UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS FÍSICAS

TESIS DOCTORAL

Algoritmos cuánticos tolerantes a fallos

Fault tolerant quantum algorithms

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Pablo Antonio Moreno Casares

Director

Miguel Ángel Martín-Delgado Alcántara

Madrid

UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE CIENCIAS FÍSICAS

TESIS DOCTORAL

Algoritmos cuánticos tolerantes a fallos

Fault tolerant quantum algorithms

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Pablo Antonio Moreno Casares

DIRECTOR

Miguel Ángel Martín-Delgado Alcántara

Fault tolerant quantum algorithms

Algoritmos cuánticos tolerantes a fallos

Pablo Antonio Moreno Casares

Director: Miguel Ángel Martín-Delgado Alcántara

Departamento de Física Teórica
Facultad de Ciencias Físicas
Universidad Complutense de Madrid

Memoria para optar al título de
Doctor en Física

October 2022

Me gustaría dedicar esta tesis a mis abuelos que siempre han sido una fuente de inspiración, y mis padres y Pili, por su apoyo constante.

Acknowledgements

En primer lugar me gustaría agradecer al director de tesis, Miguel Ángel, por su dirección a lo largo de esta tesis, y muy en particular por la confianza depositada a la hora de elegir proyectos de investigación. Creo que esta tesis representa un éxito de supervisión por su parte. Así mismo, me gustaría agradecer a los miembros de nuestro grupo de investigación, como Roberto Campos o Santiago Varona, y con otros estudiantes del departamento de física, como Roberto Ruiz, que en ocasiones me han sugerido ideas sobre cómo abordar problemas técnicos. Además, quiero agradecer especialmente a Santiago Varona su disponibilidad para leer esta tesis con antelación y sugerir correcciones y cambios, lo que se ha reflejado en una mejora de la redacción y coherencia del texto.

También quiero dar las gracias a las personas con las que he colaborado en diversos artículos. Por un lado, el grupo de Juan Miguel (Alain, Modji, Roberto...) en xanadu.ai por su confianza y tratarme como parte del equipo. Además, agradezco mucho a Juan Miguel, Sam Pallister y Peter Jonhson la valoración positiva de mi trabajo. Por otro lado, estoy muy agradecido con el grupo de Jose Orallo, de Valencia, por una estancia espectacular. Jose es una de las personas más trabajadoras que conozco y es estupendo poder investigar con él. Espero que en el futuro podamos seguir trabajando juntos. Además, le agradezco su interés por el área de AI Safety, y creo que su experiencia en evaluación de modelos de inteligencia artificial puede ser muy útil.

En la misma línea, doy las gracias a muchas de las personas relacionadas con Altruismo Eficaz que me han apoyado durante estos años, y en especial a Ryan Carey y Jaime Sevilla por su apoyo continuado, y a Victor Veitch por su predisposición a acogerme como postdoc. Me enorgullece formar parte de una comunidad a la que le preocupa mejorar el mundo cuanto sea posible. También me alegra poder contar con personas como Juan García, Pablo Melchor y Pablo Villalobos como amigos.

Finalmente, me gustaría agradecer a mi familia y a mis padres su apoyo incondicional a lo largo de estos años, así como a Pili porque es una persona maravillosa de la que estoy muy orgulloso. Sabes lo mucho que esta tesis se debe a tu ayuda constante.

Contents

Abstract

The framework of this thesis is fault-tolerant quantum algorithms, which can roughly be divided into the following non-disjoint families: a) Grover's algorithm and quantum walks, b) Shor's algorithm and hidden subgroup problems, c) quantum simulation algorithms, d) quantum linear algebra, and e) variational quantum algorithms. All of them are covered, to some extent, in this thesis.

Grover's algorithm and quantum walks are described in Chapter 2. We start by highlighting the central role that rotations play in quantum algorithms, explaining Grover's, why it is optimal, and how it may be extended. Key subroutines explained in this area are amplitude amplification and quantum walks, which will constitute useful parts of other algorithms. In this chapter, we present our Ref. [62], where we explore the heuristic use of quantum Metropolis and quantum walk algorithms for solving an NP-hard problem. This method has been suggested as an avenue to digitally simulate quantum annealing and preparing ground states of many-body Hamiltonians.

In the third chapter, in contrast, we turn to the exponential advantages promised by the Fourier transform in the context of the hidden subgroup problem. However, since this application is restricted to cryptography, we later explore its use in quantum linear algebra problems. Here we explain the development of the original quantum linear solver algorithm, its improvements, and finally the dequantization techniques that would often restrict the quantum advantage to polynomial. In this chapter, we present two publications, Refs. [65, 64], the former aimed at solving linear programming problems, and the latter at machine learning problems. As we shall see, both of them are restricted but in a different way: the first requires reading out the entire solution quantum state, while the second can be effectively dequantized.

Chapter 4 is concerned with quantum simulation. We will review classical quantum chemistry techniques, and then focus on Hamiltonian simulation and ground state preparation as the key problems to be solved. Hamiltonian simulation, in particular, will enable the use of quantum phase estimation which computes the eigenvalues or energies of a given quantum state. Our contribution [63] constitutes a detailed analysis of the cost of many Hamiltonian simulation techniques present in the literature. Variational algorithms, while many times used as a Noisy Intermediate-Scale Quantum (NISQ) alternative to quantum phase estimation, will be presented here as a viable approach to ground state preparation, the other main problem. In contrast, in Ref. [84], we detail how to prepare the Hartree-Fock state in first quantization and plane waves, ideally suited for materials, and which does not require using those techniques. Combined with a state-of-the-art Hamiltonian simulation technique called qubitization, we can

estimate the non-Clifford gate cost of running such an algorithm for realistic battery material properties.

Given the tradition of our group with error correction, we could not end this thesis without dedicating a final chapter to this topic. Here we explain the most important quantum error correction codes, the surface and color codes, and one extension of the latter, gauge color codes. They will show the complexity of implementing non-Clifford quantum gates, therefore validating their consideration as the bottleneck metric.

Resumen

El marco conceptual de esta tesis son los algoritmos cuánticos tolerantes a fallos, que pueden dividirse aproximadamente en las siguientes clases no mutuamente excluyentes: a) algoritmo de Grover y paseos cuánticos, b) algoritmo de Shor y problemas de subgrupos ocultos, c) algoritmos de simulación cuántica, d) álgebra lineal cuántica, y e) algoritmos cuánticos variacionales. Todos ellos se tratan, en cierta medida, en esta tesis.

El algoritmo de Grover y los paseos cuánticos se explican en el capítulo 2. Comenzamos destacando el papel central que juegan las rotaciones en los algoritmos cuánticos, explicando el de Grover, por qué es óptimo, y cómo puede ser extendido. Las subrutinas clave explicadas en esta área son la amplificación de la amplitud y los paseos cuánticos, que serán partes importantes de otros algoritmos. En este capítulo presentamos nuestra Ref. [62], donde exploramos el uso heurístico de los algoritmos de Metrópolis y paseos cuánticos para resolver problemas NP-difíciles. De hecho, este método ha sido sugerido como una vía para simular digitalmente el método conocido como 'quantum annealing', y la preparación de estados fundamentales de Hamiltonianos 'many-body'.

En el tercer capítulo, en cambio, nos centramos en las ventajas exponenciales que promete la transformada de Fourier en el contexto del problema de los subgrupos ocultos. Sin embargo, dado que esta aplicación está restringida a la criptografía, más adelante exploramos su uso en problemas de álgebra lineal cuántica. Aquí explicamos el desarrollo del algoritmo cuántico original para la resolución de sistemas lineales de ecuaciones, sus mejoras, y finalmente las técnicas de 'descuantización' que a menudo restringen la ventaja cuántica a polinómica. En este capítulo se presentan dos publicaciones, Refs. [65, 64], la primera orientada a la resolución de problemas de programación lineal, mientras que la segunda al aprendizaje automático. Como veremos, ambas están restringidas, pero de forma diferente: la primera requiere la lectura de estados cuánticos, mientras que la segunda puede ser efectivamente descuantizada.

El capítulo 4 se ocupa de la simulación cuántica. Revisaremos las técnicas clásicas de la química cuántica, y luego nos centraremos en la simulación Hamiltoniana y en la preparación del estado fundamental como problemas clave a resolver. La simulación Hamiltoniana, en particular, permitirá el uso de la estimación de fase cuántica, que calcula los valores propios o las energías de un estado cuántico dado. De hecho, nuestra contribución [63] constituye una comparativa detallada del coste de muchas técnicas de simulación Hamiltoniana presentes en la literatura. Los algoritmos variacionales, aunque muchas veces utilizados como alternativa a la estimación cuántica de fase en sistemas ruidosos (NISQ), se presentarán aquí como un enfoque viable a la preparación

del estado fundamental, el segundo problema principal. Por el contrario, en la Ref. [84] detallamos cómo preparar el estado Hartree-Fock en primera cuantización y ondas planas, adecuado para materiales, que no requiere de dichas técnicas. Combinado con una técnica de simulación Hamiltoniana de última generación llamada qubitización, somos capaces de estimar el coste en puertas lógicas no-Clifford de ejecutar dicho algoritmo. Dicho algoritmo puede ser usado para predecir propiedades realistas de materiales de baterías eléctricas.

Dada la tradición de nuestro grupo en la corrección de errores, no podíamos terminar esta tesis sin dedicar un capítulo final a este tema. Aquí explicamos los códigos de corrección de errores cuánticos más importantes, los códigos de superficie y de color, y una extensión de estos últimos, los códigos de gauge de color. Así mostraremos la complejidad de implementar puertas cuánticas no-Clifford, validando su consideración como métrica de referencia.

List of publications

- Casares, Pablo A. M., Roberto Campos, and Miguel Angel Martin-Delgado. "QFold: quantum walks and deep learning to solve protein folding." Quantum Science and Technology, 7.2 (2022): 025013.
- Casares, Pablo A. M., and M. A. Martin-Delgado. "A quantum active learning algorithm for sampling against adversarial attacks." New Journal of Physics, 22.7 (2022). 073026.
- Casares, Pablo A. M., and Miguel Angel Martin-Delgado. "A quantum interior-point predictor-corrector algorithm for linear programming." Journal of Physics A: Mathematical and Theoretical 53.44 (2020): 445305.
- Casares, Pablo A. M., Roberto Campos, and Miguel Angel Martin-Delgado. "TFermion: A non-Clifford gate cost assessment library of quantum phase estimation algorithms for quantum chemistry." Quantum (2022), 6:768.
- Delgado, Alain, et al. "Simulating key properties of lithium-ion batteries with a fault-tolerant quantum computer." Physical Reviews A (2022), 106:032428 – main co-author.
- Campos, R., Casares, Pablo A. M., and Martin-Delgado, M. A. "Quantum Metropolis solver: A quantum walks approach to optimization problems." (2022) arXiv preprint arXiv:2207.06462.
- Escrig, G., Campos, R., Casares, Pablo A. M., and Martin-Delgado, M. A. "Parameter estimation of gravitational waves with a quantum metropolis algorithm." (2022) arXiv preprint arXiv:2208.05506.

Additional articles not covered in this thesis, carried out during the internship, on the topic of classical Machine Learning:

- Casares, Pablo A. M., et al. "How General-Purpose Is a Language Model? Usefulness and Safety with Human Prompters in the Wild" (2022), Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI-22).

Conference contributions and internship

During this thesis, the author carried out a 3-month internship in the research group of José Hernández Orallo in the field of classical Machine Learning, as a result of which Ref. [66] was published and presented as a poster at the prestigious AAAI-22 conference.

Additionally, the author has carried out the following activities during his PhD:

- Oral presentation of Ref. [63] at the APS March meeting 2022.
- Oral presentation of Ref. [62] at the Quantum conference in Bilbao in 2022.
- Tutorial on the works of Ref. [62, 60] during the IEEE 2021 Quantum week.
- Poster presentation of the work in Ref. [62] at the ICE-6 Quantum Information in Spain conference in 2021.
- Poster presentation of the work in Ref. [62] during the Munich conference on Quantum Science and Technology 2021.
- Poster presentation of the work in Ref. [62] in the Quantum Techniques in Machine Learning 2021 conference.
- Poster presentation of the work in Ref. [64] at the conference QTech 2020.
- Poster presentation of the work in Ref. [65] during the Munich conference on Quantum Science and Technology 2020.
- Poster presentation of the work in Ref. [64] in the Quantum Techniques in Machine Learning 2020 conference.
- Workshop presentation in the Human-aligned AI Summer School in 2022 on the topic of 'How to produce high-quality research'.
- Attendance and organization of the AI Safety Research Program in 2020.
- Attendance to the Human-aligned AI Summer School in 2019.
- Attendance to the AI Safety Camp in 2019.
- Attendance to the ELLIS Quantum Machine Learning workshop in San Sebastian in 2019.

Chapter 1

Introduction

We are living an extraordinary time for quantum computing. Not even a year after the start of this thesis, in 2019, a team at Google released an article highlighting that they had been able to execute a quantum supremacy experiment, one experiment that is not feasible for classical computers [13]. This first experiment used random quantum circuits, but during these years other supremacy experiments based on boson sampling have also been carried out [267, 172]. On the other hand, this last year we have started witnessing the first experiments that aim to fault-tolerantly implement a set of quantum gates [149, 212, 202, 266], which open the door to fault-tolerant quantum algorithms, the main topic of this thesis, and the kind of algorithms that are likely to be most useful in the long term.

However, they are not the only kind of algorithms. Noisy intermediate-scale quantum algorithms (NISQ) have been proposed as a useful alternative while we are not able to achieve fault tolerance [39]. The two most famous quantum algorithms in this category are the Variational Quantum Eigensolver [200] and the Quantum Approximate Optimization Algorithm [93]. The most important limitation of these algorithms, when used in a non-fault-tolerant setup, is that it is difficult to surpass the capabilities of classical computers able to handily implement billions of logical gates with just a thousand quantum logical gates [78]. This intuition is the reason I have focussed on fault-tolerant quantum algorithms in this thesis.

The first and most famous quantum algorithms are Grover's [111] and Shor's [220], which will be the basis for the second and third chapters of this thesis. The second chapter will deal with quantum search, starting from the Grover algorithm and exploring quantum walks and amplitude amplification as important tools not only for searching itself but in other quantum algorithms. In this chapter, we present QFold [62], an article and simulation experiment where we use a heuristic quantum walk to find the

folded configuration of proteins. Unfortunately, the quantum advantage we measure is fairly small and unlikely to be directly useful, given the overhead of quantum error correction in the speed of quantum gates. On the other hand, heuristic quantum walks can digitally simulate quantum annealing processes, which might be used to prepare the ground state of fermionic Hamiltonians [161]. In general, the quantum speedups one may find with quantum walks and amplitude amplification are polynomial, often quadratic, in nature. Moreover, this advantage is often diluted when parallelism is possible. In particular, to solve an unstructured search in some fixed amount of time, one only needs linearly more classical than quantum processors [101].

The third chapter, in contrast, takes a look at the kind of problems where one may hope to find exponential speedups, mostly relying on the quantum Fourier transform. Shor's algorithm represents the paradigmatic example of these techniques, later generalized to the (Abelian) hidden subgroup problem and mostly used in cryptography applications. The price to pay for this large quantum advantage is the need for structure in the problem that is being addressed [1]. However, in 2008 there was a breakthrough that promised an exponential speedup in the ubiquitous problem of solving a linear system of equations in a very quantum way: with quantum input and output, and under sparsity and well-conditioning assumptions [117]. This would become a line of research called quantum linear algebra, and it is used in two more articles during this thesis [64, 65]. The latter uses those techniques in combination with classical methods in the literature, to propose one of the first quantum interior point methods, aimed at solving linear programming problems. However, the limitations of quantum methods and the necessity to prepare and read quantum states again reduce the quantum advantage to polynomial. In the second paper, we found one machine learning application (adversarial examples) where a readout of the quantum state is not needed. However, in a second breakthrough, an undergraduate student called Ewin Tang showed that many of these quantum linear algebra algorithms could be 'dequantized' [237]. Dequantization substitutes quantum computing for randomized classical computing with ℓ_2-sampling [238], showing again that many algorithms, and in particular ours, will only ever achieve a polynomial speedup.

But if there is a research topic where one may expect quantum computing to be particularly useful, that is in the area of simulation of other quantum systems. Thus, Chapter 4 will discuss its applications to chemistry. It has three main parts. In the first one, we review three of the most popular classical algorithms: Hartree-Fock, Density Functional Theory, and Coupled-Cluster theory. Then, we move on to Hamiltonian simulation, a core quantum technique that computes how a system evolves.

From this section, I would like to highlight one technique, called *qubitization* [169], which allows performing in an optimal number of queries a large variety of functions of the Hamiltonian. This fact will be reflected in previous chapters too [176], but its usefulness to implement Hamiltonian simulation will be particularly reflected here. Finally, in the third section, we discuss ground state preparation, perhaps the most challenging problem in quantum chemistry. This is perhaps the area where variational quantum algorithms, also known as quantum machine learning, will have the largest impact [123, 181]. In the interest of length, however, we will only review the basics.

Two articles from my thesis belong to this chapter, Refs. [63, 84]. In the first one, we present what is perhaps the first software library allowing us to compare a variety of quantum phase estimation algorithms: TFermion. This library analyzes the cost of several quantum phase estimation algorithms proposed in the literature and enables comparisons between them that were previously not possible. TFermion also led to a collaboration with quantum computing startup Xanadu, and ultimately to Ref. [84]. In this reference, we provide a thorough review of how we may use one qubitization-based first-quantization quantum algorithm to analyze the properties of battery materials. Additionally, we perform two minor but important technical advances in the area of Hartree Fock state preparation in plane waves and first quantization, and on the extension of the original algorithm to non-cubic cells. Finally, we provide detailed estimates on the number of non-Clifford gates and the time a quantum computer would require to simulate a particular battery material.

The fifth and final chapter of the thesis will answer the question of why non-Clifford gates are often the most expensive ones to be implemented in a quantum computer. Its focus will be on quantum error correction, an area of research with a long tradition in our research group. Unfortunately, the area is wide enough to only provide a review of the main topics. We start by describing the basics of Calderbank–Shor–Steane (CSS) and stabilizer codes. Then, we explain the two most popular families of error correction codes: surface codes [141] and color codes [46, 47], the latter of which was a key breakthrough by my thesis director. Unfortunately, a theorem by Eastin and Knill proves that it is impossible to construct a stabilizer code that is capable of transversally implementing a universal set of gates [89]. Therefore, while surface and color codes are very attractive, we are forced to find a way to perform those gates fault-tolerantly. The two leading approaches are distilling one kind of gates (often non-Clifford gates) in another code, what is known as magic state distillation, or using subsystem codes. The first one is popular due to its low cost at high error regimes [38], and also the reason non-Clifford gates are expensive. In this thesis, we have instead chosen to describe

gauge color codes [42], as a beautiful subsystem code framework that generalize color codes.

Chapter 2

Quantum Search

> One thing that should be learned from the bitter lesson is the great power of general-purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are search and learning.
>
> Richard S. Sutton, *The Bitter Lesson*

2.1. Objectives

- Understand the mathematical techniques behind quantum walks and their properties.
- Understand the strengths and limitations of quantum walks as a search or Markov-chain mixing technique.
- Explore the kind of problems where quantum walks might be useful, with a special focus on the quantum Metropolis algorithm as a privileged application to combinatorial optimization.
- Find out the quantum advantage associated with a heuristic quantum Metropolis algorithm, the problems where it might be applied, and its usefulness.

2.2. A tale of two rotations: Grover's algorithm

Search is one of the few basic subroutines across many algorithms in computer science. Optimal planning, simulated annealing, and reinforcement learning, widely used in many and an increasing number of applications, can all be seen as ways of searching under different conditions. Perhaps the simplest case is when one has to look for a marked item in an unordered list. Lacking any structure to guide it, a classical computer cannot but iterate over all elements of the set, perhaps allowing some parallelization, in search for the target item. Consequently, if there are N elements in the list, this implies computational complexity scaling as $O(N)$.

Can we do better? Surprisingly, yes: one may leverage quantum mechanics to improve how quickly one can search, which results in $O(\sqrt{N})$ steps. Let us assume we have an oracle such that given item $x \in \{0, ..., N-1\}$, checks whether it fulfills a given condition $f(x) = 1$. The algorithm implementing this search is called after his discoverer, Lov Grover, and implements two rotations iteratively [111], one of which uses this oracle. In fact, for almost any quantum algorithm, it is often intuitive to think in terms of rotations in the Hilbert space. The reason is that quantum operators, except for measurements, are elements of a Special Unitary group $SU(d)$, with d representing the dimension of the Hilbert space. For the Grover algorithm, starting from the uniform superposition $H^{\otimes n} |0\rangle^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} |k\rangle$, the two rotations are the following:

- A phase rotation implemented by any procedure (oracle) that identifies the marked item:

$$O : |x\rangle \mapsto (-1)^{f(x)} |x\rangle . \tag{2.1}$$

  This can be done for example by outputting a bit $|f(x)\rangle$ and implementing a Z gate over it, before uncomputing the bit

$$|x\rangle |0\rangle \mapsto |x\rangle |f(x)\rangle \mapsto (-1)^{f(x)} |x\rangle |f(x)\rangle \mapsto (-1)^{f(x)} |x\rangle |0\rangle . \tag{2.2}$$

- A second rotation called diffusion operator U_s, that implements a rotation over the initial state, taken to be a uniform superposition

$$U_s = H^{\otimes n}(\mathbf{1} - 2 |0\rangle \langle 0|) H^{\otimes n}. \tag{2.3}$$

In Fig. 2.2, we define $\theta := \arcsin \sqrt{\frac{M}{N}}$, where $M \ll N$ is the number of marked items. In each rotation, the angle grows by 2θ, so after t time steps the angle will be

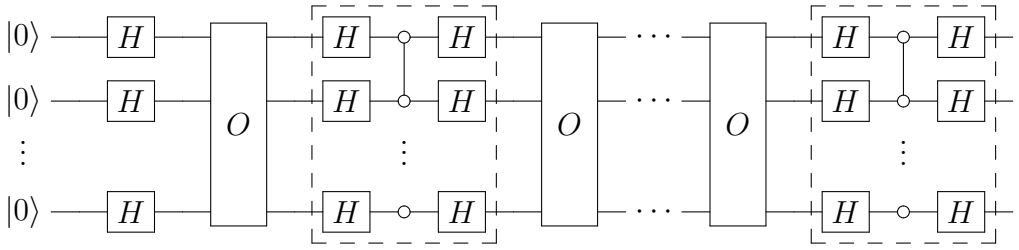


Figure 2.1 **Grover's algorithm.** The dashed box represents the diffusion operator U_s, while the oracle operator O is explicitly indicated. After the initialization, operators O and U_s must be applied $O(\sqrt{N})$ times.

$(2t+1)\theta$. Since the target angle is $\approx \frac{\pi}{2}$, the number of steps to take grows as $t \approx \frac{\pi}{4}\sqrt{\frac{N}{M}}$. In conclusion, by alternating these two rotations we can achieve a quadratic speedup over what is possible classically.

The next question one may ask is: can we do better? Unfortunately, the answer is no, as Bennett, Bernstein, Brassard, and Vazirani proved even before Grover discovered his algorithm [30]. For simplicity assume a single marked item $|\omega\rangle$. Their procedure compares two circuits. The first one contains T calls to the oracle $O = U_\omega = \mathbf{1} - 2\,|\omega\rangle\,\langle\omega|$, and other arbitrary but fixed rotations $U_1, \ldots, U_T$, [204]

$$|\psi_\omega(t)\rangle = U_\omega U_T U_\omega \cdots U_\omega U_1\,|\psi(0)\rangle\,. \tag{2.4}$$

The second is the same circuit without any calls to the oracle

$$|\varphi(t)\rangle = U_T U_{T-1} \cdots U_1\,|\psi(0)\rangle\,. \tag{2.5}$$

To compare them, we can check how much the operator U_ω changes the state at time step t

$$||(U_\omega - \mathbf{1})\,|\varphi(t)\rangle\,|| = 2|\,\langle\omega|\varphi(t)\rangle\,|. \tag{2.6}$$

This can be interpreted as an error vector between the two preparations. In fact,

$$||\,|\psi_\omega(T)\rangle - |\varphi(T)\rangle\,|| \leq 2\sum_{t=1}^{T} |\,\langle\omega|\varphi(t)\rangle\,|. \tag{2.7}$$

Since simple arithmetic shows that [204, Eq. 6.166]

$$\left(\sum_{t=1}^{T} c_t\right)^2 \leq T\sum_{t=1}^{T} c_t^2 \tag{2.8}$$

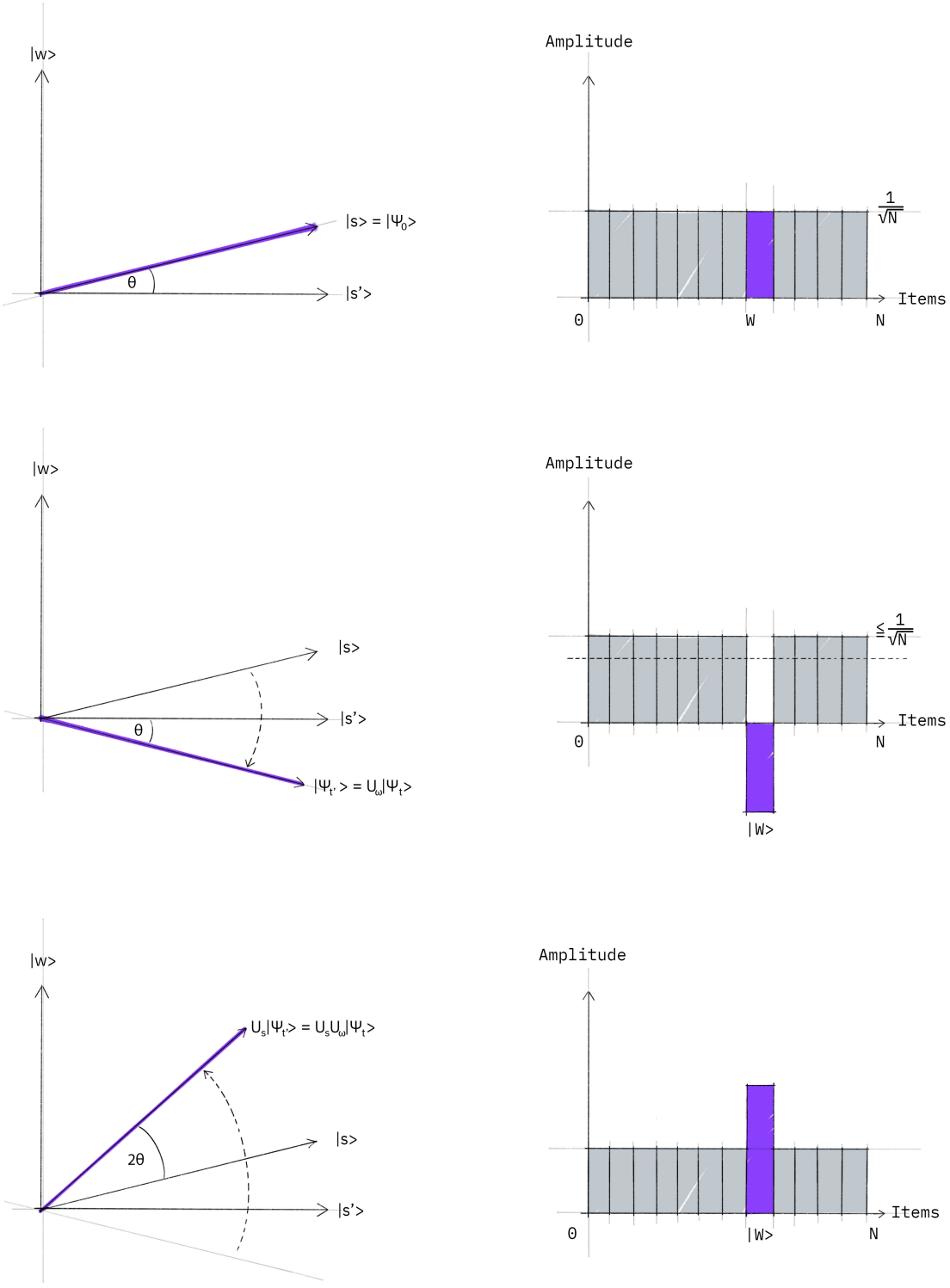


Figure 2.2 **Grover's algorithm rotations.** Top: initial state, a uniform superposition of marked and not marked states. Medium: Oracle rotation, which changes the sign of the marked states. Bottom: diffusion operator, that performs a rotation around the average. The angle $\sin\theta = \sqrt{M/N}$, for M the number of marked items in a list of N elements. Images from the Qiskit textbook under Apache 2.0 license.

we have

$$|| \, |\psi_\omega(T)\rangle - |\varphi(T)\rangle \, ||^2 \leq 4 \left(\sum_{t=1}^{T} | \, \langle \omega | \varphi(t) \rangle \, | \right)^2 \leq 4T \sum_{t=1}^{T} | \, \langle \omega | \varphi(t) \rangle \, |^2 \leq 4T^2. \tag{2.9}$$

To distinguish between states with certainty, we need states to be orthogonal. Consequently, we take $|\omega\rangle$ to be part of an orthonormal basis ω', and summing over it,

$$\sum_{\omega'} || \, |\psi_{\omega'}(T)\rangle - |\varphi(T)\rangle \, ||^2 \leq 4T \sum_{t=1}^{T} \underbrace{\sum_{\omega'} | \, \langle \omega' | \varphi(t) \rangle \, |^2}_{= \langle \varphi(T) | \varphi(T) \rangle} = 4T^2. \tag{2.10}$$

The left-hand side of this expression can be lower-bounded by $2N - 2\sqrt{N}$ [204, Eq. 6.174], so we have that $4T^2 \geq 2N - 2\sqrt{N}$, and finally $T \geq \sqrt{\frac{N-\sqrt{N}}{2}}$. In other words, if we aim to distinguish all N orthogonal possible values of $|\omega\rangle$, we need to call $T = O(\sqrt{N})$ times the oracle U_ω. Further, if we aim for a success probability of $1 - \delta$, then one can show that we need $T \geq \sqrt{\frac{N}{2}} \sqrt{1 - \sqrt{\delta}}$ [204, Chapter 6].

2.2.1. Amplitude Amplification

Grover's algorithm is, in its original form, just intended for unordered search. We would now like to study some of its main use cases and extensions. The first of such applications is known as amplitude amplification, and it is widely used in many settings [51, 111]. Let us suppose we have a quantum algorithm A that probabilistically prepares a state $|\psi\rangle$:

$$A : |1\rangle_f \, |0\rangle_s \mapsto \alpha \, |0\rangle_f \, |\psi\rangle_s + \beta \, |1\rangle_f \, |\psi^\perp\rangle_s \, , \tag{2.11}$$

where register f indicates failure in preparing the target state, and register s contains the state. By identifying $\alpha = \sin\theta$ and $\beta = \cos\theta$, we can uncover a two-dimensional rotation amenable to a Grover-like search algorithm. We define the Amplitude Amplification operator [51]

$$Q = -A R_s A^\dagger R_t, \tag{2.12}$$

where $R_s := (2 \, |10\rangle \langle 10|_{fs} - \mathbf{1})$ and $R_t := (2 \, |0\rangle \langle 0|_f - \mathbf{1}_f)$, as the rotation around the start and target states. To recover Grover's algorithm, A would have to be equal to

the Hadamard gate. Overall, this operator acts as

$$\begin{aligned} Q\left|0\right\rangle_f\left|\psi\right\rangle_s &= \cos(2\theta)\left|0\right\rangle_f\left|\psi\right\rangle_s - \sin(2\theta)\left|1\right\rangle_f\left|\psi^\perp\right\rangle_s, \\ Q\left|1\right\rangle_f\left|\psi^\perp\right\rangle_s &= \sin(2\theta)\left|0\right\rangle_f\left|\psi\right\rangle_s + \cos(2\theta)\left|1\right\rangle_f\left|\psi^\perp\right\rangle_s. \end{aligned} \tag{2.13}$$

We can also diagonalize this operator, obtaining eigenvectors

$$\left|\psi_\pm\right\rangle = \frac{1}{\sqrt{2}}\left(\left|1\right\rangle_f\left|\psi^\perp\right\rangle_s \pm i\left|0\right\rangle_f\left|\psi\right\rangle_s\right), \tag{2.14}$$

with eigenvalues $e^{\pm 2i\theta}$ respectively. In other words,

$$Q = e^{2i\theta}\left|\psi_+\right\rangle\left\langle\psi_+\right| + e^{-2i\theta}\left|\psi_-\right\rangle\left\langle\psi_-\right|. \tag{2.15}$$

As we shall see, Q is very helpful when we have to implement an operator A multiple times, but do not want the success probability to vanish exponentially in the number of steps.

2.2.2. Fixed point Amplitude Amplification and the first glimpse of Quantum Signal Processing

A different modification we analyze in this section is the possibility of an algorithm with a fixed point. Note that Grover's algorithm has an inconvenient feature: if we do not know how many marked items there are, it is unclear how to choose the number of amplification steps. In fact, given the oscillatory nature of Grover's algorithm, it is possible to overshoot and amplify 'too much'.

To remedy this, we present a second algorithm, also by Grover, which monotonically increases the probability of measuring the target [112]. Let $\left|s\right\rangle$ and $\left|t\right\rangle$ be the starting and target states, and U a rotation between the subspaces spanned by them, such that $\left|\left\langle t|U|s\right\rangle\right|^2 = 1-\varepsilon$. By defining

$$R_s = \mathbf{1} - \left(1 - \exp\left(i\frac{\pi}{3}\right)\right)\left|s\right\rangle\left\langle s\right|, \qquad R_t = \mathbf{1} - \left(1 - \exp\left(i\frac{\pi}{3}\right)\right)\left|t\right\rangle\left\langle t\right|, \tag{2.16}$$

the operator $UR_sU^\dagger R_tU$ will fulfill $\left|\left\langle t|UR_sU^\dagger R_tU|s\right\rangle\right|^2 = 1-\varepsilon^3$ [112]. If instead of $\pi/3$ we had used π, we would recover the original Grover algorithm.

We can now use this idea to monotonically improve the success probability in an Amplitude Amplification setting. We can use the recursion

$$U_{m+1} = U_m R_s U_m^\dagger R_t U_m, \qquad U_0 = A, \tag{2.17}$$

where A is defined in (2.11). The success probability scales as $|\langle t|U_m|s\rangle|^2 = 1 - \varepsilon^{3m}$. Expressed in the number of calls to the oracle q_m, $|\langle t|U_m|s\rangle|^2 = 1 - \varepsilon^{2q_m+1}$ [112]. In contrast, the classical success probability is $1 - \varepsilon^{c+1}$, for c oracle calls.

This algorithm, while being optimal under the monotonic condition, loses the original quadratic speedup [264]. To see it, imagine we want to find the marked item with failure probability δ, such that the initial and target states have a small overlap probability $\lambda = 1 - \varepsilon \to 0$. We want to find q_m such that

$$\delta \geq \varepsilon^{2q_m+1} \approx (1-\lambda)^{2q_m+1} \approx 1 - (2q_m+1)\lambda, \tag{2.18}$$

what implies that [113, Eq. 16]

$$q_m \gtrsim \frac{1-\delta}{2\lambda} = O\left(\frac{1}{\lambda}\right), \tag{2.19}$$

similar to the classical case. In contrast, Grover's original algorithm required $O(\lambda^{-1/2})$ calls to the oracle.

Can we find a way around it? We can, but it implies letting go of the monotonic assumption. As before, we assume we have an initial state $|s\rangle = A|0\rangle$, and want to prepare a target state $|t\rangle = e^{-i\xi}|t'\rangle$ such that $\langle t'|s\rangle = e^{i\xi}\sqrt{\lambda}$. In other words, $|s\rangle = \sqrt{\lambda}|t\rangle + \sqrt{1-\lambda^2}|t^\perp\rangle$. We are also provided with an oracle O which marks the target state $|t'\rangle$. The objective is finding a quantum circuit Q such that the success probability $P_L = |\langle t'|Q|s\rangle|^2$ is $1-\delta^2$, where $\delta \in [0,1]$, after L calls to the oracle.

The key idea to tackle this problem [264], is to generate a sequence of generalized Grover rotations $G(\alpha,\beta)$

$$Q = G(\alpha_L,\beta_L)\cdots G(\alpha_1,\beta_1), \qquad G(\alpha,\beta) = -S_s(\alpha)S_t(\beta), \tag{2.20}$$

where

$$S_x(\alpha) := \mathbf{1} - (1-e^{i\alpha})|x\rangle\langle x|. \tag{2.21}$$

These reflection operators $S_s(\alpha)$ and $S_t(\beta)$ can also be understood as rotations in a 2-dimensional Block sphere-like space:

$$R_\varphi(\theta) = \exp\left(-\frac{i}{2}\theta(Z\cos(\varphi) + X\sin(\varphi))\right). \tag{2.22}$$

Using this rotation and taking $\phi = 2\arcsin(\sqrt{\lambda})$, we can rewrite [264]

$$S_t(\beta) = e^{i\beta/2}R_0(\beta), \qquad S_s(\alpha) = e^{-i\alpha/2}R_\phi(\beta). \tag{2.23}$$

This formulation allows computing what angles α_i, β_i can be used to increase the success probability P_L to $1-\delta^2$. For $l = (L-1)/2$ and $j \in [1, l]$ we should take [264]

$$\alpha_j = -\beta_{l-j+1} = 2\cot^{-1}\left(\tan(2\pi j/L)\sqrt{1-\gamma^2}\right), \tag{2.24}$$

with $\gamma^{-1} := T_{1/L}(1/\delta)$, and $T_L(\cos\theta) := \cos(L\theta)$ the L^{th} first-order Chebyshev polynomial. Using these angles, one obtains that

$$P_L = |\langle t|Q|s\rangle|^2 = 1 - \delta^2 T_L^2(T_{1/L}(1/\delta)\sqrt{1-\lambda}). \tag{2.25}$$

Chebyshev polynomials fulfill that if $|x| \leq 1$, then $|T_L(x)| \leq 1$. From this it is clear that if $|T_{1/L}(1/\delta)|\sqrt{1-\lambda} \leq 1$, then we can write $P_L \geq 1-\delta^2$ as wanted. This happens whenever $\lambda \geq 1 - T_{1/L}^{-2}(1/\delta) =: w$. We will call w the width. For large L and small $\delta > 0$ [264, Eq. 2]

$$w \approx \left(\frac{\log(2/\delta)}{L}\right)^2, \tag{2.26}$$

what implies that we need to choose

$$L \geq \frac{\log(2/\delta)}{\sqrt{\lambda}}, \tag{2.27}$$

recovering the quadratic speedup that characterizes Grover's original algorithm.

Furthermore, we can concatenate this procedure to improve the target δ. Calling $Q_L(B)$ the rotation that uses BA in place of A for the definition of the initial state, and χ_1, χ_2 some phases, then if

$$Q_{L_1}|s\rangle := \sqrt{1-P_{L_1}(\lambda)}\,|t^\perp\rangle + \sqrt{P_{L_1}(\lambda)}e^{-i\chi_1}|t\rangle, \tag{2.28}$$

we also have that [264]

$$Q_{L_2}(Q_{L_1})Q_{L_1}\left|s\right\rangle := \sqrt{1-P_{L_2}(P_{L_1}(\lambda))}\left|t^{\perp}\right\rangle + \sqrt{P_{L_2}(P_{L_1}(\lambda))}e^{-i(\chi_1+\chi_2)}\left|t\right\rangle. \tag{2.29}$$

Since Q_{L_1} acts as a prefix, we can implement Q_{L_1} and then decide whether to improve the result further. Moreover, if we choose $\delta = 1$, from (2.24) we obtain $\alpha_j = \beta_j = \pm\pi$, thereby generalizing Grover's original algorithm. And if instead we chose $\delta = 0$, we obtain $-\alpha_1 = \beta_1 = \pi/3$ [264], and using recursion as explained before we recover the $\pi/3$ monotonic algorithm, but as expected without the quadratic speedup from (2.27).

2.3. Quantum walks

While the extensions of Grover's algorithm that we have discussed so far deal with unordered databases or probabilistic algorithms, there are situations where we have more information about the structure of the search space. In particular, let us consider a graph $G(X, E)$, where X denotes the vertices (states) and E the edges (state transitions). We now define a Markov chain over this graph, that for each pair of vertices x, y connected by an edge assigns a probability of transition, forming a matrix $P = (p_{xy})_{(x,y)\in E}$. We will analyze the particular case of discrete ergodic Markov chains.

Definition 1 (Discrete ergodic Markov chain) *A Markov chain $\mathcal{P} = (p_{xy})_{(x,y)\in E}$ defined over a graph $G(X, E)$ is called ergodic, if $\exists t_0$ such that $\forall x, y \in X$, and starting from x at time $t = 0$ and following probability transitions dictated by $\mathcal{P}$, then $\forall t > t_0$, the probability of finding an item y at time t is greater than 0. In mathematical notation,*

$$\exists t_0 \quad | \quad \forall x, y \in X, \quad p(x, t = 0) = 1 \Rightarrow p(y, t > t_0) > 0. \tag{2.30}$$

Ergodic Markov chains have a unique stationary distribution π, that fulfills $\pi^T P = \pi^T$, or in other words, the eigenvalue 1 has multiplicity 1 too [148, Theorem 5.9]. In this case, we define an eigenvalue gap $\delta = 1 - \lambda$ where λ is the second-largest absolute value of an eigenvalue of $\mathcal{P}$, the first being 1. This eigenvalue gap δ will determine the time required for the Markov chain to mix, or in other words, to approximate π. Finally, Markov chains are called *reversible* if they obey the detailed *balance property*

$$p_{yx}\pi_x = p_{xy}\pi_y, \tag{2.31}$$

and are called *symmetric* if $P = P^T$. Symmetric Markov chains display the uniform superposition as the stationary state [173].

2.3.1. Hitting time and search algorithms

There are two main tasks one can perform in a Markov chain when there is a set of marked items M,

- $\texttt{Detect}^{(=k)}$: Check if $M = \emptyset$ under the promise that either $M = \emptyset$ or $|M| = k$.
- $\texttt{Find}^{(=k)}$: Find $m \in M$ under the promise that $|M| = k$.

If instead of superscript $(= k)$ we use $(\geq k)$, the same promise applies with greater or equal than. If no promise is given, then $k \geq 1$ is assumed.

Classically, these two tasks are the same: one cannot detect the presence of a marked node without finding it. A simple classical algorithm for finding a marked state $m \in M \subset X$ is the following, [173]:

Algorithm 1 A basic classical search algorithm

1: **Input**: Ergodic Markov chain $\mathcal{P}$, graph $G(X, E)$, time t.
2: **Output**: Marked state m or signal that not marked item exists.
3: Initialize x sampled from the stationary distribution π.
4: **for** t steps **do**:
5: If state reached x is marked, output x.
6: Else, simulate 1 step of Markov chain $\mathcal{P}$ from x.
7: **end for**
8: Output 'no marked element exists'.

Running Algorithm 1 has three cost sources: $\texttt{S}$ for the set-up cost of initializing the data structure and sampling the initial x, often from the stationary distribution π of the corresponding Markov chain, $\texttt{U}$ for updating the item according to $\mathcal{P}$, and $\texttt{C}$ for checking if it is a marked item.

Definition 2 (Hitting time) *We define the hitting time $HT(P, M)$ of a Markov chain $\mathcal{P}$ as the expected number of evaluations of $\texttt{U}$ required to find a marked item $m \in M$ with Algorithm 1 starting from the projection of the stationary distribution π into the set of unmarked states $|\pi_U\rangle = \Pi_U |\pi\rangle$.*

The Hitting time HT from Algorithm 1 in *symmetric* Markov chains is $HT = t = O(1/(\delta\epsilon))$ [173, Proposition 1]. In such a case, its overall cost is $O\left(\texttt{S} + \frac{1}{\delta\epsilon}(\texttt{U} + \texttt{C})\right)$. Another slightly different algorithm is the following [173]:

Taking t_2 steps of the Markov Chain aims to mix the state again so that it is close to the stationary distribution π. In a symmetric Markov chain we can take $t_1 = O(1/\epsilon)$

Algorithm 2 A more efficient classical search algorithm

1: **Input**: Ergodic Markov chain $\mathcal{P}$, graph $G(X, E)$, times t_1, t_2.
2: **Output**: Marked state m or signal that not marked item exists.
3: Initialize x sampled from the stationary distribution π.
4: **for** t_1 steps **do**:
5: If state reached x is marked, output x.
6: Else, simulate t_2 steps of Markov chain $\mathcal{P}$ from x.
7: **end for**
8: Output 'no marked element exists'.

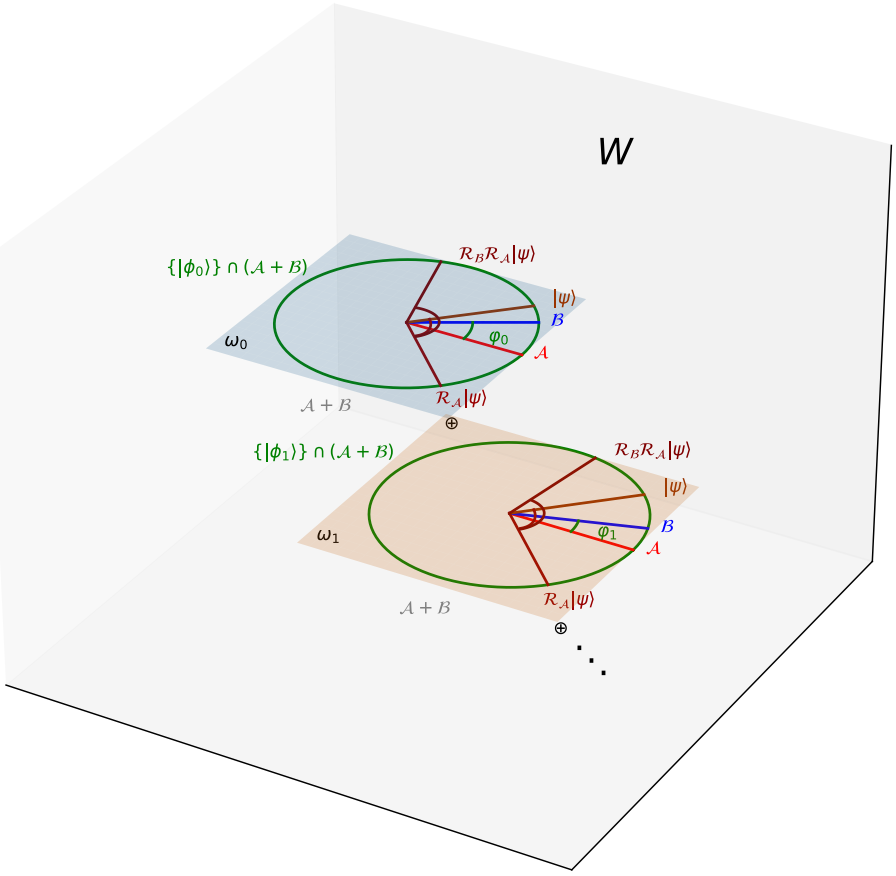


Figure 2.3 **Szegedy quantum walk.** Geometrical visualization of the action of a Szegedy quantum walk operator W, defined in (2.36). W performs a series of rotations that in the subspace $\mathcal{A} + \mathcal{B}$ may be written as a block diagonal matrix, where each block is a 2-dimensional rotation $\omega_j = R(2\varphi_j)$ given by (2.42). This figure represents the direct sum of Grover-like rotations in the subspace spanned by $\mathcal{A}+\mathcal{B}$, and therefore W. Note that $\varphi_0 = 0$ because it corresponds to eigenvalue $\lambda_0 = 1$, but is represented non-zero for clarity.

and $t_2 = O(1/\delta)$, resulting in Algorithm 2 having complexity $O\left(\mathtt{S} + \frac{1}{\epsilon}\left(\frac{1}{\delta}\mathtt{U} + \mathtt{C}\right)\right)$, [173, Proposition 1].

Our objective is to find quantum equivalent algorithms to `Detect` and `Find` which scale quadratically better in the hitting time. These algorithms will be called *quantum walks*[1], the first of which was introduced by Ambainis for Johnson graphs in Ref. [5]. Later on, Szegedy proposed a more general walk that extends to ergodic reversible Markov chains [236]. To describe it, we start by defining a bipartite Hilbert space $\mathcal{H} \otimes \mathcal{H}$, where $\mathcal{H}$ represents the space of possible solutions X. The update operator U will now map

$$U : |x\rangle |0\rangle \mapsto |\alpha_x\rangle := |x\rangle \otimes \sum_{y \in X} \sqrt{p_{xy}} |y\rangle , \tag{2.32}$$

and similarly, we can define

$$V : |0\rangle |y\rangle \mapsto |\beta_y\rangle := \sum_{y \in X} \sqrt{p_{yx}} |x\rangle \otimes |y\rangle . \tag{2.33}$$

The relation between these operators is $SU = VS$, where S is the Swap operation between both Hilbert subspaces. Using them, we can define subspaces

$$\mathcal{A} := \text{span}\{|x\rangle |0\rangle : x \in X\}, \qquad \mathcal{B} := U^\dagger V S \mathcal{A} = U^\dagger S U \mathcal{A}. \tag{2.34}$$

The projector on these subspaces

$$\Pi_{\mathcal{A}} := (\mathbf{1} \otimes |0\rangle \langle 0|), \qquad \Pi_{\mathcal{B}} := U^\dagger V S (\mathbf{1} \otimes |0\rangle \langle 0|) S V^\dagger U = U^\dagger S U (\mathbf{1} \otimes |0\rangle \langle 0|) U^\dagger S U \tag{2.35}$$

define rotations $R_{\mathcal{A}}$ and $R_{\mathcal{B}}$, which we can use to introduce the quantum walk operator

$$W = R_{\mathcal{B}} R_{\mathcal{A}} = U^\dagger S U R_{\mathcal{A}} U^\dagger S U R_{\mathcal{A}}. \tag{2.36}$$

To analyze it, we can compare it with

$$\mathcal{M} := U^\dagger V S = U^\dagger S U. \tag{2.37}$$

Since we assumed that the Markov chain is reversible, using the detailed balance equation, we can find its matrix element on the subspace $\mathcal{A}$, $\langle x, 0 | U^\dagger V S | y, 0\rangle =$

[1] In this thesis, we will focus on discrete-time quantum walks. However, it is also possible to define a continuous-time quantum walk by defining a Hamiltonian that dictates the evolution of the system. Such Hamiltonian depends on the graph, for example by dictating the allowed transitions between nodes. Continuous quantum walks are therefore implemented via Hamiltonian simulation techniques that we will see in Chapter 4.

$\sqrt{p_{yx}}\sqrt{p_{xy}} = \sqrt{\pi_x/\pi_y}p_{xy}$ [265, Pag. 756]. In matrix form, this is called the *discriminant matrix*

$$\mathcal{D} = D_\pi^{-1/2}\mathcal{P}D_\pi^{1/2} \tag{2.38}$$

where D_π is the diagonal matrix containing the entries of π, the equilibrium distribution. Since the matrix D_π is positive definite, $\mathcal{P}$, $\mathcal{D}$ and $\mathcal{M}$ have the same spectrum in the subspace defined by projector $\Pi_\mathcal{A}$, see Ref. [7, Pag. 413] and Ref. [11, Pag. 5]. These eigenvalues $1 = \lambda_0 > \lambda_1 \geq \ldots \geq \lambda_{d-1}$ can be rewritten as phases $\lambda_j = \cos\varphi_j$, and we will denote the corresponding eigenstates as $|\phi_j\rangle|0\rangle$. Since [265, Eq. S21]

$$\langle\phi_j, 0|\Pi_\mathcal{A}U^\dagger VS\Pi_\mathcal{A}|\phi_j, 0\rangle = \lambda_j = \lambda_j^\dagger = \langle\phi_j, 0|\Pi_\mathcal{A}SV^\dagger U\Pi_\mathcal{A}|\phi_j, 0\rangle\,, \tag{2.39}$$

we have,

$$\Pi_\mathcal{A}U^\dagger VS\Pi_\mathcal{A} = \Pi_\mathcal{A}SV^\dagger U\Pi_\mathcal{A}. \tag{2.40}$$

We can use this, and the definition of the spectrum of $\mathcal{M}$, to compute [265, Eq. S19]

$$\Pi_\mathcal{A}U^\dagger VS\,|\phi_j\rangle|0\rangle = \cos\varphi_j\,|\phi_j\rangle|0\rangle\,, \tag{2.41a}$$

and

$$\Pi_\mathcal{B}\,|\phi_j\rangle|0\rangle = U^\dagger VS\cos\varphi_j\,|\phi_j\rangle|0\rangle\,. \tag{2.41b}$$

Consequently, the subspace spanned by $\{|\phi_j\rangle|0\rangle, U^\dagger VS\,|\phi_j\rangle|0\rangle\}$ is preserved under $R_\mathcal{A}$ and $R_\mathcal{B}$. In such subspaces, the operation of W is a rotation

$$w_j = \begin{pmatrix}\cos(2\varphi_j) & -\sin(2\varphi_j)\\ \sin(2\varphi_j) & \cos(2\varphi_j)\end{pmatrix}. \tag{2.42}$$

In contrast to the classical walk, where the eigenvalue gap was $\delta = 1 - \lambda$, the quantum phase gap is defined as $\Delta := 2\varphi = 2\arccos\lambda$. The relation between both is $\Delta \geq 2\sqrt{1-|\lambda|^2} \geq 2\sqrt{\delta}$ [173, Theorem 7], which will ultimately originate the familiar quadratic speedup that also characterizes Grover algorithm [236].

However, so far we have not explained how to implement either of the two tasks using Szegedy's quantum walks. Let us first consider a quantum version of Algorithm 1 for `Detect`, presented in Algorithm 3. What Szegedy did in his original article [236] is a so-called *swap test*, between the stationary distribution and the same state evolved under W' defined as in (2.36) for an absorbent Markov chain $\mathcal{P}'$. $\mathcal{P}'$ is defined as $\mathcal{P}$ with the transitions from marked to unmarked states deleted.

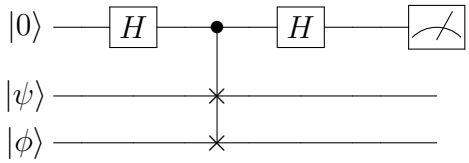


Figure 2.4 **Swap test.** Originally described in Re. [57], the probability of measurement of 0 of this algorithm is $\frac{1}{2}(1 + \| \langle\phi|\psi\rangle \|^2)$. This circuit might be consequently used to estimate $\| \langle\phi|\psi\rangle \|^2$, to precision ϵ^{-1} with $O(\epsilon^{-2})$ measurements, or alternatively using quantum amplitude estimation with complexity $O(\epsilon^{-1})$. The complexity in the failure probability is logarithmic in both cases.

Algorithm 3 Szegedy quantum detection [236, Lemma 7]

Input: Ergodic symmetric Markov chain $\mathcal{P}$, graph $G(X, E)$, overlap ϵ, eigenvalue gap δ, target failure probability p_f.
Output: Whether a marked item exists.
Define $\mathcal{P}'$ as the absorbent Markov chain corresponding to $\mathcal{P}$, and W' its corresponding quantum walk.
for $O(\log p_f^{-1})$ steps **do**:
    Pick t uniformly at random in $\left[1, O\left(\frac{1}{\sqrt{\delta\epsilon}}\right)\right]$.
    Prepare the state $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\,|\pi\rangle$, for $|\pi\rangle$ the stationary distribution.
    for t steps **do**:
        Controlled on the first qubit, evolve the state using W'.
    end for
    The resulting state is

$$\frac{1}{\sqrt{2}}\left(|0\rangle\,|\pi\rangle + |1\rangle\,(W')^t\,|\pi\rangle\right). \tag{2.43}$$

    Perform a Hadamard on the first qubit, obtaining

$$\frac{1}{2}\left(|0\rangle\,(\mathbf{1} + (W')^t)\,|\pi\rangle) + |1\rangle\,(\mathbf{1} - (W')^t)\,|\pi\rangle)\right). \tag{2.44}$$

    Measure the first qubit.
    If measurement is $|1\rangle$, output 'Marked item detected'.
end for
Output 'No marked item exists' and stop.

To see why Algorithm 3 works, remember that $\pi^T\mathcal{P} = \pi$, and therefore $W\pi = \pi$. If no marked item exists, then $W' = W$, and consequently with probability 1 we will measure $|0\rangle$ after (2.44). $t \in [1, O(1/\sqrt{\delta\epsilon})]$ is chosen so that if there are $\epsilon|X|$ marked item, then $\|(1-(W')^t)\pi\|$ does not cancel out. Overall, the complexity of this algorithm is $\tilde{O}\left(S + \frac{1}{\sqrt{\delta\epsilon}}\,(\mathtt{U} + \mathtt{C})\right)$.

We now focus on the problem of `Find`. We can remove the requirement of symmetry of the Markov chain, and only require that it is reversible (e.g., it fulfills (2.31)), while at the same time obtaining a quantum equivalent of Algorithm 2 [173]. For that, we will use a subroutine called *phase estimation*, which we will introduce in Chapter 3 and denoted here by P. Given a Hermitian operator H with eigenstates $|\phi_j\rangle$ and eigenvalues $\lambda_j := \cos\varphi_j$,

$$P(H) : |\phi_j\rangle |\mathbf{0}\rangle \to |\phi_j\rangle |\tilde{\varphi}_j\rangle , \tag{2.45}$$

where $\tilde{\varphi}_j$ is a binary approximation to φ_j.

The objective is to perform a Grover-like algorithm to amplify the marked states. Grover's algorithm would consist of rotations

$$G = \left(2 \sum_{m\in M} |m\rangle \langle m| - \mathbf{1}\right) (2 |\pi\rangle \langle \pi| - \mathbf{1}). \tag{2.46}$$

The first rotation requires the use of `C`, while the second uses `S` that prepares the stationary state $|\pi\rangle$. However, this does not use the graph structure.

Our objective is to modify Grover's algorithm and avoid the use of `S`: instead of decomposing $(2 |\pi\rangle \langle \pi| - \mathbf{1}) = \mathtt{S}(2 |\mathbf{0}\rangle \langle \mathbf{0}| - \mathbf{1})\mathtt{S}^\dagger$, we can use a combination of step operator U and $P(H)$ to perform a rotation over $|\pi\rangle$. This relies on the fact that we can use $P(H)$ to detect and apply a phase to the state with eigenvalue 1. Thus, this procedure will be more appropriate when phase estimation $P(H)$ is cheaper than the quantum state preparation subroutine `S`. The proposed algorithm is the following.

The reason we need to run $P(W)$ with precision Δ^{-1} is that this is the precision we need to distinguish between $|\pi\rangle$ and any other eigenstate of W. As we will see in Chapter 3, achieving precision Δ^{-1} means a number of logical gates scaling as $O(\Delta^{-1})$. Furthermore, the median is used to ensure that the probability of failure decreases exponentially fast, using the Median Lemma [192, Lemma 1]. The implementation of such `Median` protocol can be done with a reversible sorting protocol such as a sorting network. Overall, the complexity is $\tilde{O}\left(S + \frac{1}{\sqrt{\epsilon}}\left(C + \frac{1}{\Delta}U\right)\right)$, where $\Delta = O(\sqrt{\delta})$. Moreover, instead of using Grover's method, we can instead use the $\pi/3$ monotonically increasing algorithm [112], or the quantum signal processing algorithm [264] that we saw at the end of Section 2.2.2.

Algorithm 4 Magniez quantum search [173, Theorem 3]

Input: Ergodic reversible Markov chain $\mathcal{P}$, graph $G(X, E)$, overlap ϵ, eigenvalue gap δ, target failure probability p_f, $O(\log(\sqrt{\epsilon}\Delta p_f)^{-1}\Delta^{-1})$ auxiliary qubits $|\cdot\rangle_a$.
Output: A marked item if it exists, or 'no marked element exists' message.
Choose t uniformly at random from $[1, O(1/\epsilon)]$.
Prepare $|\pi\rangle\,|0\rangle\,|0\rangle_a^{\otimes \log p_f^{-1} \log \Delta^{-1}}$.
for t steps **do**:
    For any possible state $|x\rangle\,|y\rangle\,|z\rangle$, use check operator `C` so that

$$|x\rangle\,|y\rangle\,|z\rangle_a \mapsto (-1)^{x\in M}\,|x\rangle\,|y\rangle\,|z\rangle_a\,. \tag{2.47}$$

    Apply $P(W)$ $O(\log(\sqrt{\epsilon}p_f)^{-1})$ times in new sets of auxiliary qubits, with precision $O(\Delta^{-1})$.
    Compute the median of the estimated $|\tilde{\lambda}_j\rangle$s.
    If $\texttt{Median}(\{|\tilde{\lambda}_j\rangle\}) = 1$, apply a (-1) phase.
    Uncompute the median.
    Uncompute the $P(W)$ operators.
end for
Measure the first register
if $x \in M$ **then**
    Output x
else
    Output 'No element exists'.
end if

Quantum Fast Forwarding and an optimal algorithm

So far, however, we have not given a closed expression for the Hitting Time. The Hitting Time can be expressed as [150, Proposition 9]

$$HT(P, M) = \sum_{k=1}^{d-|M|} \frac{|\,\langle\phi'_k|\pi_U\rangle\,|^2}{1-\lambda'_k}, \tag{2.48}$$

where d is the number of eigenvalues $\lambda'_k \neq 1$ of the absorbent $\mathcal{M}'$ defined similarly as in (2.38), corresponding to eigenstates $|\phi'_k\rangle$ (see Ref. [150, Proposition 34]). The sum is explicitly designed to avoid the $|M|$ eigenstates with eigenvalue 1, the marked states, where the fraction would not be well-defined. If there are no transitions between marked states, it is easy to recognize the eigenvalue 1 states. In such a case, remember that if $|m\rangle \in M$, then $\mathcal{P}'\,|m\rangle = |m\rangle$ as the Markov chain $\mathcal{P}'$ is absorbent, so $|m\rangle$ is an eigenstate with eigenvalue 1.

The Hitting Time can sometimes be $O(1/\sqrt{\delta\epsilon})$ as we saw was the case for symmetric Markov Chains, but it may also be smaller [7]. For that reason, the algorithms we have explained so far are not optimal. However, there exists one algorithm that approximately is, achieving a quadratic speedup in the Hitting Time up to polylogarithmic complexity [7]. It is based on two main techniques.

First, it makes use of 'extrapolated' quantum walks, defined as the quantum walk corresponding to

$$\mathcal{P}(s) = (1-s)\mathcal{P} + s\mathcal{P}', \quad s \in [0,1), \tag{2.49}$$

which is still a reversible Markov chain with a unique stationary distribution that we will call $\pi(s)$ [150].

The second technique, called *Quantum Fast Forwarding*, is more involved [11]. Its objective is to simulate an approximation to $\mathcal{D}^t$ in $O(\sqrt{t})$ applications of the quantum walk operator, even if that entails some failure probability. To do so, one first proves that $\Pi_{\mathcal{A}}\mathcal{M} = \mathcal{D}$ [11, Lemma 1], as we already saw in (2.41a). Using this,

$$\mathcal{D}\left|\phi_j\right\rangle\left|0\right\rangle = \cos\varphi_j \left|\phi_j\right\rangle\left|0\right\rangle \Rightarrow \mathcal{D}^t\left|\phi_j\right\rangle\left|0\right\rangle = \cos^t(\varphi_j)\left|\phi_j\right\rangle\left|0\right\rangle. \tag{2.50}$$

We will further modify the definition of the quantum walk such that in this algorithm it will be

$$W = (2\Pi_{\mathcal{A}} - \mathbf{1})\mathcal{M}, \tag{2.51}$$

with $\mathcal{M}$ defined in (2.37). Since W can be understood as 2 rotations, we have [11, Proposition 1] and [236]

$$\Pi_{\mathcal{A}}W^t\left|\phi_j\right\rangle\left|0\right\rangle = T_t(\mathcal{D})\left|\phi_j\right\rangle\left|0\right\rangle = T_t(\cos\varphi_j)\left|\phi_j\right\rangle\left|0\right\rangle = \cos(t\varphi_j)\left|\phi_j\right\rangle\left|0\right\rangle, \tag{2.52}$$

where $T_t(\cos\varphi)$ is a Chebyshev polynomial of the first kind. In fact, in Chapter 3 and in particular, in equation (3.38), we will see that powers of quantum walk implement Chebyshev polynomials in general. Now, we compare

$$\cos^t(\theta) = 1 - \frac{t\theta^2}{2} + O(t^2\theta^4), \quad \text{and} \quad \cos(t'\theta) = 1 - \frac{t'^2\theta^2}{2} + O(t'^4\theta^4). \tag{2.53}$$

From this we check that setting $t' = \sqrt{t}$, both expressions coincide up to second order. Notice the fact that W^l implements $T_l(\cos\varphi_j)$ on the eigenstates $\left|\phi_j\right\rangle\left|0\right\rangle$, and we can use that to approximate $\cos^t(\varphi_j)$, the action of $\mathcal{D}^t$. In particular, we will implement a

Chebyshev series such that [11, Lemma 3]

$$\left|\cos^t(\theta) - \sum_{l=0}^{\lceil\sqrt{Ct}\rceil} p_l \cos(lt\theta)\right| \leq \epsilon, \tag{2.54}$$

for some coefficients p_l defined in their equation 6, and $C = 2\ln(2/\epsilon)$ the precision of the approximation. More specifically, we implement [11, Equation 8]

$$\sum_{l=0}^{\tau} p_l \Pi_{\mathcal{A}} W^l \left|\phi_j\right\rangle \left|0\right\rangle = \sum_{l=0}^{\tau} p_l T_l(\mathcal{D}) \left|\phi_j\right\rangle \left|0\right\rangle \tag{2.55}$$

for $\tau = O(\sqrt{t}\log\epsilon^{-1})$ to approximate $\mathcal{D}^t$, our objective.

We now have to explain how to implement the sum of Chebyshev terms. For that, we resort to a very standard technique that will be used in later chapters too. It is called *Linear Combination of Unitaries* (LCU) decomposition. The idea is that if we want to implement (2.55), we use the following two operators:

$$\text{Prep} : \left|0\right\rangle_a \left|\phi\right\rangle \mapsto \frac{1}{\sqrt{\sum_j p_j^2}} \sum_l \sqrt{p_l} \left|l\right\rangle_a \left|\phi\right\rangle, \tag{2.56a}$$

and

$$\text{Sel} : \left|l\right\rangle_a \left|\phi\right\rangle \mapsto \left|l\right\rangle_a T_l(\mathcal{D}) \left|\phi\right\rangle. \tag{2.56b}$$

Note that Sel is a controlled application of (2.52). Combining them, we have

$$(\left|0\right\rangle\left\langle 0\right|_a \otimes \mathbf{1})\text{Prep}^\dagger \cdot \text{Sel} \cdot \text{Prep} \left|0\right\rangle_a \left|\phi\right\rangle = \left|0\right\rangle_a \sum_l p_l T_l(\mathcal{D}) \left|\phi\right\rangle. \tag{2.57}$$

Using $Q = \text{Prep}^\dagger \cdot \text{Sel} \cdot \text{Prep}$, we can implement the Chebyshev series with some failure probability. Using these tools, Ref. [7] proposes Algorithm 5.

The Quantum Fast Forward subroutine has complexity $O(\sqrt{t\log\epsilon^{-1}})$ times the complexity of a single quantum walk step (2.51). The key result from Ref. [7] is that we can lower bound the success amplitude as

$$||\Pi_M \mathcal{D}^t(s) \left|\pi_U\right\rangle|| \geq \Omega(\log^{-1} T). \tag{2.60}$$

in (2.59). Consequently, Algorithm 5 has complexity

$$O(\mathtt{S}\sqrt{\log \mathrm{HT}} + \sqrt{\mathrm{HT}}(U + \mathtt{C})\sqrt{\log \mathrm{HT} \log\log \mathrm{HT}}), \tag{2.61}$$

Algorithm 5 Fast-forward-based quantum search [7, Theorem 3]

Input: Ergodic reversible Markov chain $\mathcal{P}$, graph $G(X, E)$, upper bound to Hitting Time HT.
Output: A marked item if it exists, or 'no marked element exists' message.
Define $T = O(HT)$ and $S = \left\{1 - \frac{1}{r} : r \in 1, 2, 4, \ldots, 2^{\log T + O(1)}\right\}$
for $O(\log T)$ steps **do** Amplitude Amplification over
    Use S to prepare

$$\sum_{t=1}^{T} \frac{1}{\sqrt{T}} |t\rangle \sum_{s \in S} \frac{1}{\sqrt{|S|}} |s\rangle |\pi\rangle . \tag{2.58}$$

    Perform measurement $\{\Pi_M, \mathbf{1} - \Pi_M\}$. If the output is marked, output it.
    Quantum Fast Forward the state

$$|s\rangle |t\rangle |\pi_U\rangle \mapsto |1\rangle |t\rangle |s\rangle \mathcal{D}^t(s) |\pi_U\rangle + |0\rangle |\ldots\rangle . \tag{2.59}$$

    with precision $\epsilon = O(\log^{-1} T)$, where $|0\rangle |\ldots\rangle$ encodes a failure state.
end for
Measure the first register.
if $x \in M$ **then**
    Output x,
else
    Output 'No element exists'.
end if

whenever an upper bound HT over the classical hitting time is known [7, Theorem 3]. If it is unknown, then one may repeat the search with an exponentially increasing number of steps until an element is found, or we flag that no element exists [50]. This increases the complexity by $O(\log \mathrm{HT})$. Alternatively, we could use fixed-point amplitude amplification [264]. In this case, we cannot use phase estimation to reflect over the stationary state, needing instead to use S to prepare it from a computational basis state.

2.3.2. Mixing time and Monte Carlo algorithms

In the previous subsection, we analyzed the capability of quantum walks to find a marked item in a Markov chain. In this one, we would instead like to study the convergence to the limiting distribution. For example, if π is the stationary distribution, we might define the mixing time as follows.

Definition 3 (Mixing time) *The Mixing time MT is defined as*

$$MT(\epsilon) = \min\{t | \forall t' > t, \forall s_0, D(\mathcal{P}^{t'} s_0, \pi) \leq \epsilon\}, \tag{2.62}$$

where D indicates the distance between the distributions

$$D(p, q) = \frac{1}{2} \sum_{v=1}^{N} |p_v - q_v|, \tag{2.63}$$

and v indicates the vertices of the Markov chain.

The validity of this definition rests on the fact that, for classical random walks,

$$\lim_{t \to \infty} \mathcal{P}^t s_0 = \pi, \tag{2.64}$$

independently of the initial state s_0 [201].

If we try to do this equivalently for quantum walks, however, we fail: due to the walk being unitary, there is no limiting distribution. One can see this by noticing that [201, Chapter 7]

$$\frac{1}{2} \| \left|\psi(t+1)\right\rangle - \left|\psi(t)\right\rangle \|^2 = \frac{1}{2} \| W^t (W - \mathbf{1}) \left|\psi(0)\right\rangle \|^2 = 1 - \mathfrak{R}(\langle\psi(0)|W|\psi(0)\rangle), \tag{2.65}$$

where $\mathfrak{R}$ indicates the real part. Since the norm above does not need to approach 0 and is in fact independent of t, the quantum walk will not converge to a stationary state. For this reason, we will have to look for alternative approaches to obtaining some sense of limiting distribution.

One possible option to define a quantum equivalent is to take a time-averaged sampling, e.g., given some possibly entangled auxiliary register $|a\rangle$ and a vertex register $|v\rangle$

$$\lim_{T \to \infty} \bar{p}_v(T) = \lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{a} |\left\langle v, a | \psi(t) \right\rangle|^2. \tag{2.66}$$

Let us analyze whether $\bar{p}_v(T)$ converges. Imagine that we start from a quantum state

$$\left|\psi(0)\right\rangle = \sum_{k,a} c_{a,k} \left|\phi_k, a\right\rangle. \tag{2.67}$$

Operator W can be similarly decomposed

$$W = \sum_{k,a} e^{2i\varphi_{k,a}} \left|\phi_k, a\right\rangle \left\langle\phi_k, a\right|. \tag{2.68}$$

The eigenvalues $\cos\varphi_{k,0}$ and the eigenvectors $|\phi_k, 0\rangle$ extend to the whole bipartite space $\cos\varphi_k$ and $|\phi_k\rangle|0\rangle$ that we previously defined in $\mathcal{A}$. Consequently, the evolved state is

$$|\psi(t)\rangle = \sum_{k,a} c_{a,k} e^{2i\varphi_{k,a}} |\phi_k, a\rangle . \tag{2.69}$$

Substituting into (2.66),

$$\begin{aligned} \bar{p}_v(T) &= \frac{1}{T}\sum_{t=0}^{T-1}\sum_b |\langle v, b|\psi(t)\rangle|^2 \\ &= \sum_{a,a',b}\sum_{k,k'} c_{a,k}c^*_{a',k'} \langle\phi_k, a|v, b\rangle\langle v, b|\phi_{k'}, a'\rangle \times \frac{1}{T}\sum_{t=0}^{T-1} e^{2i(\varphi_{k,a}-\varphi_{k',a'})t}. \end{aligned} \tag{2.70}$$

On the other hand, [201, Chapter 7]

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1} e^{2i(\varphi_{k,a}-\varphi_{k',a'})t} = \begin{cases} 1 & \varphi_{k,a} = \varphi_{k',a'} \\ 0 & \varphi_{k,a} \neq \varphi_{k',a'} \end{cases}, \tag{2.71}$$

because if the eigenvalues are different, we have an average over the approximate roots of the unit that cancels out asymptotically. Consequently, since

$$\lim_{T\to\infty}\bar{p}_v(T) = \sum_{a,b}\sum_k |c_{a,k}|^2 |\langle\phi_k, a|v, b\rangle|^2, \tag{2.72}$$

we have proven the limiting distribution exits. However, this distribution depends on the initial state coefficients $c_{k,a}$, and not only on the first eigenstate but on all of them, in contrast to the classical random walk [201].

Using this limiting distribution, we can study a possible definition of the quantum convergence time. In contrast to the hitting time, the quantum advantage in the mixing time will depend on the specific graph, and while in some systems the quantum mixing time displays a quadratic advantage [201, Table 7.1], in others the quantum walks mix more slowly than their classical counterparts [69].

Monte Carlo algorithms

The fact that the limiting probability distribution $\lim_{t\to\infty}\bar{p}_v(t) \neq \pi$ limits the usefulness of the previous definition. Instead, one may wish to prepare a coherent version of the stationary distribution, $|\pi\rangle$. This task is precisely the definition of operator `S`, which was not previously explained but was used in the search algorithms

above. Indeed, one possible way to solve the mixing problem would be running the finding problem in reverse: since all search algorithms start from the coherent stationary state and end up in a marked item if we had a marked item we could reverse the algorithm to find the stationary state. There are some problems, though, since a constant overlap with such marked states is not sufficient to achieve similar running time [68].

There are some special cases where it is possible to prepare coherent versions of the stationary state using phase estimation and measurements: since we know that the stationary state has eigenvalue 1, using phase estimation and measurements may probabilistically project the system into the stationary state. This opens the door to performing a digital simulation of an adiabatic algorithm.

In adiabatic quantum simulated annealing [222, 223, 259], we start with the stationary state $|\pi_0\rangle$ of a simple Markov chain $\mathcal{P}_0$. For example, any symmetric Markov chain has a uniform superposition as stationary state, $|\pi_0\rangle = H^{\otimes n} |0\rangle^{\otimes n}$ [173]. Then, we look for a list of Markov chains $\mathcal{P}_0, \dots \mathcal{P}_r$ that interpolates between the initial simple Markov chain $\mathcal{P}_0$ and the target $\mathcal{P} = \mathcal{P}_r$. At this point, one strategy is to use phase estimation and projective measurements to make the system evolve from $|\pi_i\rangle$ to $|\pi_{i+1}\rangle$.

Algorithm 6 Quantum Zeno effect annealing

1: **Input**: List of ergodic Markov chains $\{\mathcal{P}_i\}_{i\in\{1,\dots,r\}}$ with corresponding quantum walks $\{W_i\}_i$, such that their stationary states fulfill $|\langle\pi_{i-1}|\pi_i\rangle|^2 \geq p$.
2: **Output**: Stationary state $|\pi\rangle$ of the Markov chain $\mathcal{P}$.
3: Initialize the stationary distribution $|\pi_0\rangle$, often a uniform distribution.
4: **for** i in $1, \dots, r$ **do**:
5: Implement phase estimation $P(W_i)$ with precision $O(1/\Delta)$.
6: Measure the eigenvalue.
7: **if** Eigenvalue is not 1 **then**
8: Restart algorithm.
9: **end if**
10: **end for**
11: Output $|\pi\rangle$.

This algorithm has complexity $O(1/\Delta)$, offering a quadratic speedup in the eigenvalue gap, as the mixing time of the classical algorithm scales as $O(1/\delta)$ [259]. Two possible modifications of Algorithm 6 are the following. First, phase estimation might be substituted by randomized evolution [223]. In other words, using a random number of applications of the quantum walk W_i at each step. Such a random number should be taken at uniform from $[0, O(1/\Delta)]$. A second one is the so-called Zeno effect with rewind [161]. In this variant, instead of discarding the state whenever we

measure an eigenvalue different from 1, we can rewind the process. Using the projectors $\Pi_j = |\pi_j\rangle \langle\pi_j|$, $\Pi_j^\perp = \mathbf{1} - \Pi_j$ and $p_j = |\langle\pi_j|\pi_{j+1}\rangle|^2$, we find that transitions between Π_j and Π_{j+1}, or $\Pi_j^\perp$ and $\Pi_{j+1}^\perp$ occur with probability p_j. Similarly, the transitions $\Pi_j^\perp \leftrightarrow \Pi_{j+1}$ and $\Pi_j \leftrightarrow \Pi_{j+1}^\perp$ occur with probabilities $1 - p_j$. Consequently, even if at a given step we measure $\Pi_j^\perp$ we can later on recover one of the $|\pi_i\rangle$ states.

A third alternative to the use of projective measurements is to amplify the overlap between consecutive $|\pi_j\rangle$ [259]. Since we are only given a lower bound for the overlap between two states, it is useful to use Grover's $\pi/3$ algorithm instead of Grover's original one. We define

$$U_{i;0} = \mathbf{1}; \qquad R_i = e^{i(\pi/3)}\Pi_i + \Pi_i^\perp; \qquad U_{i;m+1} = U_{i;m} \cdot R_i \cdot U_{i;m}^\dagger \cdot R_{i+1} \cdot U_{i;m}, \quad (2.73)$$

where m controls the number of amplification steps and i the Markov chain. Using these definitions, we can obtain

$$|\pi\rangle = \prod_{j=0}^{r} U_{j;m} |\pi_0\rangle , \quad (2.74)$$

with error probability ϵ, if we choose

$$m \geq L = \frac{12r \log(2r/\epsilon)}{\log(1/(1-p))}, \quad (2.75)$$

and assume overlap $p_j \geq p$ for all j [259]. As before, the rotations around $|\pi_j\rangle$ are performed with the help of the phase estimation operator.

The quantum simulated annealing techniques explained above are particularly useful for preparing ground states of Hamiltonians. To achieve this, we can aim to prepare Gibbs states of the form

$$|\pi^\beta\rangle = Z^{-1/2} \sum_\lambda e^{-\beta H/2} |\lambda\rangle = Z^{-1/2} \sum_\lambda e^{-\beta\lambda/2} |\lambda\rangle , \quad (2.76)$$

where $|\lambda\rangle$ is an eigenstate of the Hamiltonian with eigenvalue $\lambda = E_\lambda$, and Z is the partition function that normalizes the state. When β, the inverse temperature, goes to infinity, only the eigenstate with the lowest energy survives.

These techniques can be especially helpful in the context of the Metropolis-Hastings algorithm. The key idea of the Metropolis-Hastings approach is to engineer a series of 'rapidly' mixing Markov chains [182, 120]. To do so, one starts from the detailed

balanced equation of a reversible Markov chain, (2.31), $p_{yx}\pi_x = p_{xy}\pi_y$. Consequently,

$$\frac{p_{yx}}{p_{xy}} = \frac{\pi_y}{\pi_x}. \tag{2.77}$$

The next step is to divide each transition into a proposal probability $T_{y,x}$ and an acceptance probability $A_{y,x}$, such that

$$p_{yx} \propto T_{y,x}A_{y,x}. \tag{2.78}$$

Consequently, we need to choose $A_{y,x}$ such that

$$\frac{A_{y,x}}{A_{x,y}} = \frac{\pi_y}{\pi_x}\frac{T_{x,y}}{T_{y,x}}. \tag{2.79}$$

A common choice that fulfills the previous equation is to take

$$A_{y,x} = \min\left(1, \frac{\pi_y}{\pi_x}\frac{T_{x,y}}{T_{y,x}}\right). \tag{2.80}$$

In particular, we can take a parameterized family of Markov chains

$$\mathcal{P}_\beta(y,x) = \begin{cases} T_{y,x}A_{\beta;y,x} & y \neq x \\ 1 - \sum_z T_{z,x}A_{\beta;z,x} & y = x, \end{cases} \tag{2.81}$$

where

$$A_{\beta;y,x} = \min\{1, e^{-\beta(E_y - E_x)}\}, \tag{2.82}$$

E_x indicates the 'energy' of a state x, and $T_{y,x} = 1/n_e = T_{x,y}$ for n_e the number of edges connected to x or y. This means that the Markov chain has stationary state $\pi_x = Z^{-1}e^{-\beta E_x}$, and in the limit $\beta \to \infty$, π concentrates all its probability mass on the state x with lowest energy E_x. Similarly, if $\beta = 0$, then π is the uniform distribution over all states.

Classical simulated annealing has complexity [223]

$$O\left(\frac{\|H\|}{\gamma}\frac{\log(d/\epsilon^2)}{\delta}\right), \tag{2.83}$$

where δ is the eigenvalue gap of the Markov chains $\mathcal{P}_\beta$, γ the energy gap of the target Hamiltonian, and $d = |\Omega|$ the size of the search space. In contrast, Algorithm 6 has

complexity [223, Eq. 5]

$$O\left(\left(\frac{\|H\|}{\gamma}\right)^2 \frac{\log^2(d/\epsilon)\log d\log(1/\epsilon)}{\Delta}\right), \tag{2.84}$$

and the one using Grover's $\pi/3$ amplitude amplification [259, Eq. 3],

$$O\left(\frac{\|H\|}{\gamma}\frac{\log d}{\Delta}\log\left(\frac{\|H\|}{\gamma}\log d\right)\right). \tag{2.85}$$

Finally, the Zeno algorithm has been generalized to quantum Hamiltonians, where the stationary state Eq. (2.76) contains a superposition of eigenvalues that no longer represent computational basis (classical) states. In this case, we can also obtain a quadratic speedup in the eigenvalue gap Δ, see Eq. S50 in Ref. [265].

2.4. QFold

Proteins are one of the most fundamental molecules in biochemistry. In contrast to many others, such as lipids, they exhibit very large variability in composition, which makes them a good candidate for numerous biochemical tasks. Proteins are made of chains of amino acids, which in turn are composed of a backbone chain of an amino group -NH, a central carbon labeled α, a carboxyl group -COOH, and a side chain of other atoms, also called radical, attached to the α Carbon. The different composition possibilities of the side chain define 20 amino acids.

As proteins play a central role in biochemistry, understanding their specific functions or even designing proteins with specific roles is a very important problem. However, this function is not only determined by their amino acid decomposition or by the order those amino acids appear in the protein chain, but also by how such a chain is folded into its natural configuration, often much more complex to determine. This folding is largely determined by two angles per amino acid, commonly called ψ and ϕ, see Fig. 2.6.

A breakthrough in the understanding of how proteins fold happened in 1961 when the scientist Christian Anfinsen showed that a denaturalized ribonuclease peptide (small protein) would always recover its shape and function if put back into its usual environment [10]. This and similar experiments led Anfinsen to postulate that proteins tend to fold in their thermodynamic ground state as determined by the aminoacid chain [9], and would ultimately win him the Chemistry Nobel Prize in 1972.

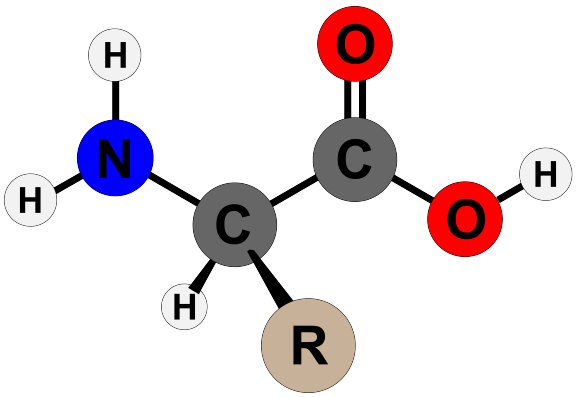


Figure 2.5 **Structure of an aminoacid.** The radical, indicated by the letter R, is a chain between 20 different options that determine the specific amino acid. Taken from Wikipedia, under CC-BY-SA 3.0 license.

While not universally true, as there are proteins with various stable configurations or others that have none, his 'thermodynamic hypothesis' has guided the research in this area since then. Unfortunately, it is known that the protein structure prediction problem is NP-hard [118], even for the simplest toy models [31]. This seems to be at odds with the fact that most proteins can fold in timescales of seconds or less. However, while baffling, scientists have so far not been able to find a computationally fast and inexpensive method to understand how proteins fold.

It is for this reason, and its NP-hardness, that this problem makes a good candidate for simulated annealing approaches, and for their quantum equivalent algorithms too. In fact, until very recently, simulated annealing has been the default approach for one of the most popular protein structure prediction computer libraries, Rosetta, [80]; and in distributed computational efforts such as Rosetta@Home [242, 81]. With the development of quantum walks over the last decade and a half, and the promise of quadratic speedups in a computationally hard problem, some effort has been put into developing tailored quantum algorithms [20, 208, 198, 95, 22, 199, 196, 261] for this problem and related ones [190, 23]. Perhaps because the computational complexity of the problem does not decrease for toy models, almost all of these manuscripts have assumed simplified lattice models.

In contrast, in recent years powerful deep learning techniques have also been applied to this problem [216, 127], producing extremely promising results in the Critical Assessment of Techniques for Protein Structure Prediction (CASP) competition [189]. In our article, we propose implementing a quantum annealing search starting from the state that the deep learning model has suggested [62]. The deep learning that we

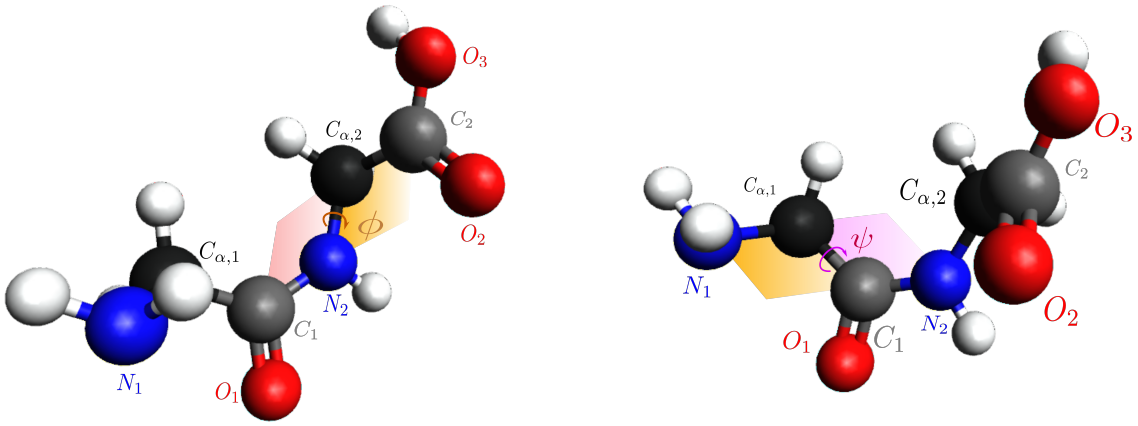


Figure 2.6 **Example of the smallest dipeptide: the glycylglycine**. Each amino acid has the chain (Nitrogen-C_α-Carboxy). Different amino acids would have a different side chain attached to the C_α instead of Hydrogen as is the case for Glycine. In each figure, we depict either angle ϕ or ψ. Angle ψ is defined as the torsion angle between two planes: the first one defined by the three atoms in the backbone of the amino acid (N_1, $C_{\alpha,1}$, C_1), and the second by the same atoms except substituting the Nitrogen in that amino acid by the Nitrogen of the subsequent one: ($C_{\alpha,1}$, C_1, N_2). For the ϕ angle the first plane is made out of the three atoms in the amino acid (N_2, $C_{\alpha,2}$, C_2) whereas the second plane is defined by substituting the Carboxy atom in the amino acid by the Carboxy from the preceding amino acid: (C_1, N_2, $C_{\alpha,2}$).

use as an initial guess will be *Minifold* [3], a small AlphaFold-inspired deep learning module with enough precision for our purposes.

2.4.1. The QFold algorithm

Relying on Anfinsen's hypothesis, our work aims to use a quantum Metropolis algorithm to find the thermodynamic ground state of the system. This is a search problem, but since there is no oracle to mark the state, the objective is instead to find the stationary state of a rapidly mixing Markov chain that favors the states with the lowest energy. Consequently, our procedure will fit into the scheme of Monte Carlo algorithms discussed in the previous subsection, and in particular, the Metropolis-Hastings algorithm defined in (2.81) and (2.82).

As we reviewed in that section, there are a few quantum Metropolis proposals that could work on this problem. However, most of them use the expensive quantum phase estimation subroutine and come with theoretical guarantees of performance, which is not the case in the commonly used classical Metropolis algorithm. As a result, Lemieux et al. [162] proposed to use a heuristic algorithm more in the spirit of the classical

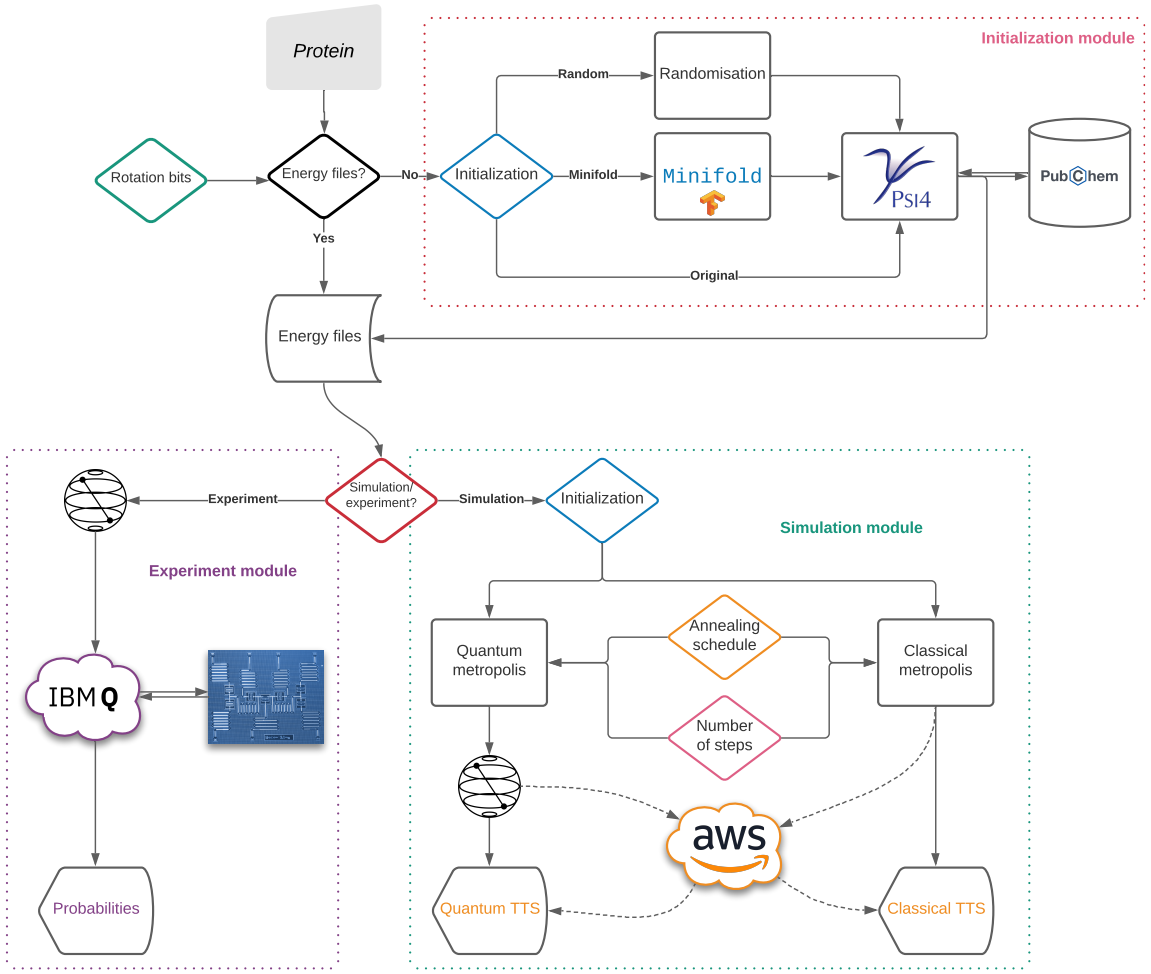


Figure 2.7 **Flow chart of the QFold algorithm** [62]. QFold has several functionalities integrated, that could be summarized in an initialization module, a simulation module, and an experiment module. We denote by diamonds each of the decisions one has to make. The top part constitutes the initialization module, where *Minifold*, a deep learning module, can be used to get a guess of the correct folding, and *Psi4* and the original geometry from *PubChem* are used to calculate the energies of rotations. The bottom half represents the `experiment` or `simulation` algorithms, which output either probabilities or quantum/classical TTS of the corresponding metropolis algorithms, making use of *Qiskit*.

Metropolis algorithm. It simply applies quantum walk steps to make the state evolve over time

$$|\psi(T)\rangle := \tilde{W}_T ... \tilde{W}_1 |\pi_0\rangle . \tag{2.86}$$

The quantum walk step $\tilde{W}$ is a modification of Sgedy's original algorithm that instead of duplicating $\mathcal{H}$ uses a coin. It uses 3 quantum registers: $|\cdot\rangle_S$ indicating the current

state of the system, $|\cdot\rangle_M$ that indexes the possible moves one may take according to $T_{y,x}$ in (2.81), and $|\cdot\rangle_C$ the Boltzmann coin register. We may also have auxiliary registers $|\cdot\rangle_A$. Then the quantum walk operator is defined as

$$\tilde{W} = RV^\dagger B^\dagger FBV, \tag{2.87}$$

where V prepares in register $|\cdot\rangle_M$ a superposition over all possible steps one may take; B rotates the coin qubit $|\cdot\rangle_C$ to have amplitude of $|1\rangle_C$ corresponding to the acceptance probability indicated by (2.82); F changes the $|\cdot\rangle_S$ register to the new configuration conditioned on the value of $|\cdot\rangle_M$ and $|\cdot\rangle_C = |1\rangle_C$; and $R = \mathbf{1} - 2\,|0\rangle\langle 0|_{MCA}$.

The metric chosen to compare the classical and quantum Metropolis algorithms is called Total Time to Solution [162]. Its role is to measure the expected number of quantum walk steps it would take to find a solution if we allow for restarts of the algorithm. Specifically,

$$TTS(t) := t\frac{\log(1-\delta)}{\log(1-p(t))}, \tag{2.88}$$

where t is the number of steps we take in the algorithm, δ is some target probability threshold, and $p(t)$ is the success probability after t steps. For each problem, we are interested in the minimum TTS, either classical or quantum, as this is the expected TTS we would recover if we were to take the optimal number of walk steps before measuring.

In our article, we explore different annealing schedules [243] including

- `Boltzmann` or `logarithmic` implements the famous logarithmic schedule [98]

$$\beta(t) = \beta(1)\log(te) = \beta(1)\log(t) + \beta(1). \tag{2.89a}$$

  Notice that the multiplication of t times e is necessary to make a fair comparison with the rest of the schedules so that they all start in $\beta(1)$.

- `Cauchy` or `linear` implements the simple schedule given by

$$\beta(t) = \beta(1)t. \tag{2.89b}$$

- `geometric` defines [140]

$$\beta(t) = \beta(1)\alpha^{-t+1}, \tag{2.89c}$$

  where $\alpha < 1$ is a parameter heuristically set to 0.9.

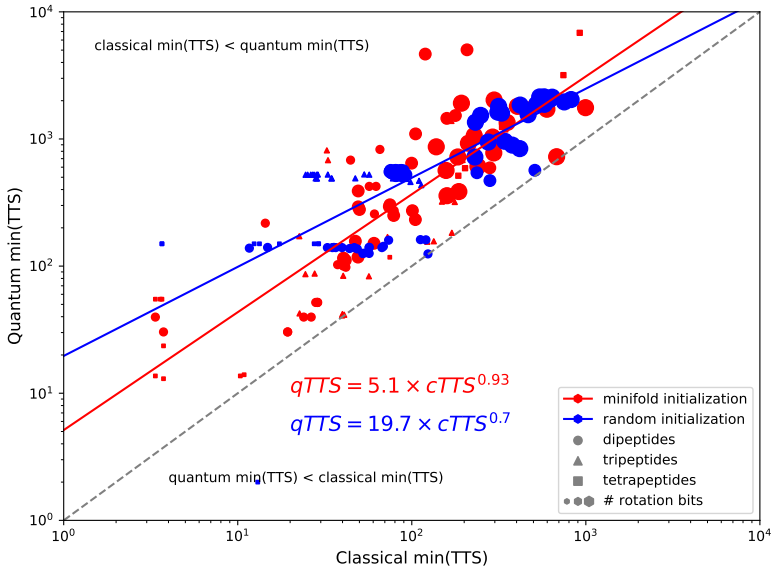


Figure 2.8 **Comparison of the classical and quantum minimum TTS achieved for the simulation of the quantum Metropolis algorithm with** $\beta = 10^3$, for 10 dipeptides (with $b = 3, 4, 5$ rotation bits of precision in the angles), 10 tripeptides ($b = 2$) and 4 tetrapeptides ($b = 1$), also showing the different initialization options (random or minifold), and the best fit lines [62]. The dashed grey line separates the space where the quantum TTS is smaller than the classical TTS. The key aspect to notice in this graph is that although for smaller instances the quantum algorithm does not seem to match or beat the times achieved by the classical Metropolis, due to the exponent being smaller than one (either 0.89 or 0.53 for minifold or random respectively) for average size proteins we can expect the quantum advantage to dominate. In the main text, we explain why the random initialization exponent seems more favorable than the minifold exponent and discuss further details respectively.

- And finally `exponential` uses

$$\beta(t) = \beta(1) \exp(\alpha(t-1)^{1/N}), \tag{2.89d}$$

where α is again set to 0.9 and N is the space dimension, which in this case is equal to the number of torsion angles.

2.4.2. Results

In our article, we analyze two kinds of experimental results: those coming from simulations and others from actual experiments in quantum computers. As mentioned, we test whether the quantum or classical Metropolis algorithms achieve a better

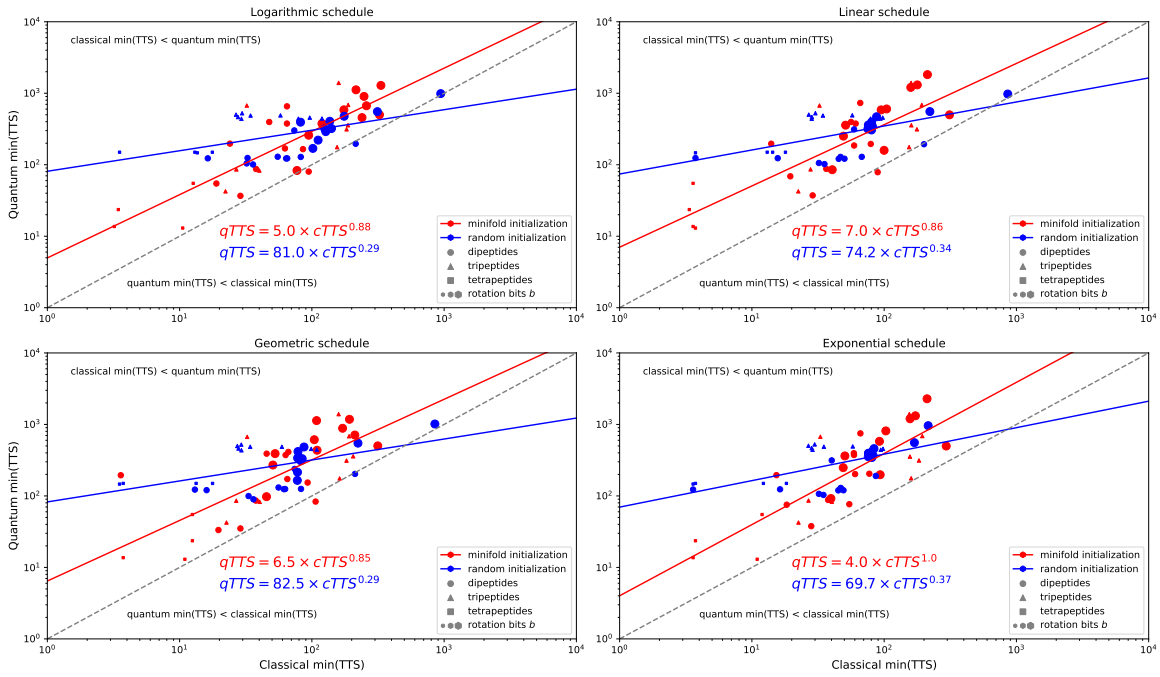


Figure 2.9 **Comparison of the classical and quantum minimum TTS achieved for a number of small peptides with several annealing schedules** [62]. This figure shows the different initialization options (random or minifold) and annealing schedules (`Boltzmann/logarithmic`, `Cauchy/linear`, `geometric` and `exponential`), and the best fit lines. The dashed grey line depicts the diagonal. The corresponding fit exponents are given in table 2.1, where we observe a small polynomial quantum advantage. On the other hand, using an exponential schedule does not seem to give any advantage when used with a minifold initialization.

Total Time to Solution. In the simulation results, we are particularly interested in understanding how this metric evolves for larger system sizes, and we depict them in Fig. 2.8 and Fig. 2.9. In this chapter, we have thoroughly explained that the expected quantum advantage is polynomial, but since the algorithm is heuristic, we are interested in finding the actual exponent that encapsulates it. A quantum advantage will be revealed whenever the quantum minimum TTS grows as a power of the classical minimum TTS with an exponent lower than 1. For instance, we would measure a quadratic quantum advantage if the exponent were 0.5. We compute the exponent using the standard technique of linear least square fitting in the logarithmic scale for the classical and quantum minimum TTS achieved.

An important remark is in order. This figure seems to suggest that the quantum advantage when we use minifold as an initialization module is smaller than whenever we use a random initialization. While this may seem to indicate that our proposed

deep learning initialization is actively harmful to the predictive power of the quantum Metropolis these results should be interpreted differently: for the smallest instances, and in particular, when using a random initialization, the minimum quantum TTS value is achieved for $t = 2$. This implies that in such cases the algorithm is avoiding the use of the Metropolis algorithm, and instead repeatedly choosing and outputting random points from the search space. The reason is that in the smallest instances, the multiplicative prefactors of the quantum walk have a larger effect than the eigenvalue gap that determines the quantum advantage. Consequently, the quantum advantage found using the minifold initialization is likely to be more representative of the actual large-size behavior.

The estimated quantum advantages indicated in Table 2.1, and shown in Fig. 2.8 and Fig. 2.9, largely match the value 0.7549 computed in the original reference by Lemieux et al. [162] for a different problem, although are slightly worse. Unfortunately, I believe the most important conclusion that can be extracted from these simulations is rather a negative result: as argued in [19] and mentioned in our article, it is very unlikely that the $\simeq 0.9$ polynomial quantum advantage that we observe might be of any use if one takes into account the error correction overhead required to operate a quantum computer. On the other hand, our results are exploratory, due to the incapability to simulate large enough systems without an actual quantum computer, so different modifications might achieve larger and more useful quantum advantages. Additionally, we assumed that the optimal choice of parameters is the same for both quantum and classical metropolis algorithms, but this choice is not necessarily true.

Fit exponents		
Schedule	Random initial.	Minifold initial.
Fixed β	0.70 ± 0.08	0.93 ± 0.06
Logarithmic	0.29 ± 0.07	0.88 ± 0.09
Linear	0.34 ± 0.07	0.86 ± 0.11
Exponential	0.37 ± 0.07	1.00 ± 0.12
Geometric	0.29 ± 0.07	0.85 ± 0.18

Table 2.1 **Table of scaling exponents for different annealing schedules and initialization options** [62]. The peptides are the same, except that for fixed β we have also included dipeptides with 5 bits of precision, which is costly for the rest of the schedules. For fixed β, the value heuristically chosen was $\beta = 1000$, while the initial β value in each of the schedules, defined in (2.89), is $\beta(1) = 50$. The uncertainty is expressed via the standard deviation in the expected exponent, calculated with the bootstrapping method [90].

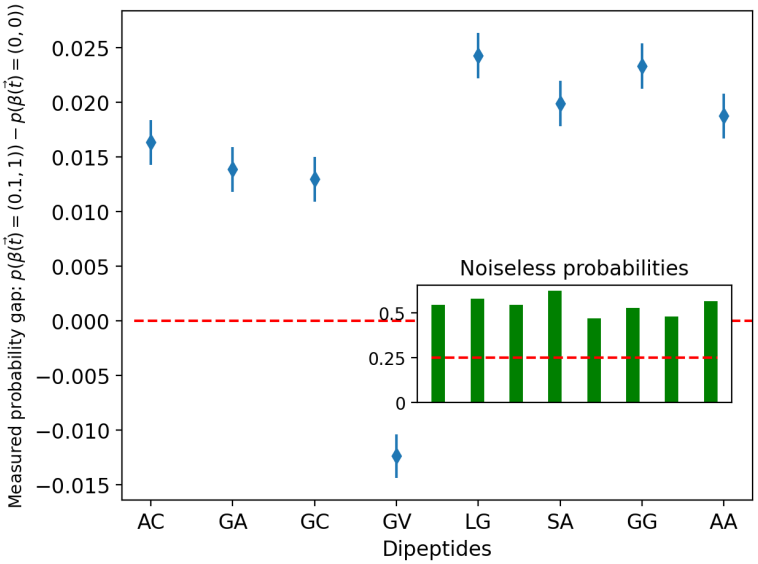


Figure 2.10 **Results from hardware measurements** corresponding to the experimental realization of the quantum Metropolis [62]. For each dipeptide we perform a student t-test to check whether the average success probabilities for $\beta(\boldsymbol{t}) = (0, 0)$ and $\beta(\boldsymbol{t}) = (0.1, 1)$ are actually different [230]. The largest p-value measured in all 8 cases is $3.94 \cdot 10^{-18}$, indicating that in all cases the difference is significant. For each dipeptide, we run 163840 times the circuit, and for the baseline 204800 times.

In addition to simulation results, we also execute a small proof-of-concept experiment in an actual IBM Quantum processor. Since NISQ devices such as this one are very limited in the depth of the circuits they can execute, we are restricted to very small search spaces, with only four positions. Consequently, our objective here is just to understand whether we can overcome the error in the circuit and measure probability differences between setting $\beta = 0$, and $\beta \neq 0$. The results are depicted in Fig. 2.10 and indicate that in 7 out of the 8 dipeptides tested, the quantum algorithm correctly increases the probability of measuring the correct state. Unfortunately, we do not have a good model for why the remaining peptide achieves a statistically significant negative result, although it is reasonable to assume this is due to the physical imperfections of the superconducting chip.

Beyond our article, recent years have seen tremendous advances in the capabilities of deep learning models to predict the structure of proteins. It is encouraging to see that the latest AlphaFold model can often achieve precision similar to costly laboratory methods [127]. Moreover, these models have been open-sourced, followed by the processing of large databases of proteins that were previously unavailable [244]. On the quantum walk side, the heuristic quantum Metropolis algorithm was later on used

for quantum ground state preparation, see [161]. In our group, its application to other problems in Machine Learning [60] and Bayesian inference [91] has been explored too.

2.5. Results

- We have explained Grover's algorithm, as well as several of their generalizations such as amplitude amplification, and fixed-point amplitude amplification. The latter has the advantage that probability does not decrease significantly if we apply more amplification steps than necessary.
- In Section 2.3 we have analyzed the quantum walk technique, which displays a similar mathematical structure as Grover's algorithm and may also be understood as its generalization.
- In particular, we have explained how quantum walks can often achieve a quadratic speedup in search problems due to the quadratically larger quantum eigenvalue gap, $\Delta = O(\sqrt{\delta})$. We have indicated a series of techniques that extend the range of search problems where quadratic advantages might be found, culminating in the Quantum Fast Forwarding algorithm that applies to ergodic Markov chains.
- Such quadratic advantage is however not always possible in mixing problems, where the objective is to prepare the stationary state of the Markov chain.
- One exception to the previous rule is the Metropolis algorithm: We have studied how the Metropolis algorithm may benefit from the quantum walk technique, providing a quadratic advantage to prepare stationary distributions of Markov chains, with applicability to hard combinatorial optimization problems.
- We have computationally analyzed a heuristic quantum Metropolis algorithm which provides a modest quantum advantage in the exponent in protein folding [62], machine learning [60] and Bayesian inference problems [91]. This advantage should be especially helpful in cases where the problem is NP-hard, and therefore it is not possible to exploit the problem structure. A clear example of this is precisely protein folding.
- We have further analyzed several annealing schedules for the heuristic quantum metropolis algorithm and found that most give a small improvement over constant-β walks. These results confirm and strengthen similar results found in one previous article by Lemieux et al., [162], for simpler Ising chain models.

Chapter 3

Quantum Linear Algebra

> Not only do these new algorithms promise exponential speedups over classical algorithms, but they do so for eminently-practical problems, involving machine learning, clustering, classification, and finding patterns in huge amounts of data. So, do these algorithms live up to the claims? That's a simple question with a complicated answer.
>
> Scott J. Aaronson, *Quantum Machine Learning Algorithms: Read the Fine Print*

3.1. Objectives

- Understand the mathematical techniques used in quantum linear algebra.
- Understand the limitations of quantum linear algebra techniques, in particular, those related to data loading and readout.
- Use quantum linear algebra techniques to improve state-of-the-art linear programming methods.
- Understand dequantization and how it limits many of the applications of quantum linear algebra to Machine Learning.

3.2. The Fourier transform & phase estimation algorithms

In the previous chapter, we explored applications related to quantum search, starting with Grover's algorithm. In this one, we want to understand how we can deal with linear algebra problems, which are present in many other computer science applications. To do so, we first have to explain the *quantum Fourier transform*, which will enable the implementation of the *quantum phase estimation* algorithm mentioned in the previous chapter. Additionally, phase estimation will be the key subroutine in Shor's algorithm, arguably the most famous quantum algorithm together with Grover's.

As a motivation for the latter, let us introduce a related one. Let $\oplus$ denote bitwise binary addition such that for example $011 \oplus 101 = 110$.

Problem 1 (Simon) *Let $f : 2^n \mapsto 2^n$ such that $f(i) = f(j)$ if and only if $i \oplus s = j$ for some secret s. Find s.*

To solve this problem, Simon proposed the following quantum algorithm [221]:

Algorithm 7 Simon's algorithm

1: **Input**: Oracle $f : |i\rangle |0\rangle \mapsto |i\rangle |f(i)\rangle$ fulfilling the promise of Problem 1.
2: **Output**: String s solving the corresponding Simon Problem 1.
3: Use Hadamard gates to create a uniform superposition,

$$\frac{1}{\sqrt{2^n}} \sum_i |i\rangle |0\rangle . \tag{3.1}$$

4: Query the oracle f.
5: (Optionally) measure the second qubit register on the computational basis, obtaining

$$\frac{1}{2}(|i\rangle + |i \oplus s\rangle) |f(i)\rangle . \tag{3.2}$$

6: Apply a second Hadamard to the first register, obtaining

$$\frac{1}{\sqrt{2^n}} \sum_j \left((-1)^{i \cdot j} |j\rangle + (-1)^{(i \oplus s) \cdot j} |j\rangle\right) = \frac{1}{\sqrt{2^n}} \left(\sum_j (-1)^{i \cdot j} (1 + (-1)^{s \cdot j}) |j\rangle\right) . \tag{3.3}$$

7: Measure j. Each $|j\rangle$ will have non-zero amplitude only if $s \cdot j = 0$.
8: Repeat until one has obtained sufficient (linearly independent) samples from j, and solve the associated linear system of equations.

Once we have obtained k linearly independent samples, the probability of subsequently measuring another independent sample is $(2^{n-1} - 2^k)/2^{n-1} \geq 1/2$. Since the

linear system of equations can be solved in time $O(n^3)$, the overall complexity is $O(n^3)$. In contrast, Simon proved that any classical algorithm requires $\Omega(\sqrt{2^n})$ calls to the oracle to find collisions with high probability [221].

Shor's algorithm solves a similar periodicity-related problem.

Problem 2 (Period finding) *Let* $f : \mathbb{N} \mapsto \{0, ..., N-1\}$ *such that* $f(i) = f(j)$ *if and only if* $i = j \mod r$. *Find* r.

There is an important difference, though. In Simon's problem, the addition was bitwise, while in this case, the periodicity is a global property. It is for this reason that Hadamard gates will not be enough. Instead, we will introduce the quantum Fourier transform, which is related to the Hadamard transform by

$$H = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = F_2 \tag{3.4}$$

The Fourier transform is a unitary transformation defined by the matrix

$$F_N = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega_N & \omega_N^2 & \omega_N^3 & \dots & \omega_N^{N-1} \\ 1 & \omega_N^2 & \omega_N^4 & \omega_N^6 & \dots & \omega_N^{2(N-1)} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega_N^{N-1} & \omega_N^{2(N-1)} & \omega_N^{3(N-1)} & \dots & \omega_N^{(N-1)^2} \end{pmatrix} = \frac{1}{\sqrt{N}} \left(e^{\frac{2\pi i}{N}(j \cdot k)} \right)_{jk}, \tag{3.5}$$

where $\omega_N = e^{2\pi i/N}$.

Applying one such matrix classically naively takes $O(N^2)$ time, but can be reduced to $O(N \log N)$ by noting that vector coordinates are transformed as

$$\begin{aligned} \bar{\boldsymbol{v}}_j = \sum_k \omega_N^{j \cdot k} \boldsymbol{v}_k &= \frac{1}{\sqrt{N}} \left(\sum_{\text{even } k} \omega_N^{jk} \boldsymbol{v}_k + \omega_N \sum_{\text{odd } k} \omega_N^{j(k-1)} \boldsymbol{v}_k \right) \\ &= \frac{1}{\sqrt{2}} \left(\frac{1}{\sqrt{N/2}} \sum_{\text{even } k} \omega_{N/2}^{jk/2} \boldsymbol{v}_k + \omega_N^j \sum_{\text{odd } k} \omega_{N/2}^{j(k-1)/2} \boldsymbol{v}_k \right), \end{aligned} \tag{3.6}$$

where the bar indicates the Fourier transform. The key takeaway is that we can compute the n-Fourier transform with 2 $(n-1)$-Fourier transforms. Moreover, we can decompose the $(n-1)$-Fourier transforms, initiating a recursion process, which spans a tree of depth $O(\log N)$ and $N/2$ terms at the deepest level. This results in complexity $O(N \log N)$, and is commonly known as the *fast Fourier transform* (FFT) [77].

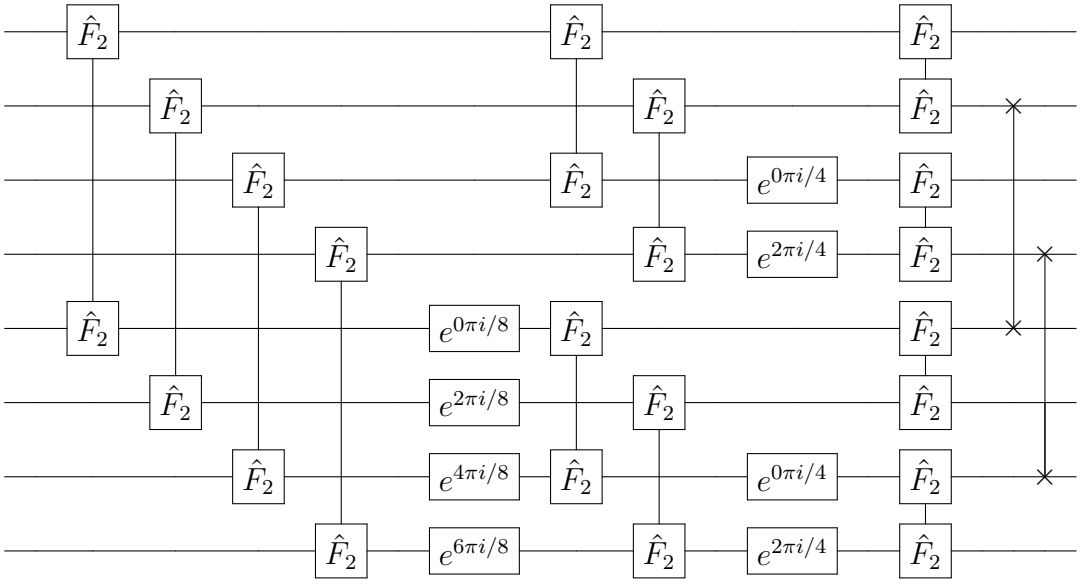


Figure 3.1 **Quantum Fermionic Fast Fourier Transform for 8 modes.** The $\hat{F}_2$ gate is defined in (3.7), while the single-qubit gates are Z-like rotations. This circuit is the same as depicted in Fig. 1 [94].

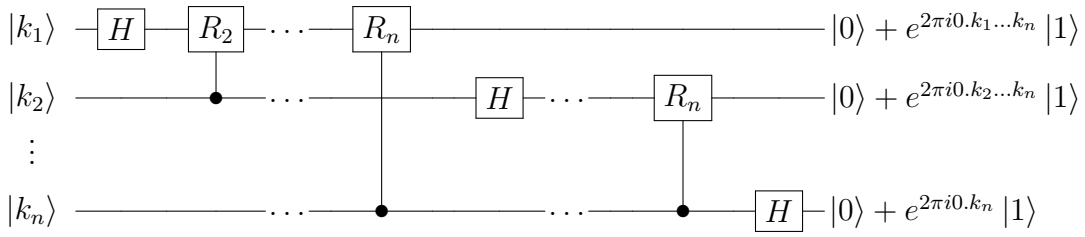


Figure 3.2 **Quantum Fourier transform.** Notice that the qubit representation of the quantum Fourier transform comes out reversed, and can be reordered using swaps if desired. Each R_l gate is defined by (3.9). The QFT circuit requires $O(n^2)$ gates, for n qubits.

In fact, this technique has also been used in quantum computing, especially for fermions [94], see Fig. 3.1. In this case, the fermionic operator $\hat{F}_2$ is implemented over two qubits

$$\hat{F}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 2^{-1/2} & 2^{-1/2} & 0 \\ 0 & 2^{-1/2} & -2^{-1/2} & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}, \tag{3.7}$$

where the last -1 is due to the anticommutation relation of fermions [94]. This matrix operator can be decomposed into two C-NOT gates sandwiching a controlled Hadamard gate in the opposite direction, followed by a C-Z to implement the final phase.

However, since the Fourier transform is unitary, there is also a direct implementation over $n = \log N$ qubits. To understand how to generate it, we rewrite [83]

$$\begin{aligned} F_N |k\rangle &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2\pi i}{N} jk} |j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{\frac{2\pi i}{N} (\sum_l j_l 2^{-l}) k} |j_1\rangle \dots |j_n\rangle \\ &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \left(\prod_{l=1}^{n} e^{\frac{2\pi i}{N} (j_l 2^{-l}) k} \right) |j_1\rangle \dots |j_n\rangle = \bigotimes_{l=1}^{n} \frac{1}{\sqrt{2}} \left(|0\rangle + e^{2\pi i k/2^l} |1\rangle \right). \end{aligned} \tag{3.8}$$

We note that the exponent in $e^{2\pi i k/2^l}$ would have equivalent effect as the fractional part of its binary expression, $e^{2\pi i 0.k_{n-l+1} \dots k_n}$ [83]. This suggests the quantum Fourier circuit depicted in Fig. 3.2, with rotation R_l defined by

$$R_l = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^l} \end{pmatrix}. \tag{3.9}$$

The (inverse) quantum Fourier transform is an important step in what might be considered one of the most useful quantum subroutines: the quantum phase estimation algorithm [141]. Imagine we have a unitary operator U with eigenstates $|\psi_i\rangle$ and eigenvalues $\lambda_i = e^{2\pi i \varphi_i}$. Quantum phase estimation has two main steps. First, we implement controlled versions of the phase kickback using operator U, for each eigenstate $|\psi\rangle$ and corresponding eigenvalue φ, synthesizing state

$$\begin{aligned} &\frac{1}{2^{t/2}} \left(|0\rangle + e^{2\pi i 2^{t-1} \varphi} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i 2^{t-2} \varphi} |1\rangle \right) \otimes \dots \otimes \left(|0\rangle + e^{2\pi i 2^0 \varphi} |1\rangle \right) \otimes |\psi\rangle = \\ &\frac{1}{2^{t/2}} \left(|0\rangle + e^{2\pi i 0.\varphi_t} |1\rangle \right) \otimes \left(|0\rangle + e^{2\pi i 0.\varphi_{t-1} \varphi_t} |1\rangle \right) \otimes \dots \otimes \left(|0\rangle + e^{2\pi i 0.\varphi_1 \dots \varphi_t} |1\rangle \right) \otimes |\psi\rangle . \end{aligned} \tag{3.10}$$

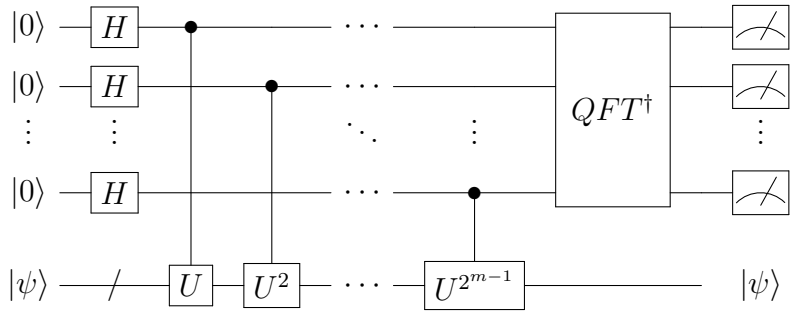


Figure 3.3 **Quantum phase estimation algorithm**. The first part of the circuit implements a phase-kickback, whose phase is then estimated using an inverse quantum Fourier transform [193]. Note that obtaining t bits of accuracy with a probability of failure smaller than p_f requires $2^m - 1$ calls to U for $m = t + \left\lceil \log_2 \left(\frac{1}{2} + \frac{1}{2p_f} \right) \right\rceil$ [75], so the complexity of the algorithm grows as $O(\epsilon^{-1}) = O(2^t)$.

Then, we identify this state with the output of (3.8). As such, an inverse QFT can recover the binary expression of the phase implemented by U. The corresponding circuit can be seen in Fig. 3.3. Interestingly, the inverse quantum Fourier transform is not necessary. We already know that a single-qubit Hadamard gate is the same as a single-qubit QFT. We can use this fact to iteratively refine an estimate of a single qubit quantum phase estimation [141, 256].

As a very simple use case, we can use phase estimation together with amplitude amplification to obtain the *amplitude estimation* algorithm, by phase estimating the eigenvalues from (2.15), [51]. Now let us turn to the Problem 2. The following Algorithm 8 solves it [193, Sec. 5.4.1] using quantum phase estimation, and can be employed for Shor's original purpose of finding prime factors.

3.2.1. The hidden subgroup problem

This problem that we just explained is an instance of a more general one called the hidden subgroup problem (HSP). To explain it, we need some basic notions of group theory.

Definition 4 (Group) *Let G be a set of elements with an operation $\times : G \times G \mapsto G$ such that*

- $g_1 \times (g_2 \times g_3) = (g_1 \times g_2) \times g_3$ *for any* $g_1, g_2, g_3 \in G$.
- *There is an identity term* $1 \in G$ *such that* $1 \times g = g = g \times 1$ *for any* $g \in G$.
- *For any* $g \in G$ *there exists* $g^{-1} \in G$ *such that* $g \times g^{-1} = 1 = g^{-1} \times g$.

Algorithm 8 Period finding algorithm

1: **Input**: Oracle $f : |j\rangle |0\rangle \mapsto |j\rangle |f(j)\rangle$ fulfilling the promise of Problem 2.
2: **Output**: Value r solving Problem 2.
3: Use Hadamard gates to create a uniform superposition

$$\frac{1}{\sqrt{2^n}} \sum_{j=0}^{n} |j\rangle |0\rangle . \tag{3.11}$$

4: Query the oracle f, obtaining

$$\frac{1}{\sqrt{2^n}} \sum_{j} |j\rangle |f(j)\rangle \approx \frac{1}{\sqrt{2^n}} \frac{1}{\sqrt{r}} \sum_{j=0}^{n} \sum_{\ell=0}^{r} e^{2\pi i j \ell / r} |j\rangle |\hat{f}(\ell)\rangle , \tag{3.12}$$

where r is the period, because one defines

$$|\hat{f}(\ell)\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{-2\pi i \ell / r} |f(j)\rangle \Rightarrow |f(j)\rangle = \frac{1}{\sqrt{r}} \sum_{j=0}^{r-1} e^{2\pi i \ell / r} |\hat{f}(\ell)\rangle . \tag{3.13}$$

The approximate equality is because 2^n is not necessarily a multiple of r.
5: Apply an inverse quantum Fourier transform, obtaining a state close to

$$\frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} |\widetilde{\ell/r}\rangle |\hat{f}(\ell)\rangle . \tag{3.14}$$

6: Measure the first register, obtaining with high probability an approximation to ℓ_1/r for a random $\ell_1 \in \{0, \ldots, r-1\}$.
7: Repeat the procedure above, and obtain another approximation of the phase, for a randomly chosen $\ell_2 \in \{0, \ldots, r-1\}$.
8: Obtain fractions ℓ_1'/r_1 and ℓ_2'/r_2 that approximate the phases measured in the previous steps, using the continued fraction algorithm (see [193, Box 5.3] and [83, Section 5.4]).
9: If ℓ_1' and ℓ_2' contain no common factors (what happens with probability at least 1/4 [193, Eq. 5.58]), then r is the smallest common multiple of r_1 and r_2.

A group is called Abelian if $g \times h = h \times g$, and a set $H \subseteq G$ is a subgroup if it is itself a group. A set of elements T, denoted as $\langle T \rangle$, is called generating set of G if using elements in T and $\times$ we can generate any $g \in G$. Finally, given a subgroup H, a (left) *coset* of H is a set $gH := \{gh | h \in H\}$ for $g \in G$. Cosets of H are either identical or disjoint [83]. Using these definitions, we can pose the following general problem.

Problem 3 (Hidden subgroup problem) *Let G be a group and $f : G \mapsto S$ a function, where S is a finite set. Let f have the property that there exists a hidden*

subgroup H such that f is constant on its cosets, e.g., $f(g) = f(g') \Leftrightarrow gH = g'H$. The goal is to find H or its generators.

Simon's and period finding problems described above are particular instances of the hidden subgroup problem. For Simon's problem, the hidden subgroup is $\{0, s\}$ for $s \in \{0,1\}^n$. In the period finding problem, on the other hand, we have $H = \{0, r, 2r, \ldots, |\mathrm{Im}f| - r\} = \langle 0, r\rangle$, for the oracle function f.

The hidden subgroup problem (HSP) is very common in cryptography. For example, the RSA cryptographic scheme can be attacked with Shor's algorithm, which finds the decomposition in primes of large composite numbers. More generally, there is a quantum algorithm that allows to solve it whenever the group is Abelian. To explain it, we first introduce some additional definitions.

Definition 5 (Representation) *A representation ρ of a group G is a map $\rho : G \mapsto GL_{\mathbb{C}}(V)$ to the general linear group of complex invertible matrices (automorphisms), such that $\rho(g_1) \cdot \rho(g_2) = \rho(g_1 \times g_2)$ for any g_1, g_2 in G. Further, a representation is called irreducible if it has no trivially invariant subspaces (cannot be decomposed in the direct sum of lower-dimensional representations), where the dimension of the representation is the dimension of the vector space V.*

Every column of the Fourier transform is a map $\chi : \mathbb{Z}_N \mapsto \mathbb{C}$ defined as $\chi_k(j) = e^{\frac{2\pi i}{N}jk} = \omega_N^{jk}$. Since $\chi_k(j + j') = \chi_k(j)\chi_k(j')$, χ_k is a 1-dimensional representation (also called *character*) of $\mathbb{Z}_N$. The characters also form a N-dimensional basis

$$|\chi_k\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \chi_k(j) \, |j\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \omega_N^{jk} \, |j\rangle \,, \tag{3.15}$$

and the Fourier transform maps between this and the computational basis: $F : |k\rangle \mapsto |\chi_k\rangle$.

An important feature of Abelian groups is that they are isomorphic to a direct product $\mathbb{Z}_{N_1} \times \ldots \times \mathbb{Z}_{N_l}$. In this product of groups, the Fourier transform is defined as the tensor product of Fourier transforms for each of the $\mathbb{Z}_N$.

Finally, to explain a general Abelian hidden subgroup problem we define the *dual subgroup* $\hat{G}$ whose elements are the characters χ_k, together with pointwise multiplication. Then, for any subgroup $H \subseteq G$, let

$$H^{\perp} = \{\chi_k | \chi_k(h) = 1, \quad \forall h \in H\}, \tag{3.16}$$

of dimension $|G|/|H|$. These definitions are sufficient to propose a general Abelian hidden subgroup problem algorithm [141, 187], which we describe in Algorithm 9.

Algorithm 9 Abelian hidden subgroup problem algorithm

1: **Input**: Oracle $f : G \to S$ for some set S fulfilling the promise of Problem 3, that is, f is constant on cosets of hidden subgroup H.
2: **Output**: Generators of hidden subgroup H.
3: **for** $O(\log |G|)$ steps **do**
4: Initialize $|0\rangle^{\otimes \log |G|} |0\rangle^{\otimes \log |S|}$.
5: Create a superposition over G:

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |0\rangle . \tag{3.17}$$

6: Compute f:

$$\frac{1}{\sqrt{|G|}} \sum_{g \in G} |g\rangle |f(g)\rangle . \tag{3.18}$$

7: Measure the second register obtaining $f(s)$ for some unknown $s \in G$:

$$\frac{1}{\sqrt{|H|}} \sum_{h \in H} |s + h\rangle . \tag{3.19}$$

8: Apply the quantum Fourier transform of $G \approx \mathbb{Z}_{N_1} \times \ldots \times \mathbb{Z}_{N_l}$ to the state obtaining

$$\begin{aligned} \frac{1}{\sqrt{|H|}} \sum_{h \in H} |\chi_{s+h}\rangle &= \frac{1}{\sqrt{|G||H|}} \sum_{h \in H} \sum_{g \in G} \chi_{s+h}(g) |g\rangle \\ &= \frac{1}{\sqrt{|G||H|}} \sum_{g \in G} \chi_s(g) \sum_{h \in H} \chi_h(g) |g\rangle = \sqrt{\frac{|H|}{|G|}} \sum_{g : \chi_g \in H^\perp} \chi_s(g) |g\rangle , \end{aligned} \tag{3.20}$$

where the last equal holds because

$$\sum_{h \in H} \chi_h(g) = \sum_{h \in H} \chi_g(h) = \begin{cases} |H| & \text{if } \chi_g \in H^\perp, \\ 0 & \text{if } \chi_g \notin H^\perp. \end{cases} \tag{3.21}$$

9: Measure the first register. Since all $|\chi_g(s)|^2 = 1$, obtain one $|g\rangle : \chi_g \in H^\perp$ uniformly at random.
10: **end for**
11: Solve the constrain satisfaction problem $\chi_g(h) = 1 \; \forall h \in H$ to find generators of H.

On the other hand, we can also define a non-Abelian quantum Fourier transform [83]:

$$F : |g\rangle \mapsto \sum_{\rho \in \hat{G}} \sqrt{\frac{\dim G}{|G|}} |\rho\rangle \sum_{i,j} \rho(g)_{i,j} |i,j\rangle . \tag{3.22}$$

This quantum Fourier transform may still sometimes be computed efficiently, for example for the symmetric group. However, it does not necessarily lead to an efficient quantum algorithm because there is not a single measurement basis that provides enough information to reconstruct H [185]. Furthermore, while it is possible to solve the non-Abelian problem with only logarithmically many *positive operator-valued measurements* (POVMs) [83], constructing those measurements is not efficient in general [92].

In specific cases, it is sometimes possible to solve the hidden subgroup problem even if the group is not Abelian. Some examples are the normal [116], nil-2 [124], and solvable [251] groups. In contrast, other problems such as the graph isomorphism problem still lack an efficient solution, and the most efficient available algorithm is Babai's classical quasi-polynomial algorithm, which takes time $O(\exp(\log^{O(1)} |G|))$ [15].

3.3. Linear Algebra

In the previous section, we introduced the quantum Fourier transform and phase estimation algorithms, and we have shown that there are important problems that can be solved exponentially faster if we use them. Perhaps the next question is whether there are problems with less structure where we can also find this kind of exponential advantage. As indicated in the previous chapter, exponential advantages are much more useful than quadratic ones because they ensure that we can overcome the hardware slowdown derived from the need to perform error correction [19].

3.3.1. Solving linear systems of equations

One important problem that can be solved using quantum phase estimation is the following:

Problem 4 (Quantum linear system of equations) *Let $|b\rangle \in \mathbb{C}^n$ be a quantum state, and $A \in \mathbb{C}^{n \times n}$. Prepare the quantum state $|x\rangle = A^{-1} |b\rangle$.*

Since solving linear systems of equations are such a common subroutine in many algorithms, certainly solving them quickly opens the door to large efficiency improvements

in many algorithms. As a result, the so-called HHL algorithm [117], named after their discoverers, was met with a lot of excitement in the community. Algorithm 10 assumes A is a Hermitian matrix, but it is easy to convert an arbitrary problem $Ax = b$ to Hermitian writing instead

$$\begin{pmatrix} 0 & A \\ A^\dagger & 0 \end{pmatrix} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{x} \end{pmatrix} = \begin{pmatrix} \boldsymbol{b} \\ \mathbf{0} \end{pmatrix}. \tag{3.23}$$

If the eigenvalues of A are $\{\lambda_j\}_j$, the new eigenvalues would be $\{\pm\lambda_j\}_j$. This procedure also works for non-square matrices A.

Algorithm 10 Quantum linear system of equations algorithm (HHL) [117].

1: **Input**: State $|b\rangle \in \mathbb{C}^n$, sparse-access oracle access to the entries of the (Hermitian) matrix $A \in \mathbb{C}^{n\times n}$.
2: **Output**: State $|x\rangle = A^{-1}|b\rangle \in \mathbb{C}^n$.
3: Initialize state $|b\rangle|0\rangle|0\rangle = \sum_j \beta_j |u_j\rangle|0\rangle|0\rangle$, the formal decomposition of $|b\rangle$ as a superposition of eigenvectors $|u_j\rangle$ of A.
4: Implement phase estimation on $U = e^{-iAt} = \sum_j e^{-i\lambda_j t}|u_j\rangle\langle u_j|$, obtaining state $\sum_j \beta_j |u_j\rangle|\tilde{\lambda}_j\rangle|0\rangle$, where $|\tilde{\lambda}_j\rangle$ is an approximation to the binary expression of eigenvalue $|\lambda_j\rangle$.
5: Apply the rotation

$$\sum_j \beta_j |u_j\rangle|\tilde{\lambda}_j\rangle|0\rangle \mapsto \sum_j \beta_j |u_j\rangle|\tilde{\lambda}_j\rangle \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j}}|0\rangle + \frac{C}{\tilde{\lambda}_j}|1\rangle\right) \tag{3.24}$$

for some $C \geq \lambda_j$. Thus $C = O(1/\kappa)$, for κ the condition number.
6: Uncompute the phase estimation to erase the second register.
7: Amplitude-amplify the component of the state with the third register at $|1\rangle$, and obtain

$$|x\rangle \propto \sum_j \frac{\beta_j}{\tilde{\lambda}_j}|u_j\rangle \approx A^{-1}|b\rangle. \tag{3.25}$$

The rotation (3.24) can be applied efficiently because we can perform a series of rotations controlled by the binary expression of $|\tilde{\lambda}_j\rangle$. To analyze the complexity of Algorithm 10, note that the cost comes from two main sources: the phase estimation subroutine, and the required amplitude amplification. If we assume that A has no more than s non-zero entries per row and column, then the Hamiltonian simulation of e^{-iAt}, which will be discussed later on in more detail, might be implemented at cost $\tilde{O}(ts^2)$ [33]. Since we want to estimate each eigenvalue to precision ϵ and each eigenvalue could be as small as $1/\kappa$, the overall cost of the quantum phase estimation

step is $\tilde{O}(ts^2) = \tilde{O}(s^2\kappa\epsilon^{-1})$. Remember that the condition number κ is defined as

$$\kappa = \left| \frac{\max_i \sigma_i}{\min_i \sigma_i} \right|, \tag{3.26}$$

where σ_i are the singular values which, for square matrices, match the absolute value of eigenvalues $|\lambda_i|$.

Second, for amplitude amplification, the largest complexity happens when almost all eigenvalues are large, so that coefficients become $\frac{1}{\kappa\lambda_i} \approx \frac{1}{\kappa}$. Consequently, the overall algorithm has complexity $\tilde{O}(s^2\kappa^2\epsilon^{-1})$. As we can see, if the size of the matrix is N, the algorithm depends only polylogarithmically on that factor.

We have seen how the basic quantum linear system algorithm can be implemented. In contrast, if one wishes to solve this problem classically, the fastest algorithm is the conjugate gradient method [219], which has complexity $O(Ns\sqrt{\kappa}\log\epsilon^{-1})$ for positive definite matrices A and $O(Ns\kappa\log\epsilon^{-1})$ otherwise. The contrast between these complexities is an exponential speedup in N. However, notice also that the classical algorithm provides a classical readout of the entire solution, which is not possible from a single copy of the quantum state $|x\rangle$. Furthermore, the complexity of parameters other than N is worse. Fortunately, there are situations where these limitations do not represent a problem and a useful application of the quantum linear system algorithm might be carried out, see for example Ref. [74].

3.3.2. Improving the QLSA performance

Condition number. Let us now see how to improve the complexity of those other parameters, starting with the dependence on κ. In the previous discussion, we highlighted how one $O(\kappa)$ contribution comes from small λ_i eigenvalues, which forces us to use a high precision in the phase estimation protocol. The second $O(\kappa)$ contribution, in contrast, happens when eigenvalues are large, decreasing the acceptance probability. This suggests performing amplitude amplification more times over those states or branches of the algorithm that display less expensive amplitude amplification. While perhaps too technical to explain here, the *variable time amplitude amplification* algorithm of Ambainis [6] does precisely this. It uses a version of Amplitude Estimation [51] and the Median Lemma [192, Lemma 1] to estimate eigenvalues from less to more precision in consecutive steps. At any step, if the eigenvalue is sufficiently large, the success part of the step after the rotation is amplified. In such a way, the success part of larger eigenvalues is amplified for more steps, but the initial amplifications are less costly because phase estimation is implemented with smaller precision.

Note though, that reducing the complexity of the condition number κ to linear might not be sufficient to justify using Algorithm 10 because we need κ to be polylogarithmic in N if we want to maintain the exponential speedup. For example, most linear systems that appear when trying to solve the finite element methods often display $\kappa = O(\text{poly}N)$ [55, 24]. For that reason, it is worth considering preconditioning as a way to reduce κ. Preconditioning are methods whose objective is to find a matrix M such that the system $MAx = Mb$ has a better condition number than the original system of equations, at a relatively low cost. The best possible preconditioner would be $M = A^{-1}$, but this is equivalent to solving the system.

Ref. [74] suggests a method to perform preconditioning on the fly on the quantum computer. This requires two main features from matrices A and M: a) that only local information of A is known, and b) that the resulting matrix MA is equally sparse as A. To meet these two criteria, they use a quantum version of Sparse Preconditioners with Approximate Inverses (SPAI) [110, 73]. These algorithms minimize [74, Eq. 10]

$$\|MA - \mathbf{1}\|_F^2 = \sum_{k=0}^{N} \|(MA - \mathbf{1})e_k\|_F^2, \tag{3.27}$$

where $e_k = (0, \ldots, 0, 1, 0 \ldots, 0)^T$, e.g., the k^{th} column of the identity matrix; and $\|A\|_F := \sqrt{\sum_{i,j} A_{ij}^2}$ is the Frobenius norm. To carry out this task, we choose a sparsity pattern of M, often taken to be the same as that of A. This results in N $s \times s$ small systems of equations, each of which can be solved in superposition in time $O(s^3)$. Similarly, one also has to multiply $M|b\rangle$, but in the end one can successfully reduce κ with only a multiplicative overhead polynomial in $s = O(\text{poly}\log N)$. Ref. [74] also shows that this technique can be used to calculate the electromagnetic scattering cross-section of an arbitrary target using the finite element methods [126] with exponential speedup over known classical methods.

Sparsity. A second important generalization is to go beyond the sparse-access oracle system and allow for dense matrices. To do so, we will assume that using quantum random access memory (qRAM) [102] we can implement the following unitaries with precision ϵ^{-1} in time $O(\text{poly}\log(N/\epsilon))$ for $A \in \mathbb{R}^{N\times N}$ [132]

$$U_{\mathcal{M}} : |i\rangle|0\rangle \to \frac{1}{||A_{i\cdot}||}\sum_j A_{ij}|ij\rangle, \qquad U_{\mathcal{N}} : |0\rangle|j\rangle \to \frac{1}{||A||_F}\sum_i ||A_{i\cdot}||\,|ij\rangle, \tag{3.28}$$

where $||A_{i\cdot}||$ stands for the ℓ_2-norm of the i^{th} row of A. This data access is sometimes called the *quantum accessible data structure*. Note that

$$\langle i,0|U_{\mathcal{M}}^{\dagger}U_{\mathcal{N}}|j,0\rangle = \frac{||A_{i\cdot}||}{||A||_F}\frac{A_{i,j}}{||A_{i\cdot}||} = \frac{A_{i,j}}{||A||_F}. \tag{3.29}$$

Using these operators implemented via qRAMs, we can efficiently perform one step of the quantum walk [132, Theorem 5.1]

$$W = (2U_{\mathcal{M}}U_{\mathcal{M}}^{\dagger} - \mathbf{1})(2U_{\mathcal{N}}U_{\mathcal{N}}^{\dagger} - \mathbf{1}) = R_{\mathcal{M}}R_{\mathcal{N}}. \tag{3.30}$$

We analyze this quantum walk as we did for Grover: the rotation implemented by W will be twice the angle $\theta_i/2$ between $U_{\mathcal{M}}|u_i\rangle$ and $U_{\mathcal{N}}|v_i\rangle$, for the singular value decomposition $A = \sum_i \sigma_i |u_i\rangle\langle v_i|$. Indeed, θ_i depends on σ_i as $\cos(\theta_i/2) = \frac{\sigma_i}{\|A\|_F}$ [132, Lemma 5.3]. This matches the relation between the rotation angles φ_i and the eigenvalues λ_i that occurred in quantum walks, except for a factor of 2 in the definition of θ_i, to respect the original paper convention.

Algorithm 11 Quantum singular value estimation [132].

1: **Input**: Matrix A, operators $U_{\mathcal{M}}$ and $U_{\mathcal{N}}$, initial state $|x\rangle = \sum_i x_i|v_i\rangle$, precision ϵ.
2: **Output**: State $\sum_i x_i|v_i\rangle|\bar{\sigma}_i\rangle$, for $\bar{\sigma}_i \approx \sigma_i$.
3: Initialize $|x\rangle = \sum_i x_i|v_i\rangle$.
4: Apply $U_{\mathcal{N}}$ to the state $|x\rangle$, obtaining $U_{\mathcal{N}}|x\rangle = \sum_i x_i U_{\mathcal{N}}|v_i\rangle$. $U_{\mathcal{N}}|v_i\rangle$ are eigenvectors of W with eigenvalues $e^{i\theta_i}$, [132, Lemma 5.3].
5: Perform phase estimation of W with precision ϵ on state $U_{\mathcal{N}}|x\rangle$, obtaining $\sum_i x_i(U_{\mathcal{N}}|v_i\rangle)|\bar{\theta}_i\rangle$ for $|\bar{\theta}_i - \theta_i| \leq \epsilon$.
6: Compute $\bar{\sigma}_i = \cos\bar{\theta}_i$ in binary reversible arithmetic, so the state is $\sum_i x_i(U_{\mathcal{N}}|v_i\rangle)|\bar{\sigma}_i\rangle$.
7: Uncompute $U_{\mathcal{N}}$ on the first register, outputting $\sum_i x_i|v_i\rangle|\bar{\sigma}_i\rangle$.

Using Algorithm 11, we can estimate the singular values of A, which are related to the eigenvalues through $|\lambda_i| = \sigma_i$. To ascertain the sign, we can implement the same algorithm with $A + \epsilon\mathbf{1}$ and compare the results. Finally, we can use this technique to substitute the quantum phase estimation algorithm in Algorithm 10, to solve the linear system of equations with a dense matrix A [262]. The complexity, instead of linear in the sparsity parameter s, will now depend on the Frobenius norm of the matrix $\|A\|_F$.

Precision. Finally, the last parameter that can be improved is the precision complexity, which in the original quantum linear system of equations was $O(\epsilon^{-1})$. Notice that the complexity of $O(\epsilon^{-1})$ is due to the phase estimation, so to improve the complexity

of the overall algorithm we need to bypass its use. For this, we will use the Linear Combination of Unitaries (LCU) approach that we have already mentioned previously, with operators Prep and Sel defined as in (2.56). Imagine that we want to implement operator $U = \sum_i \alpha_i U_i$. Then, using

$$\text{Prep}\,|\mathbf{0}\rangle \mapsto \frac{1}{\sqrt{\sum_i \alpha_i}} \sum_i \sqrt{\alpha_i}\,|i\rangle\,; \qquad \text{Sel} = \sum_i |i\rangle\langle i| \otimes U_i \tag{3.31}$$

we can define $W = \text{Prep}^\dagger \cdot \text{Sel} \cdot \text{Prep}$, which maps

$$W\,|\mathbf{0}\rangle\,|\psi\rangle = \frac{1}{\sum_i \alpha_i}\,|\mathbf{0}\rangle \sum_i \alpha_i U_i\,|\psi\rangle + |(\mathbf{0}\psi)^\perp\rangle \tag{3.32}$$

Then, one can use (variable time) amplitude amplification to amplify the part of the state that we are interested in.

Ref. [71] proposes two possible LCU compositions that can be implemented with gate complexity $O(\log \epsilon^{-1})$. The first is to approximate the application of matrix A^{-1} based on its corresponding Fourier series, where unitaries $U_j = e^{-iAt_j}$. This amounts to implementing the Hamiltonian simulation of A, as done for phase estimation. Since both A^{-1} and each of the e^{-iAt_j} operators will be diagonal on the same basis of eigenvalues, one may propose a series decomposition for scalar eigenvalues, which generalizes to the operator. The authors of Ref. [71] showed that

$$\frac{1}{x} = \frac{i}{\sqrt{2\pi}} \int_0^\infty dy \int_{-\infty}^\infty dz z e^{-z^2/2} e^{-ixyz} \tag{3.33}$$

can be approximated by the series [71, Lemma 11]

$$h(x) = \frac{i}{\sqrt{2\pi}} \sum_{j=0}^{J-1} \Delta_y \sum_{k=-K}^{K} \Delta_z z_k e^{-z_k^2/2} e^{-ixy_j z_k}, \tag{3.34}$$

for $z_k = k\Delta_z$, $y_j = j\Delta_y$, $J = \Theta(\frac{\kappa}{\epsilon}\log(\kappa/\epsilon))$, $K = \Theta(\kappa \log(\kappa/\epsilon))$, $\Delta_y = \Theta(\epsilon/\sqrt{\log(\kappa/\epsilon)})$, $\Delta_z = \Theta(1/\kappa\sqrt{\log(\kappa/\epsilon)})$; to precision ϵ^{-1} in the domain $D_\kappa = [-1, -1/\kappa] \cup [1/\kappa, 1]$. The key to avoid the $O(\epsilon)$ complexity from J is to use Hadamard gates to prepare a uniform superposition over $|j\rangle$. Then, the Sel operator will be defined as [71, Eq. 52]

$$\text{Sel} = i \sum_{j=0}^{J} \sum_{k=-K}^{K} |j,k\rangle\langle j,k| \otimes \text{sgn}(k) e^{-iAy_j z_k}, \tag{3.35}$$

which can be efficiently implemented based on the binary representation of j and k, avoiding the complexity on κ.

The second option is to implement a Chebyshev series based on quantum walks. We already saw a hint of this in (2.52). We first decompose A^{-1} as a Chebyshev series $\sum_i \alpha_i \mathcal{T}_i(A)$. Then, we explain how to perform the $\mathcal{T}_i(A)$ Chebyshev polynomial operators of the first kind using Szegedy-like quantum walks. Defining T as

$$T : |j\rangle |0\rangle \mapsto |j\rangle |\psi_j\rangle = |j\rangle \otimes \frac{1}{\sqrt{s}} \sum_{k\in[N]:A_{jk}\neq 0} \left(\sqrt{A^*_{jk}}\,|k\rangle + \sqrt{1-|A^*_{jk}|}\,|k+N\rangle\right) \tag{3.36}$$

for s the sparsity and S a swap between the two registers, one can take the quantum walk to be $W = S(2TT^\dagger - 1)$. Similarly to Szegedy quantum walks, for each eigenvalue λ of $H = A/s$, W has block-diagonal form in the invariant subspace $\text{span}\{T|\lambda\rangle, ST|\lambda\rangle\}$ [71, Lemma 15]

$$W = \bigoplus_\lambda \begin{pmatrix} \lambda & -\sqrt{1-\lambda^2} \\ \sqrt{1-\lambda^2} & \lambda \end{pmatrix}_\lambda . \tag{3.37}$$

Then, one can prove

$$W^n = \bigoplus_\lambda \begin{pmatrix} \mathcal{T}_n(\lambda) & -\sqrt{1-\lambda^2}\mathcal{U}_{n-1}(\lambda) \\ \sqrt{1-\lambda^2}\mathcal{U}_{n-1}(\lambda) & \mathcal{T}_n(\lambda) \end{pmatrix}_\lambda \tag{3.38}$$

by induction, where $\mathcal{U}_n$ represent Chebyshev polynomials of the second kind. The case $n = 1$ is clearly true because $\mathcal{T}_1(\lambda) = \lambda$ and $\mathcal{U}_0(\lambda) = 0$. Then

$$\begin{aligned} W^n W &= \bigoplus_\lambda \begin{pmatrix} \lambda\mathcal{T}_n(\lambda) - (1-\lambda^2)\mathcal{U}_{n-1}(\lambda) & -\sqrt{1-\lambda^2}(\mathcal{T}_n(\lambda) + \lambda\mathcal{U}_{n-1}(\lambda)) \\ \sqrt{1-\lambda^2}(\mathcal{T}_n(\lambda) + \lambda\mathcal{U}_{n-1}(\lambda)) & \lambda\mathcal{T}_n(\lambda) - (1-\lambda^2)\mathcal{U}_{n-1}(\lambda) \end{pmatrix}_\lambda \\ &= \bigoplus_\lambda \begin{pmatrix} \mathcal{T}_{n+1}(\lambda) & -\sqrt{1-\lambda^2}\mathcal{U}_n(\lambda) \\ \sqrt{1-\lambda^2}\mathcal{U}_n(\lambda) & \mathcal{T}_{n+1}(\lambda) \end{pmatrix}_\lambda \end{aligned} \tag{3.39}$$

because $\mathcal{T}_{n+1}(\lambda) = \lambda\mathcal{T}_n(\lambda) - (1-\lambda^2)\mathcal{U}_{n-1}(\lambda)$ and $\mathcal{U}_n(\lambda) = \mathcal{T}_n(\lambda) + \lambda\mathcal{U}_{n-1}(\lambda)$ [26]. Additionally, Lemma 14 in Ref. [71] shows that

$$g(\lambda) = 4\sum_{j=0}^{j_0}(-1)^j \left(\frac{\sum_{i=j+1}^{b}\binom{2b}{b+u}}{2^{2b}}\right)\mathcal{T}_{2j+1}(\lambda) \tag{3.40}$$

approximates λ^{-1} in the domain D_κ. As a result, we can use the LCU decomposition $\text{Prep}^\dagger \cdot \text{Sel} \cdot \text{Prep}$ to synthesize this function with cost polylogarithmic in precision ϵ^{-1}. This Chebyshev approach is marginally more efficient than the Fourier approach,

even if in both cases the complexity is polylogarithmic. Both techniques also allow for variable time-amplitude amplification to keep the complexity in the condition number linear. To do so, they only require a low precision application of quantum phase estimation. The discussion of variable time amplitude amplification applied to these two methods can be found in Section 5 in Ref. [71].

3.3.3. Qubitization

The obvious next step is to generalize from the preparation of A^{-1} to an arbitrary function $f[H] = \sum_\lambda f(\lambda) |\lambda\rangle \langle\lambda|$ for a Hamiltonian H. Since $f[H]$ does not need to be unitary, in practice this will mean finding a unitary operator U such that

$$U |G\rangle_a |\psi\rangle_s = |G\rangle_a f[H] |\psi\rangle_s + \sqrt{1 - \| f[H] |\psi\rangle \|^2} |G_\psi^\perp\rangle_{as}, \tag{3.41}$$

for some states $|G\rangle$ and $|G_\psi^\perp\rangle$ such that $\langle G_\psi^\perp|_{as} (|G\rangle_a \otimes \mathbf{1}_s) = 0$. Usually $|G\rangle$ is taken to be just $|G\rangle = |0\rangle$. Notice how U operates on the state register and an additional auxiliary one. This can be visually expressed as

$$U = \begin{pmatrix} f[H] & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{3.42}$$

which, for this reason, is called *block encoding*. The main problem with U, an example of which is a linear combination of units, is that it has some failure probability, as we already saw, and therefore requires amplitude amplification.

Instead, Ref. [169] proposes to find a different operator W, also encoding $H = \langle G|W|G\rangle$, but performing a $SU(2)$ rotation. In other words, we are looking for a quantum walk operator that can be expressed as

$$W = \bigoplus_\lambda \begin{pmatrix} \lambda & -\sqrt{1-\lambda^2} \\ \sqrt{1-\lambda^2} & \lambda \end{pmatrix}_\lambda = \bigoplus_\lambda e^{-Y_\lambda \theta_\lambda} \tag{3.43}$$

for $\theta_\lambda = \cos^{-1} \lambda$. To build this operator, they propose the construction

$$W = ((2 |G\rangle \langle G| - \mathbf{1})_a \otimes \mathbf{1}_s) SU, \tag{3.44}$$

for an operator S to be determined. They also observe that if $U^2 = \mathbf{1}$, as is the case for $U = \text{Prep}^\dagger \cdot \text{Sel} \cdot \text{Prep}$, then one can take $S = \mathbf{1}$, and automatically fulfill the conditions

required for W to implement (3.43) [169, Lemma 8]:

$$\langle G|_a SU |G\rangle_a = H; \qquad \langle G|_a SUSU |G\rangle_a = \mathbf{1}. \tag{3.45}$$

Else, one builds $U' = |0\rangle\langle 0| \otimes U + |1\rangle\langle 1| \otimes U^\dagger$ and $S' = (|1\rangle\langle 0| + |0\rangle\langle 1|) \otimes \mathbf{1}_{as}$, and taking $|G'\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|G\rangle$, one can verify

$$\begin{aligned} \langle G'|_a S'U' |G'\rangle_a = \langle G'|_a U' |G'\rangle_a = \frac{1}{2}(H + H^\dagger) = H; \\ \langle G'|_a S'U'S'U' |G'\rangle_a = \langle G'|_a (U')^\dagger U' |G'\rangle_a = \mathbf{1}. \end{aligned} \tag{3.46}$$

The procedure for generating and implementing this operator W from a block encoding U is called *qubitization*. Quantum walks, as we have seen, have the advantage of implementing Chebyshev polynomials given by (3.38). To implement arbitrary functions $f[H]$ we need to add additional degrees of freedom to rotation W. This can be done with the 2-dimensional operator

$$Z_\phi = ((1 + e^{-i\phi})|G\rangle\langle G| - \mathbf{1}) = \bigoplus_\lambda \begin{pmatrix} e^{-i\phi} & 0 \\ 0 & 1 \end{pmatrix}_\lambda, \tag{3.47}$$

and conjugating W by Z_ϕ,

$$\begin{aligned} W_\phi &= Z_{\phi-\pi/2} W Z_{-\phi+\pi/2} \\ &= \bigoplus_\lambda \begin{pmatrix} \lambda & -ie^{-i\phi}\sqrt{1-|\lambda|^2} \\ -ie^{+i\phi}\sqrt{1-|\lambda|^2} & \end{pmatrix}_\lambda = \bigoplus_\lambda e^{-i\theta_\lambda(\cos(\phi)X_\lambda + \sin(\phi)Y_\lambda)}. \end{aligned} \tag{3.48}$$

The qubitization circuit will be composed of N W_ϕ operators, for different ϕ angles $W_{\vec{\phi}} = W_{\phi_N} \dots W_{\phi_1}$. Using $W_{\vec{\phi}}$, we can implement any function $f[H] = A[H] + iC[H]$, where $A[H]$ and $C[H]$ are real polynomials of degree N (or $N/2$, depending on the specific technique used), of equal parity (respectively opposite parity). The determination of the corresponding angles is called *quantum signal processing* (see [168] and Theorems 3 and 4 in [169]), and has been used specifically for the Hamiltonian simulation of e^{-iHt}, which in turn can be used for phase estimation in Chemistry applications that we will study in Chapter 4. Crucially, the determination of these angles is computationally efficient and has been studied to have complexity scaling as $O(N^3 \text{poly}\log(N/\epsilon))$ [70, 114].

Figure 3.4 **Linear Programming problem, and different algorithms: simplex method (left) and interior point method (right).** The polyhedron represents the space of feasible solutions, e.g. those that fulfill the constraints. Such constraints are represented by planes, the faces of the polyhedra. The difference between simplex methods and interior point methods is how they transverse the space, from the border or the interior of the polyhedra. Images taken from Wikipedia for the simplex method and the interior point method, under CC-BY-SA 3.0 license.

3.4. Interior point methods

The first problem we addressed in this thesis was the study of Linear Programming problems using the quantum linear algebra tools described above [65]. Linear programming is a field where one has to optimize a linear function of multiple variables, subject to multiple constraints. For example, given $A \in \mathbb{R}^{m\times n}$, $\boldsymbol{c} \in \mathbb{R}^n$ and $\boldsymbol{b} \in \mathbb{R}^m$, the problem is to find $\boldsymbol{x} \in \mathbb{R}^n$ such that:

$$\text{minimizes } \boldsymbol{c}^T\boldsymbol{x} \tag{3.49a}$$

$$\text{subject to } A\boldsymbol{x} \geq \boldsymbol{b}, \qquad \boldsymbol{x} \geq \boldsymbol{0}. \tag{3.49b}$$

This is called the primal problem (LP). The dual problem (DP) is closely related: finding $\boldsymbol{y} \in \mathbb{R}^m$ such that

$$\text{maximizes } \boldsymbol{b}^T\boldsymbol{y} \tag{3.50a}$$

$$\text{subject to } A^T\boldsymbol{y} \leq \boldsymbol{c}. \tag{3.50b}$$

These two problems are connected by the primal-dual gap, which for linear programming problems is 0 [177]:

$$\boldsymbol{b}^T\boldsymbol{y} - \boldsymbol{c}^T\boldsymbol{x} = 0. \tag{3.51}$$

This kind of problem is abundant in operation research, yet simple enough to have been studied in detail. To solve these problems, two families of methods stand out in the literature. The first is the simplex method [79], proposed by George Dantzig in 1947, which is simple and very fast in practice and consequently widely used [191]. The

second important family of algorithms is the family of interior point methods, which in contrast to the simplex method are (weakly) polynomial in the worst case. The first algorithm of this kind was proposed in 1979 by Khachiyan, and had complexity $O(n^6 L)$ for $L = \sum_{i,j} \lceil \log(A_{ij} + 1) \rceil + 1$ [134]. The idea was subsequently improved by Karmarkar [128] making it easily implementable. A review of how this family of techniques can be found in Ref. [203].

For our quantum algorithm, we take one important and efficient classical interior point method by Ye et al., [263], and quantize some of the linear algebra that it uses. The method can be classified as a homogeneous self-dual algorithm because it solves both the dual and primal problems simultaneously by tackling a homogeneous problem whose solution can be mapped to those of the original primal and dual problems. Such a homogeneous problem (HLP) is

$$\min \theta \tag{3.52}$$

such that ($\boldsymbol{x} \geq \mathbf{0}, \tau \geq 0, \tau \in \mathbb{R}$):

$$\begin{array}{rrrcl} & +A\,\boldsymbol{x} & -\boldsymbol{b}\,\tau & +\bar{\boldsymbol{b}}\,\theta & = & \mathbf{0} \\ -A^T\,\boldsymbol{y} & & +\boldsymbol{c}\,\tau & -\bar{\boldsymbol{c}}\,\theta & \geq & \mathbf{0} \\ +\boldsymbol{b}^T\,\boldsymbol{y} & -\boldsymbol{c}^T\,\boldsymbol{x} & & +\bar{z}\,\theta & \geq & 0 \\ -\bar{\boldsymbol{b}}^T\,\boldsymbol{y} & +\bar{\boldsymbol{c}}^T\,\boldsymbol{x} & -\bar{z}\,\tau & & = & -(\boldsymbol{x}^0)^T\boldsymbol{s}^0 - 1 \end{array} \tag{3.53}$$

with

$$\begin{aligned} &\bar{\boldsymbol{b}} := \boldsymbol{b} - A\boldsymbol{x}^0, \qquad \bar{\boldsymbol{c}} := \boldsymbol{c} - A^T\boldsymbol{y}^0 - \boldsymbol{s}^0, \\ &\bar{z} := \boldsymbol{c}^T\boldsymbol{x}^0 + 1 - \boldsymbol{b}^T\boldsymbol{y}^0. \end{aligned} \tag{3.54}$$

It is called homogeneous because it has a single non-zero constraint. Ye's algorithm [263] will then follow the so-called central path through the inner space of the polyhedra corresponding to the self-dual (HLP) problem

$$\mathcal{C} = \left\{ (\boldsymbol{y}, \boldsymbol{x}, \tau, \theta, \boldsymbol{s}, k) \in \mathcal{F}_h^0 : \begin{pmatrix} X\boldsymbol{s} \\ \tau k \end{pmatrix} = \frac{\boldsymbol{x}^T\boldsymbol{s} + \tau k}{n+1}\mathbf{1} \right\}, \tag{3.55}$$

until it reaches the solution. The volume around the central path is called its neighborhood

$$\mathcal{N}(\beta) = \left\{ (\boldsymbol{y}, \boldsymbol{x}, \tau, \theta, \boldsymbol{s}, k) \in \mathcal{F}_h^0 : \left\| \begin{pmatrix} X\boldsymbol{s} \\ \tau k \end{pmatrix} - \mu\mathbf{1} \right\| \leq \beta\mu \text{ where } \mu = \frac{\boldsymbol{x}^T\boldsymbol{s} + \tau k}{n+1} \right\}. \tag{3.56}$$

Equations (3.53) and (3.56) will give rise to a dense linear system of equations that will ultimately need solving at each step of the algorithm,

$$\begin{array}{c} \\ m \\ n \\ 1 \\ 1 \\ n \\ 1 \end{array} \begin{array}{c} \begin{array}{cccccc} m & n & 1 & 1 & n & 1 \end{array} \\ \begin{pmatrix} 0 & A & -\boldsymbol{b} & \bar{\boldsymbol{b}} & 0 & 0 \\ -A^T & 0 & \boldsymbol{c} & -\bar{\boldsymbol{c}} & -1 & 0 \\ \boldsymbol{b}^T & -\boldsymbol{c}^T & 0 & \bar{z} & 0 & -1 \\ -\bar{\boldsymbol{b}}^T & \bar{\boldsymbol{c}}^T & -\bar{z} & 0 & 0 & 0 \\ 0 & S^t & 0 & 0 & X^t & 0 \\ 0 & 0 & k^t & 0 & 0 & \tau^t \end{pmatrix} \end{array} \begin{pmatrix} \boldsymbol{d_y} \\ \boldsymbol{d_x} \\ d_\tau \\ d_\theta \\ \boldsymbol{d_s} \\ d_k \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ 0 \\ 0 \\ \gamma^t\mu^t 1_{n\times 1} - X^t\boldsymbol{s}^t \\ \gamma^t\mu^t - \tau^t k^t \end{pmatrix}. \quad (3.57)$$

Since the system of equations does not need to be sparse, we rely on the dense linear system of equations algorithm of Ref. [262] via block encoding and assume access to the quantum-accessible data structure.

Ye's algorithm [263] is also a predictor-corrector method because it takes two kinds of steps: a first one to predict which direction should we modify the current state, and a corrector, which refines the solution.

1. *Predictor step:* Solve (3.57) with $\gamma^t = 0$ for $\boldsymbol{d}_{\boldsymbol{v}^t}$ where $\boldsymbol{v}^t = (\boldsymbol{y}^t, \boldsymbol{x}^t, \tau^t, \theta^t, \boldsymbol{s}^t, k^t) \in \mathcal{N}(1/4)$. Then find the biggest step length δ such that

$$\boldsymbol{v}^{t+1} = \boldsymbol{v}^t + \delta \boldsymbol{d}_{\boldsymbol{v}^t} \quad (3.58)$$

   is in $\mathcal{N}(1/2)$, and update the values accordingly. Then $t \leftarrow t + 1$.

2. *Corrector step:* Solve (3.57) with $\gamma^t = 1$ and set

$$\boldsymbol{v}^{t+1} = \boldsymbol{v}^t + \boldsymbol{d}_{\boldsymbol{v}^t}, \quad (3.59)$$

   which will be back in $\mathcal{N}(1/4)$. Update $t \leftarrow t + 1$.

The algorithm will continue until some stopping conditions are met, see section IIC in Ref. [65]. When that happens, the solution of the primal and dual problem are given $\boldsymbol{x}^*/\tau^*$ and $(\boldsymbol{y}^*/\tau^*, \boldsymbol{s}^*/\tau^*)$, respectively. Note that if the primal and dual solutions exist, then $\tau^* > 0$.

Most of the complexity of the algorithm is due to the necessity to read out the resulting quantum state. Not only because tomography is expensive, but because it is necessary to ensure that the result will not fall outside the neighborhood $\mathcal{N}(\beta)$ for specific β values. There are also limitations to our paper. First, in our article, we make

the explicit assumption of a target error for each component of the solution, rather than the vector ℓ_2-distance with the exact solution. While this is perhaps not usual, it also makes sense as otherwise ϵ will have a hidden dependence on the number of variables or constraints. Second, the system of equations will become increasingly bad conditioned, making κ very large. This large condition number may also affect the norm of the solution of each system of equations, and therefore the complexity of the precision required. Since the quantum tomography algorithm has quadratic scaling with precision, this would make our algorithm unsuitable for our purposes.

Overall, the κ dependence will likely make the performance of our algorithm worse than that of the equivalent classical algorithms. Finally, it is also worth highlighting that instead of using the exact-feasible approach of [263], and given the inexact nature of the solution of the system of equations recovered by tomography, a more appropriate approach might be to 'quantize' inexact interior point algorithms, as proposed in later work [14, 183].

3.5. Support Vector Machines

A second linear algebra problem we tackled in this thesis is related to machine learning [64]. We use quantum linear algebra techniques within an active learning framework, to make machine learning systems robust to adversarial examples. Adversarial examples are better understood in the context of classification systems. They are slightly modified versions of correctly classified examples that are, however, missidentified by the classifier [235]. Often, these changes are imperceptible to humans, yet they completely change the output of the machine learning model.

One of the reasons that have been argued for their existence is that modern machine learning models operate in very high dimensional spaces, while the actual dimension of the problem is much lower [135]. For example, an image classifier will encode each image in a giant $3 \times n \times n$-dimensional tensor, where n is the number of pixels in each direction, and 3 the colors needed to specify the red, green, and blue components of the pixel. Meanwhile, the objects these images depict live in a much lower effective dimension. The model has to partition the high-dimensional space with decision boundaries, but since the space volume is exponential in the dimension of the system, it becomes hard to make sure there is not a direction in which the decision changes abruptly. Finding these adversarial examples is often as simple as perturbing the system in the direction of the largest gradient of the decision function [104].

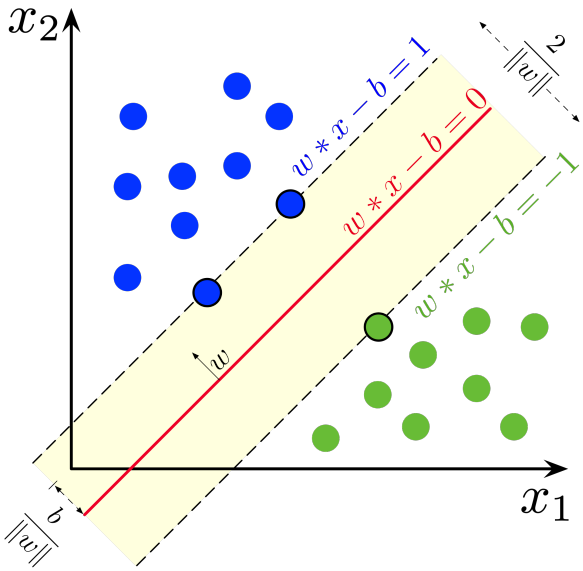


Figure 3.5 **Support Vector Machine.** The margin of the support vector machine is chosen so that one class lies in $\vec{w} \cdot \vec{x} - b \geq 1$ and the other $\vec{w} \cdot \vec{x} - b \leq -1$. This is equivalent to saying that the margin is $2/|\vec{w}|$. Image taken from Wikipedia under license CC-BY-SA 4.0.

Adversarial examples are dangerous if machine learning models are intended to make decisions in real life. For this reason, there have been efforts in preventing them. In our article, we provide a way to use theorem 11 from Ref. [135], which ensures that training the model on a δ-cover of the subclasses makes the model robust against these examples for sufficiently small δ. Subclasses must be understood in this context as manifolds of a lower dimension embedded in the high dimensional space. How large δ is, will depend on the distance between classes in the high dimensional space. Setting δ very low makes the coverage more fine-grained and more robust, but also more expensive, so it is important to reliably find the distance between classes.

That being the objective, we use Support Vector Machines, which naturally define the margin or distance of separation between classes. In particular, a linear Support Vector Machine might be defined as the decision boundary provided by

$$\vec{w} \cdot \vec{x} - b = 0, \tag{3.60}$$

where $2/|\vec{w}|$ is chosen to be the margin of the SVM, for example, the separation between the two classes. Since we want to make sure we have found the minimum distance between two classes, what we can do is find examples from both classes that minimize the margin of the trained Support Vector Machine, see Fig. 3.5. This way we will find

an accurate approximation to the maximum allowed δ. We conceptualize this objective in the 'informativeness', defined as $P_c(\vec{x}_{n+1}) \cdot |\vec{w}_{n+1}|$, where $\vec{x}_{n+1}$ is the candidate point to be added to the training set, $P_c(\vec{x}_{n+1})$ evaluates the probability that this new point is in class c, and $|\vec{w}_{n+1}|$ is proportional to the inverse size of the margin of the SVM if the point (or example) was added to such class. In conclusion, we want to find examples that maximize the 'informativeness', so we can add them to the training set, and find an accurate approximation of the maximum allowed δ.

The quantum algorithm we use in this case is a quantization of the Support Vector Machine proposed in Ref. [205]. The basic idea is to solve the linear system of equations

$$F \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & K + \gamma^{-1}\mathbf{1} \end{pmatrix} \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ \vec{y} \end{pmatrix}. \tag{3.61}$$

where $K_{ij} = \vec{x}_i \cdot \vec{x}_j$. Then, the solution vector $(b, \vec{\alpha})$ encoded as a quantum state can be used in a swap test (see Fig. 2.4) to calculate the class to which a given point $\vec{x}_{n+1}$ could be added and with which probability.

Similarly, once we have prepared $|b, \vec{\alpha}\rangle$, we can compute $\vec{w}_{n+1}$ using the definition $\vec{w} = \sum_{i=1}^{n+1} \alpha_i \vec{x}_i$. To do so, we use linear algebra to multiply

$$A\,|b, \vec{\alpha}\rangle := \begin{pmatrix} 0 & x_{1,1} & \dots & x_{1,m} \\ \vdots & & & \vdots \\ 0 & x_{n+1,1} & \dots & x_{n+1,m} \end{pmatrix} \begin{pmatrix} b \\ \alpha_1 \\ \vdots \\ \alpha_{n+1} \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{m} \alpha_j x_{1,j} \\ \vdots \\ \sum_{i=1}^{m} \alpha_j x_{n+1,j} \end{pmatrix} = \vec{w}. \tag{3.62}$$

This requires a similar approach to the linear combination of units that we use to implement A^{-1} as a Fourier or Chebyshev series, but for A defined as above. Once we have done that, we can obtain the norm $|\vec{w}_{n+1}|$ as a function of the success probability in the application of F^{-1} and A to the solution. Overall, we can see that provided qRAM access to the data this procedure can be implemented in polylogarithmic complexity in the number of points and the dimension of each element of the dataset.

3.6. Dequantization

In the previous two sections, we have exemplified how to apply quantum linear algebra techniques to solve optimization and machine learning problems. However, it is also the case that these algorithms rely on qRAM access to the data whose complexity is not counted in the final algorithm, only its execution time. Is there a way to compare

them with classical algorithms in a fair setting? This is the question we aim to answer in this section. The key to this comparison lies in the data access procedure. To compare them, Ewin Tang introduced the following concept that mimics the qRAM requirements in a classical setting [237].

Definition 6 (Sample and query access) *We say we have $O(T)$ sample and query access to a vector $x \in \mathbb{C}^N$, if we can in time $O(T)$, a) ℓ_2-sample indices i according to probabilities $\frac{|x_i|^2}{\|x\|^2}$; and b) query an index i to obtain x_i, and query for $\|x\|$ too. For a matrix $A \in \mathbb{C}^{N\times M}$, we similarly say that we have sample and query access, if we have sample and query access to the vector of norms of rows of A, $(|A_{1,*}|, \ldots, |A_{N,*}|)$, and sample and query access to each row of A taken as a vector. We will denote query access by $Q(x)$, sample and query access by $SQ(x)$, and if only an upper bound $(1+\nu)\|x\|$ to the norm is known, $SQ^{\nu}(x)$.*

This data access is similar to the one provided by qRAMs, since the qRAM allows one to query for any entry, and measuring the prepared state provides sample access. Similarly happens for the sparse-access oracle input model assumed by Algorithm 10.

Using this definition, we can implement three key subroutines in a classical computer that will help us mimic the quantum algorithms in a classical computer, as long as those algorithms operate in a low rank (low dimensional) subspace [238]. The key is to substitute the state preparation assumptions from the qRAM, with a ℓ_2-sampling assumption in classical computing. This is the reason it is called *dequantization*. Specifically, we will say we can dequantize a quantum algorithm if the quantum algorithm runs in time T and the classical algorithm in time $O(\text{poly}(T))$. The first dequantization subroutine is a dequantized swap test, which allows computing the inner product between two vectors. This 'dequantized' inner product algorithm achieves

Algorithm 12 Inner product estimation

1: **Input**: $SQ(x) \in \mathbb{C}^N$, $Q(y) \in \mathbb{C}^N$. Precision ϵ^{-1}, failure probability δ.
2: **Output**: An ϵ-precise estimation of $\langle x|y\rangle$ with probability $1-\delta$.
3: Let $s = O(\frac{1}{\epsilon^2}\log\frac{1}{\delta})$
4: Measure $i_1, \ldots, i_s$ from x.
5: Compute $z_j = x_j^{\dagger} y_j \frac{\|x\|^2}{|x_j|^2}$ for $j \in \{i_1, \ldots, i_s\}$.
6: Distribute the measurements in $O(\log\frac{1}{\delta})$ groups and compute the average within each group.
7: Compute and output the median of the averages.

precision ϵ^{-1} with a probability of failure δ.

Lemma 1 (Inner product, proposition 4.2 in [237]) . *Given $SQ^\nu(\vec{x})$ and $Q(\vec{y})$ we can estimate $\langle\vec{x}|\vec{y}\rangle$ to precision ϵ and probability $\geq 1-\delta$ in time $O\left(T\epsilon^{-2}\log\delta^{-1}\right)$.*

The quantum advantage in this subroutine stems from the use of amplitude estimation, a quadratic advantage.

The second subroutine allows obtaining sample and query access from the result of a shallow matrix-vector product. Shallow, because the length of the vector k will be relatively small compared to the number N of rows in the matrix. The technique used is based on rejection sampling: if we can sample from one distribution $Q(s)$, and want to sample from $P(s)$ while being able to compute probabilities from both, we can:

1. Sample i from Q.
2. With probability $\frac{P(i)}{MQ(i)}$ output i, else restart, where $M=\max_i \frac{P(i)}{Q(i)}$.

In our case, the target distribution P will be Ww from which we can compute the entries in complexity $O(k)$, and the input probability Q, sampling from $\|W_{*,j}\|$ with probability $\|W_{*,i}\|^2|w_i|^2$.

Algorithm 13 Shallow matrix-vector product sample and query access

Input: $SQ(V^\dagger)\in\mathbb{C}^{k\times N}$, $Q(w)\in\mathbb{C}^k$.
Output: $SQ^\nu(Vw)$.
`Query`: On input s, compute and output $(Vw)_s$.
`Sample`: Run `RejectionSampling` until success, or $kC(V,w)\log\frac{1}{\delta}$ failures.
`Norm`: Run `RejectionSampling` $\frac{k}{\nu^2}C(V,w)\log\frac{1}{\delta}$ times, and let p be the success rate. Output $pk\sum_i|w_i|^2\|V_{*,i}\|^2$.
function RejectionSampling($SQ(V^\dagger)$, $Q(w)$)
    Sample $i\in[k]$ proportional to $\|V_{*,i}\|^2|w_i|^2$.
    Sample $s\in[N]$ from $\|V_{*,i}\|$.
    Compute $p_s=\frac{(Vw)_s^2}{k\sum_j(V_{s,j}w_j)^2}=\frac{(\sum_j V_{s,j}w_j)^2}{k\sum_j(V_{s,j}w_j)^2}$.
    With probability p_s, output s; else failure $\emptyset$.
end function

Using algorithm Algorithm 13, one can prove the following result.

Lemma 2 (Thin matrix-vector product, proposition 4.3 in [237]) . *Given a matrix $V\in\mathbb{C}^{n\times k}$, $w\in\mathbb{C}^k$, and given $SQ(V^\dagger)$ and $Q(w)$, we can obtain $SQ^\nu(Vw)$ with success probability $\geq 1-\delta$ and complexities*

1. *query in time $O(Tk)$,*
2. *sample in time $O(Tk^2C(V,w)\log\delta^{-1})$,*

3. query the norm in time $O(Tk^2C(V,w)\nu^{-2}\log\delta^{-1})$.

where $C(V,w)=\sum_i \|w_iV_{*,i}\|^2/|Ww|^2$, *and* $V_{*,i}$ *is the* i^{th} *column of* V.

Notice how the product of any row of V and w is computed directly, given that $k \ll N$. Quantum algorithms will obtain a quantum advantage in this case when k is large, but this technique is often used in combination with a low-rank approximation of a matrix S resulting from the third subroutine, which we explain next.

Algorithm 14 Low-rank singular vector matrix decomposition

1: **Input**: $SQ(A) \in \mathbb{C}^{M\times N}$, singular value cutoff σ, error ϵ, failure probability δ.
2: **Output**: $SQ(S) \in \mathbb{C}^{N\times q}$, $Q(U) \in \mathbb{C}^{q\times \ell}$, $Q(\Sigma) \in \mathbb{C}^{\ell\times\ell}$.
3: $K = \frac{\|A\|_F^2}{\sigma^2}$ and $q = \Theta(\frac{K^2}{\epsilon^2}\log(\frac{1}{\delta})$.
4: ℓ_2-sample q row indices, from distribution $\frac{\|A_{i,*}\|_F^2}{\|A\|_F^2}$, rescaling $S_{i_r,*} = \frac{\|A\|_F}{\sqrt{q}\|A_{i_r,*}\|}A_{i_r,*}$.
5: Sample q column indices from distribution $\mathcal{F}$, obtained from first uniformly sampling $r \sim [q]$, and then ℓ_2-sampling c from $S_{r,*}$.
6: Let W be the resulting matrix from rescaling $W_{*,c} = \frac{\|A\|_F}{\sqrt{q}\|S_{*,q_c}\|}S_{*,q_c}$.
7: Compute singular vectors $u^{(1)},\dots,u^{(\ell)}$ of W corresponding to singular values $\sigma^{(1)},\dots,\sigma^{(\ell)}$ larger than σ.
8: Output $SQ(S)$, the matrix $U \in \mathbb{R}^{q\times\ell}$ composed of $u^{(i)}$ taken as columns, and Σ the diagonal matrix with entries $\sigma^{(i)}$.

This algorithm allows us to perform a low-rank approximation of A in the subspace spanned by the sampled left singular vectors $D = AVV^\dagger$ for $V = S^\dagger U\Sigma^{-1}$.

Theorem 1 (Low-rank singular vector approximation, theorem 4.4 in [237]) *Suppose* $O(T)$*-time* $SQ(A)$, $A \in \mathbb{C}^{n\times d}$*; a singular value threshold* σ *and an error parameter* $\epsilon \in (0, \sqrt{\sigma/||A||_F}/4]$. *Let* $K = ||A||_F^2/\sigma^2$. *Then, in time*

$$O\left(\frac{K^{12}}{\epsilon^6}\log^3\delta^{-1} + T\frac{K^8}{\epsilon^4}\log^2\delta\right) \tag{3.63}$$

we output $SQ(S)$, $S \in \mathbb{C}^{q\times n}$, $U \in \mathbb{C}^{q\times l}$, $\Sigma \in \mathbb{R}^{l\times l}$, *with* $l = \Theta(K^4\epsilon^{-2}\log^2\delta^{-1})$. *These matrices implicitly describe the low-rank approximation of* A, $D = AVV^\dagger$, *with* $V = S^\dagger U\Sigma^{-1}$. *Additionally, with probability* $\geq 1-\delta$, $||A-D||_F^2 \leq ||A-A_l||_F^2 + \epsilon||A||_F^2$.

These three subroutines are very powerful and allow to dequantize several linear algebra problems where the quantum algorithm operates in low dimensional subspaces.

3.6.1. Examples

Recommendation systems. The first example of a dequantization algorithm was applied to the quantum recommendation system algorithm by Kerenidis and Prakash [132]. In the recommendation problem, we are given a large matrix of $A \in \mathbb{R}^{M\times N}$ (users and products) and one has to decide what product has a greater fit with a given user. Mathematically, the objective is to project row A_i corresponding to the i^{th} user, into the space spanned by the singular vectors with the largest singular values, representing the products he or she is more likely to enjoy. In other words, sample from $A_i V V^\dagger = D_i$. Kerenidis and Prakash used a simple strategy, shown in Algorithm 15.

Algorithm 15 Quantum recommendation system algorithm

1: **Input**: Quantum accessible data structure of preferences matrix A (see (3.28)), threshold σ, error ϵ, user i.
2: **Output**: Sample access to any row of the low rank projection of A, $\hat{A}$.
3: Use the quantum accessible data structure to initialize state $|A_i\rangle$ corresponding to user i, formally represented as a sum over singular vectors $|A_i\rangle = \sum_j \alpha_j |v_j\rangle$.
4: Perform singular value estimation, $\sum_j \alpha_j |v_j\rangle |\bar{\sigma}_j\rangle$, using Algorithm 11.
5: Flag those singular values that are above threshold σ, $\sum_j \alpha_j |v_j\rangle |\bar{\sigma}_j\rangle |\bar{\sigma}_j \geq \sigma\rangle$,
6: Uncompute the singular values, $\sum_j \alpha_j |v_j\rangle |0\rangle |\bar{\sigma}_j \geq \sigma\rangle$.
7: Measure the last register and postselect on $|\bar{\sigma}_j \geq \sigma\rangle$.
8: Measure and output the first register on the computational basis: the result is a sample from the projection of $|A_i\rangle$, into the space spanned by the singular vectors with the largest singular values.

How can we dequantize this algorithm? Recall that we said that $D = AVV^\dagger$ represents a low-rank projection of the A matrix. In Ref. [237], showed how to obtain sample and query access to the singular value decomposition of V, proving Theorem 1. In this case, we are interested in the projection of a single row of A, so we want to sample from $A_i(S^\dagger U\Sigma)(S^\dagger U\Sigma)^\dagger = (A_i S^\dagger)(U\Sigma\Sigma^\dagger U^\dagger)S$. The first parentheses can be understood as a 'small' number of inner products (Algorithm 12), where small indicates the target rank of the decomposition. The resulting vector can then be multiplied by the second parentheses, which is a product of small matrices. Finally, we can also perform a shallow matrix-vector product with S (Algorithm 13) to obtain sample and query access to the resulting distribution.

Linear system of equations. Another subroutine we can 'dequantize' is the HHL algorithm for low-rank matrices. This is a very important condition, as otherwise Algorithm 14 fails. The quantum linear system Algorithm 10 aims to prepare $|x\rangle =$

$A^{-1}|b\rangle$. The dequantized algorithm will in contrast provide sample and query access to A^+b, where A^+ is the Moore-Penrose pseudo inverse that fulfills that if A is invertible then $A^{-1} = A^+$ [186]. The objective is to sample from

$$A^+b = (A^TA)^+A^Tb \approx \sum_{i=1}^{k} \frac{1}{\Sigma_{ii}} v_i v_i^T A^T b. \tag{3.64}$$

For this decomposition, we will need the Singular Value Decomposition of Algorithm 14. Then, notice that $v_i^T A^T b = \mathrm{Tr}(A^T v_i^T b)$ can be computed via a swap test between two-order tensors A^T and bv_i^T (see for example Ref. [238]). This provides query access to $Q(v_i^T Ab/\Sigma_{ii})$. Then, since we have $SQ(V)$, we also have sample and query access to an approximation of the objective, $SQ(A^+b)$, via the shallow matrix-vector product in Algorithm 13.

Support Vector Machine and adversarial examples. Finally, we can also dequantize our active learning algorithm to sample against adversarial examples. The idea is simple: first, we use the method just explained above to gain sample and query access to $SQ(F^+\vec{y})$, see (3.61). Once we have that, we aim to find the norm

$$||A|b,\vec{\alpha}\rangle||^2 = \langle b,\vec{\alpha}|A^\dagger A|b,\vec{\alpha}\rangle\,, \tag{3.65}$$

which can be computed as an inner product $\langle a|b\rangle$ with

$$a = \sum_i \sum_j \sum_k A_{ji}||A_{k,*}||\,|i\rangle\,|j\rangle\,|k\rangle\,, \qquad b = \sum_i \sum_j \sum_k \frac{w_j w_k A_{ki}}{||A_{k,*}||}\,|i\rangle\,|j\rangle\,|k\rangle\,, \tag{3.66}$$

similarly to the protocol used for Supervised Clustering dequantized in Ref. [238]. Key to the success of this dequantization technique is the fact that the examples from both classes live in a low-dimensional subspace, and therefore we can perform low-rank operations, which we leverage in the quantum algorithm too. The resulting quantum advantage of using quantum linear algebra instead of its dequantized counterpart will consequently be polynomial.

3.7. Results

- We have reviewed Shor's algorithm and its generalization to the Hidden Subgroup Problem over arbitrary Abelian groups.

- We have explained the basic quantum linear algorithm solver and described the complexity improvements carried out over time to its dependence over most parameters, culminating in qubitization as a flexible technique. Qubitization is intimately related to quantum signal processing, which allows to synthesize arbitrary polynomial functions of the encoded Hamiltonian.

- In Ref. [65] we have described a quantum interior point method that improved over previously available results, establishing at the time the state-of-the-art in quantum algorithms for linear programming.

- An important technical achievement in that paper was to ensure that the polynomial complexity in the precision would not hamper the convergence of the classical algorithm in which it is inspired.

- However, due to the dependence on the preparation and readout of the data often relying on qRAMs, the quantum advantage is limited. Another generic limitation to this class of quantum and classical algorithms is the polynomial dependence on the condition number, which in turn becomes increasingly larger as we proceed along the algorithm, thus why interior point methods are called weakly polynomial.

- In Ref. [64] we also explored how to apply quantum linear algebra to Machine Learning problems. In particular, we have tackled the problem of adversarial examples using quantum Support Vector Machines. Finally, we have explained that quantum active learning techniques might help provide guarantees on the robustness of a classical training set.

- The usefulness of the technique in this area relies heavily on the problem having a low-rank nature, which can also be exploited by classical algorithms. Consequently, we have also indicated how our algorithm may be dequantized, and thus implemented in a classical computer, with a polynomial slowdown.

- These points illustrate that the problems we aim to target with quantum linear algebra techniques should be chosen carefully, as generic speedups are often polynomial and not exponential. Consequently, their usefulness will depend more strongly on the error correction overhead.

Chapter 4

Quantum Chemistry

> The program that Fredkin is always pushing, about trying to find a computer simulation of physics, seems to me to be an excellent program to follow out... And I'm not happy with all the analyses that go with just the classical theory, because nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy.
>
> Richard P. Feynman, *Simulating Physics with Computers*

4.1. Objectives

- Understand the most common classical ab-initio chemistry techniques.
- Building on the quantum walk and linear algebra techniques explained in previous chapters describe the available Hamiltonian simulation techniques, as well as the tradeoffs present in this and other algorithmic choices.
- Benchmark the different Hamiltonian simulation techniques.
- Describe a problem where quantum phase estimation might be applied, and provide rigorous cost estimates.

4.2. Introduction

As indicated in the quote, one of the most promising problems to which one can apply quantum computing is the study of quantum systems, interesting for basic

science purposes but also for the chemical and material science industries. The reason quantum computing is so well suited for chemical problems is that quantum states can be represented exactly on a quantum computer, and the evolution of (closed) systems is unitary and relatively efficient to implement.

In this chapter, we will explore algorithms tailored for this research area. Quantum phase estimation will in particular stand a key role in these algorithms and will make use of Hamiltonian simulation as a key subroutine. While quantum phase estimation allows to efficiently compute the energy corresponding to an eigenstate, preparing the corresponding ground state is often hard. Consequently, this chapter is divided into three main sections: the first section reviews basic classical algorithms for quantum algorithms; the second explains Hamiltonian simulation and its application to quantum phase estimation; and finally, the third introduces different quantum state preparation techniques.

4.3. Classical quantum chemistry

4.3.1. Hartree-Fock

The Hartree-Fock procedure is considered a basic step in the calculation of an approximation to the ground state, and a stepping stone for more complex procedures afterward. The quantum chemistry problems we are interested in assume access to the molecular Hamiltonian. Let η and L represent the number of electrons and nuclei, respectively, indexed by i and ℓ, and at positions r_i and R_ℓ. Let also Z_ℓ and m_ℓ be the charge and mass of nuclei ℓ. Knowing that each particle has kinetic energy and that there is a Coulomb potential between charged particles, we can write the molecular Hamiltonian as

$$H = \underbrace{-\sum_{i=1}^{\eta} \frac{\nabla_i^2}{2m_e}}_{T} \underbrace{-\sum_{i=1}^{\eta}\sum_{\ell=1}^{L} \frac{Z_\ell}{\|R_\ell - r_i\|}}_{U} + \underbrace{\sum_{\substack{i,j=1 \\ i\neq j}}^{\eta} \frac{1}{2}\frac{1}{\|r_i - r_j\|}}_{V} \underbrace{-\sum_{\ell=1}^{L} \frac{\nabla_\ell^2}{2m_\ell}}_{T_{\text{nuclei}}} + \underbrace{\sum_{\substack{\ell,\kappa=1 \\ \ell\neq\kappa}}^{L} \frac{1}{2}\frac{Z_\ell Z_\kappa}{\|R_\ell - R_\kappa\|}}_{V_{\text{nuclei}}}. \tag{4.1}$$

We want to solve the Schrödinger equation, $H|\psi\rangle = E|\psi\rangle$, and find both $|\psi\rangle$ and E. However, this problem is very complicated, so we will make a series of approximations. The first such approximation is the so-called Born-Oppenheimer approximation: since the mass of the nuclei is much larger than the mass of the electrons $m_\ell \gg m_e$, we can decouple the degrees of freedom of the nuclei from the rest of the wave function, and

analyze them classically [48]. With this approximation, the molecular Hamiltonian (4.1) is reduced to the electronic Hamiltonian:

$$H = T + U + V, \tag{4.2}$$

with T, V, and U defined above as the kinetic, potential, and external potential operators. We also define the one-body term $h = T + U$, leaving V as the two-body component. Additionally, the Hartree-Fock procedure approximates the wave function of the electrons as a product of disentangled wave functions for each electron, with the characteristic antisymmetrization of fermionic wavefunctions

$$|\psi\rangle = \frac{1}{\sqrt{\eta!}} \sum_{\sigma \in S_\eta} \operatorname{sgn}(\sigma) \, |\phi_{\sigma(1)}\rangle \otimes \ldots \otimes |\phi_{\sigma(\eta)}\rangle = \frac{1}{\sqrt{\eta!}} \begin{vmatrix} \phi_1(\boldsymbol{x}_1) & \phi_1(\boldsymbol{x}_2) & \ldots & \phi_1(\boldsymbol{x}_\eta) \\ \phi_2(\boldsymbol{x}_1) & \phi_2(\boldsymbol{x}_2) & \ldots & \phi_2(\boldsymbol{x}_\eta) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_\eta(\boldsymbol{x}_1) & \phi_\eta(\boldsymbol{x}_2) & \ldots & \phi_\eta(\boldsymbol{x}_\eta) \end{vmatrix}. \tag{4.3}$$

The latter form gives such wavefunction the name of *Slater determinant*, and since there is no entanglement, the Hartree-Fock procedure is referred to as a mean-field approach. To simplify the notation, we will use $|i\rangle := |\phi_i\rangle$, and will also variationally enforce this basis functions to be orthogonal $\langle i|j\rangle = \delta_{i,j}$.

We want to compute the energy of this wavefunction,

$$E = \langle \psi | H | \psi \rangle = \int \psi^\dagger(\boldsymbol{r}) H \psi(\boldsymbol{r}) d\boldsymbol{r}. \tag{4.4}$$

Since the wavefunctions of individual electrons are disentangled, we can compute the one-electron one-body Hamiltonian matrix elements on individual orbitals,

$$\langle p|h(1)|q\rangle = \int d\boldsymbol{x} \phi_p^\dagger(\boldsymbol{x}) \left(\frac{\nabla^2}{2} + \sum_{\ell=1}^{L} \frac{Z_\ell}{\|R_\ell - r\|} \right) \phi_q(\boldsymbol{x}), \tag{4.5}$$

where $\boldsymbol{x} = (r, \sigma)$ for σ the spin and r_i the spatial coordinates. Similarly, the two-electron two-body term matrix elements might be written as

$$\langle pq|V(2)|rs\rangle = \frac{1}{2} \int d\boldsymbol{x}_1 d\boldsymbol{x}_2 \phi_p^\dagger(\boldsymbol{x}_1) \phi_q^\dagger(\boldsymbol{x}_2) \left(\frac{1}{\|r_1 - r_2\|} \right) \phi_r(\boldsymbol{x}_1) \phi_s(\boldsymbol{x}_2). \tag{4.6}$$

This last integral form is written by chemists in notation $[pr|qs]$, in contrast to physicists' notation of $\langle pq|rs\rangle$.

Overall, the Hartree-Fock energy is obtained by minimizing $\langle\psi|H|\psi\rangle$ subject to the orthogonality constraints we mentioned. From this procedure, we obtain [234, Eq. 2.110],

$$E_{HF} = \sum_{p=1}^{\eta} \langle p|h|p\rangle + \sum_{p>q}^{\eta} [pp|qq] - \sum_{p>q}^{\eta} [pq|qp]. \tag{4.7}$$

Since $\phi^\dagger(\boldsymbol{x})\phi(\boldsymbol{x})d\boldsymbol{r}$ represents a probability density, the first term makes sense as the weighted average of the one-body Hamiltonian term when the electron is in orbital ϕ_i. Similarly, the term called Coulomb term

$$[pp|qq] = \frac{1}{2}\int d\boldsymbol{x}_1 d\boldsymbol{x}_2 \phi_p^\dagger(\boldsymbol{x}_1)\phi_p(\boldsymbol{x}_1)\left(\frac{1}{\|r_1 - r_2\|}\right)\phi_q^\dagger(\boldsymbol{x}_2)\phi_q(\boldsymbol{x}_2) \tag{4.8}$$

can also be interpreted as a two-variable expected value, in integral form. Finally, the 'exchange' term

$$[pq|qp] = \frac{1}{2}\int d\boldsymbol{x}_1 d\boldsymbol{x}_2 \phi_p^\dagger(\boldsymbol{x}_1)\phi_q(\boldsymbol{x}_1)\left(\frac{1}{\|r_1 - r_2\|}\right)\phi_q^\dagger(\boldsymbol{x}_2)\phi_p(\boldsymbol{x}_2) \tag{4.9}$$

arises from the antisymmetry of the Hartree-Fock wave function, but has no classical meaning associated. Other terms are canceled out by orthonormality of the molecular wavefunctions $|\phi\rangle$.

Next, we separate the spin from the spatial part of the wavefunction. For example, $\phi_i(\boldsymbol{x})d\boldsymbol{x} = \varphi_i(\boldsymbol{r})\sigma(\omega)d\boldsymbol{r}d\omega$. This will lead, for example, to decomposing

$$\langle p|h|q\rangle = \int d\omega \sigma_p(\omega)\sigma_q(\omega)\int d\boldsymbol{r}\varphi_p^\dagger(\boldsymbol{r})\varphi_q(\boldsymbol{r}) =: \left[\int d\omega \sigma_p^\dagger(\omega)\sigma_q(\omega)\right](p|h|q), \tag{4.10}$$

where $(i|h|j)$ denotes the spatial integral. Similarly,

$$\begin{aligned}[pr|qs] := \frac{1}{2}\int d\boldsymbol{x}_1 d\boldsymbol{x}_2 \phi_p^\dagger(\boldsymbol{x}_1)\phi_r(\boldsymbol{x}_1)\left(\frac{1}{\|r_1 - r_2\|}\right)\phi_q^\dagger(\boldsymbol{x}_2)\phi_s(\boldsymbol{x}_2) =: \\ \left[\int d\omega_1 \sigma_p^\dagger(\omega_1)\sigma_r(\omega_1)\right]\left[\int d\omega_2 \sigma_q^\dagger(\omega_2)\sigma_s(\omega_2)\right](pr|qs),\end{aligned} \tag{4.11}$$

where rounded brackets indicate the spatial component,

$$(pr|qs) := \frac{1}{2}\int d\boldsymbol{r}_1 d\boldsymbol{r}_2 \varphi_p^\dagger(\boldsymbol{r}_1)\varphi_r(\boldsymbol{r}_1)\left(\frac{1}{\|r_1 - r_2\|}\right)\varphi_q^\dagger(\boldsymbol{r}_2)\varphi_s(\boldsymbol{r}_2). \tag{4.12}$$

This can be further simplified by taking into account that

$$\int d\omega \sigma_p(\omega)\sigma_q(\omega) = \delta_{\sigma_p,\sigma_q}. \tag{4.13}$$

The terms $\langle p|h|p\rangle$ and $[pp|qq]$ will consequently always survive, but the exchange term may not, depending on the spin of $|p\rangle$ and $|q\rangle$ involved.

Overall, the Hartree-Fock Hamiltonian acting on a molecular orbital $|i\rangle$ can be written as

$$H\,|\phi_i(\boldsymbol{x}_1)\rangle = h(\boldsymbol{x}_1)\phi_i(\boldsymbol{x}_1) + \sum_{j\neq i} \underbrace{\int d\boldsymbol{x}_2 \frac{|\phi_j(\boldsymbol{x}_2)|^2}{|r_2 - r_1|}}_{J_j(\boldsymbol{x}_1)} \phi_i(\boldsymbol{x}_1) - \sum_{j\neq i} \underbrace{\int d\boldsymbol{x}_2 \frac{\phi_j^\dagger(\boldsymbol{x}_2)\phi_i(\boldsymbol{x}_2)}{|r_2 - r_1|}}_{K_j(\boldsymbol{x}_1)} \phi_j(\boldsymbol{x}_1). \tag{4.14}$$

This is called the Fock operator [234, Eq. 3.16]

$$f(\boldsymbol{x}) = h(\boldsymbol{x}) + \sum_j (J_j(\boldsymbol{x}) - K_j(\boldsymbol{x})), \tag{4.15}$$

and leads to the eigenvalue equation that determines the Hartree Fock energy [234, Eq. 3.17]

$$f(\boldsymbol{x})\phi_i(\boldsymbol{x}) = \epsilon_i \phi_i(\boldsymbol{x}). \tag{4.16}$$

If we solve this equation, then $E_{HF} = \frac{1}{2}(\sum_i \epsilon_i + \langle i|h|i\rangle) \neq \sum_i \epsilon_i$. The reason is that, defining $J_{i,j} = \langle i|J_j|i\rangle = [jj|ii]$ and $K_{i,j} = \langle i|K_j|i\rangle = [ji|ij]$,

$$\sum_i \epsilon_i = \sum_i \langle i|h|i\rangle + \sum_{i,j}(J_{i,j} - K_{i,j}) \neq \sum_i \langle i|h|i\rangle + \sum_{i>j}(J_{i,j} - K_{i,j}) = E_{HF}, \tag{4.17}$$

so by directly adding the individual orbital energies ϵ_i, we would be double-counting the interaction between electrons in those molecular orbitals.

Without adding more structure to the orbitals, (4.16) is a complicated integro-differential equation. For this reason, Roothaan proposed decomposing each molecular orbital $|\phi_i\rangle$ as a linear combination of atomic orbitals [210]:

$$\phi_i = \sum_\mu C_{\mu,i}\tilde{\phi}_\mu, \tag{4.18}$$

where $\{\tilde{\phi}_\mu\}$ are the basis of non-orthogonal atomic orbitals. Then, defining the Fock matrix in this new basis as

$$F_{\mu,\nu} = \int d\boldsymbol{x} f(\boldsymbol{x})\tilde{\phi}_\mu^\dagger(\boldsymbol{x})\tilde{\phi}_\nu(\boldsymbol{x}) \tag{4.19}$$

Algorithm 16 Hartree-Fock procedure

1: **Input**: Access to the electronic Hamiltonian (4.2).
2: **Output**: Molecular orbitals $\{\phi_i\}$ and their energies ϵ_i.
3: Initialize the overlap matrix of atomic orbitals $S_{\mu,\nu}$.
4: Guess initial Molecular Orbital coefficients $C_{\mu,i}$.
5: **while** $C_{\mu,i}$ has not converged **do**:
6: Form the Fock matrix $F_{\mu,\nu}$. ▷ Requires computing $f(\boldsymbol{x})$
7: Solve the generalized eigenvalue equation (4.21) and find $C_{\mu,i}$.
8: **end while**

and the orbital overlap matrix as

$$S_{\mu,\nu} = \int d\boldsymbol{x} \tilde{\phi}_\mu^\dagger(\boldsymbol{x}) \tilde{\phi}_\nu(\boldsymbol{x}), \tag{4.20}$$

we rewrite (4.16) as [234, Eq. 3.35]

$$\sum_\nu F_{\mu,\nu} C_{\nu,i} = \epsilon_i \sum_\nu S_{\mu,\nu} C_{\nu,i} \Rightarrow FC = SC\epsilon. \tag{4.21}$$

A key aspect to note is that the Fock matrix will depend on the molecular orbitals through $f(\boldsymbol{x})$, but finding these coefficients requires solving (4.21). For this reason, we resort to a self-consistent procedure to generate molecular orbitals and solve the Hartree-Fock equation. From this algorithm, the costliest step is computing the Fock operator $f(\boldsymbol{x})$ which involves up to $O(N^4)$ molecular orbital integrals, due to the 4 indices in (4.8) and (4.9). This can be reduced to $O(N^3)$ using density-fitting approximations [253], or if the size of the system is large enough, neglecting coefficients of far apart atomic orbital integrals leads to $O(N^2)$. Finally, a technique called 'fast multipole' methods may bring that complexity down to $O(N)$ in the latter case [107].

4.3.2. Density Functional Theory

In the previous section, we studied how a disentangled wave function allows us to approximate the ground state and energy. The problem with wave functions, however, is that since they have to model η fermions, their dimension will scale as $\mathbb{C}^{3\eta}$, making it difficult to represent large systems accurately.

A possible alternative is to use the density functional

$$\begin{aligned} n(\boldsymbol{r}) &= \int d\boldsymbol{r}_1 \dots d\boldsymbol{r}_\eta \psi^\dagger(\boldsymbol{r}_1, \dots, \boldsymbol{r}_\eta)(\delta(\boldsymbol{r} - \boldsymbol{r}_1) + \dots \delta(\boldsymbol{r} - \boldsymbol{r}_\eta))\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_\eta) \\ &= \eta \int d\boldsymbol{r}_2 \dots d\boldsymbol{r}_\eta |\psi(\boldsymbol{r}, \boldsymbol{r}_2, \dots, \boldsymbol{r}_\eta)|^2 \end{aligned} \tag{4.22}$$

as a basic function to model the system, since its dimension will only be $\mathbb{R}^3$. Note that this density functional is different from the probability density $\rho(r) = \psi^\dagger\psi$. However, is the density functional a good 'basic variable' to model chemical properties in which we might be interested? In a breakthrough article in 1964, Hohenberg and Kohn proved two theorems that answered positively this question [122].

Theorem 2 (Hohenberg and Kohn) *In any many-body system interacting under an electronic Hamiltonian, there is a one-to-one relation between the ground state electronic density $n_0(\boldsymbol{r})$ and the external potential U up to a constant.*

The proof is carried out by reductio ad absurdum. Let us suppose that we have two external potentials $U^{(1)}$ and $U^{(2)}$, which necessarily lead to different ground states $\psi^{(1)}$ and $\psi^{(2)}$ but the same ground-state electronic density $n_0(\boldsymbol{r})$. Since we have stated they are the ground state, we have[1]

$$E^{(1)} = \langle\psi^{(1)}|H^{(1)}|\psi^{(1)}\rangle < \langle\psi^{(2)}|H^{(1)}|\psi^{(2)}\rangle = \langle\psi^{(2)}|H^{(2)}|\psi^{(2)}\rangle - \langle\psi^{(2)}|H^{(1)} - H^{(2)}|\psi^{(2)}\rangle = \\ = E^{(2)} - \int d\boldsymbol{r}(U^{(1)} - U^{(2)})n_0(\boldsymbol{r}). \tag{4.23}$$

Similarly, we can find that

$$E^{(2)} < E^{(1)} - \int d\boldsymbol{r}(U^{(2)} - U^{(1)})n_0(\boldsymbol{r}), \tag{4.24}$$

and adding them together we reach the conclusion that

$$E^{(2)} + E^{(1)} < E^{(2)} + E^{(1)}, \tag{4.25}$$

which is a contradiction. In summary, the definition of the density functional (4.16) entails that the same wave function implies the same density functional, and in this theorem, we have shown the opposite: if we have two ground states of different Hamiltonians, then they will result in different density functionals. Consequently, there is a one-to-one relationship between density functionals and external potentials.

The second theorem introduces the energy functional:

Theorem 3 (Hohenberg and Kohn) *There exists a total energy functional $E_{HK}[n]$ valid for any external potential U, which can be minimized with respect to the energy, and find the ground state energy $E[n]_{HK}(\boldsymbol{r})$ and density $n_0(\boldsymbol{r})$.*

[1] Following Richard Martin's book [175] and original proof by Hohenberg and Kohn, we assume that the Hamiltonians are not degenerate, but this condition can be relaxed, see [145].

The form of this energy functional is

$$E_{HK}[n] = T[n] + V[n] + \int d\boldsymbol{r} U(\boldsymbol{r}) n(\boldsymbol{r}) + \sum_{\ell,\kappa=1}^{L} E_{\ell,\kappa} \tag{4.26}$$

where $\sum_{\ell,\kappa=1}^{L} E_{\ell,\kappa}$ is the nuclei-nuclei interaction energy. Since this Hohenberg-Kohn functional corresponds to the expectation value of the Hamiltonian, the energy of the ground state and the energy density can be found by minimizing the expected energy $\langle\psi|H|\psi\rangle$ via the variational principle.

$$\frac{\delta}{\delta n(\boldsymbol{r})}\left[E_{HK}[n] - \mu \int d\boldsymbol{r}' n(\boldsymbol{r}')\right] = 0, \tag{4.27}$$

with μ a Lagrange multiplier. The first two terms, additionally, are independent of the external potential, and therefore the same for any system. As such, it is called universal functional,

$$F_{HK}[n] = T[n] + V[n]. \tag{4.28}$$

However, while we know how to perform the calculation of the external potential component of the energy functional, it is not clear how to describe the kinetic and potential functionals.

To overcome this limitation, a year later Kohn and Sham proposed an ansatz that assumes that the ground state density functional $n_0(\boldsymbol{r})$ is equal to that of a non-interacting system, with effective (often local) potential $V_{KS}[n]$ [147]. For such a non-interacting system, we can take the Hartree-Fock approximation

$$n(\boldsymbol{r}) = \sum_{i=1}^{\eta} |\phi_i(\boldsymbol{r})|^2. \tag{4.29}$$

The kinetic functional will be defined as

$$T_s[n] = -\frac{1}{2}\sum_{i=1}^{\eta} \langle\phi_i|\nabla^2|\phi_i\rangle = \frac{1}{2}\sum_{i=1}^{\eta} \int d\boldsymbol{r} |\nabla\phi_i|^2. \tag{4.30}$$

Similarly, the Hartree-Fock electron-electron energy functional can now be written as

$$E_{HF}[n] = \frac{1}{2}\int d\boldsymbol{r} d\boldsymbol{r}' \frac{n(\boldsymbol{r})n(\boldsymbol{r}')}{|\boldsymbol{r} - \boldsymbol{r}'|}. \tag{4.31}$$

Using all of this, the energy of the Kohn-Sham model can be written as [175, Eq. 7.5]

$$E_{KS}[n] = T_s[n] + \int d\boldsymbol{r} U(\boldsymbol{r}) n(\boldsymbol{r}) + E_{HF}[n] + \sum_{\ell,\kappa=1}^{L} E_{\ell,\kappa} + E_{xc}[n], \tag{4.32}$$

where $E_{xc}[n]$ is called the exchange-correlation functional because it captures the remaining correlation and exchange interaction that the non-interacting kinetic and potential functionals do not. Formally,

$$E_{xc}[n] = F_{HK}[n] - (T_s[n] + E_{HF}[n]) = \langle T \rangle - T_s[n] + \langle V \rangle - E_{HF}. \tag{4.33}$$

In practice, we will use different parameterized functionals, depending on the computational budget, precision, and use case. To minimize the overall energy we can use the variational principle

$$\frac{\delta E_{KS}[n]}{\delta \phi_i^\dagger(\boldsymbol{r})} = \frac{\delta T_s}{\delta \phi_i^\dagger(\boldsymbol{r})} + \left[\frac{\delta \langle U \rangle}{\delta n(\boldsymbol{r})} + \frac{E_{HF}[n]}{\delta n(\boldsymbol{r})} + \frac{\delta E_{xc}[n]}{\delta n(\boldsymbol{r})} \right] \frac{\delta n(\boldsymbol{r})}{\delta \phi_i^\dagger(\boldsymbol{r})} = 0, \tag{4.34}$$

subject to orthonormality of the molecular orbitals $\langle \phi_i | \phi_j \rangle = \delta_{i,j}$ as in the Hartree-Fock procedure. In the above equation, we also know that

$$\frac{\delta T_s[n]}{\delta \phi_i^\dagger(\boldsymbol{r})} = -\frac{1}{2}\nabla^2 \phi_i(\boldsymbol{r}), \qquad \frac{\delta n(\boldsymbol{r})}{\delta \phi_i^\dagger(\boldsymbol{r})} = \phi_i(\boldsymbol{r}), \qquad \frac{\delta \langle U \rangle}{\delta n(\boldsymbol{r})} = U(\boldsymbol{r}). \tag{4.35}$$

Using the Lagrange multiplier method, we can obtain the Kohn-Sham Schrödinger-like equations, similar to (4.16), [175, Eq. 7.11]

$$H_{KS}\phi_i(\boldsymbol{r}) = \epsilon_i \phi_i, \tag{4.36}$$

with the Kohn-Sham Hamiltonian

$$\begin{aligned} H_{KS} = -\frac{1}{2}\nabla^2 + V_{KS}(\boldsymbol{r}) &= -\frac{1}{2}\nabla^2 + U(\boldsymbol{r}) + \frac{\delta E_{HF}[n]}{\delta n(\boldsymbol{r})} + \frac{\delta E_{xc}[n]}{\delta n(\boldsymbol{r})} \\ &= -\frac{1}{2}\nabla^2 + U(\boldsymbol{r}) + V_{HF}[n](\boldsymbol{r}) + V_{xc}[n](\boldsymbol{r}). \end{aligned} \tag{4.37}$$

Equation (4.36) is solved by a self-consistent procedure similar to the Hartree-Fock method Algorithm 16, and therefore has similar complexity, $O(N^3)$ if we use density

fitting techniques [253]. In this equation, the electron-electron potential functional is

$$V_{HF}[n](\boldsymbol{r}) := \frac{\delta E_{HF}[n]}{\delta n(\boldsymbol{r})} = \frac{1}{2}\int d\boldsymbol{r}' \frac{n(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|}, \tag{4.38}$$

and $V_{xc}[n]$ is a parametrized functional, taken of the form

$$V_{xc}[n](\boldsymbol{r}) = \frac{\delta E_{xc}[n](\boldsymbol{r})}{\delta n(\boldsymbol{r})}. \tag{4.39}$$

The exchange-correlation functional $E_{xc}[n](\boldsymbol{r})$ can be written in a similar way as

$$E_{xc}[n] = \int d\boldsymbol{r} n(\boldsymbol{r}) \epsilon_{xc}([n], \boldsymbol{r}), \tag{4.40}$$

with ϵ_{xc} an energy density per electron at point $\boldsymbol{r}$, which depends only on the electronic density. It is common to break up the correlation and exchange functionals:

$$\begin{aligned} V_{xc}[n](\boldsymbol{r}) &= \epsilon_{xc}[n](\boldsymbol{r}) + n(\boldsymbol{r})\frac{\partial \epsilon_{xc}[n](\boldsymbol{r})}{\partial n(\boldsymbol{r})} \\ &= \epsilon_x[n](\boldsymbol{r}) + n(\boldsymbol{r})\frac{\partial \epsilon_x[n](\boldsymbol{r})}{\partial n(\boldsymbol{r})} + \epsilon_c[n](\boldsymbol{r}) + n(\boldsymbol{r})\frac{\partial \epsilon_c[n](\boldsymbol{r})}{\partial n(\boldsymbol{r})}. \end{aligned} \tag{4.41}$$

There are many exchange-correlation functionals according to complexity or accuracy. The simplest is called local (spin) density approximation (LSDA), which depends uniquely on the (spin) density $n(\boldsymbol{r})$. For example, based on the uniform electron gas, Dirac proposed the Slater exchange functionals [88]

$$\epsilon_x^{\mathrm{LDA}}[n] = -c_x n^{1/3}(\boldsymbol{r}); \qquad \epsilon_x^{\mathrm{LSDA}}[n] = -2^{-1/3} c_x (n_\uparrow^{1/3}(\boldsymbol{r}) + n_\downarrow^{1/3}(\boldsymbol{r})), \tag{4.42}$$

for a constant c_x. A further step in complexity is the Generalized Gradient Approximation (GGA) functionals that depend on $\nabla n(\boldsymbol{r})$ too, such as [27]

$$\epsilon_x^{\mathrm{B88}} = \epsilon_x^{\mathrm{LDA}} - \frac{\beta n^{1/3}(\boldsymbol{r})}{1 + 6\beta x \sinh^{-1} x}, \qquad x = \frac{|\nabla n(\boldsymbol{r})|}{n^{4/3}(\boldsymbol{r})}; \tag{4.43}$$

or even Meta-GGA, with dependence on $\nabla^2 n(\boldsymbol{r})$.

The most accurate functionals currently in use are hybrid functionals introduced by Becke in 1993, [29]. Inspired by the adiabatic theorem, his idea was to assume one can use annealing over the Hamiltonian

$$H_\lambda = T + U(\lambda) + \lambda V, \tag{4.44}$$

where $U(\lambda)$ is adjusted to ensure that $n(\boldsymbol{r})$ is constant. At $\lambda = 0$ we get the non-interacting Hartree-Fock limit, while at $\lambda = 1$ we recover the interacting Hamiltonian. Annealing, we take

$$E_{xc}[n] = \int_0^1 d\lambda \, \langle \psi | V_{xc}(\lambda) | \psi \rangle \, . \tag{4.45}$$

We now make two approximations: first, that $V_{xc}(\lambda)$ is linear in λ, so

$$E_{xc}[n] \approx \frac{1}{2} \left(\langle \psi | V_{xc}(0) | \psi \rangle + \langle \psi | V_{xc}(1) | \psi \rangle \right) . \tag{4.46}$$

As mentioned, Becke assumed that $\langle \psi | V_{xc}(0) | \psi \rangle = E_x^{\mathrm{HF}}[n]$ is the Hartree-Fock exchange functional, and in the non-interacting limit the correlation is 0. His second approximation was to take $\langle \psi | V_{xc}(1) | \psi \rangle \approx E_{xc}^{\mathrm{LDA}}[n]$. Overall, the hybrid functional mixes Hartree-Fock and LDA functionals

$$E_{xc}^{\mathrm{Becke}}[n] = \frac{1}{2}(H_x^{HF}[n] + E_{xc}^{\mathrm{LDA}}[n]). \tag{4.47}$$

One may also choose GGA or meta-GGA functionals instead of LDA. This led to the B3LYP, the most widely used functional nowadays [227]

$$\epsilon_{xc}^{\mathrm{B3LYP}} = (1-a)\epsilon_x^{\mathrm{LSDA}} + aE_x^{\mathrm{HF}} + b\Delta\epsilon_x^{\mathrm{B88}} + (1-c)\epsilon_c^{\mathrm{VWN}} + c\epsilon_c^{\mathrm{LYP}}, \tag{4.48}$$

where a, b and c are constants, $\epsilon_c^{\mathrm{VWN}}$ is the correlation functional of [248] of type LSDA, $\epsilon_c^{\mathrm{LYP}}$ the correlation functional from [157] of type GGA, and $\Delta\epsilon_x^{\mathrm{B88}}$ the gradient correction of the B88 GGA exchange functional [27, 28]. Finally, it is worth mentioning that 'double hybrid' functionals exist, which include an MP2 (Møller-Plesset to second order [184]) term, or in other words, perturbation theory of the Hartree-Fock solution. There have also been attempts to use machine learning to model the exchange-correlation functional [139] in especially challenging situations. More information on density functional theory can be found in chapters 6-8 of Martin's book [175].

4.3.3. Coupled-Cluster

In the previous subsection, we have studied Density Functional Theory, which adds a parametrized exchange-correlation functional ansatz that should be able to capture energy corrections that the Hartree-Fock self-consistent procedure is not capable of obtaining. However, being an ansatz, even if in practice they can be very precise, they often come without much accuracy warranty. For this reason, we will come back to the wave function as a basic variable, and explain a post-Hartree-Fock method that is very

precise: the Coupled-Cluster method. As we will see, this method is also the basis for the Unitary Coupled-Cluster ansatz used for the Variational Quantum Eigensolver.

The basic idea of the Coupled Cluster procedure is to expand the wave function as excitations of the Hartree-Fock state $|\phi_0\rangle$:

$$|\phi_{\mathrm{CC}}\rangle = e^S |\phi_0\rangle , \tag{4.49}$$

for a normal operator S. If S consists of a sum T of fermionic excitation operators of occupied orbitals and annihilation of virtual orbitals, then this flavor of Coupled Cluster is called Traditional Coupled-Cluster:

$$T = \sum_k T_k, \qquad T_k = \frac{1}{(k!)^2} \sum_{i_1<\ldots<i_k}^{\mathrm{occ}} \sum_{a_1<\ldots<a_k}^{\mathrm{vir}} t_{i_1\ldots i_k}^{a_1\ldots a_k} a_{a_1}^\dagger \ldots a_{a_k}^\dagger a_{i_1} \ldots a_{i_k}, \tag{4.50}$$

with the a_{i_j} and $a_{i_j}^\dagger$ fermionic annihilation and creation operators, and $t_{i_1\ldots i_k}^{a_1\ldots a_k}$ the amplitudes we have to fix. We will also use the notation

$$\hat{\tau}_{i_1\ldots i_k}^{a_1\ldots a_k} = a_{a_1}^\dagger \ldots a_{a_k}^\dagger a_{i_1} \ldots a_{i_k}. \tag{4.51}$$

If we consider up to order η operators T_k, we will reconstruct the Full Configuration Interaction wave function, which takes into account all possible excitations from the Hartree-Fock Slater determinant. A clear advantage of Coupled-Cluster over other methods is its preservation of size consistency, meaning that if we have two disentangled subsystems, $|\phi_A\rangle$ and $|\phi_B\rangle$, the joint Coupled-Cluster exponential operator

$$e^{S_A+S_B} |\phi_A\rangle |\phi_B\rangle = e^{S_A} |\phi_A\rangle \otimes e^{S_B} |\phi_B\rangle , \tag{4.52}$$

and the energy of the joint system will be the sum of the energy of the subsystems.

There are two main ways to use Coupled-Cluster in chemistry problems. The first is the projective approach, based on

$$He^S |\phi_0\rangle = Ee^S |\phi_0\rangle . \tag{4.53}$$

We can project the previous equation into

$$\langle\phi_0| \hat{\tau}_{i_1\ldots i_k}^{a_1\ldots a_k} = \langle\phi_{i_1\ldots i_j}^{a_1\ldots a_j}| , \tag{4.54}$$

as well as into the Slater determinant $\langle\phi_0|$. From the projection, we get the following set of equations, called 'unlinked' [8]

$$\langle\phi_0|He^T|\phi_0\rangle = E\,\langle\phi_0|e^T|\phi_0\rangle = E, \tag{4.55}$$
$$\langle\phi_{i_1\ldots i_j}^{a_1\ldots a_j}|He^T|\phi_0\rangle = E\,\langle\phi_{i_1\ldots i_j}^{a_1\ldots a_j}|e^T|\phi_0\rangle\,. \tag{4.56}$$

We can expand the first expression

$$\begin{aligned}\langle\phi_0|H|\phi_{\mathrm{CC}}\rangle = &\underbrace{\langle\phi_0|H|\phi_0\rangle}_{E_{HF}} + \underbrace{\langle\phi_0|HT_1|\phi_0\rangle}_{0} + \underbrace{\langle\phi_0|H(T_2+\frac{1}{2}T_1^2)|\phi_0\rangle}_{\neq 0}\\ &+ \underbrace{\langle\phi_0|H(T_3+T_2T_1+\frac{1}{6}T_1^3)|\phi_0\rangle+\ldots}_{0}\end{aligned} \tag{4.57}$$

The fact that $\langle\phi_0|HT_1|\phi_0\rangle = 0$ is due to the Brillouin theorem. By the definition of the molecular orbitals in the Hartree-Fock procedure,

$$\langle\phi_0|HT_1|\phi_0\rangle = \sum_i\sum_a \langle\phi_0|H|\phi_i^a\rangle = \langle a|h|i\rangle + \sum_b \langle a|J_j - K_j|i\rangle\,, \tag{4.58}$$

which is an off-diagonal matrix element and, since the Hartree-Fock procedure precisely aims to diagonalize the Fock matrix on the basis of molecular orbitals, it vanishes. Operators with 3 or more also vanish due to Slater-Condon rules: the Hamiltonian operator can only de-excite up to 2 orbitals, so the molecular orbitals being orthonormal, this leads to a non-vanishing contribution from only the 0^{th} and 2^{nd} orders. While only T_1 and T_2 contribute to the energy, these operators might in turn depend on other excitation terms T_i for $i > 2$.[2]

An alternative set of equations is the set of 'linked' couple cluster equations:

$$\langle\phi_0|e^{-T}He^T|\phi_0\rangle = E, \tag{4.59}$$
$$\langle\phi_{i_1\ldots i_j}^{a_1\ldots a_j}|e^{-T}He^T|\phi_0\rangle = 0. \tag{4.60}$$

We can expand the similarity transformed Hamiltonian $\bar{H} = e^{-T}He^T$ as

$$\bar{H} = e^{-T}He^T = H + [H,T] + \frac{1}{2!}[[H,T],T] + \frac{1}{3!}[[[H,T],T],T] + \ldots \tag{4.61}$$

[2] A related and very useful result is known as Thouless theorem [239], which states that e^{T_1} maps a single Slater determinant state into another (non-orthogonal) single determinant. Thus e^{T_1} might be understood as a basis change. A didactic explanation can be found in this nice post by Joshua Goings, https://joshuagoings.com/2013/11/26/644/.

While for an arbitrary S the expansion may not converge, if we take it as T in equation (4.50), the series for the expected value is truncated at the fourth level:

$$\bar{H} = e^{T^\dagger} H e^T = H + [H,T] + \frac{1}{2!}[[H,T],T] + \frac{1}{3!}[[[H,T],T],T] + \frac{1}{4!}[[[[H,T],T],T],T]. \tag{4.62}$$

The reason for this is the generalized Wick theorem (see [217, Section 3.7 and Chapter 10]), which states that for any two normal operators A and B

$$[A,B] = AB - BA = (:AB: + :\overbrace{AB}:) - (:BA: + :\overbrace{BA}:), \tag{4.63}$$

where $:\overbrace{AB}:$ represents the sum of normal products where there is at least one contraction, and $:AB:$ the sum of normal ordered operators. In particular,

$$:a_i^\dagger a_j: \; = a_i^\dagger a_j, \qquad :a_j a_i^\dagger: \; = -a_i^\dagger a_j, \qquad \overbrace{a_i^\dagger a_j} = \delta_{ij}, , \qquad \overbrace{a_i a_j} = 0 = \overbrace{a_i^\dagger a_j^\dagger}. \tag{4.64}$$

Since there is an even number of creation and annihilation operators in each τ operator [217], $:AB: \; =: BA:$. Then, (4.63) tells us that,

$$:AB: \; =: BA: \quad \Rightarrow \quad [A,B] =: \overbrace{AB}: - :\overbrace{BA}: . \tag{4.65}$$

In other words, in our problem the commutator $[A,B]$ only depends on sums of products with at least one contraction. We also know that the different terms in any operator T_k only contain creation operators of virtual orbitals and annihilation of occupied ones. Consequently, all contractions of operators $\hat{\tau}_{i_1 \ldots i_k}^{a_1 \ldots a_k}$ are 0, and thus they commute with one another as expected. Finally, since each electronic Hamiltonian term contains up to 4 annihilation/creation operators, from the fifth order in (4.61) there will be no annihilation/creation operators in the Hamiltonian to contract, making the corresponding $:\overbrace{AB}: \; = 0$ and leading to (4.62).

Furthermore, given T_k defined as in (4.50),

$$\langle \phi_0 | \, T_i = 0, \tag{4.66}$$

because T_i would apply creation operators to occupied orbitals, and annihilation operators to virtual ones. Taking into account that the Hamiltonian H can only de-excite up to two orbitals, we use the same Slater-Condon rule and Brillouin theorem

as above, to obtain

$$E = E_{HF} + \langle\phi_0|[H, T_2]|\phi_0\rangle + \frac{1}{2}\langle\phi_0|[[H, T_1], T_1]|\phi_0\rangle\,. \tag{4.67}$$

Either the 'linked' or 'unlinked' set of equations can then be solved algebraically.

Finally, it is worth mentioning that projective methods are not the only flavor of Coupled-Cluster, we also have variational methods which involve minimizing

$$E = \frac{\langle\phi_0|e^{S^\dagger}He^S|\phi_0\rangle}{\langle\phi_0|e^{S^\dagger}e^S|\phi_0\rangle}. \tag{4.68}$$

One advantage of this variational approach over the projective approach is that the recovered energy is always necessarily an upper bound of the ground-state energy. However, it is not possible to apply the same truncation that we used for the similarity transformed Hamiltonian $\bar{H}$, notice the different sign in the $e^{S^\dagger}$ exponential. Additionally, if S is not anti-Hermitian (e.g. $S = T$), the denominator might also be nontrivial. To ensure S is indeed Hermitian, we might instead choose $S = T - T^\dagger$, which leads to Unitary Coupled-Cluster. We will see that this is a very natural ansatz to implement in a quantum computer.

4.4. Hamiltonian simulation

In the previous section, we analyzed three classical algorithms to solve problems in quantum chemistry. We now focus on quantum algorithms. As mentioned at the beginning of the chapter, quantum computing offers two features that make it very attractive for chemistry and material science applications. First, it is capable of representing quantum systems exactly, without approximations. Second, the unitary evolution of closed quantum systems can be naturally implemented in a quantum computer. In this section, we review the problem of how to implement this evolution, under the name of Hamiltonian simulation.

Hamiltonian simulation is a key technique to solve the two most important problems in quantum chemistry: preparing the ground state of a Hamiltonian and computing its energy. Other applications include preparing and analyzing thermal and excited states. To see what we mean by Hamiltonian simulation, we start from the Schrödinger equation,

$$H\psi = i\hbar\frac{d}{dt}\psi. \tag{4.69}$$

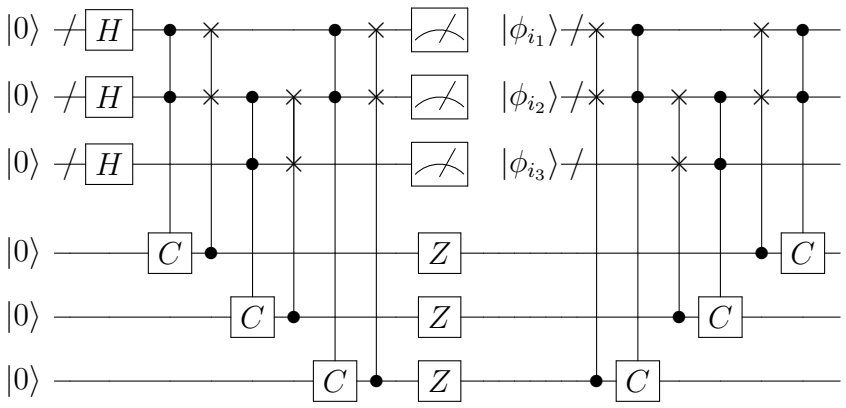


Figure 4.1 **Antisymmetrization circuit.** Example of an antisymmetrization circuit for three electrons. The operation C represents a comparison test controlled on the two registers that are being compared. The seed register (top left) is measured to post-select on the collision-free subspace. The Z gates perform the phase flip when swapping two registers. At the end of the circuit, the auxiliary record qubits (bottom register) can be discarded as they are disentangled [36]. This circuit can be extended to an arbitrary number of electrons η by increasing the size of the sorting network and adding additional auxiliary qubits for each required comparison and swap.

As our electronic Hamiltonian (4.2) is time-independent, the evolution of a quantum state ψ can be modeled as

$$\psi(x,0) = \sum_n a_n \psi_{E_n}(x) \Rightarrow \psi(x,t) = \sum_n a_n e^{-iE_n t/\hbar} \psi_{E_n}(x), \tag{4.70}$$

where a_n are the amplitudes, ψ_{E_n} the eigenstates of the Hamiltonian H with eigenvalues E_n. From this equation, it is clear that if $|a_0| \approx 1$ and we know how to implement the Hamiltonian simulation of $e^{-iHt/\hbar}$, we can use the phase estimation algorithm introduced in Chapter 3 to find E_0, the ground state energy.

4.4.1. Quantization and fermion to qubit mapping

First quantization. Before getting into details of how to implement Hamiltonian simulation, however, we need to decide how to represent the quantum state and Hamiltonian. The first such choice is whether to represent the state in first or second quantization, in other words, whether the logical qubits of our system will represent the state of each electron, or the occupancy of each orbital, respectively. For example, if we have η disentangled electrons in our system, occupying orbitals $i_1, \ldots, i_\eta$, then

the corresponding first quantized quantum state is

$$\sum_{\sigma\in S_\eta} \frac{(-1)^{\pi(\sigma)}}{\sqrt{\eta!}} \left|\phi_{\sigma(i_1)}\right\rangle \otimes \ldots \otimes \left|\phi_{\sigma(i_\eta)}\right\rangle\rangle \,. \tag{4.71}$$

For N orbitals, we need $\eta \log_2 N$ qubits to represent the state. Note how this state is antisymmetric, what can be achieved from $\left|\phi_{i_1}\right\rangle \left|\phi_{i_2}\right\rangle \ldots \left|\phi_{i_\eta}\right\rangle$ by a procedure first described in [36], and depicted in Fig. 4.1. This protocol can be carried out with minimal overhead, and fortunately only needs to be implemented once during the state preparation, because the Hamiltonian simulation operators will preserve antisymmetry throughout the algorithm.

In contrast, a second quantized state does not need to be explicitly antisymmetrized, because the creation and annihilation operators account for Fermi statistics automatically, but in contrast needs N qubits for N orbitals. If $N \gg \eta$, this implies that the second quantization requires more qubits to represent the state. Moreover, since we have to replicate Fermi statistics, the mapping from fermions to qubits is non-trivial: we have to keep track of occupation numbers as well as the parity.

Jordan-Wigner mapping. The Jordan-Wigner mapping represents [258]

$$a_j^\dagger \to Z_1 \otimes \ldots \otimes Z_{j-1} \otimes (\sigma^+)_j \otimes \mathbf{1}, \tag{4.72}$$

for X_j, Y_j, Z_j the Pauli operators acting on qubit j, and $\sigma^+ = (X + iY)/2$. This means that operator $a_j^\dagger$ acts non-trivially on j qubits: the first $j-1$ qubit Pauli gates record the phase, and the last Pauli operator creates a particle in qubit j. Using this mapping, a Hermitian operator $a_p^\dagger a_q + a_q^\dagger a_p$ is implemented as [18]

$$a_p^\dagger a_q + a_q^\dagger a_p \mapsto \begin{cases} \mathbf{1} - Z & p = q, \\ \frac{X_p \vec{Z} X_q + Y_p \vec{Z} Y_q}{2} & p \neq q, \end{cases} \tag{4.73}$$

where $X_p \vec{Z} X_q = X_p \otimes Z_{p+1} \otimes \ldots \otimes Z_{q-1} \otimes X_q$ with $p < q$, and similarly for $Y_p \vec{Z} Y_q$. In summary, the Jordan-Wigner mapping implies acting on up to $O(N)$ qubits per fermionic Hamiltonian operator. Extending this mapping to dimensions higher than one has been discussed in Ref. [246].

Parity mapping. The dual of the Jordan-Wigner mapping is the parity mapping, where the parity is saved in a single qubit, but the occupation number in $O(N)$ [61],

$$a_j^\dagger \to \mathbf{1} \otimes Z_{j-1} \otimes (\sigma^+)_j \otimes X_{j+1} \otimes \ldots \otimes X_N. \tag{4.74}$$

The mapping from Jordan-Wigner to parity mapping encoded states is

$$|i_1\rangle \otimes |i_2\rangle \otimes \ldots |i_N\rangle \to |i_1\rangle \otimes |i_1 \oplus i_2\rangle \otimes \ldots \otimes |\bigoplus_{j=1}^{N} i_j\rangle\,, \tag{4.75}$$

with $\oplus$ denoting binary sum.

Bravyi-Kitaev mapping. Is there a way to avoid the $O(N)$ cost of the Jordan-Wigner and parity mappings? The Bravyi-Kitaev encoding ensures that both parity and occupation require no more than $O(\log_2 N)$ qubit operators [52]. First proposed for $N = 2^n$ a power of two, it uses qubits to encode sums of occupation numbers. The mapping from the Jordan-Wigner mapping to the Bravyi-Kitaev matrix is carried out by the basis change [61]

$$|i_1\rangle \otimes |i_2\rangle \otimes \ldots |i_N\rangle \to |b_1\rangle \otimes |b_2\rangle \otimes \ldots |b_N\rangle\,, \qquad b_k = \sum_{l=1}^{k} [\beta_n]_k^l i_l, \tag{4.76}$$

where $\beta_0 = (1)$ and [215]

$$\beta_n = \left(\begin{array}{c|c} & \leftarrow 1 \rightarrow \\ \beta_{n-1} & \mathbf{0} \\ \hline & \\ \mathbf{0} & \beta_{n-1} \end{array}\right). \tag{4.77}$$

For example,

$$\beta_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{4.78}$$

We can check that each column of these matrices contains up to $n = \log_2 N$ 1s, which leads to $O(\log_2 N)$ weight creation and annihilation operators [215]. In fact, when N is a power of 2, operators $a_j^\dagger$ have weight no larger than $\log_2 N$ [121]. In exchange for faster simulations, the detailed explanation of the form of creation and annihilation operators becomes complicated and would require a large explanation, which the interested reader may instead find in Ref. [215].

For the sake of completeness, we mention that a closely related technique, called the Bravyi-Kitaev tree method, also achieves $O(\log_2 N)$ weight for the operators. While in practice this technique retrieves higher weight operators than the Bravyi-Kitaev method, it also allows reducing the number of qubits if N is not a power of 2 [121].[3] Other encodings include Ref. [17] based on the configuration-interaction matrix, Steudtner's method for which the number of gates to implement $a^\dagger$ and a is independent on the number of basis functions [229, 228], and Ref. [138], whose complexity on both the number of gates and qubits is polylogarithmic in the number of basis functions. Refs. [87, 86] suggest a particularly compact mapping in the number of fermionic modes (basis functions) per qubit, and finally the proposal of Ref. [254] proposes using auxiliary qubits to restrict the mapping to local operators.

4.4.2. Basis choice

In addition to first or second quantization and the mapping to qubits, we have to choose a set of basis functions to generate the wave function. The two most common options are Gaussian functions and plane waves.

[3] An implementation of Jordan-Wigner and Bravyi-Kitaev operators can be found in one tutorial on the topic in the Openfermion library.

Gaussian functions. Gaussian functions aim to generalize the well-known hydrogen atom orbitals

$$\phi_{n,l,m}(r,\theta,\phi) = R_{nl}(r)Y_{l,m}(\theta,\phi), \tag{4.79}$$

where $Y_{l,m}(\theta,\phi)$ are spherical harmonics, and the radial component is a product of Laguerre polynomials and a negative exponential term in the radius [108]. The closest choice of orbitals to emulate such behavior are Slater-type orbitals (STO),

$$R_n^{\mathrm{STO}}(r) = Nr^{n-1}e^{-\zeta r}, \tag{4.80}$$

for N a normalizing term and ζ a diffuseness parameter [61]. While Slater-type orbitals can be used as a basis and have the advantage of correct asymptotic behavior, computing the corresponding Hamiltonian integrals we saw in Section 4.3.1 must be done numerically because no analytical solutions are known. Instead, for simplicity Gaussian-type orbitals (GTO) are often used,

$$R_n^{\mathrm{GTO}}(r) = Nr^{n-1}e^{-\zeta r^2}, \tag{4.81}$$

which exhibit Gaussian behavior and are easily analytically integrable as a result. Gaussian-type orbitals can be written in spherical coordinates, with spherical harmonics modeling the angular component, or in Cartesian coordinates, with a factor $x^i y^j z^k$ instead of the spherical harmonics, $\vec{r} = (x,y,z)$ and $i+j+k$ the angular momentum.

Since Gaussian-type orbitals do not exhibit the correct asymptotic behavior in the exponent, often a linear combination of Gaussian orbitals (*primitives*) is used to emulate a single Slater-type orbital (*contraction*). Then, the Hartree-Fock or Density Functional Theory procedures will compute a linear combination of the contracted atomic orbitals to form molecular orbitals. A review of the different families of contractions of GTOs can be found in Ref. [125]. On this basis, the Hamiltonian is usually written as

$$H = \sum_{\alpha,\beta\in\{\uparrow,\downarrow\}} \sum_{p,q,r,s=1}^{N/2} h_{pqrs} a^\dagger_{p,\alpha} a^\dagger_{q,\beta} a_{r,\beta} a_{s,\alpha} + \sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p,r=1}^{N/2} h_{p,r} a^\dagger_{\sigma,p} a_{\sigma,r}, \tag{4.82}$$

with one and two body coefficients,

$$h_{pr} = \langle p|T+U|r\rangle\,, \qquad h_{pqrs} = \langle p,q|V|r,s\rangle\,. \tag{4.83}$$

Gaussian functions are well suited for isolated molecules, where the wave function exponentially vanishes at infinity.

Plane waves. The alternative to Gaussian basis functions are plane waves which, due to their periodicity, are better suited for periodic materials. If we consider a cell of volume Ω, these functions are[4]

$$\varphi_p(\boldsymbol{r}) = \sqrt{\frac{1}{\Omega}} e^{i\boldsymbol{G}_p \cdot \boldsymbol{r}}, \tag{4.86}$$

where $\boldsymbol{G}_p$ is a reciprocal lattice vector indexed by $\boldsymbol{p} \in \mathcal{G} = \left[-\frac{N^{1/3}}{2} + 1, \frac{N^{1/3}}{2} - 1\right]^3$.

On this basis, the Hamiltonian takes the form [21]

$$T = \sum_{i=1}^{\eta} \sum_{p \in \mathcal{G}} \frac{\|\boldsymbol{G}_p\|^2}{2} |\boldsymbol{p}\rangle \langle \boldsymbol{p}|_i \,, \tag{4.87}$$

$$U = -\frac{4\pi}{\Omega} \sum_{i=1}^{\eta} \sum_{q \in \mathcal{G}} \sum_{\substack{\nu \in \mathcal{G}_0 \\ (\boldsymbol{q}-\boldsymbol{\nu}) \in \mathcal{G}}} \frac{\sum_{I=1}^{L} Z_I e^{i\boldsymbol{G}_\nu \cdot \boldsymbol{R}_I}}{\|\boldsymbol{G}_\nu\|^2} |\boldsymbol{q}-\boldsymbol{\nu}\rangle \langle \boldsymbol{q}|_i \,, \tag{4.88}$$

$$V = \frac{2\pi}{\Omega} \sum_{i \neq j}^{\eta} \sum_{p,q \in \mathcal{G}} \sum_{\substack{\nu \in \mathcal{G}_0 \\ (\boldsymbol{p}+\boldsymbol{\nu}) \in \mathcal{G} \\ (\boldsymbol{q}-\boldsymbol{\nu}) \in \mathcal{G}}} \frac{1}{\|\boldsymbol{G}_\nu\|^2} |\boldsymbol{p}+\boldsymbol{\nu}\rangle \langle \boldsymbol{p}|_i \otimes |\boldsymbol{q}-\boldsymbol{\nu}\rangle \langle \boldsymbol{q}|_j \,, \tag{4.89}$$

where $\mathcal{G}_0 := \mathcal{G} \backslash \{(0,0,0)\}$. Note the approximation, called aliasing or dualling, of making the momentum exchange be confined to lie within $\mathcal{G}_0$ [207, 179].

Similarly, on the plane-wave basis and second quantization, the Hamiltonian will take the form [21, Appendix B]

$$H = \underbrace{\frac{2\pi}{\Omega} \sum_{\substack{(p,\sigma) \neq (q,\sigma') \\ \nu \neq 0}} \frac{c^\dagger_{p,\sigma} c^\dagger_{q,\sigma'} c_{q+\nu,\sigma'} c_{p-\nu,\sigma}}{\|\boldsymbol{G}_\nu\|^2}}_{V} + \underbrace{\frac{1}{2} \sum_{p,\sigma} \|\boldsymbol{G}_p\|^2 c^\dagger_{p,\sigma} c_{p,\sigma}}_{T} - \underbrace{\frac{4\pi}{\Omega} \sum_{\substack{p \neq q; \\ I,\sigma}} \left(Z_I \frac{e^{i\boldsymbol{G}_{q-p} \cdot \boldsymbol{R}_I}}{\|\boldsymbol{G}_{p-q}\|^2} \right) c^\dagger_{p,\sigma} c_{q,\sigma}}_{U} \,. \tag{4.90}$$

[4] More generally, the previous equation assumes a single k-point in the center of the Brillouin zone, the dual of the primitive unit cell, with unit vectors $\boldsymbol{b}_i$. For any other k-point, the wave functions take the form

$$\varphi_{p,k}(\boldsymbol{r}) = \sqrt{\frac{1}{\Omega}} e^{i(\boldsymbol{G}_p + \boldsymbol{k}) \cdot \boldsymbol{r}}, \tag{4.84}$$

where

$$\boldsymbol{k} = \sum_{i=1}^{3} \frac{n_i}{N_i} \boldsymbol{b}_i, \tag{4.85}$$

N_i indicates the number of plane waves in each dimension, and n_i is an integer. This is connected to Bloch's theorem [40], which states that $\psi(\boldsymbol{r}) = e^{i\boldsymbol{k}\cdot\boldsymbol{r}} u(\boldsymbol{r})$, where the potential $u(\boldsymbol{r})$ has the periodicity of the lattice. Adding more k-points is used to take into account correlations between electrons and nuclei of different unit cells, growing the system towards the thermodynamic limit.

One useful feature of plane waves is that they diagonalize the kinetic operator. In contrast, we can Fourier transform the Hamiltonian, to obtain the dual basis Hamiltonian [21, Appendix C]

$$H = \underbrace{\frac{1}{2N}\sum_{p,q,\nu,\sigma} \|\boldsymbol{G}_\nu\|^2 \cos[\boldsymbol{G}_\nu \cdot \boldsymbol{r}_{q-p}] a^\dagger_{p,\sigma} a_{q,\sigma}}_{T} \\ \underbrace{-\frac{4\pi}{\Omega}\sum_{\substack{p,I,\sigma \\ \nu\neq 0}} \left(\frac{Z_I \cos[\boldsymbol{G}_\nu \cdot (\boldsymbol{R}_I - \boldsymbol{r}_p)]}{\|\boldsymbol{G}_\nu\|^2}\right) n_{p,\sigma}}_{U} + \underbrace{\frac{2\pi}{\Omega}\sum_{\substack{(p,\sigma)\neq(q,\sigma') \\ \nu\neq 0}} \frac{\cos[\boldsymbol{G}_\nu \cdot \boldsymbol{r}_{p-q}]}{\|\boldsymbol{G}_\nu\|^2} n_{p,\sigma} n_{q,\sigma'}}_{V}, \tag{4.91}$$

in which the external potential and potential operators will be diagonal. The number of terms in the Hamiltonian in plane waves scales as $O(N^3)$ with the number of basis functions N, while in dual wave basis they scale as $O(N^2)$. This is more favorable than the $O(N^4)$ scaling in (4.82). However, this comparison is not fair since the basis used is different, and therefore N is not the same either. To compare them, we should analyze the error behavior of both basis sets, which in both cases will asymptotically decrease as $O(1/N)$ [155, 218]. This, and a thorough literature review, leads Ref. [21, Appendix E] to state that one needs approximately 10 to 20 times more plane waves than Gaussian basis functions to model a system to the same accuracy, as long as the system is periodic. This comparison, however, must be taken with care.

Finally, it is also possible to simulate isolated molecules in plane waves, by taking an 8Ω volume cell, and ensuring that at a distance larger than $D = \sqrt[3]{\Omega}$ the Coulomb interaction vanishes. This means that instead of a Fourier amplitude proportional to $\frac{4\pi}{\|\boldsymbol{G}_\nu\|^2}$, we get

$$4\pi \frac{1 - \cos[|\boldsymbol{G}_\nu| D]}{\|\boldsymbol{G}_\nu\|^2} \tag{4.92}$$

in (4.87), (4.88) and (4.89) [21, Appendix E]. In the dual basis, the approximation is even simpler, just dropping all terms $n_p n_q$ for which $|\boldsymbol{r}_p - \boldsymbol{r}_q| > D$.

Bloch and Wannier basis functions. Other basis functions have also been used throughout the literature including *Bloch wave basis* (also known as *band fermion basis*), whose defining feature is to diagonalize the one-body Hamiltonian in periodic

materials [76]. Mathematically, if the one-body Hamiltonian is

$$T+U=\sum_{\boldsymbol{k},p,q,\sigma}\underbrace{\left[\frac{|\hbar(\boldsymbol{k}+\boldsymbol{G}_p)|^2}{2m}\delta_{\boldsymbol{G}_p,\boldsymbol{G}_q}+U_{\boldsymbol{G}_p-\boldsymbol{G}_q}\right]}_{h_{\boldsymbol{k},\boldsymbol{G}_p-\boldsymbol{G}_q}}c^\dagger_{\boldsymbol{k}+\boldsymbol{G}_p,\sigma}c_{\boldsymbol{k}+\boldsymbol{G}_q,\sigma}. \tag{4.93}$$

Choosing, as we did on the plane wave basis explanation, a single crystal cell $\boldsymbol{k}$, we find can find the unitary transformation $S_{n,\boldsymbol{G}}(\boldsymbol{k})$ that diagonalizes the one-body Hamiltonian

$$h_{\boldsymbol{k},\boldsymbol{G}_p-\boldsymbol{G}_q}=\sum_n S^\dagger_{\boldsymbol{G}_p,n}(\boldsymbol{k})\epsilon_n(\boldsymbol{k})S_{n,\boldsymbol{G}_q}(\boldsymbol{k}). \tag{4.94}$$

Index n will indicate an occupation band, and can take as many values as can reciprocal vector lattices $\boldsymbol{G}$. Then, the Bloch basis functions are defined as [76]

$$\phi_{\boldsymbol{k},n}(\boldsymbol{r})=\sqrt{\frac{1}{\Omega}}e^{i\boldsymbol{k}\cdot\boldsymbol{r}}\sum_{\boldsymbol{G}}e^{i\boldsymbol{G}\cdot\boldsymbol{r}}S_{n,\boldsymbol{G}}(\boldsymbol{k}). \tag{4.95}$$

If we are interested in real-space basis functions, we can modify Bloch basis functions into *Wannier basis functions*. In their simplest form, they can be written as [250]

$$W_{\boldsymbol{R}}(\boldsymbol{r})=W_0(\boldsymbol{r}-\boldsymbol{R})=\sum_{\boldsymbol{k},n}e^{-i\boldsymbol{k}\cdot\boldsymbol{R}}\phi_{\boldsymbol{k},n}(\boldsymbol{r})=\sum_{\boldsymbol{k},n}\phi_{\boldsymbol{k},n}(\boldsymbol{r}-\boldsymbol{R}). \tag{4.96}$$

More complex options may include a gauge transformation of the Bloch basis.

4.4.3. Hamiltonian simulation techniques

In this subsection, we review the main techniques of Hamiltonian simulation, that is, of implementing e^{-iHt}. This will be important as a subroutine in quantum phase estimation and state preparation, as we will see. Since we are interested in Hamiltonian simulating the electronic Hamiltonian (4.2), we will assume the Hamiltonian is a Linear Combination of Unitaries from now on, $H=\sum_{l=1}^{L}a_lH_l$, with $a_l>0$.

Trotter. The Trotter-Suzuki decomposition was the first method proposed to implement Hamiltonian simulation [195] and does not require the H_l above to be unitary but only Hermitian. The key idea is to decompose $e^{-it\sum_l a_lH_l}$ as a product of terms of the form e^{-itH_l}. Since this induces an error dependent on the time length of each time segment, we divide the total time simulation into many small segments. For example,

the first-order Trotter formula is

$$e^{-iHt} = \underbrace{\left(\prod_l e^{-ia_l H_l t/r}\right)^r}_{\mathcal{S}_1(H;t/r)} + O\left(\sum |[H_{l_1}, H_{l_2}]| t^2/r\right). \tag{4.97}$$

The error will also multiplicatively depend on the commutator of the different H_l terms, such that if for example they all commute, the Hamiltonian simulation can be fast-forwarded, e.g., implemented in a single time segment. In other words, in such a case, we can take t as large as wished and $r = 1$ still get an accurate result.

Similarly, we can propose a second-order formula to decrease the simulation error,

$$e^{-iHt} = \underbrace{\left(\left(\prod_{l=1}^{L} e^{-ia_l H_l t/2r}\right)\left(\prod_{l=L}^{1} e^{-ia_l H_l t/2r}\right)\right)^r}_{\mathcal{S}_2(H;t/r)} + O\left(\sum |[[H_{l_1}, H_{l_2}], H_{l_3}]| t^3/r^2\right), \tag{4.98}$$

and inductively [233, 72, 232],

$$\mathcal{S}_{2k}(H;t/r) = \mathcal{S}_{2k-2}^2(H;p_k t/r)\mathcal{S}_{2k-2}(H;(1-4p_k)t/r)\mathcal{S}_{2k-2}^2(H;p_k t/r), \tag{4.99}$$

with $p_k = 1/(4 - 4^{1/(2k-1)})$. The Hamiltonian simulation error ϵ_{HS} will decrease as [178]

$$\left\| e^{-iHt/r} - \mathcal{S}_k(H;t/r) \right\|_2 \leq W_k \left(\frac{t}{r}\right)^{k+1} \leq \frac{\epsilon_{HS}}{r}, \tag{4.100}$$

$$W_k = O\left(\max_{\boldsymbol{i}} [[\ldots[H_{l_{i_1}}, H_{l_{i_2}}], H_{l_{i_3}}], \ldots], H_{l_{i_{k+1}}}]\right). \tag{4.101}$$

While the complexity of the method is in any case polynomial on the Hamiltonian simulation error ϵ_{HS}, some techniques can help reduce it. The first is to introduce randomness either in the ordering of the $a_l H_l$ terms [72], or treat the a_l/λ for $\lambda = \sum a_l$ as probabilities of applying H_l for fixed amounts of time [58]. Other randomization protocols have also been explored [249].

The second line of research aims to bound to the norm of the commutators [143, 59, 232, 178]. For $N \gg \eta$, the so-called SHC-bound [232, 178] scales as $O(N^3)$, while for N closer to η tighter bounds can be found in Ref. [178]. Some of these use the fermionic semi-norm

$$\|X\|_\eta := \max_{\phi,\psi} |\langle\phi|X|\psi\rangle|_\eta, \tag{4.102}$$

where $|\phi\rangle$ and $|\psi\rangle$ contain η fermions. This semi-norm can be used to eliminate nonphysical terms, which lead to higher error than possible in the commutators, due to the particle-conserving nature of our system.

Taylor series. While Trotter series decomposition is popular and flexible, its error scales only inverse polynomially. Post-Trotter methods aim to improve this situation with a polylogarithmic error complexity. One possibility is Taylor series decomposition:

$$U_r = e^{-iHt/r} \approx \sum_{k=0}^{K} \frac{1}{k!}(-iHt/r)^k = \sum_{k=0}^{K} \sum_{l_1,\ldots,l_k=1}^{L} \underbrace{\frac{(-it/r)^k}{k!} a_{l_1}\ldots a_{l_k}}_{b_j} \underbrace{H_{l_1}\ldots H_{l_k}}_{U_j}. \tag{4.103}$$

To implement this LCU decomposition $\sum_j b_j U_j$, we use the same Prep and Sel operators that we have defined on other occasions:

$$\text{Prep} : |0\rangle \mapsto \sum_j \sqrt{b_j}\,|j\rangle\,, \qquad \text{Sel} : |j\rangle\,|\psi\rangle \mapsto |j\rangle\,U_j\,|\psi\rangle\,, \tag{4.104}$$

Then, we can perform $U_{LCU}^{Tay} = (\text{Prep}^\dagger \otimes \mathbf{1})\text{Sel}(\text{Prep} \otimes \mathbf{1})$, which exhibits some failure probability, as the action of $\text{Prep}^\dagger$ does not fully return the value of the auxiliary register $|j\rangle$ to $|0\rangle$. Consequently, it is convenient to implement the algorithm in short time segments of order $\tau = \ln 2$, so that (oblivious) amplitude amplification can eliminate such error [34]. Overall, the value of K can be chosen to be $\left\lceil -1 + 2\frac{\log(\epsilon_{HS}/r)}{\log\log(\epsilon_{HS}/r)+1} \right\rceil$ [169, Lemma 5].

Qubitization. Since the Hamiltonian is already written as a linear combination of unitaries, we can instead describe $U_{LCU} = (\text{Prep}^\dagger \otimes \mathbf{1})\text{Sel}(\text{Prep} \otimes \mathbf{1})$ with different prepare and select operators

$$\text{Prepare} : |0\rangle \mapsto \sum_l \sqrt{a_l}\,|l\rangle\,, \qquad \text{Select} : |l\rangle\,|\psi\rangle \mapsto |l\rangle\,H_l\,|\psi\rangle\,. \tag{4.105}$$

The matrix form of U_{LCU} is

$$U_{LCU} = \begin{pmatrix} H/\lambda & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{4.106}$$

a *block encoding* as we explained in Section 3.3.3. This operator can also be studied by its action on state $|0\rangle\,|\psi\rangle$,

$$U_{LCU}\,|0\rangle\,|\psi\rangle = |0\rangle\,\frac{H}{\lambda}\,|\psi\rangle + \sqrt{1 - \frac{\|H\,|\psi\rangle\,\|}{\lambda}}\,|1\rangle\,|\psi^\perp\rangle\,. \tag{4.107}$$

While this operator also has some probability of failure that requires amplitude amplification, an alternative is to use Prep and Sel as part of a quantum walk operator Q, acting in the same way as U_{LCU} in the previous equation

$$\begin{aligned} Q\left|0\right\rangle\left|\psi_k\right\rangle &= \cos(\theta_k)\left|0\right\rangle\left|\psi_k\right\rangle - \sin(\theta_k)\left|1\right\rangle\left|\psi_k^{\perp}\right\rangle, \\ Q\left|1\right\rangle\left|\psi_k^{\perp}\right\rangle &= \cos(\theta_k)\left|1\right\rangle\left|\psi_k^{\perp}\right\rangle + \sin(\theta_k)\left|0\right\rangle\left|\psi_k\right\rangle, \end{aligned} \tag{4.108}$$

for $\cos\theta_k = \frac{E_k}{\lambda}$ and $(\left|\psi_k\right\rangle, E_k)$ eigenvector and eigenvalue tuples. Notice that θ_k here plays the role of 2θ in (2.15) or $2\varphi_j$ in (2.42). In other words,

$$Q = \bigoplus_k \begin{pmatrix} \frac{E_k}{\lambda} & -\sqrt{1-\frac{E_k^2}{\lambda^2}} \\ \sqrt{1-\frac{E_k^2}{\lambda^2}} & \frac{E_k}{\lambda} \end{pmatrix}_k = \bigoplus_k e^{-iY_k\theta_k}. \tag{4.109}$$

To build operator Q, Ref. [169, Corollary 9] suggests using

$$Q = \underbrace{\text{Prep}(2\left|0\right\rangle\left\langle 0\right| \otimes \mathbf{1} - \mathbf{1})\text{Prep}^{\dagger}}_{\text{Rotation 1}} \cdot \underbrace{\text{Sel}}_{\text{Rotation 2}} \tag{4.110}$$

whenever $U_{LCU}^2 = 1$, as is the case here. Diagonalizing it, we get

$$Q = \bigoplus_k \left(e^{i\theta_k}\left|\theta_k\right\rangle\left\langle\theta_k\right| + e^{-i\theta_k}\left|-\theta_k\right\rangle\left\langle-\theta_k\right|\right). \tag{4.111}$$

Provided eigenstate $\left|\psi_0\right\rangle$, performing quantum phase estimation directly over Q is sufficient to recover $\pm\theta_0$, which allows computing the energy as $E_0 = \lambda\cos(\pm\theta_0)$ with no Hamiltonian simulation error [36].

This protocol, called *qubitization*, is even more powerful when combined with a technique called *quantum signal processing* [168], which we have mentioned previously. Introducing

$$Z_\phi := (1+e^{-i\phi})\text{Prep}\left|0\right\rangle\left\langle 0\right|\text{Prep}^{\dagger} - \mathbf{1} = \bigoplus_k \begin{pmatrix} e^{-i\phi} & 0 \\ 0 & 1 \end{pmatrix}_k, \tag{4.112}$$

one can form operators

$$W_\phi = Z_{\phi-\pi/2}QZ_{-\phi+\pi/2} = \bigoplus_k \begin{pmatrix} -ie^{-i\phi} & 0 \\ 0 & 1 \end{pmatrix}_k \begin{pmatrix} \frac{E_k}{\lambda} & -\sqrt{1-\frac{E_k^2}{\lambda^2}} \\ \sqrt{1-\frac{E_k^2}{\lambda^2}} & \frac{E_k}{\lambda} \end{pmatrix}_k \begin{pmatrix} ie^{-i\phi} & 0 \\ 0 & 1 \end{pmatrix}_k. \tag{4.113}$$

A string of these, $W_{\vec{\phi}} = W_{\phi_Q} \dots W_{\phi_1}$, can be used to synthesize $A[H] + iB[H]$, where A and B are polynomials of degree Q (or $Q/2$) of $\cos(\theta_k/2)$ and A and B have equal [169, Theorem 3] (or respectively opposite, [169, Theorem 4]) parity. For example, this quantum signal processing technique can be used to simulate e^{-iHt} with cost polylogarithmic in the precision parameter, by decomposing it into a Jacobi-Anger series [169, Theorem 1]. Furthermore, quantum signal processing is Hamiltonian-query optimal, and is not restricted to Linear Combination of Unitaries but can also be used with other oracle access models such as sparse or density matrices.

Interaction picture. While using qubitization we can perform errorless Hamiltonian simulation suitable for phase estimation, the algorithm still bears a linear dependence on the one-norm of the Hamiltonian λ. Implementing the simulation in the interaction picture aims to reduce such a complexity factor. In particular, if $H = A + B$, one can form the interaction picture Hamiltonian

$$H_I(t) = e^{iAt} B(t) e^{-iAt}, \tag{4.114}$$

and if $\|A\| \gg \|B\|$, this will reduce the norm of the Hamiltonian from $\|A+B\|$ to $\|B\|$. In this framework, the state will evolve as [171]

$$|\psi(t)\rangle = e^{-iAt} \mathcal{T} \left[e^{-i \int_0^t H(s) ds} \right] |\psi(0)\rangle \,. \tag{4.115}$$

In summary, we have to implement two parts, e^{-iAt} and the time-ordered exponential [136, 171]. The former might be easy to implement if all operators in A commute, so this is a practical requirement for the interaction picture algorithm. The latter requires a Dyson series expansion

$$U(t) = \mathcal{T} \left[e^{-i \int_0^t H(s) ds} \right] = \sum_{k=0}^{\infty} (-i)^k D_k, \qquad D_k = \frac{1}{k!} \int_0^t \dots \int_0^t \mathcal{T}[H(t_k) \dots H(t_1)] d^k t, \tag{4.116}$$

which, similarly to the Taylor series, has logarithmic complexity in the Hamiltonian simulation precision ϵ_{HS}^{-1}. The required block encoding of operator B makes use of a Linear Combination of Unitaries, and operators Prep_B and Sel_B, allowing to perform

a block encoding of a time segment of $e^{-i(A+B)\tau}$ as [231]

$$e^{-i(A+B)\tau} = e^{-iA\tau} \lim_{\substack{K\to\infty \\ M\to\infty}} \sum_{k=0}^{K} \frac{(-i\tau)^k}{M^k k!} \sum_{m_1=0}^{M-1} \cdots \sum_{m_k=0}^{M-1}$$
$$\left(e^{-i\tau(-1/2-m'_k)A/M} B e^{-i\tau(m'_k-m'_{k-1})A/M} B \dots B e^{-i\tau(m'_2-m'_1)A/M} B e^{-i\tau(m'_1+1/2)A/M}\right)$$
$$\approx \left(\langle 0|\, \mathrm{Prep}_B^\dagger\right)^{\otimes K} \sum_{k=0}^{K} \frac{(-i\lambda_B\tau)^k}{M^k k!} \sum_{m_1,\dots,m_k=0}^{M-1} \left(e^{-i\tau(M-1/2-m'_k)A/M} \mathrm{Sel}_B\right.$$
$$\left. e^{-i\tau(m'_k-m'_{k-1})A/M} \mathrm{Sel}_B \dots \mathrm{Sel}_B e^{-i\tau(m'_2-m'_1)A/M} \mathrm{Sel}_B e^{-i\tau(m'_1+1/2)A/M}\right)\left(\mathrm{Prep}_B\, |0\rangle\right)^{\otimes K}, \tag{4.117}$$

with $m'_1, \dots, m'_k$ the (time) ordered integers $m_1, \dots, m_k$. Due to the use of block encodings, this approach similarly requires splitting the total time evolution into small time segments and using oblivious amplitude amplification. As a consequence, the cost scales polylogarithmically with the inverse Hamiltonian simulation error ϵ_{HS}. Importantly, this technique may also be naturally used for time-dependent Hamiltonians, by taking A as the time-independent component and B as the time-dependent one.

Rank factorization. While not a Hamiltonian simulation technique of its own, rank factorization approaches have been used in the literature to reduce the 1-norm of the Hamiltonian in the context of second quantization and qubitization or to simplify the implementation of Prep and Sel. Note that the Hamiltonian 1-norm is a multiplicative factor in many of the state-of-the-art techniques above, and the reason why interaction picture and other methods were used in the first place [167]. Further, rank factorization methods have been found helpful when used in combination with QROM [18]. We briefly describe the single and double-rank techniques, noting that the state of the art is currently at the tensor hypercontraction techniques described in Ref. [158]. We follow the explanations in the appendices of this last reference.

We already discussed that the second-quantized Hamiltonian can be written as a one and two body term, see (4.82). Specifically,

$$T = \frac{1}{2} \sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p,q=1}^{N/2} T_{pq} (a^\dagger_{p,\sigma} a_{q,\sigma} + a^\dagger_{q,\sigma} a_{p,\sigma}), \tag{4.118}$$

and

$$V = \frac{1}{8} \sum_{\alpha,\beta\in\{\uparrow,\downarrow\}} \sum_{p,q,r,s=1}^{N/2} V_{pqrs}(a^\dagger_{p,\alpha}a_{q,\alpha} + a^\dagger_{q,\alpha}a_{p,\alpha})(a^\dagger_{r,\beta}a_{s,\beta} + a^\dagger_{s,\beta}a_{r,\beta}). \tag{4.119}$$

The next step is to Jordan-Wigner map these terms, using (4.73). If we define

$$Q_{pq\sigma} = \begin{cases} X_{p,\sigma}\vec{Z}X_{q,\sigma} & p < q, \\ Y_{p,\sigma}\vec{Z}Y_{q,\sigma} & p > q, \\ -Z_{p,\sigma} & p = q, \end{cases} \tag{4.120}$$

we can rewrite

$$T = \frac{1}{2} \sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p,q=1}^{N/2} T_{pq}Q_{pq\sigma} + \sum_{p=1}^{N/2} T_{pp}\mathbf{1}, \tag{4.121}$$

and

$$V = \frac{1}{8} \sum_{\alpha,\beta\in\{\uparrow,\downarrow\}} \left(\sum_{p,q,r,s=1}^{N/2} V_{pqrs}Q_{pq\alpha}Q_{rs\beta} + \sum_{p,q,r=1}^{N/2} V_{pqrr}Q_{pq\alpha} + \sum_{p,r,s=1}^{N/2} V_{prrs}Q_{rs\beta} + \sum_{p,r=1}^{N/2} V_{pprr}\mathbf{1} \right). \tag{4.122}$$

This suggests rearranging

$$T' = \frac{1}{2} \sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p,q=1}^{N/2} (T_{pq} + \sum_{r=1}^{N/2} V_{pqrr})Q_{pq\sigma}, \tag{4.123}$$

and

$$V' = \frac{1}{8} \sum_{\alpha,\beta\in\{\uparrow,\downarrow\}} \sum_{p,q,r,s=1}^{N/2} V_{pqrs}Q_{pq\alpha}Q_{rs\beta}, \tag{4.124}$$

plus a term proportional to the identity which can be omitted in the Hamiltonian simulation. The single-rank factorization consists of performing a Cholesky decomposition such that we can approximate

$$V_{pqrs} \approx \sum_{l=1}^{L} W^{(l)}_{pq}W^{(l)}_{rs}. \tag{4.125}$$

Consequently, the two-body term might be approximated by [35]

$$W = \frac{1}{8} \sum_{l=1}^{L} \left(\sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p,q}^{N/2} W^{(l)}_{pq}Q_{pq\sigma} \right)^2. \tag{4.126}$$

The double-rank factorization goes a step further and diagonalizes each $W_{pq}^{(l)}$, approximating V with [247]

$$F = \frac{1}{2}\sum_{l=1}^{L} U_l \left(\sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p=1}^{\Xi^{(l)}} f_p^{(l)} n_{p,\sigma} \right)^2 U_l^\dagger. \tag{4.127}$$

Discarding again the terms proportional to the identity, after the Jordan-Wigner mapping this operator looks

$$F' = \frac{1}{8}\sum_{l=1}^{L} U_l \left(\sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{p=1}^{\Xi^{(l)}} f_p^{(l)} Z_{p,\sigma} \right)^2 U_l^\dagger. \tag{4.128}$$

4.5. Quantum state preparation

Before diving into the different techniques to prepare ground states, let us briefly discuss the complexity of this and previous tasks.

Definition 7 (Bounded-error Quantum Polynomial-time (BQP)) *This complexity class is composed of all problems solvable by a quantum Turing machine in polynomial time and probability of error at most* 1/3*.*

BQP is the class of problems that are 'efficiently' solvable by a universal quantum computer. It is the quantum equivalent to the Bounded-error Probabilistic Polynomial-time (BPP), where the quantum Turing machine is substituted by a classical Turing machine. Running the algorithm multiple times, the 1/3 failure probability might be exponentially reduced. A larger complexity class is the following.

Definition 8 (Quantum Merlin-Arthur (QMA)) *Complexity class of binary decision problems where positive instances can be efficiently verified by a BQP solver with success probability 2/3, and negative instances rejected with similar probability.*

In more poetical words, Merlin, a powerful agent with unbounded computational resources, wants to convince Arthur (with access to a universal quantum computer) of the character, positive or negative, of one instance of the problem. Is there a quantum state and a polynomial-time single-interaction protocol by which Arthur can correctly verify or reject such a quantum state? This definition is the quantum equivalent of the NP complexity class, which would apply to a classical BPP verifier instead.

We know that Hamiltonian simulation [129], quantum phase estimation [260] and the HHL algorithm [117] are BQP-complete problems, any other BQP problem can be

reduced to them. Similarly, finding the non-interacting Kohn-Sham functional for a time-dependent electronic density is also BQP [255]. In contrast, distinguishing the ground state of a two-body Hamiltonian provided the promise of an energy gap [131], and finding the universal functional in DFT [213], are both QMA problems. However, while finding the ground state of general electronic Hamiltonians is computationally difficult, this does not preclude the possibility of finding average-case efficient algorithms for quantum systems found in nature. After all, those natural systems are often able to find the ground state, so we might as well. The situation is similar to the one we encountered in Section 2.4 with protein folding, and we hope to similarly find an efficient algorithm in practice.

Similar to our approach with protein folding, one idea is to use digital quantum simulation of quantum annealing. Indeed, the techniques developed by Ref. [162] and that we use in [62] are well-tailored to finding the ground state of quantum Hamiltonians [161, 265]. We already saw that the complexity of quantum Metropolis algorithms for this purpose scales as $O(\Delta^{-1})$, where Δ is the eigenvalue gap. Overall, it should be noted that this problem is so hard, that not even quantum computers are expected to offer an exponential quantum advantage [159].

4.5.1. Unitary coupled-cluster & variational quantum eigensolver

One alternative to quantum phase estimation is called the variational quantum eigensolver. It is based on the following idea: let the qubit-mapped Hamiltonian be

$$H = \sum_{i,\alpha} h_{i,\alpha} \sigma^i_\alpha + \sum_{i,j,\alpha,\beta} h_{i,j,\alpha,\beta} \sigma^i_\alpha \sigma^j_\beta + \dots, \tag{4.129}$$

where σ are Pauli operators and $\alpha, \beta, \dots \in \{x, y, z\}$, and $i, j, \dots$ indicate the qubit where such Pauli operator should be applied. Then, the energy of the system can be computed as the weighted average of the expectation values of each of the qubit Pauli string operators

$$E = \langle H \rangle = \sum_{i,\alpha} h_{i,\alpha} \langle \sigma^i_\alpha \rangle + \sum_{i,j,\alpha,\beta} h_{i,j,\alpha,\beta} \langle \sigma^i_\alpha \sigma^j_\beta \rangle + \dots, \tag{4.130}$$

The variational quantum eigensolver is based on the idea that we can prepare the ground state by variationally finding the state $|\psi\rangle$ that minimizes the ground state

energy

$$E = \min_{\psi} \frac{\langle\psi|H|\psi\rangle}{\langle\psi|\psi\rangle}. \tag{4.131}$$

The question then is to find a good ansatz for the state $|\psi\rangle$. The most popular choice is to use the Unitary Coupled-Cluster applied to the Hartree-Fock state as a way to parametrize the ground state:

$$|\psi\rangle = e^{T-T^\dagger} |\phi_0\rangle . \tag{4.132}$$

The operator in the exponent is anti-Hermitian, so the exponential is a unitary operator that can be implemented in a quantum computer. To implement it, sometimes classical Møller-Plesset 2 (perturbation theory over Hartree-Fock) coefficients might be used as a possible educated guess for the initialization for the parameters [209], and we will also need one of the Hamiltonian simulation techniques described in the previous section Section 4.4.3. For experiments in NISQ devices, this often entails implementing Trotter simulation of $U(\boldsymbol{t}) = e^{\frac{1}{2}\sum_{j=1}^{J} t_j(\tau_j - \tau_j^\dagger)}$. Then, to optimize the parameters t_j of energy E in (4.130) it is customary to use any variation of gradient descend (Stochastic Gradient Descend, ADAM, ...), where gradients are computed via the parameter shift rule [32, 214, 257]:

$$\frac{\partial E}{\partial t_j} = \frac{1}{2\sin\alpha} \sum_k h_k \left(\mathrm{Tr}[P_k U^\dagger(\boldsymbol{t}_+)\rho_0 U(\boldsymbol{t}_+)] - \mathrm{Tr}[P_k U^\dagger(\boldsymbol{t}_-)\rho_0 U(\boldsymbol{t}_-)] \right), \tag{4.133}$$

for $\rho_0 = |\phi_0\rangle\langle\phi_0|$, P_k the Pauli string operators in (4.130), h_k their weights, $\boldsymbol{t}_\pm = (t_1, \ldots, t_{j-1}, t_j \pm \alpha, t_{j+1}, \ldots, t_J)$, and finally α an arbitrary parameter that should be chosen as $\pi/4$ for maximum accuracy.

The previous procedure, however, is very costly due to the sheer number of parameters t_j to optimize in any molecule. For that reason, other ansätze have been proposed. For example, the Qubit Coupled-Cluster suggests using

$$U(\boldsymbol{t}) = \prod_j e^{t_j P_j}, \tag{4.134}$$

where P_j are Pauli string operators (not necessarily fermionic operators) that conserve the number of particles in the system [211].

Another possibility is using a small ansatz and progressively adding more terms according to their importance. This is the central idea of the adaptive derivative assembled pseudo-Trotter VQE (ADAPT-VQE) [109]. If we have a pool of $\{\tau_j\}$ operators, at any given step i one chooses, with replacement, the operator τ_{j_i} with the

largest gradient and adds it to the state preparation

$$|\psi\rangle = e^{t_{j_i}(\tau_{j_i}-\tau_{j_i}^\dagger)} \dots e^{t_{j_2}(\tau_{j_2}-\tau_{j_2}^\dagger)} e^{t_{j_1}(\tau_{j_1}-\tau_{j_1}^\dagger)} |\phi_0\rangle \,. \tag{4.135}$$

After choosing the operator, one optimizes the value of the associated parameter t_{j_i} and freezes it. This ansatz has the advantage that it can be stopped at arbitrary lengths.

While the previous quantum methods are based on the variational approach of the Unitary Coupled Cluster, we know from our discussion in Section 4.3.3 that there is also a projective approach. This is the main idea of the projective quantum eigensolver (PQE), where one aims to minimize the residuals [225]

$$r_m = \langle \phi_m | U^\dagger(\boldsymbol{t}) H U(\boldsymbol{t}) | \phi_0 \rangle \to 0. \tag{4.136}$$

This forms a system of non-linear equations that might be solved with classical methods. The projective approach also allows for an adaptive version called selective, that progressively grows

$$H \to e^{t_{j_i}(\tau_{j_i}^\dagger-\tau_{j_i})} \dots e^{t_{j_2}(\tau_{j_2}^\dagger-\tau_{j_2})} e^{t_{j_1}(\tau_{j_1}^\dagger-\tau_{j_1})} H e^{t_{j_1}(\tau_{j_1}-\tau_{j_1}^\dagger)} e^{t_{j_2}(\tau_{j_2}-\tau_{j_2}^\dagger)} \dots e^{t_{j_i}(\tau_{j_i}-\tau_{j_i}^\dagger)}. \tag{4.137}$$

Note that in this case the operator is built in reverse order, and the i^{th} operator is chosen to zero out the largest residual at step i [225].

Finally, let us briefly mention that variational circuits such as these may find a problem of vanishing gradients, known as barren plateaus [180]. The origin of this phenomenon is often the set of gates of the ansatz resembling a random set of unitaries. Fortunately, using the set of fermionic operators of the Unitary Coupled-Cluster as ansatz may avoid this problem [67, 93, 115]. A detailed discussion of the quantum variants of Unitary Coupled-Cluster can be found at Ref. [8], and of variational algorithms and techniques in general in Ref. [67].

4.5.2. Projection methods

Imaginary time evolution and quantum Krylov subspace. Besides implementing excitations of the Hartree-Fock as a way to prepare the ground state of the system, an alternative idea is to filter out the excited states. For example, in the imaginary time evolution, one seeks to implement an approximation to operator $e^{-\tau H}$ [188]. It is then prescribed to use tomography to select unitary operators $e^{-i\tau A}$ that approximate the action of $e^{-\tau H}$.

This imaginary time evolution technique might also be used to generate excited states. In particular, we can generate a Krylov subspace by iteratively applying the $e^{-\tau H}$ operator, $|\psi_l\rangle = e^{-l\tau H}|\psi_0\rangle$, where $|\psi_0\rangle$ is often taken the Hartree-Fock state $|\phi_0\rangle$. The even-l states $|\psi_0\rangle$, $|\psi_2\rangle$, $|\psi_4\rangle$, ... form a basis to describe the ground state [188]. The Hamiltonian matrix elements of these states are then computed as $\langle\psi_l|H|\psi_{l'}\rangle = \langle\psi_{(l+l')/2}|H|\psi_{(l+l')/2}\rangle$, and diagonalizing it, one can get an approximation to the ground state and excited states. This method is known as QLanczos [188], and may even be used to prepare multi-reference states [226, Section 2.3]: if the approximate ground state of the Hamiltonian contains similar amplitudes of various Slater determinants, we can find them by repeatedly measuring it. The generation of Krylov states and diagonalization may then be repeated with each of the Slater determinant reference states [226, Figure 1]. A similar approach of diagonalizing the Hamiltonian subspace spanned by the most relevant Slater determinants is also pursued in Ref. [240] but in the Configuration Interaction framework.

Linear Combination of Unitaries. Instead of using tomography to find operators that project out the higher energy states, we can also use Linear Combination of Unitaries to synthesize a function that probabilistically projects into the ground state. For example, taking inspiration from the linear system of equation solver, we can implement linear combinations of unitaries that simulate H^{-1} as decomposed in (3.33). More generally, we can find a similar linear combination of unitaries to implement H^{-K} for any H with a positive spectrum [156], or H^K if the spectrum is negative [37], either of which may require shifting the Hamiltonian by a constant. Other alternatives include approximating $\cos^K H$ [97] or $e^{-t^2H^2}$ [130].

The key is that once a block encoding of these operators is applied, we have to amplify the success probability, via standard amplitude amplification, or the fixed point method from Section 2.2.2. The cost of this projection algorithm will depend crucially on two main parameters: Δ, the eigenvalue gap, and γ, the overlap of the initial state with the ground state. The former plays a role in the choice of parameter K necessary to distinguish the ground state, while the latter will be determinant in the number of rounds of amplitude amplification required to have a success probability close to 1. Furthermore, Δ will appear divided by the Hamiltonian one-norm λ due to the block encoding. Consequently, during amplitude amplification, the number of calls to the oracle implementing Hamiltonian simulation U_H will scale as $\tilde{O}(\lambda/(\Delta\gamma))$, and $O(1/\gamma)$ calls to U_I, the oracle preparing the initial state, will be required.

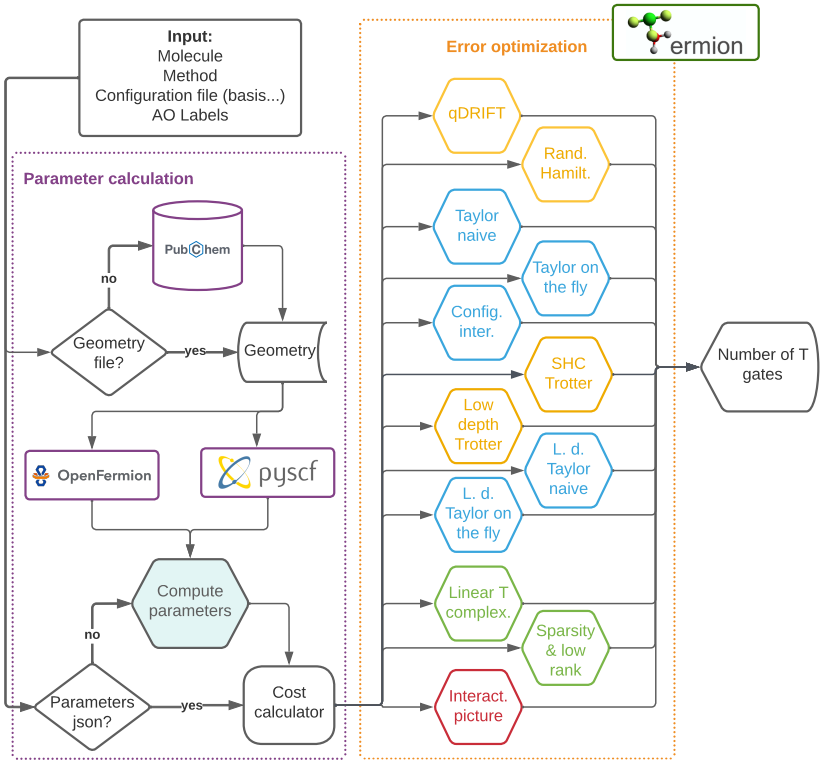


Figure 4.2 **Flowchart of the architecture of TFermion** [63], divided into two parts: the first one centered on the computation of the parameters needed for the cost estimate; and a second one on using such parameters to compute the number of T-gates. Methods are colored according to the Hamiltonian simulation technique used: yellow for Trotter, blue for Taylor series, green for qubitization, and red for interaction picture.

Qubitization. Having discussed qubitization and signal processing as natural successors to the linear combination of unitaries approach, the reader might not be surprised to find out that this technique can also be used to filter out unwanted eigenstates. The objective, as in the previous cases, is to synthesize a Grover-like rotation that amplifies the ground state. The natural choice for such rotation is a sign function affecting only the lowest eigenstate,

$$R_{<\mu} = \sum_{k:\lambda_k<\mu} |\phi_k\rangle \langle\phi_k| - \sum_{k:\lambda_k>\mu} |\phi_k\rangle \langle\phi_k| \,, \tag{4.138}$$

and to implement one such block encoding we will need an approximation to the ground state energy and the eigenvalue gap Δ, just as in the above case.[5] Specifically, Ref. [163, Lemma 3] shows that one needs a polynomial of degree $O\left(\frac{\lambda}{\Delta}\log\epsilon^{-1}\right)$ to approximate the sign function to error ϵ, except in the segment $[\mu-\Delta/2, \mu+\Delta/2]$. This polynomial can be prepared with a similar number of queries to the Hamiltonian block encoding oracle U_H [163, Theorem 1]. Furthermore, if desired, a block encoding of a projector can be prepared as

$$P_{<\mu} = \frac{1}{2}(R_{<\mu} + \mathbf{1}), \tag{4.139}$$

that will succeed to prepare the ground state with a probability of at least γ^2. Consequently, this also requires (fixed point) amplitude amplification. Overall, the cost is $O(\frac{\lambda}{\Delta\gamma}\log\epsilon^{-1})$ calls to the Hamiltonian block encoding U_H, and $O(1/\gamma)$ calls to the initial state preparation oracle U_I. This procedure also needs an upper bound estimate μ of the ground state energy, and if not available a binary search procedure based on limited-precision amplitude estimation must be used [163, Lemma 7].

4.6. TFermion

Since quantum chemistry and material science seem such good applications of quantum computing, there has been abundant research not only on developing new algorithms but on estimating the actual cost of implementing those. Starting with the seminal work of Ref. [206] on FeMoco, a molecule that is capable of fixing atmospheric nitrogen, there have been numerous articles trying to estimate the cost of using quantum phase estimation algorithms to compute chemical reaction rates [206, 247], and analyze battery properties [137, 84] or biological enzymes [103].

The cost is often measured in the number of non-Clifford gates, for example, T or Toffoli gates. T gates are $\pi/8$ single-qubit rotations, while Toffoli gates are the quantum equivalent of classical AND gates. These are the most costly gates because they cannot be implemented transversally in the surface [142, 96] or color codes [46, 47], and thus require costly magic state distillation [99] or other more expensive alternatives [38].

On the other hand, choosing the appropriate algorithm to tackle a given problem is not obvious and requires a deep understanding of each of the available algorithms. In this context, TFermion is a library our group has developed to facilitate the T gate

[5] This procedure to implement reflections might also be used in Section 2.3.1 in general and in Algorithm 4 in particular, to implement quantum search algorithms without the need for quantum phase estimation, even if asymptotic complexity is similar.

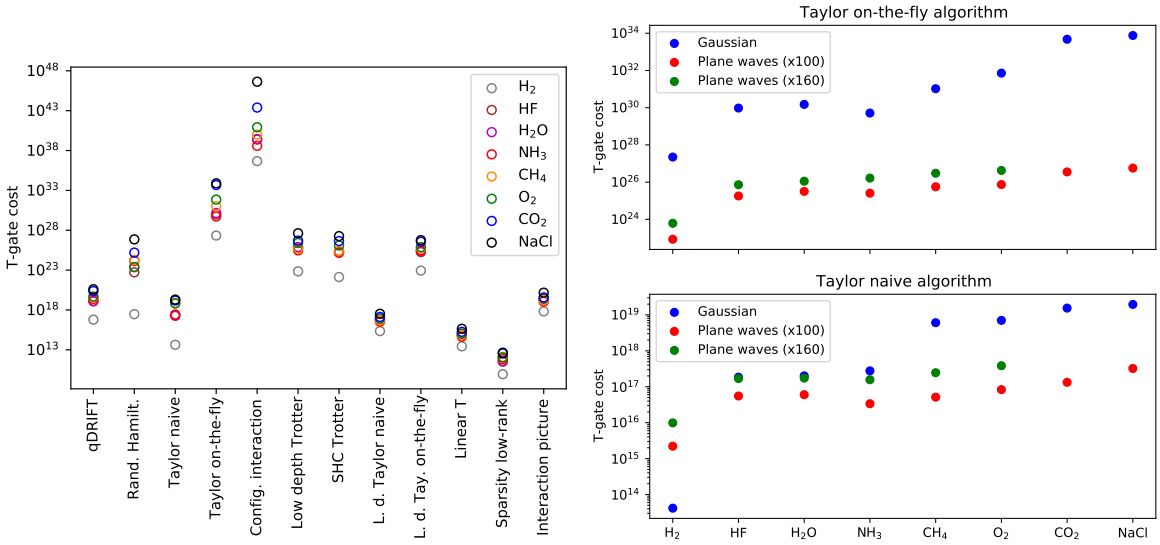


Figure 4.3 **Examples of the results that can be obtained with TFermion** [63]. Left: the choice of basis and Hamiltonian simulation used will have a profound impact on the overall cost of the quantum phase estimation algorithm. Right: Our library allows us to compare the cost of the same algorithm with a different basis (plane waves vs Gaussians) provided a fair comparison between the number of functions on either basis can be established.

cost estimation of performing phase estimation over a given molecule, with the different methods proposed in the literature [63].

In our article, we explain how several quantum phase estimation implementations can be compiled into a circuit, as well as give the first estimates of their T-gate cost. This library may find use cases such as comparing the cost of different algorithms. As an example, we analyzed the impact of the basis used for Taylor series-based quantum phase estimation, see Fig. 4.3. Note however that this comparison should be taken with care as we did not compute the amount of error induced in the algorithm due to the use of a finite number of basis functions in each case. Future extensions to the library should include newer algorithms, the number of qubits required, or the cost of state preparation.

4.7. Lithium batteries

The second article on this topic presented in this thesis, Ref. [84], is a study on the advantages and cost of analyzing lithium battery properties using quantum phase estimation. In our objective toward decarbonization of energy generation, batteries still

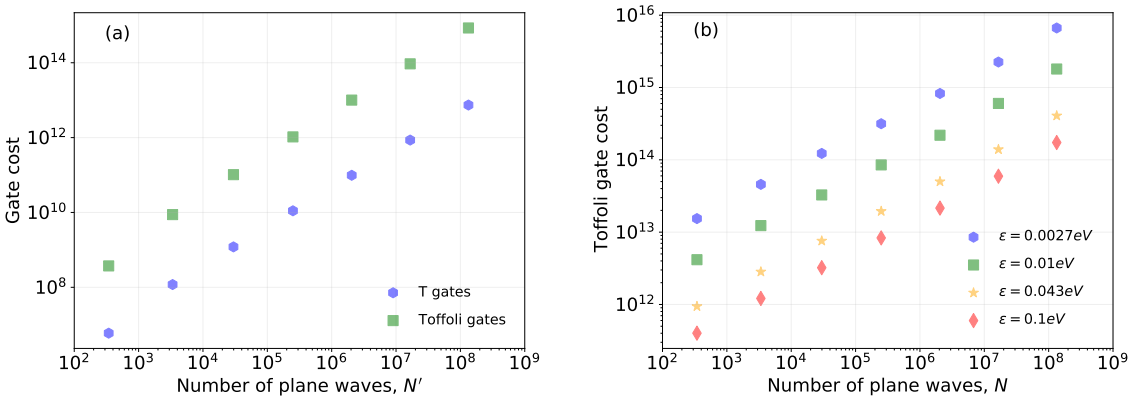


Figure 4.4 **Non-Clifford gate cost for the execution of Quantum Phase Estimation** algorithm in a fault-tolerant quantum algorithm for Li_2FeSiO_4 cathode material [84]. Left: Toffoli and T-gate cost of performing the Givens rotations explained below to perform the basis change from molecular orbital to plane wave basis, for different basis sizes. Right: Toffoli cost of the quantum phase estimation as a function of the number of basis functions, and target precision. Different error targets are required for the estimation of different properties.

represent a notable technological limitation. Therefore, being able to better predict their properties computationally has substantial value. The cathode is one of the key components of the battery and will oftentimes determine some of its most important properties. For our work, we opted for Li_2FeSiO_4, a common candidate material for the cathode of batteries [174]. During discharge, it undergoes the following chemical reaction,

$$Li_xFeSiO_4 + (2-x)Li \rightarrow Li_2FeSiO_4. \tag{4.140}$$

In other words, Li atoms are inserted in the structure of the material. The voltage can then be computed as

$$V = -\frac{[E_{Li_2FeSiO_4} - E_{Li_xFeSiO_4} - (2-x)E_{Li}]}{(2-x)e}. \tag{4.141}$$

Other properties such as the ionic mobility inside the cathode, and the thermal stability of the material, can also be predicted from the ground state energy of the different phases of the material.

In our article, we suggest using quantum phase estimation to compute these ground-state energies, which is an efficient approach as described throughout this chapter. The currently most promising implementation is based on a first quantization and plane waves representation of the quantum state, and qubitization as the Hamiltonian

simulation [16, 231]. This choice of this combination stems from the use of plane waves to simulate a periodic material, first-quantization to reduce the one-norm of the Hamiltonian, and qubitization as a rapid Hamiltonian simulation technique.

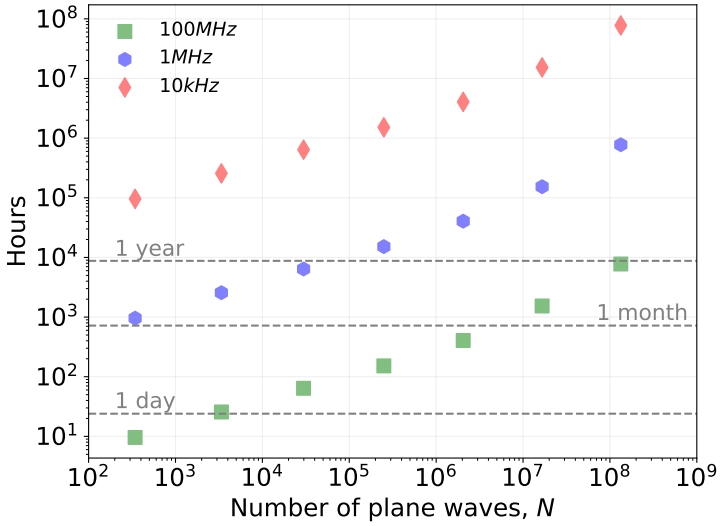


Figure 4.5 **Execution time estimates for the qubitization quantum phase estimation algorithm** [84]. Assuming chemical accuracy $\varepsilon = 0.043$ eV, this figure represents the time cost of implementing the quantum phase estimation algorithm for Li_2FeSiO_4 at varying clock rates for the synthesis of non-Clifford gates. The total number of qubits is 2,375 for $n_p = 4$ and 6,652 for $n_p = 9$. We compute the distillation time as the product of the number of Toffoli gates, the surface code distance d, and the clock frequency, all divided by a small n_p factor originating from the techniques in [170] that parallelize the CSWAPs and arithmetic computations. We compute d in this figure as in the moderate error case of Ref. [137]. We emphasize that these are rough estimates whose main purpose is to provide a method to interpret the gate cost.

The main contributions of our article are three-fold: first, we slightly extend the applicability of the original algorithm from the orthonormal cubic computational cell to orthogonal, a parallelepiped. This adaptation requires minor changes in the Prep component of the algorithm, in particular of a state with amplitudes proportional to $\frac{1}{\|G_\nu\|}$, which slightly increases the cost of the algorithm.

More importantly, we explain how to perform the Givens rotations required to map the Hartree Fock from molecular orbitals to the plane wave basis in first quantization [144] and make sure that the procedure conserves antisymmetrization. The Hartree-Fock state that we take as the initial state will only be a computational basis state in the molecular orbital basis. However, to use our Hamiltonian simulation

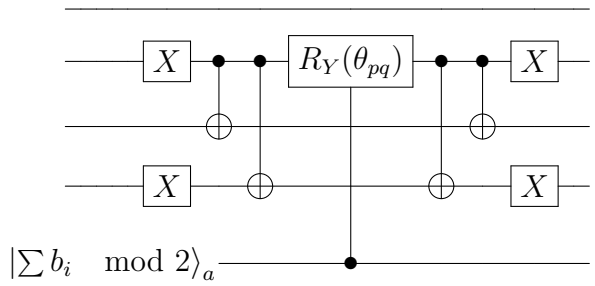


Figure 4.6 **Circuit diagram of an example controlled rotation** $R_Y(\theta_{pq})$ [84]. The rotation is performed on the subspace spanned by $|p\rangle = |0101\rangle$ and $|q\rangle = |0010\rangle$. The following procedure is applied to the bits where they differ, namely the last three qubits. First, we apply X gates such that $|p\rangle \to |0000\rangle$ and $|q\rangle \to |0111\rangle$. Then, CNOT gates map these states to $|0000\rangle$ and $|0100\rangle$ respectively. This allows us to perform a rotation on the second qubit controlled on the auxiliary qubit $|\sum b_i \mod 2\rangle_a$. Finally, the CNOTs and X gates are uncomputed, yielding the desired controlled rotation on the subspace span$\{|p\rangle, |q\rangle\}$.

method, we require such a state to be mapped to the plane wave basis, a procedure that has been previously explored in the literature [195, 224, 144]. Such basis change can be written as the application of the operator

$$U(u) = e^{\sum_{pq} [\log u]_{pq} a_p^\dagger a_q}, \tag{4.142}$$

according to Thouless theorem [239], see Footnote 2. To implement it, the Kivlichan method diagonalizes the corresponding operator matrix, decomposing it in two-orbital Givens rotations

$$R_Y(\theta_{pq}) = \begin{pmatrix} \cos(\theta_{pq}) & -\sin(\theta_{pq}) \\ \sin(\theta_{pq}) & \cos(\theta_{pq}) \end{pmatrix}, \tag{4.143}$$

between the η occupied orbitals and the $N - \eta$ unoccupied ones [144]. Givens rotations are rather straightforward in second quantization, just a two-qubit rotation [12]. In contrast, in first quantization, it becomes much more challenging as the information about the occupation of an orbital is delocalized among all registers. In this representation, registers track the orbital in which each electron is, not whether particular orbitals are occupied or empty. Thus, our contribution is a method to perform such rotations in first quantization.

To explain how the procedure works, the key aspect to understand is that given the fermionic nature of the system, $R_Y(\theta_{pq})$ will only behave non-trivially when the occupation of $|p\rangle$ and $|q\rangle$ is different, else it acts as the identity operator. Consequently, we can implement the Givens rotation on $|p_1, \ldots, p_\eta\rangle$ with the following procedure [84]:

1. Initialize η auxiliary qubits $|b_j\rangle_j$ in the state $|0\rangle_1 \dots |0\rangle_\eta$.

2. For $1 \leq j \leq \eta$: If $p_j \in \{p, q\}$, flip the auxiliary qubit $|0\rangle_j$ to $|1\rangle_j$.

3. For $1 \leq j \leq \eta - 1$: Controlled on the auxiliary qubit $|b_j\rangle_j$, swap the j-th and η-th register.

4. The auxiliary qubits are now in some state $|b_1\rangle_1 \dots |b_\eta\rangle_\eta$, where each b_j indicates if $p_j \in \{p, q\}$. Controlled on the parity of $\sum_{i=1}^{\eta} b_i$, apply $R_Y(\theta_{pq})$ on the subspace $\text{span}\{|p\rangle, |q\rangle\}$ of the η-th register. This step is illustrated for an example in Fig. 4.6 and can be easily generalized.

5. Undo the controlled swaps and uncompute the auxiliary qubits by applying the same operators in steps 2 and 3.

In the article, we show in a principled way that this procedure preserves the antisymmetry of the state, which can also be explicitly checked.

Finally, while most elements of the algorithm presented in our article could be previously found in the literature, a third important contribution is understanding which ones to use and how to combine them into a complex algorithm capable of performing this computationally challenging task. The main conclusion of our work is that the algorithm is almost efficient enough, that provided with a quantum computer with high but achievable clock rates we could run quantum phase estimation in a reasonable amount of time for useful battery materials. However, various limitations should be addressed in future work, such as the extension to non-orthogonal cells, or going beyond Hartree-Fock as a way to prepare ground states. The latter might be especially important as the overlap between the Hartree Fock and ground states may decrease exponentially with the system size [146]. Overall, this article and similar previous ones indicate that chemistry and material science might be one of the most promising applications of quantum computing.

4.8. Results

- We have described Hartree-Fock, Density Functional Theory, and Coupled Cluster, and established their connections with quantum algorithms or as state preparation methods.

- We have described all the algorithmic choices involved in chemistry calculations, with special emphasis on Hamiltonian simulation.

- We have released TFermion, the library corresponding to Ref. [63], that enables researchers to easily compare quantum algorithms present in the literature, for which only complexity estimates were available. This enables, for example, the comparison of different basis functions, or Hamiltonian simulation techniques.
- In Ref. [84], we have described how state-of-the-art first-quantization algorithms may be applied to analyze Li-ion battery properties, such as the thermal stability or the energy capacity of the battery. First-quantization techniques are especially useful to reduce the cost of the plane wave basis. Together with qubitization, this makes for very efficient quantum algorithms.
- Our results in Refs. [63, 84] indicate that the most promising Hamiltonian simulation techniques are either:

  1. Qubitization, making use of Gaussian basis functions with rank-factorization.
  2. Qubitization or interaction picture Hamiltonian simulation techniques, in plane wave basis and first quantization.

  Depending on the system (isolated molecules or periodic materials) one or the other might be preferable.
- The estimated number of logical gates for this algorithm means this is a robust quantum computing application candidate, and could be implemented in a realistic amount of time in a fault-tolerant quantum computer.
- In Ref. [84], we have made technical contributions to state preparations and the applicability of the first-quantization algorithm. More specifically, we have explained how to implement Givens rotations in first-quantization, without which state preparation in plane waves is not completely defined.
- The second technical contribution is the extension of this algorithm to orthogonal unit cells.
- We have understood that the main bottleneck in quantum chemistry is the state preparation problem. We have described a few techniques for that purpose.

Chapter 5

Quantum error correction

> With group and eigenstate, we've learned to fix
> Your quantum errors with our quantum tricks.
>
> Daniel Gottesman, *Quantum Error Correction Sonnet.*

5.1. Objectives

- Understanding why quantum error correction plays a key enabling role in quantum chemistry, and why non-Clifford gates are often the most expensive kind of logic gate.
- Understanding the topological error codes and their two most important representatives: surface and color codes.
- Describing a procedure of how quantum fault tolerance might be achieved with error correction techniques.
- Analyzing how a Machine Learning decoder can be flexibly used with various topological codes.

5.2. Introduction to error codes

In the previous chapters, we have reviewed the different families of algorithms that one may wish to implement, and how they relate to each other. In particular, in Chapter 4 we have explored the use of these algorithms, and have explored the time and resource cost of fault-tolerantly implementing them. There, we referred to the

necessity of distilling magic states to implement T gates. In this chapter, we explain the most important topological quantum error correction codes, the leading approach to ensure that quantum computing protocols can be implemented fault tolerantly. We start describing in the next section some basic concepts of quantum error correction, following the presentation in Ref. [204, Chapter 7].

The objective of error correction might appear challenging: not only do we have to prevent decoherence of our data, but we have to ensure that we can fault tolerantly implement a continuous set of gates. Unfortunately, being continuous, even minor errors in such gates would accumulate, ultimately leading to computational failure [204]. However, even if such is the case, we can decompose the possible errors in a discrete set. To see how, imagine we have an encoded qubit $|\psi\rangle = a|0\rangle + b|1\rangle$. Further, suppose that there is an arbitrary unitary operator U that acts upon our qubit, and an environmental qubit is assumed to be initialized to 0 on some basis, $|0\rangle_E$, and to which we have no access. In general, the action of U is

$$\begin{aligned} U : |0\rangle |0\rangle_E &\mapsto |0\rangle |e_{00}\rangle_E + |1\rangle |e_{01}\rangle_E \\ |1\rangle |0\rangle_E &\mapsto |0\rangle |e_{10}\rangle_E + |1\rangle |e_{11}\rangle_E , \end{aligned} \tag{5.1}$$

where $|e_{ij}\rangle$ need not be normalized or orthogonal states. This means that U transforms an arbitrary quantum state as

$$\begin{aligned} U : (a|0\rangle + b|1\rangle)|0\rangle_E \mapsto & a(|0\rangle |e_{00}\rangle_E + |1\rangle |e_{01}\rangle_E) \\ & + b(|0\rangle |e_{10}\rangle_E + |1\rangle |e_{11}\rangle_E). \end{aligned} \tag{5.2}$$

We can rewrite the resulting state as

$$\begin{aligned} &= (a|0\rangle + b|1\rangle) \otimes \frac{1}{2}(|e_{00}\rangle_E + |e_{11}\rangle_E) \\ &+ (a|0\rangle - b|1\rangle) \otimes \frac{1}{2}(|e_{00}\rangle_E - |e_{11}\rangle_E) \\ &+ (a|1\rangle + b|0\rangle) \otimes \frac{1}{2}(|e_{01}\rangle_E + |e_{10}\rangle_E) \\ &+ (a|1\rangle - b|0\rangle) \otimes \frac{1}{2}(|e_{01}\rangle_E - |e_{10}\rangle_E). \end{aligned} \tag{5.3}$$

Taking a closer look, this is equivalent to

$$\mathbf{1}|\psi\rangle \otimes |e_{\mathbf{1}}\rangle + X|\psi\rangle \otimes |e_X\rangle + Y|\psi\rangle \otimes |e_Y\rangle + Z|\psi\rangle \otimes |e_Z\rangle , \tag{5.4}$$

where X, Y and Z represent the Pauli operators. We can always perform such an expansion because the set of Pauli operators and the identity span the state of 2×2 matrix. For n-qubit states $|\psi\rangle$, we can similarly expand any error in the basis of tensor products of Pauli operators, $E_a \in \{\mathbf{1}, X, Y, Z\}^{\otimes n}$ such that

$$|\psi\rangle \otimes |0\rangle_E \mapsto \sum_a E_a |\psi\rangle \otimes |e_a\rangle_E , \tag{5.5}$$

where $|e_a\rangle_E$ need not be mutually orthogonal. If we are able to project into one of the E_a possibilities and distinguish which one it was, we can implement $E_a^\dagger$ to correct our data qubits (not the environmental ones).

The errors we will aim to correct are a subset $\mathcal{E} \subset \{E_a\}$. If E_a is composed of t non-trivial Pauli operators, then we will say it has *weight* t. In a binary (qubit) code, we encode 2^k 'code words' $|\bar{i}\rangle$ in a n-qubit space. The code is further characterized by the *code distance* d, the minimum weight of the operator changing the code word, e.g.,

$$d = \min_{E_a \in \mathcal{E}} t(\boldsymbol{E_a}) : \langle \bar{i}|E_a|\bar{j}\rangle \neq C_a \delta_{ij}, \tag{5.6}$$

with C_a a normalization coefficient. Overall, quantum error corrections codes will be denoted by their properties $[[n, k, d]]$, where the double bracket identifies them in contrast to single-bracket classical codes. In general, if errors E_a and E_b have weight smaller than $d/2$,

$$\langle \bar{i}|E_b^\dagger E_a|\bar{j}\rangle = C_{ab}\delta_{ij} \tag{5.7}$$

with C_{ab} an arbitrary Hermitian matrix, is a necessary and sufficient condition to be able to recover the correct code word [204, Equation 7.19]. In other words, if the error E_a has weight less than $d/2$, we will be able to correct the error using the correction procedure $E_b^\dagger$, which can only recover the code word $|\bar{i}\rangle$ because $E_b^\dagger E_a$ has weight smaller than d. In contrast, if one error had a weight equal to or larger than $d/2$, then it could be misidentified and corrected as a different code word.

As an introduction to quantum error correction codes, let us review one important family of classical codes, binary linear codes. In these codes, the code subspace is spanned by a set of binary vectors $\{v_i\}$, such that any message $\alpha_1, \ldots, \alpha_k$ is encoded as

$$\alpha_1, \ldots, \alpha_k \mapsto \sum_i \alpha_i v_i, \tag{5.8}$$

where summation is carried out modulo two. These generating vectors might be written as a $k \times n$ *generating matrix*, G

$$G = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_k \end{pmatrix} \tag{5.9}$$

and encoded message as $v(\alpha) = \alpha G$. Alternatively, one can indicate $n - k$ linear constraints, which form a $(n - k) \times n$ *parity check matrix* H, such that $Hv = 0$ for all the generating vectors. Consequently, we also have $HG^T = 0$. The parity check matrix can be used to detect errors e in the code, as in that case $H(v + e) = He \neq 0$ will indicate an error syndrome.

The distance d in these codes is defined as the minimum Hamming weight of vectors $\{v_i\}$, defined as the number of non-zero elements of the vectors,

$$d = \min_i |v_i|_1. \tag{5.10}$$

Similarly to our discussion above, the correctable errors are those of weight $t < d/2$ [204]. From this code C, we can generate its *dual*, by taking the transpose of $HG^T = 0$, $GH^T = 0$. In the new $n - k$ code $C^\perp$, H^T is the generating matrix and G is the parity check matrix.

Can we say anything more about the relation between the primal and its dual code? Since $GH^T = 0$, $\forall v \in C$ and $\forall u \in C^\perp$, we know that $v \cdot u = 0$, and consequently $(-1)^{v \cdot u} = 1$. On the other hand, if $u \notin C^\perp$ but $v = \alpha G$, it is clear that [204]

$$\sum_{v \in \{0,1\}^k} (-1)^{v \cdot w} = 0, \quad \forall w \neq 0 \quad \Rightarrow \quad \sum_{v \in C} (-1)^{v \cdot u} = \sum_{\alpha \in \{0,1\}^k} (-1)^{\alpha \cdot Gu} = 0. \tag{5.11}$$

In summary,

$$\sum_{v \in C} (-1)^{v \cdot u} = \begin{cases} 2^k & u \in C^\perp, \\ 0 & u \notin C^\perp. \end{cases} \tag{5.12}$$

As the eager reader might have noticed, this poses a relation between both codes via the Hadamard transformation, which we explore next.

5.2.1. CSS codes

The concept of dual codes can be exploited to generate one family of quantum codes called Calderbank–Shor–Steane (or CSS) codes [193]. The construction is as follows:

let C_1 be one code defined by $(n-k_1)\times n$ parity check matrix H_1, and similarly C_2 with $(n-k_2)\times n$ parity check matrix H_2. We choose C_2 to be a subcode of C_1, by imposing that all constraints of C_1 are also obeyed by C_2, so $k_2 < k_1$. This construction allows defining equivalence classes in C_1: two code words $u, v \in C_1$ are equivalent, if they are the same up to an element of C_2 [204],

$$u \equiv v \iff \exists w \in C_2 : u + w = v. \tag{5.13}$$

A CSS code is a code encoding $k_1 - k_2$ logical bits, where each equivalence class is one of the $2^{k_1-k_2}$ code word. The basis elements are

$$|\bar{w}\rangle_F = \frac{1}{\sqrt{2^{k_2}}} \sum_{v\in C_2} |v+w\rangle\,, \tag{5.14}$$

where the subindex denotes that this basis will be able to protect against bit-flip errors. As hinted on the discussion on the dual code, we can transform the code C on its dual code $C^\perp$ using the bitwise Hadamard over all qubits

$$\begin{aligned} |\bar{w}\rangle_P =& H_2^{\otimes n} |\bar{w}\rangle_F = \frac{1}{\sqrt{2^n}} \sum_u \frac{1}{\sqrt{2^{k_2}}} \left(\sum_{v\in C_2} (-1)^{u\cdot v} \right) (-1)^{u\cdot w} |u\rangle \\ \overset{(5.12)}{=}& \frac{1}{\sqrt{2^{n-k_2}}} \sum_{u\in C_2^\perp} (-1)^{u\cdot w} |u\rangle\,. \end{aligned} \tag{5.15}$$

Furthermore, if we shift w by an element $c \in C_2$ (e.g., staying in the same equivalence class), the resulting $|\bar{w}\rangle_P$ will not change because $(-1)^{u\cdot(w+c)} = (-1)^{u\cdot w}(-1)^{u\cdot c} = (-1)^{u\cdot w}$ because $u \cdot c = 0$.

Given that the Hadamard is self-inverse, we can use it to move back and forth between the primal and dual representations, and consequently correct the phase-flip errors as bit-flip errors in the dual code. The distance for flip errors will be the minimum weight of generating vectors of C_1, while for phase errors it will be the minimum weight of generating vectors of $C_2^\perp$ [204]. The overall distance will consequently be the smallest of either.

5.2.2. Stabilizer codes

Another particularly important class of quantum error correction codes are so-called called stabilizer codes, which make use of the Pauli structure of errors in (5.5). Let $P = \pm\{\mathbf{1}, X, Z, XZ = iY\}^{\otimes n}$ be the group of the n-fold tensor product of Pauli

operators, with iY in place of Y so that all matrix entries are real; and let $\mathcal{S} \subset P$ be an Abelian subgroup. Then, since the elements commute and can be jointly diagonalized, we define the *stabilizer code* as the $+1$ eigenspace of $\mathcal{S}$, $\mathcal{H}_\mathcal{S}$ [204]:

$$\psi \in \mathcal{H}_\mathcal{S} \iff M\psi = \psi \quad \forall M \in \mathcal{S}. \tag{5.16}$$

In particular, $-\mathbf{1} \notin \mathcal{S}$, because it has no $+1$ eigenvalue. $\mathcal{S}$ can be characterized by its generators, $\{M_i\}$. These operators can be understood as the parity check operators of the code.

Errors, on the other hand, will map the state outside the joint $+1$ eigenstate of all generators. For each E_a there will be at least one generator M_i such that

$$M_i E_a \left|\psi\right\rangle = -E_a \left|\psi\right\rangle = -E_a M_i \left|\psi\right\rangle . \tag{5.17}$$

In that case, we see that the error and generator anticommute, instead of commute. This allows us to define an *error syndrome* s_{ia}

$$M_i E_a = (-1)^{s_{ia}} E_a M_i, \tag{5.18}$$

which will be $s_{ia} = 0$ if error E_a commutes with the stabilizer, and $s_{ia} = 1$ if it anticommutes. Once identified thanks to the syndrome, we can apply a recovery procedure E_b that will hopefully recover the encoded state by mapping us back to $\mathcal{H}_\mathcal{S}$.

Finally, we have to explain how logical operators will act on the code. They will be related to the normalizer of $\mathcal{S}$.

Definition 9 (Normalizer and centralizer) *The normalizer of $\mathcal{S}$ in group G, $\mathcal{N}(\mathcal{S})$ is defined as*

$$\mathcal{N}(\mathcal{S}) = \{g \in G | g\mathcal{S} = \mathcal{S}g\} = \{g \in G | g\mathcal{S}g^{-1} = \mathcal{S}\}. \tag{5.19}$$

In other words, the normalizer is the set of elements of the group G that take components of $\mathcal{S}$ to possibly different ones of $\mathcal{S}$ under conjugation. The centralizer is similar, but commutes element-wise with each element of $\mathcal{S}$

$$\mathcal{Z}(\mathcal{S}) = \{g \in G | \forall s \in \mathcal{S}, gs = sg\} = \{g \in G | \forall s \in \mathcal{S}, gsg^{-1} = s\}. \tag{5.20}$$

If G is not clear, then we will indicate it by explicitly writing $\mathcal{Z}_G(\mathcal{S})$ and $\mathcal{N}_G(\mathcal{S})$.

The group G will often be the Pauli group over the physical qubits, for instance in the stabilizer surface and color codes. Since $\mathcal{S}$ is Abelian, $\mathcal{S} \subset \mathcal{N}(\mathcal{S})$. Moreover, in the

stabilizer codes we will explore next, $\mathcal{Z}(\mathcal{S}) = \mathcal{N}(\mathcal{S})$. Why is the normalizer important? It will represent all elements of P that commute with but are not necessarily in $\mathcal{S}$. Since normalizer operators commute with components of the stabilizer, their syndrome will always be $+1$. On the other hand, since they do not necessarily belong to $\mathcal{S}$, they can induce logical changes in the encoded information. Thus, logical operators in stabilizer codes will be related to the $\mathcal{N}(\mathcal{S})/\mathcal{S}$ quotient group.

5.3. Topological error correction

Topological codes are the most important family of both CSS and stabilizer codes [41]. In these codes, the physical qubits are represented as some element of a lattice embedded in a differentiable manifold. Such elements are often given the name of d-cells, where d represents the dimension. For example, 0-cells are vertices, and 1-cells are edges... Topological codes receive this name because they encode logical information in topologically non-trivial objects in the manifold. Since topological elements are robust to local deformations, these codes display great robustness and are often considered a key aspect of almost any error correction scheme. The two most important codes are the toric code (also known as surface code) [142] and the color code [46]. In this section, we mostly follow the presentation by Ref. [41] of these codes.

5.3.1. Homology

Let us start defining some concepts that will, later on, be useful for understanding these error correction schemes. The first concept we will introduce are d-chains, subsets of d-cell elements e_i^d that can be understood as a formal sum

$$c^d = \sum_i c_i e_i^d, \quad c_i \in \{0, 1\}. \tag{5.21}$$

The set of d-chains will be called C_d, and has Abelian subgroup structure. In fact, using the notation $\mathbb{Z}_2 = \mathbb{Z} \mod 2$, then $C_0 \simeq \mathbb{Z}_2^V$ for V the set of vertices. Similarly happens for C_1 chains, made of edges E, C_2 made of faces F, or C_3 made of cells C. d-chains with different dimensions are connected by the boundary operators,

$$\partial_d : C_d \mapsto C_{d-1}, \tag{5.22}$$

which are group homomorphisms, or in other words $\partial_d(e_i^d + e_j^d) = \partial_d(e_i^d) + \partial_d(e_j^d)$. These operators map each d-chain to its boundary, for example they map an edge to its two boundary vertices.

There are also two very important classes of d-chains. First, we have *d-cycles* $z^d \in Z_d \subset C_d$, defined by those d-chains that have no boundary: $\partial_d z^d = 0$. Second, we have *d-boundaries* $b^d \in B_d \subset C_d$, which are themselves the boundary of a $d+1$-chain. In other words,

$$Z_d = \ker \partial_d, \qquad B_d = \operatorname{Im} \partial_{d+1}. \tag{5.23}$$

Since $\partial_d \circ \partial_{d+1} = 0$, boundaries are also cycles $B_d \subset Z_d$. This relation allows defining a homology group structure,

$$H_d := \frac{Z_d}{B_d} = \frac{\ker \partial_d}{\operatorname{Im} \partial_{d+1}}. \tag{5.24}$$

Interestingly, H_d will only depend on the topology of the system. For example, if the manifold is an orientable surface, its Euler characteristic is

$$\chi = V - E + F \tag{5.25}$$

and the genus of the surface is[1]

$$g = 1 - \chi/2. \tag{5.26}$$

The genus is related to H_1 by $H_1 \simeq \mathbb{Z}_2^{2g}$ [41].

5.3.2. Surface code

The surface code was the first topological code proposed [142], and arguably the most promising near-future quantum error-correction code, due to its high threshold [96]. It is also sometimes called the toric code because in its simplest version it is embedded in a torus. While it can be extended to higher dimensions, we will first explain how to describe the $2d$ toric version. The first key component of the code is deciding where to place the qubits, which in this code will correspond to edges. For every 1-chain of qubits c, we can define X and Z Pauli operators

$$X_c = \bigotimes_{i \in c} X_i, \qquad Z_c = \bigotimes_{i \in c} Z_i, \qquad c \in C_1. \tag{5.27}$$

[1] The genus is a topologically invariant of a surface. It indicates the largest number of nonintersecting topologically non-trivial closed curves that the surface can contain.

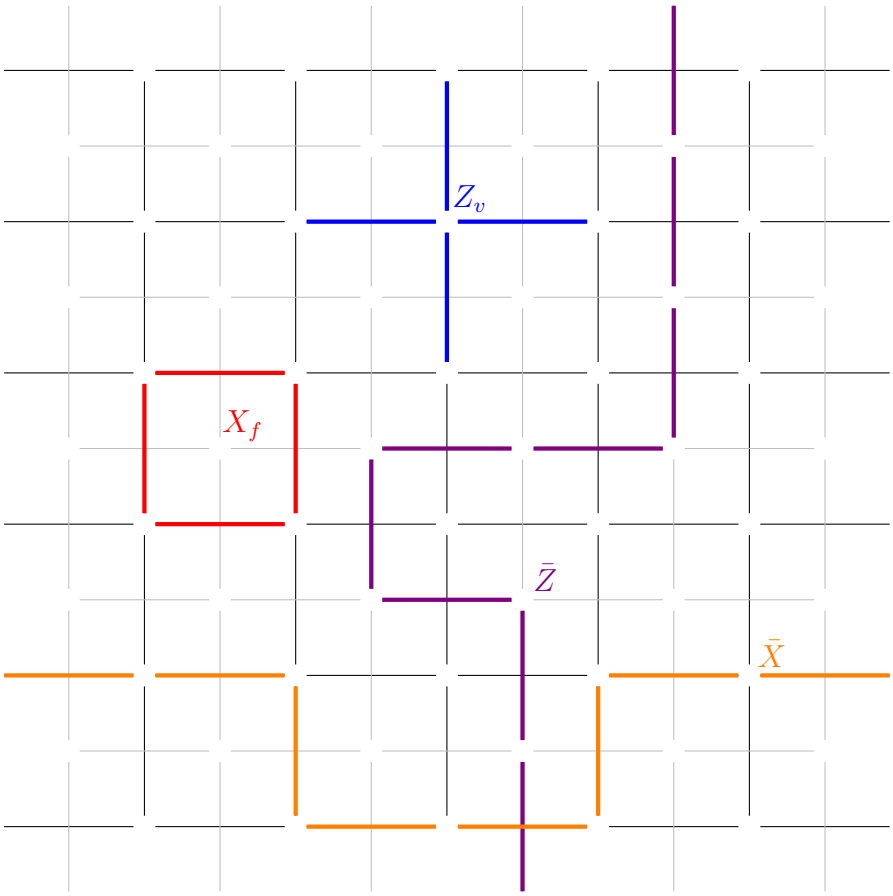


Figure 5.1 **Surface code** of $d = 6$ encoding a single logical qubit with primal (gray) and dual (light gray) lattices. Each edge in the primal or dual lattice is a qubit. Depicted also two stabilizers $X_f, Z_v \in \mathcal{S}$, and representatives of the two logical operator classes $\bar{X}, \bar{Z} \in \mathcal{N}(\mathcal{S})/\mathcal{S}$, which commute with the former because either they share an even number of qubits (eg $\bar{Z}$ and X_f), or because they are composed of the same Pauli operators ($\bar{Z}$ and Z_v). Note how $\bar{Z}$ ($\bar{X}$) starts and finishes in soft (rough) boundaries.

From our discussion in the previous section, we define the code as states in the Homology class H_1. For example, if $z \in Z_1$ a closed curve without boundary, we define a logical qubit state as

$$|\bar{z}\rangle = \sum_{b \in B_1} |z + b\rangle \Rightarrow |\bar{z}\rangle \in H_1. \tag{5.28}$$

This is what gives the surface code its CSS character, identifying B_1 with the C_2 code, and Z_1 with the C_1 code, in the definition of the CSS code. Since topological codes

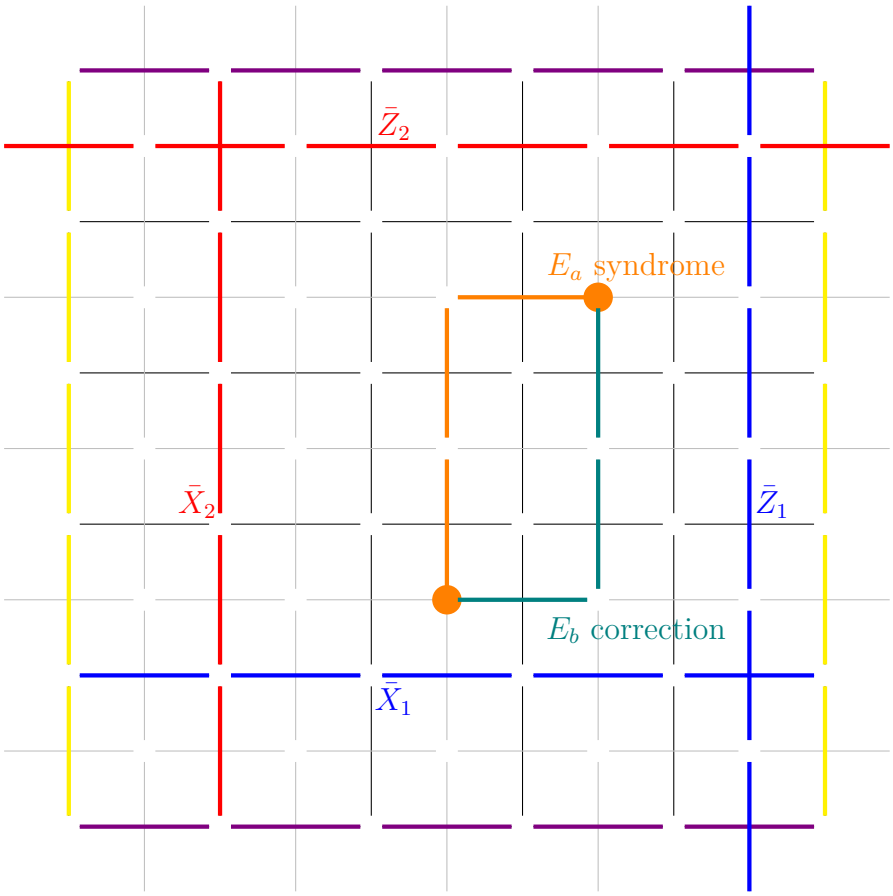


Figure 5.2 **Toric code** with distance $d = 5$. The opposite sides of the lattice are identified. Two logical qubits can be encoded in this code, whose logical Pauli operators are depicted in blue and red. If an error happens, it will leave behind two excited stabilizers (in this case of type X). We can correct it by implementing E_b via one of the shortest paths, such that the probability of a phase or bit flip is minimized. In the depicted case, $E_b E_a \in B_1^*$, so it does not change the logical information.

are also stabilizer codes, we define the stabilizer operators:

$$X_f = \prod_{e \in \partial_2 f} X_e, \qquad Z_v = \prod_{e | v \in \partial_1 e} Z_e, \qquad \mathcal{S} = \{X_f, Z_v\} \tag{5.29}$$

for f indicating a face, and e an edge, and v a vertex, see Fig. 5.1. These stabilizer generators commute with each other because they share an even number of edges, and are subject to the constraints

$$\prod_f X_f = 1, \qquad \prod_v Z_v = 1. \tag{5.30}$$

The number of encoded qubits will be $k = 2g$ because $|H_1| = 2^{2g}$ as we saw at the end of Section 5.3.1, or we may alternatively compute it as the number of qubits minus the number of independent stabilizers, $k = E - (V + F - 2) = 2 - \chi = 2g$.

We can use the transversal Hadamard gate, as described previously, to transform the code into its dual, commonly denoted with an asterisk. Thus, we obtain dual boundary operators

$$\partial_d^* : C_{d-1}^* \mapsto C_d^* \tag{5.31}$$

and dual cycles Z_1^* and boundaries B_1^*. Conjugating by the transversal Hadamard also maps $X_f \to Z_{f^*}$ and $Z_v \to X_{v^*}$. In summary, this gate translates us to the dual code, which in this case is again the surface code. Using both the primal and dual codes, we can explore the form of the logical operators in this code. Our objective is to relate Z_1/B_1 and $\mathcal{N}(\mathcal{S})/\mathcal{S}$. One Pauli operator in the code can be written as

$$A = i^\alpha X_c Z_{c^*}, \qquad (c, c^*) \in C_1 \times C_1^*, \qquad \alpha \in \mathbb{Z} \mod 4 \tag{5.32}$$

This operator will be in $\mathcal{N}(\mathcal{S})$ if it commutes with all the operators in $\mathcal{S}$. In particular

$$[X_c, Z_v] = 0 \iff v \notin \partial_1 c, \qquad [Z_{c^*}, X_{f^*}] = 0 \iff f^* \notin \partial_2 c^*. \tag{5.33}$$

Or in other words, c and c^* do not have boundaries

$$A \in \mathcal{N}(\mathcal{S}) \iff (c, c^*) \in Z_1 \times Z_1^*. \tag{5.34}$$

Thus, the normalizer is composed of any closed string in the primal and dual lattices.

Now that we have characterized the normalizer, let us take a look at the stabilizer group $\mathcal{S}$. An element of the stabilizer group can be written as

$$B = \prod_i X_{f_i} \prod_j Z_{v_j} = X_{\partial_2 c_2} Z_{\partial_1^* c_0^*}. \tag{5.35}$$

for some set of faces $c_2 = \sum_i f_i$ and (dual) vertices $c_0^* = \sum_i v_i^*$. Therefore,

$$B \in \mathcal{S} \iff (c, c^*) \in B_1 \times B_1^*. \tag{5.36}$$

Elements of the stabilizer (plaquette and vertex operators in Fig. 5.2) will deform closed strings without modifying the encoded logical information. Knowing this allows

us to understand that the logical Pauli operators are

$$\frac{\mathcal{N}(\mathcal{S})}{\mathcal{S}} \simeq \frac{Z_1}{B_1} \times \frac{Z_1^*}{B_1^*} = H_1 \times H_1^* \simeq H_1^2. \tag{5.37}$$

In a $d \times d$ torus, this means we have a $[[2d, 2, d]]$ quantum code (vertical and horizontal $\bar{X}$ and $\bar{Z}$ strings), see Fig. 5.2. An alternative way of introducing a non-trivial topology is to create boundaries in the code. There are two types of boundaries, in the primal (soft) and dual (rough) lattices. However, each side of the square can only contain one kind of boundary. For this reason, the number of logical qubits is reduced to just one, and the code will be $[[2d(d-1)+1, 1, d]]$, see Fig. 5.1.

In conclusion, the surface code allows to transversally implement $\bar{X}$ and $\bar{Z}$ as the product of X and Z operators over strings. Using two copies of the surface code, we can also implement a transversal C-Z gate. However, we cannot transversally generate the Clifford group.

Definition 10 (Clifford group and hierarchy [105]) *The Clifford group is the set of quantum gates C such that if P is a Pauli operator, then $C^\dagger PC$ is also a Pauli operator. In other words, it is the normalizer of the Pauli group, $\mathcal{N}(P)$. In general, the d-level of the Clifford hierarchy $\mathcal{L}_d$ is defined as those gates C that map $P \in \mathcal{L}_{d-1}$ to the same level under conjugation,*

$$C \in \mathcal{L}_d \iff \forall P \in \mathcal{L}_{d-1}, C^\dagger PC \in \mathcal{L}_{d-1}, \qquad \mathcal{L}_d := \mathcal{N}(\mathcal{L}_{d-1}) \tag{5.38}$$

where $\mathcal{L}_1 = \langle i\mathbf{1}, Z, X\rangle$ is the Pauli group, and $\mathcal{L}_2 = \langle S, H, C\text{-}Z\rangle$ is the Clifford group itself.

Two well-known families of gates in the Clifford hierarchy are the single-qubit $e^{i\pi/2^d}$ rotations (S-family), and the d-qubit controlled Z gates (C-Z family).

What about higher dimensions? In three dimensions, for example, qubits will be attached to faces, X-type stabilizers to cells, and Z-type stabilizers to edges. Its dual-code will thus have qubits, X-type stabilizers, and Z-type stabilizers attached to edges, vertices, and faces respectively. Furthermore, in this code, it is possible to transversally implement CC-Z gates [245]. If we go up to four dimensions, we will be able to prepare self-correcting quantum memories [4, 85, 119]. Finally, d copies of the d-dimensional surface code will be able to transversally generate, via a mapping to color codes [154], the d-qubit C-Z gate, which is in the $\mathcal{L}_d$ but not in $\mathcal{L}_{d+1}$. This saturates

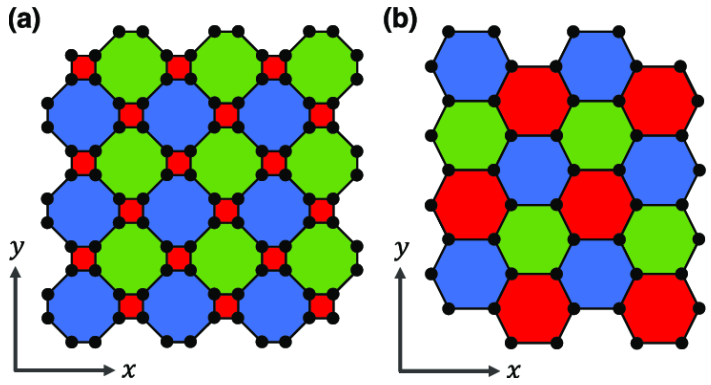


Figure 5.3 **Color codes.** Qubits are placed on the vertices, and stabilizers on the faces. (a) 4-8-8 color code, (b) 6-6-6 color code. Taken from Ref. [160] under CC-BY 4.0 license.

the Bravyi-König theorem[2], which states that d-dimensional topological codes can only generate gates up to the d^{th} level of the Clifford hierarchy [54]. A good review on low overhead implementation of the surface code might be found in the readable Ref. [164].

5.3.3. Color code

A second important family of error correction codes is color codes. As for the surface code, we start the discussion in 2 dimensions. In such a case, their lattice fulfills two key requirements:

1. Each vertex is trivalent.
2. Each face is assigned one of three labels (red, green, and blue), so that neighbor faces have different colors.

In the color code, each qubit is attached to the vertices, as indicated in Fig. 5.5, and the stabilizer generators are face operators

$$X_f = \prod_{v \in f} X_v; \qquad Z_f = \prod_{v \in f} Z_v \tag{5.39}$$

From this equation we can recognise that the color code is self-dual. Similar to (5.30), the stabilizers in the color code are subject to the following constraints

$$\prod_{f \in F_r} X_f = \prod_{f \in F_g} X_f = \prod_{f \in F_b} X_f, \qquad \prod_{f \in F_r} Z_f = \prod_{f \in F_g} Z_f = \prod_{f \in F_b} Z_f. \tag{5.40}$$

[2]As we shall see, due to the Eastin-Knill theorem [89], saturating here means being able to implement one gate from $\mathcal{L}_d$, not any gate.

From this, the number of independent stabilizers generators is $2F-4 = 2(|F_r|+|F_b|+|F_g|)-4$. We compute the number of encoded qubits as the number of physical qubits (vertices, V) minus the number of stabilizer generators ($2F-4$). Given that the lattice is trivalent, $E = \frac{3}{2}V$, and since the Euler characteristic is $\chi = 2(1-g) = V-E+F$, we have

$$k = V - 2F + 4 = V + 4 + 2(V - E - 2(1-g)) = 3V - 2E + 4 - 4 + 4g = 4g, \quad (5.41)$$

doubling the amount of qubits we got in the surface code.

In the color code, there are 6 types of closed string operators, three different colors, and 2 different Pauli operators, X and Z. Denoting by γ a closed string, c the color, and V_c^γ the vertices bounding a c-colored edge in γ, these are

$$X_\gamma^c = \prod_{v \in V_c^\gamma} X_v, \qquad Z_\gamma^c = \prod_{v \in V_c^\gamma} Z_v. \quad (5.42)$$

However, not all string operators are independent, because

$$X_\gamma^r X_\gamma^g X_\gamma^b = \mathbf{1}, \qquad Z_\gamma^r Z_\gamma^g Z_\gamma^b = \mathbf{1}, \quad (5.43)$$

so it is sufficient to consider just two colors. To understand the commutation rules of these string operators, the key is to check whether $X_\gamma^{c'}$ and Z_γ^c coincide in an even or odd number of qubits. If they cross an even number of times $[X_\gamma^{c'}, Z_\gamma^c] = 0$. This necessarily happens, for example, if $c = c'$ [41]. In contrast, if $c \neq c'$ and Z_γ^c and $X_\gamma^{c'}$ cross an odd number of times, $\{X_\gamma^{c'}, Z_\gamma^c\} = 0$.

Let us now analyze the logical operators in this code. Similar to the surface code, if a closed string γ may be contracted to a point via stabilizer generators, then it is a homologically trivial stabilizer element. As such, since only two colors are independent,

$$\mathcal{S} \simeq B_1^{\text{red},X} \times B_1^{\text{red},Z} \times B_1^{\text{green},X} \times B_1^{\text{green},Z}. \quad (5.44)$$

On the other hand, those strings that cannot be contracted but commute with the stabilizers form the normalizer,

$$\mathcal{N}(\mathcal{S}) \simeq Z_1^{\text{red},X} \times Z_1^{\text{red},Z} \times Z_1^{\text{green},X} \times Z_1^{\text{green},Z}. \quad (5.45)$$

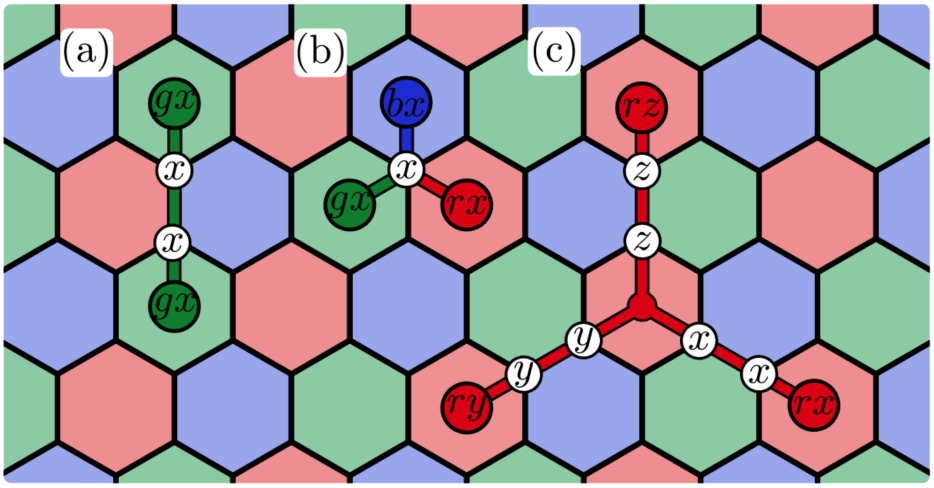


Figure 5.4 **Error strings in the color code**. Similar to Fig. 5.2, the errors should be corrected by an open string of operator joining excitations, so this figure may alternatively be interpreted as fusion rules of excitations. (a) Two excitations of the same color and type X/Z can be joined by a string of the same color. (b) Three excitations of the same type but different colors can be joined at the vertex they share. (c) Three excitations of types X, Z, and XZ, and the same color might also be joined to correct the errors. Figure from Ref. [133] under CC-BY 4.0 license.

And finally, we obtain logical operators $\bar{X}_i$, $\bar{Z}_i$ as

$$\frac{\mathcal{N}(\mathcal{S})}{\mathcal{S}} \simeq H_1^{\text{red},X} \times H_1^{\text{red},Z} \times H_1^{\text{green},X} \times H_1^{\text{green},Z}. \tag{5.46}$$

As such, the color code is a CSS code. If closed strings γ are elements of the normalizer of the stabilizer group $\mathcal{N}(\mathcal{S})$, open strings represent errors and whose boundaries will signal a $+1$ syndrome, see Fig. 5.4. These errors E_a can be corrected closing the string with $E_b^\dagger = E_b$, so hopefully $E_b^\dagger E_a \in \mathcal{S}$ and the logical information is preserved.

To compute the distance of a color code, we have to find the smallest non-detectable error, or in other words, the smallest weight between the logical operators. However, in this case, non-trivial elements of the normalizer do not need to be single-color strings. Rather, the string may branch into a string net. On the other hand, similarly to the surface code, we may introduce boundaries of a given color by removing stabilizer operators of such color. Similar to the surface code, red close string operators γ may end up in red boundaries, making them commutative with any stabilizer and therefore an element of $\mathcal{N}(\mathcal{S})$. In contrast, if a red closed string γ only encloses red boundaries, then it is part of the stabilizer group $\mathcal{S}$, as it can be contracted to the boundary [41].

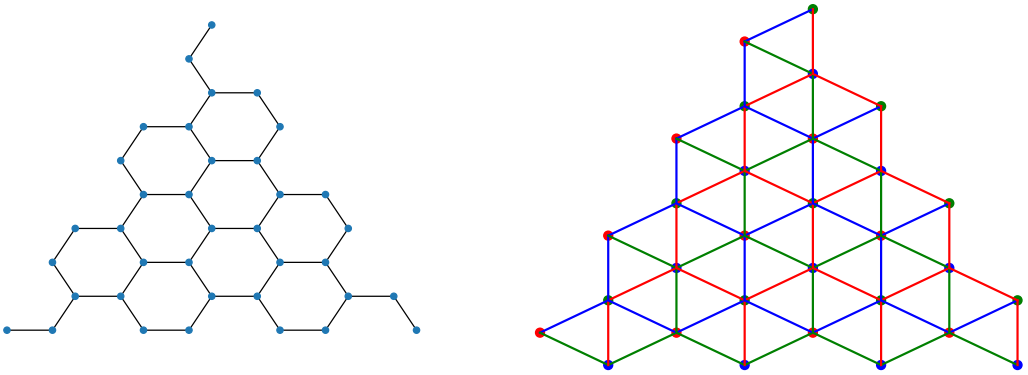


Figure 5.5 **6-6-6 Color code, primal and dual lattices.** Left: trivalent primal lattice, with qubits on the vertices. Right: dual lattice, with qubits on the faces. In this dual lattice, each edge can be assigned the complementary color of the vertices bounding it. To introduce boundaries allowing to encode logical qubits, we remove the stabilizers corresponding to the bottom blue vertices, upper-right green vertices, and upper-left red vertices.

We have already seen that the color code encodes twice the number of logical qubits than the equivalent surface code. However, the most remarkable difference between these codes is the color code capability to transversally implement the

$$S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \tag{5.47}$$

and Hadamard gates, which together with the C-Not allow these codes to generate the Clifford group. Transversal implementation of gates is very useful because it prevents errors from propagating in the code.

The Hadamard gate can be easily implemented on the triangular codes such as Fig. 5.5 and Fig. 5.6, because both $\bar{X}$ and $\bar{Z}$ strings are applied over the same physical qubits. As such, $H^{\otimes n}\bar{X}H^{\otimes n} = \bar{Z}$, and consequently, we can take $\bar{H} = H^{\otimes n}$. As for the $\bar{S}$ gate, the desired behavior would be $\bar{S}^\dagger\bar{Z}\bar{S} = \bar{Z}$ and $\bar{S}^\dagger\bar{X}\bar{S} = \bar{Y}$. However, to transversally implement this gate we need to make sure that $\bar{S} = S^{\otimes n}$ does not take us outside the code space. Instead, we find that $(S^{\otimes n})^\dagger X_f S^{\otimes n} = i^v X_f Z_f$, where face f has v vertices [46]. Staying in the $+1$ subspace, so the stabilizer group is preserved, is recovered for codes where the number of qubits per face is a multiple of 4, like the 4-8-8 code in Fig. 5.3. Furthermore, if the number n of qubits in the code is such that

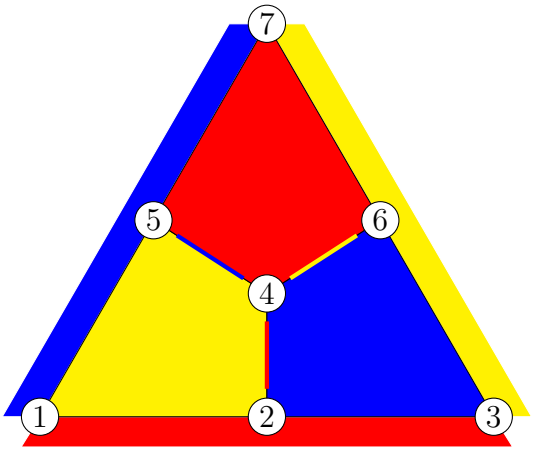


Figure 5.6 **Simplest color code** of distance $d = 3$, with boundaries. Operator X on vertices 2, 4, 5, and 6 would form a string net with the three colored edges. A weight-three logical operator would be string net red boundary – 2– 4 –7 – blue and yellow boundaries.

$n = 1 \mod 4$, then we obtain

$$S^{\otimes n}\bar{X}(S^\dagger)^{\otimes n} = i^{n \mod 4}\bar{X}\bar{Z} = i\bar{X}\bar{Z} = \bar{Y}, \qquad S^{\otimes n}\bar{Z}(S^\dagger)^{\otimes n} = \bar{Z}. \tag{5.48}$$

Thus, in this code, we can implement the $\bar{S} = S^{\otimes n}$ transversally, and consequently, also the entire Clifford group.

If we have a member from the 6-6-6 color code family, Fig. 5.5 we can still find a way to transversally implement the $\bar{S}$ rotation [151]. The key aspect to notice is that the set of vertices can be split in two disjoint sets S_1 and S_2 under the condition that two vertices in an edge are part of different sets and $|S_1| = |S_2| + 1$. Since the number of qubits is odd, but each stabilizer is applied to an even number of qubits, we can choose $\bar{S} = S^{\otimes(q\in S_1)}(S^\dagger)^{\otimes(q\in S_2)}$. Then [151],

$$\bar{S}\bar{X}\bar{S}^\dagger = i^{|S_1|-|S_2|}\bar{X}\bar{Z} = i\bar{X}\bar{Z} = \bar{Y}, \qquad \bar{S}\bar{Z}\bar{S}^\dagger = \bar{Z}. \tag{5.49}$$

The stabilizer subspace will similarly be preserved because in each stabilizer there is an equal number of qubits in S_1 and S_2 [151]

$$\bar{S}X_f\bar{S}^\dagger = i^{|S_1\cap f|-|S_2\cap f|}X_fZ_f = X_fZ_f \in \mathcal{S}; \qquad \bar{S}Z_f\bar{S}^\dagger = Z_f \in \mathcal{S}. \tag{5.50}$$

Thus, a transversal gate is also possible in the 6-6-6 codes if we take the members of the family depicted in Fig. 5.5.

Similar to the extension of surface codes we saw in the previous Section 5.3.2, we can also generalize the surface code to a family of color codes of d dimensions [47]. In general, the conditions will be

1. The lattice is d-valent: each vertex is connected to d edges.
2. d-cells can be $d+1$-colored such that no adjacent d-cells share color.

We depict the simplest three-dimensional color code in Fig. 5.7. In this case, the qubits are still on the vertices, and the Z stabilizers are still attached to faces, but the X stabilizers are instead cell stabilizers X_c. The attractiveness of the 3d color code is its capability to transversally implement the gate

$$T = \sqrt{S} = \begin{pmatrix} 1 & 0 \\ 0 & i^{1/2} \end{pmatrix} \tag{5.51}$$

If each stabilizer has weight multiple of 8, and $n \mod 8 =: l \in \{1,3,5,7\}$ then $\bar{T} = T^{\otimes n}$ implements the logical operator

$$\bar{T}\,|\bar{0}\rangle = |\bar{0}\rangle\,; \qquad \bar{T}\,|\bar{1}\rangle = i^{l/2}\,|\bar{1}\rangle\,, \tag{5.52}$$

and maps the code subspace to itself. A similar construction might be implemented for the d-dimensional color code, with gate

$$R_d = \begin{pmatrix} 1 & 0 \\ 0 & 2^{\frac{i\pi}{2^d}}, \end{pmatrix} \tag{5.53}$$

thus saturating the Bravyi-König bound [54, 197, 252]. In such color codes, the Z stabilizers are attached to faces and X stabilizers to d-cells. However, this arrangement does not allow us to transversally implement the Hadamard gate anymore, as X and Z stabilizers no longer have the same support. Finally, it is worth mentioning that, similarly to surface codes, in $d = 4$ we similarly find a self-correcting quantum code [45].

5.4. Towards fault-tolerant quantum computing

So far we have discussed two of the main families of error-correcting codes, the surface and color codes. Unfortunately, we have not found any code that allows us to fault-tolerantly and transversally implement a set of universal gates, $\{H, \text{C-}Z, T\}$. Is this possible? It turns out that the answer is negative: according to a theorem by

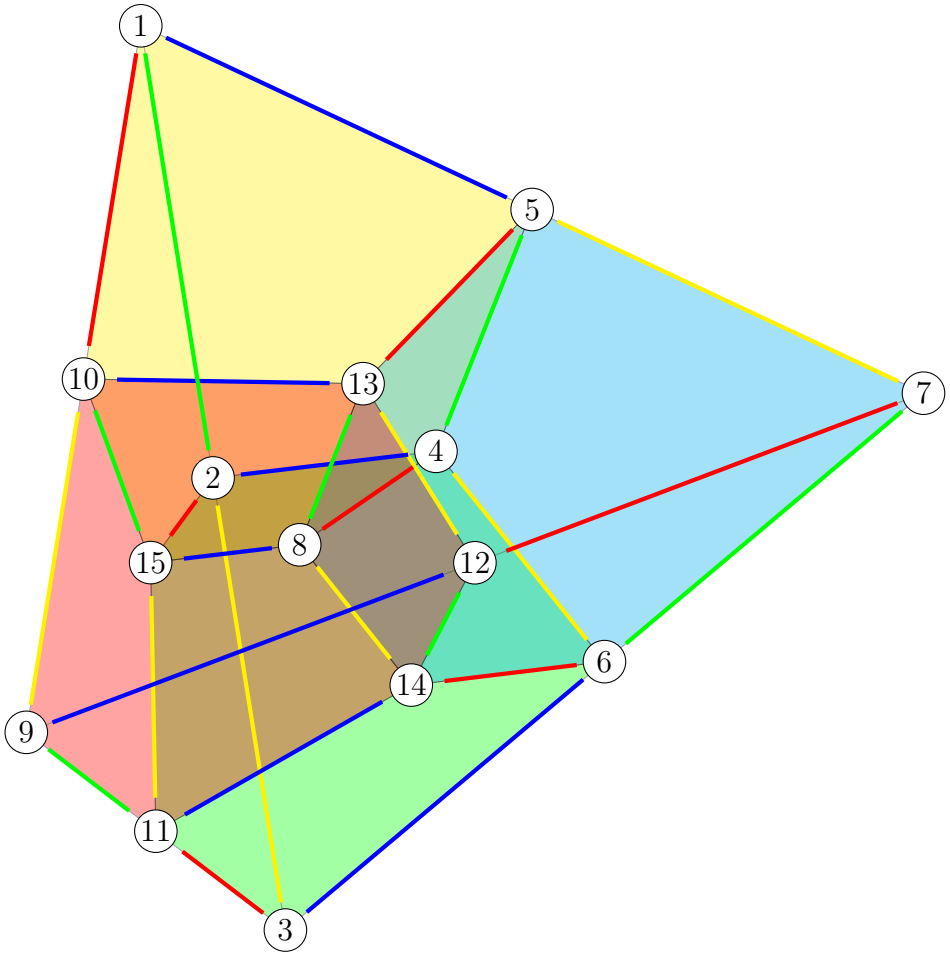


Figure 5.7 **Simplest example of the 3d color code**, similar to Fig. 5.6. X stabilizers are attached to cells, while Z stabilizers do so in faces. Boundaries must be placed around the code, such that they are of a different color than the cells they touch. This color code admits transversal implementation of C-Not and T gates.

Eastin and Knill, any transversal set of gates generates a finite group and therefore cannot be universal [89].

What alternatives do we have? The most popular option is to couple a high-threshold error code, such as the surface code, with a smaller stabilizer code that is only used to generate magic states $T|+\rangle$. Each of these states will, later on, be consumed to implement one T gate on the surface code via quantum teleportation [53]. To generate these magic states, it is customary to implement a distillation procedure: starting with many noisy copies implemented via non-fault tolerant T gates, one can generate a few cleaner copies. The most famous of these distillation procedures is known as the 15 to 1 protocol [53], which takes 15 noisy copies and outputs a single

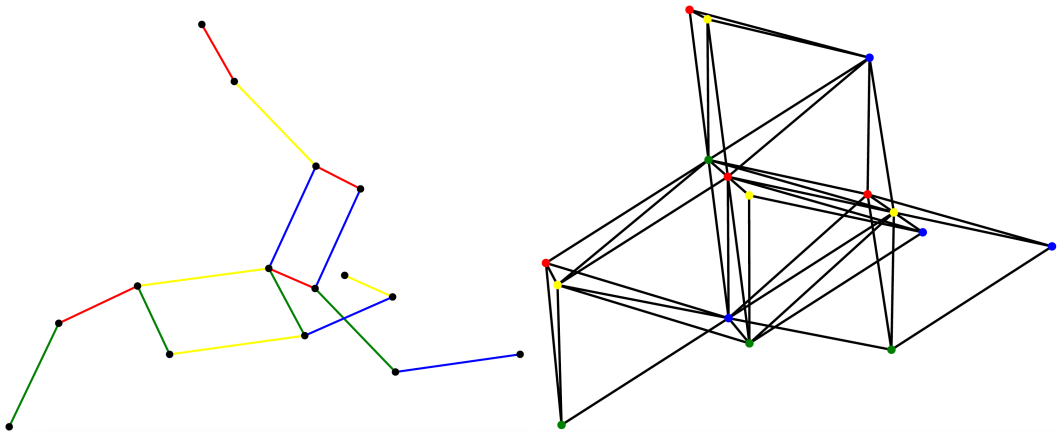


Figure 5.8 **Three-dimensional color code with boundaries, primal and dual lattices,** generalization of Fig. 5.5. Left: 4-valent primal lattice, with qubits on the vertices. Edges are shown with the complementary color of the vertices in the corresponding dual face. Right: dual lattice, with qubits on the cells. Each face can be assigned the complementary color of the vertices bounding it. To introduce boundaries allowing to encode logical qubits, we remove the stabilizers corresponding to the boundary k-faces, see (5.54).

one with less error. In particular, assuming Pauli error p, the output state will be subject to error $35p^3$.

Magic state distillation is rather costly, and while there have been many advances since the early protocols, they are still often considered the bottleneck step to implementing fault-tolerant quantum algorithms. A recent review of several distillation methods may be found in Ref. [165]. The alternative approach is using subsystem codes, two codes that can implement different sets of gates but are based on the same lattice and encode the same number of logical qubits. The most prominent example is gauge color codes, which we briefly review next.

5.4.1. Gauge color codes

Gauge color codes are a generalization of color codes, first introduced by Héctor Bombín in Ref. [42]. Here we will mostly follow their description by Ref. [151], that more naturally describes them in the dual picture in Fig. 5.5 over a lattice $\mathcal{L}$. Such lattices are a collection of d-simplices δ, defined using independent vertices e_i,

$$\delta = \left\{ \sum_{i=0}^{d} t_i e_i \Big| \sum_i t_i = 1 \right\}, \quad \dim \delta := d, \quad t_i \in \mathbb{R}^+, \quad e_i = (\underbrace{0, \ldots, 0}_{i-1}, 1, 0, \ldots, 0). \tag{5.54}$$

For example, 0-simplices are vertices, 1-simplices are edges, 2-simplices are faces (triangles), and 3-simplices are cells (tetrahedra). The code is the union of a set of d-simplices of the same dimension $\mathcal{L} = \cup_i \delta_i$, and $\dim \mathcal{L} = \dim \delta_i$ is the dimension of the color code.

If we have a d-simplex δ, we can also define its k-faces [151]

$$\Delta_k(\delta) = \{\sigma \subset \delta | \sigma \text{ is a } k\text{-simplex, with } k \leq d\}. \tag{5.55}$$

Each d-simplex contains $\binom{d+1}{k+1}$ such k-faces. Additionally, we will denote by $\partial\mathcal{L}$ the boundary of $\mathcal{L}$, and $\Delta'_k(\mathcal{L}) := \Delta_k(\mathcal{L}\backslash\partial\mathcal{L})$. Color codes, as we have discussed, are also characterized by a function, called 'color' that maps each vertex to $\mathbb{Z}_{d+1}$,

$$\text{color} : \Delta_0\mathcal{L} \mapsto \mathbb{Z}_{d+1}. \tag{5.56}$$

The possible values in $\mathbb{Z}_{d+1}$ are understood as different colors. We can also assign colors to a d-simplex as the union of colors of its vertices

$$\text{color}(\delta) = \bigsqcup_{v \in \Delta_0(\delta)} \text{color}(v). \tag{5.57}$$

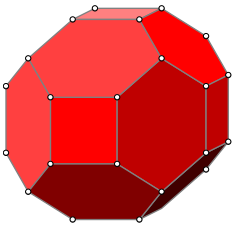
Apart from $\mathcal{L}$ being $(d+1)$ colorable, we need $\mathcal{L}$ to be the triangulation of a d-simplex. Then, we attach a qubit to each d-simplex in the interior of $\mathcal{L}$, such that for $\sigma \subset \mathcal{L}\backslash\partial\mathcal{L}$, we define the set of qubits connected to σ as

$$\mathcal{Q}(\sigma) = \{\delta \in \Delta_d(\mathcal{L}) | \sigma \subset \delta\}. \tag{5.58}$$

For example, in the two-dimensional color code from the dual lattice in Fig. 5.5, δ would refer to the triangles and σ to the edges or vertices. More concretely, to each edge, there are two triangles/qubits attached, and to each vertex, there are six, except in the border. When we say an operator is supported over a simplex σ, we mean that it is applied over the corresponding qubits, e.g., $X(\sigma) = X[\mathcal{Q}(\sigma)]$.

Color codes are also CSS subsystem stabilizer codes, a generalization of stabilizer codes in which code generators do not need to commute [153]. They are defined by their gauge group [42]:

$$\mathcal{G} = \langle X(\delta), Z(\sigma) | \forall \delta \in \Delta'_{d-2-z}(\mathcal{L}), \forall \sigma \in \Delta'_{d-2-x}(\mathcal{L}) \rangle \,, \tag{5.59}$$


$= Z_S = X_S \in \mathcal{S}$

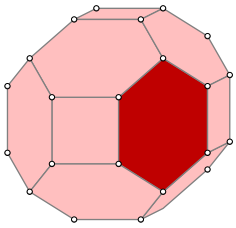

$= Z_G = X_G \in \mathcal{G}$

Figure 5.9 **Gauge elements in three-dimensional gauge color code** $CC_{\mathcal{L}}(x = 0, z = 0)$, depicted in the primal lattice $\mathcal{L}^*$. In the primal lattice depicted, qubits are attached to vertices, gauge operators to faces (right figure), and stabilizer operators to cells (left figure), as explained in the main text. Figure similar to those appearing in Ref. [56], generating code modified from the one courteously provided by Prof. Benjamin Brown.

with[3] $x+z \leq d-2$. This induces a stabilizer group over the gauge group, $\mathcal{S} = \mathcal{Z}_{\mathcal{G}}(\mathcal{G}) \subset \mathcal{G}$ [42],

$$\mathcal{S} = \langle X(\delta), Z(\sigma) | \forall \delta \in \Delta'_x(\mathcal{L}), \forall \sigma \in \Delta'_z(\mathcal{L}) \rangle \,. \tag{5.60}$$

We refer to such color code as the $CC_{\mathcal{L}}(x, z)$. For instance, in 2 dimensions there is only one color code on $\mathcal{L}$, $CC_{\mathcal{L}}(0, 0)$. Stabilizer generators are attached to vertices ($\Delta_0(\mathcal{L})$) and coincide with the gauge generators, whereas physical qubits are attached to faces ($\delta \in \Delta_{d=2}(\mathcal{L})$). The support of a stabilizer operator on vertex σ are those qubits whose faces contain the vertex σ, i.e. $\mathcal{Q}(\sigma)$. Any color code for which $\mathcal{G} = \mathcal{S}$ is a stabilizer color code. Else, it is a subsystem color code.

Let us give two more examples: the three-dimensional color code with X stabilizers attached to vertices and Z stabilizers attached to edges is the stabilizer color code $CC_{\mathcal{L}}(0, 1)$. On the other hand, the $CC_{\mathcal{L}}(0, 0)$ subsystem color code has, in the primal (dual) lattice $\mathcal{L}^*$ ($\mathcal{L}$), gauge generators attached to faces (edges) and stabilizer generators attached to cells (vertices), see Fig. 5.9. Since two faces (edges) may share an odd number of qubits, gauge stabilizers do not need to commute. In three dimensions, there are four edges arriving at each vertex, so the corresponding faces do not need to share an edge.

Code words in CSS codes are those that lie in the $+1$ eigenspace of the element of the stabilizer. Furthermore, two bit-strings are equivalent if they are connected by an element of $\mathcal{G} \backslash \mathcal{S}$ [151]. As a consequence, one may formally decompose the space as a tensor product of logical and gauge qubits: $|\psi\rangle |g\rangle$. This allows to distinguish two types of logic gates. *Bare* logical operators $\bar{U}_{\text{bare}} : |\psi\rangle |g\rangle \mapsto (\bar{U} |\psi\rangle) |g\rangle$ do not

[3] The $d-2$ condition is due to the minimal dimension of color codes being $d = 2$.

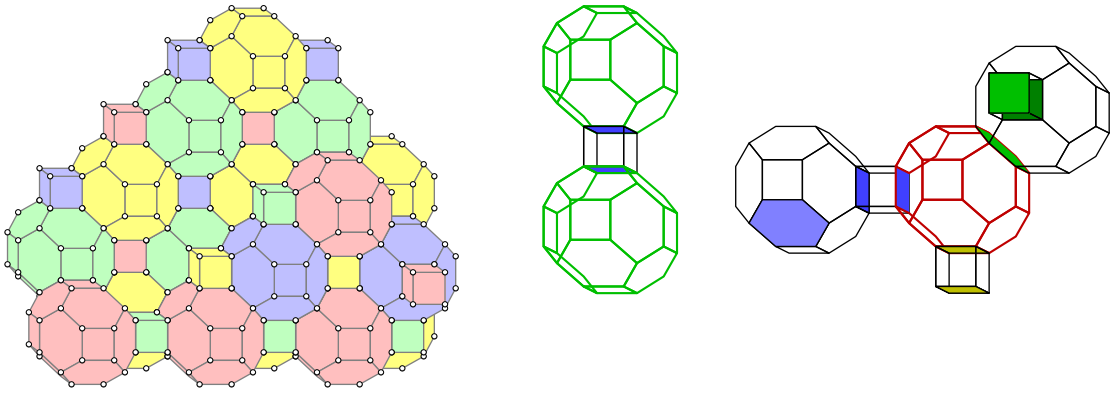


Figure 5.10 **Gauge color code**. Left: structure of the primal lattice $\mathcal{L}^*$. The lattice is composed of truncated octahedrons and cubes, colored with four possible values. Qubits lie in the vertices. Right: Errors in the color code form either a single string, of color blue-green as in the first figure; or a stringnet such as the one shown in the second figure of colors red-blue, red-yellow and red-green. Strings and stringnets may end in a boundary. Each colored face indicates a -1 gauge face measurement. Furthermore, at each cell, the product of the gauge measurements of each color gives us repeated syndrome measurements that enable single-shot error correction [43]. Figures similar to those appearing in Ref. [56], generating code modified from the one courteously provided by Prof. Benjamin Brown.

affect the gauge qubits. We denote by $\bar{X} = X^{\otimes n}$ and $\bar{Z} = Z^{\otimes n}$ the bare logical Pauli operators X and Z [151]. In contrast, *dressed* logical may also affect them, $\bar{U}_{\text{dressed}} : |\psi\rangle |g\rangle \mapsto (\bar{U} |\psi\rangle) |g'\rangle$. We then define the logical states as

$$|\bar{0}\rangle |g_x\rangle := \sum_{X_G \in \mathcal{G}} X_G |0\rangle^{\otimes n}, \qquad |\bar{1}\rangle |g_x\rangle := \bar{X} |\bar{0}\rangle |g_x\rangle, \tag{5.61}$$

where we have used that elements of the gauge group do not change the logical information, and $|\bar{0}\rangle |g_x\rangle$ and $|\bar{1}\rangle |g_x\rangle$ are $+1$ eigenstates of the $X_G \in \mathcal{G}$ generators [151]. Alternatively, we can encode the logical qubits using

$$|\bar{0}\rangle |g_z\rangle := \sum_{X_S \in \mathcal{S}} X_S |0\rangle^{\otimes n}, \qquad |\bar{1}\rangle |g_z\rangle := \bar{X} |\bar{0}\rangle |g_z\rangle, \tag{5.62}$$

with $|\bar{0}\rangle |g_z\rangle$ and $|\bar{1}\rangle |g_z\rangle$ are $+1$ eigenstates of the $Z_G \in \mathcal{G}$ generators, as $Z_G X_S |0\rangle^{\otimes n} = X_S Z_G |0\rangle^{\otimes n} = X_S |0\rangle^{\otimes n}$ [151].

Apart from Pauli $\bar{X}$ and $\bar{Z}$, we want to know what gates we can implement transversally. C-Z is transversal because it is a CSS code and can therefore be

implemented physical-qubit-wise between two copies of the code. Similarly, if the code is self-dual, $CC_{\mathcal{L}}(x,x)$, then $\bar{H} = H^{\otimes n}$ transforms the $\bar{X}$ into $\bar{Z}$ as strings have support over the same physical qubits. However, it is also important to notice that this operator is a dressed logical operator, as it exchanges $X_G \leftrightarrow Z_G$ [151].

Finally, to complete a universal set of gates, we want to explore the possibility of applying $\bar{R}_\ell$, defined in (5.53). As we did for the 6-6-6 two-dimensional color code, we divide the physical qubits in two disjoint sets $S_1 \cup S_2 = \mathcal{Q}(\mathcal{L})$ such that the logical gate is defined as $\bar{R}_\ell = (R_\ell^k)^{\otimes(q\in S_1)}(R_\ell^{-k})^{\otimes(q\in S_2)}$, with $k \in \{1,\dots,2^n-1\}$ ensuring

$$k(|S_1| - |S_2|) = 1 \mod 2^\ell. \tag{5.63}$$

Finding such a k is always possible, [151, Page 7] and it will implement the desired behavior. On the other hand, though, we want to make sure that $\bar{R}_\ell$ commutes with gauge operators, so we need [151]

$$\forall X_G \in \mathcal{G}, \quad |S_1 \cap \mathcal{Q}(X_G)| = |S_2 \cap \mathcal{Q}(X_G)| \mod 2^\ell. \tag{5.64}$$

In the two-dimensional color code, this condition meant that stabilizer/gauge operators had equal support over S_1 and S_2 elements. By imposing this constraint, we are making sure that $\bar{R}_\ell$ will be a bare logical operator. One way of enforcing (5.64) is to satisfy that, for any subset of gauge operators $\{X_{G_1},\dots,X_{G_m}\}$, [42, Equation 3] and [151, Lemma 6]

$$\left|S_1 \bigcap_{i=1}^{m} \mathcal{Q}(X_{G_i})\right| = \left|S_2 \bigcap_{i=1}^{m} \mathcal{Q}(X_{G_i})\right| \mod 2^{\ell-m+1}. \tag{5.65}$$

Since gauge operators X_{G_i} are attached to $d-2-z$ simplices δ_i, we can analyze the qubits to which they are attached. Let

$$\bigcap_{i=1}^{m} \mathcal{Q}(\delta_i) = \mathcal{Q}(\tau) \text{ or } \emptyset. \tag{5.66}$$

If the intersection is not an empty set, then $\cup_i \delta_i \subset \tau \in \mathcal{L}\backslash\partial\mathcal{L}$. Now, if $\dim\tau = m < d$, we have [151, Lemma 7]

$$|S_1 \cap \mathcal{Q}(\tau)| = |S_2 \cap \mathcal{Q}(\tau)|\,, \tag{5.67}$$

and we can use this fact to prove (5.65):

$$\left|S_1 \bigcap_{i=1}^{m} \mathcal{Q}(X_{G_i})\right| - \left|S_2 \bigcap_{i=1}^{m} \mathcal{Q}(X_{G_i})\right| = |S_1 \cap \mathcal{Q}(\tau)| - |S_2 \cap \mathcal{Q}(\tau)| = 0. \tag{5.68}$$

In summary, to transversally implement $\bar{R}_\ell$ one possibility is to ensure that the union of up to ℓ gauge operators is contained in a simplex τ of dimension smaller than d. Let us see what this entails with respect to the coloring. We know that $d \geq \text{color}(\tau) \geq \text{color}(\cup_i \delta_i) \geq m(d-1-z)$, because each c-simplex is colored with $c+1$ colors. Taking $m = \ell$, we find that

$$\ell \leq \frac{d}{d-1-z} \tag{5.69}$$

Since $x + z \leq d-2$, ℓ is maximized for $z = d-2$, which makes self-duality impossible but ensures that $\ell = \frac{d}{d-1-z} = d$. Therefore, we have seen that self-dual gauge color codes allow transversal implementation of $\bar{H}$, and $CC_{\mathcal{L}}(0, d-2)$ that of $\bar{R}_d$, but both conditions are incompatible for $d > 2$, as would be expected from Eastin-Knill theorem [89]. We can also arithmetically check that we fail to simultaneously obtain self-duality and transversal implementation of $\bar{R}_3$ in any dimension, as we would need

$$x = z \geq \frac{2d-3}{3} \Rightarrow x + z \geq \frac{4}{3}d - 2 > d - 2 \Rightarrow\!\!\!\!\!\!\not\leq x + z < d - 2. \tag{5.70}$$

There is a way out, however, called *gauge fixing*, which helps connect the self-dual codes $CC_{\mathcal{L}}(z, z)$, where we can transversally apply $\bar{H}$, and $CC_{\mathcal{L}}(0, d-2)$, where we can transversally apply $\bar{R}_d$.

To understand how it works, consider the three-dimensional case $CC_{\mathcal{L}}(0, 0)$ and $CC_{\mathcal{L}}(0, 1)$. In the former, stabilizer operators are applied over cells in $\mathcal{L}^*$, while in the latter, $Z_S \in \mathcal{S}$ stabilizers are applied over faces. This means imposing additional constraints over the Z measurement of such faces, over the subsystems code $CC_{\mathcal{L}}(0, 0)$. Thus, the gauge fixing procedure checks such face stabilizers and corrects them, so they are in the $+1$ subspace of the Z face stabilizers of $CC_{\mathcal{L}}(0, 1)$. In other words, if the gauge qubits are in the state $|g_z\rangle$, we can implement $\bar{R}_3$ transversally. However, when we apply $\bar{H} = H^{\otimes n}$ we transform:

$$|\psi\rangle |g_z\rangle \mapsto (\bar{H} |\psi\rangle) |g_x\rangle . \tag{5.71}$$

Gauge fixing immediately after (measuring the face stabilizers and correcting them) ensures we can recover the $|g_z\rangle$ state necessary to implement any further $\bar{R}_3$ gates. The entire procedure can be applied transversally:

$$|\psi\rangle |g_z\rangle \xmapsto{H^{\otimes n}} (\bar{H} |\psi\rangle) |g_x\rangle \xmapsto{\text{gauge fix}} (\bar{H} |\psi\rangle) |g_z\rangle . \tag{5.72}$$

It is possible to use gauge fixing between $CC_{\mathcal{L}}(x, z)$ and $CC_{\mathcal{L}}(x', z')$ as long as they encode the same number of logical qubits, and $x \geq x'$ and $z \geq z'$. In that case $\mathcal{S}' \subseteq \mathcal{S}$ and $\mathcal{G} \subseteq \mathcal{G}'$ [151]. Gauge color codes are also interesting because they allow for so-called 'single-shot error correction': instead of measuring high-weight stabilizer operators in 3 dimensions, we can measure the gauge operators, from which the stabilizer operators' eigenvalues can be deduced. Furthermore, since each cell contains faces of different colors, we automatically recover several parity checks that allow us to identify measurement errors [43]. Fascinatingly, one may also create gauge subsystem codes and achieve single-shot error corrections from the toric code [152]. Finally, (gauge) color codes allow switching back and forth between 2 and 3 dimensions, instead of between two three-dimensional color codes [44]. This procedure is called *code-switching* and similarly allows for a universal and transversal set of gates. Though beautiful, in the high physical error regime magic state distillation has still been found to offer an advantage over this procedure [38].

5.5. Concluding thoughts

In this chapter, we have reviewed the basics of quantum error correction, with special attention to the most popular topological stabilizer codes: the surface and color codes. Not only that, but we have built our way up and explained how to achieve fault tolerance via gauge fixing in the latter. However, there are many topics we have not discussed. For example, there are many other codes [82], some with the same geometry as some superconducting quantum chips [100], others specifically designed for photons [106, 194, 241] which can be embedded in full fault-tolerant quantum computing architectures [25, 49]. Alternatively, there have also been quantum computing proposals that make use of the topological properties of Majorana fermions [166].

However, what is most exciting is that just a few years after Google's quantum supremacy experiment [13, 267, 172] we are now approaching the physical error required to start implementing quantum error correction. There have been several experiments approaching the error-correction threshold [149, 212, 202, 266, 2], which open the door to the implementation of many fault-tolerant quantum algorithms discussed in this thesis. In conclusion, we can only say that we are living in an exciting time for quantum computing.

5.6. Results

- We have explained the basics of classical and quantum error correction, with a focus on Stabilizer and CSS codes.
- We have presented the family of Topological error-correcting codes, and their two well-known members: the surface and color codes. We have indicated how both encode logical qubits and how to implement logic gates.
- We have also described how Clifford gates can be efficiently implemented in a color code, but also that non-Clifford gates require either magic state distillation or gauge fixing techniques. Further, we have explained that the theorem of Easting and Knill forbids the existence of a topological code with a set of universal and transversal logic gates, even if do not restrict ourselves to three dimensions [89].
- We have described the gauge color codes as a way to achieve fault tolerance, via the procedure of gauge fixing. This method allows the implementation of non-Clifford gates in the 3d color code, while the rest of the operations are implemented in its two-dimensional version.
- We are working on a Machine Learning decoder, following work previously carried out in our group.

Bibliography

[1] Aaronson, S. and Ambainis, A. (2009). The need for structure in quantum speedups. *arXiv preprint arXiv:0911.0996*.

[2] Acharya, R., Aleiner, I., Allen, R., Andersen, T. I., Ansmann, M., Arute, F., Arya, K., Asfaw, A., Atalaya, J., Babbush, R., et al. (2022). Suppressing quantum errors by scaling a surface code logical qubit. *arXiv preprint arXiv:2207.06431*.

[3] Alcaide, E. (2019). Minifold: a deeplearning-based mini protein folding engine. https://github.com/EricAlcaide/MiniFold/.

[4] Alicki, R., Horodecki, M., Horodecki, P., and Horodecki, R. (2010). On thermal stability of topological qubit in kitaev's 4d model. *Open Systems & Information Dynamics*, 17(01):1–20.

[5] Ambainis, A. (2004). Quantum walk algorithm for element distinctness. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '04, page 22–31, USA. IEEE Computer Society.

[6] Ambainis, A. (2010). Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations. *arXiv preprint arXiv:1010.4458*.

[7] Ambainis, A., Gilyén, A., Jeffery, S., and Kokainis, M. (2020). Quadratic speedup for finding marked vertices by quantum walks. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 412–424.

[8] Anand, A., Schleich, P., Alperin-Lea, S., Jensen, P. W., Sim, S., Díaz-Tinoco, M., Kottmann, J. S., Degroote, M., Izmaylov, A. F., and Aspuru-Guzik, A. (2022). A quantum computing view on unitary coupled cluster theory. *Chemical Society Reviews*.

[9] Anfinsen, C. B. (1973). Principles that govern the folding of protein chains. *Science*, 181(4096):223–230.

[10] Anfinsen, C. B., Haber, E., Sela, M., and White Jr, F. (1961). The kinetics of formation of native ribonuclease during oxidation of the reduced polypeptide chain. *Proceedings of the National Academy of Sciences of the United States of America*, 47(9):1309.

[11] Apers, S. and Sarlette, A. (2019). Quantum fast-forwarding: Markov chains and graph property testing. *Quantum Information & Computation*, 19(3–4):181–213.

[12] Arrazola, J. M., Di Matteo, O., Quesada, N., Jahangiri, S., Delgado, A., and Killoran, N. (2022). Universal quantum circuits for quantum chemistry. *Quantum*, 6:742.

[13] Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J. C., Barends, R., Biswas, R., Boixo, S., Brandao, F. G., Buell, D. A., et al. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510.

[14] Augustino, B., Nannicini, G., Terlaky, T., and Zuluaga, L. F. (2021). An infeasible-inexact quantum interior point method for convex quadratic symmetric cone optimization.

[15] Babai, L. (2016). Graph isomorphism in quasipolynomial time. In *Proceedings of the forty-eighth annual ACM Symposium on Theory of Computing*, pages 684–697.

[16] Babbush, R., Berry, D. W., McClean, J. R., and Neven, H. (2019). Quantum simulation of chemistry with sublinear scaling in basis size. *npj Quantum Information*, 5(1):1–7.

[17] Babbush, R., Berry, D. W., Sanders, Y. R., Kivlichan, I. D., Scherer, A., Wei, A. Y., Love, P. J., and Aspuru-Guzik, A. (2017). Exponentially more precise quantum simulation of fermions in the configuration interaction representation. *Quantum Science and Technology*, 3(1):015006.

[18] Babbush, R., Gidney, C., Berry, D. W., Wiebe, N., McClean, J. R., Paler, A., Fowler, A., and Neven, H. (2018a). Encoding electronic spectra in quantum circuits with linear t complexity. *Physical Review X*, 8(4):041015.

[19] Babbush, R., McClean, J. R., Newman, M., Gidney, C., Boixo, S., and Neven, H. (2021). Focus beyond quadratic speedups for error-corrected quantum advantage. *PRX Quantum*, 2(1):010103.

[20] Babbush, R., Perdomo-Ortiz, A., O'Gorman, B., Macready, W., and Aspuru-Guzik, A. (2012). Construction of energy functions for lattice heteropolymer models: a case study in constraint satisfaction programming and adiabatic quantum optimization. *arXiv preprint arXiv:1211.3422*.

[21] Babbush, R., Wiebe, N., McClean, J. R., McClain, J., Neven, H., and Chan, G. K.-L. (2018b). Low-depth quantum simulation of materials. *Physical Review X*, 8(1):011044.

[22] Babej, T., Fingerhuth, M., et al. (2018). Coarse-grained lattice protein folding on a quantum annealer. *arXiv preprint arXiv:1811.00713*.

[23] Banchi, L., Fingerhuth, M., Babej, T., Ing, C., and Arrazola, J. M. (2020). Molecular docking with gaussian boson sampling. *Science Advances*, 6(23):eaax1950.

[24] Bank, R. E. and Scott, L. R. (1989). On the conditioning of finite element equations with highly refined meshes. *SIAM Journal on Numerical Analysis*, 26(6):1383–1394.

[25] Bartolucci, S., Birchall, P., Bombin, H., Cable, H., Dawson, C., Gimeno-Segovia, M., Johnston, E., Kieling, K., Nickerson, N. H., Pant, M., et al. (2021). Fusion-based quantum computation. *arXiv preprint arXiv:2101.09310*.

[26] Bateman, H. (1953). *Higher transcendental functions, volume II*, volume 1. McGraw-Hill Book Company.

[27] Becke, A. D. (1988). Density-functional exchange-energy approximation with correct asymptotic behavior. *Physical Review A*, 38(6):3098.

[28] Becke, A. D. (1993a). Density-functional thermochemistry. iii. the role of exact exchange. *The Journal of Chemical Physics*, 98(7):5648–5652.

[29] Becke, A. D. (1993b). A new mixing of hartree–fock and local density-functional theories. *The Journal of Chemical Physics*, 98(2):1372–1377.

[30] Bennett, C. H., Bernstein, E., Brassard, G., and Vazirani, U. (1997). Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523.

[31] Berger, B. and Leighton, T. (1998). Protein folding in the hydrophobic-hydrophilic (hp) is np-complete. In *Proceedings of the second annual International Conference on Computational Molecular Biology*, pages 30–39.

[32] Bergholm, V., Izaac, J., Schuld, M., Gogolin, C., Alam, M. S., Ahmed, S., Arrazola, J. M., Blank, C., Delgado, A., Jahangiri, S., et al. (2018). Pennylane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*.

[33] Berry, D. W., Ahokas, G., Cleve, R., and Sanders, B. C. (2007). Efficient quantum algorithms for simulating sparse hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371.

[34] Berry, D. W., Childs, A. M., Cleve, R., Kothari, R., and Somma, R. D. (2015). Simulating hamiltonian dynamics with a truncated taylor series. *Physical Review Letters*, 114(9):090502.

[35] Berry, D. W., Gidney, C., Motta, M., McClean, J. R., and Babbush, R. (2019). Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization. *Quantum*, 3:208.

[36] Berry, D. W., Kieferová, M., Scherer, A., Sanders, Y. R., Low, G. H., Wiebe, N., Gidney, C., and Babbush, R. (2018). Improved techniques for preparing eigenstates of fermionic hamiltonians. *npj Quantum Information*, 4(1):1–7.

[37] Bespalova, T. A. and Kyriienko, O. (2021). Hamiltonian operator approximation for energy measurement and ground-state preparation. *PRX Quantum*, 2(3):030318.

[38] Beverland, M. E., Kubica, A., and Svore, K. M. (2021). Cost of universality: A comparative study of the overhead of state distillation and code switching with color codes. *PRX Quantum*, 2(2):020341.

[39] Bharti, K., Cervera-Lierta, A., Kyaw, T. H., Haug, T., Alperin-Lea, S., Anand, A., Degroote, M., Heimonen, H., Kottmann, J. S., Menke, T., et al. (2022). Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, 94(1):015004.

[40] Bloch, F. (1929). Über die quantenmechanik der elektronen in kristallgittern. *Zeitschrift für physik*, 52(7):555–600.

[41] Bombín, H. (2013). An introduction to topological quantum codes. *arXiv preprint arXiv:1311.0277*.

[42] Bombín, H. (2015a). Gauge color codes: optimal transversal gates and gauge fixing in topological stabilizer codes. *New Journal of Physics*, 17(8):083002.

[43] Bombín, H. (2015b). Single-shot fault-tolerant quantum error correction. *Physical Review X*, 5(3):031043.

[44] Bombín, H. (2016). Dimensional jump in quantum error correction. *New Journal of Physics*, 18(4):043038.

[45] Bombín, H., Chhajlany, R. W., Horodecki, M., and Martin-Delgado, M.-A. (2013). Self-correcting quantum computers. *New Journal of Physics*, 15(5):055023.

[46] Bombín, H. and Martin-Delgado, M. A. (2006). Topological quantum distillation. *Physical Review Letters*, 97(18):180501.

[47] Bombín, H. and Martin-Delgado, M. A. (2007). Topological computation without braiding. *Physical Review Letters*, 98(16):160502.

[48] Born, M. and Oppenheimer, R. (1927). Zur quantentheorie der molekeln. *Annalen der Physik*, 389(20):457–484.

[49] Bourassa, J. E., Alexander, R. N., Vasmer, M., Patil, A., Tzitrin, I., Matsuura, T., Su, D., Baragiola, B. Q., Guha, S., Dauphinais, G., et al. (2021). Blueprint for a scalable photonic fault-tolerant quantum computer. *Quantum*, 5:392.

[50] Boyer, M., Brassard, G., Høyer, P., and Tapp, A. (1998). Tight bounds on quantum searching. *Fortschritte der Physik: Progress of Physics*, 46(4-5):493–505.

[51] Brassard, G., Hoyer, P., Mosca, M., and Tapp, A. (2002). Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74.

[52] Bravyi, S. and Kitaev, A. Y. (2002). Fermionic quantum computation. *Annals of Physics*, 298(1):210–226.

[53] Bravyi, S. and Kitaev, A. Y. (2005). Universal quantum computation with ideal clifford gates and noisy ancillas. *Physical Review A*, 71(2):022316.

[54] Bravyi, S. and König, R. (2013). Classification of topologically protected gates for local stabilizer codes. *Physical Review Letters*, 110(17):170503.

[55] Brenner, S. C., Scott, L. R., and Scott, L. R. (2008). *The mathematical theory of finite element methods*, volume 3. Springer.

[56] Brown, B. J., Nickerson, N. H., and Browne, D. E. (2016). Fault-tolerant error correction with the gauge color code. *Nature Communications*, 7(1):1–8.

[57] Buhrman, H., Cleve, R., Watrous, J., and De Wolf, R. (2001). Quantum fingerprinting. *Physical Review Letters*, 87(16):167902.

[58] Campbell, E. T. (2019). Random compiler for fast hamiltonian simulation. *Physical Review Letters*, 123(7):070503.

[59] Campbell, E. T. (2021). Early fault-tolerant simulations of the hubbard model. *Quantum Science and Technology*, 7(1):015007.

[60] Campos, R., Casares, P. A. M., and Martin-Delgado, M. A. (2022). Quantum metropolis solver: A quantum walks approach to optimization problems. *arXiv preprint arXiv:2207.06462*.

[61] Cao, Y., Romero, J., Olson, J. P., Degroote, M., Johnson, P. D., Kieferová, M., Kivlichan, I. D., Menke, T., Peropadre, B., Sawaya, N. P., et al. (2019). Quantum chemistry in the age of quantum computing. *Chemical Reviews*, 119(19):10856–10915.

[62] Casares, P. A. M., Campos, R., and Martin-Delgado, M. A. (2022a). Qfold: quantum walks and deep learning to solve protein folding. *Quantum Science and Technology*.

[63] Casares, P. A. M., Campos, R., and Martin-Delgado, M. A. (2022b). TFermion: A non-Clifford gate cost assessment library of quantum phase estimation algorithms for quantum chemistry. *Quantum*, 6:768.

[64] Casares, P. A. M. and Martin-Delgado, M. A. (2020a). A quantum active learning algorithm for sampling against adversarial attacks. *New Journal of Physics*, 22(7):073026.

[65] Casares, P. A. M. and Martin-Delgado, M. A. (2020b). A quantum interior-point predictor–corrector algorithm for linear programming. *Journal of Physics A: Mathematical and Theoretical*, 53(44):445305.

[66] Casares, P. A. M., Sheng Loe, B., Burden, J., hEigeartaigh, S., and Hernández-Orallo, J. (2022c). How general-purpose is a language model? usefulness and safety with human prompters in the wild. *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, 36(5):5295–5303.

[67] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S. C., Endo, S., Fujii, K., McClean, J. R., Mitarai, K., Yuan, X., Cincio, L., et al. (2021). Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644.

[68] Chakraborty, S., Luh, K., and Roland, J. (2020a). Analog quantum algorithms for the mixing of markov chains. *Physical Review A*, 102(2):022423.

[69] Chakraborty, S., Luh, K., and Roland, J. (2020b). How fast do quantum walks mix? *Physical Review Letters*, 124(5):050501.

[70] Chao, R., Ding, D., Gilyen, A., Huang, C., and Szegedy, M. (2020). Finding angles for quantum signal processing with machine precision. *arXiv preprint arXiv:2003.02831*.

[71] Childs, A. M., Kothari, R., and Somma, R. D. (2017). Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950.

[72] Childs, A. M., Ostrander, A., and Su, Y. (2019). Faster quantum simulation by randomization. *Quantum*, 3:182.

[73] Chow, E. (2000). A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM Journal on Scientific Computing*, 21(5):1804–1822.

[74] Clader, B. D., Jacobs, B. C., and Sprouse, C. R. (2013). Preconditioned quantum linear system algorithm. *Physical Review Letters*, 110(25):250504.

[75] Cleve, R., Ekert, A., Macchiavello, C., and Mosca, M. (1998). Quantum algorithms revisited. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 454(1969):339–354.

[76] Clinton, L., Cubitt, T., Flynn, B., Gambetta, F. M., Klassen, J., Montanaro, A., Piddock, S., Santos, R. A., and Sheridan, E. (2022). Towards near-term quantum simulation of materials. *arXiv preprint arXiv:2205.15256*.

[77] Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301.

[78] Coudron, M. and Menda, S. (2020). Computations with greater quantum depth are strictly more powerful (relative to an oracle). In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 889–901.

[79] Dantzig, G. B. (1951). Application of the simplex method to a transportation problem. *Activity Analysis of Production and Allocation*.

[80] Das, R. and Baker, D. (2008). Macromolecular modeling with rosetta. *Annual Review of Biochemistry*, 77:363–382.

[81] Das, R., Qian, B., Raman, S., Vernon, R., Thompson, J., Bradley, P., Khare, S., Tyka, M. D., Bhat, D., Chivian, D., et al. (2007). Structure prediction for casp7 targets using extensive all-atom refinement with rosetta@ home. *Proteins: Structure, Function, and Bioinformatics*, 69(S8):118–128.

[82] Dauphinais, G., Ortiz, L., Varona, S., and Martin-Delgado, M. A. (2019). Quantum error correction with the semion code. *New Journal of Physics*, 21(5):053035.

[83] De Wolf, R. (2019). Quantum computing: Lecture notes. *arXiv preprint arXiv:1907.09415*.

[84] Delgado, A., Casares, P. A. M., dos Reis, R., Zini, M. S., Campos, R., Cruz-Hernández, N., Voigt, A.-C., Lowe, A., Jahangiri, S., Martin-Delgado, M. A., Mueller, J. E., and Arrazola, J. M. (2022). Simulating key properties of lithium-ion batteries with a fault-tolerant quantum computer. *Phys. Rev. A*, 106:032428.

[85] Dennis, E., Kitaev, A. Y., Landahl, A., and Preskill, J. (2002). Topological quantum memory. *Journal of Mathematical Physics*, 43(9):4452–4505.

[86] Derby, C. and Klassen, J. (2021). A compact fermion to qubit mapping part 2: Alternative lattice geometries. *arXiv preprint arXiv:2101.10735*.

[87] Derby, C., Klassen, J., Bausch, J., and Cubitt, T. (2021). Compact fermion to qubit mappings. *Physical Review B*, 104(3):035118.

[88] Dirac, P. A. M. (1930). Note on exchange phenomena in the thomas atom. *Mathematical Proceedings of the Cambridge Philosophical Society*, 26(3):376–385.

[89] Eastin, B. and Knill, E. (2009). Restrictions on transversal encoded quantum gate sets. *Physical Review Letters*, 102(11):110502.

[90] Efron, B. (1992). Bootstrap methods: another look at the jackknife. In *Breakthroughs in Statistics*, pages 569–593. Springer.

[91] Escrig, G., Campos, R., Casares, P. A. M., and Martin-Delgado, M. A. (2022). Parameter estimation of gravitational waves with a quantum metropolis algorithm. *arXiv preprint arXiv:2208.05506*.

[92] Ettinger, M., Høyer, P., and Knill, E. (2004). The quantum query complexity of the hidden subgroup problem is polynomial. *Information Processing Letters*, 91(1):43–48.

[93] Farhi, E., Goldstone, J., and Gutmann, S. (2014). A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*.

[94] Ferris, A. J. (2014). Fourier transform for fermionic systems and the spectral tensor network. *Physical Review Letters*, 113(1):010401.

[95] Fingerhuth, M., Babej, T., et al. (2018). A quantum alternating operator ansatz with hard and soft constraints for lattice protein folding. *arXiv preprint arXiv:1810.13411*.

[96] Fowler, A. G., Mariantoni, M., Martinis, J. M., and Cleland, A. N. (2012). Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86(3):032324.

[97] Ge, Y., Tura, J., and Cirac, J. I. (2019). Faster ground state preparation and high-precision ground energy estimation with fewer qubits. *Journal of Mathematical Physics*, 60(2):022202.

[98] Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741.

[99] Gidney, C. and Fowler, A. G. (2019). Efficient magic state factories with a catalyzed $|CCZ\rangle$ to 2 $|T\rangle$ transformation. *Quantum*, 3:135.

[100] Gidney, C., Newman, M., Fowler, A., and Broughton, M. (2021). A fault-tolerant honeycomb memory. *Quantum*, 5:605.

[101] Gingrich, R. M., Williams, C. P., and Cerf, N. J. (2000). Generalized quantum search with parallelism. *Physical Review A*, 61(5):052313.

[102] Giovannetti, V., Lloyd, S., and Maccone, L. (2008). Quantum random access memory. *Physical Review Letters*, 100(16):160501.

[103] Goings, J. J., White, A., Lee, J., Tautermann, C. S., Degroote, M., Gidney, C., Shiozaki, T., Babbush, R., and Rubin, N. C. (2022). Reliably assessing the electronic structure of cytochrome p450 on today's classical computers and tomorrow's quantum computers. *Proceedings of the National Academy of Sciences*, 119(38):e2203533119.

[104] Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

[105] Gottesman, D. and Chuang, I. L. (1999). Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, 402(6760):390–393.

[106] Gottesman, D., Kitaev, A. Y., and Preskill, J. (2001). Encoding a qubit in an oscillator. *Physical Review A*, 64(1):012310.

[107] Greengard, L. and Rokhlin, V. (1987). A fast algorithm for particle simulations. *Journal of Computational Physics*, 73(2):325–348.

[108] Griffiths, D. J. and Schroeter, D. F. (2018). *Introduction to quantum mechanics*. Cambridge university press.

[109] Grimsley, H. R., Economou, S. E., Barnes, E., and Mayhall, N. J. (2019). An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature Communications*, 10(1):1–9.

[110] Grote, M. J. and Huckle, T. (1997). Parallel preconditioning with sparse approximate inverses. *SIAM Journal on Scientific Computing*, 18(3):838–853.

[111] Grover, L. K. (1998). Quantum computers can search rapidly by using almost any transformation. *Physical Review Letters*, 80(19):4329.

[112] Grover, L. K. (2005). Fixed-point quantum search. *Physical Review Letters*, 95(15):150501.

[113] Grover, L. K., Patel, A., and Tulsi, T. (2006). Quantum algorithms with fixed points: The case of database search. *arXiv preprint quant-ph/0603132*.

[114] Haah, J. (2019). Product decomposition of periodic functions in quantum signal processing. *Quantum*, 3:190.

[115] Hadfield, S., Wang, Z., O'gorman, B., Rieffel, E. G., Venturelli, D., and Biswas, R. (2019). From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34.

[116] Hallgren, S., Russell, A., and Ta-Shma, A. (2003). The hidden subgroup problem and quantum computation using group representations. *SIAM Journal on Computing*, 32(4):916–934.

[117] Harrow, A. W., Hassidim, A., and Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502.

[118] Hart, W. E. and Istrail, S. (1997). Robust proofs of np-hardness for protein folding: general lattices and energy potentials. *Journal of Computational Biology*, 4(1):1–22.

[119] Hastings, M. B., Watson, G. H., and Melko, R. G. (2014). Self-correcting quantum memories beyond the percolation threshold. *Physical Review Letters*, 112(7):070501.

[120] Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1).

[121] Havlíček, V., Troyer, M., and Whitfield, J. D. (2017). Operator locality in the quantum simulation of fermionic models. *Physical Review A*, 95(3):032332.

[122] Hohenberg, P. and Kohn, W. (1964). Inhomogeneous electron gas. *Physical Review*, 136(3B):B864.

[123] Huang, H.-Y., Kueng, R., Torlai, G., Albert, V. V., and Preskill, J. (2021). Provably efficient machine learning for quantum many-body problems. *arXiv preprint arXiv:2106.12627*.

[124] Ivanyos, G., Sanselme, L., and Santha, M. (2008). An efficient quantum algorithm for the hidden subgroup problem in nil-2 groups. In *Latin American Symposium on Theoretical Informatics*, pages 759–771. Springer.

[125] Jensen, F. (2013). Atomic orbital basis sets. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 3(3):273–295.

[126] Jin, J.-M. (2015). *The finite element method in electromagnetics*. John Wiley & Sons.

[127] Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589.

[128] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM Symposium on Theory of Computing*, pages 302–311.

[129] Kassal, I., Jordan, S. P., Love, P. J., Mohseni, M., and Aspuru-Guzik, A. (2008). Polynomial-time quantum algorithm for the simulation of chemical dynamics. *Proceedings of the National Academy of Sciences*, 105(48):18681–18686.

[130] Keen, T., Dumitrescu, E., and Wang, Y. (2021). Quantum algorithms for ground-state preparation and green's function calculation. *arXiv preprint arXiv:2112.05731*.

[131] Kempe, J., Kitaev, A. Y., and Regev, O. (2006). The complexity of the local hamiltonian problem. *SIAM Journal of Computing*, 35(5):1070–1097.

[132] Kerenidis, I. and Prakash, A. (2016). Quantum recommendation systems. *arXiv preprint arXiv:1603.08675*.

[133] Kesselring, M. S., Pastawski, F., Eisert, J., and Brown, B. J. (2018). The boundaries and twist defects of the color code and their applications to topological quantum computation. *Quantum*, 2:101.

[134] Khachiyan, L. G. (1979). A polynomial algorithm in linear programming. In *Doklady Akademii Nauk*, volume 244, pages 1093–1096. Russian Academy of Sciences.

[135] Khoury, M. and Hadfield-Menell, D. (2018). On the geometry of adversarial examples. *arXiv preprint arXiv:1811.00525*.

[136] Kieferová, M., Scherer, A., and Berry, D. W. (2019). Simulating the dynamics of time-dependent hamiltonians with a truncated dyson series. *Physical Review A*, 99(4):042314.

[137] Kim, I. H., Liu, Y.-H., Pallister, S., Pol, W., Roberts, S., and Lee, E. (2022). Fault-tolerant resource estimate for quantum chemical simulations: Case study on li-ion battery electrolyte molecules. *Physical Review Research*, 4(2):023019.

[138] Kirby, W., Fuller, B., Hadfield, C., and Mezzacapo, A. (2022). Second-quantized fermionic operators with polylogarithmic qubit and gate complexity. *PRX Quantum*, 3:020351.

[139] Kirkpatrick, J., McMorrow, B., Turban, D. H., Gaunt, A. L., Spencer, J. S., Matthews, A. G., Obika, A., Thiry, L., Fortunato, M., Pfau, D., et al. (2021). Pushing the frontiers of density functionals by solving the fractional electron problem. *Science*, 374(6573):1385–1389.

[140] Kirkpatrick, S., Gelatt Jr, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

[141] Kitaev, A. Y. (1995). Quantum measurements and the abelian stabilizer problem. *arXiv preprint quant-ph/9511026*.

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

[143] Kivlichan, I. D., Gidney, C., Berry, D. W., Wiebe, N., McClean, J. R., Sun, W., Jiang, Z., Rubin, N., Fowler, A., Aspuru-Guzik, A., et al. (2020). Improved fault-tolerant quantum simulation of condensed-phase correlated electrons via trotterization. *Quantum*, 4:296.

[144] Kivlichan, I. D., McClean, J. R., Wiebe, N., Gidney, C., Aspuru-Guzik, A., Chan, G. K.-L., and Babbush, R. (2018). Quantum simulation of electronic structure with linear depth and connectivity. *Physical Review Letters*, 120(11):110501.

[145] Kohn, W. (1985). Highlights of condensed matter theory. In *International School of Physics" Enrico Fermi*, pages 1–15.

[146] Kohn, W. (1999). Nobel lecture: Electronic structure of matter—wave functions and density functionals. *Reviews of Modern Physics*, 71(5):1253.

[147] Kohn, W. and Sham, L. J. (1965). Self-consistent equations including exchange and correlation effects. *Physical Review*, 140(4A):A1133.

[148] Koralov, L. and Sinai, Y. G. (2007). *Theory of probability and random processes*. Springer Science & Business Media.

[149] Krinner, S., Lacroix, N., Remm, A., Di Paolo, A., Genois, E., Leroux, C., Hellings, C., Lazar, S., Swiadek, F., Herrmann, J., et al. (2022). Realizing repeated quantum error correction in a distance-three surface code. *Nature*, 605(7911):669–674.

[150] Krovi, H., Magniez, F., Ozols, M., and Roland, J. (2016). Quantum walks can find a marked element on any graph. *Algorithmica*, 74(2):851–907.

[151] Kubica, A. and Beverland, M. E. (2015). Universal transversal gates with color codes: A simplified approach. *Physical Review A*, 91(3):032330.

[152] Kubica, A. and Vasmer, M. (2021). Single-shot quantum error correction with the three-dimensional subsystem toric code. *arXiv preprint arXiv:2106.02621*.

[153] Kubica, A. and Yoshida, B. (2018). Ungauging quantum error-correcting codes. *arXiv preprint arXiv:1805.01836*.

[154] Kubica, A., Yoshida, B., and Pastawski, F. (2015). Unfolding the color code. *New Journal of Physics*, 17(8):083026.

[155] Kutzelnigg, W. and Morgan III, J. D. (1992). Rates of convergence of the partial-wave expansions of atomic correlation energies. *The Journal of Chemical Physics*, 96(6):4484–4508.

[156] Kyriienko, O. (2020). Quantum inverse iteration algorithm for programmable quantum simulators. *npj Quantum Information*, 6(1):1–8.

[157] Lee, C., Yang, W., and Parr, R. G. (1988). Development of the colle-salvetti correlation-energy formula into a functional of the electron density. *Physical Review B*, 37(2):785.

[158] Lee, J., Berry, D. W., Gidney, C., Huggins, W. J., McClean, J. R., Wiebe, N., and Babbush, R. (2021). Even more efficient quantum computations of chemistry through tensor hypercontraction. *PRX Quantum*, 2(3):030305.

[159] Lee, S., Lee, J., Zhai, H., Tong, Y., Dalzell, A. M., Kumar, A., Helms, P., Gray, J., Cui, Z.-H., Liu, W., et al. (2022). Is there evidence for exponential quantum advantage in quantum chemistry? *arXiv preprint arXiv:2208.02199*.

[160] Lee, S.-H. and Jeong, H. (2022). Universal hardware-efficient topological measurement-based quantum computation via color-code-based cluster states. *Physical Review Research*, 4(1):013010.

[161] Lemieux, J., Duclos-Cianci, G., Sénéchal, D., and Poulin, D. (2021). Resource estimate for quantum many-body ground-state preparation on a quantum computer. *Physical Review A*, 103(5):052408.

[162] Lemieux, J., Heim, B., Poulin, D., Svore, K., and Troyer, M. (2020). Efficient Quantum Walk Circuits for Metropolis-Hastings Algorithm. *Quantum*, 4:287.

[163] Lin, L. and Tong, Y. (2020). Near-optimal ground state preparation. *Quantum*, 4:372.

[164] Litinski, D. (2019a). A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum*, 3:128.

[165] Litinski, D. (2019b). Magic state distillation: Not as costly as you think. *Quantum*, 3:205.

[166] Litinski, D. and von Oppen, F. (2018). Quantum computing with majorana fermion codes. *Physical Review B*, 97(20):205404.

[167] Loaiza, I., Khah, A. M., Wiebe, N., and Izmaylov, A. F. (2022). Reducing molecular electronic hamiltonian simulation cost for linear combination of unitaries approaches. *arXiv preprint arXiv:2208.08272*.

[168] Low, G. H. and Chuang, I. L. (2017). Optimal hamiltonian simulation by quantum signal processing. *Physical Review Letters*, 118(1):010501.

[169] Low, G. H. and Chuang, I. L. (2019). Hamiltonian simulation by qubitization. *Quantum*, 3:163.

[170] Low, G. H., Kliuchnikov, V., and Schaeffer, L. (2018). Trading t-gates for dirty qubits in state preparation and unitary synthesis. *arXiv preprint arXiv:1812.00954*.

[171] Low, G. H. and Wiebe, N. (2018). Hamiltonian simulation in the interaction picture. *arXiv preprint arXiv:1805.00675*.

[172] Madsen, L. S., Laudenbach, F., Askarani, M. F., Rortais, F., Vincent, T., Bulmer, J. F., Miatto, F. M., Neuhaus, L., Helt, L. G., Collins, M. J., et al. (2022). Quantum computational advantage with a programmable photonic processor. *Nature*, 606(7912):75–81.

[173] Magniez, F., Nayak, A., Roland, J., and Santha, M. (2011). Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164.

[174] Manthiram, A. (2020). A reflection on lithium-ion battery cathode chemistry. *Nature Communications*, 11(1):1–9.

[175] Martin, R. M. (2020). *Electronic structure: basic theory and practical methods*. Cambridge university press.

[176] Martyn, J. M., Rossi, Z. M., Tan, A. K., and Chuang, I. L. (2021). Grand unification of quantum algorithms. *PRX Quantum*, 2(4):040203.

[177] Matousek, J. and Gärtner, B. (2006). *Understanding and using linear programming*. Springer Science & Business Media.

[178] McArdle, S., Campbell, E. T., and Su, Y. (2022). Exploiting fermion number in factorized decompositions of the electronic structure hamiltonian. *Physical Review A*, 105(1):012403.

[179] McClain, J., Sun, Q., Chan, G. K.-L., and Berkelbach, T. C. (2017). Gaussian-based coupled-cluster theory for the ground-state and band structure of solids. *Journal of Chemical Theory and Computation*, 13(3):1209–1218.

[180] McClean, J. R., Boixo, S., Smelyanskiy, V. N., Babbush, R., and Neven, H. (2018). Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1):1–6.

[181] McClean, J. R., Rubin, N. C., Lee, J., Harrigan, M. P., O'Brien, T. E., Babbush, R., Huggins, W. J., and Huang, H.-Y. (2021). What the foundations of quantum computer science teach us about chemistry. *The Journal of Chemical Physics*, 155(15):150901.

[182] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

[183] Mohammadisiahroudi, M., Fakhimi, R., and Terlaky, T. (2022). Efficient use of quantum linear system algorithms in interior point methods for linear optimization. *arXiv preprint arXiv:2205.01220*.

[184] Møller, C. and Plesset, M. S. (1934). Note on an approximation treatment for many-electron systems. *Physical Review*, 46(7):618.

[185] Moore, C., Russell, A., and Schulman, L. J. (2008). The symmetric group defies strong fourier sampling. *SIAM Journal on Computing*, 37(6):1842–1864.

[186] Moore, E. H. (1920). On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26:394–395.

[187] Mosca, M. and Ekert, A. (1998). The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *NASA International Conference on Quantum Computing and Quantum Communications*, pages 174–188. Springer.

[188] Motta, M., Sun, C., Tan, A. T., O'Rourke, M. J., Ye, E., Minnich, A. J., Brandao, F. G., and Chan, G. K. (2020). Determining eigenstates and thermal states on a quantum computer using quantum imaginary time evolution. *Nature Physics*, 16(2):205–210.

[189] Moult, J., Pedersen, J. T., Judson, R., and Fidelis, K. (1995). A large-scale experiment to assess protein structure prediction methods. *Proteins: Structure, Function, and Bioinformatics*, 23(3):ii–iv.

[190] Mulligan, V. K., Melo, H., Merritt, H. I., Slocum, S., Weitzner, B. D., Watkins, A. M., Renfrew, P. D., Pelissier, C., Arora, P. S., and Bonneau, R. (2020). Designing peptides on a quantum computer. *bioRxiv*, page 752485.

[191] Murty, K. G. (1983). *Linear programming*. Springer.

[192] Nagaj, D., Wocjan, P., and Zhang, Y. (2009). Fast amplification of qma. *Quantum Information & Computation*, 9(11):1053–1068.

[193] Nielsen, M. A. and Chuang, I. L. (2002). *Quantum Computation and Quantum Information*. American Association of Physics Teachers.

[194] Noh, K., Chamberland, C., and Brandão, F. G. (2022). Low-overhead fault-tolerant quantum error correction with the surface-gkp code. *PRX Quantum*, 3(1):010315.

[195] Ortiz, G., Gubernatis, J. E., Knill, E., and Laflamme, R. (2001). Quantum algorithms for fermionic simulations. *Physical Review A*, 64(2):022319.

[196] Outeiral, C., Morris, G. M., Shi, J., Strahm, M., Benjamin, S. C., and Deane, C. M. (2021). Investigating the potential for a limited quantum speedup on protein lattice problems. *New Journal of Physics*, 23(10):103030.

[197] Pastawski, F. and Yoshida, B. (2015). Fault-tolerant logical gates in quantum error-correcting codes. *Physical Review A*, 91(1):012305.

[198] Perdomo-Ortiz, A., Dickson, N., Drew-Brook, M., Rose, G., and Aspuru-Guzik, A. (2012). Finding low-energy conformations of lattice protein models by quantum annealing. *Scientific Reports*, 2:571.

[199] Perdomo-Ortiz, A., Truncik, C., Tubert-Brohman, I., Rose, G., and Aspuru-Guzik, A. (2008). Construction of model hamiltonians for adiabatic quantum computation and its application to finding low-energy conformations of lattice protein models. *Physical Review A*, 78(1):012320.

[200] Peruzzo, A., McClean, J. R., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P. J., Aspuru-Guzik, A., and O'brien, J. L. (2014). A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1):1–7.

[201] Portugal, R. (2013). *Quantum walks and search algorithms*, volume 19. Springer.

[202] Postler, L., Heuβen, S., Pogorelov, I., Rispler, M., Feldker, T., Meth, M., Marciniak, C. D., Stricker, R., Ringbauer, M., Blatt, R., et al. (2022). Demonstration of fault-tolerant universal quantum gate operations. *Nature*, 605(7911):675–680.

[203] Potra, F. A. and Wright, S. J. (2000). Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1-2):281–302.

[204] Preskill, J. (2022). Lecture notes for physics 219: Quantum computation. *Caltech Lecture Notes*.

[205] Rebentrost, P., Mohseni, M., and Lloyd, S. (2014). Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503.

[206] Reiher, M., Wiebe, N., Svore, K. M., Wecker, D., and Troyer, M. (2017). Elucidating reaction mechanisms on quantum computers. *Proceedings of the National Academy of Sciences*, 114(29):7555–7560.

[207] Remler, D. K. and Madden, P. A. (1990). Molecular dynamics without effective potentials via the car-parrinello approach. *Molecular Physics*, 70(6):921–966.

[208] Robert, A., Barkoutsos, P. K., Woerner, S., and Tavernelli, I. (2021). Resource-efficient quantum algorithm for protein folding. *npj Quantum Information*, 7(1):1–5.

[209] Romero, J., Babbush, R., McClean, J. R., Hempel, C., Love, P. J., and Aspuru-Guzik, A. (2018). Strategies for quantum computing molecular energies using the unitary coupled cluster ansatz. *Quantum Science and Technology*, 4(1):014008.

[210] Roothaan, C. C. J. (1951). New developments in molecular orbital theory. *Reviews of Modern Physics*, 23(2):69.

[211] Ryabinkin, I. G., Yen, T.-C., Genin, S. N., and Izmaylov, A. F. (2018). Qubit coupled cluster method: a systematic approach to quantum chemistry on a quantum computer. *Journal of Chemical Theory and Computation*, 14(12):6317–6326.

[212] Ryan-Anderson, C., Bohnet, J., Lee, K., Gresh, D., Hankin, A., Gaebler, J., Francois, D., Chernoguzov, A., Lucchetti, D., Brown, N., et al. (2021). Realization of real-time fault-tolerant quantum error correction. *Physical Review X*, 11(4):041058.

[213] Schuch, N. and Verstraete, F. (2009). Computational complexity of interacting electrons and fundamental limitations of density functional theory. *Nature Physics*, 5(10):732–735.

[214] Schuld, M., Bergholm, V., Gogolin, C., Izaac, J., and Killoran, N. (2019). Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99(3):032331.

[215] Seeley, J. T., Richard, M. J., and Love, P. J. (2012). The bravyi-kitaev transformation for quantum computation of electronic structure. *The Journal of Chemical Physics*, 137(22):224109.

[216] Senior, A., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Zidek, A., Nelson, A., Bridgland, A., et al. (2020). Improved protein structure prediction using potentials from deep learning. *Nature*.

[217] Shavitt, I. and Bartlett, R. J. (2009). *Many-body methods in chemistry and physics: MBPT and coupled-cluster theory*. Cambridge university press.

[218] Shepherd, J. J., Grüneis, A., Booth, G. H., Kresse, G., and Alavi, A. (2012). Convergence of many-body wave-function expansions using a plane-wave basis: From homogeneous electron gas to solid state systems. *Physical Review B*, 86(3):035111.

[219] Shewchuk, J. R. et al. (1994). An introduction to the conjugate gradient method without the agonizing pain.

[220] Shor, P. W. (1999). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332.

[221] Simon, D. R. (1997). On the power of quantum computation. *SIAM Journal on Computing*, 26(5):1474–1483.

[222] Somma, R. D., Boixo, S., and Barnum, H. (2007). Quantum simulated annealing. *arXiv preprint arXiv:0712.1008*.

[223] Somma, R. D., Boixo, S., Barnum, H., and Knill, E. (2008). Quantum simulations of classical annealing processes. *Physical Review Letters*, 101(13):130504.

[224] Somma, R. D., Ortiz, G., Gubernatis, J. E., Knill, E., and Laflamme, R. (2002). Simulating physical phenomena by quantum networks. *Physical Review A*, 65(4):042323.

[225] Stair, N. H. and Evangelista, F. A. (2021). Simulating many-body systems with a projective quantum eigensolver. *PRX Quantum*, 2(3):030301.

[226] Stair, N. H., Huang, R., and Evangelista, F. A. (2020). A multireference quantum krylov algorithm for strongly correlated electrons. *Journal of Chemical Theory and Computation*, 16(4):2236–2245.

[227] Stephens, P. J., Devlin, F. J., Chabalowski, C. F., and Frisch, M. J. (1994). Ab initio calculation of vibrational absorption and circular dichroism spectra using density functional force fields. *The Journal of Physical Chemistry*, 98(45):11623–11627.

[228] Steudtner, M. (2019). *Methods to simulate fermions on quantum computers with hardware limitations*. PhD thesis, Leiden.

[229] Steudtner, M. and Wehner, S. (2018). Fermion-to-qubit mappings with varying resource requirements for quantum simulation. *New Journal of Physics*, 20(6):063010.

[230] Student (1908). The probable error of a mean. *Biometrika*, pages 1–25.

[231] Su, Y., Berry, D. W., Wiebe, N., Rubin, N., and Babbush, R. (2021a). Fault-tolerant quantum simulations of chemistry in first quantization. *PRX Quantum*, 2(4):040332.

[232] Su, Y., Huang, H.-Y., and Campbell, E. (2021b). Nearly tight trotterization of interacting electrons. *Quantum*, 5:495.

[233] Suzuki, M. (1991). General theory of fractal path integrals with applications to many-body theories and statistical physics. *Journal of Mathematical Physics*, 32(2):400–407.

[234] Szabo, A. and Ostlund, N. S. (2012). *Modern quantum chemistry: introduction to advanced electronic structure theory*. Courier Corporation.

[235] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

[236] Szegedy, M. (2004). Quantum speed-up of markov chain based algorithms. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 32–41. IEEE.

[237] Tang, E. (2019). A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 217–228.

[238] Tang, E. (2021). Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. *Physical Review Letters*, 127(6):060503.

[239] Thouless, D. J. (1960). Stability conditions and nuclear rotations in the hartree-fock theory. *Nuclear Physics*, 21:225–232.

[240] Tubman, N. M., Mejuto-Zaera, C., Epstein, J. M., Hait, D., Levine, D. S., Huggins, W., Jiang, Z., McClean, J. R., Babbush, R., Head-Gordon, M., et al. (2018). Postponing the orthogonality catastrophe: efficient state preparation for electronic structure simulations on quantum devices. *arXiv preprint arXiv:1809.05523*.

[241] Tzitrin, I., Bourassa, J. E., Menicucci, N. C., and Sabapathy, K. K. (2020). Progress towards practical qubit computation using approximate gottesman-kitaev-preskill codes. *Physical Review A*, 101(3):032315.

[242] University of Washington (2021). Rosetta@home. boinc.bakerlab.org.

[243] Van Laarhoven, P. J. and Aarts, E. H. (1987). Simulated annealing. In *Simulated Annealing: Theory and Applications*, pages 7–15. Springer.

[244] Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., et al. (2022). Alphafold protein structure database: Massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444.

[245] Vasmer, M. and Browne, D. E. (2019). Three-dimensional surface codes: Transversal gates and fault-tolerant architectures. *Physical Review A*, 100(1):012312.

[246] Verstraete, F. and Cirac, J. I. (2005). Mapping local hamiltonians of fermions to local hamiltonians of spins. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09012.

[247] von Burg, V., Low, G. H., Häner, T., Steiger, D. S., Reiher, M., Roetteler, M., and Troyer, M. (2021). Quantum computing enhanced computational catalysis. *Physical Review Research*, 3(3):033055.

[248] Vosko, S. H., Wilk, L., and Nusair, M. (1980). Accurate spin-dependent electron liquid correlation energies for local spin density calculations: a critical analysis. *Canadian Journal of Physics*, 58(8):1200–1211.

[249] Wan, K., Berta, M., and Campbell, E. T. (2022). Randomized quantum algorithm for statistical phase estimation. *Physical Review Letters*, 129(3):030503.

[250] Wannier, G. H. (1937). The structure of electronic excitation levels in insulating crystals. *Physical Review*, 52(3):191.

[251] Watrous, J. (2001). Quantum algorithms for solvable groups. In *Proceedings of the thirty-third annual ACM Symposium on Theory of Computing*, pages 60–67.

[252] Watson, F. H., Campbell, E. T., Anwar, H., and Browne, D. E. (2015). Qudit color codes and gauge color codes in all spatial dimensions. *Physical Review A*, 92(2):022312.

[253] Werner, H.-J., Manby, F. R., and Knowles, P. J. (2003). Fast linear scaling second-order møller-plesset perturbation theory (mp2) using local and density fitting approximations. *The Journal of Chemical Physics*, 118(18):8149–8160.

[254] Whitfield, J. D., Havlíček, V., and Troyer, M. (2016). Local spin operators for fermion simulations. *Physical Review A*, 94(3):030301.

[255] Whitfield, J. D., Yung, M., Tempel, D. G., Boixo, S., and Aspuru-Guzik, A. (2014). Computational complexity of time-dependent density functional theory. *New Journal of Physics*, 16(8):083035.

[256] Wiebe, N. and Granade, C. (2016). Efficient bayesian phase estimation. *Physical Review Letters*, 117(1):010503.

[257] Wierichs, D., Izaac, J., Wang, C., and Lin, C. Y.-Y. (2022). General parameter-shift rules for quantum gradients. *Quantum*, 6:677.

[258] Wigner, E. and Jordan, P. (1928). Über das paulische äquivalenzverbot. *Zeitschrift für Physik*, 47:631.

[259] Wocjan, P. and Abeyesinghe, A. (2008). Speedup via quantum sampling. *Physical Review A*, 78(4):042336.

[260] Wocjan, P. and Zhang, S. (2006). Several natural bqp-complete problems. *arXiv preprint quant-ph/0606179*.

[261] Wong, R. and Chang, W.-L. (2021). Quantum speedup for protein structure prediction. *IEEE Transactions on NanoBioscience*.

[262] Wossnig, L., Zhao, Z., and Prakash, A. (2018). Quantum linear system algorithm for dense matrices. *Physical Review Letters*, 120(5):050502.

[263] Ye, Y., Todd, M. J., and Mizuno, S. (1994). An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm. *Mathematics of Operations Research*, 19(1):53–67.

[264] Yoder, T. J., Low, G. H., and Chuang, I. L. (2014). Fixed-point quantum search with an optimal number of queries. *Physical Review Letters*, 113(21):210501.

[265] Yung, M.-H. and Aspuru-Guzik, A. (2012). A quantum–quantum metropolis algorithm. *Proceedings of the National Academy of Sciences*, 109(3):754–759.

[266] Zhao, Y., Ye, Y., Huang, H.-L., Zhang, Y., Wu, D., Guan, H., Zhu, Q., Wei, Z., He, T., Cao, S., et al. (2022). Realization of an error-correcting surface code with superconducting qubits. *Physical Review Letters*, 129(3):030501.

[267] Zhong, H.-S., Wang, H., Deng, Y.-H., Chen, M.-C., Peng, L.-C., Luo, Y.-H., Qin, J., Wu, D., Ding, X., Hu, Y., et al. (2020). Quantum computational advantage using photons. *Science*, 370(6523):1460–1463.