t)} \dots e^{\Delta t A_q(t)} e^{\frac{\Delta t}{2}\mathcal{J}} && (3.1.19) \\
&= e^{\Delta t A_1(t+\frac{\Delta t}{2})} \dots e^{\Delta t A_q(t+\frac{\Delta t}{2})} \quad , && (3.1.20)
\end{aligned}
$$

which approximates U with an error that is at most $\mathcal{O}((\Delta t)^2)$. We can improve the approximation with the second-order decomposition formula

$$
\begin{aligned}
U_2(t+\Delta t, t) &= e^{\frac{\Delta t}{2}\mathcal{J}} e^{\frac{\Delta t}{2} A_1(t)} \dots e^{\frac{\Delta t}{2} A_{q-1}(t)} e^{\Delta t A_q(t)} e^{\frac{\Delta t}{2} A_{q-1}(t)} \dots e^{\frac{\Delta t}{2} A_1(t)} e^{\frac{\Delta t}{2}\mathcal{J}} && (3.1.21) \\
&= e^{\frac{\Delta t}{2} A_1(t+\frac{\Delta t}{2})} \dots e^{\frac{\Delta t}{2} A_{q-1}(t+\frac{\Delta t}{2})} e^{\Delta t A_q(t+\frac{\Delta t}{2})} e^{\frac{\Delta t}{2} A_{q-1}(t+\frac{\Delta t}{2})} \dots e^{\frac{\Delta t}{2} A_1(t+\frac{\Delta t}{2})} \quad , && (3.1.22)
\end{aligned}
$$

that gives an error of $\mathcal{O}((\Delta t)^3)$. Higher-ordered decompositions can be built from (1.3.26), which we rewrite

$$
U_T(\Delta t, t) = U_T(t_r, t_{r-1}) U_T(t_{r-1}, t_{r-2}) \dots U_T(t_2, t_1) U_T(t_1, t) t_j = p_j \Delta t \quad , \quad j = 1, \dots, r \quad , \tag{3.1.23}
$$

for some appropriate positive integer r, where $p_1 + p_2 + \cdots + p_r = 1$. Now, we can approximate each $U_T(t_j, t_{j-1})$ to first order with $Q^{(1)}(p_j\Delta t; t_j)$ obtaining

$$U_m^{(1)}(t + \Delta t, t) = Q^{(1)}(p_r\Delta t; t_r) \dots Q^{(1)}(p_2\Delta t; t_2)Q^{(1)}(p_1\Delta t; t_1) \quad , \tag{3.1.24}$$

where, from equation (3.1.20), the explicit form of the $Q^{(1)}(p_j\Delta t; t_j)$'s is

$$Q^{(1)}(p_j\Delta t; t_j) = e^{\frac{p_j\Delta t}{2}\mathcal{J}} e^{p_j\Delta t A_1(t)} \dots e^{p_j\Delta t A_q(t)} e^{\frac{p_j\Delta t}{2}\mathcal{J}} \tag{3.1.25}$$

$$= e^{p_j\Delta t A_1(t_j)} \dots e^{p_j\Delta t A_q(t_j)} \quad , \tag{3.1.26}$$

with $t_j = t + (p_1 + p_2 + \cdots + p_{j-1} + \frac{1}{2}p_j)t$ that is the time instant where each operator is evaluated. Hence, we have

$$U(t + \Delta t, t) = U_m^{(1)}(t + \Delta t, t) + \mathcal{O}((\Delta t)^{m+1}) \quad , \tag{3.1.27}$$

and thus m is the order of the decomposition and the parameters $\{p_j\}$ can be analytically or numerically found [32, 33] so that $U_m(t + \Delta t, t)$ becomes of m-th order in Δt.
Similarly, we approximate each $U_T(t_r, t_{r-1})$ to second order as $S^{(2)}(p_j\Delta t; t_j)$. Hence, we get

$$U_m^{(2)}(t + \Delta t, t) = S^{(2)}(p_r\Delta t; t_r) \dots S^{(2)}(p_2\Delta t; t_2)S^{(2)}(p_1\Delta t; t_1) \quad , \tag{3.1.28}$$

where each operator $S^{(2)}(p_j\Delta t; t_j)$ is the second order approximant of $S(p_j\Delta t)$, and from equation (3.1.22)

$$S^{(2)}(p_j\Delta t; t_j) = e^{\frac{p_j\Delta t}{2}\mathcal{J}} e^{\frac{p_j\Delta t}{2}A_1(t)} \dots e^{\frac{p_j\Delta t}{2}A_{q-1}(t)} e^{p_j\Delta t A_q(t)} e^{\frac{p_j\Delta t}{2}A_{q-1}(t)} \dots e^{\frac{p_j\Delta t}{2}A_1(t)} e^{\frac{p_j\Delta t}{2}\mathcal{J}} \tag{3.1.29}$$

$$= e^{\frac{p_j\Delta t}{2}A_1(t_j)} \dots e^{\frac{p_j\Delta t}{2}A_{q-1}(t_j)} e^{p_j\Delta t A_q(t_j))} e^{\frac{p_j\Delta t}{2}A_{q-1}(t_j)} \dots e^{\frac{p_j\Delta t}{2}A_1(t_j)} \quad . \tag{3.1.30}$$

Finally, we obtain

$$U(t + \Delta t, t) = U_m^{(2)}(t + \Delta t, t) + \mathcal{O}((\Delta t)^{m+1}) \quad . \tag{3.1.31}$$

Note that in this second case we still have $t_j = t + (p_1 + p_2 + \cdots + p_{j-1} + \frac{1}{2}p_j)t$, but that the parameters $\{p_j\}$ in $U_m^{(2)}$ are different from those in $U_m^{(1)}$. Neverthless, they can still be determined so that $U_m^{(2)}$ may become of the m-th order of Δt. Some additional remarks on the concepts we've just discussed follow.
In contrast to the t-independent case, achieving scaling as $\mathcal{O}((\Delta t)^{m+1})$ is not always possible for arbitrarily large m. This scaling is achievable only if derivatives of all orders exist. When higher-order derivatives are undefined, Suzuki's method can still be applied to obtain error scaling as $\mathcal{O}((\Delta t)^{m+1})$; however, the maximum achievable m depends on

the existence of specific derivative orders. What we have seen is at the basis of the recursive method provided by Suzuki [35] for generating increasingly accurate approximations of $U(t+\Delta t, t)$ for a long time interval Δt. In simple terms, this method takes as input a symmetric decomposition formula $U_m(t+\Delta t)$, which approximates an ordered operator exponential $U(t+\Delta t)$ with an error no larger than $\mathcal{O}((\Delta t)^{2m+1})$. It then outputs a new symmetric approximation formula $U_{m+1}(t+\Delta t)$ with an improved error scaling, often reduced to $\mathcal{O}((\Delta t)^{2m+3})$. First of all it is important to note that Suzuki's recursive method doesn't directly approximate $U(t+\Delta t, t)$. Instead, it constructs a higher-order approximation formula from a lower-order one. This means that the method alone isn't sufficient to approximate $U(t+\Delta t, t)$; it needs to start with a suitable initial approximation in order to produce accurate results. Last but not least, it turns out that we don't always obtain a higher order decomposition formula from a lower order one. In [35], it is demonstrated that higher-order approximations can be achieved when the operator and it's derivative are sufficiently smooth. It also provides error bounds for these approximation and establish conditions for their convergence. Conversely, when $H(t)$ or its derivatives change discontinuously or contain singularities, then Suzuki's method may not yield a product formula of the expected order.
However, in this thesis we will assume that $H(t)$ are well-behaved, and will not make much use of Suzuki's recursive formula. In stead, we will settle for using (3.1.24) and (3.1.28).

3.1.3 Construction of Unitary 2-Qubits Operators

Let us recall the time-dependent Hamiltonian we are interested in, that is (2.1.4)

$$H(t) = (1-\lambda(t))H_x + \lambda(t)H \quad , \tag{3.1.32}$$

where H_x is explicitly (2.1.7) and H is (2.1.6). Our initial state is (2.2.2). Note that, in general, H_x and H do not commute, nor do the individual terms $H_{j,j+1}$ in the summation of H. To construct quantum gates for DQC, we seek 2-qubits unitary operators that accurately reproduce a discrete version of the system evolution. To do this, we make use of the techniques introduced in sections 3.1.1 and 3.1.2.
We focus on an elementary decomposition strategy. The study of this simplified decomposition is justified by the fact that, even in this case, the results we are interested in still hold true.

Hamiltonian evolution decomposition We can divide the evolution time interval in n steps of duration $\delta t = \Delta t/n$. For each time subinterval we define a time-ordered operator $Q_{j,T}(t_j, t_{j-1})$, where $t_j = t_0 + j\delta t$, whose form is entirely analogous to $U(t+\Delta t, t)$, but acts on an interval δt:

$$U_T(t_0+\Delta t, t_0) = Q_{n,T}(t_n, t_{n-1})Q_{n-1,T}(t_{n-1}, t_{n-2})\dots Q_{2,T}(t_2, t_1)Q_{1,T}(t_1, t_0) \quad , \tag{3.1.33}$$

with $j = 1, \ldots, n$. According to equation (3.1.24), each $Q_{j,T}(t_j, t_{j-1})$ can be approximated to first order

$$Q_{j,T}(t_j, t_{j-1}) \approx Q^{(1)}_{j,T}(t_j, t_{j-1}; t'_j) \quad , \tag{3.1.34}$$

up to an error $\mathcal{O}(\delta t^2) = \mathcal{O}((\frac{\Delta t}{n})^2)$. The total approximation error on n steps thus become $\mathcal{O}(\frac{(\Delta t)^2}{n})$. The right-hand side of (3.1.34) is a time-independent operator that act in the time interval $[t_{j-1}, t_j]$ and is evaluated at $t'_j = t_{j-1} + \delta t/2$. Explicitly, we can write

$$Q^{(1)}_{j,T}(t_j, t_{j-1}; t'_j) = e^{-i\delta t H(t'_j)} \quad . \tag{3.1.35}$$

Now, we redefine $H(t) = (1 - \lambda(t))H_x + \lambda(t)H$ as

$$H(t) = H^{eff}_x(t) + H^{eff}(t) \quad , \tag{3.1.36}$$

and substituting it into (3.1.35), we obtain

$$Q^{(1)}_{j,T}(t_j, t_{j-1}; t'_j) = e^{-i\delta t (H^{eff}_x(t'_j) + H^{eff}(t'_j))} \quad . \tag{3.1.37}$$

As we saw before, the two Hamiltonians in the exponential do not commute, so we have to apply Trotter formula to proceed with the approximation.

$$Q^{(1)}_{j,T}(t_j, t_{j-1}; t'_j) = e^{-i\delta t H^{eff}_x(t'_j)} e^{-i\delta t H^{eff}(t'_j)} + \mathcal{O}(\delta t^2) \quad , \tag{3.1.38}$$

and we handle the two exponentials separately.
The first term can be expand as $H^{eff}_x(t) = (1 - \lambda(t'_j)) \sum_{a=0}^{N-1} h_x \sigma^x_a$, thus all terms in the summation commute with each other. It follows that the first exponential is easily decomposed as

$$e^{-i\delta t H^{eff}_x(t'_j)} = e^{-i\delta t (1-\lambda(t'_j)) \sum_{a=0}^{N-1} h_x \sigma^x_a} \tag{3.1.39}$$

$$= \prod_{a=0}^{N-1} e^{-i\delta t (1-\lambda(t'_j) h_x \sigma^x_a} \quad . \tag{3.1.40}$$

The second exponential of (3.1.38) requires a bit of work. Let us recall the explicit form of the second Hamiltonian (2.1.6):

$$H^{eff}(t'_j) = \lambda(t'_j) \sum_{a=0}^{N-1} H_{a,a+1} \tag{3.1.41}$$

$$= \lambda(t'_j) \sum_{a=0}^{N-1} (\Delta(j) \sigma^z_a \sigma^z_{a+1}) \quad . \tag{3.1.42}$$

Note that even though each pair of operators $H_{a,a+1}$ and $H_{a+1,a+2}$ commute, the fact that they act on a common qubit means that the corresponding unitary operators can

not be applied at the same time. One potential solution to this issue consists in dividing the original Hamiltonian H in two parts:

$$H_{even} = \sum_{a=0}^{N/2-1} H_{2a,2a+1} = \sum_{a=0}^{N/2-1} \Delta(2a)\sigma_{2a}^z\sigma_{2a+1}^z \quad , \tag{3.1.43}$$

$$H_{odd} = \sum_{a=0}^{N/2-1} H_{2a+1,2a+2} = \sum_{a=0}^{N/2-1} \Delta(2a+1)\sigma_{2a+1}^z\sigma_{2a+2}^z \quad , \tag{3.1.44}$$

where $2a$ and $2a+1$ are labels for even and odd qubits in the chain, respectively. Note that each $H_{2a,2a+1}$ in (3.1.43) acts on qubits that are not shared with any other interaction term, therefore all $H_{2a,2a+1}$'s commute with each other. The same happens in (3.1.44). Then, we can rewrite the exponential $e^{-i\delta t H^{eff}(t'_j)}$ as

$$e^{-i\delta t H^{eff}(t'_j)} = e^{-i\delta t\lambda(t'_j)H} \tag{3.1.45}$$

$$= e^{-i\delta t\lambda(t'_j)H_{even}} e^{-i\delta t\lambda(t'_j)H_{odd}} \tag{3.1.46}$$

$$= \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a+1,2a+2}} \quad . \tag{3.1.47}$$

Therefore, the explicit form of the operator $Q_{j,T}^{(1)}$ becomes

$$Q_{j,T}^{(1)}(t_j, t_{j-1}; t'_j) = \prod_{a=0}^{N-1} e^{-i\delta t(1-\lambda(t'_j))h_x\sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a+1,2a+2}} + \mathcal{O}(\delta t^2) \quad . \tag{3.1.48}$$

Since the decomposition is repeated over n steps, we will have that

$$U_T(t_0+\Delta t, t_0) \approx \prod_{j=1}^{n}\{\prod_{a=0}^{N-1} e^{-i\delta t(1-\lambda(t'_j))h_x\sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\delta t\lambda(t'_j)H_{2a+1,2a+2}}\} + \mathcal{O}(n\delta t^2) \quad . \tag{3.1.49}$$

Finally, this form of U_T can be straightforwardly implemented in terms of quantum gates: indeed, up to a small error, it is decomposed as the product of one- and two-qubit unitary operators, which can be implemented as local quantum gates.

To end this section, we recall that the efficiency of QA hinges on the annealing time Δt, which is determined by the smallest spectral gap encountered during the process.

A narrow spectral gap, especially during first-order phase transitions, can significantly slow down the computation, as a slower evolution is required to maintain the system in its ground state. This limitation can make QA less effective for certain challenging problems. Several strategies have been proposed to mitigate this, such as using heuristic choices for the initial state, modifying the evolution to enlarge the minimum gap, or avoiding problematic first-order transitions altogether. However, these methods often require prior knowledge of the spectral gap, which is not always available.

DQA still inherits the core adiabatic limitations of QA—particularly the dependency on the spectral gap. If the gap becomes vanishingly small, the evolution time can stretch to infinity to ensure the system remains in its ground state, making it impractical for certain complex problems.

In contrast, the Quantum Approximate Optimization Algorithm (QAOA) presents a fundamentally different, non-adiabatic approach. As we show in the next section, QAOA uses a parameterized quantum circuit composed of discrete unitary gates to approximate solutions to combinatorial optimization problems. The algorithm iteratively refines parameters through a classical optimization routine to find the best solution. Unlike QA and DQA, QAOA does not rely on a slow, continuous evolution and is therefore not constrained by small spectral gaps. The effectiveness of QAOA depends instead on the circuit depth p and the number of iterations needed for the classical optimization to converge.

Chapter 4

QAOA

The Quantum Approximate Optimization Algorithm (QAOA) is a hybrid quantum-classical algorithm designed to solve combinatorial optimization problems. It has gained significant attention due to its potential to outperform classical algorithms in finding solutions to complex optimization tasks, particularly as quantum hardware advances.
We recall section 2.1 where we introduced the cost function $C(z)$ and how to map it into a quantum Hamiltonian. We can map $C(z)$ into H such that the desired solution corresponds to the eigenstate (once measured it will appear as a string of bits) belonging to the minimum/maximum eigenvalue of system's Hamiltonian. QAOA is an iterative algorithm that uses quantum states superposition to explore the solution space of a given problem, weighs various solutions and gradually converge towards the optimal or near-optimal one through the application of quantum gates.
In this context, *Satisfiability* problems focus on determining if there exists an assignment of binary variables, denoted as a string z, that satisfies a set of logical clauses $C_\alpha(z)$. *MaxSat*, a more challenging variant, aims to find the assignment z that maximizes the objective function $C(z)$, reaching its optimal value $C_{max}(z)$. Unlike exact algorithms, approximate optimization algorithms like QAOA are designed to find a solution z such that the resulting $C(z)$ is as close as possible to $C_{max}(z)$.

4.1 Overview and Cost and Mixing Operators

QAOA operates by repeatedly applying two evolution operators, namely $U(H, \gamma)$ and $U(B, \beta)$, to the system. This process effectively produces a trotterized approximation of the final state in the Quantum Adiabatic Algorithm.
Further, the algorithm depends on an integer positive parameter p that is the depth of the quantum circuit used. Specifically, it indicates the number of alternating layers of quantum gates applied to the quantum state to approximate the solution to the combinatorial optimization problem. As p increases, the quality of the approximation improves, but at the cost of greater computational complexity.

Again, we deal with a computational problem whose cost function can be mapped into the custom Hamiltonian of a N-particles quantum ad-hoc model (2.1.6), but this time we will not use (2.2.2) as initial state. Indeed, we define the system initial state as

$$|s\rangle = |+\rangle^{\otimes N} = \frac{1}{\sqrt{2^N}} \sum_z |z\rangle \quad . \tag{4.1.1}$$

This represent the superposition of all possible bit strings $|z\rangle$ for a system of N qubits, each of which can be in state $|0\rangle$ or $|1\rangle$. The coefficient $\frac{1}{\sqrt{2^N}}$ ensures that $|s\rangle$ is properly normalized.
This state is easy to initialize, as $|s\rangle = H^{\otimes N} |0\rangle^{\otimes N}$, where $H^{\otimes N}$ is the tensor product of N Hadamard operators and $|0\rangle$ is he initial state of each qubit of the string[1].
We now define two unitary operators at the basis of QAOA protocol:

1. The cost operator
$$U(H, \gamma) = e^{-i\gamma H} \quad , \quad \gamma \in \mathbb{R} \quad . \tag{4.1.2}$$
$U(H, \gamma)$ encodes information about the objective function in the evolution of the quantum state, while γ is a tunable parameter. This operator applies a phase shift to each computational basis state $|z\rangle$, with the phase being proportional to the corresponding value of the Hamiltonian H, hence $U(H, \gamma) |z\rangle = e^{-i\gamma H(z)} |z\rangle$. It acts like a rotation of angle γ on the quantum state around the $\hat{z}$ axis in the Bloch sphere. This means that states with lower $H(z)$ values (corresponding to better solutions) receive smaller phase shifts, while states with higher energy values (worse solution) receive larger phase shifts: in this way, better solutions have more "weight" in the final optimization.
We recall that, since the algorithm must be implemented on a quantum computer with at most 2-qubit gates and the $H_{a,a+1}$'s terms in H, share a qubit pairwise as seen in previous sections, we need to apply a decomposition similar to the one already discussed. Thus, the cost operator will have the form
$$U(H, \gamma) = \prod_{a=0}^{N/2-1} e^{-i\gamma H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\gamma H_{2a+1,2a+2}} \quad . \tag{4.1.3}$$
We have just kept part of the decomposed operator (3.1.48) but we have not included any kind of time dependence.

[1]Note that the initial state we used for the QA and DQA protocols (2.2.2) is different from $|s\rangle$ in (4.1.1), though both are uniform superpositions. The key difference lies in the relative phases: $|s\rangle$ has all positive amplitudes, while (2.2.2) can be rewritten as $|-\rangle^{\otimes N} = \frac{1}{\sqrt{s^N}} \sum_{z \in 0,1^N} (-1)^{Hamming(z)} |z\rangle$, where *Hamming(z)*represents the number of 1's in the bit string z, and the phase $(-1)^{Hamming(z)}$ comes from the tensor product of the individual $|-\rangle$ states.

2. The "mixing" operator

$$U(H_x, \beta) = e^{-i\beta H_x} = \prod_{a=0}^{N-1} e^{-i\beta h_x \sigma_j^x} \quad , \quad \beta \in [0, \pi] \quad . \tag{4.1.4}$$

It acts on each qubit j with the "*flip operator*" σ_j^x, which satisfies the eigenvalues equations: $\sigma_j^x |0\rangle = +|1\rangle$ and $\sigma_j^x |1\rangle = +|0\rangle$. It facilitates the transition between different basis states, leading to fluctuations across all potential configurations of qubits when applied to a multi-body system. In fact, from Pauli exponential formula:

$$e^{-i\beta\hat{\sigma}_j^x} |0\rangle = \cos(\beta) |0\rangle - i \sin(\beta) |1\rangle \quad , \quad e^{-i\beta\hat{\sigma}_j^x} |1\rangle = \cos(\beta) |1\rangle - i \sin(\beta) |0\rangle \quad . \tag{4.1.5}$$

The above exponentials create quantum superpositions among various potential configurations, subsequently 'mixing' the probabilities associated with each quantum state $|z\rangle$. The angle β governs the amplitude of rotation about the $\hat{x}$-axis, thereby influencing the level of superposition. When the operator (4.1.4) is applied repeatedly across multiple layers, it facilitates exploration of the solution space and prevents the system from being trapped in a specific configuration that may represent a non-optimal local minimum.

QAOA alternates between applying the operators (4.1.2) and (4.1.4) for p cycles. This alternation aims to achieve a balance between exploring known solutions and exploring new ones, crucial for effectively approximating the optimal solution to a given combinatorial problem.
For any integer $p \geq 1$ and $2p$ angles $(\gamma_1, ..., \gamma_p) \equiv \boldsymbol{\gamma}$ and $(\beta_1 ... \beta_p) \equiv \boldsymbol{\beta}$ let us define the angle-dependent quantum state:

$$\begin{aligned} |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle &= U(H_x, \beta_p) U(H, \gamma_p) \dots U(H_x, \beta_1) U(H, \gamma_1) |s\rangle \\ &= \prod_{a=0}^{N-1} e^{-i\beta_p h_x \sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\gamma_p H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\gamma_p H_{2a+1,2a+2}} \\ &\qquad \dots \\ &\qquad \prod_{a=0}^{N-1} e^{-i\beta_1 h_x \sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\gamma_1 H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\gamma_1 H_{2a+1,2a+2}} |s\rangle \quad . \end{aligned} \tag{4.1.6}$$

It can be proved that $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ can be produced by a quantum circuit of depth at most $mp + p$. Finally, let $E_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$ be the expectation value of H in the final state:

$$E_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = \langle \boldsymbol{\gamma}, \boldsymbol{\beta}| H |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle \quad , \tag{4.1.7}$$

and m_p be the minimum value of (4.1.7) over the angles:

$$m_p = \min_{\boldsymbol{\gamma},\boldsymbol{\beta}} E_p(\boldsymbol{\gamma},\boldsymbol{\beta}) \quad . \tag{4.1.8}$$

We have defined all quantities in the algorithm.
A simplistic explanation of the latter is useful to understand its general purpose: we pick a certain p and a set $(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$, which controls the evolution operators in the quantum circuit and somehow makes $E_p(\boldsymbol{\gamma},\boldsymbol{\beta})$ as small as possible. Now, we run the quantum circuit associated with QAOA for p times (depth of the algorithm). The circuit alternates between the cost Hamiltonian (4.1.2) and the mixing Hamiltonian (4.1.4) for each pair of angles $(\gamma_i^*,\beta_i^*), i = 0,...,p$, and yields the quantum state $|\boldsymbol{\gamma^*},\boldsymbol{\beta^*}\rangle$. We measure this state in the computational basis several times, say M times, and get multiple bit-strings $z^j, j = 1,...,M$, each of which allows us to compute $C(z)$. $E_p(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$ is evaluated as the average value over the $C(z)$'s in order to mitigate statistical noise and provide a better estimate of the expected outcome. This process yields an error on E_p that scales as $\mathcal{O}(1/\sqrt{M})$.
In the general case, when the number of layers p is not fixed, we may need to execute the algorithm multiple times and iterate through various updated sets of angles $(\boldsymbol{\gamma},\boldsymbol{\beta})$ before we get to the optimal one $(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$. This optimal set is the one that brings $E_p(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$ sufficiently close to m_p. Hence, the process involves not only finding the optimal parameters $(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$ for each layer of the quantum circuit but also determining the optimal number of layers p, adding an additional dimension to the search space and increasing the complexity of the optimization task.
In contrast, when p is fixed — a case we will explore later — this situation presents a straightforward exception to the typical procedure. In this scenario, we can first run a classical algorithm to efficiently determine $(\boldsymbol{\gamma^*},\boldsymbol{\beta^*})$ and then execute the quantum algorithm only once.
It could seem trivial, but let us recall that computing the mean of the final samples is significant due to the probabilistic nature of quantum algorithms. Each time we run the algorithm, we measure $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ and get a different string z since the measurement is random and depends on the probability distribution associated with the system. In the distribution, strings z that yield a lower value of $C(z)$ are more likely to occur. So, QAOA does not guarantee you to get the optimal string with each run. Instead, it improves the distributions of strings z by optimizing the parameters $(\boldsymbol{\gamma},\boldsymbol{\beta})$. If the mean value improves, then the distribution is shifting toward configuration of z which minimize $C(z)$. At the end of the optimization process we can run the circuit some more times and select the string z that produces the lowest value of $C(z)$ among the measured outcomes.

4.2 Classical optimization of the angles $(\boldsymbol{\gamma}, \boldsymbol{\beta})$

As described in [10] there are many methods that can be implemented on classical computers to pick good sets $(\boldsymbol{\gamma}, \boldsymbol{\beta})$. The article considers a specific maximization problem, the MaxCut problem on graphs with bounded degrees (so that we have reduced local complexity around each vertex), and shows efficient classical calculations of general validity to find an optimal set of angles that minimize[2] $E_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$.

Fixed-p case

If we consider the scenario in which p doesn't grow with n (p is fixed while the number of variables increase) we can adopt the strategy used in [10]. We will see an adapted version for application on a spin chain, rather than on graphs, and we will mention the graph case as a generalization of the spin chain.
We can write H as $H = \sum_{j=0}^{N-1} H_{j,j+1}$, thus the explicit form of (4.1.7) becomes

$$\begin{aligned}
E_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) &= \langle s| U^\dagger(H,\gamma_1)\dots U^\dagger(H_x,\beta_p) H U(H_x,\beta_p)\dots U(H,\gamma_1)\,|s\rangle \\
&= \sum_{j=0}^{N-1} \langle s| U^\dagger(H,\gamma_1)\dots U^\dagger(H_x,\beta_p) H_{j,j+1} U(H_x,\beta_p)\dots U(H,\gamma_1)\,|s\rangle \\
&= \sum_{j=0}^{N-1} \langle s| \prod_{a=0}^{N/2-1} e^{i\gamma_1 H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{i\gamma_1 H_{2a+1,2a+2}} \prod_{a=0}^{N-1} e^{i\beta_1 h_x \sigma_a^x} \dots \\
&\qquad \dots \prod_{a=0}^{N/2-1} e^{i\gamma_p H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{i\gamma_p H_{2a+1,2a+2}} \prod_{a=0}^{N-1} e^{i\beta_p h_x \sigma_a^x} H_{j,j+1} \\
&\qquad \prod_{a=0}^{N-1} e^{-i\beta_p h_x \sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\gamma_p H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\gamma_p H_{2a+1,2a+2}} \dots \\
&\qquad \dots \prod_{a=0}^{N-1} e^{-i\beta_1 h_x \sigma_a^x} \prod_{a=0}^{N/2-1} e^{-i\gamma_1 H_{2a,2a+1}} \prod_{a=0}^{N/2-1} e^{-i\gamma_1 H_{2a+1,2a+2}} |s\rangle \quad . \qquad (4.2.1)
\end{aligned}$$

In each term of the sum above, the operator

$$U^\dagger(H,\gamma_1)\dots U^\dagger(H_x,\beta_p) H_{j,j+1} U(H_x,\beta_p)\dots U(H,\gamma_1) \quad , \qquad (4.2.2)$$

[2] Actually, the article of reference aims to maximize E_p. Obviously, the two situations are completely analogous, except for a global "$-$" sign, thus we have decided to remain consistent with the problem discussed throughout the thesis.

involves only the j-th and $j+1$-th spins and those ones at most p steps away from j and $j+1$. We can easily see this for the $p=1$ case, where (4.2.2) is

$$U^\dagger(H,\gamma_1)U^\dagger(H_x,\beta_1)H_{j,j+1}U(H_x,\beta_1)U(H,\gamma_1) \quad . \tag{4.2.3}$$

Now, the factors in $U(H_x,\beta_1)$ which do not involve spins j or $j+1$ commute through $H_{j,j+1}$ and we get

$$U^\dagger(H,\gamma_1)e^{i\beta_1(\sigma_j^x+\sigma_{j+1}^x)}H_{j,j+1}e^{-i\beta_1(\sigma_j^x+\sigma_{j+1}^x)}(H,\gamma_1) \quad . \tag{4.2.4}$$

Analogously, any factors in the operator $U(H,\gamma_1)$ which do not involve qubits j or $j+1$ will commute through and cancel out. So the operator in equation (4.2.4) only involves qubits j or $j+1$ and those adjacent to j or $j+1$. We can generalize for any p, and conclude that the operator in (4.2.2) that represent a general term of the sum in (4.2.1) depends only on the local subchain $g(j,j+1)$ of size $2p$ (as the interaction degree is always 2 - or 1 for boundary spins - for each spin), which is small and independent of N, involving spins j and $j+1$ and spins at most p steps away from j or $j+1$. For each subchain $g(j,j+1)$, we define the the restriction of the following elements to the qubits in $g(j,j+1)$, starting from the cost Hamiltonian

$$H^{g(j,j+1)} = \sum_{\substack{j=0 \\ j,j+1\in g(j,j+1)}}^{N-1} H_{j,j+1} \quad , \tag{4.2.5}$$

the operators

$$U(H^{g(j,j+1)},\gamma) = \prod_{\substack{a=0 \\ 2a,2a+1\in g(j,j+1)}}^{N/2-1} e^{-i\gamma H_{2a,2a+1}} \prod_{\substack{a=0 \\ 2a+1,2a+2\in g(j,j+1)}}^{N/2-1} e^{-i\gamma H_{2a+1,2a+2}} \quad , \tag{4.2.6}$$

$$U(H_x^{g(j,j+1)},\beta) = \prod_{\substack{a=0 \\ a\in g(a,a+1)}}^{N-1} e^{-i\beta h_x\sigma_a^x} \quad , \tag{4.2.7}$$

and the subchain initial state $|s,g(j,j+1)\rangle$

$$|s,g(j,j+1)\rangle = \prod_{j\in g(j,j+1)} |+\rangle_j \quad . \tag{4.2.8}$$

Using this, we can rewrite the expectation value $E_p(\gamma,\beta)$ for each term in terms of the local subchains. Return to (4.2.1): each j-th spin of our model belongs to a g-type subgraph $g(j,j+1)$, so let us define

$$\begin{aligned} e_g(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle s,g(j,j+1)|\, U^\dagger(H^{g(j,j+1)},\gamma_p)\ldots U^\dagger(H_x^{g(j,j+1)},\beta_1) \\ H_{j,j+1}U(H_x^{g(j,j+1)},\beta_1)\ldots U(H^{g(j,j+1)},\gamma_p)\,|s,g(j,j+1)\rangle \quad , \end{aligned} \tag{4.2.9}$$

which is the contribution of each j to (4.2.1).
Note that if $g(j, j+1)$ and $g(j', j'+1)$ give rise to isomorphic subsystems, the corresponding functions of (γ, β) are the same. Our case is trivial because we have subchains of dimension $2p$ and bounded degrees, hence they are obviously all isomorphic. Therefore, we only need to calculate E_p through the contribution of one representative subchain type g, using w_g as the number of its occurences in the original sum:

$$E_p(\boldsymbol{\gamma}, \boldsymbol{\beta}) = w_g e_g(\boldsymbol{\gamma}, \boldsymbol{\beta}) \quad . \tag{4.2.10}$$

As stated in [10], the value of (4.2.10) can be evaluated on a classical computer since the max number of qubits involved in (4.2.9) is determined by the size of the subchain. Thus, the Hilbert space grows as 2^{2p} for a subchain of size $2p$, and is independent of N. For large values of p, the optimization may exceed current classical capabilities. However, it is important to note that the required resources do not scale with the number of qubits N. Therefore, the complexity remains manageable even for larger systems, making it feasible to optimize for small p and run the quantum algorithm just once.
We can extend the QAOA algorithm by designing it for graphs instead of chains. By considering an input graph with N nodes representing spins and a set of m edges $\{(j, k)\}$ representing interactions between arbitrary pairs of qubits j, k, we can generalize models as the one we used to more complex configurations. While in spin chains interactions occur linearly between adjacent qubits, spin graphs allow arbitrary interactions, expanding the applicability of QAOA to optimization problems with more complex topologies, beyond just linear spin chains. Even in the case of graphs, if the parameter p is fixed or scales slowly with N, we can first pre-determine the algorithm's angle parameters (γ, β) and then run QAOA just once. In particular, we will have different kind of subgraphs and we can write the analogous of (4.2.10) as

$$E_p(\gamma, \beta) = \sum_g w_g e_g(\boldsymbol{\gamma}, \boldsymbol{\beta}) \quad , \tag{4.2.11}$$

and each $e_g(\boldsymbol{\gamma}, \boldsymbol{\beta})$ involves a number of qubits that scales as

$$q_{tree} = 2\frac{(v-1)^{p+1} - 1}{(v-1) - 1} \tag{4.2.12}$$

(v is the maximum degree of the graph). Thus, each (4.2.9) works in a Hilbert space whose dimension is at most $2^{q_{tree}}$. Further, there are only finitely many subgraph types for each p.
This makes the approach computationally efficient, as the resource requirements do not grow with N, allowing the optimization of the angles to be done classically before applying the quantum algorithm, reducing the need for multiple quantum runs.

4.3 Comparison between QA and QAOA performances

We now recall some important differences between QA and QAOA as reported in [10]. Consider a time dependent Hamiltonian of the same kind of (3.1.32), let us say

$$H(t) = \left(1 - \frac{t}{T}\right) H_x + (t/T)H \quad , \tag{4.3.1}$$

where T represents the total runtime. It is evident that at $t = 0$ (4.3.1) is dominated by H_x, whose fundamental eigenstate $|s\rangle = |+\rangle^{\otimes n}$ (associated to the minimum eigenvalues) is easy to initialize on a quantum computer. Note that $|s\rangle$ is the minimum energy eigenstate for H_x but not necessarily in H (it can be an high energy state for H). Since H_x has only non-negative off-diagonal elements, the ground state is non-degenerate and the energy gap between the ground state and the first excited state is > 0 for any $t < T$ (by the Perron-Frobenius theorem). Then, for T large enough, the QA algorithm will succesfully yield the optimal solution.
Instead, QAOA makes use of an alternation of the cost operator and the mixing operator. The sum of γ and β angles is analogous to the total evolution time. For a better approximation, we want each angle to be as small as possible and a longer runtime (this approximates the adiabatic evolution limit). These two conditions combined force p to be large (perhaps, exponentially large in some cases). In [10], it is discussed that there always exists p large enough and a set $(\boldsymbol{\gamma}, \boldsymbol{\beta})$ such that $E_p(\boldsymbol{\gamma}, \boldsymbol{\beta})$ is as close to m_p as desired. Further, at $p - 1$ we have $m_p \leq m_(p - 1))$, that can be viewed as a constrained minimization at p. Hence,

$$\lim_{p\to\infty} m_p = \min_z C(z) \quad , \tag{4.3.2}$$

which forms the theoretical foundation of the QAOA algorithm.
There are some other subtle distinctions between QA and QAOA.
QA works by generating a state that exhibits a large overlap with the optimal string, while QAOA aims to provide a good approximation of the optimal solution, but is not necessary for the two states to share a significant overlap. For instance, at $p = 1$, QAOA achieves a $\frac{3}{4}$ approximation rate, yet the overlap with the optimal z is an exponentially small overlap [10].
It is also noteworthy that in QA, the success probability is not a monotonic function of the runtime T. This is studied in [8], for a particular 20-qubit-instances of Max2Sat. In QAOA the quality of approximation improves as p increases.
To end this section we report to the reader the results of the analysis carried out in [21], which presents slightly modified version of QAOA specifically adapted for studying the ground state of molecular systems.[3].

[3]The chemical adaptation of QAOA differs from the standard version in several key aspects. It uses the Hartree-Fock state as the initial state, which reduces the complexity of the Hilbert space

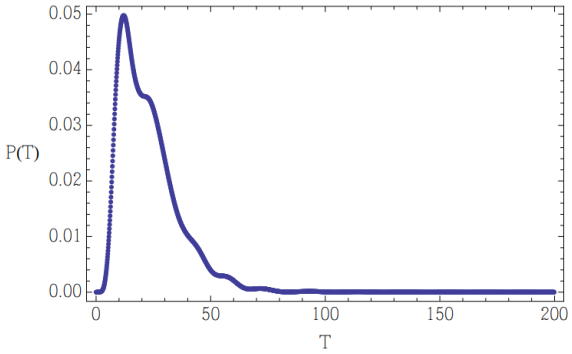

Figure 4.1: The success probability in a QA as a function of total evolution time T, from [8].The probability increases initially but then drops sharply, with the eventual rise for large T remaining unseen within simulation times that are computationally practical

Figure 4.2 presents the performance of a "chemical" QAOA in producing final states with a good squared overlap with the ground state of the molecular system. While QAOA can achieve good energy minimization without necessarily obtaining a good overlap, ensuring this overlap, similar to the case of QAA, indirectly leads to effective energy minimization. These images are significant because, as shown by their similarity to the performance diagrams of a standard QAOA in Figure 4.3, the results are general and do not depend on the QAOA adaptation for specific molecular chemistry problems, nor on the choice of cost Hamiltonian, mixing Hamiltonian, or the initial state.
In the chemical QAOA, the angles sets γ and β are defined as

$$(\gamma, \beta) = \{(\gamma_j, \beta_j)\} \quad , \quad j = 1, \ldots, p \quad , \tag{4.3.3}$$

$$\gamma_j = \gamma(f_j) = \Delta f_j \quad , \quad \beta_j = \beta(f_j) = \Delta(1 - f_j) \quad , \tag{4.3.4}$$

$$f_j = \frac{j}{p+1} \quad , \tag{4.3.5}$$

hence, the parameter Δ simplifies the optimization process by combining the two distinct parameters sets (γ, β), into one. This approach reduces computational complexity by optimizing a single parameter that controls both the cost and mixing Hamiltonians. In standard QAOA, Δ can typically be interpreted as a step size used to adjust the evolution over layers, balancing exploration and exploitation. A smaller Δ results in a more gradual evolution, while a larger p increases the number of layers for better approximation.

by incorporating prior knowledge about the molecular system. Unlike the standard QAOA, which typically starts from an equal superposition state and evolves to find a single eigenstate approximating the ground state, the chemical version works with a non-diagonal cost Hamiltonian. The algorithm's goal is to approximate the ground state as a linear superposition of eigenstates, and its performance is evaluated through the squared overlap between the output state and the true ground state, rather than focusing solely on energy minimization.

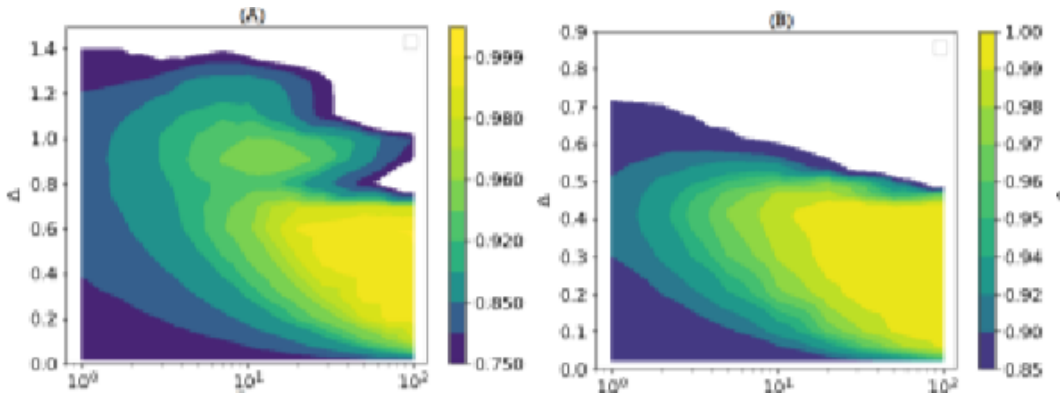

Figure 4.2: (Right) P2 with initial squared overlap 0.77; (Left) CO2 with initial squared overlap 0.85, from [21]. White regions above the colored areas represent parameter settings where squared overlap with target ground state is less than its initial value.

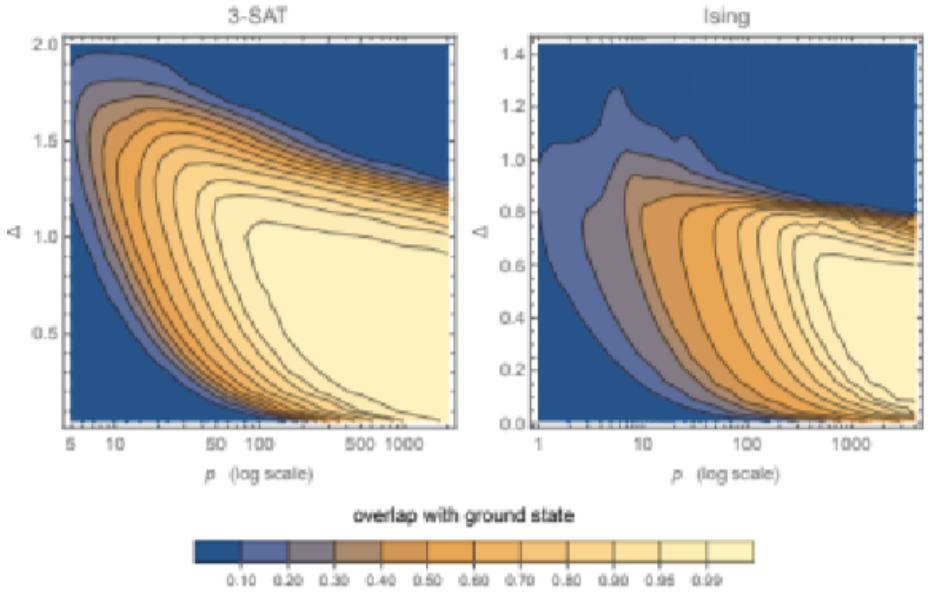

Figure 4.3: Squared overlap with the ground state(s) after completing QAOA is shown as a function of the parameters Δ and p. The left plot corresponds to a 20-variable random 3-SAT instance with 80 clauses, while the right depicts results for a 6-variable fully connected Ising spin system. Diagrams from [21].

Conclusions

This thesis provides an in-depth analysis of the quantum adiabatic algorithm (QAA) and its digital implementations, emphasizing their theoretical underpinnings and potential applications in solving computationally complex problems. By leveraging the adiabatic theorem, QAA establishes a framework where a quantum system can be guided from an initial state to the ground state of a target Hamiltonian, which encodes the solution to an optimization problem. The digital adaptations, particularly Digital Quantum Annealing (DQA), offer an avenue to implement this approach on gate-based quantum devices, making it accessible for the Noisy Intermediate-Scale Quantum (NISQ) era.
The study also highlights the relevance of the Quantum Approximate Optimization Algorithm (QAOA), which shares conceptual similarities with QAA but introduces a discrete, hybrid quantum-classical approach. While QAOA offers flexibility and modularity, challenges in parameter optimization remain a significant barrier, especially as circuit depth increases. Similarly, DQA faces constraints in maintaining adiabaticity and mitigating errors, yet it demonstrates promising adaptability to existing quantum hardware.
A comparative analysis reveals that both DQA and QAOA retain the core principles of adiabatic evolution while addressing practical limitations such as noise and hardware imperfections. This positions them as strong candidates for achieving quantum advantage in optimization problems, even under current technological constraints. However, scaling these algorithms to larger systems or more complex problems requires a deeper understanding of spectral gaps, error resilience, and hybrid optimization techniques.
The thesis underscores the transformative potential of adiabatic-based quantum computation. By encoding combinatorial optimization problems into quantum Hamiltonians, these methods exploit quantum mechanics' inherent parallelism and tunneling properties. Despite the challenges, their adaptability to current quantum hardware and their theoretical scalability highlight their value in both academic research and real-world applications. Future directions include refining algorithms to optimize performance on NISQ devices, developing advanced parameter selection strategies for QAOA, and improving error mitigation techniques for DQA. Additionally, research into hybrid frameworks that combine quantum and classical resources more effectively could unlock new capabilities. Ultimately, this thesis demonstrates that adiabatic-based quantum approaches, especially in digital and variational forms, are pivotal steps toward realizing practical quantum computation.

Bibliography

[1] Tameem Albash and Daniel A Lidar. "Adiabatic quantum computation". In: *Reviews of Modern Physics* 90.1 (2018), p. 015002.

[2] Mohammad HS Amin. "Consistency of the adiabatic theorem". In: *Physical review letters* 102.22 (2009), p. 220401.

[3] Joseph E Avron and Alexander Elgart. "Adiabatic theorem without a gap condition". In: *Communications in mathematical physics* 203 (1999), pp. 445–463.

[4] Bonnie Berger and Tom Leighton. "Protein folding in the hydrophobic-hydrophilic (HP) is NP-complete". In: *Proceedings of the second annual international conference on Computational molecular biology*. 1998, pp. 30–39.

[5] Sergio Blanes et al. "The Magnus expansion and some of its applications". In: *Physics reports* 470.5-6 (2009), pp. 151–238.

[6] Jean-Philippe Bouchaud. "Crises and collective socio-economic phenomena: simple models and challenges". In: *Journal of Statistical Physics* 151 (2013), pp. 567–606.

[7] Joseph D Bryngelson and Peter G Wolynes. "Spin glasses and the statistical mechanics of protein folding." In: *Proceedings of the National Academy of sciences* 84.21 (1987), pp. 7524–7528.

[8] Elizabeth Crosson et al. "Different strategies for optimization using the quantum adiabatic algorithm". In: *arXiv preprint arXiv:1401.7320* (2014).

[9] Alexander Elgart and George A Hagedorn. "A note on the switching adiabatic theorem". In: *Journal of Mathematical Physics* 53.10 (2012).

[10] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm". In: *arXiv preprint arXiv:1411.4028* (2014).

[11] Richard P Feynman. "Simulating physics with computers". In: *Feynman and computation*. cRc Press, 2018, pp. 133–153.

[12] Fabio Franchini et al. *An introduction to integrable techniques for one-dimensional quantum systems*. Vol. 940. Springer, 2017.

[13] Yimin Ge, András Molnár, and J Ignacio Cirac. "Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states". In: *Physical review letters* 116.8 (2016), p. 080503.

[14] John J Hopfield. "Neural networks and physical systems with emergent collective computational abilities." In: *Proceedings of the national academy of sciences* 79.8 (1982), pp. 2554–2558.

[15] Siddharth Jain. "Solving the traveling salesman problem on the d-wave quantum computer". In: *Frontiers in Physics* 9 (2021), p. 760783.

[16] Sabine Jansen, Mary-Beth Ruskai, and Ruedi Seiler. "Bounds for the adiabatic approximation with applications to quantum computation". In: *Journal of Mathematical Physics* 48.10 (2007).

[17] Tadashi Kadowaki and Hidetoshi Nishimori. "Quantum annealing in the transverse Ising model". In: *Physical Review E* 58.5 (1998), p. 5355.

[18] Tosio Kato. "On the adiabatic theorem of quantum mechanics". In: *Journal of the Physical Society of Japan* 5.6 (1950), pp. 435–439.

[19] Joris Kattemölle and Jasper Van Wezel. "Variational quantum eigensolver for the Heisenberg antiferromagnet on the kagome lattice". In: *Physical Review B* 106.21 (2022), p. 214429.

[20] Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. "Optimization by simulated annealing". In: *science* 220.4598 (1983), pp. 671–680.

[21] Vladimir Kremenetski et al. "Quantum alternating operator ansatz (QAOA) phase diagrams and applications for quantum chemistry". In: *arXiv preprint arXiv:2108.13056* (2021).

[22] Seth Lloyd. "Universal quantum simulators". In: *Science* 273.5278 (1996), pp. 1073–1078.

[23] Andrew Lucas and Ching Hua Lee. "Multistable binary decision making on networks". In: *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics* 87.3 (2013), p. 032806.

[24] Wilhelm Magnus. "On the exponential solution of differential equations for a linear operator". In: *Communications on pure and applied mathematics* 7.4 (1954), pp. 649–673.

[25] Glen Bigan Mbeng, Angelo Russomanno, and Giuseppe E Santoro. "The quantum Ising chain for beginners". In: *SciPost Physics Lecture Notes* (2024), p. 082.

[26] Quantum Mechanics Messiah. *Vol. II, Chap. XXI. 13.* 1962.

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

[28] John Preskill. "Quantum computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[29] Peter W Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th annual symposium on foundations of computer science*. Ieee. 1994, pp. 124–134.

[30] Karthik Srinivasan et al. "Efficient quantum algorithm for solving travelling salesman problem: An IBM quantum experience". In: *arXiv preprint arXiv:1805.10928* (2018).

[31] Masuo Suzuki. "Generalized Trotter's formula and systematic approximants of exponential operators and inner derivations with applications to many-body problems". In: *Communications in Mathematical Physics* 51.2 (1976), pp. 183–190.

[32] Masuo Suzuki. "Fractal decomposition of exponential operators with applications to many-body theories and Monte Carlo simulations". In: *Physics Letters A* 146.6 (1990), pp. 319–323. ISSN: 0375-9601. DOI: https://doi.org/10.1016/0375-9601(90)90962-N. URL: https://www.sciencedirect.com/science/article/pii/037596019090962N.

[33] Masuo Suzuki. "General decomposition theory of ordered exponentials". In: *Proceedings of the Japan Academy, Series B* 69.7 (1993), pp. 161–166.

[34] Hale F Trotter. "On the product of semi-groups of operators". In: *Proceedings of the American Mathematical Society* 10.4 (1959), pp. 545–551.

[35] Nathan Wiebe et al. "Higher order decompositions of ordered operator exponentials". In: *Journal of Physics A: Mathematical and Theoretical* 43.6 (2010), p. 065203.