UNIVERSITAT DE BARCELONA

Algorithmic strategies for seizing quantum computing

Adrián Pérez Salinas

Tesi doctoral

Algorithmic Strategies for seizing Quantum Computing

Autor: Adrián Pérez Salinas

Director: José Ignacio Latorre Sentís
Director: Artur García Sáez

UNIVERSITAT DE BARCELONA

Algorithmic Strategies for seizing Quantum Computing

Programa de doctorat en Física

Autor: Adrián Pérez Salinas

Director: José Ignacio Latorre Sentís
Director: Artur García Sáez

Tutor: Joan Soto Riera

Barcelona, setembre 2021

[illegible]

ADRIÁN PÉREZ SALINAS

LIBERTAS PERFVNDET · OMNIA LVCE

UNIVERSITAT DE BARCELONA

Chapters images, credits to original sources:
Contents:
library_mistress in `Flickr` : *Alicia Martin: Biografias - Cascade of books*
Agradecimientos:
Unknown author : *Ave del Paraíso*
List of publications:
Myself : *My own signature*
Resumen:
Linnaea Mallette: *Porthole, public domain picture*
Introduction:
Sara Díaz : *Photo from Park Güell*
Quantum and Classical ML:
Ramón y Cajal: *Drawing of the cells of the chick cerebellum from "Estructura de los centros nerviosos de las aves"*
Data re-uploading strategy for QML:
Maurits Cornelius Escher : *Ascending and Descending*
Unary strategy:
Wikimedia commons : *Fishing net IMGP8396.jpg*
Conclusions:
NovoaR in `Flickr`: *Ronsel*
Appendices A and B:
Wikipedia - *Trencadís*
Bibliography:
Erik Desmazières - *La Bibliothèque de Babel*

LaTeXtemplate *The Legrand Orange Book*, version `2.1.1 (14/2/16)`, by Mathias Legrand, with modifications

Quantum circuits were designed with *Q-circuit*, source can be found at `CTAN`

First printing, XXXXX 2021

A una estrella despistada

Abstract

Quantum computing is a nascent technology with prospects to have a huge impact in the world. Its current status, however, only counts on small and noisy quantum computers whose performance is limited. In this thesis, two different strategies are explored to take advantage of inherently quantum properties and propose recipes to seize quantum computing since its advent. First, the re-uploading strategy is a variational algorithm related to machine learning. It consists in introducing data several times along a computation accompanied by tunable parameters. This process permits the circuit to learn and mimic any behavior. This capability emerges naturally from the quantum properties of the circuit. Second, the unary strategy aims to reduce the density of information stored in a quantum circuit to increase its resilience against noise. This trade-off between performance and robustness brings an advantage for noisy devices, where small but meaningful quantum speed-ups can be found.

Contents

Agradecimientos

Σὰ βγεῖς στὸν πηγαιμὸ γιὰ τὴν Ἰθάκη,
νὰ εὔχεσαι νἆναι μακρὺς ὁ δρόμος,
Cuando emprendas tu viaje a Ítaca
pide que el camino sea largo
Konstantinos Petrou Kavafis

Shaking dust from my shoes
There's a road ahead
And there's no way back home
Ohh but I have to say
Leaving home ain't easy
Freddie Mercury

A pesar de que esta tesis la firma una persona, lo cierto es que poner el punto y final a este proyecto de libro no hubiera sido posible sin la ayuda de mucha gente, de todos los que pusieron un adoquín más en este camino que llamamos la vida. En mi caso, el camino tiene una parada de avituallamiento en la que me harán entrega de un papelito que me da el derecho a hacerme llamar doctor. Creo que es un papel que no vale nada, nada comparado con las ampollas que me ha levantado el sendero en los pies. Lo importante no es el final, es el viaje, y sobre todo las paradas que se hacen en el camino para compartir el té con alguien que en ese momento deja de ser un extraño.

Las dos personas que más influencia han tenido en esta tesis han sido José Ignacio y Artur. Ellos tenían el mapa del camino que había que seguir mientras yo sólo veía maleza. Gracias a sus consejos y sus indicaciones ha sido posible llegar hasta el final de este proyecto. Durante todo el doctorado han sido ellos los que han peleado por hacer posible que yo continuara en la brecha, a pesar de todos los obstáculos que han ido apareciendo en el camino. Ha sido un viaje lleno de momentos difíciles, la pandemia que nos confinó a todos en casa, los desplazamientos intercontinentales de José Ignacio, las nuevas y exigentes responsabilidades de Artur en el BSC, la financiación perdida y recuperada a falta de año y medio... Pero al final y casi de milagro todo ha ido saliendo bien. Muy atrás quedarán ya las discusiones armados con una tiza, pero estas fueron el sustrato que me permitió crecer después. Por ello les estaré a los dos eternamente agradecido.

Ha habido otras muchas personas que me han ayudado a considerarme investigador. Quiero hacer una mención especial para Stefano Carrazza, con quien entendí que se podían hacer cosas más allá de los muros del grupo, y otra para Alba Cervera, a la que considero la hermana mayor que todo lo sabe y que nunca tuve, y de quien aprendí una barbaridad durante el poco tiempo que compartimos en la facultad. Quiero también acordarme de las muchas horas en común que pasé, sobre todo al principio, con Carlos, con Diego y con Sergi, y de todo lo que conseguimos juntos. Empezamos algunos, Elies, Josep, y con el tiempo conocí a nuevos compañeros, a Axel, a Bruna. De todos he aprendido y me he llevado cosas buenas. Gracias también al grupo experimental, con Pol y David a la cabeza, por conseguir que nuestra imaginación despreocupada tenga un sitio por donde correr. Ir al laboratorio era como volver al pueblo a visitar a los primos lejanos, siempre un motivo de alegría. Thanks also to all collaborators I had during these years Juan, Abdullah, Stavros, Tarun, and everyone with whom I ever talked about quantum computing. Mille grazie a Lorenzo e Nicole per avermi insegnato ad insegnare.

I would also like to thank the people composing my thesis tribunal: Juanjo García-Ripoll, Jordi Tura i Brugués, Leandro Aolita, and also Ramón Muñoz-Tapia, Sofyan Iblisdir and Bruno Julià-Díaz for grainting me the honor to join the scientific guild.

Gracias también a Jorge Cortada y a Jordi Planagumà por el primer impulso que necesitábamos para correr. Después, el suelo se derrumbó bajo nuestros pies, pero la inercia nos permitió encontrar un asidero del que colgarse.

En el mundo de la computación cuántica no todo es investigación y academia, sino también inversiones delicadas y contactos. Por eso quiero agradecer a Parfait Atchadé, y también a los Qapitanes Sergio, Miquel y Guille por enseñarme que hay vida más allá de la ciencia. Gracias a vosotros ahora puedo mirar en otras direcciones.

Esta tesis doctoral culmina una etapa que comenzó hace casi tres años con un salto al vacío, un cambiar una ciudad por otra sin mucho más que la vida empaquetada como buenamente se pudo en una maleta. El agradecimiento más especial de todos es para Sara, para quien se lió la manta a la cabeza y se montó en el tren a mi lado aquella mañana de enero rumbo a una aventura desconocida. Las cosas habrían sido muy diferentes sintigo, tú lo sabes bien. Han sido muchos días juntos en casa (muy juntos, recuerda los meses del COVID), que nos han transformado en personas diferentes una y otra y otra vez más. Pase lo que pase, nunca podré agradecer lo suficiente la generosidad que supuso dejarlo todo atrás para acompañarme. Creo que, después de todo, las cosas han salido razonablemente bien, pero todavía nos queda por vivir.

Aunque se vean sólo las flores, hay que tener presente las ramas que las soportan. Gracias también a mis padres porque aceptaron el viaje con un nudo en la garganta que nunca dejaron de reconocer, y porque me animaron a perseguir aquello que yo deseara. Y todo ello sin tener en cuenta los veintitantos años de cobijo anteriores que hicieron de mí quien hoy soy. Ha sido mucho tiempo de crecer, no juntos pero sí a su sombra, y aunque ahora toca enfrentarse a la intemperie es indescriptible la sensación de tener una casa a la que volver en cualquier momento. Me queda pendiente un pase privado de la defensa de tesis sólo para ellos, para que me pregunten todo lo que quieran durante el tiempo que quieran. Una manera de rememorar otros tiempos de espectáculo. Les debo toda una vida.

Aunque se recojan sólo los frutos, hay que tener presente las raíces que los alimentan. Esta tesis sólo fue posible gracias a cuatro vidas de trabajo, y sólo mucho tiempo después se ha podido apreciar. Sé que dos de mis abuelos harán el enorme esfuerzo de recorrer medio país para presenciar el final de esta etapa, como han hecho tantas veces antes, más cerca de casa, y estarán toda la mañana sentados mirándome, sin entender gran cosa, pero qué más da si está allí, tan guapo, tan seguro de sí mismo, tan haciendo lo que mejor sabe hacer. Uno que me vio comenzar no me verá terminar, o sí, si desde San Isidro los curas le ponen línea directa. Y mi estrellita despistada nunca pudo entender de qué iba todo esto. Por la inmensa desgracia que supone estar pero no estar, esta tesis es sobre todo para ella. Gracias también a la rama del árbol de la tía Nines, para quienes la vida siempre ha ido partido a partido. A Eriste, a Arán y a Leo, me hubiera encantado veros crecer.

No puedo dejar de pensar mientras escribo en que me encantaría que me interrumpiese mi hermano pequeño para proponerme un tentempié, una partida a la consola, o para preguntarme cómo se puede resolver tal integral o tal ejercicio. Creo que nunca he llegado a entender cuánto he llegado a echarlo de menos. La lástima es que se nos haya acabado eso de crecer juntos, y sólo nos queden algunos días sueltos. Alguna vez tenía que llegar al final. Ojalá que nunca se deje ganar y siempre quede una revancha pendiente. Quiero también pedirle perdón por haberlo dejado solo en un momento tan difícil en casa. Sé que no me guarda rencor. Por supuesto, gracias también a María por estos últimos años, y por una portada tan especial para esta tesis.

Gracias también a aquellos con los que siempre he podido contar sin

compartir sangre. A todos y cada uno de los Vragasbundos, a Pedro, a Sergio, a Helen, a Flow, a María, a Luisal, a Rodri, y especialmente a Diego. Pasar estos años sabiendo que no era el único en una situación incierta ha significado mucho para mí. Gracias también a la gente de la universidad, a Marina, Martín y Teresa en Madrid, y un recuerdo especial para Isa; a los que me he ido encontrando por el camino y siguen cerca, a Sofía, a Elena; a los que forman parte de los recuerdos de mi infancia, especialmente a Ana.

Los profesores que me han ayudado a ser como soy también merecen un espacio en estos agradecimientos. Gracias a todos los que he conocido durante muchos años. Gracias a Sergio Montañez (yo también escribo mi tesis a ritmo de rock, yo he elegido a Mark Knopfler) y a Ana Gutiérrez por mostrarme el mundo de la Física. Gracias también a Juanflo, han sido muchísimos años, y a Chemi, por enseñarme música y por hacerme crecer como persona.

Ser foraster mai no és fàcil. Cal aprendre un idioma, cal aprendre els costums de la nova ciutat. Però això sempre es possible si hi ha una família que t'acull. Moltes gràcies a tots els Warriors, a O'Kantz, a Uri, a Laia, a Marc, a Pol, a Raquel, a Max, a Simona i Pere, a Ramón, a Pedret, a Anna i Mar (Warriors honorífiques); i molt especialment a José-Sensei per fer la nostra estada a L'Hospitalet de Llobregat molt més agradable. Moltes gràcies també a la gent del Freiburg, a Valentín, a Sergio, a Raúl i a Guille, per crear un lloc on sentir-se com a casa. Gracias también a Alex, el escurialense por dejarme ser su aprendiz de forastero en Barcelona y en la ciencia.

Por último, este trabajo ha sido de los que pudieron continuar mientras el mundo paraba en mitad de una pandemia mundial provocada por el COVID-19. A todos los sanitarios que se dejaron la piel, y en algunos casos la vida, gracias por cerrar la herida que sufrimos todos.

[illegible]

List of publications

The re-uploading strategy, Chapter 3, of this thesis is based on references [Dut+21; Pér+20a; Pér+21a; Pér+21b]. Corresponding software can be found in references [Pér19; Pér21].

The unary strategy, Chapter 4, of this thesis is based on reference [Ram+21]. Corresponding software can be found in reference [RP20].

A main work during the last two years of this thesis is the development of the software for quantum circuits `Qibo`, from reference [Eft+20a], with corresponding code in [Eft+20b]. `Qibo` is an open-source software to write and execute quantum circuits both on simulation and on actual quantum machines. The software has a variety of useful tools and is constantly growing to provide solutions to new challenges.

Additionally, reference [Pér+20b] was published during the development of this thesis, but it is finally not included.

Articles

[Pér+20a] A. Pérez-Salinas et al. *Data re-uploading for a universal quantum classifier*. Quantum 4 (Feb. 2020). DOI: `10.22331/q-2020-02-06-226`.

[Pér+20b] A. Pérez-Salinas et al. *Measuring the Tangle of Three-Qubit States*. Entropy 22.4 (Apr. 2020). DOI: `10.3390/e22040436`.

[Pér+21a] A. Pérez-Salinas et al. *Determining the proton content with a quantum computer*. Physical Review D 103.3 (Feb. 2021). DOI: `10.1103/physrevd.103.034027`.

[Pér+21b] A. Pérez-Salinas et al. *One qubit as a universal approximant.* Physical Review A 104.1 (July 2021). DOI: 10.1103/physreva.104.012405.

[Ram+21] S. Ramos-Calderer et al. *Quantum unary approach to option pricing.* Physical Review A 103.3 (Mar. 2021). DOI: 10.1103/physreva.103.032414.

Preprints

[Dut+21] T. Dutta et al. *Realization of an ion trap quantum classifier.* (2021). arXiv: 2106.14059 [quant-ph].

[Eft+20a] S. Efthymiou et al. *Qibo: a framework for quantum simulation with hardware acceleration.* (2020). arXiv: 2009.01845 [quant-ph].

Software

[Eft+20b] S. Efthymiou et al. *Quantum-TII/qibo: Qibo 0.1.1.* Version v0.1.1. (Oct. 2020). DOI: 10.5281/zenodo.4071702. URL: http://doi.org/10.5281/zenodo.4071702.

[Pér19] A. Pérez-Salinas. *Quantum classifier with data re-uploading.* (2019). URL: https://github.com/AdrianPerezSalinas/universal_qlassifier.

[Pér21] A. Pérez-Salinas. *Universal-Approximator.* (2021). URL: https://github.com/UB-Quantic/Universal-Approximator.

[RP20] S. Ramos-Calderer and A. Pérez-Salinas. *Quantum Finance.* (2020). URL: https://github.com/UB-Quantic/quantum-unary-option-pricing.

Resumen

Castellano

La computación cuántica es una tecnología emergente con potencial para resolver problemas hoy impracticables. Para ello son necesarios ordenadores capaces de mantener sistemas cuánticos y controlarlos con precisión. Sin embargo, construir estos ordenadores es complejo y a corto plazo sólo habrá ordenadores pequeños afectados por el ruido y sujetos a ruido (NISQ). Para aprovechar los ordenadores NISQ se exploran algoritmos que requieran pocos recursos cuánticos mientras proporcionan soluciones aproximadas a los problemas que enfrentan.

En esta tesis se estudian dos propuestas para algoritmos NISQ: *re-uploading* y *unary*. Cada estrategia busca tomar ventaja de diferentes características de la computación cuántica para superar diferentes obstáculos. Ambas estrategias son generales y aplicables en diversos escenarios.

En primer lugar, *re-uploading* está diseñado como un puente entre la computación cuántica y el aprendizaje automático (Machine Learning). Aunque no es el primer intento de aplicar la cuántica al aprendizaje automático, *re-uploading* tiene ciertas características que lo distinguen de otros métodos. En concreto, *re-uploading* consiste en introducir datos en un algoritmo cuántico en diferentes puntos a lo largo del proceso. Junto a los datos se utilizan también parámetros optimizables clásicamente que permiten al circuito aprender cualquier comportamiento. Los resultados mejoran cuantas más veces se introducen los datos. El *re-uploading* cuenta con teoremas matemáticos que sustentan sus capacidades, y ha sido comprobado con éxito en diferentes situaciones tanto simuladas como experimentales.

La segunda estrategia algorítmica es *unary*. Consiste en describir los problemas utilizando sólo parte del espacio de computación disponible dentro del ordenador. Así, las capacidades computacionales del ordenador no son óptimas, pero a cambio las operaciones necesarias para una cierta tarea se simplifican. Los resultados obtenidos son resistentes al ruido, y mantienen su significado, y se produce una compensación entre eficiencia y resistencia a errores. Los ordenadores NISQ se ven beneficiados de esta situación para problemas pequeños. En esta tesis, *unary* se utiliza para resolver un problema típico de finanzas, incluso obteniendo ventajas cuánticas en un problema aplicable al mundo real.

Con esta tesis se espera contribuir al crecimiento de los algoritmos disponibles para ordenadores cuánticos NISQ y allanar el camino para las tecnologías venideras.

English

Quantum computing is an emergent technology with prospects to solve problems nowadays intractable. For this purpose it is a requirement to build computers capable to store and control quantum systems without losing their quantum properties. However, these computers are hard to achieve, and in the near term there will only be Noisy Intermediate-Scale Quantum (NISQ) computers with limited performance. In order to seize quantum computing during the NISQ era, algorithms with low resource demands and capable to return approximate solutions are explored.

This thesis presents two different algorithmic strategies aiming to contribute to the plethora of algorithms available for NISQ devices, namely *re-uploading* and *strategy*. Each strategy takes advantage of different features of quantum computing, namely the superposition and the density of the Hilbert space in *re-uploading*, and entanglement among different partitions of the system in *unary*, to overcome a variety of obstacles. In both cases, the strategies are general and can be applied in a range of scenarios. Some examples are also provided in this thesis.

First, the *re-uploading* is designed as a meeting point between quantum computing and machine learning. Machine learning is a set of techniques to build computer programs capable to learn how to solve a problem through experience, without being explicitly programmed for it. Even though the *re-uploading* is not the first attempt to join quantum computers and machine learning, this approach has certain properties that make it different from other methods.

In particular, the *re-uploading* approach consists in introducing data into a classical algorithms in different stages along the process. This is a main difference with respect to standard methods, where data is uploaded at the beginning of the procedure. In the *re-uploading*, data is accompanied by tunable classical parameters that are optimized by a classical method according to a cost function defining the problem. The joint action of data and tunable parameters grant the quantum algorithm

a great flexibility to learn a given behavior from sampling target data. The more re-uploadings are used, the better results can be obtained.

In this thesis, *re-uploading* is presented by means of a set of theoretical results supporting its capabilities, and simulations and experiments to benchmark its performance in a variety of problems.

The second algorithmic strategy is *unary*. This strategy describes a problem making use of only a small part of the available computational space. Thus, the computational capabilites of the computer are not optimal. In exchange, the operations required to execute a certain task become simpler. As a consequence, the retrieved results are more resilient to noise and decoherence, and meaningful. Therefore, a trade-off between efficiency and resillience against noise arises. NISQ computers benefit from this circumstance, especially in the case of small problems, where even quantum advantage and advantage over standard algorithms can be achieved.

In this thesis, *unary* is used to solve a typical problem in finance called option pricing, which is of interest for real world applications. Options are contracts to buy the right to buy/sell a given asset at certain time and price. The holder of the option will only exercise this right in case of profit. Option pricing concists in estimating this profit by handling stochastic evolution models.

This thesis aims to contribute to the growing number of algorithms available for NISQ computers and pave the way towards new quantum technologies.

1. Introduction

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 Feynman

The scientific and innovation community is nowadays immersed in the advent of a new technological paradigm that promises to change the world as it is known. The emergence of quantum technologies will likely have a great impact in many different areas. In particular, a great revolution is expected to occur in the field of computing, as firstly proposed by Feynman [Fey82; Pre21]. Quantum computing is the main field of study of the present thesis. However, it is also valuable to highlight other emerging quantum technologies such as communication [GT07], cryptography [Ben+92] or materials [Nat16].

The first inspiration to come to the idea of quantum computers was the hardness to simulate natural phenomena. Classical computers were, and still are, extremely valuable to deal with the description of the world the human beings live in, but they fail at describing the Nature using the laws of quantum mechanics. The exponentially large size of the quantum Hilbert space is in fact responsible for this claim. The ideal solution to this problem was first sighted by Feynman [Fey82], that is, to build a computer following the same rules as Nature to describe its quantum behavior, that is, a quantum computer.

In addition to simulating quantum mechanics, there are other relevant consequences that emerge from this new paradigm of computation. Classical computers are formally defined with the concept of a Turing Machine [Tur38]. Quantum computers cannot be a Turing Machine, but a different kind, commonly known as a Quantum Turing Machine, where all classical components are substituted by their more-sophisticated quantum counterparts. Quantum computing was gradually extended until its range of applicability reached fields outside the pure simulation of Physics, for instance in the seminal works from Refs. [Bra+02; BV97; Deu85; Gro96; Sho97], where some problems are treated whose resolution using quantum devices is more efficient than using classical ones. This case is commonly known as a quantum advantage. Quantum advantages can be exponential, the most desired but rarest kind, for instance in case of Shor's integer-factoring algorithm [Sho97], and polynomial as in Grover's search algorithm [Gro96]. The main difference between both cases steeps in two different complexity classes available in the field of quantum computing [AB09; Vaz02]

In addition to the theoretical advantages of quantum computing when dealing with a variety of problems, classical computers are slowly approaching their physical limits. Moore's law [Moo65] roughly predicts that the components of classical computer will reduce their size and energy requirements at a constant rate. This rate will necessarily come to an end in the fabrication of components as it is nowadays performed when the limit of quantum mechanics is reached. This phenomenon has already started [Ben18].

Quantum computing is different from classical computing from first principles. There exist two main inherently quantum properties that differentiate both paradigms [NC10], namely entanglement and superposition. Both are born from the definition of the quantum bit - qubit - and the conjunction of several qubits.

One qubit is conceived as the quantum counterpart of a classical bit. If bits can take values 0 and 1, then qubits are

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{1.1}$$

with $|\alpha|^2 + |\beta|^2 = 1$. The contemporary coexistence of both states is known as superposition, that is, quantum states are linear combinations of several well-defined states. A qubit only maintains its superposition state if it remains unobserved. At the moment a measurement is performed, the quantum state collapses to one of its well-defined states $|0\rangle\,/\,|1\rangle$ with probability $|\alpha|^2/|\beta|^2$ and the superposition is lost.

If several qubits are set together, it is possible to obtain a quantum state as

$$|\psi\rangle = \sum_{i=0}^{2^n-1} \alpha_i\,|i\rangle\,, \tag{1.2}$$

with $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$ and $|i\rangle$ is the combination of $|0\rangle$ and $|1\rangle$ that corresponds to the binary representation of i. It is clear that there are 2^n available complex coefficients, while a naive product conjunction of single-qubit states would return only $2n$ degrees of freedom. Entanglement is responsible for the emergent properties among different quantum systems to give rise to joint systems much larger than the sum of isolated parties. In particular, the dimensionality of the Hilbert space grows exponentially with the number of qubits. It is also important to note that quantum computing makes use of exponentially high-dimensional spaces, but these spaces are dense as well. In contradistinction, classical computing is built upon discrete possibilities of strings composed of 0's and 1's.

Despite the theoretical advantages presented above and the aforementioned results, there are prominent inconvenients to be overcome before the existence of quantum computers is a reality. Those inconvenients are mainly related to quantum decoherence preventing quantum systems with many entangled particles to show quantum behavior and the inherent difficulty of experimental accurate control over these systems [HR96; Lan95; Unr95]. Even though different proposals already appeared to solve or at least reduce the impact of decoherence and reach a completely operational fault-tolerant quantum computing [AB08; Got97; Sho95; Sho96; Ste96], a physical realisation of such devices is still far from being possible.

The current scenario corresponds to the Noisy Intermediate-Scale Quantum (NISQ) era [Pre18], that is a moment in time when available quantum computers have moderate numbers of qubits, around hundreds of them, and the logical operations available to apply on the qubits are not completely accurate. In addition, quantum states cannot be indefinitely maintained along time and the purely quantum properties are steadily lost during the execution of a quantum algorithm. The two greatest achievements until the present time are a double attainment of the so-called quantum supremacy, that is, using a quantum computer to solve a problem more efficiently and with better performance than any classical computer [Aru+19; Zho+20]. In both cases, the problems solved are picked to favor the quantum implementation, and they are of no particular interest except for the experiment itself. However, the technological improvement needed to actually accomplish this goal must be highlighted. On the other hand, computers in the NISQ era are not expected to change the world by themselves, but rather to be an intermediate step towards a new generation of devices.

One of the most prominent fields of research for NISQ devices are Variational Quantum Algorithms (VQA) [Bha+21; Cer+20]. These family of algorithms are constructed as hybrid models combining quantum and classical resources. The chosen models are usually quantum circuits with fixed architectures but adjustable classical parameters. The circuits are trained using a classical optimizer in such a way that an optimal configuration of parameters suffices to reach approximate solutions to a problem of interest. Variational approaches aim to use classical resources to mitigate possible hardware imperfections and limit the demand of quantum requirements.

Another burning question is how to construct algorithms for quantum computers that are resilient enough to noise as to retrieve meaningful outcomes from the device. An extended, not unique, approach consists in spreading the information contained within a quantum system into a much larger one. The information is now encoded as a global property of the quantum system, much harder to destroy and with a chance to be recovered from a partial decay of the system. Specific implementations differ among different examples [Kit03; Ste96].

In this thesis a work for both a VQA and a deterministic noise-resilient algorithm are presented. In the first case, a VQA framework is developed to present a general strategy capable to address a wide variety of Machine Learning (ML) problems. The strategy is here referred as the *re-uploading strategy*. For the noise-resilient algorithm, an example of a real-world problem related to financial calculations is explored to demonstrate quantum advantage already feasible on current quantum computers. It is here called *unary strategy*.

Re-uploading strategy

The re-uploading strategy described along Ch. 3 is a very general framework to bring the fields of quantum computing and ML together. ML comprisses all algorithms that can learn how to solve particular problems from sampling data, without being explicitly designed for it. Re-uploading is not the first approach to attempt this path, see Ch. 2 for a brief review on different subjects on this topic. However, re-uploading comes up with a novel idea: data serving as input to any ML algorithm is introduced several times into a quantum circuit. The re-uploading strategy belongs to the class of hybrid quantum-classical algorithm.

The idea behind the repeated injection of data into the quantum circuit is to explore all the available Hilbert space and take advantage of it. As mentioned before, the Hilbert space available for performing computations is not only highly dimensional, but also dense, in contradistinction to the classical scheme of computation. To accomplish this goal, the data is interspersed with series of tunable parameters. These parameters, when optimized, drive the behavior of the quantum circuit to approximately what is required to solve a given problem. This behavior is learnt by sampling a training data as in all ML algorithms.

The key ingredient that makes the re-uploading strategy useful for ML is the natural emergence of non-linear properties. Non-linearities are needed in ML to make models capable to approximate arbitrary functions, and then capable to learn and mimic the properties of any data. In classical models for ML, non-linearities are artificially introduced, unlike in the re-uploading strategy here presented, where non-linearities arise naturally from the quantum properties of the circuits.

In this thesis, several aspects of re-uploading are treated. First, theoretical support is given as a justification to use the method with general purposes. Although theoretical support is made explicit for particular cases, general arguments are given to glimpse universality for broader situations. Then, several applications of the re-uploading strategy for different problems are developed with satisfying results to benchmark its performance in different scenarios. The implementations have been implemented on noiseless and noisy simulations of quantum systems, and on actual quantum devices, where the performance degrades in average as the noise increases. The examples of this thesis include regression of test functions, classification of data and extracting physical results from experimental data using the re-uploading strategy.

Unary strategy

The unary strategy aims to explore the possibility to reduce the overall performance of quantum algorithms in exchange to gain robustness against noise. In the present work, this is accomplished by reducing the Hilbert space used along the computation. This way, the information in the quantum state does not spread over all the available space, and thus decoherence and noise do not destroy the quantum state entirely. The unary strategy is used in this work to solve a problem of quantitative finances called option pricing. This is a real-world problem with realistic applications.

The algorithm is built on the unary representation of quantum states. The Hilbert space considered takes into account only those states in the computational basis where there is only one qubit in the $|1\rangle$ state, and all others qubit are $|0\rangle$. On the one hand, this reduces significantly the amount of information that can be stored into the quantum state, from $e^{\mathcal{O}(n)}$ to $\mathcal{O}(n)$. On the other hand, this reduction simplifies the circuit needed to execute a given algorithm, which translates into a mitigation of potential errors. In addition, the unary algorithm always resides in the restricted area of the Hilbert space, and thus any measurement must reflect this fact. This triggers a native post-selection mechanism that permits to mitigate errors.

The financial problem to be solved is known as option pricing. The holder of an option gets the right to buy/sell a given asset at a certain price and date. This right is only exercised if it grants some economical benefit. The problem of option pricing consists in estimating the expected pay-off of this option by running a stochastic model of price evolution.

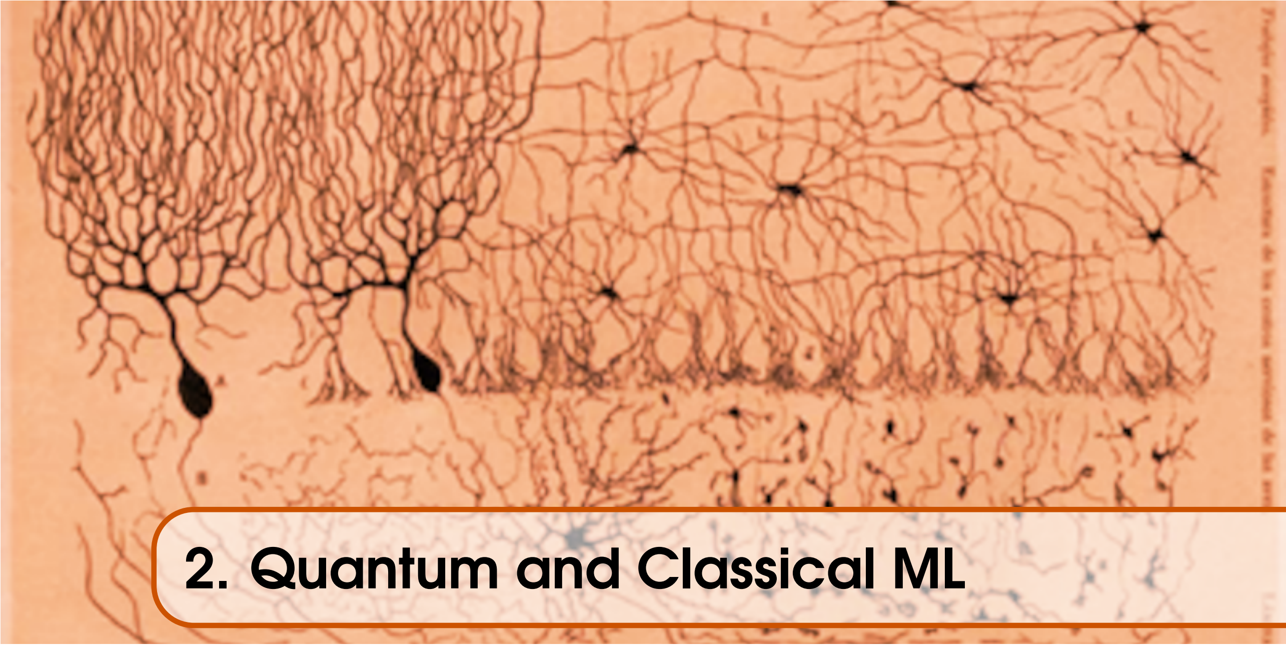
Even though the theoretical and asymptotical performance of the unary algorithm is low with respect to the standard representations. The aim of unary strategy is not to compete against other efficient methods, but rather to show that it is possible to obtain a trade-off between the quantum advantage obtained and the resilience against noise. In the current NISQ era, robustness brings advantage since more theoretically efficient approaches retrieve meaningless final results. In addition, a slight quantum advantage is possible even in the less efficient unary scheme thanks to the Quantum Amplitude Estimation (QAE) recipe. Furthermore, practical applications of the option pricing problem make it useful in the first NISQ era.

Structure of this thesis

Chapter 2 covers a brief overview of the status of both classical and quantum ML. This chapter gives, first, an overall context of the current situation, and second, all background serving as preliminary contents for next chapters.

In Chapter 3, the entire re-uploading strategy is covered as described above. This chapter is composed by different sections treating distinct subjects. Technical appendices to this chapter can be found in App. A. The content of this chapter is based on the works in Refs. [Dut+21; Pér+20a; Pér+21a; Pér+21b; Pér19; Pér21].

In Chapter 4, the unary strategy and its implementation for financial problems is explained. Technical appendices to this chapter can be found in App. B. This chapter is based on the works in Refs. [Ram+21; RP20]



2. Quantum and Classical ML

The world isn't getting any easier. With all these new inventions I believe that people are hurried more and pushed more... The hurried way is not the right way; you need time for everything - time to work, time to play, time to rest.

Hedy Lamarr

Machine Learning (ML) is nowadays one of the most important fields of computation, being ubiquitious both in research and industry. In recent years, it has gained a strong presence mainly due to the improvement of computing techniques and the increase of available data, both aspects supported by the emergent surge of technological capabilities. ML has been used to develop algorithms capable to solve complex tasks in an automatic manner. For instance, a classic problem of ML is to automatically recognize handwritten digits [Den12]. Current capabilities allow to solve much more complex problems, being the most prominent playing chess [CHH02], Go [Sil+16] or solving the protein folding problem [Jum+21]. The scientific research also benefits from the development of ML [Car+19]

ML is a broad field including all different algorithms and techniques with the capability to improve automatically by collecting experience and sampling data. [MCM13; Mit+97; Rus10]. These algorithms can learn in a general sense, and are prepared to carry specific tasks without being specifically programmed with this purpose.

The main three steps needed to carry a given ML algorithm are essentially:

1 **Model design:** the architecture of the ML model is created. These

models usually have some more or less fixed structure with tunable parameters. Those parameters can reach the number of millions in some complex cases. The chosen architecture can be in principle designed from scratch. Nevertheless, there already exists a catalogue of pre-defined models whose competitive performances have been proven under broad conditions. In most cases, the pre-defined architectures can be adjusted to match the needs of the problem to solve.

2 **Training:** once the model is obtained, it is compulsory to tune the parameters in such a way that the algorithm is capable to solve a given task. This is done by learning from some training dataset. The action of the model is to receive some input (x) and return some output $(y) = f(x)$. Ideally, the output y provides the solution to the problem of interest. The training is done by improving the obtained solution for the values of x provided by the dataset. In case the dataset also includes the expected solution, the training is performed by comparing output and solution for the same point and minimizing those differences. Otherwise, some more imaginative methods are needed.

3 **Generalization:** the final aim of any ML technique is to generalize the method, that is, being able to provide good results even for data previously unseen. This step lies at the core of the ML strategies. This step is checked by making the algorithm act on a different dataset known as the test set. The output of this test set should be correct and similar to the one of the training set. Otherwise, the traning must be repeated to ensure generalization. There exist techniques for achieving this goal.

There exist mainly three different approaches to tackle a problem using ML techniques, depending on the features of the datasets to deal with:

- **Supervised learning:** In supervised learning the dataset contains couples of both input and output. The goal of the algorithm is to mimic the general behavior mapping input to output, both for the training and for unseen data. Common applications of this approach are regression and classification [Nie15].
- **Unsupervised learning:** In this case the dataset only has input, and there is no possible reference for the possible output of the algorithm. The training of the algorithm is accomplished by comparing outputs of different inputs and understanding the similarities. A celebrated application of unsupervised learning is clustering [DHS12; Est02].
- **Reinforcement learning:** The main relationship for this problem is the one between an agent (the model) and the environment. The agent must obtain the best possible cumulative reward through a given process by combining strategies of inmediate gains and

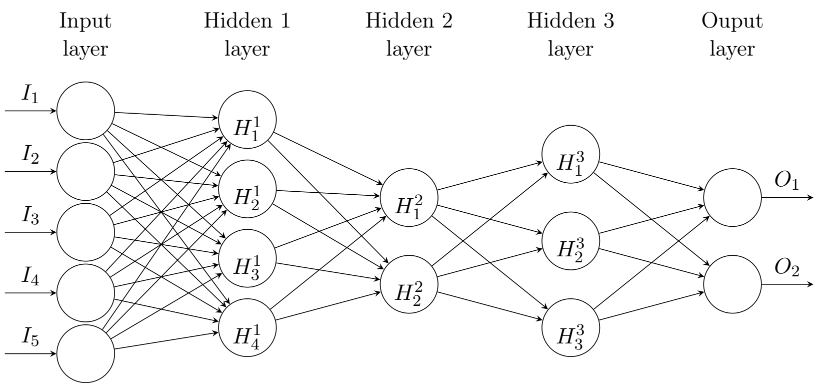


Figure 2.1: *Feedforward Neural Networks (FfNN). Left layer is the input, while right layer is the output. Data is processed from left to right. In the same layer, several processings are taken in parallel, one in each neuron. Different steps are then carried in different layers.*

further exploring [KLM96; Sch98].

The aim of this section is not to provide an exhaustive review of ML techniques. Instead, it is a bibliographical shallow survey of those techniques of both classical and quantum ML related to future content in this thesis, in particular in Ch. 3.

2.1 Classical Machine Learning

In this section general concepts of Classical Machine Learning (CML) are covered as a technical introduction to ML.

2.1.1 Neural Networks

Classical Machine Learning (CML) is mostly dominated by the presence of Neural Networks (NN). They are a successful family of models capable to solve a great variety of tasks with high performance. NNs are inspired in animal brains. All NNs are composed by unit cells, neurons, that process data. Each neuron receives some input and returns some output. Input data is processed in each neuron depending on some fixed behavior, commonly known as the propagation function, and tunable parameters. In a NN, neurons are connected to other neurons following a specific architecture. The connections between neurons can also be tunable.

The main strength of a NN resides on the emergent properties appearing from the correlations between all different neurons. Data is processed in several parallel and / or subsequent steps to give the chance to the NN to disclose the most important joint properties of data. Therefore, a NN is a method capable to find the most interesting pieces of the dataset to solve the problem of interest.

Feedforward Neural Networks (FfNN) are the most general model for NNs and their basic architecture [Hof92; KLM96; Nie15]. This architecture is versatile and provide good performances for a wide variety of problems. Standard FfNNs can be simply defined as a series of layers where neurons are connected between consecutive layers, see Fig. 2.1. Each neuron receives some input $\vec{x}$ coming from the previous layer and produces an output y, which is forwarded to the next layer. The general behaviour of each neuron in the FfNN is

$$y = \sigma\left(\vec{w} \cdot \vec{x} + b\right), \tag{2.1}$$

where $\vec{w} \cdot \vec{x} = \sum_j w_j \cdot x_j$, $\vec{w}$ is the weight vector connecting neurons, b is the bias and $\sigma(\cdot)$ is known as the activation function. This function can be chosen among different options, and it is not required that all neurons have the same one. It is important to mention that this scheme of weights and biases plays a key role to grant the NN enough flexibility as to solve different problems by adjusting its parameters.

The explicit computation performed in every neuron is then as follows. The role of the input layer is to introduce data into the NN, possibly with some pre-processing performed within the neuron. In general, this processing is void and the action of the input layer is just an identity map. However, depending on the problems it can be convenient to apply some refinement of raw data. In the case of the hidden layers, the processing is given by the function

$$y_j^l = \sigma\left(\sum_{k=0}^{n_{l-1}} w_{j,k}^l y_k^{l-1} + b_j^l\right); \tag{2.2}$$

where l is the index of the hidden layer and j, k run over all neurons of the corresponding layer. In terms of indices, the input layer can be labelled with $l = 0$. Notice that this definition is recursive and the activation function of each neuron has some other activation functions, possibly different, together with the corresponding weights and biases. Therefore, each layer adds a new step in the complexity of the final output provided by the NN.

It is worth mentioning that there are theoretical results supporting the general use of the FfNN model. The first result guaranteeing that a FfNN can represent any continuous function, that is most functions of interest, was obtained in Ref. [Cyb89]. In this preliminar but fundamental result, the FfNN is restricted to be a single-hidden-layer NN, and the $\sigma(\cdot)$ function is restricted to be a sigmoid (hence the symbol). A sigmoid function satisfies the property

$$\sigma(x) \rightarrow \begin{cases} 1 & \text{as} \quad t \rightarrow \infty \\ 0 & \text{as} \quad t \rightarrow -\infty \end{cases}, \tag{2.3}$$

and the most celebrated example is $\sigma(x) = (1 + e^{-x})^{-1}$. Further results extended the role of the $\sigma(\cdot)$ function to be any non-constant non-bounded continuous function [Hor91]. The results supporting universality for multi-layer NNs were achieved later [Les+93].

Apart from the basic FfNN model, there exists a great variety of NNs whose peak performances are achieved under different conditions. Recursive NNs apply the same connections recursively over a structured input. They are for instances broadly used for Natural Language Processing [HMS05]. Recurrent NNs connect layers with themselves and are useful for handwriting or speech recognition [Gra+09; LW15]. Autoencoders efficiently encode high-dimensional into small parameter spaces data [Kra91], and they are applied in problems such as face-recognition and face-generation [KW19]. Convolutional NNs apply filters to input data and are commonly used in image processing [Val+20]. Boltzmann Machines (BM) have capabilities to learn probability distributions over sets of inputs [AHS85b; DS19].

Training the NN

The aim of any NN, in particular the FfNN model, is to learn from input data to return input capable to solve a given problem. For the simple assumption of a supervised learning problem, a given sample of outputs is provided by the problem itself, namely $\vec{y}_o$. The action of the NN can be in general described as a function

$$NN(\vec{x}; W, B), \tag{2.4}$$

where $W = \{w^l_{j,k}\}$ is the set of all weights and $B = \{b^l_j\}$ is the set of all biases. It is straightforward to see that the aim is to make $\vec{y} \approx NN$. In order to train the NN it is required to measure and minimize the differences between these two quantities. This is usually accomplished by defining a cost function $\chi^2(W, B)$ such that the approximation is better as $\chi^2(W, B)$ decreases. A common example of this quantity is

$$\chi^2(W, B) = \frac{1}{2} \underset{\{\vec{x}, \vec{y}\}}{\text{Average}} \left(\vec{y} - NN(\vec{x}; W, B)\right)^2, \tag{2.5}$$

although other possibilities can also be considered [Nie15].

The next step consists in finding an optimal set of parameters (W, B) such that

$$(W, B)_{\text{opt}} = \underset{\{W,B\}}{\operatorname{argmin}} \, \chi^2(W, B). \tag{2.6}$$

This operation can be achieved in many different ways. In general, this problem is passed to an optimization program returning an instance of the optimal parameters, the corresponding value of $\chi^2(W, B)$ and other informations depending on the method to optimize. Optimizing a multi-variable function as in this scenario is in general a NP-hard problem [Jai17; Par17], unless the landscape of $\chi^2(W, B)$ is convex, as in this case [Nie15]. There exists a large variety of optimization methods capable to solve different problems, for instance those based on gradients, such as Stochastic Gradient Descent (SGD) [Nie15], quasi-Newton methods [Byr+95; Nas84; NW06], conjugate-direction [Pow64], simplex sampling [NM65] or genetic strategies [Han06]. The performance of each method strongly depends on the characteristics of the function to optimize, and in general it is not possible to know which method is more convenient for a particular problem prior to carrying the optimization.

The family of NNs have found in SGD a method returning high quality solutions for many instances of problems of interest. The gradient-descent piece consists in updating the sets of parameters (W, B) along many iterations by

$$W^l = W^{l-1} - \eta_W \frac{\partial \chi^2(W,B)}{\partial W} \tag{2.7}$$

$$B^l = B^{l-1} - \eta_B \frac{\partial \chi^2(W,B)}{\partial B}, \tag{2.8}$$

where $\eta_{W,B}$ can change along the process following different recipes [KB17; Nes83; Qia99; Sut13; Zei12], and the derivative $\partial\chi^2/\partial\{W, B\}$ can be computed exactly or approximately [Spa05; Spa98]. Thus, any gradient-descent based method looks for a standard steepest descent. NNs can use SGD methods very efficiently thanks to two features, namely batch optimization and backpropagation [Nie15].

Batch optimization consists in estimating the gradient over only a subset of the training set instead of averaging all the possible values of $\{\vec{x}\}$. In every iteration, the choice of the training data subset is different. Batch optimization brings two advantages respect to the standard gradient-descent strategy. First, there is an improvement in the efficiency since the number of function evaluations required per iteration is reduced. After few iterations all the available values of input $\{\vec{x}\}$ have been used and thus participate in the optimization process, and thus the final result is statistically identical to a standard procedure. Second, the alternation between among subsets of $\{x\}$ permits to circumvent local minima. In case a given subset encounters a local minima for a given configuration, this scenario disappears in the next iteration. It is then more likely to reach nearly-optimal minima.

Backpropagation techniques lie at the core of optimization in NNs [Nie15]. This method allows to compute the gradient values with respect to all different weights and biases efficiently. The results from layer $i + i$ are recovered for layer i successively, and the global amount of operations required is largely diminished. The backpropagation algorithm has as main computing rules

$$\frac{\partial \chi^2(W,B)}{\partial b^i_j} = \delta^i_j \tag{2.9}$$

$$\frac{\partial \chi^2(W,B)}{\partial w^i_{j,k}} = y^{i-1}_k \delta^i_j, \tag{2.10}$$

with

$$\delta^L_j = (O_j - y^L_j)\sigma'(z^L_j) \tag{2.11}$$

$$\delta^l_j = \sum_k w^{l+1}_{j,k} \delta^{l+1}_k \sigma'(z^l_j). \tag{2.12}$$

Backpropagation is efficiently implemented because the number of operations needed to obtain an estimate of the derivative is small. In addition, it is entirely based on linear algebra and matrix-vector multiplication. The hardware progress accomplished in the last years focuses on fast and efficient implementation of these operations, in particular through Graphical Processing Units (GPU) [NVF20].

An extremely successful training process for NNs with large number of parameters may lead to an overlearning phenomenon, see Fig. 2.2. Overlearning appears when the training data offers a great complexity. A model with a large enough number of parameters can represent all tiny details present in the training data, when properly trained. However, it is important to have in mind that the scope of ML in general is not to fit a given training data, but to generalize the properties of the training data to provide competitive solutions for unseen datasets. Thus, overfitting must be avoided. There are several techniques with this purpose, like controlling the number of parameters (W, B), adding a normalization term $\lambda(|W|^2 + |B|^2)$ to the cost function, or controlling the performance on the test set to pick the best configuration with respect to this metric, probably not the optimal one according to $\chi^2(W,B)$.

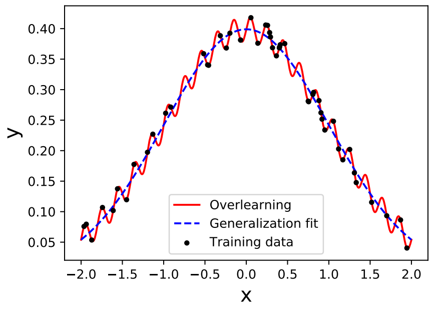


Figure 2.2: *Graphical description of the overlearning phenomenon. The model is capable to fit the training data (black dots) extremely well (solid red line). However, the general behavior of the training data (dashed blue line) is lost in the process and not captured by the fitting model.*

2.1.2 Support Vector Classifier

Support Vector Classifiers (SVC) are an alternative method to NNs for supervised and unsupervised learning [Ben+02; CV95]. In these models, the classification of data is performed by means of a support vector, hence the name, capable to distinguish different classes. For instance, a couple $\{\vec{x}, y\}$, with $y = \pm 1$, is classified by means of the hyperplane

$$\vec{w} \cdot \vec{x} - b = 0, \tag{2.13}$$

where the classes $y = \pm 1$ correspond to both sides of the plane, ideally with some margin if data is linearly separable. In case it is not possible, the optimal boundary can be found by optimizing the cost function

$$\chi^2(\vec{w}, b) = \frac{1}{2} \underset{\{\vec{x},\vec{y}\}}{\text{Average}} \left(\max(0, 1 - y(\vec{w} \cdot \vec{x} - b)\right) + \lambda |\vec{w}|^2, \tag{2.14}$$

where λ determines a trade-off between increasing the margin-size and retaining all samples in the correct side of the space. This parameter is also related to the overlearning processes of Sec. 2.1.1.

There exists an alternative method to linearly separate non-separable data, known as the kernel trick [Pre+86]. In this case, the data is embedded into some non-linear function $\varphi(\vec{x})$ and a kernel function

$$K_{ij} = k(\vec{x}_i, \vec{x}_j) = \varphi(\vec{x}_i) \cdot \varphi(\vec{x}_j). \tag{2.15}$$

Subject to this transformation, weights and bias are transformed accordingly. Tha main purpose of this kernel trick is to find a mapping from input data $\vec{x}$ to other embedded space where classification can be done linearly.

2.2 Quantum Machine Learning

Quantum computers have some properties that make them suitable, in principle, to solve problems related to ML more efficiently or accurately than the standard classical counterparts. In particular, a quantum computer with n qubits can store up to $\mathcal{O}(2^n)$ real numbers in its inner quantum state, while n bits are only capable to store n binary variables. In recent years, a new surge of Quantum Machine Learning (QML) algorithms has emerged dealing with many different problems. Some of these examples are reviewed in this section, see for instance Refs. [Bha+21; Cer+20] for condensed overviews. Complete reviews in QML can be read in Refs. [DB17; Per+18].

This section does not include any quantum-inspired algorithm designed to be executed on a classical computer in spite of their potential utility. Quantum-inspired methods, in particular the celebrated Tensor Networks (TN)s [BB17; Orú14a; Orú14b; Vid04; Vid07; Vid08; VMC08], take advantage of efficient representations of quantum states to approximate them with high accuracies. TNs were originally conceived to store quantum states, but they present high levels of flexibility and capabilities to carry arbitrary data structures. Some methods apply the general TN structure to solve ML problems, both in the field of quantum physics [Tor+20] and general problems [Con+21; CWZ21; Mar+20; RS20; SS17; Wan+20a]. General ML strategies used to solve problems of quantum physics are neither considered in this section [Hua+21b].

QML has not accomplished yet an efficient and scalable manner to introduce arbitrary data into a quantum circuit. This lack of uploading methods constitutes a problem when looking for quantum advantages, specially for exponential speed-ups. The reason is that embedding data in a quantum state exploiting the complete storage capability requires the specification of exponentially many terms. In case this translates into an exponential number of operations, a bottleneck appears preventing any remarkable quantum speed-up. This problem is expected to be overcome by a Quantum Random Access Memory (QRAM) [GLM08], that is, a quantum operation whose action is

$$U(x)\left|0\right\rangle = \left|\psi(x)\right\rangle , \tag{2.16}$$

where x is the input data and $\left|\psi(x)\right\rangle$ is a quantum state where the input data is encoded in some convenient manner. QRAMs aim to condensate the loading of large amounts of data into a small number of quantum operations. Nevertheless, no experimental implementation of a QRAM has been achieved yet.

2.2.1 Supervised learning in QML

Many QML algorithms developed so far tackle the problem of supervised learning, both for classification or regression of data. Strategies can include hybrid quantum-classical schemes for optimization, but most of them follow the same scheme of embedding classical data into the quantum circuits and look for the optimal measurement dividing them, as it was carried by classical SVCs.

To extend classical kernels to the realm of quantum computing it is compulsory to define a quantum operation $V(\vec{x})$ taking as input the data of interest and performing the operation

$$\left|\psi(\vec{x})\right\rangle = V(\vec{x})\left|0\right\rangle , \tag{2.17}$$

where the choice of $\left|0\right\rangle$ as initial state is arbitrary since any other quantum state could by chosen and the corresponding transformation could be absorbed in $V(\vec{x})$. In the case where the data of interest is quantum, there is no special needs to embed it in a quantum circuit.

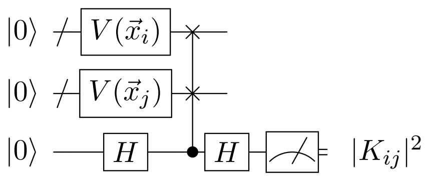


Figure 2.3: *Quantum circuit for computing the kernel of two input data. $V(\vec{x})$ embeds data into the quantum circuit. The ancilla qubit performs a standard swap test to measure the value of the kernel for the $\vec{x}$ instances of interest.*

The kernel function can be directly measured as

$$K_{ij} = \langle \psi(\vec{x}_i) | \psi(\vec{x}_j) \rangle \,, \quad (2.18)$$

where the final K matrix is hermitian with $K_{ii} = 1$. Ideally, $|K_{ij}| \sim 0$ if $\vec{x}_i$ and $\vec{x}_j$ differ between them. Moreover, the properties of this kernel are a cornerstone in the success of any supervised learning task.

The existence of this kernel allows to perform supervised learning both for classification and regression. During the first period of QML, these methods were used to develop standard tools of ML using quantum circuits with no variational parameter, such as a standard SVC [RML14] and Principal Component Analysis (PCA) [LMR14]. Recent works show that quantum kernel methods can only achieve quantum advantage if an appropriate kernel is computed more efficiently using quantum means than classical computation [KBS21]. In fact, a theoretical quantum speed-up on supervised learning has been already accomplished [LAT21]. The main feature of such algorithm is the dataset to be classified, specifically chosen to be mapped to the discrete-logarithm problem. The discrete logarithm belongs to the BQP class and can be solved with a quantum computer exponentially faster than using a classical one [Sho97].

Two problems arise when implementing kernel methods as described above on actual quantum computers. First, finding embeddings $V(\vec{x})$ leading to kernel methods with competitive performances is far from trivial. The Hilbert space reaches 2^n dimensions, where n is the number of qubits. The enormous dimensionality of the Hilbert space opens up the space to embed input data in such a way that different instances lie far from each other in the HIlbert space. To develop the mapping, it is in general a requirement to include deep circuits with many entangling gates and high connectivity between qubits. The literature does not provide embeddings for almost any problem. Second, deep and complex circuits do not perform properly on nowadays quantum computers due to noise and decoherence. Thus, it is not expected that experimental implementations of kernel methods provide competitive results in the near-term. In fact, the discrete logarithm example [LAT21] requires a Quantum Phase Estimation (QPE) step, including a Quantum Fourier Transform (QFT) [NC10], which is out of scope for any state-of-the-art quantum device.

It is also worth it mentioning a general algorithm to be used, among others, in QML. The HHL algorithm [HHL09] is a linear algebra algorithm to invert matrices exponentially more efficient than any classical algorithm under certain conditions. These conditions are a best-case scenario and are usually not completely fulfilled. Inverting matrices has plenty of applications in the field of ML, in particular for SVCs [RML14] or bayesian learning [Zha+19]. However, the hardware demands of this algorithm and derivatives cannot be yet satisfied by current machines.

In order to bring QML closer to the NISQ, several approaches emerged to include VQAs in kernel recipes to exploit the capabilities of these methods. VQAs constitute the largest family of quantum algorithms at the present time. The usefulness of such algorithms is that they are expected to provide approximate solutions to specific problems without a great size or quality of the quantum computers, that is during the NISQ era. VQAs are based on hybrid quantum-classical schemes. The quantum part is composed by a circuit with a fixed structure and gates depending on classical parameters. The exact operation performed by the circuit depends on the set of parameters serving as input. The classical part is a classical optimizer looking for an optimal set of circuit parameters such that a given function of the measurement is minimized. Canonic examples of a VQA are the Variational Quantum Eigensolver (VQE) [Per+14] and Quantum Approximate Optimization Algorithm (QAOA) methods [FGG14]. From a mathematical perspective, a VQA can be seen as a circuit $U(\theta)$ performing the operation

$$|\psi(\theta)\rangle = U(\theta)\,|0\rangle\,. \tag{2.19}$$

This $U(\theta)$ is commonly known as the *Ansatz* of the circuit and comprehends both the architecture and the distribution of classical parameters. Then, an observable M is measured for the output state $|\psi(\theta)\rangle$, and some cost function encoding the problem and depending both on M and θ, $\chi^2_M(\theta)$ drives the search for the optimal configuration of parameters as

$$\theta_{\text{opt}} = \underset{\theta}{\operatorname{argmin}}\,\chi^2_M(\theta). \tag{2.20}$$

This kind of algorithms is only successful if two conditions are matched. First, $U(\theta)$ must be flexible enough as to obtain an accurate approximation to the lowest possible value of $\chi^2_M(\theta)$. Finding an Ansatz with the properties to approximate a given state is no trivial problem [NY21; SJA19]. Nevertheless, the existence Solovay-Kitaev theorem guarantees accurate approximations with a manageable depth of the Ansatz [DN06; NC10]. Second, the classical optimizer must be capable to find a competitive configuration of optimal parameters. Related to this topic, it has been recently demonstrated that training any VQA is a NP problem [BK21], and further difficulties, such as Barren Plateaus (BP) to be commented later, appear [McC+18]. As an attempt to avoid these problems, adiabatic strategies have been added to the existing variational ones [GL18; STC21]

Mixing the framework of VQAs and kernel methods requires to define the global operation

$$|\psi(\vec{x}, \theta)\rangle = U(\theta)V(\vec{x})\,|0\rangle\,, \tag{2.21}$$

and the output of the quantum circuit can be defined as

$$K_{ij} = \langle\psi(\vec{x}_i, \theta)|\, M\, |\psi(\vec{x}_j, \theta)\rangle\,, \tag{2.22}$$

where M is an observable encoding some property of interest. It is possible to compute the kernel by comparing the states corresponding to labels i, j. However, it is also common to retrieve information only of one index, say i, by directly measuring that state. For example, taking $|\psi(\vec{x}_i, \theta)\rangle$ as a single-qubit state it is possible to relate K_{ii} to some function $f(\vec{x}_i) \in [-1, 1]$ by selecting $M = Z$. This approach is usually taken both in classification [Llo+20; Sch+20] and regression [Mit+18].

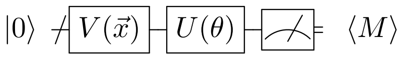


Figure 2.4: *Quantum circuit for measuring the expected value of a given data $\vec{x}$, as in Eq. (2.22), with $\langle M\rangle = \langle\psi(\vec{x}, \theta)|\, M\, |\psi(\vec{x}, \theta)\rangle$. If no comparison between input data $\vec{x}_i, \vec{x}_j$ is desired, no ancilla qubit is required.*

The canonical example of kernel methods both with and without VQA pieces is described in Ref. [Hav+19]. In this work, a simple yet useful embedding map to classify 2-dimensional input data instances into two different classes is implemented on a superconducting device. The classes are defined based on measurement outcomes. The standard kernel method show a clear separation between classes. The classifier using a VQA reaches a success of $\sim 95\%$.

Notice that the operator $U^\dagger(\theta)MU(\theta)$, where θ stands for a set of tunable parameters, is equivalent to finding the optimal measurement to distinguish a certain property [Sch21]. This observation links with the optimal separation of quantum states [Hel76]. Two quantum states are completely distinguishable if they are orthogonal, and the certainty to discriminate them decreases as the relative overlap increases. The computational cost of obtaining the optimal measurement strategy grows exponentially with the number of qubits. Thus, variational methods aim to look for approximately optimal measurements with reduced computational costs.

In order to improve the performance of these variational methods, two main actions can be carried. The first one is to include variational parameters ϕ into the embedding layer, $V(\vec{x}, \phi)$ to gain expressibility in the kernel method. The second approach is to introduce the data $\vec{x}$ redundantly into the circuit. A motivation is to multiply the available information in the circuit while circumventing the no-cloning theorem [WZ82]. This theorem prevents the copy of quantum states. The idea of redundancy was included in theoretical [VT20], numerical [Mit+18] and experimental works [Hav+19].

A conjunction of both redundancy and variational parameters in the embedding step was developed in terms of the *re-uploading* strategy, greatly detailed in Ch. 3 [Pér+20a]. In the re-uploading and similar strategies, the embedding and optimal measurement steps are fused into one combined step. The addition of flexible embedding schemes allow to find a quantum circuit capable to separate data in a nearly optimal manner. In addition, a measurement maximizing distinguishability is also accomplished. This strategy has already shown its universal representation capability [Pér+21b; SSM21].

Training a VQA

Several properties of quantum computers must be considered when QML algorithms rely on findign an optimal configuration of parameters must be found. Many references delegate this search to the classical optimizer without providing any other instruction to help with the procedure. It would be straightforward to think that the process followed by any classical optimizer is equivalent both for quantum and classical computation. It is in fact expected that VQA can outperform classical methods during the NISQ era [HN21] under certain conditions. This is, however, not true in most cases.

The landscape defined by any quantum variational method is affected by sampling uncertainties, what difficults the attainment of accurate measurements. The only possible strategy to retrieve information from a quantum circuit is by measuring observables. A measurement of interest is repeated N to return an estimate of the expected value of such observable as

$$\langle \bar{M} \rangle = \langle \psi | \, M \, | \psi \rangle_N \,, \tag{2.23}$$

where the approximation to the exact expected value $\langle M \rangle$ measured with an infinite number of shots is bounded by

$$|\langle \bar{M} \rangle - \langle M \rangle| \sim \mathcal{O}(N^{-1/2}). \tag{2.24}$$

This phenomenon is inherently quantum and cannot be avoided due to the nature of quantum computers. It can be only mitigated considering more shots or performing techniques such as Quantum Amplitude Estimation (QAE) [Bra+02]. Many modern classical optimization algorithms are designed to seize all the numeric precision computers can provide [nik+20; Vir+20], in particular for gradient-based methods [Byr+95; NC10; NW06; Spa98]. The fine-tuning is required to achieve the best possible final results. The reason for this phenomenon is that many algorithms take advantage of approximation methods whose peak performances are matched at a particular and small computing precision. In case these conditions are not fulfilled, the performance of the recipes degrades severely.

A variety of methods have been developed to deal with the problem of computing gradients in VQAs. First, there exist recipes to analytically evaluate the gradient of a given quantum circuit with respect to some parameter [Mit+18; Sch+19]. The evaluation is exact up to measurement uncertainty. The key observation to design a gradient evaluator for quantum circuits depending on $\vec{\theta}$ is known as the *parameter shift rule*. Gradient components with respect to some parameters are evaluated as a combination of two circuit evaluations as

$$\frac{\partial f}{\partial \theta_i} = r \left(f(\vec{\theta} + s\hat{e}_i) - f(\vec{\theta} - s\hat{e}_i) \right) , \tag{2.25}$$

where $s = \pi/4r$ and r is the absolute value of the eigenvalues of the unitary gate where θ_i comes into, annd $\hat{e}_i$ is the unitary vector in the direction i, that is only the i-th component of $\vec{\theta}$ is modified. Notice that this observation may looks similar to an standard finite-differences method. However, it does not depend on any approximation and is, in fact, exact. This way, only a sampling uncertainty error comes into the calculation. Further works have extended these results to consider also measurement phenomena [Swe+20], and have generalized the parameter shift rule to compute not only first, but higher order derivatives [Hub+21].

Measurement uncertainty is not the only source of inaccuracies appearing when optimizing VQAs. The presence of noise and decoherence also contribute to erratic evaluations of functions on a quantum circuit. There already exist attempts to use the possible noise occurring in the quantum devices to estimate gradients more accurately [MBE21].

Another important inconvenient encountered for quantum circuit optimization is related to the landscape defined by the parameters and the cost function to be minimized. Nowadays it is not clear what is the shape of the parameters landscape, and thus it is difficult to actually design an optimization method suiting these properties.

A first approach to minimize an objective function taking into account the geometry of the landscape is provided by the Natural Gradient (NG) algorithm. This algorithm is inherited from classical computing [Ama98], and consists in evaluating gradients by weighting them with the Fisher information matrix [Fis22; Sav76]. The aim of doing such trick is to gain information about the landscape for optimizing a given function since the Fisher information gives insights on the local geometry structure of the landscape, and it is then more likely to obtain competitive results. Since the first proposal of NG for quantum circuits [Sto+20], several recipes have been published to take profit of this feature [Bec+20; Cer+21a; Gac+21; Mey21]. The main reason preventing a systematic application of NG to the field of VQA is the large amount of measurements needed per iteration.

A prominent problem for all the family of VQAs is known as the BPs. This phenomenon is recognized when the average value of the derivatives of a given circuit vanishes exponentially as the system size increases [McC+18]. The direct consequence is that the landscape becomes essentially flat and it is then extremely difficult to find a parameter configuration with low values of the cost function of interest. The phenomenon of BPs appear in many instances of VQAs [Bra+20a; Bra+20b; HD21a]. The immediate consequence of the existence ofBPs is that an exponentially large number of measurements is needed to resolve an average derivative. BPs compromise the performance of gradient-free optimization methods as well [Arr+20].

Two different sources give rise to BPs. First, the mapping of a set of classical algorithms to an exponentially large Hilbert space induces large separations between close sets, that is, the probability of two objects to be close is exponentially small. The appearance of BPs is closely related with the definition of the cost function of interest [Cer+21b; UB21; UBY20]. In fact, local cost functions reduce the presence of BP. The expressibility of the Ansätze has also a relationship with this source of error [Hol+21; NY21; SJA19]. The landscape of BPs due to cost function can be understood as a large flat space with a narrow deep well in some point. It is extremely difficult to find the optimal point since most of the parameter space does not provide any information about it. The second source of error is the noise of the quantum circuit [Wan+21b]. Random errors make different outputs indistinguishable, and thus the landscape is in average flattened.

BPs are nowadays the main problem to be overcome for achieving quantum advantage in any VQA-dependent procedure. In case the BP issue cannot be solved, it will be exponentially hard to train a quantum circuit. Current methods cannot deal with a problem of such difficulties, and quantum hardware is not competitive enough yet to execute this kind of algorithms accurately. However, it has been shown for small circuits that finding an optimal set of parameters is feasible. The improvement of all different parts of VQAs will translate into the exploitation of NISQ devices.

2.2.2 Other approaches in QML

This chapter is mainly focused on supervised learning due to its connection to the novel work presented in subsequent chapters. Nevertheless, other approaches in QML have also experimented large improvements in recent years. In this section, some major contributions to this field are covered in order to give a broad perspective of the status of the field.

Quantum Boltzmann Machines

BMs are a model for ML coming from its classical approach. Even though the model is well understood from a theoretical and mathematical point of view, it is not broadly used. The main reason is that the training of such model is not efficient in the classical case [AHS85a]. This property may change with the advent of quantum computers since the training methods differ and could take advantage of the special features of BMs.

The idea of BMs come from statistical physics. There exists a system with a total energy, and the probability of finding a particular state of the system when meauring depends on this energy. In a nutshell, a BM is composed by neurons of two kinds: some neurons are visible and some are hidden. The neurons can take continuous or discrete values depending on the exact choice, and these neurons are connected by a two-local hamiltonian

$$H(x; a, b) = \sum_{\{i,j\}} a_{ij} x_i x_j + \sum_i b_i x_i, \tag{2.26}$$

where the weights $a_{i,j}$ and b_i are supposed to control the behavior of the model. In this hamiltonian there is no distinction between visible (v) and hidden (h) neurons. The BM is then sampled to obtain a specific configuration of the visible neurons. The probability of obtaining a given visible result v is

$$P(v) = Z^{-1} \sum_h e^{-H((v,h);a,b)}, \tag{2.27}$$

where Z is the partition function taking all possible outcomes into account for normalization.

The main application of BMs, both in classical and quantum computers, is to learn a probability distribution. The distribution can be taught as a training dataset to mimic, or can be modeled by looking at the relationships between different inputs as in unsupervised or reinforcement learning schemes.

BMs have been recently explored both for circuit and adiabatic quantum computing. In the case of circuits, examples considering VQAs [ZLW21] and time evolution [Shi+20] were developed with appreciable success. In the case of quantum annealing, the computer is used to train the model adiabatically [Dix+21], leading to a much more efficient optimization technique.

Autoencoders

Autoencoders are general recipes to transform raw data into much more compressed versions of it to store and send it more efficiently while minimizing the loss of information. In the most extreme case where the only information of interest is a label, a very large and sophisticated data instance can be compressed to just an integer variable. In any case, autoencoders must carry a compressor and an uncompressor to recover the original data.

Quantum computing has developed several autoencoding algorithms during recent years [ROA17; VPB18; Wan+17]. In addition to theoretical proposals, experimental implementations have been already achieved [PTP19]. It is worth it highlighting the work from Ref. [Bra21], which uses the data re-uploading strategy presented in Ch. 3 to build an autoencoder.

Generative models

The aim of generative models, both for CML and QML is to learn a probability distribution and sample synthetic data from this learning [Du+20; LW18a; VBB19]. An interesting feature of generative models in QML is that models receive as input some white noise that creates similar but different outcomes. Quantum computers are capable to generate purely random states that are beneficial in this subject. In addition, it has been already shown that Born machines are capable to give rise to generative models more efficiently than any classical method [Coy+20].

The model of quantum Generative Adversarial Networks (qGAN) is the generative proposal for quantum computing, in this case using VQAs. The overview of this approach is that two networks, a generator and a discriminator, compite against each other. Only the discriminator must be previously trained. This way, the generator learns to produce data very close to the real one, and the discriminator knows to distinguish accurately [LW18b]. Several examples of this approach have recently emerged [RA19; ZLW19].

Quantum neural networks

In quantum neural networks, every processing unit from the classical models is substituted by a qubit, and the connections among neurons are translated into quantum gates. For instance, standard FfNN-like quantum circuits [Alt01; FN18; TG19] and other models with different purposes and architectures [CCL19; FG20] and convenient trainability properties [Pes+20; Zha+20].

Reinforcement learning

In reinforcement learning for QML, the main expectation is that quantum phenomena, namely superposition and entanglement, can provide quantum speed-up with respect to classical methods [Don+08; DTB16; DTB17]. Preliminary reinforcement learning algorithms have been already tested [Jer+21].

In the quantum approach for reinforcement learning, agent and environment have access to different Hilbert spaces and interchange information via unitary maps. The quantum mapping opens up a larger range of possibilities as compared to the classical counterpart since there is much more freedom in the former.

Reinforcement learning has been used both as an application of VQAs and quantum annealing. In the case of variational circuits, subsequent improvements have been accomplished increasing the efficiency and hardness of the problem to solve [Che+20; LS20; LS21]. For the case of annealing, the main quantum advantage can be achieved by optimization of complex systems, in particular for BM [Cra+19]. Experimental implementations have also appeared recently [Cár+18; Lam17; Yu+19].

3. Data re-uploading strategy for QML

Las cosas que parecen duras tienen una elasticidad...
...una elasticidad retardada.

Julio Cortázar

One of the main features of quantum computing relies on the high density of the computational space. Classical computing is built upon discrete frameworks of strings composed of 0's and 1's, in contradistinction to quantum computing where all superpositions of a number of states are, in principle, available. Thus, every quantum computational system conforms a dense space where an infinite number of states can be described. The main focus now are the capabilities of the smallest possible quantum system. A single qubit is defined by the computational eigenstates $|0\rangle$ and $|1\rangle$. As mentioned, this qubit can in principle be any state of an infinite plethora of superpositions of both eigenstates. This observation permits proposing algorithms where the focus is not on the different states carried by a quantum computation, but rather on the coefficients that define a given quantum state. In fact, a qubit can only store one bit of information (0, 1, alternatively) in their states of the computational basis, but there is room for two natural numbers (or one complex) in its internal degrees of freedom. An example of a state fulfilling this condition would be

$$|\psi\rangle = \sqrt{1-f^2}\,|0\rangle + f e^{i\phi}\,|1\rangle\,, \tag{3.1}$$

with $f \in [0,1]$ and $\phi \in [0,2\pi)$. In this example, the encoded complex number is $f e^{i\phi}$

It is straightforward and efficient to drive one qubit to the state $|\psi\rangle$ by means of some single-qubit rotation. For example, starting from the standard state $|0\rangle$, $|\psi\rangle = R_z(\phi) R_y(2 \arcsin(f)) |0\rangle$. It is also direct to extend this recipe to store functions $f(x), \phi(x)$ instead of fixed values in a quantum circuit, only if the functional forms are known. Therefore, there is room for any complex function $z(x) \in \mathbb{C}; |z(x)| \leq 1$ in a single-qubit circuit. Thus, storing mathematical functions can be achieved by passing the dependency on x to the parameters defining the operations in the circuit.

In case the quantum system to manipulate is larger, the complexity of the algorithm to tune all the coefficients of such state grows exponentially with the number of qubits. In addition to naive approaches, some other sophisticated techniques exist to deal with similar problems, like signal processing or qubitization [Hua+21a; LC17; LC19].

The situation becomes much more complicated when the function to store in the quantum system is known only through samples, that is, its functional form remains veiled. In this case, it is simply not possible to load well-defined functions to the circuit. This scenario becomes of interest in the field of ML. The re-uploading strategy here presented is a ML framework for learning functions from samples, that is solving a problem generalizing data, using a quantum computer assisted by a classical optimizer.

Re-uploading makes use of several gates depending on independent variables or data x and some tunable parameters. The gates are standard single-qubit rotations on the Bloch sphere. These gates are applied sequentially, that is, the independent variable is *re-uploaded* throughout the circuit. The more re-uploadings of x come into the circuit, the more flexibility the circuit has to produce a determined output state. The dependency in x within the gates is linear and kept simple, so that the scheme can adapt to any kind of output state. It also avoids the emergence of biases since no feature of the desired function is introduced in the gates. The tunable parameters present in the different gates have the power to arbitarily shape the final output state of a quantum circuit given a sufficient number of re-uploadings. To find the optimal parameters, classical optimization methods are utilized to force the output state to match some conditions encoded within the definition of a loss function. With these ingredients it is possible to claim that single-qubit systems are capable to store in its degrees of freedom functions only learnt from samples.

The re-uploading strategy permits to circumvent a fundamental limitation existing in quantum computing, namely the no-cloning theorem [WZ82]. Quantum no-cloning theorem prevents a quantum state to be copied into some other quantum register. This limits the available processing of quantum data to either modify it only once or to have several copies of such data to process it in different steps. In contradistinction, classical devices can copy data. For instance, a NN takes the same input as many times as desired when processing it. The implicit solution considered by the re-uploading strategy consists therefore in using a classical device to copy data and introduce it repeatedly in the quantum circuit. In fact, using this line of thought it is possible to find an equivalence between NNs and this re-uploading scheme, see Sec. 3.1.2.

This re-uploading technique introduces changes in the overall structure of Quantum Machine Learning (QML) algorithms as well. As seen in Ch. 2, many of these algorithms are performed in three steps, namely uploading data, processing data and measurement. Re-uploading combines the upload and processing steps into one stage. This avoids several difficulties concerning these issues. Re-uploading tecnhniques do not make use of sophisticated embedding schemes nor large quantum systems to transfer classical data into the quantum ciruits. The linear encoding used in this scheme is capable to capture correlations among different features of the sampling data. Due to the classical optimization in the recipe, this step is accomplished with no prior knowledge on the dataset the quantum algorithm deals with.

The linear encoding gives also rise to highly non-trivial functions naturally due to the quantum nature of the algorithm. One single-qubit operation is capable to introduce a rotation in the Bloch sphere, where the only complexity in the operation depends on the parameter describing such rotation, in this case a linear dependency. This does not supply enough computational power to deal with non-trivial problems. However, a consecutive application of non-commuting single-qubit operations triggers the appearence of non-linear terms. These terms are the reason why the re-uploading strategy is a flexible and general model to address QML problems.

The re-uploading technique emphasizes the capabilities of quantum circuits that make use of a small number of quantum resources. They are of most importance to the field of quantum computing, in particular for the first phase of quantum computing or NISQ era [Pre18]. In fact, algorithms that need few qubits may be proven relevant even though they do not attempt any quantum improvement, since they may be useful pieces of larger and more advantageous circuits. Considering circuits with few quantum resources as the building blocks for larger circuits avoids a drawback to arise as well. Even though it is possible to store a complex function in a single-qubit circuit, retrieving that information from the quantum state is costly and requires a large amount of measurements, for instance performing full tomography methods [DPS03]. Indeed, the Holevo bound [Hol73] limits the accesible information when a quantum system of n qubits is measured to only n classical bits at a time.

The linear encoding used in the re-uploading scheme is another advantage when using this strategy as a piece of larger circuits. Due to this encoding it is possible to include in the upload-processing step data stored in some quantum register. This is feasible using controlled gates whose control qubits are the data register, and the target qubits are the processing ones. To do so, a map between the original scheme and the controlled rotations must be included into the quantum circuit.

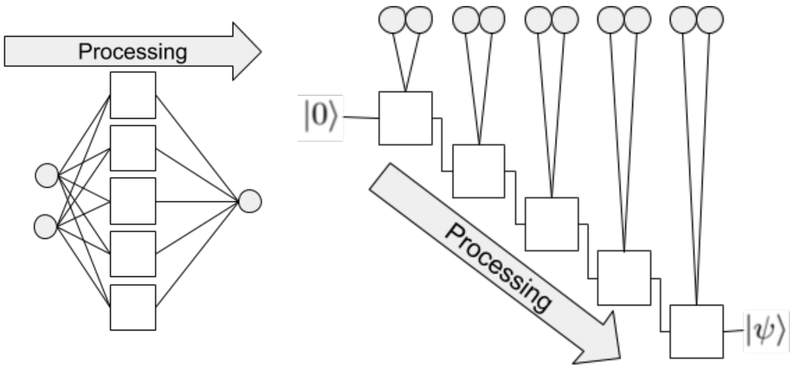
This chapter is based on the articles from Refs. [Dut+21; Pér+20a; Pér+21a; Pér+21b]. It is structured as follows. First, the theoretical aspects and comparison of re-uploading with CML approaches are detailed in Sec. 3.1. Then, numerical benchmarks for some test functions both in classical simulators and experimental devices can be found in Sec. 3.2. Two applications based on this scheme with superficial modifications are then explained in the following sections. Sec. 3.3 contains a quantum classifier accomplished in classical simulations, and Sec. 3.4 describes the experimental implementation of this classifier on an ion-trapped quantum device. Sec. 3.5 is devoted to a machine learning approach to High Energy Physics (HEP) application to determine the content of protons from experimental data obtained at Large Hadron Collider (LHC).

3.1 Theoretical support

There exists a fundamental question in the field of ML of whether a given model can represent any function. In this case, the model is a quantum circuit following the re-uploading strategy, and any possible functionality must be encoded within the degrees of freedom of the output state. If that is the case, it is important to find the sequence of gates required to accomplish this goal. This section is devoted to answer these questions.

In classical cases, this problem was solved by a series of theorems establishing that a given function can be re-expressed as a linear combinations of other specific functions. The most fundamental family of theoretical results in this field is the harmonic anaylisis and Fourier series [Dir29; Rie67]. Those results demonstrate that a great range of functions can be re-expressed as sums of trigonometric functions with fixed frequencies. In classical machine learning, the Universal Approximation Theorem (UAT) proves that a NN with a unique intermediate hidden layer can converge to approximate any continuous function [Cyb89; Hor91]. In both approaches, it is important to notice that each step of the process, namely neurons for UAT or terms for Fourier series, is fed with the original data of the problem. The query complexity of the process increases linearly with the number of steps, namely the degree of approximation.

Data re-uploading strategies are conceived as a quantum analogous of the well-known classical model of NNs. In the case of feed-forwarding NNs, data is entered in the network in such a way that it is processed by



(a) Classical NN (b) Re-uploading Quantum Circuit

Figure 3.1: *Simplified working schemes of a classical NN and a single-qubit re-uploading scheme. In the NN, the processing layer receives information from every neuron in the input layer, and the processing is done in parallel. In contradistinction, the single-qubit circuit receives input from the previous processing unit and the data (introduced classically). It processes everything sequentially. The output of both models are flexible enough to accomodate many constraints.*

subsequent layers of neurons. The key observation is that the original data is introduced as many times as neurons in the first hidden layer, and then the result of one layer is introduced several times in the next one until the output layer is reached. Strictly speaking, data is re-uploaded onto the NN. In case NN were affected by some sort of no-cloning theorem, they could not work as they do. Thus, a circuit is designed such that its architecture allows data to be introduced several times. This observation is critical to develop theoretical support for the expressibility of quantum circuits in order to support progress in QML [BGL20; GTN21; LAT21; Llo+20; LMR13; Mit+18; NY21; Pér+20a; Pér+21a; RML14; SJA19; SSM21; Zhu+19].

An architecture where data can be re-uploaded and processed along the computation is designed. Fig. 3.1 shows a comparison between a single-layer classical NN model and the quantum circuit here proposed. In the case of NN, data is re-uploaded many times in one step, once per neuron, and processed in parallel. Then, all partial processings are collected into a final output neuron. This model can approximate any continuous function in the output neuron, as stated in Th. 3.1.3, [Cyb89; Hor91]. In the case of the quantum circuit, data points are introduced in each processing unit, in this case a single-qubit operation. As it is a single-qubit system, there is no room for parallel processing, and then the calculation is done sequentially, since every gate receives as input the result from the last one.

In this chapter, it is shown that quantum and classical models here discussed are formally equivalent for the same number of processing units. Two independent proofs that any bounded complex function can be approximated in a convergent way by a single-qubit quantum circuit are presented. Thus, it constitutes a **single-qubit approximant**. This demonstrates the precise representation power of a single-qubit circuit, which increases as more layers are added. As for classical models, query complexity, achieved through re-uploading of data, is attached to accuracy. The first proof makes contact with harmonic analysis. This is a natural step as single-qubit gates are expandable in Fourier series that can be arranged to fit existing theorems. The second method is analogous to the UAT using a translation into quantum circuits. In both cases, the quantum theorems inherit the applicability and characteristics of their classical counterparts.

In this section, the problem is first defined in Sec. 3.1.1. The two theorems constituting the core of the work are presented in Sec. 3.1.2, and demonstrated in Sec. 3.1.3. Conclusions are commented in Sec. 3.1.4.

3.1.1 Set-up of the problem

The most general representation of a single-qubit quantum state stores a single complex number, as stated in Eq. (3.1), explicitly with f, ϕ real numbers and $f \in [0,1]$, $\phi \in [0, 2\pi)$. The aim is to encode a complex function within the values (f, ϕ) by defining them as $f : \mathbb{R}^m \to [0,1]$ and $\phi : \mathbb{R}^m \to [0, 2\pi)$. The functional forms of $f(x)$ and $\phi(x)$ are unknown, otherwise solving this problem is trivial. To do so, the circuit $\mathcal{U}^{(k)}_{f,\phi}(x)$ is designed in such a way that its output state approximates the desired complex function as

$$\langle 1 | \mathcal{U}^{(k)}_{f,\phi} | 0 \rangle \sim f(x) e^{i\phi(x)}, \tag{3.2}$$

where k stands for the number of uploadings of data, also referred to as query complexity. Note that building an approximation to a bounded complex function suffices to address any bounded complex function by shifting and re-scaling the target function to another one that fits in the model. In addition, approximating a complex function includes the capability of fitting real-valued functions by either setting $\phi(x) = 0$ or relating the real-valued function to the modulus of other complex functions. The latter approach is lesser demanding since a degree of freedom is set free. Initial $|0\rangle$ and comparing $\langle 1|$ can be chosen arbitrarily without loss of generality, since any state can be transformed into any other by slightly modifying the $\mathcal{U}^{(k)}_{f,\phi}$ operation. The general operation is described in a specific way.

Definition 3.1.1 The k-th approximating circuit is defined as

$$\mathcal{U}^{(k)}_{f,\phi}(x,\Theta) = \prod_{i=1}^{k} U(x, \vec{\theta}_i), \tag{3.3}$$

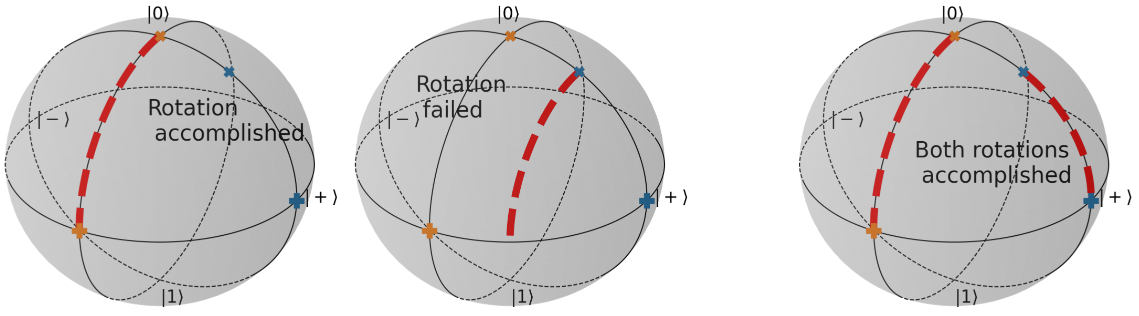
where $U(x, \vec{\theta})$ is a fundamental gate depending on x and a set of parameters $\vec{\theta}$, with $\Theta = \{\vec{\theta}_1, \ldots, \vec{\theta}_k\}$.

This general construction allows to obtain great performances in QML problems for a wide variety of fundamental gates $U(x, \vec{\theta})$, as it is seen in later examples. However, there are two particular choices, to be defined later, that provide mathematically provable universality. The expected behavior of this operation $\mathcal{U}^{(k)}_{f,\phi}$ is that the approximation from Eq. (3.2) will improve as the number k increases, that is, as the independent variable is re-uploaded multiple times and the query complexity increases. The appropriate choice of the parameters Θ enables a systematic approximation of any functionality. The optimal configuration will depend on $f(x)$ and $\phi(x)$.

It is possible to interpret the operation for the re-uploading strategy in Def. 3.1.1 for a single-qubit system by making use of the geometrical arguments picturing the Bloch sphere. Two initial states on the Bloch sphere are considered. Those states must be rotated by the same operation $U(x, \vec{\theta})$ to another arbitrary pair of states. In the case where rotations do not depend on x is first considered, many different operations can be applied, but all of them can be fused into an overall one. The optimal rotation for one point does not fit another one, and thus accomodating several data is not possible, see Fig. 3.2(a) for an illustration. On the other hand, making the operations x-dependent provides the possibility to transport different points along different paths, see Fig. 3.2(b). As more layers are applied, more independent rotations are included into the circuit, and the conjunction of many simple operations permits to obtain systems of increasing flexibility.

It is also worth to mention that the re-uploading strategy allows to increase the distinguishability between two data points. General strategies of QML look for circuits that provides optimal measurement schemes for some given datasets, see Sec. 2.2.1. In that case the performance of the classifier is upper-bounded by the distinguishability obtained by the embedding scheme when uploading data into the quantum circuit. Re-uploading circumvents this limitation. This becomes particularly useful when two close points x_1, x_2 have very different properties, for example in the vicinity of the border between two classes in a supervised learning problem.

In general, the set of parameters for a given gate $\vec{\theta}_i$ is composed of a set of angles. The quest for the optimal set of parameters Θ is driven



(a) x-independent rotations (b) x-dependent rotations

Figure 3.2: *Geometrical interpretation of standard single-qubit operations (a) and the re-uploading strategy (b). The initial states, X crosses around the $|0\rangle$ state, must move towards two different target states, $+$ crosses in the equatorial plane $\langle Z\rangle = 0$. Standard operations are combination of rotations, what leads to another rotation. Thus, it is not possible to achieve a tailored rotation for both initial states. On the contrary, many x-dependent rotations allows to obtain overall flexible rotations.*

by optimizing a particular loss function $\mathcal{L}(\Theta; f, \phi, x)$. This loss function must be designed in such a way that Eq. (3.2) becomes an equality as $\mathcal{L} \rightarrow 0$. The optimal parameters are then

$$\Theta_{\text{opt}} = \text{argmin}_{\Theta}\mathcal{L}(\Theta; f, \phi, x). \tag{3.4}$$

The presence of classical optimization methods makes this scheme belong to the family of variational algorithms.

Optimization techniques

In this chapter, optimizers are taken as black boxes. Finding the optimal configuration corresponding to a minimum (or equivalently a maximum) of a given cost function is a extremely complicated, in fact a NP, problem. As the number of parameters increases, the search space grows exponentially, and so does the difficulty of the problem. Of course, this also depends on the particular loss function that is to be solved. Purely convex functions are usually easier to optimize since all movements lead to the global minimum. Take for instance the function $f(x) = |x|^2$. In this case, the obvious solution is $x = 0$. However, this scenario is certainly not common in general optimization problems.

One of the hardest features of the optimization procedures is that there is no possible way, apart from exhaustive search, to ensure that a given configuration Θ_{opt} provided by an optimizer is the best possible one, that is the global minimum. One can know if this is a local minimum, that is whether there are or not better points in the vicinity of that solution, but all remaining space is unexplored. This becomes particularly difficult when the loss function is full of local minima, for example if $f(x)$ is some combination of trigonometric functions. In this case it is very likely to get trapped in a local minimum, while this local minimum will not probably be the global one.

NNs are commonly trained using the Stochastic Gradient Descent (SGD) [Nie15] with standardized acceptable results, see Sec. 2.1.1. This is a consequence of a previous knowledge on the landscape of the loss functions. Unlike NNs, quantum algorithms do not have any particular optimization method with high performance in wide varieties of problems. In the case of the QML problems presented along this chapter, even the landscape is unknown. Therefore, the optimization strategy to follow to obtain the best possible results is far from being trivial.

A first attempt on developing direct SGD was intended following known recipes for estimating gradients on quantum circuits, see Sec. 2.2.1, [Hub+21; Sch+19; Swe+20]. This was tried in the seminal quantum classifier paper [Pér+20a]. However, the results obtained in this scenario did not rise to the challenge when compared against other methods. In summary, final results did get stuck in local minima far away from results obtained with other methods.

There are two optimization methods that have worked along the examples presented in this chapter, namely the quasi-Newton `L-BFGS(-B)` method [Byr+95], and the genetic `CMA` algorithm [Han06].

The `L-BFGS(-B)` method constructs in every iteration an approximation of the Hessian matrix taking as input information the function evaluation in the current iteration and previous ones, up to an adjustable limit. On the one hand, this method can explore the vicinity of a given point by computing first and second order derivatives. On the other hand, the algorithm saves memories of the previous steps, and then small increases of the landscapes can be overcome using informetion collected before. This idea follows from the inertial thought implemented in other optimizers [KB17; Nie15]. Thus, the search for optimal points is a downhill descent in average. These properties make the algorithm resilient to get stuck in shallow local minima. In this work, the method is implemented as given by `scipy` [Vir+20].

The `L-BFGS(-B)` method is sometimes used in NNs when the training dataset is small. This is understood in terms of local minima. Landscapes of NNs, as in the quantum models observed in this section, are full of local minima. SGD overcomes them by using different batches of training data and descending in different directions so that the final average is a good result. If there is no data available, some other method less sensitive to local minima is needed.

The second method commonly used through this chapter is the genetic algorithm `CMA`. Genetic algorithms are inspired by biological evolution and natural selection. An initial population is generated, and from those individuals only the most adapted ones are chosen to propagate their information to the next generation, which is created by adding changes to their parents. This way, the landscape is explored and only the individuals with smaller values of the cost function survive, contributing to the overall improvement of the population. This algorithm, which is not gradient-based, permits exploring vast landscapes where the derivative information is missing or useless. In exchange, large numbers of function evaluations are usually required to go depp into the landscape and find actual global minima, specially due to the BP phenomenon, see Sec. 2.2.1 [McC+18]. Nevertheless, genetic algorithms can extract the interesting combination of parameters very efficiently. The particular `CMA` strategy chosen stands for the specific manner to create new generations. It generates new individuals as gaussian perturbations of the parents [Han06]. In this chapter, the algorithm is implemented as provided by Ref. [nik+20].

It is worth mentioning that optimization is carried mostly on quantum simulators in all examples here presented, unless stated otherwise. This eliminates a further difficulty that optimization of quantum circuits have. In the future, specific optimization methods for quantum computing will be needed to address these problems efficiently. The reason is that quantum computng has two main sources of errors that degrade the retrievement of values for loss functions: noise and decoherence from the circuit, and sampling uncertainty from the measurements. In principle, optimization techniques can help to mitigate systematic errors, but statistical ones pose a challenge on the attainment of acceptable results.

In terms of noise and decoherence, the functions evaluations are distorted when measured, and thus it is not possible to accurately determine the function values. Thus, the optimizer receives corrupted information and is is simply not possible to drive the search for optimal parameters efficiently. As the quality of quantum computers increases towards fault tolerant computing, these errors will be slowly corrected and mitigated.

Sampling uncertainty adds further unaccuracies to the measured values obtained. This becomes important when decoherence is neglictible since it becomes the main source of errors. The error can be only reduced by increasing the number of measurements, but it is not possible to overcome it. The effect of sampling uncertainties attacks the core of many optimization methods that rely on high precision computing. This dependency must be circumvented to progress towards efficient optimizers, see Sec. 2.2.1 [Swe+20].

3.1.2 Two theorems on universality

The structure of the algorithm previously presented is completed with the design of the single-qubit gates U aforementioned in Definition 3.1.1.

In the following, two sets of single-qubit gates are used to construct quantum circuits that represent arbitrary complex functions. Each set is based on known results from the theory of function approximations, namely Fourier series [Dir29; Rie67] and UAT [Cyb89; Hor91], respectively. The range of applicability of these theorems for quantum circuits and the conditions for universality are thus inherited from their classical counterparts.

Non-linearities

Before coming into technical details for demonstrating universality, it is important to highlight a requirement of $\mathcal{U}_{f,\phi}^{(k)}$, that is the emergence of non-linearities. Non-linearities are needed in all methods looking for universal representability. They appear as an essential ingredient in all classical theorems. For instance, the Fourier series [Dir29; Rie67] are built upon trigonometric functions, and UAT [Cyb89; Hor91] explicitly require non-linear functional forms. While in the classical case non-linearities are introduced artificially, the quantum case makes them appear taking advantage of inherent properties of quantum operations. For this purpose, the fundamental gates $U(x, \vec{\theta})$ cannot commute between them for any combination of (x, θ), except for the trivial case where two gates are equal. Due to the mathematical structure of the $SU(2)$ group containing all possible single-qubit operators, non-linear terms emerge naturally. This can be observed via the Baker-Campbell-Haussdorf (BCH) formula [Eic68; Hal15; VV18].

Definition 3.1.2 BCH formula
Let $e^M = e^Y e^Z$ be, for M, Y, Z matrices and possibly $[Y, Z] \neq 0$. The BCH formula gives the solution to the matrix M in terms of a formal series not necessarily convergent whose first terms are

$$M = Y + Z + \frac{1}{2}[Y, Z] + \frac{1}{12}\left([Y, [Y, Z]] + [Z, [Z, Y]]\right) + \dots \quad (3.5)$$

The quantum operations $R_z(\alpha), R_y(\varphi)$, corresponding to exponentiation for $= -i\alpha/2\sigma_z, Y = -i\varphi/2\sigma_y$, are considerd. The overall operation of interest is $U = e^{-i\varphi/2\sigma_y} e^{-i\alpha/2\sigma_z}$. Following the BCH formula, this final operation can be written as

$$U = e^{-iM}; \qquad M = \frac{\varphi}{2}\sigma_Y + \frac{\alpha}{2}\sigma_z + \frac{\alpha\varphi}{4}\sigma_X - \frac{\alpha\varphi}{24}\left(\varphi\sigma_Y + \alpha\sigma_Y\right). \quad (3.6)$$

In this particular case, the series comes to an end because of the commutation relationships between the Pauli matrices σ_Y, σ_Z. However, even in this simple case, it is possible to detect polynomial terms up to degree 3 $(\alpha^2\varphi, \alpha\varphi^2)$, while the original operations depend only on variables with polynomial degree 1.

This formulation gives some insight on how to design global universal operations [Pér+20a]. Rotations around at least two axis are required to make non-linear quantities appear. However, it is also remarkable that a rotation around a third axis contributes with more parameters, but it is not strictly needed since this third axis emerges naturally with the first non-commuting term.

The single-qubit classifier is constructed as a series of gates which are in general $SU(2)$ matrices. There exist many possible decompositions of an $SU(2)$ rotational matrix. In particular,

$$U(\beta, \varphi, \alpha) = e^{\frac{-i\beta}{2}\sigma_z} e^{\frac{-i\varphi}{2}\sigma_y} e^{\frac{-i\alpha}{2}\sigma_z}. \tag{3.7}$$

According to Def. 3.1.1, the overall operation is generated as a series of simple rotations. Thus, with no loss of generality, $\beta = 0$, since this parameter can always be absorbed by the α parameter of the following layer. Using the $SU(2)$ decomposition law, the above parametrization can be written in a single exponential

$$U(\beta = 0, \varphi, \alpha) = e^{-i\vec{\omega}(\varphi,\alpha)\cdot\vec{\sigma}}, \tag{3.8}$$

with $\vec{\omega}(\varphi, \alpha) = (\omega_1(\varphi, \alpha), \omega_2(\varphi, \alpha), \omega_3(\varphi, \alpha))$ and

$$\omega_1(\varphi, \alpha) = d\, \mathcal{N} \sin(-\alpha/2) \sin(\varphi/2), \tag{3.9}$$
$$\omega_2(\varphi, \alpha) = d\, \mathcal{N} \cos(\alpha/2) \sin(\varphi/2), \tag{3.10}$$
$$\omega_3(\varphi, \alpha) = d\, \mathcal{N} \sin(\alpha/2) \cos(\varphi/2), \tag{3.11}$$

where $\mathcal{N} = \left(\sqrt{1 - \cos^2 d}\right)^{-1}$ and $\cos d = \cos(\alpha/2)\cos(\varphi/2)$.

The re-uploading scheme codifies an independent variable x into the parameters of the U gate as $\alpha(x), \varphi(x)$, where the exact dependency is to be defined yet. Thus

$$\mathcal{U}^{(k)}_{f,\phi}(x, \Theta) = \prod_{i=1}^{k} e^{-i\vec{\omega}(\varphi(x),\alpha(x))\cdot\vec{\sigma}}. \tag{3.12}$$

Applying the BCH formula to the above equation, the overall expression of the global operation is

$$\mathcal{U}^{(k)}_{f,\phi}(x, \Theta) = \exp\left(-i \sum_{i=1}^{k} \vec{\omega}(\varphi_i(x), \alpha_i(x)) + \mathcal{O}_{\text{corr}}\right), \tag{3.13}$$

where $\mathcal{O}_{\text{corr}}$ involves higher order terms of Pauli matrices due to the property $[\sigma_i, \sigma_j] = 2i\varepsilon_{ijk}\sigma_k$, where ε_{ijk} is the Levi-Civitta symbol.

All $\vec{\omega}$ terms are trigonometric function, and thus unconstant, bounded and continuous, as required by UAT [Hor91]. Hence, the sum of all of them must fulfill the same properties.

$$\sum_{i=1}^{k} \vec{\omega}(\varphi_i(x), \alpha_i(x)) = \vec{\eta}(x). \tag{3.14}$$

There are still remaining terms $\mathcal{O}_{\text{corr}}$ of the BCH expansion. Instead of applying such expansion, it is possible to use again the SU(2) group composition law to obtain the analytical formula of $\mathcal{U}^{(k)}_{f,\phi}(x, \Theta) = e^{i\vec{\xi}(x)\cdot\vec{\sigma}}$, where $\vec{\xi}(x)$ will be an inextricably trigonometric function of x. The $\mathcal{O}_{corr}$ terms are proportional to $\vec{\sigma}$ matrices, so $\mathcal{O}_{corr} = \vec{\varrho}(x)\cdot\vec{\sigma}$ for some function $\vec{\varrho}(x)$. Then,

$$\mathcal{U}^{(k)}_{f,\phi}(x, \Theta) = e^{i\vec{\xi}(x)\cdot\vec{\sigma}} = e^{i\vec{\eta}(x)\cdot\vec{\sigma}+i\vec{\varrho}(x)\cdot\vec{\sigma}}. \tag{3.15}$$

Thus, $\mathcal{O}_{corr}$ terms can be absorbed in $\vec{\eta}(\vec{x})$. As the resulting function is a combination of trigonometric function, which satisfies the constraints for Th. 3.1.3, it is expected to have enough flexibility as to represent a huge variety of functions.

Notice that the arguments on the appearances of higher-order terms may provide insights and lines of thought towards the attainment of universality, but they do not suffice to demonstrate it. Up to this point there is no further knowledge of the inner structure of $\vec{\xi}(x)$. The many combined trigonometric functions can contribute to cancel terms with each other so that the final result does not provide universality, even though it has all the required ingredients. Therefore, further development is needed to prove universality.

Theorems

Two different methods are explored to link the universality of the quantum circuit here presented with standard mathematical representability theorems, namely Fourier series and UAT. In both cases, first the mathematical theorems are stated. Then, a quantum gate is defined such that the repeated application of this gate allows to extend those mathematical results to a quantum circuit.

Fourier series as a constructive method permits expressing a great range of target functions defined within an interval as a sum of a set of known functions, see Th. 3.1.1. If a circuit as explicited in Def. 3.1.1 is created by repeating the gate $U^{\mathcal{F}}$from Def. 3.1.3, the output state is compatible with a Fourier series, see Th. 3.1.2. Intuitively, $\alpha, \beta, \varphi, \lambda$ are related to the coefficients of a single Fourier step, while ω may be identified as the corresponding frequency. The relationship between these parameters and the original Fourier coefficients is explicitly shown in Sec. 3.1.3.

FOURIER SERIES

Theorem 3.1.1 Fourier series
[Car66; Dir29; Rie67]
Let z be any function $z : \mathbb{R} \to \mathbb{C}$ with a finite number of finite discontinuities integrable within an interval $[a, b] \in \mathbb{R}$ of length P. Then

$$z_N(x) = \sum_{n=-N}^{N} c_n e^{i\frac{2\pi n x}{P}}, \tag{3.16}$$

where

$$c_n = \frac{1}{P} \int_P z(x) e^{-i\frac{2\pi n x}{P}} dx, \tag{3.17}$$

approximates $z(x)$ as

$$\lim_{N\to\infty} z_N(x) = z(x). \tag{3.18}$$

Definition 3.1.3 Let the fundamental Fourier gate $U^{\mathcal{F}}$ be

$$U^{\mathcal{F}}(x; \underbrace{\omega, \alpha, \beta, \varphi, \lambda}_{\vec{\theta}}) = R_z(\alpha + \beta) R_y(2\lambda) R_z(\alpha - \beta) R_z(2\omega x) R_y(2\varphi), \tag{3.19}$$

with $\alpha, \beta, \varphi, \lambda, \omega \in \mathbb{R}$.

Theorem 3.1.2 Quantum Fourier series
Let f, ϕ be any pair of functions $f : \mathbb{R} \to [0, 1]$ and $\phi : \mathbb{R} \to [0, 2\pi)$, such that $z(x) = f(x)e^{i\phi(x)}$ is a complex function with a finite number of finite discontinuities integrable within an interval $[a, b] \in \mathbb{R}$ of length P. Then, there exists a set of parameters $\{\vec{\theta}_1, \vec{\theta}_2, \ldots, \vec{\theta}_N\}$ such that

$$\langle 1| \prod_{i=1}^{N} U^{\mathcal{F}}(x, \vec{\theta}_i) |0\rangle = z_N(x), \tag{3.20}$$

with $z_N(x)$ the N-terms Fourier series.
Proof in Sec. 3.1.3.

UAT

Theorem 3.1.3 Universal Approximation Theorem
[Cyb89; Hor91; Les+93]
Let I_m denote the m-dimensional cube $[0,1]^m$. The space of continuous functions on I_m is denoted by $C(I_m)$, and $|\cdot|$ denotes the uniform norm of any function in $C(I_m)$. Let $\sigma : \mathbb{R} \to \mathbb{R}$ be any non-constant bounded continuous function. Given a function $f \in C(I_m)$ there exists an integer N and a function

$$G(\vec{x}) = \sum_{n=1}^{N} \alpha_n \sigma(\vec{w}_n \cdot \vec{x} + b_n), \tag{3.21}$$

such that

$$|G(\vec{x}) - f(\vec{x})| < \varepsilon, \qquad \forall \vec{x} \in I_m, \tag{3.22}$$

for $\vec{w}_n \in \mathbb{R}^m$ and $b_n, \alpha_n \in \mathbb{R}$ for any $\varepsilon > 0$.

Definition 3.1.4 Let the fundamental UAT gate U^{UAT} be

$$U^{\mathrm{UAT}}(\vec{x}; \underbrace{\vec{\omega}, \alpha, \varphi}_{\vec{\theta}}) = R_y(2\varphi) R_z(2\vec{\omega} \cdot \vec{x} + 2\alpha), \tag{3.23}$$

with $\{\vec{\omega}, \alpha, \varphi\} \in \{\mathbb{R}^m, \mathbb{R}, \mathbb{R}\}$.

Theorem 3.1.4 Quantum UAT

Let f, ϕ be any pair of functions $f : I_m \to [0,1]$ and $\phi : I_m \to [0, 2\pi)$, such that $z(\vec{x}) = f(\vec{x}) e^{i\phi(\vec{x})}$ is a complex continuous function on I_m, with $I_m = [0,1]^m$. Then there is an integer N and a set of parameters $\{\vec{\theta}_1, \vec{\theta}_2, \ldots, \vec{\theta}_N\}$ such that

$$\left| f(\vec{x}) e^{i\phi(\vec{x})} - \langle 1| \prod_{i=1}^{N} U^{\mathrm{UAT}}(\vec{x}, \vec{\theta}_i) |0\rangle \right| < \epsilon, \tag{3.24}$$

for any $\epsilon > 0$.
Proof in Sec. 3.1.3.

When the building blocks are the $U^{\mathcal{F}}(x, \vec{\theta}_i)$ defined in Eq. (3.19), the unitary operation as defined in Eq.(3.3) generates a total unitary gate that outputs a N-term Fourier series when applied to an initial state $|0\rangle$. Taking $|0\rangle$ as the initial state implies no loss of generality, since $|0\rangle$ can be transformed into any other initial state by adjusting the first $U^{\mathcal{F}}$. The Fourier series behavior is only achieved if all $\{\vec{\theta}_i\}$ take specific values leading to a final result that exactly matches the Fourier coefficients. However, since this procedure relies on quantum-classical variational methods, optimal parameters are searched by means of a classical optimizer. This freedom gives room to configurations surpassing the performance of the standard Fourier series, especially for shallow circuits. However, the recipe to construct the Fourier series by performing well-defined calculations is instead lost. In addition, the Fourier series is obtained only in the last step. Intermediate steps have the functional forms of a Fourier series, but not its values. For details on the proof of this theorem the reader is referred to Sec 3.1.3.

The UAT, Th. 3.1.3 demonstrates that any continuous function of a m-dimensional variable can be uniformly approximated as a sum of a specific set of functions with adjustable parameters. The first formulation restricted the functions to be sigmoidal functions [Cyb89]. Later works extended the result to any non-constant bounded continuous function [Hor91]. This theorem is directly applied to NNs containing one hidden layer.

This theorem is an existence theorem, and thus it does not specify how many terms from Eq. (3.21) are needed to achieve an accuracy ε nor what values the parameters must take. Note that although the UAT supports approximations for real functions, it can be immediately applied to complex functions by substituting the real-valued function $\sigma(\cdot)$ with some complex-valued function. In particular, it works if $\sigma(\cdot) \to e^{i(\cdot)}$. A proof is shown in Appendix A.1.

In the case of quantum circuits as designed in Def. 3.1.1, a quantum analogous of the UAT is available if the repeated gates is the one from Def. 3.1.4. Intuitively, $\vec{\omega}$ and α are equivalent to the weights and bias in a NN, while φ plays the role of the coefficient. The output state, although different to the standard UAT, fulfills the same requirements to ensure universality, see Th. 3.1.4. Both classical and quantum theorems are analogous, and one can arrive at its proof by following the steps developed in Ref. [Cyb89]. All theorems supporting the original formulation of the UAT also hold for the quantum version. For more details on the demonstration of the quantum UAT, the reader is referred to Sec. 3.1.3.

The quantum universality theorems here proposed inherit the range of applicability, advantages and limitations of their classical counterparts. The Fourier approach is guaranteed to work for all integrable functions with a finite number of finite discontinuities. This range of functions includes –but is not limited to– continuous functions. The UAT only gives support to continuous functions, which is useful from a practical perspective, but less robust than the Fourier series.

The Fourier theorem holds for functions depending on a single variable. However, the extension to multi-dimensional spaces is complicated and requires a space of parameters whose size increases exponentially with the number of dimensions [Rie67]. However, in the UAT case the use of multi-variable $\vec{x}$ arises naturally by adjusting the dimension of the weights.

3.1.3 Proofs of universality theorems

This section is devoted to the proofs of Theorems 3.1.2 and 3.1.4 supporting universality for quantum circuits. The reader interested in final results may skip this subsection without any regret.

Demonstration for the quantum Fourier series

The quantum circuit proposed in Theorem 3.1.2 fulfills the requirement that every new gate plays the role of a new step in the original Fourier series. The proof is based on an inductive procedure and can be then decomposed in two steps. First, it is shown that the first gate of the circuit is equivalent to the 0-th constant Fourier term. Then, if there are N gates in a row forming a N-term Fourier series, adding a new gate provides a $(N+1)$-terms Fourier series if previous values are modified.

Let the fundamental gate $U^{\mathcal{F}}(x, \vec{\theta})$ defined in Eq. (3.19) be

$$U^{\mathcal{F}}(x;\vec{\theta}) = U^{\mathcal{F}}(x;\omega,\alpha,\beta,\varphi,\lambda) = R_z\left(\alpha+\beta\right)R_y(2\lambda)R_z\left(\alpha-\beta\right)R_z(2\omega x)R_y(2\varphi) =$$
$$= \begin{pmatrix} \cos\lambda\cos\varphi e^{i\alpha}e^{i\omega x} - \sin\lambda\sin\varphi e^{i\beta}e^{-i\omega x} & -\cos\lambda\sin\varphi e^{i\alpha}e^{i\omega x} - \sin\lambda\cos\varphi e^{i\beta}e^{-i\omega x} \\ \sin\lambda\cos\varphi e^{-i\beta}e^{i\omega x} + \cos\lambda\sin\varphi e^{-i\alpha}e^{-i\omega x} & -\sin\lambda\sin\varphi e^{-i\beta}e^{i\omega x} + \cos\lambda\cos\varphi e^{-i\alpha}e^{-i\omega x} \end{pmatrix}, \tag{3.25}$$

It is possible to recast the above choice of fundamental gate using the following redefinition of parameters,

$$a_+ = \cos\lambda\cos\varphi e^{i\alpha}, \tag{3.26}$$
$$a_- = -\sin\lambda\sin\varphi e^{i\beta}, \tag{3.27}$$
$$b_+ = -\cos\lambda\sin\varphi e^{i\alpha}, \tag{3.28}$$
$$b_- = -\sin\lambda\cos\varphi e^{i\beta}. \tag{3.29}$$

A more compact representation of the fundamental gate follows

Lemma 3.1.5 The fundamental gate can be expressed as

$$U^{\mathcal{F}}(x;\omega,\alpha,\beta,\varphi,\lambda) = \begin{pmatrix} a_+e^{i\omega x} + a_-e^{-i\omega x} & b_+e^{i\omega x} + b_-e^{-i\omega x} \\ -b_-^*e^{i\omega x} - b_+^*e^{-i\omega x} & a_-^*e^{i\omega x} + a_+^*e^{-i\omega x} \end{pmatrix}, \tag{3.30}$$

as can be verified by simple substitution from Definition 3.1.3.

Note that this expression corresponds to a unitary matrix, due to the relations involved in the definition of the coefficients $a_\pm$ and $b_\pm$. Note also that a unitary matrix has three degrees of freedom, which are here fixed by 5 parameters. An intuition behind the role of these parameters is that $\alpha, \beta, \varphi, \lambda$ are related to the coefficients of one Fourier step, that is $a_\pm, b_\pm$, while ω can be identified with the corresponding frequency.

A total circuit can be constructed by multiplying k fundamental gates to obtain $\mathcal{U}^{(k)}_{f,\phi}$ as in Definition 3.1.1. Starting with this composite gate, the proof for the main Fourier approximation theorem 3.1.2 is feasible.

Proof. The proof of this constructive theorem consists in making contact with harmonic analysis and proceeds by induction.

i) The first circuit consists only of one fundamental gate, chosen with frequency $\omega = 0$, that is

$$U_0^{\mathcal{F}} = \begin{pmatrix} A_0 & B_0 \\ -B_0^* & A_0^* \end{pmatrix}, \tag{3.31}$$

This, indeed corresponds to the first constant term of Fourier series.

ii) It is assumed that the N-th approximant circuit takes the form for a Fourier series, but not its value constraints.

$$\prod_{i=0}^{N} U_i^{\mathcal{F}} = \begin{pmatrix} \sum_{n=-N}^{N} A_n e^{i\Omega_n x} & \sum_{n=-N}^{N} B_n e^{i\Omega_n x} \\ -\sum_{n=-N}^{N} B_n^* e^{-i\Omega_n x} & \sum_{n=-N}^{N} A_n^* e^{-i\Omega_n x} \end{pmatrix}. \tag{3.32}$$

where the frequencies are Ω_n are free, and to be fixed later. The result of adding a new fundamental gate my left multiplication. corresponds to

$$\prod_{i=0}^{N+1} U_i^{\mathcal{F}} = \begin{pmatrix} \sum_{n=-N-1}^{N+1} \tilde{A}_n e^{i\tilde{\Omega}_n x} & \sum_{n=-N-1}^{N+1} \tilde{B}_n e^{i\tilde{\Omega}_n x} \\ -\sum_{n=-N-1}^{N+1} \tilde{B}_n^* e^{-i\tilde{\Omega}_n x} & \sum_{n=-N-1}^{N+1} \tilde{A}_n^* e^{-i\tilde{\Omega}_n x} \end{pmatrix} \tag{3.33}$$

where the new coefficients $\tilde{\Omega}_n$ need to be fixed and frequencies in terms of the old ones Ω_n and the new single gate frequency ω added to the circuit. It is easy to see that the addition of a gate changes the frequency in one unit, that is, $\tilde{\Omega}_n = \Omega_n \pm \omega$. Then, the general structure of the series can be adapted to a Fourier expansion by choosing

$$\Omega_n = (2n+1)\frac{\pi}{2}. \tag{3.34}$$

After fixing the values that the frequencies must take, it is straightforward to re-arrange terms in the matrix and reach

$$\tilde{A}_0 = A_0 a_- - B_0^* b_- \tag{3.35}$$
$$\tilde{A}_{\pm n} = A_{\pm n} a_- - B_{\mp n}^* b_- + A_{\pm(n-1)} a_+ - B_{\mp(n-1)}^* b_+ \tag{3.36}$$
$$\tilde{A}_{\pm(N+1)} = A_{\pm N} a_+ - B_{\mp N}^* b_+ \tag{3.37}$$
$$\tilde{B}_0 = B_0 a_- + A_0^* b_- \tag{3.38}$$
$$\tilde{B}_{\pm n} = B_{\pm n} a_- + A_{\mp n}^* b_- + B_{\pm(n-1)} a_+ + A_{\mp(n-1)}^* b_+ \tag{3.39}$$
$$\tilde{B}_{\pm(N+1)} = A_{\mp N}^* a_+ + A_{\mp N}^* b_+ \tag{3.40}$$

This provides the explicit connection between approximant circuits and Fourier expansions for the coefficients of the global unitary matrix.

■

The above theorem is sufficient to prove that the output probability of a series of approximant circuits can reproduce any functionality. Notice that the proof ensures that for any number of re-uploadings N it is possible to arrange terms in such a way that the final output state is a Fourier series. However, the intermediate steps follow a Fourier-like expression but do not maintain the required values of coefficients and frequencies. The mathematical form is the same, but the values of different coefficients must be changed to match the Fourier ones only in the last step.

Demonstration for the quantum UAT

An alternative manner to design a single-qubit universal approximant is related to the equivalent UAT broadly used in NNs [Cyb89]. The idea is to start from a different fundamental gate.

Let the *fundamental gate* $U^{\mathrm{UAT}}(\vec{x};\vec{\theta})$ defined in Eq. (3.23) be explicitly

$$U^{\mathrm{UAT}}(x;\vec{\omega},\alpha,\varphi) = R_z(2\,(\vec{\omega}\cdot\vec{x}+\alpha))R_y(2\varphi) = \\ = \begin{pmatrix} \cos(\varphi)e^{i(\vec{\omega}\cdot\vec{x}+\alpha)} & -\sin(\varphi)e^{i(\vec{\omega}\cdot\vec{x}+\alpha)} \\ \sin(\varphi)e^{-i(\vec{\omega}\cdot\vec{x}+\alpha)} & \cos(\varphi)e^{-i(\vec{\omega}\cdot\vec{x}+\alpha)} \end{pmatrix}, \tag{3.41}$$

A full circuit can be constructed by multiplying k fundamental gates to obtain $\mathcal{U}_{f,\phi}^{(k),\mathrm{UAT}}$ as in Def. 3.1.1. The quantum UAT 3.1.4 using this fundamental gate is now proven.

Proof. The classical U^{UAT} is defined in Eq. (3.41). By direct inspection it is straigthforward to check that every entry in this matrix can be understood as one term of $\bar{f}_N$ in Eq. (3.21). From this definition the recursive rule that defines all steps is obtained. If

$$A_N = \langle 0| \prod_{n=1}^{N} U_n^{UAT} |0\rangle \tag{3.42}$$

$$B_N = \langle 1| \prod_{n=1}^{N} U_n^{UAT} |0\rangle \tag{3.43}$$

then the updating rule is

$$A_{N+1} = A_N \cos(\varphi_{N+1}) e^{i\vec{\omega}_{N+1}\cdot\vec{x}} e^{i\alpha_{N+1}} - B_N \sin(\varphi_{N+1}) e^{i\vec{\omega}_{N+1}\cdot\vec{x}} e^{i\alpha_{N+1}} \tag{3.44}$$

$$B_{N+1} = A_N \sin(\varphi_{N+1}) e^{-i\vec{\omega}_{N+1}\cdot\vec{x}} e^{\alpha_{N+1}} + B_N \cos(\varphi_{N+1}) e^{-i\vec{\omega}_{N+1}\cdot\vec{x}} e^{i\alpha_{N+}} \tag{3.45}$$

Having this updating rule in mind, it is possible to write

$$B_N = \sum_{m=0}^{2^{N-1}} c_m(\varphi_1, \dots, \varphi_N) e^{i\delta_m(\alpha_1,\dots,\alpha_N)} e^{i\vec{w}_m(\vec{\omega}_1,\dots,\vec{\omega}_N)\cdot\vec{x}}, \tag{3.46}$$

where the inner dependencies of c_m are products of sines and cosines of φ_n, and those of δ_m and $\vec{w}_m$ are linear combinations of α_n and ω_n.

The procedure follows now as in the proof of the UAT in Ref. [Cyb89]. S is the set of functions of the form $B_N(\vec{x})$, and $C^{\mathbb{C}}(I_m)$ the set of continuous complex-valued functions in I_m, defined as in Theorem 3.1.3. It is assumed that $S \subset C^{\mathbb{C}}(I_m)$, and $S \neq C^{\mathbb{C}}(I_n)$. Now, the Theorem A.2.1 is applied, known as Hahn-Banach theorem. This theorem allows to state that there exists a linear functional L acting on $C^{\mathbb{C}}(I_n)$ such that

$$L(S) = L(\bar{S}) = 0, \qquad L \neq 0. \tag{3.47}$$

Notice that this theorem is applicable since there are no restriction in working only with real numbers.

Theorem A.2.2, known as Riesz representation theorem, is called now. The functional L is

$$L(h) = \int_{I_n} h(x) d\mu(x) \tag{3.48}$$

for $\mu \in M(I_n)$ non-null and $\forall\, h \in C^{\mathbb{C}}(I_n)$. In particular,

$$L(h) = A_N(\vec{x}) d\mu(\vec{x}) = 0, \tag{3.49}$$

and thus

$$\int_{I_n} e^{iv\vec{m}(\omega_1,\dots,\omega_N)\cdot\vec{x}} d\mu(\vec{x}) = 0. \tag{3.50}$$

This is the usual Fourier transform of μ. By calling Theorem A.2.3, Lebesgue Bounded Convergence theorem, if the $\mathcal{FT}(\mu) = 0$, then $\mu = 0$, and a contradiction is encountered with the only made assumption.

The measure of all half-planes being 0 implies that $\mu = 0$. $\vec{w}$ is fixed, and for a bounded measurabe function h, the linear functional is defined as

$$F(h) = \int_{I_n} h(\vec{w} \cdot \vec{x}) d\mu(x), \tag{3.51}$$

which is bounded on $L^\infty(\mathbb{R})$ since μ is a finite signed measure. Let h be an indicator of the half planes $h(u) = 1$ if $u \geq -b$ and $h(u) = 0$ otherwise, then

$$F(h) = \int_{I_n} h(\vec{w} \cdot \vec{x}) d\mu(x) = \mu(\Pi_{\vec{w},b}) + \mu(H_{\vec{w},b}) = 0. \tag{3.52}$$

By linearity, $F(h) = 0$ for any simple function, such as sum of indicator functions of intervales [Ash72].

In particular, for the bounded measurable functions $s(u) = \sin(\vec{w} \cdot \vec{x}), c(u) = \cos(\vec{w} \cdot \vec{x})$

$$F(c + is) = \int_{I_n} \exp\{i\vec{w} \cdot \vec{x}\} d\mu(\vec{x}) = 0. \tag{3.53}$$

The Fourier Transform of this F is null, thus $\mu = 0$.

■

For the sake of completeness, the three theorems required for the proof are covered in App. A.2.

Link to output of quantum circuits

Last sections were devoted to prove that specific series of circuits return functionalities able to represent a wide range of functions. In this last step, previous results are related to the output of quantum circuits.

Theorem 3.1.6 The computational basis output of a single-qubit quantum circuit can provide a convergent approximattion to any desired function.

Proof. The output of a k-th approximant circuit can be cast a an approximation expansion of an arbitrary function. It is sufficient to initalize a register in the $|0\rangle$ state and measure the output in the computational basis. It follows

$$\langle 1| \prod_{i=0}^{N} U_i |0\rangle = z_N(x) \tag{3.54}$$

where $z_N(x)$ can take different forms and U can provide Fourier or UAT approximations.

If the fundamental gate is $U^{\mathcal{F}}$, then the output is the truncated Fourier series

$$z_N(x) = \sum_{n=-N}^{N} B_n e^{i2\pi n x}, \tag{3.55}$$

where B_n are free complex coefficients. This result holds for single-variable functions.

If the fundamental gate is U^{UAT}, then the output is a function

$$z_N(\vec{x}) = \sum_{m=0}^{2^{N-1}} c_m(\varphi_1, \ldots, \varphi_N) e^{i\delta_m(\alpha_1,\ldots,\alpha_N)} e^{i\vec{\omega}_m(\vec{\omega}_1,\ldots,\vec{\omega}_N)\cdot\vec{x}}, \tag{3.56}$$

according to Eq. (3.46). This result holds for single- and multi-variable functions.

According to Theorems 3.1.1 and 3.1.3, both expressions can approximate any desired function.

■

3.1.4 Discussion

The theorems here presented demonstrate formally that a single-qubit circuit has enough flexibility to store any complex function $z(x)$ in its inner degrees of freedom by sequentially applying single-qubit gates depending on the independent variable x and tunable parameters. This result is useful when the functional form of $z(x)$ is unknown and can only be learnt by sampling data. The learning process is performed by a classical optimizer.

This result guarantees that two different independent real functions can be represented by single-qubit circuits. It provides the highest possible degree of compression of data in such a small state since there is no more possible room in the Hilbert space. In addition, there are no fundamental limitations in the complexity or dimensionality of the functions that can be encoded into the circuit.

Two different approaches were followed for the proof, leading to two different sets of single-qubit gates. First, there is a link between Fourier series and quantum circuits if a quantum gate with 5 tunable parameters is defined, only if x is one-dimensional. A quantum circuit composed by N gates provides an output state whose components can be written as a N-terms Fourier series. Unlike in the classical case, since the process is variational, this result ensures at least the Fourier series, but permits better approximations. This proof inherits the assumptions of Fourier theorems. For the second proof, a link with the UAT was found by applying gates depending on 3 parameters. Such circuit delivers a output state compatible with the formulation of UAT. This proof is inherited from the equivalent one for NNs and supports only approximation for continuous functions. In exchange, multidimensional dependencies on x are supported as well.

These results can serve as a starting point for studying the expressibility of quantum systems beyond one qubit, see also [NY21; SJA19; SSM21]. When more qubits are added, the exponential size of the Hilbert space triggered by the entanglement is likely to play an essential role to be fully understood yet.

The core of the proof relies on the non-linearities arising from the consecutive application of non-commuting gates, and the linear encoding of data as arguments for the different gates. As a consequence, rotations around only two axis are necessary and sufficient to achieve any kind of non-linearities. The linear encoding provides unbiased approximations. Therefore, although the theorems give support to specific designs of quantum gates, it is likely that some other gates constructed taking these two ingredients into account can also provide accurate approximations. This can be advantageous for some problems with challenging properties.

These results will serve as full theoretical support to the examples provided in Secs. 3.2, and as partial support to the examples provided in Secs. 3.3 and 3.5.

3.2 Numerical benchmark

In this section, the practical performance of the theorems explained in Sec. 3.1.2 is explored by fitting test functions from sample data using the theoretical models here presented. Two different kinds of benchmarks for real and complex functions, respectively, are presented. These benchmarks collect results using both $U^{\mathcal{F}}$ and $U^{\rm UAT}$ gates from Defs. 3.1.3 and 3.1.4. Benchmarks are performed using first simulations that include no decoherence nor sampling uncertainty, and then passing the obtained results to a quantum device to test the experimental performance. Simulations with up to 6 layers are considered.

The aim of this benchmark is to compare the results of quantum and classical methods. The classical Fourier representation can be obtained by following Theorem 3.1.1. In the UAT case, the description from Theorem 3.1.3 is followed, with $\sigma(\cdot)$ being a cosine for real functions and $e^{i(\cdot)}$ for complex functions. The parameters are found by employing specific classical optimization methods. For the quantum UAT case, $H\left|0\right\rangle = \left|+\right\rangle$ is taken as the initial state. Note that the choice of initial state does not compromise the validity of any result since it is possible to transform any state into any other by adjusting the first layer of the approximation method.

All simulations are performed using the framework `QIBO` [Eft+20a]. The code computing the numerical experiments as well as the final results can be found on `GitHub` [Pér21]. Benchmarks for real and complex functions are described in Sec. 3.2.1 and Sec. 3.2.2, respectively. Results are covered in Sec. 3.2.3. Final remarks can be read in Sec. 3.2.4.

3.2.1 Benchmark for real functions

For the first benchmark, a single-variable, real-valued function $-1 \leq f(x) \leq 1$ related to the observable $\langle Z \rangle \sim f(x)$ is considered. The quantum state to represent is then

$$|\psi(x)\rangle_Z = \sqrt{\frac{1+f(x)}{2}}\,|0\rangle + e^{i\phi}\sqrt{\frac{1-f(x)}{2}}\,|1\rangle\,, \tag{3.57}$$

where ϕ is a phase that in general may be x-dependent, but it is neglected at this stage. The χ^2 function that drives the optimization is then

$$\chi^2 = \frac{1}{M}\sum_{j=1}^{M}\left(\langle Z(x_j)\rangle - f(x_j)\right)^2, \tag{3.58}$$

where M is the total number of samples of x.

The Z benchmark is first tested against four different functions of interest

$$\begin{aligned} \mathrm{ReLU}(x) \quad &= \quad \max(0,x), &(3.59)\\ \tanh(ax) \quad &\text{for} \quad a=5, &(3.60)\\ \mathrm{step}(x) \quad &= \quad x/|x|; \quad 0 \text{ if } x=0, &(3.61)\\ \mathrm{poly}(x) \quad &= \quad |3x^3(1-x^4)|. &(3.62)\end{aligned}$$

All functions are conveniently rescaled to fit the limits $-1 \leq f(x) \leq 1$. In all cases, $x \in [-1,1]$. The ReLU$(\cdot)$ and tanh$(\cdot)$ functions are chosen given the central role they play in the field of ML. step$(\cdot)$ presents a discontinuity, which implies a challenge in the approximation. poly$(\cdot)$ is chosen as it contains wavy features arising from non-trigonometric functions.

Next, the approach is tested against four functions of two variables in order to check how the quality of the approximations evolves as more dimensions are added to the problem. Those are known 2D functions named `adjiman, brent, himmelblau, threehump` [Ard16]. These functions are chosen as representatives of a variety of difficulties the algorithm needs to overcome. In the 2D case, the functions are conveniently rescaled to fit the limits $-1 \leq f(x,y) \leq 1$ and $(x,y) \in [-5,5]^2$. A definition of these functions can be found in Appendix A.3.

In this benchmark, both the UAT and Fourier quantum and classical methods are considered for the one-dimensional functions. However, 2D functions are only tested for UAT methods since theorem 3.1.2 from Sec. 3.1 does not support multidimensional Fourier series.

3.2.2 Benchmark for complex functions

In order to test the performance of the presented algorithm for fitting complex functions, a tomography-like benchmark is proposed. Since complex functions have real and imaginary parts, one needs to measure at least two observables in the qubit space. In this case, the observables are $\langle X \rangle$ and $\langle Y \rangle$ for the real and imaginary parts, that is $\langle X \rangle + i \langle Y \rangle \sim f(x) e^{ig(x)}$. The quantum state that permits this identification is

$$|\psi(x)\rangle_{XY} = \sqrt{\frac{1 + \sqrt{1 - f(x)}}{2}} \, |0\rangle + e^{ig(x)} \sqrt{\frac{1 - \sqrt{1 - f(x)}}{2}} \, |1\rangle \, . \quad (3.63)$$

for $0 \leq f(x) \leq 1$It is then possible to construct a χ^2 function as

$$\chi^2 = \frac{1}{M} \sum_{j=1}^{M} \left| \langle X(x) \rangle + i \langle Y(x) \rangle - f(x) e^{ig(x)} \right|^2 . \quad (3.64)$$

For the $X - Y$ benchmark the algorithm is tested against all possible combinations of real and imaginary parts of the functions defined in Eqs. (3.59)–(3.62), conveniently renormalized to ensure that $\langle X \rangle^2 + \langle Y \rangle^2 \leq 1$.

3.2.3 Results

In all results presented in this section there are three different final values. First, the Fourier and the UAT classical methods are used to approximate a target function. The Fourier method is obtained following the constructive recipe of Th. 3.1.1. The UAT is applied using a single-hidden-layer NN. Second, the same function is approximated using the quantum procedures defined in this work, simulating the wave function evolution with classical methods. In both cases, the best outcome obtained with different initial conditions used in the optimization step is retained. Finally, the parameters obtained using the simulation of the quantum procedure are translated to the experimental device to execute the circuit of interest in the actual superconducting machine. Details concerning the experimental implementation can be found in App. A.4. The theoretical optimal parameters may be, in principle, different than the experimental ones. Hence, an optimization performed directly on the experimental parameters could improve the final results [Pér+–].

The resulting fit for all four single-variable real-valued functions from Eqs. (3.59)–(3.62) is shows in Fig. 3.3. In this case the Z benchmark with 5 layers is considered. A classical approximation (blue), a quantum exact simulation (red) and its experimental implementation (green) are depicted. All methods follow the overall shape of the target function. Classical Fourier approximations return less accurate predictions on the value of $f(x)$ due to the periodic nature of the model. The quantum Fourier and both classical and quantum UAT models return better results for all values of x. This behaviour is observed in all benchmarks. The experimental results retain the qualitative properties of the exact models,

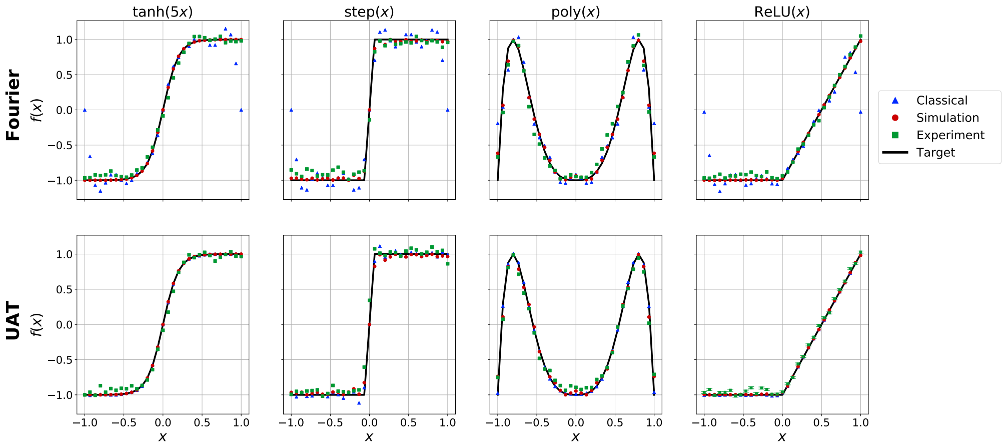


Figure 3.3: *Fits for four real-valued functions using the Z benchmark with five layers. Blue triangles represent classical models, namely Fourier and UAT, while red dots represent its quantum counterparts computed using a classical simulator. Green squares are the experimental execution of the optimized quantum model using a superconducting qubit. The target function is plotted in black for comparison. The analysis for experimental errors is plotted for the* ReLU *function and the UAT model.*

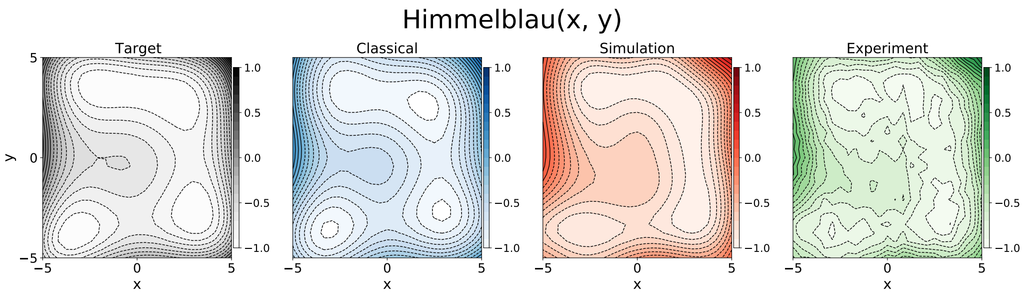
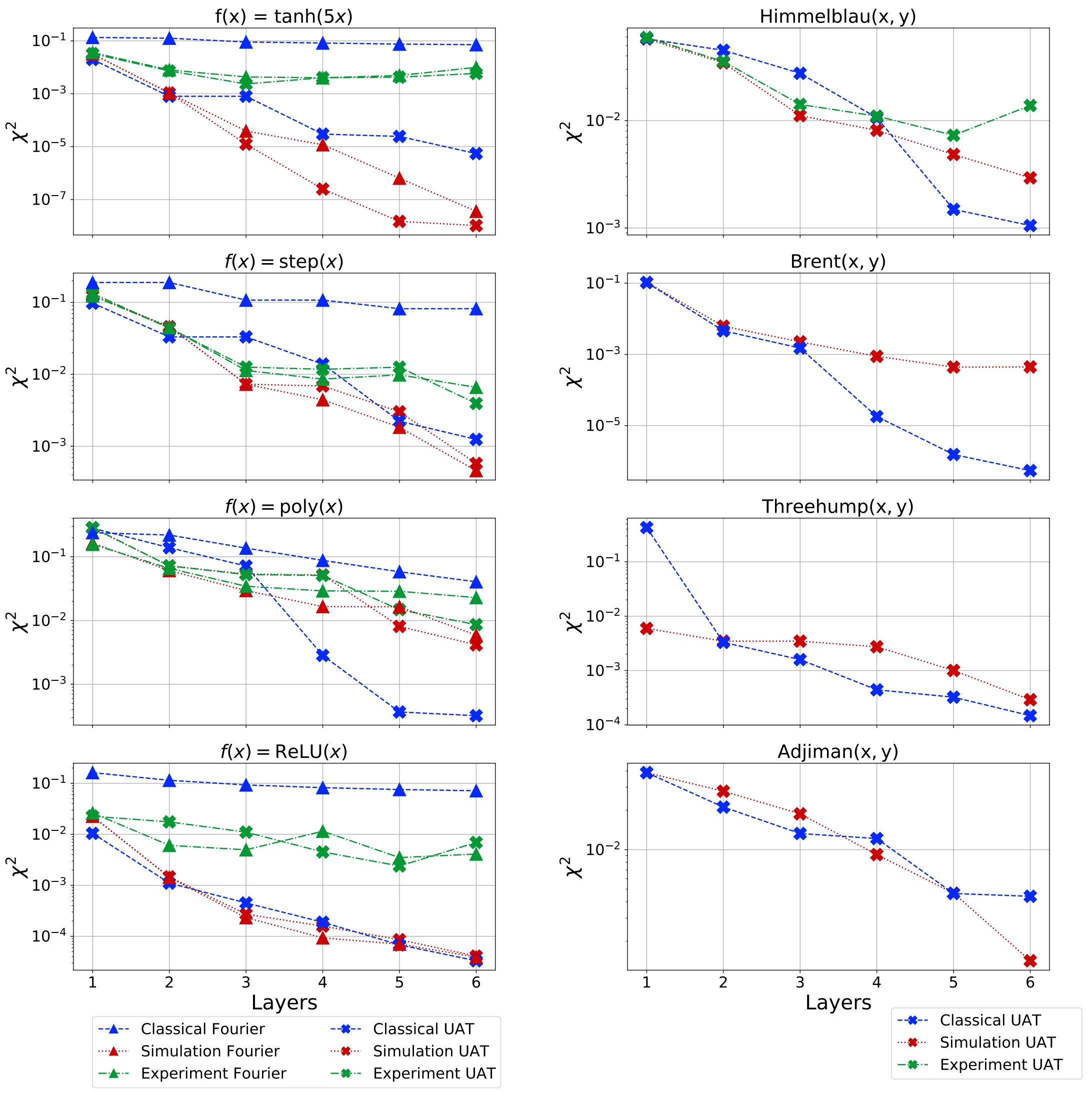


Figure 3.4: *Fits for the 2D function Himmelblau properly normalized using the Z benchmark for five layers. The blue (classical) plot represents the classical UAT model, while the red (simulation) plot represents its quantum counterpart simulated. The green (experiment) plot is the experimental execution of the optimized quantum model. The target function is painted in black (target). In all drawings, the lines corresponds to the same levels in the Z axis.*

although a loss in performance is visible. In addition, an analysis of experimental uncertainties is also depicted at the UAT ReLU plot from Fig. 3.3.

Fig. 3.5(a) shows a summary of the values of χ^2 for classical and their



(a) 1D functions (b) 2D functions

Figure 3.5: *Values of χ^2 for the Z benchmark in all four test 1D and 2D functions using classical computation (blue scatter), classical simulation of the quantum algorithm (red scatter) and experimental implementation with a superconducting qubit (green scatter). Fourier models are depicted with triangles, while UAT models are represented by crosses. In the 2D case, only UAT models are considered.*

analogous quantum simulated models and their experimental validation. In the case of classical and simulated quantum models a general trend towards better approximations –implying lower values of χ^2– is observed with an increasing numbers of layers.

The simulated Fourier model performs better than its classical counterpart. This is due to the fact that a classical Fourier series does not contain tunable parameters, while its quantum version does. However, the result from the classical Fourier series constitutes a lower bound for any approximation method based on optimization since at least the quality of the Fourier series is guaranteed.

In the UAT case of Fig. 3.5(a), no approach returns significantly better results. The classical algorithm performs better in the poly(x) case, but the results with the simulated quantum method improve the classical ones in the $\tanh(5x)$ case. Both models present similar trends as the number of layers increases.

Despite the fact that the Fourier model contains more parameters

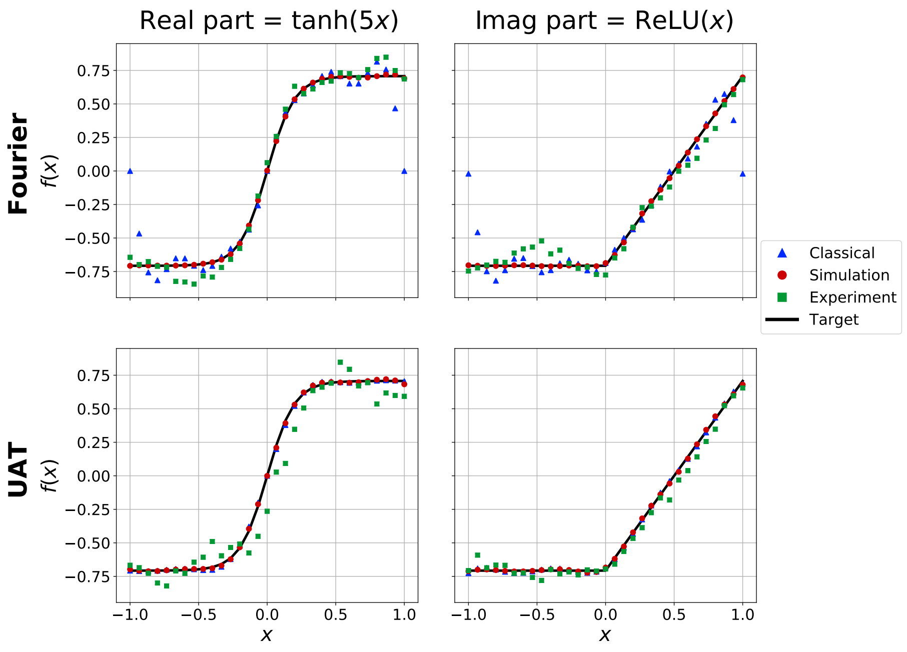


Figure 3.6: *Fits for the complex function $f(x) = \tanh(5x) + i\mathrm{ReLU}(x)$ properly normalized using the $X - Y$ benchmark for five layers. Blue triangles represent a classical model, while red dots represent its quantum counterparts computed using a classical simulator. Green squares are the experimental execution of the optimized quantum model using a superconducting qubit. The target function is plotted in black for comparison.*

than the UAT model, the latter performs better as seen in Fig. 3.5(a). Therefore, the UAT method seems more appropriate for the functions used here.

The experimental realization of the quantum approximation models suffers from circuit noise and sampling uncertainties, and therefore degrades the quantity χ^2. This is more prominent as more layers are added to the model. As a direct consequence, the approximation of the quantum model to the target function loses accuracy. The inherent sampling uncertainty sets a lower bound in the value of χ^2 obtained through experiments.

In general, Fig. 3.5(a) supports the claim that every layer grants the model more flexibility, and thus enhances the capability of fitting the target function. This flexibility is given by the number of re-uploadings of the independent variable and not by the amount of parameters. In

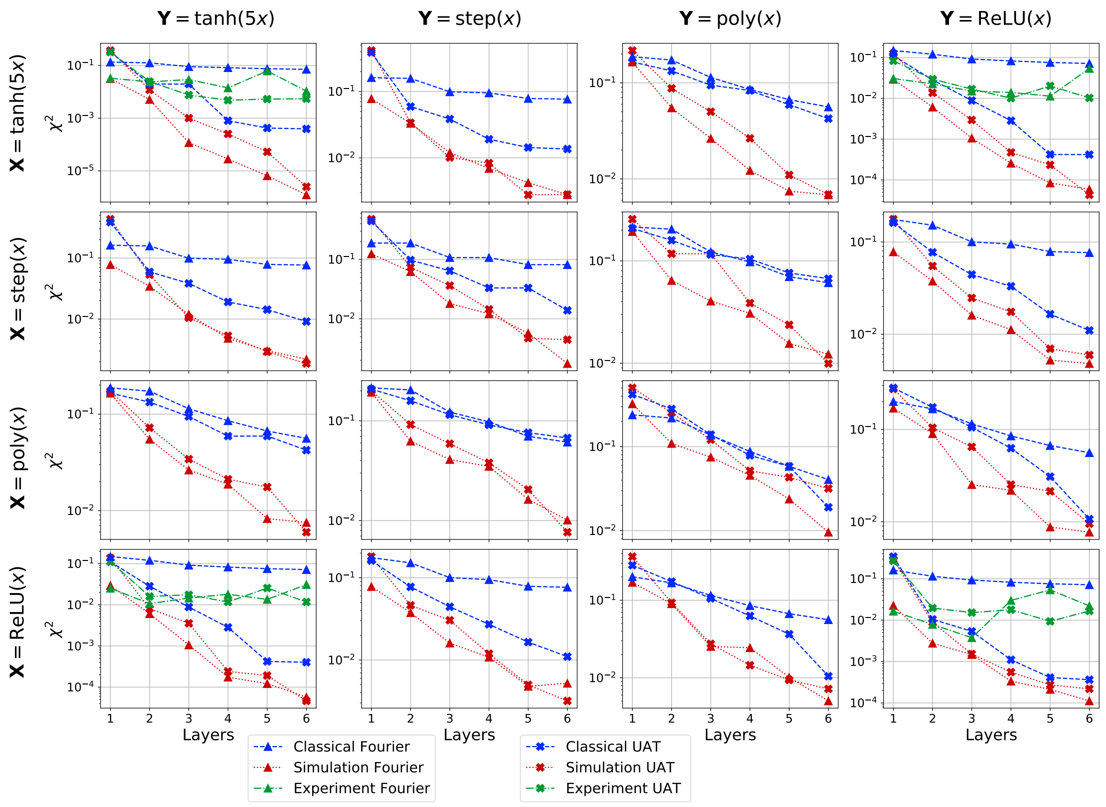


Figure 3.7: *Values of χ^2 for the X-Y benchmark in all possible combinations for real and imaginary parts of the four test functions from Eqs. (3.59) to (3.62) using classical computation (blue scatter), classical simulation of the quantum algorithm (red scatter), and experimental implementation with a superconducting qubit (green scatter). Fourier models are depicted with triangles, while UAT models are represented by crosses.*

addition, having too many parameters likely hinders the optimization procedure.

Figure 3.4 depicts the approximations obtained for the Himmelblau(x, y) function comparing the target function and all different methods considered. Figure 3.5(b) summarizes the values of χ^2 for all 2D-functions taken into account in this work.

All different executions capture the overall shape of the function, but some differences exist in the different plots. Classical simulations return values for $Z < -1, Z > 1$, and thus lead to three minima in this case. On the other hand, the quantum simulation cannot clearly distinguish those minima. The experimental execution presents sharp contours because of the inherent noise and sampling uncertainty.

The values of χ^2 in Fig. 3.5(b) measure the accuracy of the approximations. As before, is is seen that a larger number of layers provides better approximations to the target function. In agreement to the one-dimensional Z benchmark, the scaling is similar for both quantum and classical methods.

A complex function in the $X - Y$ benchmark is depicted in Fig. 3.6. In that case, the X measurement leads to $\tanh(5x)$ while the imaginary part contains $\mathrm{ReLU}(x)$. All the observations made for the Z benchmark hold in this case.

Fig. 3.7 shows values of χ^2 for all possible combinations of real and imaginary parts using the functions described in Eqs.(3.59)–(3.62), being the real and imaginary parts. In this case, it is possible to see a common advantage for the quantum models. In particular, the functions $\tanh(5x)$ and $\mathrm{ReLU}(x)$ work better in any combination. This reflects the behaviour already observed in Fig. 3.5(a), where these functions present better performance than other functions considered.

3.2.4 Discussion

The results presented in this section are the numerical and experimental confirmations that the theoretical works presented in Sec. 3.1 have some utility in practice. With this purpose, numerical evidences on the flexibility and approximation capabilities of the quantum circuits are given.

The benchmarks were obtained using first simulations of quantum systems optimized by classical means for a set of test functions. As benchmarks, 1D and 2D real functions, and 1D complex functions are included. Final results are compared against classical counterparts. In general, it is possible to see that quantum and classical methods scale equivalently. This is the numerical confirmation that quantum procedures can perform similarly to classical ones, at least theoretically.

A further step has been carried by implemented the found solutions in an actual quantum device built upon superconducting trasmon qubits.

No further optimization is performed in this stage. Experimental results confirm the trend seen by simulation, and the finite coherence and imperfections of the qubit do not seem to impact results significantly for small numbers of gate.

The obvious next step to tackle in this experiment is to actually carry the optimization procedure on the quantum device. However, the lack of accurate retrievement of values for the cost function limits the performance achievable in this direction. Future work will deal with problems of this kind [Pér+–].

3.3 Re-uploading for a quantum classifier

The re-uploading strategy can be used to address classification problems with minimal amounts of quantum resources. As it is seen in this section, even a single-qubit circuit is a system versatile enough as to build a universal quantum classifier upon it. The computational capabilities are obtained by combining quantum features and classical optimization subroutines. In fact, there exist no theoretical limits in the dimensionality of data that can be classified with this strategy. On the other hand, multiple class classification can be accomodated even in the smallest possible Hilbert space. The main reason why this is possible is the density of the Hilbert space the quantum processing takes place in.

The single-qubit classifier here proposed does not attempt to address classically intractable problems, and the achievement of quantum advantage is out of scope. The focus of the work of this chapter is rather to illustrate that even the most minimalistic quantum systems provide large computational power. In this work, the aim is to distill the minimal amount of quantum resources required to solve a given supervised learning problem in practice. Quantum resources include qubits and gates. The problems to be solved in this section are not trivial, even though they can be expressed with simple datasets.

This quantum classifier also extends the idea of re-uploading to multiple qubits and entangling architectures. There is currently no theoretical work supporting the universality of such circuits. However, extensive benchmarks were carried to compare the efficiency of this strategy as entanglement expands the superpositions carried along with the classification. The main result in this section is to show that there is a trade-off between the number of qubits and depth of the circuit. That is, fewer qubits may be used at the price of re-uploading data in several steps along the computation. In summary, similar results can be obtained for similar query complexities even in the case of different architectures. Thus, despite the lack of theoretical evidence, it is expected that quantum systems with several entangled qubits possess more capabilities to perform classification than the single-qubit systems. To what extent complex systems outperform simple ones is still unclear.

The power of single- and multi-qubit classifiers following the data re-uploading strategy is illustrated by a series of examples. All the examples here presented were computed by means of classical simulations. Later, this approach was experimentally demonstrated on a trapped-ions quantum computer, see Sec. 3.4. First, it is attempted to distinguish between points in a plane divided in two classes. The classes are determined by a geometrical boundary. Then, the number of regions, that is classes, is extended on the plane to address more complicated datasets. As a last extension, the dimensionalty of the datasets is increased from plane (2D) to points in space (3D) and hyperspace (4D). A complete benchmark is done for every problem as addressed with quantum circuits with differentes properties, that is number of qubits and entanglement schemes.

This section is structured as follows. First, the adaptations needed to transform the general re-uploading strategy to a single-qubit quantum classifier scheme including measurement strategy are explained in Sec. 3.3.1. The extension from single- to multi-qubit is depicted in Sec. 3.3.2. Results are reviewed in Sec. 3.3.3. Sec. 3.3.4 covers final comments.

The contents of this section are based on the work in Ref. [Pér+20a]. This work constitutes the seminal paper of all the re-uploading strategy. It is important to mention that some technical details were polished as more comprehension of the scheme was acquired. These improvements, however, do not compromise the validity of the results here displayed, even though these implementations do not constitute an optimal one.

3.3.1 Quantum classifier

In this section the different ingredients required to build the quantum classifier are described in detail, namely the quantum circuit, the measurement strategy, the cost functions and the general working principle for mixing all pieces in a consistent way.

Quantum circuit

We take as the starting line of the quantum classifier the general scheme defined in Def. 3.1.1, and specify the quantum circuit used in terms of their processing gates. For this application of the re-uploading strategy, the gates take the role of Defs. 3.1.3 and 3.1.4, but they do not exactly match the ones previously proposed. Nevertheless, it mantains the most important properties, namely re-uploading and linear encoding of data, and non-commuting matrices interspersed. In the single-qubit classifier, data is introduced in simple rotations which are easy to characterize. The definition of each layer is

$$U(\vec{x};\vec{\theta},\vec{w}) = U_3(\vec{\theta}+\vec{w}\circ\vec{x}), \tag{3.65}$$

where $\vec{\theta}, \vec{w}, \vec{x}$ are 3D vectors and $\vec{w} \circ \vec{x} = \left(w^1 x^1, w^2 x^2, w^3 x^3\right)$ is the Hadamard product of two vectors. U_3 is the most general single-qubit unitary gate defined as $U_3(\vec{\theta}) = R_z(\theta_1) R_y(\theta_2) R_z(\theta_3)$. In case the data points have dimension lesser than 3, the rest of the components of $\vec{x}$ until reaching this dimensionality are set to 0, and the corresponding terms in $\vec{w}$ are then irrelevant.

It is also possible to enlarge the dimensionality of the input space in the following way. The definition of one layer can be extended to

$$\hat{U}\left(\vec{x}; \vec{\theta}, \vec{w}\right) = U\left(\vec{x}^{(j)}; \vec{\theta}^{(j)}, \vec{w}^{(j)}\right) \cdots U\left(\vec{x}^{(1)} \vec{\theta}^{(1)}, \vec{w}^{(1)}\right), \tag{3.66}$$

where each data point is divided into j vectors of dimension three. In general, each unitary U could absorb as many variables as degrees of freedom in an SU(2) unitary. Each set of variables act at a time, and all of them have been shown to the circuit after j iterations. Then, the layer structure follows. The complexity of the circuit only increases linearly with the size of the input space. This recipe to enlarge dimensionality is not supported by the theorems from Sec. 3.1. However, numerical results show that slight modifications on the scheme permit to keep the functioning principle of the quantum algorithm.

Further refinement and understanding of the re-uploading technique sheds light on some properties of the proposed circuit that could be more efficient only in the single-qubit case. Since one layer is defined as the sequence of gates $R_z R_y R_z$, when two layers are set together pairs of R_z gates appear in consecuive positions. These gates do commute between them, and therefore the inner parameters of those gates can fuse to give rise to only one parameter, and non-linearities do not rise. This way, the overall depth of the circuit is reduced. As a consequence, this scheme includes some tunable parameters whose presence does not entail any improvement in the capability of the circuit.

For the sake of completeness the complete circuit is made further explicit now. Following the recipe from Def. 3.1.1, the overall operation for the classifier is

$$\mathcal{U}^{(k)}(\vec{x}; \Theta, W) = \prod_{i=1}^{k} \hat{U}(\vec{x}; \vec{\theta}_i, \vec{w}_i), \tag{3.67}$$

where the optimals parameters $\Theta = \{\vec{\theta}_i\}, W = \{\vec{w}_i\}$ are different for each layer (i). Then, the output state of this circuit depending on $\vec{x}$ will be

$$|\psi(\vec{x}; \Theta, W)\rangle = \mathcal{U}^{(k)}(\vec{x}; \Theta, W) |0\rangle \tag{3.68}$$

The quest for optimal configurations of Θ and W is done by classically minimizing some loss function. This loss function χ^2 will depend both on the parameters and on the dataset to be classified. The quantity

χ^2 is designed in such a way that $\chi^2 \to 0$ implies that the circuit can correctly guess all the classes from the training dataset. However, it is worth to mention than an optimal set of parameters do not always imply a good performance on unseen test datasets, since classifiers must learn to generalize their training sets, see 2.1.1. Therefore, to measure the quality of the quantum classifier the accuracy $\mathcal{A}$, measured on the testset as

$$\mathcal{A}(\Theta, W) = \frac{\text{Correct guesses}}{\text{Number of samples}}, \tag{3.69}$$

is considered. Thus, benchmark not only the adaptability of the quantum classifier, but also its generalization capability are benchmarked

Measurement

The measurement strategy to retrieve useful information for the classification from the circuit is key to achieve versatile models. The final states $|\psi(\vec{x}; \Theta, W)\rangle$ are measured, and the results are used to compute the cost function χ^2 that quantifies the error made in the classification of the training set. The minimization of this quantity in terms of the classical parameters of the circuit can be organized using any preferred supervised machine learning technique.

To design a successful measurement recipe find an optimal way to distinguish among output quantum states belonging to different classes must be found. Following the guiding principle of Ref. [Hel76] the optimal scenario to distinguish between two quantum states appears when the states are orthogonal. This is easily achieved for the single-qubit case for binary classifications. Classes A and B are attached to two orthogonal states, for instance $|0\rangle$ ad $|1\rangle$. The output state of the circuit is then measured, with a probability $P(0), P(1)$ of obtaining $|0\rangle, |1\rangle$ respectively, with $P(0) + P(1) = 1$. A given datapoint is then guessed as A if $P(0) > P(1)$, and as B otherwise. A more flexible criterium is possible by introducing some boundary λ. In this case a point is guessed as A if $P(0) > \lambda$. Notice that for $\lambda = 1/2$ the direct comparison is recovered. Thus, the presence of λ can only improve the final result. An optimal λ is chosen to get the best possible accuracy $\mathcal{A}$ on some unseen test set.

It is convenient for understanding more complex datasets to picture the division of the Hilbert space into different areas, where each area corresponds to a different class. In this dichotomic classification, the Bloch sphere is divided in northern $|0\rangle$ and southern $|1\rangle$ hemispheres. The parameter λ can move the boundary from the equator to some other convenient parallel.

The assignment of classes to the output reading of a single qubit becomes a much more difficult problem when many classes are involved in the dataset. The Hilbert space spanned in a single qubit is two dimensional, and there only exist pairs of orthogonal states. Thus, it is convenient to develop some other strategies to distinguish between different classes when measuring the output states.

The geometrical vision provides insights to address this problem. One possible strategy is to divide the Bloch sphere through parallels. The quantity $P(0)$ is compared to three thresholds $0 \leq \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq 1$. The value of $P(0)$ must then fall into one of these areas. A second more robust strategy consists in dividing the Bloch sphere into equivalent classes as separate as possible. This is obtained by computing the overlap between the output state and some label-state that targets a given class. The set of label-states is chosen in such a way that the members are maximally orthogonal among them. This strategy divides the Bloch sphere into different classes, where the boundary between two of them lies on the points where the relative fidelity of a given state with two different label states is equal.

Finding the optimal set of label-states is a problem on its own. It can be solved almost trivially in a small number of cases. For 2 classes, solutions are $|0\rangle$ and $|1\rangle$. For 3 classes, the label-states must form a regular triangle in any equatorial plane. For 4 classes, a tetrahedron is needed. This opens up the field of platonic solids [AS03], that are the solutions to 4, 6, 8, 12 and 20 classes, corresponding to the number of verteices. For other numbers of classes, solutions must be found through some other method. Figure 3.8 shows the particular cases that can be applied to a classification task of four and six classes.

In general, a good measurement strategy may need some prior computational effort and refined tomography of the final state. For a single-qubit classifier, the tomography protocol will only require three measurements. There is an alternative experimental approach to avoid tomography. Comparing some output state with some other is equivalent to measure their relative fidelities [NC10], that is

$$\mathcal{F}_y(\vec{x}, \Theta, W) = |\langle\phi_y|\psi(\vec{x}; \Theta, W)\rangle|^2 \tag{3.70}$$

This can be done by adding a unitary gate U_y at the end of the circuit such that $V_y|0\rangle = |\phi_y\rangle$ and measuring the probability of getting $|0\rangle$ as

$$\mathcal{F}_y(\vec{x}, \Theta, W) = |\langle 0| V_y^\dagger \mathcal{U}(\vec{x}; \Theta, W) |0\rangle|^2 \tag{3.71}$$

In the case where more qubits are available, a swap test can be used as well [Kan+19].

A fidelity cost function

We propose a simple cost function motivated by the geometrical interpretation introduced above. The final goal of the quantum classifier

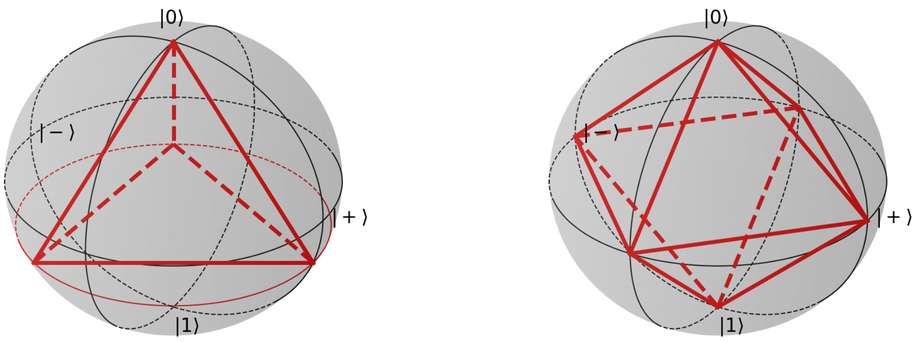


Figure 3.8: *Representation in the Bloch sphere of four and six maximally orthogonal points, corresponding to the vertices of a tetrahedron and an octahedron respectively. The single-qubit classifier will be trained to distribute the data points in one of these vertices, each one representing a class.*

is then to force the output states $|\psi(\vec{x};\Theta,W)\rangle$ to be geometrically close to the corresponding label state. That implies that the average fidelity between the output states and the corresponding label states must be maximal in average. The following cost function carries out this task:

$$\chi_f^2(\Theta,W) = \sum_{\{\vec{x},y\}}^{M} \left(1 - \mathcal{F}_y(\vec{x},\Theta,W)^2\right), \tag{3.72}$$

where data and classes pairs $\{\vec{x},y\}$ run over the training dataset.

A weighted fidelity cost function

The next step is to define a refined version of the previous fidelity cost function to be minimized. Maximixing the overlap between output states and the label states is equivalent to minimizing the overlap to other label states. Since the room in a single-qubit Hilbert space has dimension two, the latter overlap cannot be zero. A vector whose components are the relative fidelities among the label state is defined as,

$$(Y_y)_i = |\langle\phi_y|\psi_i\rangle|^2. \tag{3.73}$$

This quantity stores the expected fidelities for a successful classification. For example, given a four-class classification and using the vertices of a tetrahedron as label states (as shown in Figure 3.8), the ideal output state would have complete overlap with the corresponding class, say 0, and a relative fidelity 1/3 for any other class. Thus, $\vec{Y_y} = (1, 1/3, 1/3, 1/3)$.

One expects $Y_s(\vec{x}) = 1$, where s is the correct class, and $Y_r(\vec{x}) = 1/3$ for the other r classes. In general, $Y_y(\vec{x})$ can be written as a vector with one entry equal to 1, the one corresponding to the correct class, and the others containing the overlap between the correct class label-state and the other label-states.The expected fidelities are then compared with the obtained fidelities for a given output state of a successful classification, $Y_y(\vec{x})$.

With this definition, a cost function inspired by conventional cost functions in artificial NNs can be constructed. By weighting the fidelities of the final state of the circuit with all label states, the *weighted fidelity* cost function is

$$\chi^2_{wf}(\Theta, W, \vec{\alpha}) = \frac{1}{2} \sum_{\{\vec{x},y\}}^{M} \left(\sum_{j=1}^{\mathcal{C}} \left(\alpha_j \mathcal{F}_j(\vec{x}; \Theta, W) - (Y_y)_j \right)^2 \right), \tag{3.74}$$

where M is the total number of training points, $\mathcal{C}$ is the total number of classes, $(\vec{x}, y)$ are set of training data and classes and $\vec{\alpha} = (\alpha_1, \cdots, \alpha_{\mathcal{C}})$ are introduced as *class weights* to be optimized together with Θ and W parameters. The weighted fidelity cost function needs more parameters than the fidelity one. Notice however that the parameters in $\vec{\alpha}$ have the capability to enlarge and decrease the size of the areas corresponding to the different classes by weighting the classes. Large values of α_y increase the probability to guess the class y. This is useful when datasets are unbalanced.

The main computational difference between χ^2_f and χ^2_{wf} lies in how many relative fidelities must be measured to compute the cost function. The χ^2_{wf} requires as many fidelity measurements as classes for every evaluation of the cost function in the optimization subroutine, while the χ^2_f needs just one. For few classes and one qubit, it is not a such a big difference since tomography is not extremely costly. The extra cost in terms of number of parameters and then hardness of the optimization affects only the classical part of the computation. The classical requirements are increased to reduce the overall contribution of quantum resources. This fact links with the NISQ line of thought.

Besides the larger computational cost of χ^2_{wf} with respect to χ^2_f, there is a qualititative difference betweeen both. The fidelity cost function retrieves information about how close an output state is from its corresponding label state. Thus, the only mechanism of the optimizer to obtain good results is to minimize this distance. Loosely speaking, χ^2_f looks for an optimal configuration by moving towards where the output state should be. On the contrary, χ^2_{wf} retrieves a more complete information, since the distance to all label states is measured in every interaction. There is then more insight when updating the parameters in every new evaluation. The geometrical interpretation is that the output state moves away from the wrong classes while approaching the right one. The differences between both cost functions induce that the χ^2_{wf} is expected to provide better results.

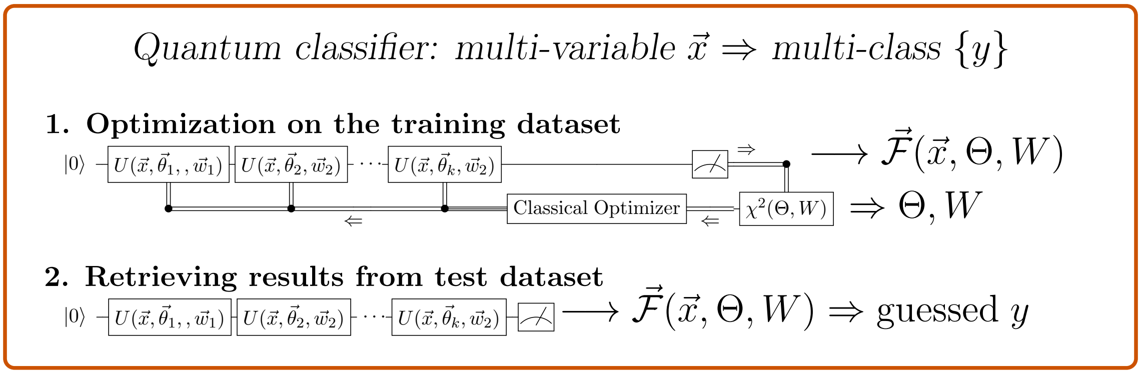

Quantum classifier: multi-variable $\vec{x} \Rightarrow$ multi-class $\{y\}$
1. Optimization on the training dataset
$|0\rangle$
$U(\vec{x},\vec{\theta}_1,,\vec{w}_1)$
$U(\vec{x},\vec{\theta}_2,\vec{w}_2)$
$U(\vec{x},\vec{\theta}_k,\vec{w}_2)$
$\longrightarrow \vec{\mathcal{F}}(\vec{x},\Theta,W)$
Classical Optimizer
$\chi^2(\Theta,W)$
$\Rightarrow \Theta, W$
2. Retrieving results from test dataset
$\longrightarrow \vec{\mathcal{F}}(\vec{x},\Theta,W) \Rightarrow$ guessed y

Figure 3.9: *Functioning scheme of the quantum classifier. In the first stage, a training dataset is used to find optimal parameters. The output states are measured to construct a cost function χ^2_f, χ^2_{wf}, to be minimized. Once this process is complete, second stage starts. In this step, the quantum circuit is executed using data points vecx from the testset. Fidelities are measured, and from that information a class y is guessed.*

Optimization and retrievement of results

The cost function defines the two steps needed to carry classification task, namely the training and the retrievement of results.

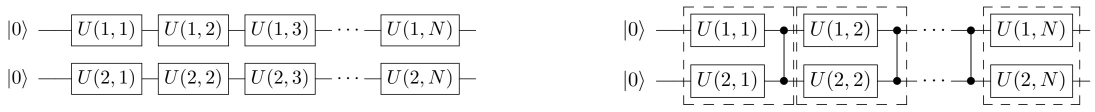
For optimizing the circuit, the well-known hybrid classical-quantum models can be used, see Fig. 3.9. The quantum circuit is executed and measured several times. From those measurements the fidelity results from Eq. (3.70) are retrieved. If the fidelity cost function χ^2_f is used, then is is only needed to measure the fidelity with respect to the label state corresponding to the class of the data point. If χ^2_{wf} is the chosen cost function, then all fidelities must be measured for each point. Then, those results are given to the classical optimizer to look for the optimal set of parameters Θ, W.

When the set of optimal parameters has been obtained, it is time to run the quantum circuit with data points from the testset. At the end of the execution, measurements are made to obtain the relative fidelities between the output state and all label states. In this step it is compulsory to take all possibilities into account since no prior knowledge is available. From that information, it is possible to guess a class y, commonly by choosing the largest fidelity. The number of correctly guessed points will determine the accuracy $\mathcal{A}(\Theta, W)$.

3.3.2 From single- to multi-qubit quantum classifier

Entanglement makes the dimensionality of the Hilbert space of composite quantum systems increase exponentially as more particles are added. This phenomenon lies at the core of the impossibility to simulate quantum mechanics by classical means. Therefore, quantum advantage can only be achieved by those quantum algorithms whose number of qubits is at least moderately large, only if they are entangled. In this case, the single-qubit quantum classifier cannot expect to overcome any classical method.

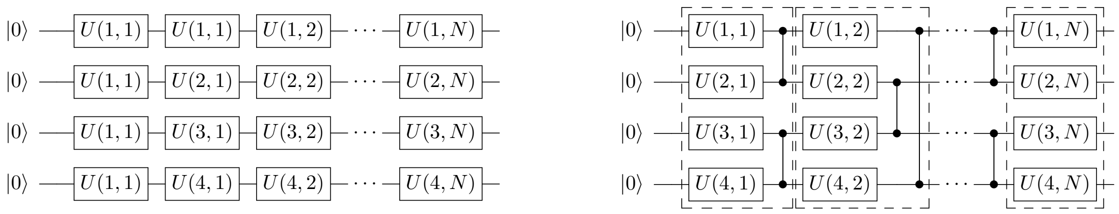
On the other hand, the classical analog to the single-qubit classifier here presented is a single-hidden-layer NN, as stated in Theorem 3.1.4. Nevertheless, using this architecture is rare. The amount of neurons needed to achieve good performance and the inefficiency of training methods prevents an extended use of this NN. Instead, other kinds of NNs were developed to circumvent this limitation. One of the most celebrated models is the deep FfNN, see Sec. 2.1.1. In this case, many neurons are distributed through several layers, and each layer is connected to the previous and next ones, except for the extrema. Connections between two consecutive layers can happen in principle among all possible pairs of neurons. In this case, data is processed in several steps, and more



(a) Ansatz with no entanglement

(b) Ansatz with entanglement

Figure 3.10: *Two-qubit quantum classifier circuit without and with entanglement. Here, each layer includes a single-qubit operation with data re-uploading per plus a CZ gate if entanglement is considered. The last layer never carries an entanglement gate. For a fixed number of layers, the number of parameters to be optimized doubles the one needed for a single-qubit classifier. The depth is also doubled, up to physical realization of the CZ gate.*



(a) Ansatz with no entanglement

(b) Ansatz with entanglement

Figure 3.11: *Four-qubit quantum classifier circuit without and with entanglement. Here, each layer includes a single-qubit operation with data re-uploading per qubit plus two CZ gates if entanglement is considered. The order of CZ gates alternates in each layer between (1)-(2) and (3)-(4) qubits and (2)-(3) and (1)-(4) qubits. The last layer never carries an entanglement gate. For a fixed number of layers, the number of parameters to be optimized quadruples the one needed for a single-qubit classifier. The depth is only doubled, up to physical realization of the CZ gate.*

complex features and correlations of the training data are captured. In addition, the training of such models can be done more efficiently using techniques such as back-propagation [GBC16; Nie15; RHW86].

The natural step to take is to extend the single-qubit formalism to a framework which is more similar to the deep-learning approach of classical methods. The generalization should carry entanglement as a crucial resource for obtaining quantum advantage. The expectation of the model is to improve the overall performance of the quantum classifier as more qubits are added, aiming to mimic the features of hidden layers in deep NNs. Ideally, incorporating entanglement among qubits could lead to a reduction in the number of re-uploadings needed to solve some classification problem. However, even though the relationship between a single-qubit quantum classifier and a single-hidden-layer NN is well understood, see Fig. 3.1 and properly supported by theoretical results, see Th. 3.1.4, the interpretation of multi-qubit classifiers in terms of deep-learning scheme is far from straightforward.

In contradistinction to the single-qubit quantum classifier, the extension to many qubits and addition of entanglement does not have yet theoretical results supporting its applicability and universality. However, it is expected that these circuits have at least the same flexibility as single-qubit circuits. The work in Ref. [SSM21] deals with the representation capabilities of single-depth circuits when distributed along several qubits. Thus, this work does not deal with the possible correlations among subsequent re-uploadings of data on different qubits.

As in multi-layer NNs, the design of multi-qubit quantum classifiers is heuristic. In the classical case there is no recipe to know the optimal number of layers and neurons, and neurons per layer, to solve some problem with the best possible performance. In general, it depends strongly on the problem. In the quantum case, several properties of the circuits must be decided. For instance, how to upload data onto the circuit, namely whether all features are included in all gates and repeated, or distributed along larger parts of the circuit. Another purely quantum extra degree of freedom is the addition of entanglement. Entanglement is in general an open problem when finding Ansätze for parameterized quantum circuits [NY21; SJA19]

Figures 3.10 and 3.11 show the explicit circuits proposed for classification in this work. In these models, layers are a set of parallel single-qubit operations occurring simultaneously. Entanglement is introduced by means of CZ gates between layers of rotations. In the case of circuits with entanglement, those gates are absorbed in the definition of layer. For the two-qubit circuit, the CZ entangling gate is applied after each set of rotations, except for the last one. For the four-qubit classifier, two CZ gates are applied after each rotation set interspersed between qubits (1)-(2) and (3)-(4); and (2)-(3) and (1)-(4) qubits, see Fig. 3.11 for a graphical description. In both images, the gate $U(i, j)$ corresponds to the gates from Eq. (3.65) and (3.66) depending on the dimension of the training data, applied on i-th qubit, j-th layer.

Each rotational layer is composed by the same gates as in the single-qubit case. Thus, the the number of parameters needed to define the circuit is multiplied times the number of qubits present. The depth of the circuit, however, is only doubled with respect to the single-qubit one, up to the physical realization of entangling gates.

Measurement strategy and cost function for a multi-qubit classifier

In the single-qubit classifier, the only available measurement strategy consists on comparing the output quantum state of the circuit with the label-states representing the different classes. This can be done using performing simple tomography procedures or directly measuring relative fidelities. However, when more qubits are considered, this protocol becomes rapidly outdated. First, the number of dimension of the Hilbert space defined by the quantum system increases exponentially, and thus the possibilities of finding maximally orthogonal sets label-states become much more diverse. Second, tomography protocols suffer from exponentially larger costs in terms of number of measurements as the size of the quantum system increases. To overcome these barriers, two different measurement strategies for the multi-qubit classifiers are proposed.

First, the single-qubit measurement is generalized in a natural way. The final output state is compared with one chosen label-state from the computational basis. This, however, becomes unrealizable for a large number of qubits. Due to the exponential increase of the dimensionality of Hilbert spaces, the number of orthogonal states becomes quickly much larger than the number of classes provided by the dataset. With this method, it is only possible to retrieve information from an exponentially small subspace of the Hilbert space. In particular for the first steps of the optimization, where the output state is created following random parameters, this measurement protocol only captures insufficient and random oddments.

The second strategy consists in measuring only one qubit and assign different classes depending on the result. Notice that this qubit cannot be longer described as a pure state $|\psi\rangle$, but rather as a density matrix ρ. This permits to compress the information of large Hilbert spaces into smaller subspaces. This strategy follows similar to the single-qubit one. This approach aims to join ideas of binary multi-qubit classifiers [FN18] and the possibility of multi-class classification by introducing thresholds and single-qubit label states, see Section 3.3.1.

Another piece that should be adapted to accomodate multi-qubit measurements is the definition of the cost function. A different cost function for each measurement strategy is attached. The new cost functions are inspired in the previous χ^2_f, χ^2_{wf} from Eqs. (3.72) and (3.74).

For the first strategy, the fidelity cost function χ^2_f is used. Its generalization to more qubits is straightforward. However, the orthogonal states used for a multi-qubit classifier are taken as the computational basis states. A more sophisticated set of states could be considered to improve the performance of this method.

For the second strategy, the weighted fidelity cost function χ^2_{wf} is used. As stated above, only one qubit is considered, thus

$$\mathcal{F}_{y,q}(\vec{x};\Theta,W) = \langle\phi_y|\,\rho_q(\vec{x};\Theta,W)\,|\phi_y\rangle\,, \tag{3.75}$$

where ρ_q is the reduced density matrix of the qubit to be measured. Then, the weighted fidelity cost function can be adapted as

$$\chi^2_{wf}(\Theta,W,\alpha) = \frac{1}{2}\sum_{\{\vec{x},y\}}\sum_{j=1}^{\mathcal{C}}\left(\sum_{q=1}^{Q}\left(\alpha_{j,q}F_{c,q}(\vec{x};\Theta,W) - (Y_y)_j\right)^2\right), \tag{3.76}$$

where an average is computed over all Q qubits that form the classifier. Notice that the α parameters evolved from being a vector to a $\mathcal{C}\times Q$ matrix. Eventually, the number of measured qubits and effectively the number of optimizable parameters can be reduced.

3.3.3 Numerical benchmark of the quantum classifier

In this section the performance of the quantum classifier previously described is numerically benchmarked against several distinct classification problems. The results here present demonstrate numerically that this method is capable to successfully solve multi-class classification problems for multi-dimensional data. A single-qubit classifier suffices, but multi-qubit circuits reach comparable final results with less processing layers. In summary, the flexibility of the quantum classifier depends mainly on the query complexity of the algorithm, that is, how many times the data is re-uploaded into the circuit. The complete code can be viewed in Ref. [Pér19].

The benchmark is carried by constructing several different classifiers with different numbers of layers. This makes possible to control the query complexity of a given circuit. Then, the models are trained to obtain the optimal parameters $\{\Theta, W\}$ for each layer, plus $\{\alpha\}$ when applicable. The cost functions used to drive the optimization are χ^2_f, χ^2_{wf} from Eqs. (3.72) and (3.74) for the single-qubit classifiers, and its multi-qubit analogues, see Eq. (3.76), for multi-qubit classification.

The datasets to classify for the benchmark are composed by random points in the space $[-1,1]^d$ whose classes are defined by geometrical means. The data points are always the same, while the assignment of

	χ^2_f			χ^2_{wf}				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.50	0.75	–	0.50	0.76	–	0.76	–
2	0.85	0.80	0.73	0.94	0.96	0.96	0.96	0.96
3	0.85	0.81	0.93	0.94	0.97	0.95	0.97	0.96
4	0.90	0.87	0.87	0.94	0.97	0.96	0.97	0.96
5	0.89	0.90	0.93	0.96	0.96	0.96	0.96	0.96
6	0.92	0.92	0.90	0.95	0.96	0.96	0.96	0.96
8	0.93	0.93	0.96	0.97	0.95	0.97	0.95	0.96
10	0.95	0.94	0.96	0.96	0.96	0.96	0.96	0.97

Table 3.1: *Results of the single- and multi-qubit classifiers with data re-uploading for the circle problem. Numbers indicate the success rate, that is correctly guessed samples over total number of samples. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not, respectively. Minimization was in this case done using the L-BFGS-B algorithm. For this problem, both cost functions χ^2_f, χ^2_{wf} lead to high success rates, although the weighted fidelity one achieves it with lesser numbers of layers. The multi-qubit classifier increases this success rate, although entanglement does not produce any appreciable effect.*

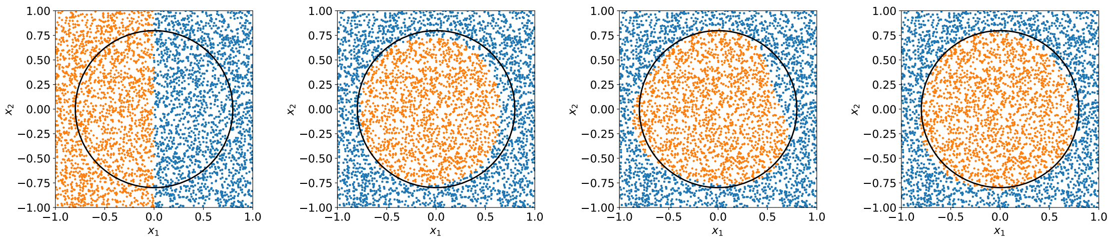
classes differs depending on the problem. After the training is complete, the classifier is tested against an unseen testset one order of magnitude larger than the training set. The reason to proceed in this way is to check the generalization capabilities of the quantum classifier without facing a too costly optimization step.

Classifications with 1, 2 and 4 qubits were carried, with and without entanglement for the multi-qubit cases, for the two cost fuctions defined above. The number of layers tested in every case are $L = \{1, 2, 3, 4, 5, 6, 8, 10\}$. The problems tackled run from simple two-dimensional binary classification to multi-dimensional and multi-class classifications. Non-convex classes, which are considered difficult for classification, are also covered.

Binary classification of a circle

This example is the easiest one considered.The dataset is a random set of data points $\{\vec{x}\} \in [-1,1]^2$, that is $\vec{x} = (x_1, x_2); -1 \leq x_i \leq 1$. The classification task is to determine whether these points satisfy the condition $x_1^2 + x_2^2 < r^2$ for some radius r. From a geometrical perspective, this is equivalent to infer from a point whether it is inside a circle of radius r or not. The value of r was chosen in such a way that the area of the circle is half the total area of the feature space, in this case $r = \sqrt{2/\pi}$. This way, a random classifier obtains an accuracy of 50%. The training dataset is composed of 200 random points, while the testset has 4000 previously unseen points to densely populate the feature space.

The results of the classification for the circle problem are summarized in Tab. 3.1. First, it is worth noticing that the χ^2_{wf} cost function delivers much better results than the simpler χ^2_f, in particular for classifiers with



(a) 1 layer (b) 2 layers (c) 4 layers (d) 8 layers

Figure 3.12: *Results of the circle classification obtained with a single-qubit classifier with different number of layers using the L-BFGS-B minimizer and χ^2_{wf} cost function. One layer returns a random classifier, while the circular shape is already captured with two layers. More layers refine the previous result.*

few layers. With χ^2_{wf}, the single qubit classifier achieves over 90% of success with only two layers, 12 parameters. The two- and four-qubit classifiers reach that threshold with two layers as well, that is 22 and 42 parameters. In addition, the introduction of entanglement does not change the final result in any case. The results show a saturation in the success rate, so that adding more layers leads to no further improvement. The flexibility achieved with few layers is enough to capture all the available data, and the fine tuning required here to improve the results needs of further strategies in the training, such as increasing and densifying the training dataset around the border between classes.

In Fig. 3.12 it is depicted the evolution of the single-qubit classifier for the circle problem as more layers (1, 2, 4, 8) are added. The first layer cannot provide any information since the classification is essentially random. However, adding a second layer is enough for the classifier to broadly capture the general properties of the classifier. When this comprehension is complete, subsequent layers lead to further refinements of the classification.

The characterization of a closed curve is not a trivial problem in classical Machine Learning. Single-layer NNs work in a linear regime, that is, the approximations to any curve must be done by superposition of many linear functions. The quantum classifier is constructed using rotations as building-blocks. Thus, the classification of a circle seems a natural function to classify, in the same sense as a linear dataset is easily understood by a NN. Thus, the quantum classifier must be tested in more complex scenarios to properly benchmark its capabilities.

Multi-class classification: 3 circles

Multi-class classification is now addressed for the first time. Here it is shown that a single-qubit classifier is capable to solve this problem.

	χ_f^2			χ_{wf}^2				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.73	0.56	–	0.75	0.81	–	0.88	–
2	0.79	0.77	0.78	0.76	0.90	0.83	0.90	0.89
3	0.79	0.76	0.75	0.78	0.88	0.89	0.90	0.89
4	0.84	0.80	0.80	0.86	0.84	0.91	0.90	0.90
5	0.87	0.84	0.81	0.88	0.87	0.89	0.88	0.92
6	0.90	0.88	0.86	0.85	0.88	0.89	0.89	0.90
8	0.89	0.85	0.89	0.89	0.91	0.90	0.88	0.91
10	0.91	0.86	0.90	0.92	0.90	0.91	0.87	0.91

Table 3.2: *Success rates of the single- and multi-qubit classifiers with data re-uploading for the 3-circles problem. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not, respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is used. Weighted fidelity cost function presents better results than the fidelity cost function. The multi-qubit classifier reaches 0.90 success rate with a lower number of layers than the single-qubit classifier. The introduction of entanglement slightly increases the success rate respect the non-entangled circuit.*

The same 2D plane as for the circle problem is divided in four different regions of different shapes and sizes. In this case, three classes correspond to three different circular sectors with different centers and radii. The fourth class is the remaining space among circles. This dataset is referred to as the *3 circles* problem. This dataset is non-linear and conceptually difficult to solve for a classical NN.

The summary of results for this multi-class problem is depicted in Tab. 3.2. The single-qubit classifiers surpasses the 90% threshold with 10 layers, 54 parameters. In this problem, the difference between cost functions χ_f^2 and χ_{wf}^2 is smaller than in the circle problem. In addition, the classifier saturates at success rates of $\sim 91\%$. The introduction of several qubits and entanglement makes possible to reach the saturation regime with less parameters, specially for the weighted fidelity function.

In light of these results it is possible to observe that the performance of the classifier does not only depend on the number of parameters, but also on the minimization process and the presence of local minima. Notice that success rates do not always improve with the number of layers and parameters.

As for the previous problem, the evolution of the final results as more layers are taken into account is observed. Fig. 3.13 shows this data for an increasing number of layers. It is worth mentioning that in the very first attempt with one layer the classifier is capable to identify four different regions, that is four different classes, but the boundaries among them are only vaguely learnt. A significant change is observed from 3 to 4 layers. At this stage, the geometrical figures of the datasets are captured by the classifier. With 10 layers, further refinements are accomplished.

The 3-circles problem is an illustrative example to see how the output

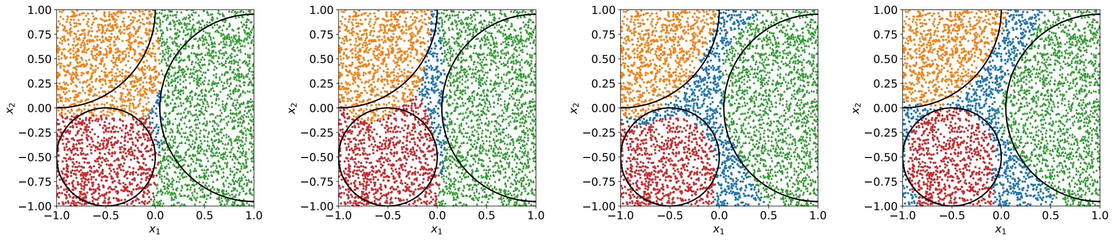
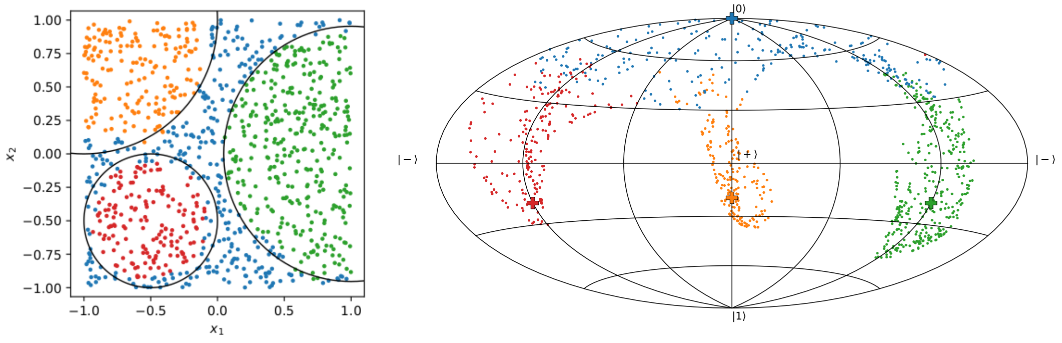

(a) 1 layer (b) 3 layers (c) 4 layers (d) 10 layers

Figure 3.13: *Results of the 3 circles classification obtained with a single-qubit classifier with different number of layers using the L-BFGS-B minimizer and χ^2_{wf} cost function. With one layer, the classifier intuits the four regions although the central one is difficult to tackle. With more layers, this region is clearer for the classifier and the circular regions are adjusted.*



(a) Guessed points (b) Output states on the Bloch sphere

Figure 3.14: *Left: Results for a classification problem. The color corresponds to the classification guess. Bottom: output states of all data points projected on the Bloch sphere, where some states are printed for reference. Color corresponds to the actual class, and not to the classifier guess. Crosses stand for the label states of the different classes. Points belonging to the same class tend to gather around its label state.*

state behaves for classifying data. In Fig. 3.14 we can see a classification of data, as extracted from the documentation `Qibo` [Eft+20b]. The first plot, Fig. 3.14(a), shows the guesses of the classifier (left) and what points are correctly guesses by the classifier (left), as in other examples here provided. In the second plot, Fig. 3.14(b), the output states of all different points are projected in a Bloch sphere as in a world map. The states $|0\rangle, |1\rangle, |+\rangle, |-\rangle$ are printed for reference. The colors correspond to the actual classes, and not to the guesses of the classification, while crosses stand for the label states of each class. It is straightforward to see that all points corresponding to the same class gather around the label

	χ^2_f			χ^2_{wf}				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.87	0.87	–	0.87	0.87	–	0.90	–
2	0.87	0.87	0.87	0.87	0.92	0.91	0.90	0.98
3	0.87	0.87	0.87	0.89	0.89	0.97	–	–
4	0.89	0.87	0.87	0.90	0.93	0.97	–	–
5	0.89	0.87	0.87	0.90	0.93	0.98	–	–
6	0.90	0.87	0.87	0.95	0.93	0.97	–	–
8	0.91	0.87	0.87	0.97	0.94	0.97	–	–
10	0.90	0.87	0.87	0.96	0.96	0.97	–	–

Table 3.3: *Success rates of the single- and multi-qubit classifiers with data re-uploading for the four-dimensional hypersphere problem. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not, respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is used. The fidelity cost function is likely to get stuck in local minima for the multi-qubit classifiers. χ^2_{wf} results are much better, peaking at 0.98 success rates with only two layers in the entangled four-qubit classifier. Unlike in other examples, the presence of entanglement significantly improves the performance.*

states.

Classification of high-dimensional datasets: hypersphere

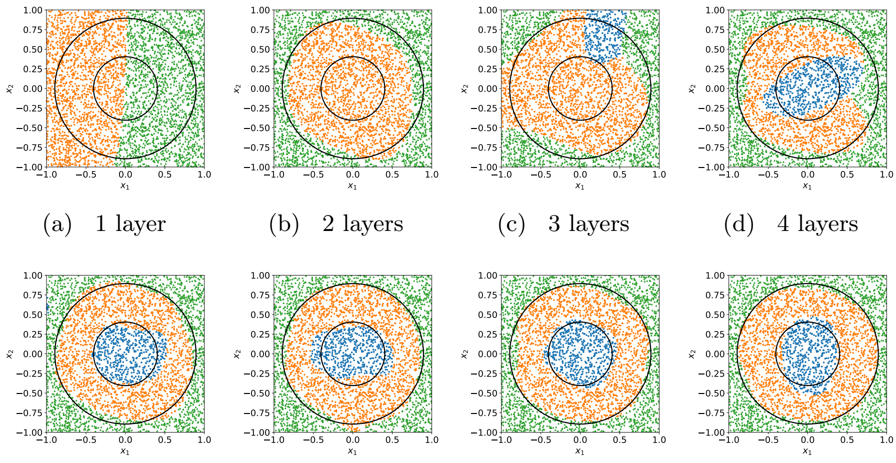
The quantum classifier does not have any restriction in the dimension of the re-uploaded data. As mentioned in Sec. 3.3.1, every gate can accomodate up to three features of data. In case the dimension is larger, then the re-uploading steps can be split into different steps, see Eq (3.66). Larger dimensionality can be managed by using more gates.

With this idea more complicated classifications can be addressed. In particular, the classification of a 4D hypersphere is used as testbed. This problem is just an extension of the circle one, where the radius of the hypersphere changed to fill half the volume of the feature space. This time, the training set is composed of 1000 random points.

Results are summarized in Tab. 3.3. A single-qubit classifier reaches its maximum success rate 97% for 8 layers, 82 parameters, with the χ^2_{wf} cost function. Two-qubit classifiers peak at 5 layers (62 parameters) for the entangled case, and four-qubits classifiers have their maximum with 2 layers (82 parameters). Note that, in this case, entanglement provides better final results as compare to other problems. Four-qubit classifiers with more layers are not considered due to their training computational cost.

Classification of non-convex datasets: tricrown

Non-convex patterns are usually difficult to classify for in Supervised Learning frameworks. In this example two concentric circles with different radii defining three different classes of similar sizes are studied. Thus, this is a non-convex and multi-class problem.



(a) 1 layer (b) 2 layers (c) 3 layers (d) 4 layers

(e) 5 layers (f) 6 layers (g) 8 layers (h) 10 layers

Figure 3.15: *Results obtained with the single-qubit classifier for the tricrown problem, using the weighted fidelity cost function during the training. Notice that 2 layers can capture 2 classes reasonably well, while the inner one is forgotten. 4 layers locate all classes in their corresponding places, and 5 layers learn the general shape of the datset. Then, further layers add refinement to the classification.*

	χ^2_f			χ^2_{wf}				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.34	0.51	–	0.43	0.77	–	0.81	–
2	0.57	0.63	0.59	0.76	0.79	0.82	0.87	0.96
3	0.80	0.68	0.65	0.68	0.94	0.95	0.92	0.94
4	0.84	0.78	0.89	0.79	0.93	0.96	0.93	0.96
5	0.92	0.86	0.82	0.88	0.96	0.96	0.96	0.95
6	0.93	0.91	0.93	0.91	0.93	0.96	0.97	0.96
8	0.90	0.89	0.90	0.92	0.94	0.95	0.95	0.94
10	0.90	0.91	0.92	0.93	0.95	0.96	0.95	0.95

Table 3.4: *Success rates of the single- and multi-qubit classifiers for the tricrown problem. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not, respectively. The χ^2_{wf} cost function presents better success rates than χ^2_f. The multi-qubit classifiers improve the results obtained with the single-qubit classifier but the using of entanglement does not introduce significant changes.*

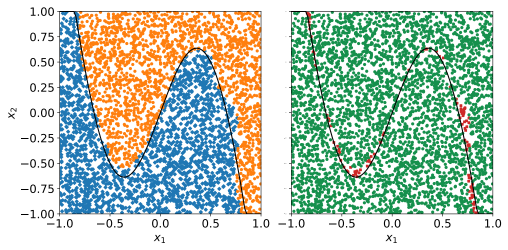
Results for this tricrown problem are summarized in Tab. 3.4. A success rate of 93% is achieved with 10 layers for the single-qubit classifier and cost function χ^2_{wf}. Two-qubit classifiers peak at 94% (3 layers), while four-qubit obtain 96% (2 entangled layers).

The results of different layers for single-qubit classifiers with different numbers of layers show the evolution in the performance of the classification, see Fig. 3.15. 4 layers are needed to learn the concentric patterns, and the fifth one establishes the topology of the different regions. From this point, more layers just refine the final results.

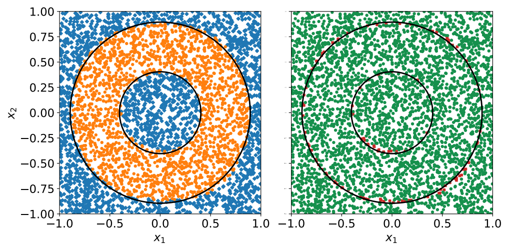
Other datasets: non-convex, sphere, squares, wavy lines

The single- and multi-qubit classifiers are tested in more datasets to complete the benchmark. These extra datasets cover different kinds of training data and aim to show that the quantum classifier can adapt itself to large varieties of problems. From a qualitative point of view, the results here presented are just an extension of the ones seen before. Thus, the reader can skip these lines to Sec. 3.3.3 without any regrets. All tables and figures summarizing data are depicted at the end of the section. The datasets are similar to all previous datasets. Data points $\vec{x} \in [-1, 1]^d$, and the sizes are 200 / 4000 for the training / test sets. The only exception is the *sphere* problem, where the sizes are 500 / 4000. The design of the problems was made in such a way that a random classifier guesses 1 / (# classes) correctly.

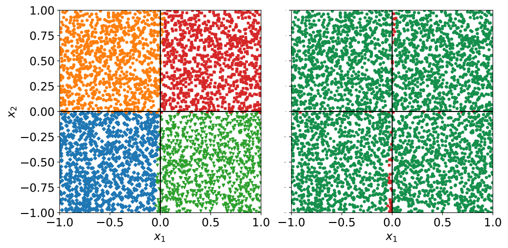
The results described in the following reinforce the properties previously



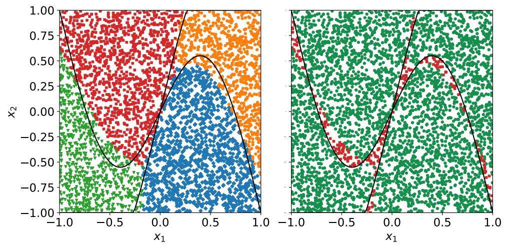
(a) Non-convex: χ^2_{wf}, 1 qubit, 6 layers



(b) Crown: χ^2_{wf}, 2 qubits without entanglement, 4 layers



(c) Squares: χ^2_{f}, 2 qubits without entanglement, 6 layers



(d) Wavy lines: χ^2_{wf}, 2 qubits with entanglement, 6 layers

Figure 3.16: *Best results for 2D problems analyzed in this section. The problem and architecture is detailed in each caption. For similar results, the simpler architecture was chosen. Colors in the left part of each figure represent the different classes obtained from the classifier outputs, while the right images show correctly (green) and wrongly (red) classified points. Black solid lines define the problem boundaries.*

	χ^2_f			χ^2_{wf}				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.49	0.55	–	0.49	0.76	–	0.76	–
2	0.82	0.75	0.75	0.86	0.94	0.85	0.96	0.96
3	0.93	0.74	0.85	0.96	0.95	0.95	0.95	0.97
4	0.93	0.74	0.88	0.95	0.96	0.97	0.95	0.96
5	0.91	0.95	0.90	0.97	0.95	0.96	0.95	0.97
6	0.96	0.94	0.93	0.98	0.97	0.97	0.95	0.97
8	0.96	0.96	0.95	0.98	0.98	0.97	0.96	0.97
10	0.95	0.92	0.95	0.96	0.96	0.96	0.96	0.97

Table 3.5: *Results of the single- and multi-qubit classifiers with data re-uploading for the non-convex problem. Numbers indicate the success rate, i.e. the number of data points classified correctly over the total number of points. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is use. Both cost functions lead to higher success rates, although the weighted fidelity cost function is better. It achieves the 0.98 success with two qubits, entanglement, and four layers.*

observed. The classifier starts in a first stage where the performance improves as more layers are added. Then, a stationary stage is reached. Larger number of qubits and entanglement advance the appearance of this stationary regime.

In the following, each different problem carries its own results table. The best instance for each problem in a 2D feature space is plotted in Fig. 3.16. Notice in this figure that all missed points are located near the borders of the problem. This means that the classifier is properly understanding the properties of the dataset, but more training is needed to fine-tune the bordering regions.

Non-convex problem

Classification problems where classes are mutually non-convex are considered difficult since the separation between both classes is hard to characterize. In this problem, both zones are separated through the line $x_2 = -2x_1 + 3/2\sin(\pi x_1)$. With this boundary, there is no area so small that the classifier can achieve good results even if this area is neglected.

Tab. 3.5 summarizes the results for this problem. Best performance (98%) is achieved with a single-qubit classifier of 6 layers (32 parameters) using the χ^2_{wf} cost function. The fidelity cost function χ^2_f gets 96% in the same conditions, but peaks at 97% for 2 entangled qubits with 8 layers (80 parameters). See this example in Fig. 3.16(a).

Crown

This one is a binary version of the tricrown dataset. The most interesting feature of this dataset is that one class is composed of two different disconnected regions. Thus, the classifier must find a way to

	χ_f^2			χ_{wf}^2				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.44	0.50	–	0.44	0.59	–	0.66	–
2	0.48	0.50	0.51	0.53	0.73	0.72	0.70	0.96
3	0.91	0.50	0.56	0.74	0.75	0.95	0.78	0.96
4	0.80	0.74	0.56	0.86	0.97	0.97	0.92	0.96
5	0.90	0.93	0.88	0.89	0.97	0.96	0.97	0.94
6	0.92	0.91	0.94	0.95	0.94	0.95	0.95	0.93
8	0.90	0.93	0.95	0.92	0.94	0.94	0.96	0.94
10	0.90	0.92	0.91	0.92	0.95	0.93	0.96	0.93

Table 3.6: *Results of the single- and multi-qubit classifiers with data re-uploading for the crown problem. Numbers indicate the success rate, i.e. the number of data points classified correctly over the total number of points. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is use. As happens in other problems, the results obtained with the weighted fidelity cost function are better than the ones obtained with the fidelity cost function. The multi-qubit classifiers and the introduction of entanglement increase the success rates.*

	χ_f^2			χ_{wf}^2				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.53	0.70	–	0.53	0.70	–	0.70	–
2	0.77	0.73	0.53	0.78	0.94	0.96	0.96	0.96
3	0.76	0.74	0.77	0.78	0.92	0.94	0.94	0.95
4	0.84	0.83	0.78	0.89	0.92	0.94	0.95	0.94
5	0.89	0.85	0.77	0.90	0.94	0.94	0.95	0.94
6	0.90	0.89	0.88	0.92	0.87	0.93	0.94	0.94
8	0.89	0.87	0.90	0.93	0.92	0.89	0.94	0.93
10	0.93	0.91	0.90	0.93	0.94	0.92	0.92	0.92

Table 3.7: *Results of the single- and multi-qubit classifiers with data re-uploading for the three-dimensional sphere problem. Numbers indicate the success rate, i.e. the number of data points classified correctly over the total number of points. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is use. The weighted fidelity cost function is better than the fidelity cost function. There are no significant differences between the two-qubit and the four-qubit classifiers. Both are better than the single-qubit classifier and the introduction of entanglement does not increase the success rates.*

understand disjoint regions as belonging to the same class.

The χ_f^2 function reaches its best result (94%) for 2 entangled qubits and 6 layers (60 parameters). For χ_{wf}^2, a 97% accuracy is obtained for 2 unentangled qubits, 4 layers (40 parameters). See Tab. 3.6 for a summary, and Fig. 3.16(b) for an example.

Sphere

This quantum classifier is able to classify multidimensional data, as shown with the four-dimensional hypersphere. A three-dimensional figure is also tested, a regular sphere of size half the feature space.

For this problem, the fidelity cost function reaches its maximum, 93%, with a single-qubit classifier of 10 layers (60 parameters). The same success is obtained with a two-qubit entangled classifier and 6 layers (72 parameters). With the weighted fidelity, this success rate grows up to 96% for two- and four- qubit classifier of 2 layers (24 and 48 parameters respectively) with and without entanglement. All results are written in Table 3.7.

Squares

This problem divides a 2D area into four quadrants with straight lines. This is one of the easiest problems for a NN. By construction, NNs can establish a separation between classes by using biases, and thus straight lines are immediate to understand. This problem aims to see how a quantum classifier performs against a NN in the latter's field.

The fidelity cost function reaches 99% of success in a two-qubit classifier without entanglement and 6 layers (60 parameters). Any two-qubit result is comparable with the success rate of the classical models. Something similar can be found for the weighted fidelity. The maximum success, 96%, is obtained with a two-qubit entangled classifier with 4 layers (40 parameters). The results are written in Table 3.8 and the best performance is plotted in Figure 3.16(c).

	χ^2_f			χ^2_{wf}				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.58	0.48	–	0.70	0.92	–	0.90	–
2	0.76	0.96	0.97	0.74	0.91	0.94	0.95	0.95
3	0.90	0.96	0.98	0.90	0.94	0.95	0.95	0.95
4	0.89	0.98	0.96	0.88	0.94	0.95	0.95	0.95
5	0.91	0.97	0.98	0.89	0.94	0.94	0.95	0.94
6	0.92	0.99	0.94	0.93	0.94	0.94	0.94	0.94
8	0.93	0.98	0.94	0.93	0.94	0.95	0.95	0.94
10	0.94	0.97	0.93	0.94	0.94	0.94	0.94	0.93

Table 3.8: *Results of the single- and multi-qubit classifiers with re-uploading data for the squares problem. Numbers indicate the success rate. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not respectively. The L-BFGS-B minimization method is used. In this problem, the fidelity cost function χ^2_f presents better results. It achieves the 0.99 success with the two-qubit classifier with six layers and no entanglement.*

Wavy lines

This problem is the four-class version of the non-convex problem. Now the area is divided into four regions by two different functions. The borders' equations are $x_2 = \sin(\pi x_1) \pm x_1$. The important feature of this problem is that there are some areas in the problem too small to be caught by the classifier.

As can be seen in Figure 3.16(d), most of the failure points are in these small non-convex areas. The classifier would rather adjust the rest of the points instead of tuning those zones and losing everything else. The results for this problem are not as good as for other problems, but still 94% for the fidelity cost function is obtained, two entangled qubits and 10 layers (200 parameters) and the weighted fidelity, four entangled qubits and 4 layers (80 parameters).

	χ_f^2			χ_{wf}^2				
Qubits	1	2		1	2		4	
Layers		No Ent.	Ent.		No Ent.	Ent.	No Ent.	Ent.
1	0.70	0.52	–	0.76	0.75	–	0.88	–
2	0.86	0.75	0.80	0.84	0.89	0.88	0.91	0.92
3	0.74	0.82	0.84	0.84	0.92	0.91	0.92	0.92
4	0.80	0.85	0.87	0.87	0.89	0.93	0.92	0.93
5	0.85	0.90	0.88	0.87	0.92	0.92	0.93	0.93
6	0.92	0.92	0.91	0.88	0.93	0.94	0.93	0.93
8	0.90	0.91	0.91	0.92	0.92	0.92	0.93	0.94
10	0.92	0.91	0.93	0.90	0.93	0.93	0.93	0.93

Table 3.9: *Results of the single- and multi-qubit classifiers with re-uploading data for the wavy lines problem. Numbers indicate the success rate. Words "Ent." and "No Ent." refer to considering entanglement between qubits or not respectively. The L-BFGS-B minimization method with the weighted fidelity and fidelity cost functions is used. Results with the weighted fidelity cost function and multi-qubit classifiers are vaguely better than other configurations. Entanglement does not change significantly the results.*

Comparison to classical classifiers

The field of Machine Learning has a great development for classical computers. Thus, the performance of the quantum classifiers against classical methods can be compared to check if this proposal can in some sense compete against well established methods.

To do so, the standard ML library `scikit-learn` [Ped+11] is used to solve the same examples as in the quantum classifier. The classical models here used are simple ones in order to get fair comparisons, that is, quantum and classical models with similar complexities. The aim of this benchmark is not to review the capabilities of classical Machine Learning, which are known to be extensive, but rather to settle whether quantum or classical models work better for similar circumstances.

The results were obtained using two different methods. First, single-hidden-layers NNs of 20 neurons with all activation functions available in `scikit-learn` and a *lbfgs* solver. The function used to this extent is `sklearn.neural_network. MLPClassifier`. For a SVC, `sklearn .svm.SVC` with different kernels was used. Note that NNs have a controllable

Problem (# classes)		Classical classifiers		Quantum classifier	
		NN	SVC	χ^2_f	χ^2_{wf}
Circle	(2)	0.96	0.96	0.96	0.97
Crown	(2)	0.96	0.82	0.95	0.97
Non-Convex	(2)	0.98	0.97	0.96	0.98
Sphere	(2)	0.93	0.91	0.93	0.96
Hypersphere	(2)	0.89	0.92	0.91	0.98
Tricrown	(3)	0.96	0.83	0.93	0.97
3 circles	(4)	0.94	0.92	0.91	0.91
Squares	(4)	0.99	0.95	0.99	0.95
Wavy Lines	(4)	0.98	0.89	0.93	0.94

Table 3.10: *Comparison between single-qubit quantum classifier and two well-known classical classification techniques: NNs with a single hidden layer composed of 100 neurons and Support Vector Classifiers (SVC), both with the default parameters as defined in* `scikit-learn` *python package [Ped+11]. Results of the single-qubit quantum classifier are obtained with the fidelity and weighted fidelity cost functions, χ^2_f and χ^2_{wf} defined in Eq. (3.72) and Eq. (3.74) respectively. This table shows the best success rate, being 1 the perfect classification, obtained after running ten times the NN and SVC algorithms and the best results obtained with single-qubit classifiers up to 10 layers.*

number of tunable parameters that can be adjusted to match the number of the quantum classifier. For SVCs this is not possible and the complexity of the algorithm depends on the size of the training set.

A summary of results can be seen in Tab. 3.10. In classical models, only the best final result is retrieved for each problem and NN or SVC. For quantum results, single-qubit results with different cost functions are depicted. In all problems it is possible to see that quantum and classical performance are at least comparable to classical methods.

It is important to mention that even though final results of quantum and classical methods perform in a similar way, the effort required to achieve these results is very different. Quantum algorithms are not as efficient as classical ones.

3.3.4 Discussion

The quantum classifier here proposed and based on the general re-uploading strategy has shown capabilities to successfully accomplish non-trivial supervised learning tasks, in an equivalent sense as simple NNs can. The classification problems here addressed consists in learning complex geometrical figures in multidimensional spaces.

The key ingredient for the quantum classifier, apart from the re-uploading scheme, is the measurement strategy. A set of label states is created so that each one corresponds to a different class. The quantum circuit is forced to deliver output states in average as close as possible to the corresponding label state, depending on the class of the training data. The identification of the class is then defined as the smallest distance between the output and the different label states. These target states are chosen in a maximally orthogonal way as a strategy to make them as distinguishable as possible. As a consequence, each label corresponds to a region in the Bloch sphere.

The optimization process is driven by two different cost functions. The simplest one, fidelity cost function, simply measures the fidelity between the output state and its corresponding label. A more sohpsticated one, weighted fidelity cost function, is inspired in NNs. It measures the distance to all classes and compare it to the ideal case.

The single-qubit classifier is extendable to multi-qubit architectures. This allows for the introduction of entanglement between the qubits. An entangling Ansatz is shown as a proof of concept, while exhaustive exploration of Ansätze is out of the scope for the moment.

We benchmarked several quantum classifiers with different numbers of layers, qubits and entanglement mappings against classical classification methods. The test problems are data points embedded in 2D, 3D and 4D feature spaces, where each class is defined by means of geometrical figures. In all cases, the single-qubit classifier provides success rates of over 90%. More qubits and entanglement can increase this success while reducing the effective number of layers. However, the number of calls to classical data remains approximately constant for different classifier with the same performance. In terms of cost functions, the weighted fidelity one helps to find better results, at the cost of increasing the number of trainable parameters. In addition, as more layers are considered, the probability to get stuck in local minima increases as well, as expected from an optimization problem involving large numbers of parameters.

After publishing the original work on the quantum classifier [Pér+20a], several publications have been developed based on this method. Reference [Eas+21] carried an exhaustive study and nice representations of the re-uploading classifier. Reference [HD21b] develops an exploration on the landscape of loss functions generated by the quantum classifier. This model is also part of the documentation of some quantum computing packages [Ahm19; Qib20].

3.4 Experimental quantum classifier

A surge of algorithms for QML that theoretically or even numerically work has appeared during recent years. However, examples of successful implementations of some QML recipe on an actual quantum processor are much scarcer. Quantum devices at the state of the art suffer the effects of noise and decoherence, thus the performance of their calculations is limited. In addition, different experimental platforms have different properties that make them more or less suitable for a particular algorithm, and it is then difficult to gain insights on the optimal experimental configuration to accomplish a given task.

A technology that suits the data re-uploading strategy algorithm is the ion trap. The framework here described makes use of quantum systems that are sparse in qubits, but those qubits must be faithfully controlled to successfully accomplish the QML task faced. Ion trap devices are capable to control small systems very accurately, even though the scalability of the machines is not as convenient as in other platforms like photonics or superconductors [Bro+11; RK19; Wan+21a]. In recent years, ion-trap experiments have shown its worth and widened the range of feasible experiments [FMM+17; Hem+18; JDK+20; NCK+20; RTP+20]. Those results support the choice of ion traps for the implementation here presented as well.

In this chapter, it is shown that the data re-uploading strategy can be implemented on an ion-trap quantum processing unit (QPU). The experiment here performed consists in applying a simplified version of the quantum classifier, that is, a simplified circuit to solve the same problems. This simplified version is close to the formulation described in Th. 3.1.4, since it provides the best results with the smallest number of parameters and depth of the circuit. The experimental approach here presented constitutes, to the best of our knowledge, the first successful implementation of a quantum classifier in a system that is very sparse in qubits.

The main difference between running quantum experiments on classical simulators or actual quantum devices is that simulated methods do not always capture the complexity and features of the experiment faithfully. Elements as native gates or noise environment are completely device-dependent. In this work, the qubit is controlled in an optimal way, so that the inherent properties of the system are utilized to improve the overall performance of the classifier.

In this section, the translation from simulation to experiment is treated in Sec. 3.4.1. Results are reviewed in Sec. 3.4.2. Final remarks are collected in Sec. 3.4.3.The experimental setup is detailed in App. A.5.

3.4.1 Single-qubit quantum classifier on the experiment

In order to implement the quantum classifier on the ion-trap quantum device, some shallow modifications must be taken into account.

Quantum circuits

The circuits used to perform the classification tasks on the experiment are a simplified version of the simulated ones. The reason to simplify the algorithm is to obtain a quantum circuit with equivalent query complexity as in the original formulation, but less parameters and rotations. This is an attempt to reduce the overall number of operations required to carry the task. The circuit is again defined as a series of gates, as in Eq. (3.67), but the building blocks, corresponding to Eqs. (3.65) changes. Two different single-qubit operations are proposed.

The Ansatz A is directly inherited from Th. 3.1.4, and it is defined as

$$U_A(\vec{x}, \vec{\theta}, \vec{w}) = R_z(\theta_z) R_y(\vec{w} \cdot \vec{x} + \theta_y), \tag{3.77}$$

where $\vec{x}$ is the data point and $\theta_z, \theta_y, \vec{w}$ are tunable parameters. Note that this operation unlike the one in Eq. (3.65) compresses all the x-dependency in only one argument of the rotation, namely the angle around the y-axis. On the other hand, the z-rotation is only included to generate non-linearities. It is also worth mentioning that this Ansatz permits the data $\vec{x}$ to have any number of features. All data can be easily accomodated by increasing the dimensionality of $\vec{w}$.

Ansatz B is more similar to the definition in Eq. (3.65). For simplicity, only 2D problems are addressed with this Ansatz. The gate is defined as

$$U_B(\vec{x}, \vec{w}, \vec{\theta}) R_z(w_2 x_2 + \theta_2) R_y(w_1 x_1 + \theta_1). \tag{3.78}$$

In this case, the non-linearities are expected to arise faster than for Ansatz A since both features of $\vec{x}$ interact with each other.

Optimization technique

The working procedure used for the experimental quantum classifier follows the scheme depicted in Fig. 3.9 for the simulated one. The retrieving stage is similar, up to the technical details required to use an actual quantum machine, to be detailed later. However, the optimization stage itself is performed in two steps, a simulated and a experimental one, see Fig. 3.17 for a schematic description of the optimization procedure.

First, the training set is considered for optimization on a simulated framework, as in the original proposal. This stage returns a set of optimal parameters in the simulated scenario $\Theta^{\rm sim}, W^{\rm sim}$, attached to a given value of the cost function $\chi^2_f(\Theta^{\rm sim}, W^{\rm sim})$ and a theoretical accuracy $\mathcal{A}(\Theta^{\rm sim}, W^{\rm sim})$ computed on the test set. In this experiment only the fidelity cost function χ^2_f from Eq. (3.72) is considered.

The second step is then performed on the experiment. The parameters $\Theta^{\rm sim}, W^{\rm sim}$ provide a given accuracy when the quantum circuit is run on the experimental setup. It is likely that the imperfections of the experimental device limit the capabilities of the algorithm. In order to mitigate this effect, a new optimization step is performed on the quantum machine taking as starting point the previously obtained $\Theta^{\rm sim}, W^{\rm sim}$. This will ideally lead to some new parameters $\Theta^q = \Theta^{\rm sim} + \delta\Theta$ and equivalently for W. The new set of parameters is ideally capable to

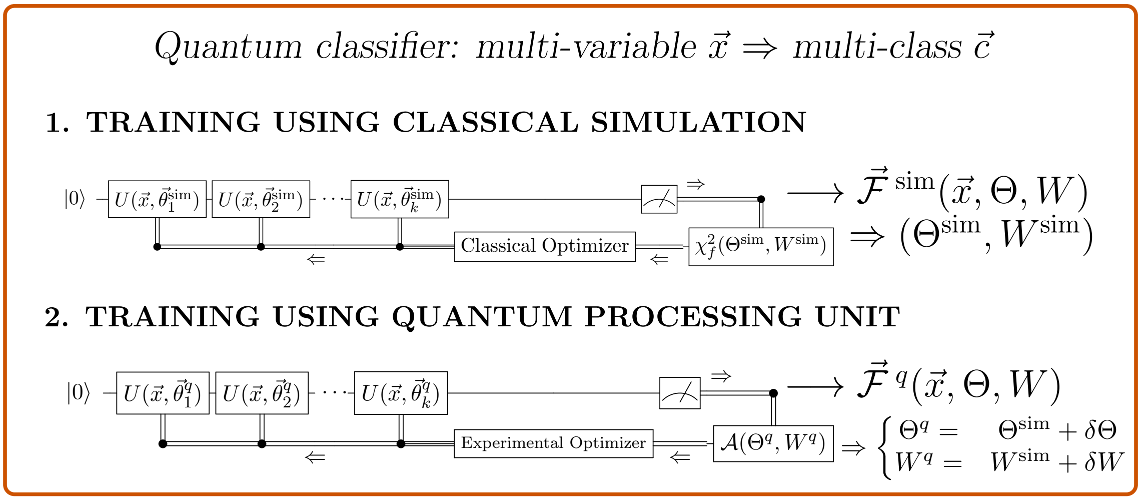

Quantum classifier: multi-variable $\vec{x} \Rightarrow$ multi-class $\vec{c}$
1. TRAINING USING CLASSICAL SIMULATION
$|0\rangle$
$U(\vec{x}, \vec{\theta}_1^{\text{sim}})$
$U(\vec{x}, \vec{\theta}_2^{\text{sim}})$
$U(\vec{x}, \vec{\theta}_k^{\text{sim}})$
Classical Optimizer
$\chi_f^2(\Theta^{\text{sim}}, W^{\text{sim}})$
$\longrightarrow \vec{\mathcal{F}}^{\text{sim}}(\vec{x}, \Theta, W)$
$\Rightarrow (\Theta^{\text{sim}}, W^{\text{sim}})$
2. TRAINING USING QUANTUM PROCESSING UNIT
$|0\rangle$
$U(\vec{x}, \vec{\theta}_1^{q})$
$U(\vec{x}, \vec{\theta}_2^{q})$
$U(\vec{x}, \vec{\theta}_k^{q})$
Experimental Optimizer
$\mathcal{A}(\Theta^q, W^q)$
$\longrightarrow \vec{\mathcal{F}}^{q}(\vec{x}, \Theta, W)$
$\Rightarrow \begin{cases} \Theta^q = \Theta^{\text{sim}} + \delta\Theta \\ W^q = W^{\text{sim}} + \delta W \end{cases}$

Figure 3.17: *Schematic description of the training algorithm used in this work. In a first step, data re-uploading is trained using a classical simulation. The simulated quantum circuit is composed of single-qubit gates U depending on variational parameters (Θ, W) and the data points $\vec{x}$. The output state is measured to obtain the fidelity between the output state and the corresponding label state. This quantity encodes the probabilities that will serve to classify the given pattern into a category. A classical optimization is performed to obtain optimal values $\Theta^{\text{sim}}, W^{\text{sim}}$. This optimization is driven by the cost function χ_f^2 evaluated on training data. In a second step, a further optimization is accomplished only using the quantum device, taking as starting point the values $\Theta^{\text{sim}}, W^{\text{sim}}$ and delivering a set Θ^q, W^q suiting better the experiment. The quantity to minimize is the accuracy $\mathcal{A}$ evaluated on the test dataset. The aim of the experimental optimization is to mitigate and even compensate possible systematic experimental errors.*

suit better the requirements and mitigate any systematic error of the experimental setup, and thus to obtain improved accuracies.

The experimental optimization is carried by scanning the parameter space in the vicinity of $\Theta^{\text{sim}}, W^{\text{sim}}$ and retrieving the configuration Θ^q, W^q with best accuracy $\mathcal{A}^q$. This second step is available for quantum devices only if the loss function in the parameter space near the vicinity of the minimum is smooth and large deviations from the optimal parameters translate into small changes in the loss function. Unfortunately, this full scan is extremely time expensive for current machines, and then it is only possible to provide results optimized in these two steps in a small number of examples.

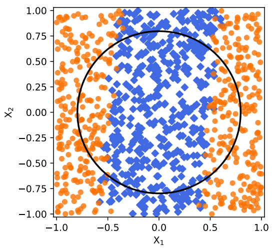
3.4.2 Results

Results from the experimenta setup were obtained following the same problems as in the simulated classifier, and all descriptions hold in this approach. To reduce the computational cost of the experimental implementation, the training dataset is set to 250 points, and the test dataset to 1000.

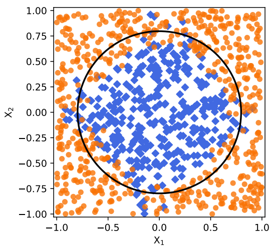
We describe the results for the circle problem in a detailed way to compare the performance of the theoretical model and its experimental counterpart. Fig. 3.18 shows experimental results for this classification problem for an increasing number of layers, and a comparison between the test data as classified by the QPU and an ideal classical qubit simulator. It is possible to see the improvement in the classification of results as more layers are added, up to 4 layers. In that case, the classification accuracy for the QPU is $\mathcal{A}^q = 93 \pm 2\%$, slightly lower than its classically simulated counterpart $\mathcal{A}^{\rm sim} = 97\%$. The error here refers to the standard deviation of 10 repeated trials performed on the same dataset and it reflects the underlying systematic uncertainty leading to an uncertainty of the accuracy. This confirms experimentally the expected behavior of a re-uploading scheme, namely the classifier is more accurate as more layer are added to the quantum circuit.

It is worth noting a difference between simulated and experimental results. In Fig. 3.18(e), the guessed boundary between classes is sharply defined, even though it does not match exactly the theoretical boundary and the classification is slightly deformed. That is a consequence of simulation, the output state is described with arbitrary accuracy and thus the border between two zones in the Bloch sphere is perfectly defined. The results on the experimental data, 3.18(e), show uncertainty in the determination of classes. For instance, in the higher part of the circuit, there is a cluster of points where different classes are interspersed. The origin of this phenomenon is the sampling uncertainty. All points near the boundary can be wrongly measured, leading to misclassification. In addition to the uncertainty region, some outliers are misclassified, see blue points at the border of the feature space and an orange one in the center. In the simulation scenario, the reason for misclassification is a defective training or model.

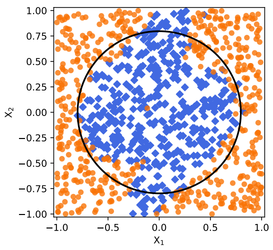
Results for the experimental optimization step are also provided in this example. Figs. 3.19(a) show the landscape of the accuracy for a specific subset of parameters in the vicinity of the optimum point as provided by the classical simulation. The smoothness of the landscape around the starting point $(\Theta^{\rm sim}, W^{\rm im})$ induces that only three parameters are actually contributing to significant changes in the cost function, and thus the computational cost of scanning for the optimal configuration is manageable. A deviation between the theoretical and experimental optimal parameters is appreciable. In addition to the landscape, the error in the classification is depicted in Fig. 3.19(a), where the experimental errors are estimated separatedly, see App. A.5. After corrections, accuracies for the simulated and experimental setups are similar. The experimental



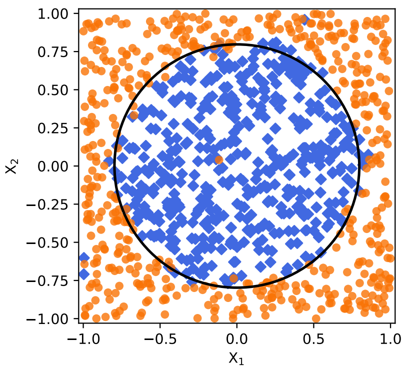
(a) QPU, 1 layer



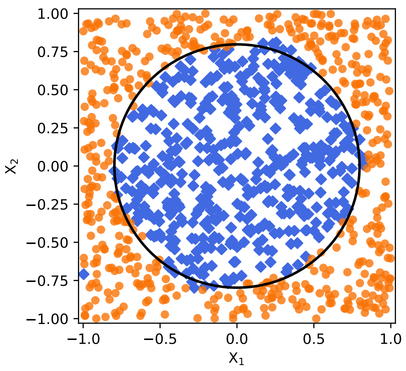
(b) QPU, 2 layers



(c) QPU, 3 layers



(d) QPU, 4 layers; $\mathcal{A}_q = 93 \pm 2\%$

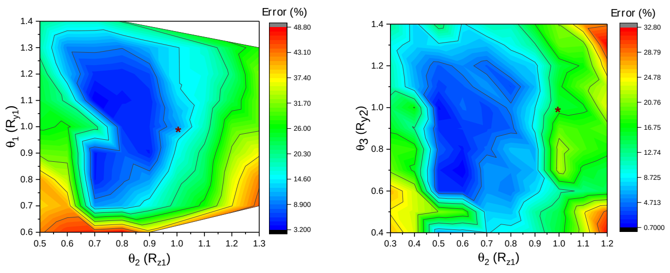


(e) Cl. simulation, 4 layers; $\mathcal{A}_{\text{sim}} = 97\%$

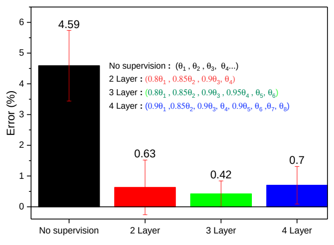
Figure 3.18: *Classifier test results. The ion trap based QPU classifier performed on* 1000 *random data points tests depicted in blue for points within and orange outside the boundary separating the circular feature shown in solid line. The depth of the circuit is increased by one, starting from 1-layer in (a) to 4 in (d). The result of the 4-layer QPU classifier (d) is compared with the same 4-layer simulator (e). Notice that the border between classes in the experimental results is not as sharply defined as in the simulated classification. This difference is due to the uncertainty of the quantum measurements and systematic errors.*

optimization brings an improvement of nearly 5% with respect to the initial parameters $(\Theta^{\text{sim}}, W^{\text{sim}})$.

The next result presented corresponds to the 4D-hypersphere problem, depicted in Fig. 3.20. In this figure, the number of points guessed as inside and outside the hypersphere are printed in different colors and represented in a histogram. A black line corresponding to the boundary is also depicted. The x axis corresponds to the radius of the data point. The overlap region around the boundary corresponds to the failure rate, and thus to that area where the classification is ambiguous. This

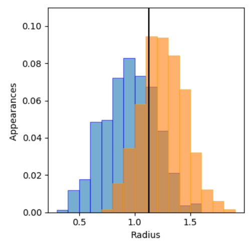


(a) Landscape of $\mathcal{A}_q$ around $\boldsymbol{\theta}_{\text{sim}}(\star)$

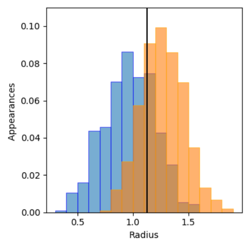


(b) Error in classification

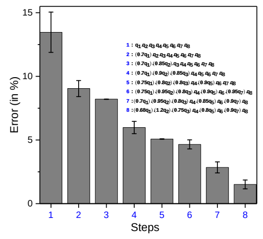
Figure 3.19: *Experimental optimization: The ion trap based QPU classifier used for binary classification of data within or outside a bounded circle is trained by varying the training parameters (θ_1, θ_2, θ_3) in the vicinity of the parameters obtained from the simulated training. The error surface plotted in color-coded surface plot in (a) shows the deviation of the optimal parameters from the trained minimum ($\star$). These plots are for 2-layer QPU and corresponds to Fig. 3.18(b). Similar training performed on the QPU leads to the betterment of the accuracy in the 3 and 4 layer QPU as shown in (b). Note that the improvement is more pronounced in the first two layers only.*



(a) hypersphere - QPU



(b) hypersphere - Simulation

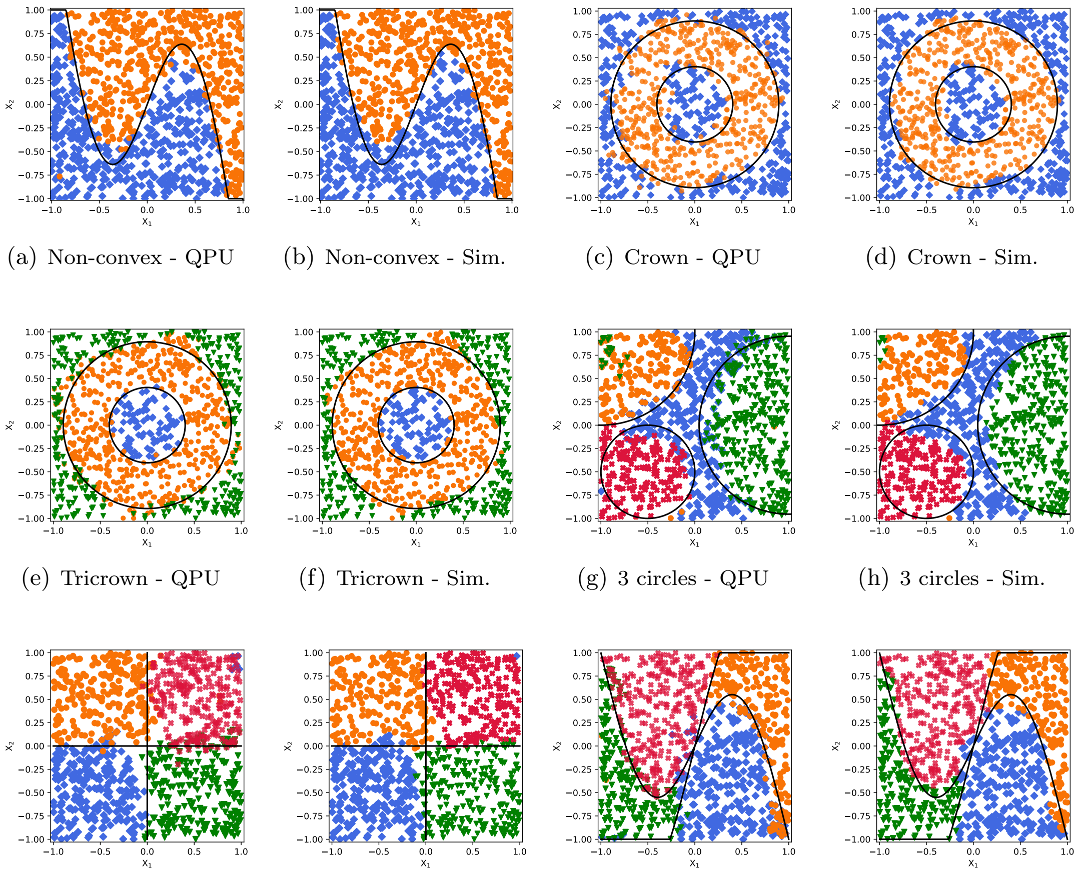


(c) Experimental optimization

Figure 3.20: *Binary classification of the hypersphere dataset. The histograms in (a) and (b) represent the class association of points within a hyper shell (given by the bin-width) denoted by blue (classified as within the hyper-sphere) and orange (outside the hyper-sphere) for QPU (a) and simulation (b). The overlap region shows the ambiguity in classifying the points within a certain radius. The accuracy of the QPU is improved by performing the experimental optimization near the vicinity of the simulated optima in a series of ten training steps. The reduction in the error with respect to the simulated results (b) is shown in (c).*

representation is only feasible due to the spherical symmetry of the problem.

Our results show that is is feasible to classify the data as well as the simulator does. As in the circle example, the parameter set inherited



(a) Non-convex - QPU (b) Non-convex - Sim. (c) Crown - QPU (d) Crown - Sim.

(e) Tricrown - QPU (f) Tricrown - Sim. (g) 3 circles - QPU (h) 3 circles - Sim.

(i) squares - QPU (j) squares - Sim. (k) wavy lines - QPU (l) wavy lines - Sim.

Figure 3.21: *Classifier test results for other classification problems. The ion trap based QPU classifier performed on* 1000 *random test data points. Colors and symbols stand for different classes, separated by the lines shown in solid black. The results are computed using 4 layers, both from QPU and simulation (Sim.). Notice that the border between classes in the experimental results is not as sharply defined as in the simulated classification. This difference is due to the uncertainty of the quantum measurements.*

from classical simulation implies a high error rate. This error can be further reduced from $\sim 13\%$ to $\sim 2\%$ after the experimental optimization step is performed, see Fig. 3.20(c).

Equivalent results for all problems presented in the theoretical model of the classifier were tested without experimental optimization. Results for 2D problems are depicted in Fig. 3.21. A summary of results comparing Quantum Processing Unit (QPU) and simulated methods is written in Tab. 3.11

Problem (# classes)		Classical algorithms		Quantum re-uploading			Ansatz
		Neural Network	Support Vector Machine	Simulation	QPU($\boldsymbol{\theta}^{\mathrm{sim}}$)	QPU($\boldsymbol{\theta}^{\mathrm{q}}$)	
Circle	(2)	0.98	0.96	0.97	0.93	0.96	A
Crown	(2)	0.71	0.82	0.92	0.87		B
Non-Convex	(2)	0.98	0.79	0.95	0.92		B
Sphere	(2)	0.95	0.91	0.74	0.66		A
Hypersphere	(2)	0.76	0.92	0.75	0.64	0.73	A
Tricrown	(3)	0.97	0.83	0.95	0.91		A
3 circles	(4)	0.93	0.92	0.90	0.85		B
Squares	(4)	0.99	0.95	0.97	0.93		A
Wavy Lines	(4)	0.99	0.89	0.94	0.90		A

Table 3.11: *Comparison between single-qubit re-uploading quantum classifier and two well-known classical classification techniques, namely single-hidden-layer neural networks and support vector machines. The experimental data and its simulated analogue is provided here with 4 Layer and 100 repetitions on the quantum part, and its equivalent in complexity for the neural network. The uncertainty of experimental data is* $\pm 2\%$. *The error refers to the standard deviation of* 10 *repeated trials performed on the same dataset and it implies that underlying systematic uncertainty leads to an uncertainty of the accuracy. Only two cases have been further optimized using an exploration done only with the quantum device.*

3.4.3 Discussion

The experimental implementation of the quantum classifier on an ion trap QPU was accomplished. The key ingredient for this implementation is the high quality of the control achieved by ion trap platforms for small systems. This is, to the best of our knowledge, the first experimental attainment of supervised learning problems with a single-qubit quantum processing unit. The experiment shows an advantage on the number of physical gates required when compared to classical approaches.

The experimental classifier is trained in two steps, a simulated and an experimental one, to enhance the performance of the algorithm. Gate-level fine tuning on the application of operations is also performed for further experiments. The experimental optimization ehances the accuracy up to $5-10\%$ depending on the problem, and final results are comparable to those obtained by the classical simulation of the quantum classifier and by classical methods, for models with similar numbers of parameters.

As in Sec. 3.2, this experiment suffers a lack of purely experimental optimization, in spite of the experimental optimization step. This stage is a refinement of parameters to obtain the same results provided by classical simulators while mitigating possible experimental noise. The difficulty of this task is not to be compared with a full optimization. Future works are expected to deal with this subject.

This work is the second experimental confirmation that the data re-uploading strategy is a useful scheme towards implementing QML algorithms in NISQ devices with few qubits.

3.5 Data re-uploading for determining the proton content

It has been shown through this chapter that data re-uploading is a general strategy for applying QML to classical data. Theoretical mathematical support that ensures the universality of this strategy is provided, and it is demonstrated by means of both numerical classical simulation and experiments on quantum processing devices that regression of functions and classification of data is possible on some testbeds. In this last section of the chapter a real-world machine learning problem is addressed with the data re-uploading strategy and show that this approach is general enough to be useful in a huge variety of scenarios. The problem here faced is related to High Energy Physics (HEP): determining the proton content from experimental data using ML strategies. This problem is of most importance since the knowledge of hadron contents, in particular for protons, is crucial for HEP experiments, such as LHC. The standard model predicts the behavior of the interactions between fundamental particles, but not between hadrons. Thus, the inner structure of hadrons must be well known in order to interpret experimental results properly.

This is not the first attempt to develop works in the frontier between quantum computing and HEP. This field is highly demanding in terms of computational resources, and thus any advantage provided by quantum computing would be useful. Some examples are computation of helicities [Bep+21], simulation of final-state radiation [Nac+21], and studies on the description of hadronic structure [Ale+19b; Li+21; LLa20].

Quantum Chromodynamics (QCD) provide theoretical recipes to understand and compute all possible interactions between particles in the standard model with strong charge, namely quarks and gluons [HM85]. These particles compose the nucleons, namely protons and neutrons. Thus, from a theoretical perspective, it is possible to know the exact content of a nucleon from first principles. However, the complexity of the calculations needed to accomplish this task is enormous. One must take into account all different possibilities of interactions to occur, which is extremely hard since the number of particles is not conserved, but it rather increases during collision processes. In addition, the theory is highly non-perturbative, and thus one cannot approximate to some given order since the remaining contribution is never neglictible with respect to the considered one. In summary, theoretical computa-tions to describe nucleons are not feasible nowadays.

The main framework to describe the non-perturbative structure of hadrons, protons in this case, are Parton Distribution Functions (PDF) [FC20; Fey88]. PDFs are typically determined from regression models that compare a large amount of experimental data and theoretical predictions. A well established technique for obtaining PDFs is the NNPDF methodology [Bal+15], where regression models are implemented through NNs.

The approach here proposed to address the PDF problem with QML is to make use of data re-uploading strategy to extract PDFs from experimental data, which is referred to as quantum PDF (qPDF). To accomplish this task, two steps were taken. First, the reference data is a set of PDFs obtained by classical methods and a quantum circuit, the qPDF, able to mimic this behavior is designed. Second, the obtained simulated results are submitted to public quantum hardware to benchmark the current performance of experimental devices on this real-world problem. Finally, the quantum model substitute the NN in NNPDF methodology to learn PDF directly from experimental data.

There are several reasons to attempt qPDF recipes, mainly regarding the efficiency of the algorithms. Since this example is the only one presented in the chapter that potentially matches the requirements of real-world problems, any waving must be taken into account. The first one is the reduction in energy consumption required to perform computations. Secondly, it is shown in the results here presented that the number of parameters required to reach acceptable qPDF fits is in average lower than in modern classical approaches. Furthermore, since PDF determination is an inherently quantum problem, the presence of entanglement may bring advantage to solve the regression task. Finally, quantum hardware can bring advantage in terms of running time as compared to the classical counterparts. The lower number of parameters implies that the number of operations needed to obtain comparable solutions is smaller. In addition, this model possesses an exact hardware representation, which is not possible in classical cases.

The results presented in this section are conceived as a proof-of-concept for future implementations of qPDFs on quantum hardware. Neither the performance of quantum simulation nor the quality of quantum hardware at the present time suffice to implement the qPDF approach more efficiently than current classical methods used in modern PDF determinations.

This section is structured as follows. First, the definition for the qPDF model is described in Sec. 3.5.1. Sec. 3.5.2 is devoted to the exact quantum circuits here used in terms of gates, parameters and test performance. Experiments for the circuits in actual quantum devices are carried in Sec. 3.5.3. Sec. 3.5.4 describes how to use the qPDF framework to extract PDFs from experimental data from LHC. Final remarks are written in Sec. 3.5.5.

Workflow design

The complete workflow of this project is composed of three different steps, schematically shown in Fig. 3.22.

1 Design of the most adapted quantum circuit Ansatz for qPDF

2 Feasibility study to deploy the qPDF architecture on real quantum devices
3 Integration of the qPDF model into a global PDF fitting framework taking the role of classical NNs

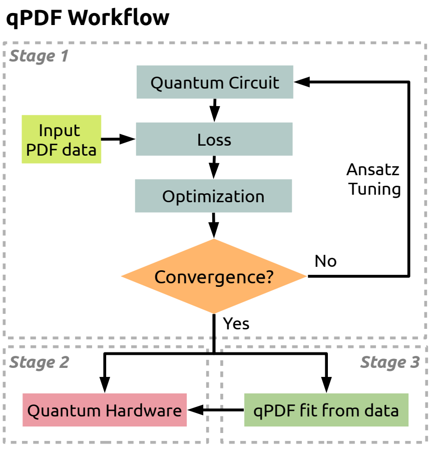


Figure 3.22: *Schematic workflow for the implementation of qPDF. The initial input data is a classical model for PDF. The Ansätze are modified until they are flexible enough to fit these data. Then, the same model is testes on an experimental quantum device. As a last step, the simulated circuit is introduced into a full PDF procedure to compare classical and quantum performances.*

In the first step several Ansätze for circuits were tested to choose the one with more flexibility to learn and generalize PDF-like datasets. The re-uploading strategy provides a general framework to design circuits capable to represent any data, but particular architecture is not determined by this approach. This stage is analogous to the hyper-optimization tuning performed in classical machine learning applications. However, since there are many possible architectures and no pre-defined initial Ansatz is assumed, empirical tests and refinement through trial and error is needed. The calculations are done using the exact classical simulation provided by `QIBO` [Eft+20a; Eft+20b], where both exact wavefunctions and expected values for hamiltonians are computed. For all tests, the model is trained to fit PDF generated by classical means, in particular from the NNPDF3.1 set [Bal+17]. This serves to find the optimal quantum circuits defined in the next subsection.

The second stage deals with the deployment of the qPDF model in actual quantum devices. In this case, both measurement and its uncertainty and noise models become relevant. This stage helps to determine the minimum number of measurements required to retrieve acceptable descriptions of PDFs on quantum hardware. The experimental implementation of the qPDF model is done using the `Qiskit` language [Ale+19a] from the `OpenQASM` [Cro+17] codegenerated by `Qibo`.

Finally, we use the qPDF model in an actual functioning classical PDF fitter based on experimental data, mostly from LHC measurements. The quantum model is integrated in the NNPDF fitting framework `n3fit` [CC19; FC20] using the simulation tools. This implementation opens the possibility to perform quantum fits of PDF on similar datasets for modern PDF releases.

3.5.1 Quantum circuits for PDF

We define in the following the Ansätze for quantum circuits needed to accomplish the qPDF fitting. First, the PDFs are defined as

$$f_i(x, Q_0), \tag{3.79}$$

where $x \in [0, 1]$ is the momentum fraction of the incoming hadron carried by the parton of flavour i (quarks and gluon), at an energy scale Q_0. The normalization rule for PDFs is

$$\sum_i \int_{x=0}^{x=1} x f_i(x, Q_0) = 1. \tag{3.80}$$

Following this definition, modifications to the data re-uploading are now added to accomodate it to PDF. These modifications leave unaffected the most important properties of data re-uploading, namely query complexity and arising of non-linearities.

To introduce the x variable in the quantum circuit, the same structure as in the previous examples is followed

$$|\psi(x, \Theta)\rangle = \mathcal{U}^{(k)}(x, \Theta) |0\rangle = \prod_{j=1}^{k} U(x, \vec{\theta}_i) \; |0\rangle \,, \tag{3.81}$$

where Θ are tunable parameters to be optimized through a loss function. The exact definition of the different layers is left for later since there are some features of the problem to be discussed before.

The second key ingredient in the model is the way to retrieve information from the circuit. In this problem, the structure of the proton is determined through several different functions, as many as flavours in the protons. Thus, many different independent measurements must be designed to extract information. A n-qubit circuit is considered to run the quantum algorithm on, where each qubit corresponds to one flavour. The set of hamiltonians to build is then

$$Z_i = \bigotimes_{j=0}^{n} Z^{\delta_{ij}}, \tag{3.82}$$

where δ_{ij} is the Kronecker delta. The choice of hamiltonians is heuristic, and measures the population of $|0\rangle$ and $|1\rangle$ states of a particular qubit. The values of qPDFs will be then related to the probability to measure one particular qubit in the excited state. The function

$$z_i(x, \Theta) = \langle\psi(x, \Theta)| \, Z_i \, |\psi(x, \Theta)\rangle \,, \tag{3.83}$$

is taken as the outcome of the circuit. The next step is then to relate this outcome to the PDF values. Every $z_i(x, \Theta)$ will be associated to only one parton i, and thus as many qubits as partons are needed. The qPDF model at a given (x, Q_0) coordinates is then

$$qPDF(x, Q_0, \Theta) = \frac{1 - z_i(x, \Theta)}{1 + z_i(x, \Theta)}. \tag{3.84}$$

This choice, as well as the hamiltonian, is heuristic and supported by empirical results. It only allows the qPDFs to take positive values, although there is no upper bound in this quantity. This is no hard constraint since it is possible to drop the possitivity with simple re-scaling. Theoretical motivations can be drawn from the fact that PDFs can be made non-negative [CFH20], but their values can in principle grow to any real value, see for instance the gluon PDF in Fig. 3.24.

3.5.2 Ansätze

It is seen and discussed that the re-uploading of classical data in conjunction with tunable weights and biases following the scheme $wx + b$ permits to represent arbitrary functions, for example PDFs, in single-qubit systems, see Theorems 3.1.2 and 3.1.4. On the other hand, Ref. [SSM21] shows that a Fourier approximation of an arbitrary function is possible even if the weights are fixed. Therefore, two different Ansätze were considered in this work, where the main difference between them is the presence or absence of tunable weights.

The x independent variable of the dataset is constrained to take values between 0 and 1. However, the representation of PDF is usually given in a logarithmic scale in x since variations in PDF among orders of magnitude are prominent. This observation encourages using not only x as the independent variable, but also $\log(x)$ to properly capture the behavior of the functions at small scales.

The first Ansatz, named Weighted Ansatz, is inherited from the works in Refs. [Pér+20a] and later [Pér+21b]. The x variable is introduced following the weights and biases scheme. The single-qubit operation acting as building block of the Ansatz is

$$U_w(\theta, x) = R_z(w_2 \log(x) + b_2) R_y(w_1 x + b_2). \tag{3.85}$$

Note the presence of rotations around two different axis, Y and Z, in the definition of this gate, to make non-linearities emerge from the quantum mechanical nature of the single-qubit operations. In addition, each axis introduces one scale, namely linear or logarithmic.

The second Ansatz, Fourier Ansatz, limits the weights to fixed values [SSM21]. Its single-qubit operation is

$$U_f(\theta, x) = R_y(b_2) R_z(w_2) R_y(-\pi/2 \log x) R_y(b_1) R_z(w_1) R_y(\pi x), \tag{3.86}$$

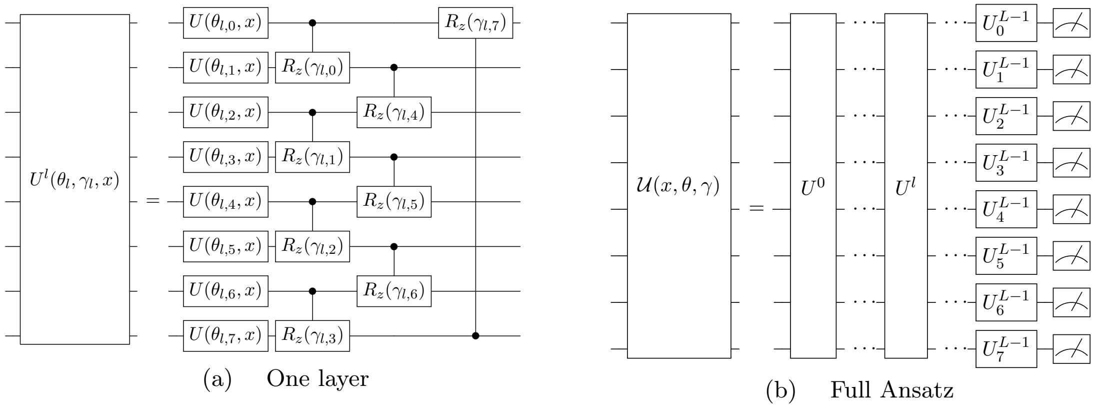


Figure 3.23: *On the left, an example of one layer architecture is shown. On the right, the scheme of a full Ansatz circuit including 8 qubits and entangling gates. The $U^l(\theta_l, \gamma_l, x)$ from the left figure enters the full ansatz as U^l. Note that the last layer does not have any entangling gate.*

where again two axis are involved. The choice of the weights π in the linear scale and $-\pi/2$ in the logarithmic scale depends on the dataset. For the PDF set chosen to test the Ansätze, $x \in [10^{-4}, 1]$. This way, all the amplitude of the single-qubit gates is exploited.

The single-qubit operations are introduced into a more general structure to create several-qubits multi-layered global Ansätze to fit the PDFs. The reason for this procedure is that it is expected to obtain better approximations as more layers are added to the circuit and the query complexity of the algorithm increases. The construction of layers is made in two steps. First, a layer of parallel single-qubit operations is applied to each qubit in the circuit. Second, a layer of entangling gates is added to the circuit. In this problem, all entangling gates are controlled Z rotations depending on some parameter $R_z(\gamma)$. Entangling gates connect one qubit with next or previous one alternatively. Sections of rotations and entangling gates are interspersed along the circuit, except for the last iteration where only single-qubit gates are considered. Parameters for every gate are independent from all other parameters and optimized simultaneously. See Fig. 3.23 for a schematic description of the Ansatz structure.

For the first stage of tuning Ansätze, simulation methods are used. The optimization procedure takes as loss function the standard Pearson's χ^2[Pea00] to compare qPDF predctions against classical calculations of NNPDF3.1 NNLO [Bal+17]. This dataset is composed by a central value f_i and an uncertainty σ_{f_i}, both depending on x and Q_0. For this exercise, a grid of x values $x \in [10^{-4}, 1]$ and a fixed value $Q_0 = 1.65$ GeV are considered. The dataset of interest contains 8 different flavours, namely

Single-flavour fit		Multi-flavour fit	
Layers (Parameters)	χ^2	χ^2	Layers (Parameters)
1 (32)	28.6328		1 (32)
2 (64)	1.0234	–	–
3 (96)	0.0388	0.1500	2 (72)
4 (128)	0.0212	0.0320	3 (112)
5 (160)	0.0158	0.0194	4 (152)
6 (192)	0.0155	0.0154	5 (192)

Single-flavour fit		Multi-flavour fit	
Layers (Parameters)	χ^2	χ^2	Layers (Parameters)
1 (32)	900.694		1 (32)
2 (64)	57.2672	–	–
3 (96)	0.0410	47.4841	2 (72)
4 (128)	0.0232	0.0371	3 (112)
5 (160)	0.0165	0.0216	4 (152)
6 (192)	0.0156	0.0160	5 (192)

Table 3.12: *Comparison of χ^2 values for the Weighted (left) and Fourier (right) Ansätze average of all single-flavour fits and the corresponding multi-flavour fit.*

	Single-flavour		Multi-flavour	
	Weighted	Fourier	Weighted	Fourier
Qubits (q)	1 (per flavour)		8	
Layers (l)	5		5	
Parameters	$2 \cdot l \cdot q$ weights $2 \cdot l \cdot q$ biases	$4 \cdot l \cdot q$	$16 \cdot l$ weights $16 \cdot l$ biases	$32 \cdot l$
	No entanglement		$8(l-1)$ entangling	

Table 3.13: *Summary for the Ansätze chosen for this work. The preferred number of l ayers was chosen as a compromise between small χ^2 and number of parameters. Results depicted in Tab. 3.12 determine that the multi-flavour Weighted Ansatz is the best candidate model.*

quarks, antiquarks and the gluon: $i \in \{\bar{s}, \bar{u}, \bar{d}, g, d, u, s, c(\bar{c})\}$.

Values of Pearson's χ^2 after full optimization are summarized in Tab. 3.12, both for Weighted and Fourier Ansätze. In both cases, the left column shows an average fit for all flavours when optimized individually, while the right column shows the fit for all flavors simultaneously. Comparisons are made between models with similar numbers of parameters. For the same number of layers, circuits with entanglement have more parameters since every entangling gate has one of them. Thus, unentangled circuits of n layers with entangled ones with $n-1$ layers are compared until both numbers of parameters agree. The reason for this comparison is that entanglement is expected to improve the overall quality of this method.

Standard classical optimization methods were used to find optimal parameters. The optimization procedure was carried in two steps. First, the `CMA` genetic algorithm is used to find optimal solutions in the single-flavour scenario [Han06; nik+20]. Then, this result is taken as starting point in the multi-flavour optimization carried through the `L-BFGS-B` method [Byr+95] included in the library `scipy` [Vir+20]. This two-step optimization ensures that competitive results are obtained for the multi-flavour cases.

Some results of the final configuration after full optimization are depicted in Figs. 3.24 and 3.25. Fig. 3.24 show the multi-flavour qPDF

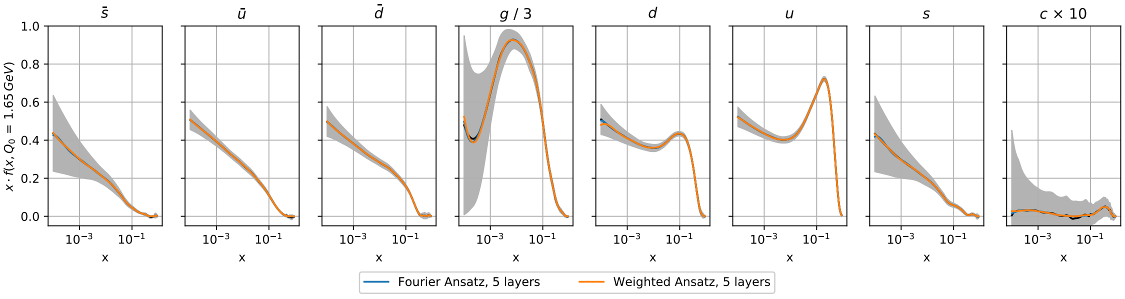


Figure 3.24: *Multi-flavour qPDF fits using the Weighted Ansatz (orange curves) and the Fourier Ansatz (blue curves) with 5 layers and 8 qubits. The mean value and 1σ uncertainty of the target PDF data is shown by means of a solid black line and a shaded grey area.*

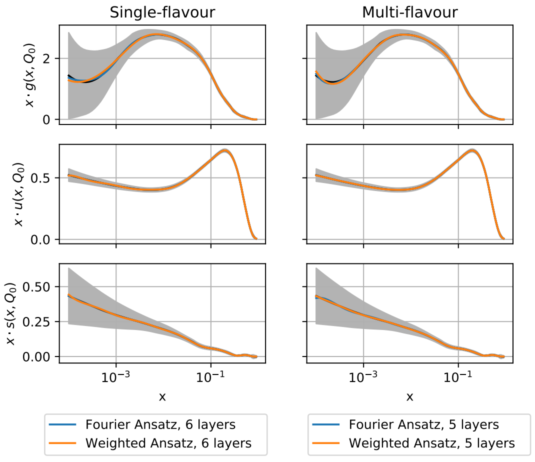


Figure 3.25: *Comparison between single-flavour fits (left) and multi-flavour fits (right) for the gluon, up and strange quarks PDFs. For the single-flavour fits the Weighted Ansatz (orange curves) and Fourier Ansatz (blue curves) are composed by 1 qubits and 6 layers. On the other hand for the multi-flavour fits, the Ansätze are composed by 8 qubits and 5 layers. The mean value and 1σ uncertainty of the target PDF data is sshown by means of a solid black line and a shaded grey area.*

both for Weighted and Fourier Ansätze, for 5 layers, and 8 qubits, as compared to the classical results. In Fig. 3.25, a comparison of single- and multi-flavour optimizations focusing on certain flavours is detailed, for Weighted and Fourier Ansätze with similar numbers of parameters. Notice that in both cases qPDFs and classical data overlap with high accuracy

The results here presented permit to claim several interpretations. First, entanglement is not enough to provide good approximations by its own. Entanglement can in principle access to the correlations between different qubits, which in this case encode different flavours and qPDF. However, every layer introduces a new re-uploading of data and takes another step towards non-linearity, which is needed for representing arbitrary functions. Results from Tab. 3.12 show that entanglement, and possibly more parameters, help to obtain better results when comparing models with the same query complexity. For the same number of parameters, both number of layers and entanglement provide the same capabilities. Thus, both entanglement and query complexity contribute to the overall performance. Secondly, the goodness of the Weighted Ansatz respect to the Fourier one is unveiled. Built-in weights grant large flexibility, especially in cases with small numbers of layers.

We retain the Weighted Ansatz with 5 layers as the final model, both for single- and multi-flavour scenarios. For the sake of comparison, equivalent Fourier Ansätze were chosen as well. The total amount of parameters is 192, which is a manageable number, see Tab. 3.13 for a detailes comparison. In addition, tests run on both Ansätze reveiled that the Weighted Ansatz is easier to train using gradient-based methods like `L-BFGS-B`.

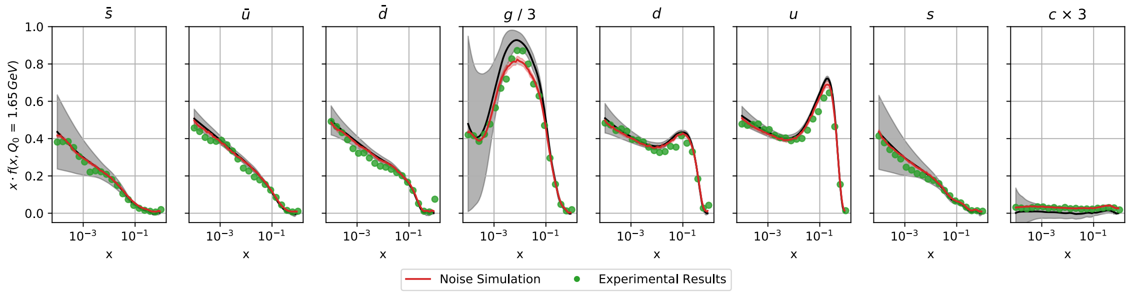
3.5.3 Experimental configuration

Low-depth quantum circuits are able to represent a full set of PDFs when the calculations are carried by means of classical simulation. However, in the results presented up to this moment, no measurement uncertainty or noisy executions of quantum circuits have been taken into account. In this section it is explored how the capabilities are transferred the theoretical models to realistic quantum computers. First, the trained single-flavour model onto the IBM Athens quantum processor [Ale+19a] is loaded to check how much the noise degrades the final results. The resilience of ths quantum model in the single-flavour scenario is expected to be larger than in the multi-flavour case because of the absence of entangling gates. Gate fidelities for two-qubit gates are around an order og magnitud worse than for single-qubit operations.

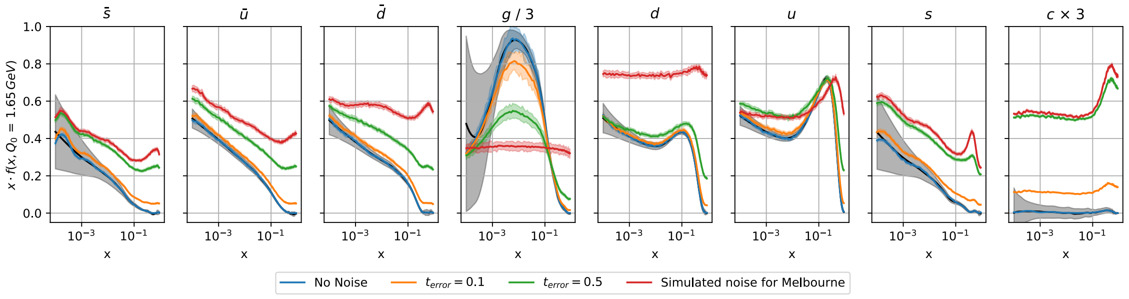
For experimental results, each parton is evaluated at 20 values of x logarithmically spaced in $x \in [10^{-4}, 1]$. The expecation value $z_i(x, \Theta)$ is evaluated for every x with $2^{13} = 8192$ shots. Then, each evaluation is repeated 5 times to probe statistical averages and uncertainties in estimation. See Fig. 3.26(a) for a comparison between experimental results in the IBM Athens quantum and its corresponding noisy simulation as provided by `qiskit`. From those results it is possible to claim that single-flavor models with only one qubit perform properly on cloud-accessible quantum processors, and it is then possible to extrapolate this model to actual machines. In addition, the agreement between experimental results and simulation credits the simulation environment to properly simulate the real situation.

The next step is to extend the experimental implementation to multi-flavor models. Quantum computers required for theses models must have several qubits and be able to execute two-qubit gates. It is expected that decoherence play an important role in this case. It is benchmarked by executing the optimized model on the `qiskit` simulator for noisy computers, taking as noise model the one corresponding to the best possible configuration of the IBM Melbourne quantum processor. This processor was chosen since it is the only publicly available processor with enough number of qubits. The 8 qubit circuit is mapped onto Melbourne in a way that it matches the chip architecture and the entangling gates are directly applicable.

The first step of the multi-flavor execution on a virtual quantum



(a) Single flavour fits in IBM Athens



(b) Multi-flavour fits in IBM Melbourne-like simulator

Figure 3.26: *a) Single-flavour fit for all flavours. The red lines represent the prediction of the qPDF model with simulated noise from the IBM Athens processor [Ale+19a]. Green points are the results of running the circuit on the Athens quantum processor. The mean value and 1σ uncertainty of the target PDF data is shown by means of a solid black line and a shaded grey area. b) Multi-flavour fit for all flavours. Blue lines are the mean and the blue shadowed area the 1σ uncertainty of the circuit measurement results for an ideal noise free quantum device. The red curve refers to simulated circuit measurements using the noise model for the IBM Melbourne processor [Ale+19a]. Similarly, green and orange curves show simulation results with noise reduced by 50% and 90% respectively. The mean value and 1σ uncertainty of the target PDF data is shown by means of a solid black line and a shaded grey area. In both cases the Weighted Ansatz, for 5 layers and 8 qubits was used.*

computer is to add measurement gates. In this case, the circuit must be run 8 times, one per flavor, to measure each qubit independently from the other ones. As in the single-flavor scenario, each qubit is measured 8192 times, a sufficient number to estimate the expected value of the hamiltonians of interest with low uncertainty, and thus reconstruct the PDF accurately. It is immediately seen that the errors present in the IBM Melbourne device drastically deteriorate the performance of the pre-trained quantum model, see Fig. 3.26(b). To understand better the effect of noise, the noise environment is used to create simulators with the same noise structure as IBM Melbourne but smaller noise values. Thus, a noise model $\mathcal{N} = \mathcal{N}_{\text{Melbourne}} t$ is available, where $\mathcal{N}_{\text{Melbourne}}$ is the noise model provided by `qiskit` and t is an interpolation parameter. This way, the noise model can be linearly scaled down while maintaining all the characteristics, namely connections among qubits, single-, two-qubit and readout errors and thermalization. For $t = 0$, no noise is considered, while for $t = 1$, the full device is simulated. Results for $t = \{0, 0.1, 0.5, 1\}$ are depicted in Fig 3.26(b).

A summary of the obtained results is also depicted in Fig. 3.27. Here, the relation between the obtained χ^2 and the ideal one is explored as a function of the error parameter t. The aim is to explore how robust must a quantum computer be in order fo return acceptable representations of qPDF. It is possible to see that a χ^2 an order of magnitude larger than the ideal one is achieved at (extrapolated) value $t \sim 0.007$.

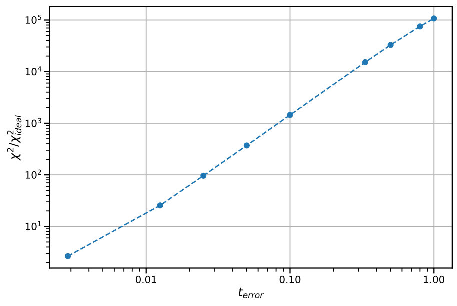


Figure 3.27: *The error as a function of the error interpolation parameter t_{error}. The y-axis is given as the ratio between the error, χ^2, and the error on an ideal quantum computer, χ^2_{ideal}.*

The analysis shows that even though it is theoretically possible to fit PDF with the qPDF model here proposed, as it was demonstrated by means of classical simulations, the noise and decoherence in state-of-the-art quantum devices are still to high to provide accurate computation frameworks.

3.5.4 qPDF determination from experimental data

In previous sections the process of finding a quantum circuit capable to capture the properties of physical PDFs by mimicking classically known results is described. Furthermore, the possibility to extend theoretical models to experimental devices is also explored. In this section the last step of the workflow is addressed, see Fig. 3.22, andthe qPDF model is used to learn PDFs from the only available dataset in reality, that is experimental measurements of physical observables, in this case physical cross sections measured at the LHC.

In this stage it is proven that the qPDF methodology has the potential to replace NNs underlying at the core of the NNPDF family of proceedings to learn PDFs from experimental data classically. Current quantum devices are far from supplying enough computational capability to tackle this problem in practice. However, classical simulation shows that the data re-uploading strategy can indeed replace NNs as a universal function approximation of arbitrary functions, such as PDFs, at least from a theoretical perspective.

This section describes the NNPDF methodology and the changes needed to incorporate the qPDF model to the general framework to perform a full fit. The dataset used to fit PDFs is the NNPDF3.1, including deep inelastic scattering and hadronic collider data. Finally, the obtained PDFs and qPDF are compared showing that results are compatible and usable in realistic computation of physical observables.

The NNPDF methodology

NNPDF methodology is based on two main ingredients. First, a Monte Carlo approach to synthetic generation of artificial measurements is required. Second, NNs are used to model PDFs. In the following some main aspects are outlined, and the reader is referred to a more in-depth review for further details [Bal+15].

First, data replicas must be generated. This procedure propagates experimental uncertainties through the PDF fit by leveraging experimental uncertainties obtained from experiments. Synthetic copies of data are then created and indistinguishable from actual data.

The PDF fit is done following the functional form

$$f_i(x, Q_0) = x^{-\alpha_i}(1-x)^{\beta_i} NN_i(x, Q_0), \tag{3.87}$$

where i is the parton of interest and $NN(\cdot)$ is the function provided by the NN. The preprocessing factors $x^{-\alpha_i}, (1-x)^{\beta_i}$ guarantee a correct behavior for $x \approx \{0, 1\}$, where there could be a lack of experimental data to properly constraint the NN. This function cannot be directly compared to experimental data, it must be convoluted with the partonic cross rection to obtain physical predictions comparable to measurable observables,

$$P = \int \mathrm{d}x_1 \, \mathrm{d}x_2 \, f_1^i(x_1, q^2) f_2^j(x_2, q^2) |M_{ij}(\{p_n\})|^2, \tag{3.88}$$

where x_1, x_2 are momentum fractions of two particles, and $\{i, j\}$ run over all possible partons. The quantity M_{ij} is the matrix element for the given processes, computed analytically with other methods, and $\{p_n\}$ represent all momenta involved in the computation. Nevertheless, a numerical integration of this quantity would be impractible. Instead, theoretical predictions are approximated as products between the PDF functions and a fastkernel table with the relevant information [Bal+10; BCH17].

The optimization process consists in minimizing the Pearson's χ^2 defined as

$$\chi^2 = \sum_{i,j}^{N_{\rm dat}} (D-P)_i \sigma_{ij}^{-1} (D-P)_j, \tag{3.89}$$

where D_i and P_i are respectively the i-nth data point from the training set and its theoretical prediction and σ_{ij} is the experimental covariance matrix.

This procedure is then repeated for each synthetic replica. In all cases, only the experimental data changes and must be calculated in every step. The final PDF is the average among all replicas and the error bands are given by enveloping 68% of the replicas, that is the number associated to a 1σ width in a normal distribution.

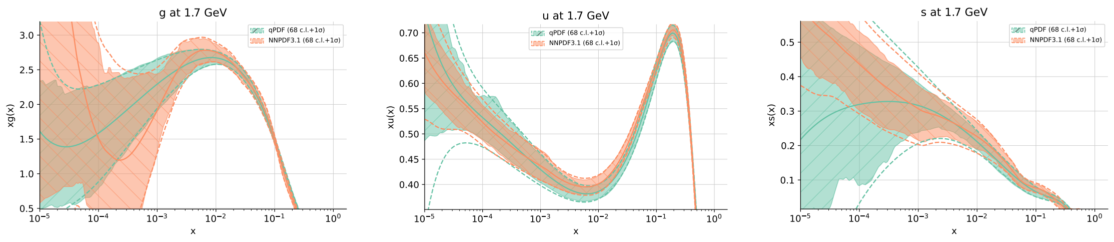
The latest NNPDF methodology, NNPDF3.1, as described in Ref. [CC19] is taken. The NN module is replaced with the qPDF model. Since both `qibo` and NNPDF3.1 are based on `tensorflow`, integration can be accomplished without adapting the programming language. A number of changes must be done in order to make the integration complete. The dataset included in this fit correspond to that of NNPDF3.1, which is detailed in Ref. [Bal+17] and includes data from deep-inelastic scattering experiments, fixed-target data and hadronic collider data from experiments at Tevatron and LHC. Technical details can be found in App. A.6.

Experimental qPDF results

We compare the published reference PDFs and their uncertainties to the qPDF results obtaining with this method to check that both results are compatible. Quarks u and d, and gluon g are depicted in Fig. 3.28. For these flavors, the qPDF central result is within the $1-\sigma$ uncertainty bar of the reference classical PDF, and both classical and quantum uncertainty bars have strong overlaps for the considered range.

Phenomenological implications must be addressed to benchmark the performance of the qPDF model. Examples related to the most common Higgs production channels with $m_H = 125$ GeV are summarized in Tab. 3.14. In this table, cross sections are depicted for different channels and compared to reference values [Alw+14; Car+20; SC20]. For all examples, reference values and qPDF results are compatible within uncertainty ranges.

These results support the claim that the NNPDF methodology where NNs are replaced with quantum circuits can be used for regression



(a) Gluon PDF. (b) u quark PDF. (c) s quark PDF.

Figure 3.28: *Fit results for the gluon and the u and s quarks. As previously seen in Fig. 3.24, qPDF is able to reproduce the features of NNPDF3.1. This is also true when the fit performed by comparing to data and not by comparing directly to the goal function. The differences seen at low-x can be attributed to the lack of data in that region.*

Channel	NNPDF3.1 NNLO	qPDF
ggH	31.04 ± 0.30 pb	31.71 ± 0.51 pb
$t\bar{t}H$	0.446 ± 0.003 pb	0.464 ± 0.008 pb
WH	0.133 ± 0.002 pb	0.135 ± 0.002 pb
ZH	0.0181 ± 0.0002 pb	0.0184 ± 0.0002 pb
VBF	2.55 ± 0.03 pb	2.62 ± 0.04 pb

Table 3.14: *The cross-sections for Higgs production at 13 TeV in various channels at NLO using the settings described in the text. From top to bottom: gluon fusion, $t\bar{t}H$ production, WH production, ZH production and vector boson fusion. Standard Model Higgs boson with mass $m_H = 125$ GeV is assumed.*

problems to unknown functional forms, in particular the PDFs and inner structure of the proton. Classical state-of-the-art and quantum results are coherent from a phenomenological point of view. In addition, it is reasonable that the same level of accuracy from classical methods can be reached with adequate fine-tuning.

3.5.5 Discussion

It is possible to highlight certain advantages that the quantum model here proposed has in comparison to standard ML methodology for fitting PDFs. First, non-linearities emerging from quantum operations and entanglement help to reduce the number of parameters required to obtain a flexible PDF representation, as compared to equivalent NNs. Second, from a hardware perspective, the implementation of a qPDF fit on a quantum processor and the use of native gates as operations can accelerate the evaluation and training of PDFs. In addition, it is expected that the energy consumption of quantum devices is smaller than of classical hardware based on hardware accelerators such as graphical processing units.

However, the current status of quantum devices does not provide enough quality to implement this model. All typical difficulties from experimental quantum hardware appear in this problem, including noise, decoherence and measurement uncertainty. More accurate or even fault tolerant quantum computing will be necessary for succesfully carrying this method on quantum devices.

On the other hand, the results here presented should be considered a proof-of-concept. Standard machine learning implementations are so optimized that no quantum simulation algorithm is competitive in terms of performance nor efficiency against classical methods. The advantage of qPDFs will come as the quantum hardware becomes more precise.

The work here presented is a first attempt to join quantum machine learning into the field of PDF determination. This approach can open a new surge of algorithms for the field of HEP, which can benefit from quantum computing.

3.6 Conclusions

Throughout this chapter the re-uploading strategy is developed since its inception to several examples of applications. All the contents here covered support the claim that data re-uploading is a general strategy to employ in QML problems and to apply in the first generation of NISQ devices, including some already existing processors.

Re-uploading strategy is a general approach whose main focus is to replace a NN with some quantum circuit. The circuit can be as small as one qubit. The key ingredient is to upload data several times in subsequent operations applied to the circuit to permit multiple processing of data. As compared to NNs, each operation receiving data is the quantum counterpart of a neuron. For single-qubit circuits, the comparison to single-hidden-layer NNs is direct. In fact, it is mathematically proven that both models are formally equivalent. Even though the relationship between deep learning classical approaches and multi-qubit re-uploading circuits is not clear, it is expected that steps are taken in this direction in the future.

Data is uploaded to the circuit following the linear mapping used in NNs, that is as $\vec{x} \rightarrow \vec{w} \cdot \vec{x} + \theta$, with $\vec{w}$ the weight and θ the bias. This encoding allows to encode data in an unbiased manner, thus it is possible to carry supervised learning problems without any prior knowledge of the dataset to classify.

The cornerstone of the re-uploading strategy is that non-linearities emerge naturally from the quantum properties of the circuits. By applying two gates around different axis, one of them depending linearly on x, non-linear terms arise as a consequence of the non-commutativity of quantum operations. Quantum circuits are capable to represent any functional form thanks to this inherently quantum property.

Data re-uploading is also a strategy to circumvent the no-cloning theorem. In this approach, classical data is introduced several times in the circuit, but the copy of data is not performed using quantum, but classical resources. It is then possible to increase the number of calls from the circuit to the data, unlike in most QML approaches where data is introduced at the beginning of the algorithm.

The performance of the re-uploading strategy is directly related to the query complexity of the circuit, that is the number of re-uploading along the algorithm. In fact, this quantity remains approximately constant for similar performances, while the number of qubits or depth in circuits are more unstable. Entanglement plays a role in the performance, but the improvement attained is not as significant as expected.

Present results give support to different QML applications. The first one is regression. It is shown in Sec. 3.2 that the quantum algorithm is capable to learn arbitrary functional forms from sampling data. Sections 3.3 and 3.4 address supervised classification tasks successfully. In both cases, both numerical and experimental benchmarks are performed. Also in both cases, the training step is mostly carried on classical simulations of quantum systems. The reason to proceed in this way is that optimization on quantum devices is much harder than on classical devices. Thus, actual quantum optimization is a task left for future extensions of the present work.

In Sec. 3.5, a real-world problem is addressed with the re-uploading strategy. The problem is determining the proton content from experimental data of HEP. This application, although only classically simulated, presents a strong evidence that re-uploading approaches can actually deal with real problems. Current status of experimental devices does not endure running running this kind of algorithms due to the yet persistent quantum noise and decoherence.

It is expected that future improvements in the device quality and optimization methods will help the data re-uploading strategy to enhance its range of applicability and performance, and it will hopefully be an interesting ingredient in the development of QML.

4. Unary strategy for finance

Más vale el buen nombre que las muchas riquezas.
Miguel de Cervantes

Quantum computing posseses the inherent property of entanglement. Entanglement is the reason for the exponential size of the Hilbert space with respect to the number of qubits. Entanglement is also a key feature in the speed up achieved in the most prominent quantum algorithms [Gro96; Sho97]. However, there is another application available for entanglement, that is effectively distributing the information of a small quantum system across a larger one. The reason is that this procedure allows to store the information as a global and share property of many small quantum system, making it resilient against errors and decoherence. For example, this strategy lies at the core of quantum error correction codes [Cor+98; Got97; Sho95].

In this chapter an application of this line of thought is explored, looking for computing financial products using a near-term NISQ device. The main difference between the algorithm here proposed and other quantum algorithms, in particular for finance, is that the **unary representation** is used, that is, the Hilbert space is restricted to those components of the computational basis with only one $|1\rangle$ qubit, and $|0\rangle$ for all other qubits. Previous works considering the unary representation can be found in Refs. [Bab+18; Pou+18; SW20].

The unary basis has some advantages and inconvenients with respect to the standard binary computations. It is clear that the unary basis does not have the capabilities to store an exponential number of coefficients with respect to the number of qubits, but rather a linear one. Thus, a worse asymptotic scaling is retrieved in the unary case. However, this permits to simplify logical operations carried on those quantum states and leads to lesser gates to be applied on a given circuit. In addition, the unary representation brings a native post-selection strategy that results in error mitigation. It is likely that all algorithms to be executed NISQ must perform some error mitigation technique to obtain proper results.

The properties of the unary basis make it useful in a near-term regime, both with respect to the size of the problem and the quality of the quantum computer. In case the size of the problem increases, the overhead of the standard binary representation with respect to the unary one is compensated by the asymptotic behavior. In terms of resilience against error, the unary basis can provide better results as well. Quantum advantage is feasible for small problems, and it is also possible to find interesting problems where this small size is useful.

The benefits of the unary representation are explored in the field of quantitative finances. This field is expected to be transformed with the bloom of the first generation of quantum computers, see Ref. [OML19b]. In recent years there has been a surge of methods and algorithms for solving financial problems using quantum computers [Egg+19; Mar+21; OML19a] , in particular for hard optimization problems [KPS19; Mol+18; Pra15; RL18; Ros+16].

The prominent problem of pricing financial derivatives is taken into consideration. Many computational obstacles of this problem can be overcome by quantum computing, in particular, the pricing of european options. Options are contracts that allow the holder to buy / sell some asset at a pre-established time at a future date. The problem is then to estimate if the price of the given option will increase or not with respect to the agreed value. The evolution of the asset price follows a stochastic process described by the Black-Scholes model [BS73]. Then, a payoff function, specified by contract as well, must be incorporated to this evolution to obtain the expected return of the option. The main method to classically perform this computation is the costly Monte Carlo simulation.

Quantum algorithms have been already proposed to solve the option pricing problem more efficiently that their classical counterparts [RGB18; Sta+20; WE19]. The key ingredient to develop this algorithm is that quantum computers provide a quadratic speed-up in the number of evaluations required to obtain a given accuracy using a Monte Carlo simulation. This exploits the idea of QAE [AR20; Bra+02; Gri+21; Mon15]. The quantum advantage achieved via Quantum Amplitude Estimation (QAE) holds only if there exists an efficient way of loading of data, namely a distribution in the asset prices, into the quantum circuit. With this purpose, qGANs [DK18; LW18b] have been analyzed to address this issue [ZLW19].

The algorithm here proposed to solve the option pricing problem is divided in three steps. First, a circuit working on the unary basis of the asset prices is constructed. The evolution of the asset price is computed using an amplitude distributor such that the output state corresponds to a probability distribution of prices. Second, the payoff is computed using quantum gates. This steps greatly simplifies thanks to the unary representation. Finally, a QAE procedure is carried to obtain quantum advantage. This last step is common to previous approaches.

The option pricing problem suits the requirements for a unary basis approach since a great accuracy is not needed to obtain results of interests. The estimates for the number of gates indicate that the crossing point between the unary and the binary algorithm is located at least in a number of qubits rendering a good precision, $< 1\%$. This rate matches the usual precision obtained in real-world applications. It is worth mentioning that this estimation relies on machines suiting perfectly the needs of this algorithm, but real hardware architecture can lean the scale towards the unary approach.

This chapter is organized as follows. First, the unary representation and its corresponding operations are presented and defined in Sec. 4.2. Sec. 4.1 covers the required background to present the financial problem here addressed. The application of the unary representation to finance is depicted in Sec. 4.3. A comparison between the previous binary algorithm and the unary one in this chapter is performed in Sec. 4.4, and the corresponding results are depicted in Sec. 4.5. Final remarks can be read in Sec. 4.6.

4.1 Background

The unary algorithm here presented is constructed upon three different legs coming from different fields. They are the economical Black-Scholes model as applied for European options, the QAE procedure providing quantum advantage respect to standard Monte Carlo, and a quantum algorithm developed in the standard binary representation to compute the payoff of an European option pricing, as described in Ref. [Sta+20].

4.1.1 European options and the Black-Scholes model

In the market of financial derivative, options are contracts signed to acquire the right to buy/sell (*call*/*put*) some asset at a previously established price (*strike*). The contract expires at a future point in time (*maturity date*). The holder of the contract will only execute the call/put option if the actual price is lower/higher than the agreed strike, and thus some benefit is obtained from the trading.

To decide whether an option is profitable at the time of the contract sign, the holder must estimate the expected payoff of the option. This quantity will depend on the evolution of the price of the asset, which follows a stochastic process. A simple, yet successful model for option pricing is the Black-Scholes model [BS73], to be detailed shortly. The second step consists in computing the contract specified payoff function over the Black-Scholes distribution of prices to obtain the expected return. In particular, European options only allow to call/put the asset exactly at the maturity date. This is then the only data of interest. In contradistinction, other options in the market, for instance the Asiatic or American options rely on more sophisticated payoff functions.

The Black-Scholes model for the evolution of an asset price is described by the stochastic equation

$$\mathrm{d}S_T = S_T\, r\, \mathrm{d}T + S_T\, \sigma\, \mathrm{d}W_T, \tag{4.1}$$

where T is the time and S_T is the price at time T, r is the interest rate, σ is the volatility and W_T describes a Brownian process. Brownian processes W_T are continuous stochastic evolutions starting at $W_0 = 0$ and consisting of independent gaussian increments. Specifically, let $\mathcal{N}(\mu, \sigma_s)$ be a normal distribution of mean μ and standard deviation σ_s. Then, the increment related to two steps T, S of the Brownian processes is $W_T - W_S \sim \mathcal{N}(0, T - S)$, for $T > S$.

The most important feature for the Black-Scholes model is that there exists a first-order-approximate analytical solution to Eq. (4.1). The solution reads

$$S_T = S_0 e^{(r-\frac{\sigma^2}{2})T} e^{\sigma W_T} \;\sim\; e^{\mathcal{N}\left(\left(r-\frac{\sigma^2}{2}\right)T, \sigma\sqrt{T}\right)}, \tag{4.2}$$

corresponding to a log-normal distribution. A more detailed description of the procedure to solve this model is outlined in App. B.1.

The expected return of an option is computed by integrating the payoff function over the price probability distribution. This step is usually carried by means of a Monte Carlo simulation. Depending on the option, this procedure can be extremely costly.

For European options, the payoff function is computed as

$$f(S_T, K) = \max(0, S_T - K), \tag{4.3}$$

and thus the expected payoff is

$$C(S_T, K) = \int_K^\infty (S_T - K)\; dS_T, \tag{4.4}$$

where K is the agreed strike price. Since European options are executable only at the maturity date, the expected payoff can be computed only by integrating at time T. In contradistinction, Asiatic options return an average over time. American options allow to execute the contract at any point before the maturity date, thus the calculation of the expected payoff requires more sophisticated and costlier methods [KV90].

4.1.2 Quantum Amplitude Estimation

QAE is an inherently quantum technique designed to estimate the probability of measuring a certain outcome from a given state more efficiently as direct sampling. For a given precision, the number of function calls is quadratically reduced with respect to direct sampling [Bra+02; Suz+20b]. This technique lies at the core of the quantum advantage obtained for Monte Carlo simulations [Mon15]. QAE extends the ideas developed by Grover algorithm [Gro96].

The QAE procedure considers an algorithm $\mathcal{A}$ such that

$$\mathcal{A}\left|0\right\rangle_n \left|0\right\rangle = \sqrt{1-a}\left|\psi_0\right\rangle_n \left|0\right\rangle + \sqrt{a}\left|\psi_1\right\rangle_n \left|1\right\rangle , \tag{4.5}$$

where the last qubit serves as a flag qubit separating *good*($\left|0\right\rangle$) and *bad*($\left|1\right\rangle$) outcomes. The states $\left|\psi_{0,1}\right\rangle_n$ can be non-orthogonal. The final state can be sampled N times to obtain an estimate $\bar{a}$ with an accuracy

$$|a - \bar{a}| \sim \mathcal{O}(N^{-1/2}), \tag{4.6}$$

as dictated by the sampling error of a multinomial distribution.

For implementing QAE it is required to construct the operator

$$\mathcal{Q} = -\mathcal{A}\mathcal{S}_0\mathcal{A}^\dagger\mathcal{S}_{\psi_0}, \tag{4.7}$$

where the operators $\mathcal{S}_0$ and $\mathcal{S}_{\psi_0}$ are inherited from Grover,

$$\mathcal{S}_0 = \mathbf{I} - 2\left|0\right\rangle_n \left\langle 0\right|_n \otimes \left|0\right\rangle\left\langle 0\right| , \tag{4.8}$$

$$\mathcal{S}_{\psi_0} = \mathbf{I} - 2\left|\psi_0\right\rangle_n \left\langle \psi_0\right|_n \otimes \left|0\right\rangle\left\langle 0\right| . \tag{4.9}$$

The $\mathcal{S}_0$ operator changes the sign of the $\left|0\right\rangle_n \left|0\right\rangle$ state, while $\mathcal{S}_{\psi_0}$ takes the role of an oracle and changes the sign of all bad outcomes.

It turns out that the eigenvalues of $\mathcal{Q}$ are $e^{\pm i2\theta_a}$, where $a = \sin^2(\theta_a/2)$. The original algorithm makes use of the QPE algorithm [NC10] to obtain the numerical values of these eigenvalues [Bra+02]. The accuracy obtained with N function calls is given by

$$|a - \bar{a}| \sim \mathcal{O}(N^{-1}) \tag{4.10}$$

with probability at least $8/\pi^2 \approx 81\%$. This approach will only be useful in the case fault tolerant computers are available. The high quality of the hardware required to perform Quantum Phase Estimation successfully prevents to use it for QAE. Further details are available in App. B.3.1.

In order to anticipate the use of QAE techniques to the NISQ era, some approaches have emerged without the Quantum Phase Estimation requirement [Gri+21; Suz+20b]. These examples are less resource-demanding. The key property that allows to circumvent the use of Quantum Phase Estimation is

$$\begin{aligned}\mathcal{Q}^m\mathcal{A}\left|0\right\rangle = \cos\left((2m+1)\theta_a\right)\left|\psi_0\right\rangle_n \left|0\right\rangle + \\ + \sin\left((2m+1)\theta_a\right)\left|\psi_1\right\rangle_n \left|1\right\rangle .\end{aligned} \tag{4.11}$$

An integer m is chosen to prepare the state here described and measure the outcome with N shots, so that the value $\sin^2((2m+1)\theta_a)$ is estimated with a precision of $\mathcal{O}(N^{-1/2})$. The process is repeated several times with different values of m defined by a set of $\{m_j\}$. Finally, all measurements are combined to extract a final estimation whose precision is bounded by $\sim N^{-1/2}M^{-1}$, with $M = \sum_{j=0}^{J} m_j$, where J is the last index. The exact scaling of the precision depends on the choice of $\{m_j\}$. Further details on the iterative method are outlined in App. B.3.2.

4.1.3 Binary algorithm

The algorithm constructed on the standard binary basis laying the groundwork for the unary algorithm here presented is introduced in Ref. [Sta+20]. The algorithm is divided in three different parts. A complete scheme is depicted in Fig. 4.1.

Amplitude distributor: this element encodes into the circuit the distribution of prices for a given asset with a certain interest rate and volatility. The operator representing piece is labeled as $\mathcal{D}$. In this algorithm, $\mathcal{D}$ is implemented using qGANs [DK18; LW18b; ZLW19]. These methods demand previous knowledge on the classical solution of the Black-Scholes model from Eq. (4.2).

Payoff calculation: the expected payoff is encoded into the amplitude of an ancillary qubit. For computing the payoff, it is only required to encode this information properly and then measure the ancilla qubit to retrieve the information. This piece is further subdivided in two more steps. First, a comparator $\mathcal{C}$ separated the prices in larger and smaller than the strike K. Then, a set of controlled rotations $\mathcal{R}$ encode the expected information into the amplitude of the ancilla qubit.

Quantum Amplitude Estimation: the ancilla qubit is measured using the QAE recipe to obtain an estimate of the expected payoff more efficiently than by classical methods. The operator $\mathcal{Q}$, which includes the $\mathcal{D}$ and $\mathcal{R}$, is applied several times.

For further detals on the binary algorithm, the reader is referred to Refs. [Ram+21; Sta+20].

4.2 The unary representation

The key ingredient of the complete unary strategy is the unary representation. In this representation, the Hilbert space corresponding to the computational framework is spanned by those quantum states with one qubit in the $|1\rangle$ state while all other qubits remain in $|0\rangle$. Thus, the

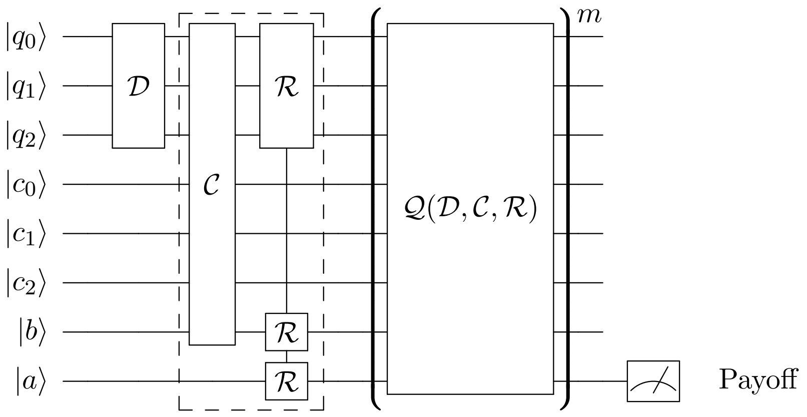


Figure 4.1: *Full circuit for the binary algorithm for option pricing that include all three steps, namely, the amplitude distributor $\mathcal{D}$, payoff estimator comprised of the comparator and payoff estimator $\mathcal{C}$ and $\mathcal{R}$ respectively, followed by components of QAE, $\mathcal{Q}$. The operator $\mathcal{Q}$ is repeated m times, where m depends on the QAE algorithm. The payoff is indirectly measured in the last qubit.*

general representation of a n-qubits quantum state in the unary basis is

$$|\psi\rangle = \sum_{i=0}^{n-1} a_i \, |i\rangle_n = \sum_{i=0}^{n-1} a_i \left(\bigotimes_{j=0}^{n-1} |\delta_{i,j}\rangle \right) =$$
$$= a_0 \, |00\ldots01\rangle_n + a_1 \, |00\ldots10\rangle_n + \ldots + a_{n-2} \, |01\ldots00\rangle_n + a_{n-1} \, |10\ldots00\rangle_n \,, \tag{4.12}$$

where $|i\rangle_n$ corresponds to the i-th element of the unary basis and $\delta_{i,j}$ is the Kronecker delta. These states satisfy the normalization $\sum_{i=0}^{n-1} |\psi_i|^2 = 1$ as well.

A well known example of a state in the unary representation is the W state defining a three-qubit multipartite entanglement class [DVC00]. The properties of the unary representation make it probably better suited to run on NISQ devices than the binary one. This claim will be detailed later. While it holds that exponentially more qubits are required to store equivalent quantum states, the unary approach is capable to retain more useful information and it is then much more robust against noise.

For a fixed number of qubits n, the unary representation only provides n different states from the computational basis, as a n-level qudit. In this sense, adding a qubit to this system is equivalent to adding a new level to an isolated qudit. The unary representation allows for a restricted Hilbert space whose number of degrees of freedom grows linearly, unlike in the standard binary representation, where the dimensionality increases exponentially. Indeed, only n out of 2^n states are taken into consideration, and the quantum states that belong to the unary representation are a restricted part of the total Hilbert space.

Since the unary representation cannot expand through the whole Hilbert space, only those quantum operations compatible with this feature are applied. Such operations are entangling gates acting on the $|01\rangle, |10\rangle$ states of two qubits, namely

$$\boxed{\mathcal{U}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \multicolumn{2}{c}{\multirow{2}{*}{$\mathcal{U}$}} & 0 \\ 0 & & & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.13}$$

where U is any arbitrary single-qubit operation. Implementing this operation among all possible pairs of qubits is enough to obtain arbitrary operations. In the case only gates between first neighbors are allowed, adding standard SWAP gates, namely Eq. (4.13) with $U = X$, is enough to implement any operation U between any pair of qubits. This property only holds if the initial state belongs to the unary basis. Many computers start in the $|0\rangle^{\otimes n}$ state, and thus adding one X gate suffices to initialize the unary representation.

This problem requires gates transporting amplitude from one qubit to the neighbor one. Those gates are the equivalent to R_y and R_x, namely partial-SWAP and partial-iSWAP gates, defined as

$$\boxed{\begin{matrix}\text{partial-}\\ \text{SWAP}(\theta)\end{matrix}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta/2) & -\sin(\theta/2) & 0 \\ 0 & \sin(\theta/2) & \cos(\theta/2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.14}$$

$$\boxed{\begin{matrix}\text{partial-}\\ \text{iSWAP}(\theta)\end{matrix}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta/2) & -i\sin(\theta/2) & 0 \\ 0 & -i\sin(\theta/2) & \cos(\theta/2) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{4.15}$$

Although the partial-iSWAP may seem more artificial for moving probability amplitudes from one state to other, this gate is more convenient than the standard partial-SWAP. This entangling gate comes naturally from the capacitive coupling of superconducting qubits [Bia+10; SS03]. As a matter of fact, Google's Sycamore chip in which the supremacy experiment was performed [Aru+19] allows for this type of gates natively. They are also of great importance for quantum chemistry applications [Bar+18; Gar+20] or combinatorial optimization [CEB20; Had+19; Wan+20b].

By correctly setting the parameters in the partial-iSWAP gates it is possible to obtain any quantum state representing a probability distribution as

$$|\psi\rangle = \sum_{i=0}^{n-1} e^{i\phi_i}\sqrt{p_i}\,|i\rangle_n\,, \tag{4.16}$$

where p_i is the probability of measuring the i-th state, and complex phases $e^{i\phi}$ are simply ignored.

The unary representation resides within a restricted and small part of the Hilbert space, and all unused space can be used as a flag for the appearence of error. The post-selection mechanism provided by the unary representation is based on a high distinguishability between those states inside and outside the unary basis. Simple measurements of the output state in the Z basis are enough to check if an error has occurred or not. All read-out must reflect one and only one $|1\rangle$, and all other $|0\rangle$s. Any failed repetition can be discarded. A trade-off between number of executions and error mitigation is obtained. The noisier a quantum computer is, the more repetitions will be ignored, in exchange of retrieving only runs where no error is detected. Notice that other error mitigation techniques are also applicable. However, the scope of this work is to focus on the native mechanism.

The properties of the unary representation make it probably better suited to run on NISQ devices than the binary one. While it holds that exponentially more qubits are required to store equivalent quantum states, the unary approach is capable to retain more useful information and it is then much more robust against noise.

4.3 Unary algorithm

The option pricing problem is addressed in this chapter by designing an algorithm relying on the unary representation. The economical tool used in this approach is the Black-Scholes model described in Sec. 4.1.1[BS73]. The distribution of asset prices is encoded into the quantum circuit in a unary state. The structure of this algorithm follows the one in Sec. 4.1.3 [Sta+20], namely amplitude distributor module, payoff computation and QAE module.

There are two main advantages of the unary representation as compared to the binary one. In terms of circuit complexity, the unary scheme allows for a significant simplification of all different pieces composing the algorithm. The unary approach brings further benefit in practice due to the native post-selection strategy that results in error mitigation. On the other hand, the unary algorithm requires more qubits than a binary one for a given precision. Both features make the unary approach adapted to run in NISQ devices.

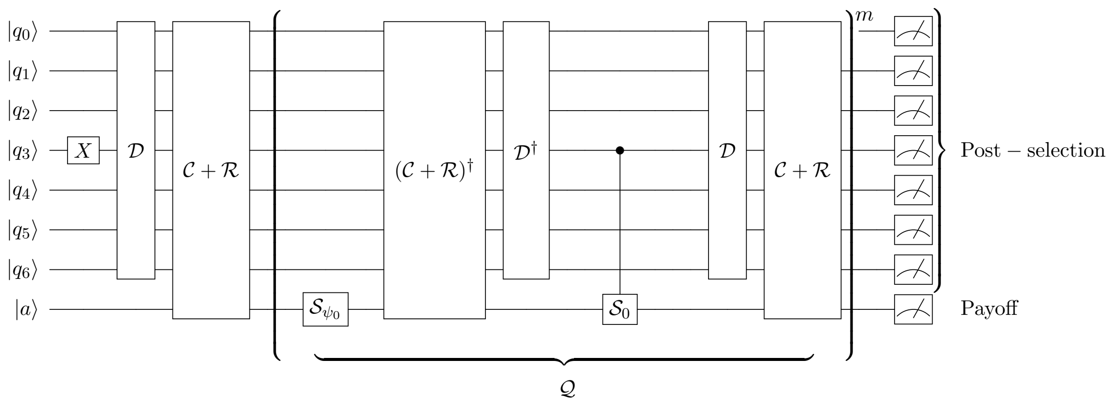


Figure 4.2: *Full circuit for the option pricing algorithm in the unary representation. The gate $\mathcal{D}$ is the probability distributor, and $\mathcal{C}+\mathcal{R}$ represent the computation of the payoff. After applying the algorithm, the oracle $\mathcal{S}_{\psi_0}$, the reverse algorithm and $\mathcal{S}_0$ follow. The last step is applying the algorithm again. This block $\mathcal{Q}$ is to be repeated for Amplitude Estimation. Measurements in all qubits is a requirement for post-selection. The qubit labelled as q_3 is the one starting the unary representation.*

4.3.1 Description of the algorithm

The global structure of the unary algorithm is inherited from the binary one, as outlined in Sec. 4.1.3. A scheme for the full circuit is depicted in Fig. 4.2. In summary, the circuit is composed by one first X gate that initializes the unary basis, one set of amplitude distributor ($\mathcal{D}$) and payoff calculator ($\mathcal{C}+\mathcal{R}$), and m rounds of Amplitude Estimation $\mathcal{Q}=\mathcal{A}\mathcal{S}_0\mathcal{A}^\dagger\mathcal{S}_{\psi_0}$. Read-out in all qubits is a requirement for post-selection to reduce errors.

Amplitude distributor $\mathcal{D}$

The starting point is the probability distribution of asset prices is based on the Black-Scholes solution from Eq. (4.2). In the unary representation, each state in the computational basis will correspond to a specified value in the price, thus the probability distribution is discretized. In particular, the qubit activated as $|1\rangle$ determines the asset value. The precision is determined by the number of qubits n in the circuit. The final price distribution at any time can be mapped to the unary representation by a fixed-depth quantum circuit. The probability of the asset to take a price is captured by the probability of measuring the corresponding qubit as $|1\rangle$. This step takes the role of a Monte Carlo spread of asset values. See Fig. 4.3 for a graphical scheme on this idea.

The circuit needed to encode the asset prices into the quantum system acts as a distributor of probability amplitudes. The initial state is set as $|00\ldots010\ldots00\rangle_n$, i. e., the active qubit in $|1\rangle$ is the middle one. To initialize the unary representation only one X gate is needed. Then,

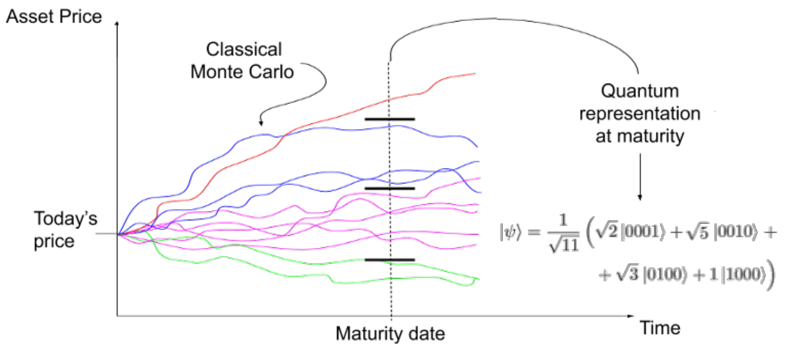

Asset Price
Classical Monte Carlo
Quantum representation at maturity
Today's price
Maturity date
Time
$|\psi\rangle = \frac{1}{\sqrt{11}}\left(\sqrt{2}\,|0001\rangle + \sqrt{5}\,|0010\rangle + \right.$
$\left. + \sqrt{3}\,|0100\rangle + 1\,|1000\rangle\right)$

Figure 4.3: *Scheme for the quantum representation of a given asset price at maturity date. For a given number of Monte Carlo paths, a binning scheme must be applied in such a way that the prices of the asset are separated according to its value. Different Monte Carlo paths that end up in the same bin are color coded accordingly. Each bin is mapped then to an element of the unary basis, whose coefficient is the number of Monte Carlo paths in this bin. The quantum representation of the asset price at maturity contains all possible Monte Carlo paths simultaneously. The precision is then bounded by the numbers of bins that can be stored on a quantum state, i. e. how many qubits are available.*

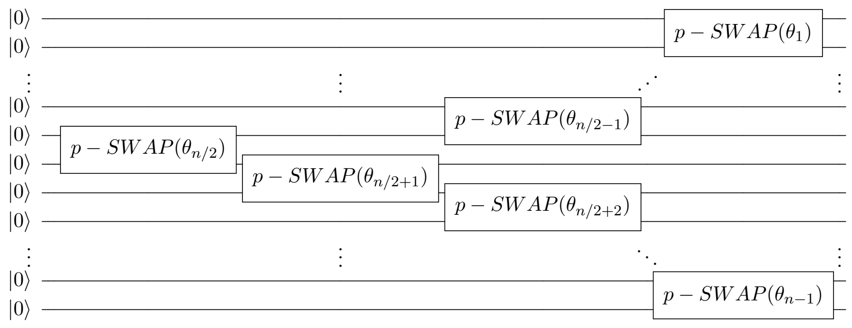


Figure 4.4: *Quantum circuit for loading any probability distribution in the unary representation $\mathcal{D}$, plus unary initalization. The circuit works as a distributor of amplitude probabilities from its middle qubit to the ones in the edges, using partial-SWAP gates that act only on nearest neighbors. Time dependence is encoded in the angles determining the gates. The first X gate is needed to start the unary representation, but it does not take part in the distribution procedure.*

coefficients in the final register encoding the asset price distribution is generated using partial-SWAP gates between the different qubits, see Eq. (4.14). Every partial-SWAP gate substracts amplitude from one state and passes it to the next one. In the first step, partial-SWAP gates connect the middle qubit and its first neighbors. Subsequent steps propagate the effect to further qubits. Since the middle qubit has a probability of being measured $p_{n/2} = 1$, this mechanism distributes the whole amplitude to the rest of asset prices far from the central value. See Fig. 4.4 for a graphical description of this procedure. Note that the partial-SWAP gate can be substituted by partial-iSWAP gates, see Eq. (4.15), for convenience when applied to experimental setups. The parameters needed to match the final asset price distribution can be exactly computed. A detailed procurement of those parameters is described in App. B.2.

Any final probability distribution in the asset prices at any time t can be obtained with the Amplitude Distributor $\mathcal{D}$. The circuit depth is independent of time. All the necessary information, including time dependency, is carried within the set of angles defining the partial-SWAP or -iSWAP gates $\{\theta_1, \theta_2, \ldots, \theta_{n-1}\}$. Given n qubits, the depth is always $\lfloor n/2 \rfloor + 1$. Similar ideas were exploited to describe the exact solution of the Ising model [Cer18; Heb+17; VCL09].

To map a known probability distribution into the unary system, exactly $(n-1)$ parameters are required. This claim holds in the cases the final probability distribution is classically computable or remains unknown. In the first case, since the final distibution is available, the quest for quantum parameters can be addressed by solving an exact set of n equations and $n-1$ variables, see App. B.2. In case only the differential equation is known, but not its solution, other methods can attempt to solve this problem [IOL07].

Payoff calculator $\mathcal{C} + \mathcal{R}$

The circuit design to compute the expected payoff acts right after the amplitude distributor $\mathcal{D}$ to encode the expected return on an ancillary qubit. In the unary algorithm, this step is significantly simpler than for the binary counterpart. The procedure attempts to prepare an entangled state of the form

$$|\Psi\rangle = \sqrt{1-a}\,|\psi_0\rangle_n\,|0\rangle + \sqrt{a}\,|\psi_1\rangle_n\,|1\rangle\,, \tag{4.17}$$

where $|\psi_{0,1}\rangle_n$ are states in a superposition of the basis elements corresponding to asset prices below (0) and above (1) the strike K respectively. The payoff is encoded in the amplitude $\sqrt{a}$, with $|a| \leq 1$. Notice that the expected return will in general not be bounded by 1, thus a re-scaling must be applied to relate the quantum procedure to the exconomical values. After this step, QAE can be applied.

In the European option case, the most relevant point is to distinguish between those prices S_i contributing to the expected return, that is prices greater than the strike K, and those that do not contribute. Only prices $S_i \geq K$ will have some effect on the ancillary qubit. This is acomplished by the $\mathcal{C}$ piece of the circuit. In the unary representation, this tasks turns out to be very simple. The computation of the expected payoff can be carried by applying single-qubit-controlled Y rotations, $cR_y(\theta)$, summarized as the $\mathcal{R}$ operator. The control qubits are those encoding a given price S_i. Only those prices above the agreed strike K will control

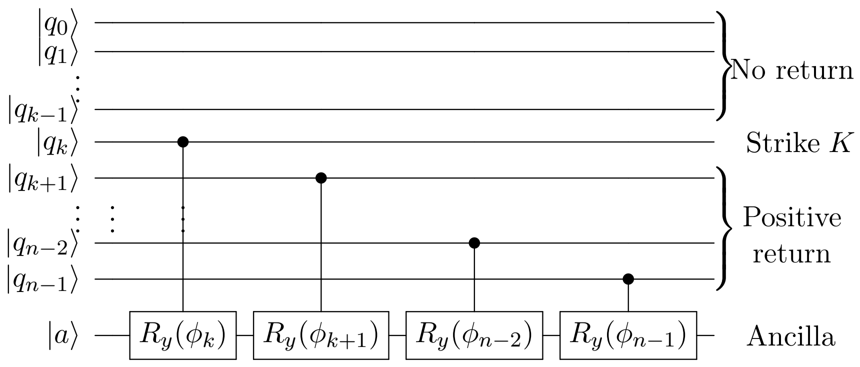


Figure 4.5: *Quantum circuit that encodes the expected payoff in an ancillary qubit in the unary representation $\mathcal{C}+\mathcal{R}$. Each qubit with a mapped option price higher than the designated strike controls a cR_y gate on the ancilla, where the rotation angle is a function of its contribution to the expected payoff. The comparator $\mathcal{C}$ is constructed through the control wires, while the $\mathcal{R}$ piece is performed by rotations in the last qubit.*

applied gates to the ancillary qubit. The depth of this circuit will be $n-k$, where k is the unary label of the strike K. See Fig. 4.5 for a graphical description of this part.

The rotation angle for each controlled rotation depends on the price represented by the control qubit. Each price contributes differently to the expected payoff. The angle will be

$$\phi_i = 2\arcsin\sqrt{\frac{S_i-K}{S_{max}-K}}, \tag{4.18}$$

where the denominator is introduced for normalization. Recall that $|a|\leq 1$. The application of the payoff calculator to a proper quantum state representing a price distribution results in

$$|\Psi\rangle = \sum_{S_i\leq K}^{n-1}\sqrt{p_i}\,|i\rangle_n|0\rangle + \sum_{S_i>K}^{n-1}\sqrt{p_i}\cos(\phi_i/2)\,|i\rangle_n|0\rangle + \\ + \sum_{S_i>K}^{n-1}\sqrt{p_i}\sqrt{\frac{S_i-K}{S_{max}-K}}\,|i\rangle_n|1\rangle\,. \tag{4.19}$$

This state is now in the form of Eq. 4.17. The probability of measuring $|1\rangle$ in the ancillary qubit is

$$P(|a\rangle) = \sum_{S-i>K} p_i\frac{S_i-K}{S_{max}-K}. \tag{4.20}$$

The encoded payoff is easily recovered after measuring the probability of obtaining $|1\rangle$ as the outcome. A simple multiplication times the normalization factor returns the quantity of interest.

Amplitude Estimation

Note that Eq. (4.19) suits the application of QAE on it. As described in Sec. 4.1.2, the key ingredient is the operator $\mathcal{Q} = \mathcal{S}_{\psi_0}\mathcal{A}^\dagger\mathcal{S}_0\mathcal{A}$. In this section a description on how to implement the operators $\mathcal{S}_{\psi_0}$ and $\mathcal{S}_0$ in the unary implementation is carried. A graphical assistance is depicted in Fig. 4.6

The oracle operator $\mathcal{S}_{\psi_0}$ acts by identifying those elements of the quantum state corresponding to accepted outcomes. This task is already performed by the algorithm $\mathcal{A}$, and the information is carried by the ancilla qubit, with $|1\rangle$ for accepted results. Thus, the function of this oracle can be achieved by performing local operations in the ancilla qubit. The required operation is

$$\mathcal{S}_{\psi_0} = (I^{\otimes n} \otimes (XZX)), \quad (4.21)$$

where the X gates could even be deleted since they add a global sign.

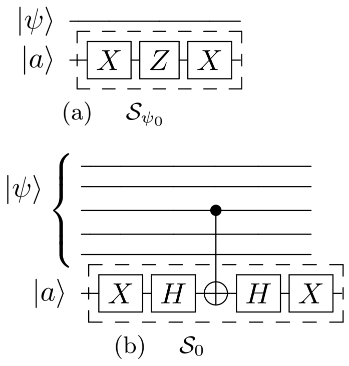


Figure 4.6: *Quantum circuit representation of $\mathcal{S}_{\psi_0}$ (a) and $\mathcal{S}_0$ (b) required to perform Amplitude Estimation in the unary basis. Notice that operator $\mathcal{S}_0$ is much simpler in the unary representation as it does not require multi-controlled CNOT gates.*

For the case of the operator $\mathcal{S}_0$ a detail that greatly simplifies this computation is remarkable . The operator $\mathcal{S}_0$ is normally defined using $|0\rangle$ since most quantum algorithms start on that state, as depicted in Eq. (4.17). However, a more apt definition should instead include a generic $|initial\rangle$ state as a basis for operator $\mathcal{S}_0$, the state onto which the algorithm $\mathcal{A}$ is first applied. For the unary case, except for the first extra X gate, it is possible to consider the algorithm as starting in that state of the unary basis, heavily simplifying the overall construction. That being the case, $\mathcal{S}_0$ can be constructed out of 2 single-qubit gates and one entangling gate. This supports a great simplification of the unary algorithm with respect to the binary one.

With the operator $\mathcal{Q}$ constructed, QAE schemes can be performed. Since the unary algorithm is aimed towards NISQ devices, it is feasible to use an Iterative Quantum Amplitude Estimation (IQAE) scheme without QPE, as mentioned in Sec. 4.1.2 and explained in detail in App. B.3. In the implementation here presented, only the options with minimal consecutive executions of $\mathcal{Q}$ are considered.

4.3.2 Error mitigation

The algorithms of the NISQ era need to present resilience against gate errors, noise and decoherence. The literature provides a variety of error mitigation techniques, see Refs. [EBL18; TBG17]. Some of those techniques might find valid applications in the unary algorithms as well. However, in this unary approach the focus lies on the native error mitigation poperties described in Sec. 4.2.

Since the algorithm is designed to perform in the unary basis, all outcomes must reflect this fact. The strategy is as simple as measuring all qubits and not only the ancilla one. This step allows to ensure that the output state did not suffer errors taking it out the unary repesentation. This way, all outcomes not belonging to the unary representation are discarded. This triggers a trade-off between number of accepted samples and reduction of errors.

4.4 Comparison between unary and binary algorithms

The unary algorithm is conceived since the beginning as an alternative approach to the binary one with some properties that make it suit better for implementation in NISQ devices. In this section a comparison the resource demands of both algorithms in terms of circuit design and number of gates is treated. The analysis includes the proper algorithm and the QAE procedure. A detailed treatment of errors is then left for subsequent sections.

4.4.1 Gate count

In order to properly count the number of gates required for executing the algorithm, some particular choices have been made. In practice, quantum computers make use of a native set of gates with the capability to construct any unitary with some overhead in the number of operations. The count is carried taking CNOT and partial-iSWAP gates as the native entangling gates. A chip connectivity between all qubits with theoretical common operations, see Sec. 4.4.2 for further details, is also assumed for the sake of simplicity. The overhead of extra SWAP gates to account for non-existing connections is not considered in these calculations. In all cases, the counting for single-qubit gates is made by compiling subsequent gates into a single one. All two-qubit gates are decomposed into the native entangling gate and a number of single-qubit gates, counting for all possible overheads. This is particularly prominent in the binary case where many Toffoli gates are needed.

The unary algorithm requires $\mathcal{O}(n)$ partial-SWAP gates for accomplishing the amplitude distributor $\mathcal{D}$. For computing the payoff, $\mathcal{O}(\kappa n)$ controlled-rotation gates are needed, where $0 \leq \kappa \leq 1$ stands for the qubit corresponding to the strike price K. The results from Table 4.1, left, collects the gate counting of the full circuit for both the unary and binary algorithms, as a function of the number of qubits. for CNOT and

Unary	CNOT				partial-iSWAP			
	$\mathcal{D}$	$\mathcal{C}+\mathcal{R}$	$\mathcal{S}_{\psi_0}$	$\mathcal{S}_0$	$\mathcal{D}$	$\mathcal{C}+\mathcal{R}$	$\mathcal{S}_{\psi_0}$	$\mathcal{S}_0$
1-qubit gates	2n	2κn	1	4	1	κ10n	1	9
2-qubit gates	4n	2κn	0	1	n	κ5n	0	2
Circuit depth	3n	4κn	1	5	n/2	15κn	1	10

Binary	CNOT				partial-iSWAP			
	$\mathcal{D}$	$\mathcal{C}+\mathcal{R}$	$\mathcal{S}_{\psi_0}$	$\mathcal{S}_0$	$\mathcal{D}$	$\mathcal{C}+\mathcal{R}$	$\mathcal{S}_{\psi_0}$	$\mathcal{S}_0$
1 qubit gates	3nl	(16+5κ)n	1	20n - 23	8nl	(86+5κ)n	1	80n - 113
2 qubit gates	nl	14n	0	12n - 18	2nl	28n	0	24n - 36
Circuit depth	nl+1	(27+2κ)n	1	24n - 30	6nl+1	(97+2κ)n	1	90n - 129

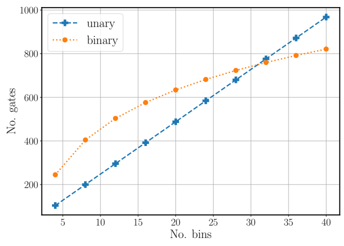
Table 4.1: *Scaling of the number of 1- and 2-qubit gates and circuit depth as a function of the number of qubits n representing the asset value in unary and binary representations, for the amplitude distributor $\mathcal{D}$, payoff estimator $\mathcal{C}+\mathcal{R}$ and QAE operators $\mathcal{S}_{\psi_0}$ and $\mathcal{S}_0$. Ideal chips architectures are assumed. The scalings in case CNOT or partial-iSWAP gates are implemented are compared. In case the experimental device can implement both CNOT and partial-iSWAP basic gates, the total amount of gates and total depth would be reduced. For the unary circuit, the parameter $0 \le \kappa \le 1$ depends on the position of the strike in the qubit register. The parameter $0 \le \kappa \le 1$ characterizes the number of 1s in the binary representation of the strike price. For the amplitude distributor, l is the number of layers of the qGAN.*

partial-iSWAP gates as the native entangling gates.

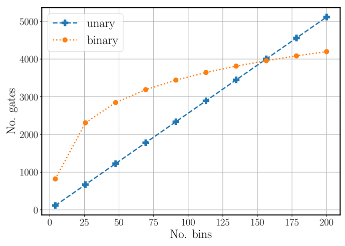
The amplitude distribution module $\mathcal{D}$ substantially benefits from having partial-iSWAP gates as the native operation. However, this implies an overhead for the payoff computation $\mathcal{C}+\mathcal{R}$, where CNOT gates introduce a gain. In the ideal case where both partial-iSWAP interactions between first neighbors and CNOT-based connections to the ancilla qubit are available, the best possible scaling is obtained. Explicitly, the total number of gates would be $(4\kappa+1)n+1$, and the depth of the circuit would become $(4\kappa+\frac{1}{2})n$.

The gate count for the binary algorithm is summarized in Tab. 4.1, right, in the same conditions as the unary one. The CNOT-connection turns out to be more convenient in this case. The results here provided include also the qGAN piece used for uploading the asset price distribution into the quantum circuit, thus a dependency on the number of layers emerges. However, no training cost required for qGANs is considered.

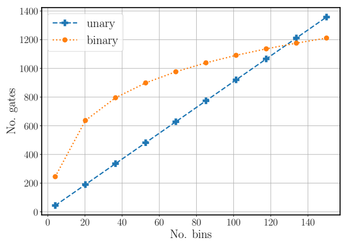
It is worth emphasizing that the gate overhead for the unary algorithm is much lower than for the binary case. The main reason for this result is that the unary circuit does not implement any three-qubit gate, which is only decomposable in a large number of two-qubit gates. This simplification is overcome by the exponential advantage of the binary algorithm for large numbers of n, provided that the asset prices distribution is efficiently uploaded.



(a) CNOT gates



(b) partial-iSWAP gates



(c) Best combination

Figure 4.7: *Scaling of the number of gates required for the full algorithm, including a step, $m = 1$, of Amplitude Estimation, with the number of bins, for different native gates: CNOT gates (a), partial-iSWAP gates (b) and the best possible combination (c), in which one is allowed both CNOT and iSWAP gates as native to the device. The scaling is calculated assuming ideal connectivity, which would largely hinder the binary implementation were that not the case.*

A complete comparison between unary and binary circuits is shown in Fig. 4.7. In this comparison, the numbers $\kappa = 1/2$, number of controlled rotations in the unary algorithm, and $l = \log_n(2)/2$, number of layers in the qGAN for the binary case, are fixed. The comparison is made for the same precision in the asset price representation. For a given number of n bins, the unary algorithm needs n qubits, thile the binary representation has enough with only $\log_2(n)$ of them, plus qubits overhead. The simple operations of the unary algorithm make this option mode convenient for a number of bins $n \sim 100$. For large values of n, the binary approach outperforms the unary option, as the number of gates is logarithmically lower. In case quantum resources have limited numbers and quality, as it is expected in NISQ devices, circumstances favor the unary approach to the detriment of the binary one.

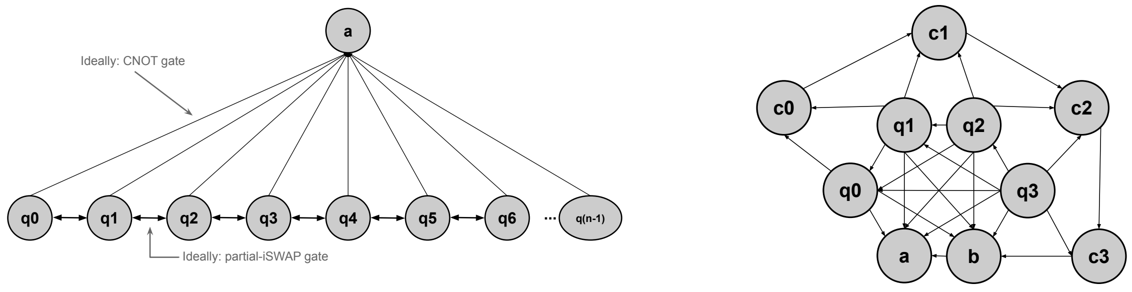
4.4.2 Ideal chip architecture

From the theoretical description of the algorithms described in Sec. 4.3 for the unary algorithm and Ref. [Sta+20] for the binary one it is possible to infer the connectivity requirements. Those are described in this section, where the condition is that any pair of qubits with a common interaction in theory has direct connection in the chip.

The unary algorithm can be performed with a very simple chip. The amplitude distribution module $\mathcal{D}$ only needs local interaction between first-neighbor qubits to upload the asset prices to the quantum register. Thus, the qubits can be arranged on a 1D line with two-local interactions. In addition, if this connection carries a partial-SWAP native gate, the $\mathcal{D}$ operations can be performed without any overhead. This realization of the quantum circuit would result in a decrease in the number of needed gates by factor of 6 in the amplitude distributor module as compared to the CNOT entangling gate. Note also that superconducting qubits allow for a natural implementation of the partial-iSWAP gate [Bia+10]. For the expected payoff, the ancillary qubit interacts with any other qubit from the quantum register, ideally relying on CNOT gates.

On the other hand, the binary algorithm for payoff calculation needs a much more complex chip connectivity. For the sake of comparison with the simplest chip architecture presented for the unary algorithm, the most basic connectivity needed to perform the steps described for the binary scheme is displayed in Fig. 4.8. In the binary case, $\log_2(n)$ qubits (q in the figure) are needed to store the price distribution with the same accuracy as the unary case with n qubits. A total amount of $\log_2(n)+1$ auxiliary ancilla qubits are needed (in the figure c and b). They payoff is stored at another ancilla a. A total number of $2(\log_2(n)+1)$ qubits is needed.

It is clear that the number of necessary qubits for the binary algorithm, including ancillas, scales asymptotically better than in the unary approach. Nevertheless, the need for Toffoli gates and almost full connectivity may eliminate this advantage in practical problems for NISQ devices. The simplicity of the architecture needed to implement the unary algorithm might yield an advantage over alternative algorithms as well.



(a) Unary algorithm (b) Binary algorithm

Figure 4.8: *Ideal chip architecture to implement the unary and binary algorithms for option pricing. a) In the unary chip, only a single ancilla qubit, labelled as* a *in the image, has to be non-locally controlled by the rest of the qubits. All other interactions are first-nearest-neighbor gates. b) For the binary implementation with 4 qubits of precision,* q_0,q_1,q_2,q_3, *where* a *and* c *stand for ancillary and carrier qubit, respectively, and* b *is another ancilla. The algorithm requires a number of ancillary and carrier qubits equal to the number of precision qubits plus two, 4+2 in this example. Full connectivity is needed between the precision qubits and two ancillas.*

4.5 Results

In this section the binary and unary algorithms for a given option pricing problem are simulated. The aim is to compare the performance of both approaches. Two main comparisons are made relating this topic. First, the performances are tested for circuits without any noise, where the only source of error comes from the sampling uncertainty at the measurement step. In a second step, both approaches are tested against increasing noise levels in order to verify the resilience to error of both unary and binary algorithms. Both steps are presented as the same process. The final target of these calculations is to discern which approach is more beneficial for NISQ computers. The calculations were carried using the `qiskit` [Ale+19a] framework. The code is publicly available in Ref. [RP20].

The financial problem addressed in this example is a standard European option with fixed properties. The asset price at $T = 0$ is $S_0 = 2$, its interest rate is $r = 0.05$, and its volatility is $\sigma = 0.4$. The maturity time is $T = 0.1$ years. The agreed strike price is $K = 1.9$. The simulation of the asset price is carried taking the Black-Scholes model from Sec. 4.1.1 up to three standard deviations in the log-normal probability distribution. For the quantum circuits simulation, a 8 bins model is considered. Thus, the unary algorithm encodes the asset price distribution in 8 qubits, and the binary one in $\log_2(8) = 3$ of them. Ideal chip structured from Sec. 4.4.2 are considered. In terms of payoff calculation, the reference value is computed classically, with a precision of 10^4 bins.

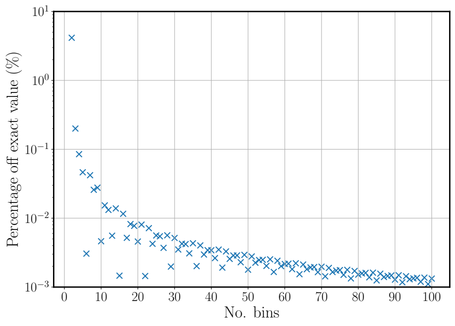


Figure 4.9: *Percentage error from the exact value of the expected payoff, for the classical computation, as a function of the number of bins in the probability distribution. With only ~ 50 bins, errors for the option price below 0.5% are already reached.*

First of all, it is important to estimate the range of applicability of the unary and binary algorithms as a function of the number of bins n. From the calculations presented in Sec. 4.4.1, it is possible to conclude that the crossing point in the number of gates lies at $n \sim 100$. That is, below this threshold, the unary algorithm needs less gates than the binary one to be executed. Real-world applications require at least an accuracy $< 1\%$. The question to answer here is whether this accuracy can be reached in the range of applicability of the unary or the binary algorithm.

The error of the expected payoff as a function of the number of bins n is plotted in Fig. 4.9. The error depends both in the binning n and the position of the strike K, that is, if K lies at the center or the extrema of a bin. The error will decrease in general as n increases. Therefore, the results fall within a reasonable accuracy around the exact value for a sufficiently large number of bins. The results in Fig. 4.9 show that ~ 50 bins are enough to achieve accuracies near 0.5%. This regimes corresponds to an advantage for the unary approach. This shows that the unary algorith can be implemented and obtain better performance than the binary one, and still return accurate results with small discretization errors.

The noise maps used in the simulations to model the noise are simple yet descriptive. The complete model is controlled by a tunable parameters ε such that it vanishes for perfect circuits, and reaches $\varepsilon = 1$ for a random execution. For single- and two-qubit gate errors, a depolarizing noise is considered. This noise is described by the transformation

$$\rho \rightarrow (1-\varepsilon)\rho + \frac{\varepsilon}{d}\,\mathrm{Tr}(\rho) I, \tag{4.22}$$

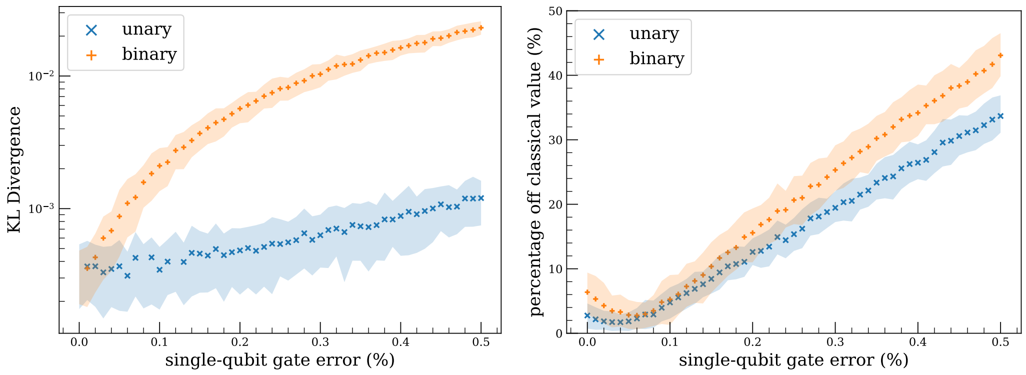
where I is the identity gate of dimension d. The depolarizing transformation occurs after each gate, with the difference that for two-qubit gates the error grows up to 2ε. For measurement errors, the probability of obtaining a wrong outcome is modeled as 10ε and symmetric, that is measuring incorrect $|0\rangle$ or $|1\rangle$ is equally probable. It is remarkable that no thermal relaxation nor thermal dephasing have been included into the noise models. The reason is that the execution times for the considered circuits are far below coherence times of qubits due to the shallow depth of the circuits. The execution time of a single-qubit gate is ~ 1000 shorter than the decoherence time. This description is adjusted to a simplified version of state-of-the-art computers [Aru+19].

Amplitude Distribution loading - $\mathcal{D}$

The capabilities of the unary and binary amplitude distribution modules are compared in Fig. 4.10(a). The quantity here depicted is the Kullback-Leibler divergence [KL51]. This quantity measures the distance between two probability distributions, and it vanishes when both are indistinguishable. It is clear to see that the approximation of $\mathcal{D}$ for the unary algorithm achieves better results than the binary distributor. For the maximum amount of noise allowed, the difference rises up to an order of magnitude.

Expected Payoff - $\mathcal{C} + \mathcal{R}$

It is shown in Fig. 4.10(b) the average error of the expected payoff as computed with the unary and binary approaches, when compared to the classical value. The unary algorithm presents slightly better results than the binary one for all errors considered. This difference is too



(a) Amplitude distribution $\mathcal{D}$ (b) Expected payoff calculation $\mathcal{C} + \mathcal{R}$

Figure 4.10: *a) Kullback-Leibler divergence between the target probability distribution and those achieved by the quantum algorithms, b) percentage error in the payoff calculation for depolarizing and measurement errors. Calculations made for equivalent 8 unary and 3 binary qubits for depolarizing and measurement errors (only in b)), up to 0.5% for single-qubit gates, 1% for two qubit gates and 5% for read-out errors, consistent with state-of-the-art devices. Crosses stand for average results, and the shaded regions encompass the central* 70% *of the instances. Each probability distribution is estimated using* 100 *experiments with* 10^4 *samples each. The shaded regions encompass the central 70% of the instances in each case. The unary algorithm is more robust against these errors.*

small to conclude that the unary approach is more convenient. Note that the deviations in the expected payoff reach a minimum for a finite value in the error parameter $\varepsilon \sim 0.005\%$. This is easy to understand by considering the theoretical expected value. Both algorithms return low approximations to the expected value, with the values of a from Eq. (4.19) $a < 0.5$. As the noise is considered, the value of a tends to its random value $a = 0.5$. Thus, there is a middle point where a corresponds to an accurate approximation of the expected value.

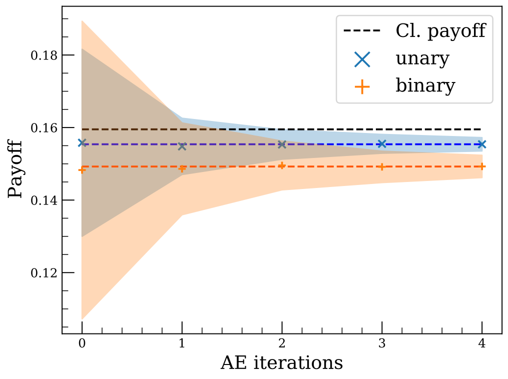
Quantum Amplitude Estimation

In this QAE part, results will be presented in three steps. First, instances without any noise considered are shown. The noiseless devices present convergent results within errors due to approximations. Then, an analysis of the errors and the statistical uncertainty of the expected payoff value when applying QAE with errors is performed. In this case, an extension to larger numbers of bins n is carried for the unary algorithm.

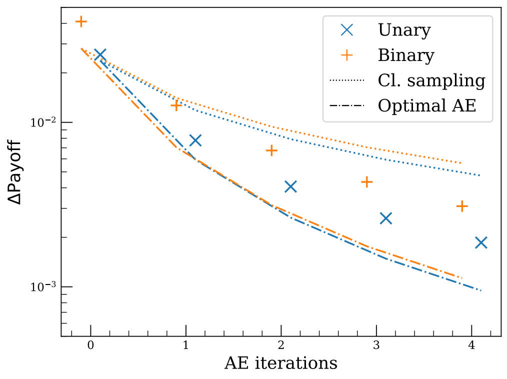
Only QAE without QPE can be performed on NISQ devices. In these simulations, a procedure based on weighted averages that consider both mean values and uncertainties is used, for a given series of QAE steps, see App. B.3.2 for further details. In our results, every instance has been repeated 100 times. The choice of m_j is linear, $m_j = j$, with $j = \{0, 1, 2, \ldots\}$, in order to control how the performance evolves. The confidence level was adjusted to $1 - \alpha = 0.95$.

Figure 4.11 shows the increasing accuracy of the expected payoff using a QAE recipe as more iterations are utilized. These results confirm that QAE reduces the statistical uncertainty of the final results as more sophisticated circuits, that is with more iterations, are considered.

Interesting results arise when considering the robustness against noise, in particular in this example depolarizing and read-out errors, of both unary and binary algorithms. The results in the deviation of the expected payoff with respect to the ideal case is depicted in Fig. 4.12. The number of QAE iterations was limited to 4 due to the computational cost of each simulation. Unary and binary algorithms show very different behaviors. First, the unary case endures the application of QAE with $M = \{0, 1, 2, 3, 4\}$ iterations when the noise levels are moderate. The errors in the expected payoff reach a 60% for the maximum noise level allowed. The worsening of the results is gradual. Take, for instance $M = 2$. Results with low errors are obtained up to error rates $\varepsilon \sim 0.3\%$.

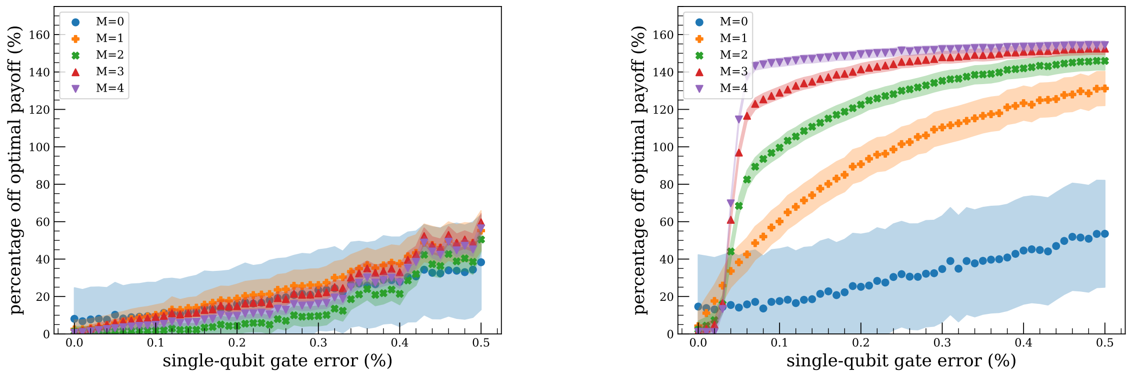


(a) Expected payoff, QAE for noiseless devices



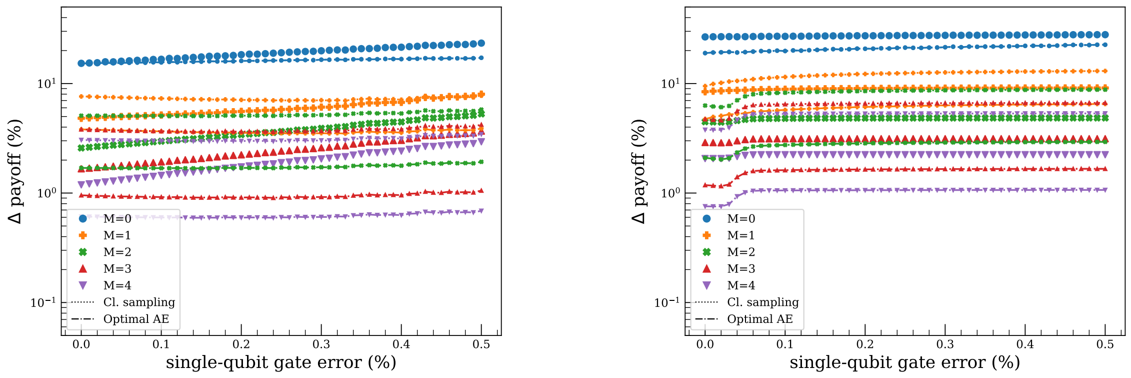
(b) Expected payoff, QAE for noiseless devices

Figure 4.11: *a) Mean and uncertainty of the outcomes of the expected payoff, details can be found at App. B. The dashed lines indicate the exact values. Unary and binary approaches are depicted, and convergence to the optimal values are obtained for both. Notice that these values are not the same since the outcomes of both algorithms are not equally related to the payoff due to bining. The shaded regions correspond to the statistical uncertainty. b) Statistical uncertainties in the expected payoff. The dotted lines indicate the uncertainty given by classical sampling, while the dot-dashed lines represent the optimal uncertainty provided by Amplitude Estimation. Results of the simulations lie in between. In this figure procedures with the same number of applications of the $\mathcal{A}$ or $\mathcal{A}^\dagger$ operators, for noiseless circuits, are compared.*



(a) Unary (b) Binary

Figure 4.12: *Results of the errors in the expected payoff respect to the optimal value, for the unary (a) and binary (b) representation, with M iterations of QAE considering depolarizing and read-out errors together. Scattering points stand for average values, while the shaded region corresponds to the statistical uncertainties. In the unary case, the expected payoff is resilient to errors, while the binary approach returns acceptable results only for $M = 0$, while $M \geq 1$ rapidly saturates to a random circuit.*



(a) Unary (b) Binary

Figure 4.13: *Results of the sampling uncertainties of the expected payoff, same conditions as above. Scattering points represent the obtained uncertainties while dash-point lines represent theoretical bounds. For every color and symbol, the lower bound is for optimal QAE, and the upper bound is for sampling. In every case, each iteration of QAE reduces the uncertainty. For the unary case, the scattering points tend to return larger uncertainties as the errors increase, while for the binary case the uncertainties remain approximately constant. This difference is a direct consequence of the reduction of valid samples triggered by post-selection.*

Beyond this threshold, the returns become slightly more erratic. The results from the binary algorithm are totally different. When no QAE step is taken, $M = 0$, the results are comparable to the unary ones, see for instance Fig. 4.10. However, the accuracy disappears completely at the first iteration $M \geq 1$ with small levels of noise $\varepsilon \sim 0.04\%$. A regimen of saturation is immediately reached in the binary case. This stationary regime corresponds to the random circuit where $a = 0.5$. The differences between both behaviors can be attributed to the post-selection regime resulting in a native mitigation of errors. The simpler structure of the unary circuit plays also a role.

It is also interesting to study the evolution of the uncertainty in the expected payoff calculation as more iterations of QAE are introduced into the circuit. This kind of errors is an exclusive consequence of the sampling uncertainty in the measurement step, which cannot be avoided. This can be observed in Fig. 4.13, where the obtained uncertainties are bounded between the classical sampling and the optimal QAE.

There appears a very remarkable behavior of the uncertainties in the unary approach to be noticed. The obtained uncertainties present a tendency to increase as errors get larger, unlike in the binary algorithm that does not present this feature. The reason lies in the native post-selection procedure only applicable in the unary representation. As errors become more likely to happen, the post-selection filter rejects more instances. The direct consequence is that the number of accepted shots drops for large errors, causing less certain outcomes. The joint action of this processes is that the uncertainty decreases more slowly for the unary algorithm than for the binary one. This behaviour contrasts with the error obtained in Fig. 4.12, where the binary results reflect a poor performance.

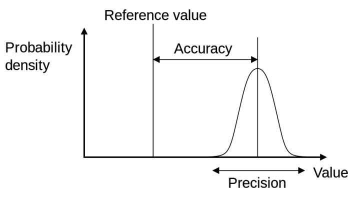
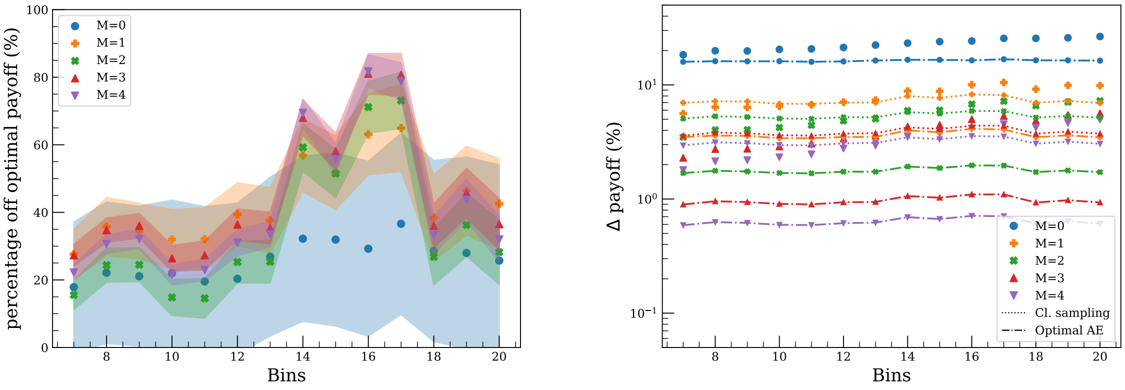


Figure 4.14: *Graphical explanation for the difference between accuracy and precision [Pek07].*

The apparently contradictory result is related to the distinction between *accuracy* and *precision*. *Accuracy* stands for how close is a measurement to the exact value of a quantity, and *precision* encodes the dispersion of different measurements, see Fig. 4.14. QAE is an algorithm to increase the precision of a measurement with respect to the number of samples, but it does not provide any further information regarding the accuracy. Indeed, QAE for the binary algorithm reflects the expected tendency for the increase in precision, but comes with very poor results in accuracy. The unary algorithm grows slower in terms of precision, but maintains more accurate results. This decrease in precision might lead to losing the quantum advantage provided by QAE in the presence of significant error. In App. B.3.2 the limit of QAE iterations that can be performed given the error rates of the quantum device while still maintaining quantum advantage for the unary representation is further studied.

The results presented up to this moment support the use of QAE procedures even in NISQ devices in the unary representation. The resilience against noise shown by this approach is greater than in the binary algorithm. The noise here considered must be moderate to retrieve useful information from the calculation. However, the noise levels here considered are compatible with state-of-the-art computer [Aru+19].

As a last step, the results of the unary algorithm are extended to larger number of bins $8 \leq n \leq 20$. Figure 4.15 show the deviation in the expected payoff and the sampling uncertainty for many different numbers n with a noise level fixed at $\varepsilon = 0.3\%$ for depolarizing and measurement errors. The purpose of this calculation is to extract the behavior of the deviation in the payoff as more qubits and complex circuits are taken into account. It is clearly seen that the errors increase with the number of bins n, as expected. Larger systems imply more gates, and thus



(a) Errors of expected payoff (b) Sampling uncertainties of expected payoff

Figure 4.15: *Results for the error and sampling uncertainties of the expected payoff for increasing number of bins for up to M iterations of Amplitude Estimation for the unary approach, considering depolarizing and read-out errors together. For the error in payoff, scattering points represent the mean values obtained for the experiment, while shadowed areas include 70% of the instances. In the sampling uncertainties, scattering points represent uncertainties obtained and dash-point lines represent theoretical bounds, where each line is accompanied with the corresponding marker. For every color and symbol, the lower bound is for optimal quantum advantage, and the upper bound is for sampling. The noise lvel is fixed to $\varepsilon = 0.3\%$. Each experiment is repeated only 10 times to reduce computational costs.*

the errors are more likely to appear. In particular, for between 13 and 18 qubits, a larger error is observed. It is expected that this behavior encounters a completely random regime for a sufficiently large number of gates, although this regime is not observed. In contradistinction, the binary algorithm finds this situation at early stages. As in previous results, the increasing uncertainty with the number of bins n reflect the more measurements rejected due to larger error probabilities. A slower convergence is the direct consequence.

4.6 Conclusions

This chapter has explored the strategy of encoding information in a quantum state using a low level of compression. In particular, the unary representation here presented utilizes only those state in the computational basis with only one $|1\rangle$ among all qubits, while all others are $|0\rangle$. This choice is not unique, but it is representative of the idea of dilluting the information across a large Hilbert space. The unary approach brings a clear advantage with respect to standard algorithms, namely the simplification achieved with the unary representation make the computation easier to execute and more resilient to noise and decoherence. It is even possible to reach quantum advantage in this regime. On the other hand, the storage capability of the quantum state is exponentially reduced.

The range of applicability of the unary representation is a NISQ regime with few and noisy qubits, at a middle stage between state-of-the-art current and fault-tolerant computers. In case the unary representation returns a simplified circuits easier to implement in a quantum computer for small numbers of qubits, the execution is more robust against noise and more profitable. This situation cannot be maintained for large number of qubits since the exponential capabilities of standard algorithms overcome any other feature of the algorithm.

The unary representation is tested for solving the financial problem of European option pricing. The economical problem is solved by means of the celebrated Black-Scholes models for the evolution of stochastic prices. The accuracies required in finance to solve this problem usefully are about $< 1\%$. These values are compatible with developing a unary algorithm in a regime with few qubits. The problem is solved in three steps, namely uploading of a probability distribution of prices, computation of the expected return and an iterative QAE procedure. Each piece makes explicit use of the unary representation to simplify its implementation.

For uploading the distribution of prices at the maturity date when the option expires, the unary representation uses a circuit where the only operations are partial-SWAP gates. The parameters controlling these gates can be found by classical means if the classical distribution is known, which is guaranteed in this problem. The circuit has a linear depth with the number of qubits. For the computation of the expected return, all steps can be taken with controlled-rotation gates at most. The number of operations is at most equal to the number of qubits. The quantity of interest is transported to an ancillary qubit. In the iterative QAE step, the required operators offer a much simpler implementation, allowing for a general reduction in the algorithmic complexity. The overall simplification comes in terms of number of gates and required connectivity to accomplich all operations.

The unary representation permits a native post-selection method that results in a strong error mitigation. Since the unary algorithm resides in a restricted region of the Hilbert space, the outcomes must reflect this property. Any outcome with zero or more than one qubit in the $|1\rangle$ state is automatically rejected. This ability reduces the acceptance of erroneous outcomes mitigating errors and increasing the performance of the quantum algorithm. In exchange, the effective number of measurements is reduced due to the active filter.

The use of QAE pieces triggers the presence of quantum advantage by substituting Monte Carlo methods with its quantum analogues. This advantage could even be checked on state-of-the-art quantum computers. Experimentally, the advantage is exclusive of the unary algorithm since the binary one presents so erratic results that no useful quantity can be extracted. In summary, QAE on unary representation allows to maintain robust results at the price of reducing the convergence rate. Nevertheless, the attainment of quantum advantage is still feasible.

The results presented through this chapter entails that using quantum algorithms with a dilluted encoding of information may present advantages in the NISQ era with respect to the standard dense algorithms. The advantages are essentially a greater resilience to noise and a greater resilience to noise and simple implementation of operations. The advantage can only be obtained in a regime with few qubits. In some particular cases, like the one here presented, a dilluted representation can bring quantum advantage even for real-world applications.

5. Final remarks

It is better to be lucky. But I would rather be exact. Then when luck comes you are ready.
Ernest Hemingway

This thesis covers two different strategies related to seizing quantum computers during the NISQ era, known as re-uploading and unary strategies. The aim of both strategies is to take profit of two purely quantum properties that settle the difference between quantum and classical computing: superposition (for re-uploading) and entanglement (for unary).

The re-uploading strategy is developed along Ch. 3. Re-uploading is a general technique to bring the fields of Machine Learning (ML) and quantum computing together. The differential element of the re-uploading strategy with respect to other Quantum Machine Learning (QML) techniques is that data must be introduced sequentially and several times in the quantum system. Data is accompanied by some classical parameters that can be optimized to make the quantum circuit behave as desired. Thus, the re-uploading strategy is a hybrid quantum-classical Variational Quantum Algorithms (VQA). This procedure permits to load and process data in the same step, unlike in most QML examples, where both steps are separate. One important property of the re-uploading strategy is that non-linearities, a requirement in ML to ensure the solving capabilities of a given model, emerge naturally form the quantum-mechanical properties of the system. It is demonstrated in this thesis that the re-uploading model is formally equivalent to other classical ML models.

From a practical perspective, it is shown that the re-uploading strategy performs successfully when facing a variety of problems on regression and classification of data. First, classical simulations of the quantum method were attempted, to pave the way towards experimental implementations. For simple problems suiting small computers, experiments on superconducting and ion-trap qubits were satisfactorily performed, while problems requiring larger computers do not return meaningful results.

The unary strategy consists in using only some of the computational states available in a complete Hilbert space to implement an algorithm, in this case to solve the financial problem of option pricing. Reducing the available space translates into a losing of asymptotical performance, but as a trade-off a great resilience against noise is obtained. This robustness is achieved thanks to a simplification of the circuits with respect to other standard algorithms, and to a post-selection mechanism that allows to easily detect corrupted outcomes. In all steps of the calculation, all qubits of the quantum state are entangled to all others.

The aim of the unary algorithm is to be useful during the first stage of the NISQ era. A quantum advantage is obtained in the algorithm here presented, not as prominent as with other standard methods, but maintaining the usability of final outputs. The trade-off between performance and resilience against noise brings an advantage for the unary algorithm, at least for those problems whose size requirements do not exceed the range of advantage. This situation is fulfilled in the case of NISQ computers for the option pricing problem of interest with acceptable precision range.

Both strategies are in a position to be implemented on current or near-future quantum devices, thus contributing with useful recipes to the status of NISQ computing. There are still many open research lines in both strategies to improve the performance and applicability of both methods. For example, the re-uploading strategy could benefit of efficient training procedures and specific embeddings to upload data, while the unary strategy could still explore further error mitigation and detailed differences between experimental implementations of the recipe here presented and other standard methods taken as reference. Nevertheless, the novel proposals here described will hopefully help to begin the era of a profitable employment of NISQ devices, as soon as the hardware is ready to be used.

Appendices

A. Technical appendices for re-uploading

A.1 Classical UAT for complex functions

The standard formulation of the UAT supports the approximation of complex function using $e^{i(\cdot)}$ as the activation function.

The approximations according to the UAT of the function are followed

$$z(\vec{x}) = a(\vec{x}) + ib(\vec{x}), \tag{A.1}$$

using trigonometric functions as $\sigma(\cdot)$,

$$a(x) = \sum_{j=1}^{N} \alpha_i \cos(\vec{w}_j \cdot \vec{x} + a_j) \tag{A.2}$$

$$b(x) = \sum_{j=1}^{N} \beta_i \sin(\vec{v}_j \cdot \vec{x} + b_j). \tag{A.3}$$

Then

$$z(x) = \sum_{j=1}^{N} \alpha_i \cos(\vec{w}_j \cdot \vec{x} + a_j) + i \sum_{j=1}^{N} \beta_i \sin(\vec{v}_j \cdot \vec{x} + b_j), \tag{A.4}$$

and this equation is can be rearranged as

$$z(x) = \sum_{j=1}^{N} \frac{\alpha_j}{2} \left(e^{i(\vec{w}_j \cdot \vec{x} + a_j)} + e^{-i(\vec{w}_j \cdot \vec{x} + a_j)} \right) + \frac{\beta_j}{2} \left(e^{i(\vec{v}_j \cdot \vec{x} + b_j)} - e^{-i(\vec{v}_j \cdot \vec{x} + b_j)} \right), \tag{A.5}$$

what encourages the UAT formulation for complex functions as an analogous to Eq. (3.21)

$$G(\vec{x}) = \sum_{n=1}^{N} \gamma_n e^{i\delta_n} e^{i\vec{u}_n \cdot \vec{x}}. \tag{A.6}$$

A.2 Mathematical theorems for proving quantum UAT

Theorem A.2.1 : Hahn-Banach [Ban29; Hah27]
Set $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$. Let V be a $\mathbb{K}-$vector space with a seminorm $p : V \to \mathbb{R}$. If $\varphi : U \to \mathbb{K}$ is a $\mathbb{K}-$linear functional on a $\mathbb{K}-$linear subspace $U \subset V$ such that

$$|\varphi(x)| \leq p(x) \qquad \forall x \in U, \tag{A.7}$$

then there exists a linear extension $\psi : V \to \mathbb{K}$ of φ to the whole space V such that

$$\psi(x) = \varphi(x) \qquad \forall x \in U \tag{A.8}$$

$$|\psi(x)| \leq p(x) \qquad \forall x \in V \tag{A.9}$$

Theorem A.2.2 : Riesz Representation [Rie14]
Let X be a locally compact Hausdorff space. For any positive linear functional ψ on $C(X)$, there exists a uniruq regular Borel measure μ such that

$$\forall f \in C_c(X) : \qquad \psi(f) = \int_X f(x) d\mu(x) \tag{A.10}$$

Theorem A.2.3 : Lebesgue Bounded Convergence [Wei74]
Let $\{f_n\}$ be a sequence of complex-valued measurable functions on a measure space (S, Σ, μ). Suppose that $\{f_n\}$ converges pointwise to a function f and is dominated by some integrable function $g(x)$ in the sense

$$|f_n(x)| \leq g(x), \qquad \int_S |g| d\mu < \infty \tag{A.11}$$

then

$$\lim_{n\to\infty} \int_S f_n d\mu = \int_S f d\mu \tag{A.12}$$

A.3 Definitions of 2D functions for universality

The definitions used for the 2-dimensional functions [Ard16] that serve for benchmarking the proposed algorithms are defined as

$$\text{Himmelblau}(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2, \tag{A.13}$$

$$\text{Brent}(x,y) = \left(\frac{x}{2}\right)^2 + \left(\frac{y}{2}\right)^2 + e^{-\left(\left(\frac{x}{2}-5\right)^2 + \left(\frac{y}{2}-5\right)^2\right)}, \tag{A.14}$$

$$\begin{aligned}\text{Threehump}(x,y) = 2\left(\frac{2x}{5}\right)^2 - 1.05\left(\frac{2x}{5}\right)^4 + \frac{1}{6}\left(\frac{2x}{5}\right)^6 + \\ + \left(\frac{2x}{5}\right)\left(\frac{2y}{5}\right) + \left(\frac{2y}{5}\right)^2,\end{aligned} \tag{A.15}$$

$$\text{Adjiman}(x,y) = \cos(x)\sin(y) - \frac{x}{y^2+1}, \tag{A.16}$$

where a normalization to $-1 \leq f(x,y) \leq 1$ is applied after this definition. A graphical representation of these functions is depicted in Fig. A.1.

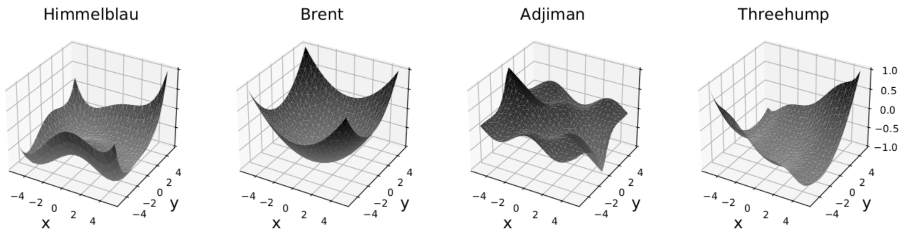
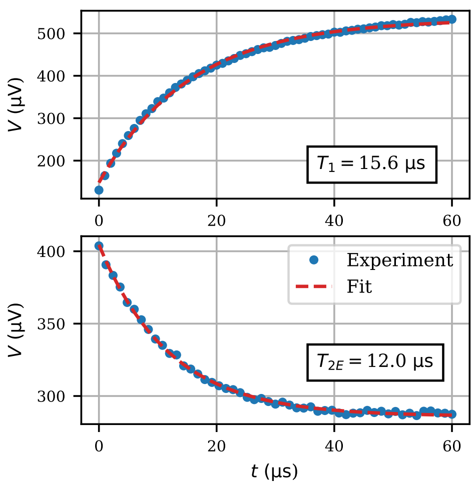


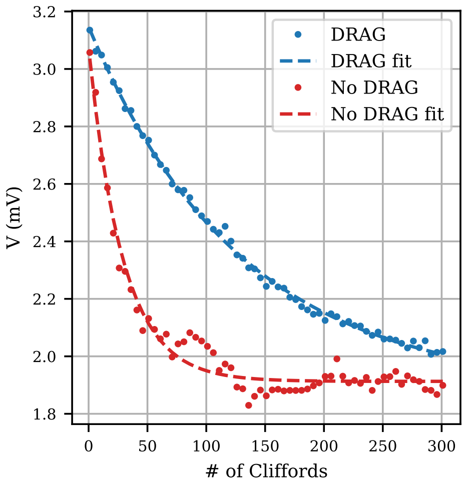
Figure A.1: *Graphical representation of 2-dimensional functions utilized for benchmarking. A regularization is applied to obtain $Z \in [-1, 1]$.*

A.4 Superconducting experiment for a universal approximant

The experimental implementation of the single-qubit universal approximant as detailed in Sec. 3.2 was performed in a superconducting qubit circui. The qubit is a 3D transmon geometry [Pai+11] located inside an aluminum three-dimensional cavity. The cavity bare frequency, $\omega_c = 2\pi \times 7.89$ GHz, is greatly detuned from the qubit frequency, $\omega_q = 2\pi \times 4.81$ GHz. Hence, there is a qubit state-dependent dispersive shift on the cavity resonance, $2|\chi| = 2\pi \times 1.5$MHz. The qubit anharmonicity is $\alpha = -2\pi \times 324$ MHz and the qubit relaxation and spin-echo decay times are, respectively, $T_1 = 15.6\ \mu s$ and $T_{2E} = 12.0\ \mu s$. These time scales exceed the operation



(a) T_1 and T_{2E} fits



(b) Randomized Benchmarking

Figure A.2: *a) T_1 measurement with exponential fit (top) and spin-echo measurement, T_{2E}, with exponential fit (bottom). b) Randomized benchmarking of the DRAG corrected pulses. The fit corresponds to the expression $Ap^n + B$, where A and B have dimensions of voltage, n is the number of Clifford gates, and p is the fidelity per gate. $\epsilon = 1 - p$ is the error per gate.*

times needed to implement the algorithm up to 6 layers by 2 orders of magnitude. See Fig. A.2(a) for a experimental fit on these times.

The experiment was realized in a dilution fridge with a base temperature of approximately 20 mK. The qubit rotation pulses were defined by an arbitrary waveform generator and then upconverted with a microwave signal generator to the gigahertz frequency range before being sent to the qubit/cavity system. The signal was low-pass filtered and attenuated by a total of 50dB before reaching the aluminum cavity. The input port of the cavity was undercoupled while the output port was overcoupled in order to maximize the readout signal amplitude. The outgoing signal was amplified by a cryogenic low noise amplifier and a second amplification stage at room temperature. The downconversion is performed with the same microwave generator as used in the upconversion of the measurement pulse, guaranteeing phase coherence in the downconversion process. The signal is read out in a digitizer, with a Field Programmable Gate Array (FPGA) that demodulates and averages the results before sending the data to the main measurement computer.

In order to implement the gate sequences defined in Sec. 3.2, the correspondence between logical and physical gates as shown in Fig. A.4 is followed. The phase of each pulse is selected at the pulse generator to modify the rotation axis, producing either R_X or R_Y rotations as required.

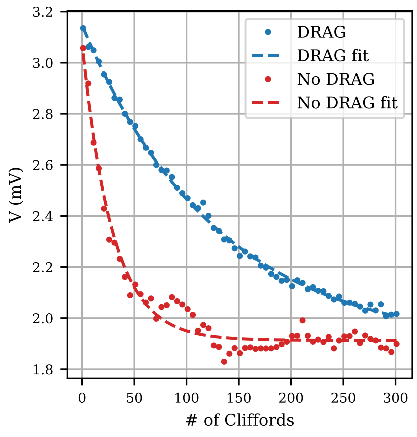
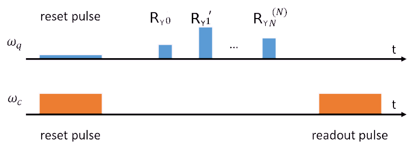
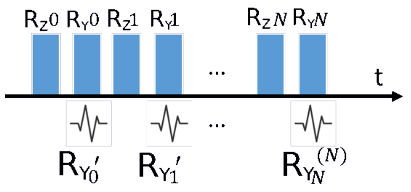


Figure A.3: *Randomized benchmarking of the DRAG corrected pulses. The fit corresponds to the expression $Ap^n + B$, where A and B have dimensions of voltage, n is the number of Clifford gates, and p is the fidelity per gate. $\epsilon = 1 - p$ is the error per gate.*

The R_Z rotations are, in turn, virtual [McK+17a]. The microwave pulses incorporate a DRAG correction [Cho+10; Mot+09] which leads to an error per gate $\epsilon = 0.01$ found with randomized benchmarking [MGE11], see Fig. A.3. Randomized benchmarking measures errors in Clifford gates and not arbitrary angle rotations, which are instead used in this experiment, yet offers a reasonable estimate on the overall fidelity of the implemented gates. The gate error observed is probably limited due to a non-ideal filtering of the measurement lines in the fridge. In order to achieve better qubit state readout visibility and shorter operation times,



(a) Full sequence



(b) Pulse sequence

Figure A.4: *a) Complete pulse sequence. First, the reset protocol is performed which corresponds to two pulses at the cavity and the qubit frequencies, respectively. Note that the qubit pulse is of considerably lower amplitude than the cavity pulse. Also, both pulses have a longer duration than the qubit rotation sequence (timings not to scale). The R_Y pulses are shown to have different amplitudes to determine each rotation angle. Finally, the readout corresponds to a pulse at the cavity frequency which is later read out by a digitizing card. b) Sequence performed in the experiment. Blue boxes represent actual pulses. Logical R_Y and R_Z rotations are explicitly shown below the blue boxes [McK+17a]. Note that R_Z pulses do not correspond to any microwave pulse, instead subsequent pulses change rotation axis, indicated by a prime, $R_{Y,N}^{(N)}$.*

Optimal parameters	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}	p_{11}	p_{12}
	-2.501	1.685	1.757	2.105	3.822	-1.788	-1.507	-4.640	0.430	1.875	5.038	-1.906

Rotational angles*	$R_{Z,1}$	$R_{Y,1}$	$R_{Z,2}$	$R_{Y,2}$	$R_{Z,3}$	$R_{Y,3}$	$R_{Z,4}$	$R_{Y,4}$
	$p_1 + p_2 x$	p_3	$p_4 + p_5 x$	p_6	$p_7 + p_8 x$	p_3	$p_{10} + p_{11} x$	p_{12}
$x = -0.5$	2.939	1.757	0.194	4.495	0.813	0.430	5.639	4.377
$x = 0$	3.782	1.757	2.105	4.495	4.776	0.430	1.875	4.377
$x = 1$	5.467	1.757	5.927	4.495	0.136	0.430	0.630	4.377

* Angles between 0 and 2π

Table A.1: *Optimal parameters and angles obtained for* ReLU(x) *and 4 layers. Above the 12 parameters that define the rotational angles obtained through simulations. Below the corresponding angles of the 8 rotations for three different values of x. Note that R_Y rotations are not x-dependent, hence they are equal for all three x values.*

a reset protocol is applied prior to the main sequence [Gee+13].

Figure A.4 shows the total pulse sequence, which includes preparation and measurement pulses in addition to the pulse sequence shown in the main text. The Y rotations are performed through microwave pulses at the qubit frequency while the Z rotations, as already stated, are phase changes in subsequent pulses. Both qubit and cavity pulses are generated at 70 MHz and then upconverted to the gigahertz range. The qubit pulses are gaussian pulses with a total duration of 21 ns. A proper DRAG correction is performed with a resulting error per gate of $\epsilon = 0.01$ as shown in Fig. A.3. The cavity pulse has a total length of around 2 μs. The reset protocol consists of a pulse driving the qubit and a pulse driving the cavity mode, with a total duration of around 2 μs. An example of the rotation angles for the ReLU(x) function in the 4-layer case is shown in Table A.1.

The readout consists of a cavity tone at the frequency of the cavity for the qubit in the $|0\rangle$ state. High/low transmission corresponds to the qubit being in the ground/excited state, assuming the system does not escape from the computational basis. Each data point requires around $5 \cdot 10^4$ measurements in order to average out the amplifier noise. A reset protocol that drives the qubit into the ground state is implemented prior to each individual sequence. This has two benefits. The first one allows to start with a qubit state nearly polarized into the ground state. A second benefit is the reduction in the overall duration of the experiment, since the waiting time between individual measurements is not limited by the qubit relaxation time.

A.5 Ion trap experiment for a universal classifier

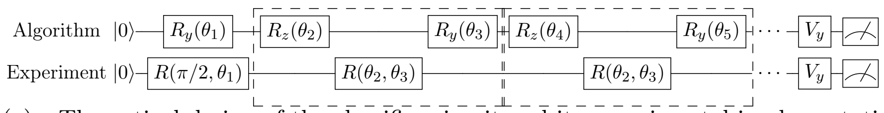
The ion-trap qubit utilized in this experiment is realized on a ^{138}Ba$^+$ ion trapped and laser cooled in a linear blade trap. The mapping between computational and electronic states is $|0\rangle \to S_{\frac{1}{2},-\frac{1}{2}}$ and $|1\rangle \to D_{\frac{5}{2},-\frac{1}{2}}$.

Both states are coupled by an electric quadrupole $E2$ transition at 1762 nm wavelength. To manipulate the qubit an ultra-low-linewidth laser is compulsory. Laser cooling of the ion to the Doppler limit, ensures that the qubit is not influenced by the external motion. The most relevant parameters of the trapped ion in this experiment are the qubit coherence time (5 ms) and the Rabi π-time (12 μs) of the qubit [DM20; Yum+17]. The qubit is well-characterized in terms of both its internal [DM20] and external states [Hor+20].

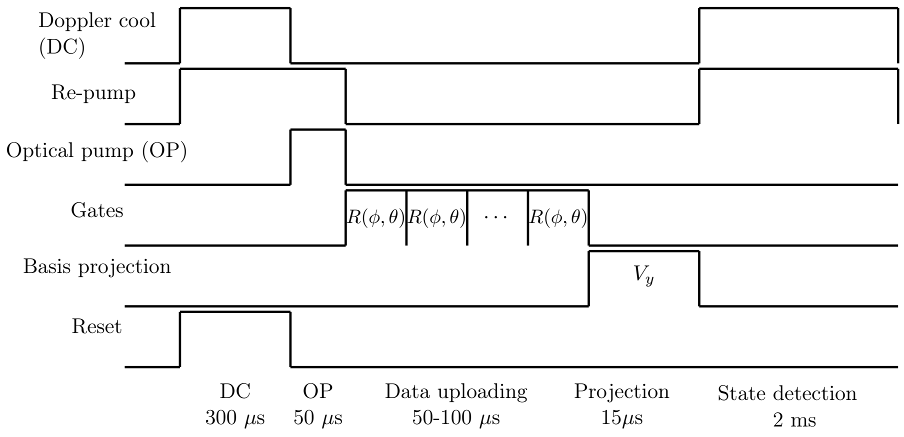
An ultra-low linewidth laser at 1762 nm wavelength is required to control the qubit. The laser achieves an estimated linewidth of ≤ 100 Hz. Prior to performing each algorithmic cycle the qubit is first Doppler cooled to the Lamb-Dicke regime [DM20] via a fast dipole transition (between S-P levels) at 493 nm and a re-pump laser (between D-P levels) at 650 nm. Subsequently, the qubit is initialized to the state $\mid 0\rangle \equiv \left| \mathrm{S}_{\frac{1}{2},-\frac{1}{2}} \right\rangle$ by optical pumping [Deh57], achieved by selectively de-populating the state $\mathrm{S}_{\frac{1}{2},+\frac{1}{2}}$. A high fidelity state initialization of $\geq 99\%$ is achieved within an optical pumping time of $\leq 50\mu$s only. Once the qubit is initialized, any single qubit rotational gate is implemented by resonantly driving the qubit with full control over the laser phase, power and laser on-time. A direct digital synthesizer is used to control the phase, frequency and power of the laser as the sequential gates are applied on the qubit. The synthesizer eliminates the long term frequency drift of the synthesizer clock, thus maintaining the phase relations of the sequential gates of the classifier. As classical data is uploaded in-the-fly, the latency of uploading the synthesizer parameters play a crucial role. The latency is minimized by pre-loading the full sequence of the phase, frequency and power data to an on-chip memory of the synthesizer. The synthesizer output is then controlled by an external trigger generated from a FPGA.

The classification algorithm requires the repetition of any pair of non-commuting rotational gates. R_z and R_y are chosen, instead of the active gates of the ion qubit R_x and R_y. Active rotations are realized by the application of resonant laser at the qubit frequency. On the other hand, R_z can be implemented by a combination of the other two active gates or by varying the qubit energy [Mas17; McK+17b]. Both these methods are error prone as the qubit-light interaction is switched on for certain time. It is chosen instead to perform the R_z by changing the laser phase without interacting with the ion [Kni+08].The resultant error is thus limited by only the R_y gate in each layer. Furthermore, every $R_z(\phi)$ gate followed by a $R_y(\theta)$ gate can be concatenated to a single gate $R(\phi, \theta)$ representing a qubit rotation, in a Bloch sphere representation, about the $(\phi, 0)$ axis by angle θ. The concatenation makes the effective circuit depth half of the original.

Each data point shown in Fig. 3.18 or any of the other classifier plots,



(a) Theoretical design of the classifier circuit and its experimental implementation



(b) Operational scheme for the ion-trap experiment

Figure A.5: *Experimental implementation of quantum classifier circuit: (a) The algorithm to implement the quantum classifier is represented by gates in the top qubit, grouped in pairs R_y, R_z. In the bottom qubit, couples of gates are concatenated into one rotational gate $R(\phi, \theta)$ with modified rotation axis. For one qubit it is possible to define at most two orthogonal states, so the unitary operation V_y adds any rotation to accomodate label states, see Eq. (3.70).(b) The time sequence used in the experiment to perform each classifier measurement.*

comprises of 100 repeat experiments. Every experiment, consists of a sequence of operations performed on the qubit as shown in Fig. A.5. Depending on the data to be uploaded, single qubit gates are implemented by resonantly driving the qubit transition between $|0\rangle, |1\rangle$ with well-controlled phase and operation time, while the frequency and power of the laser are kept constant. An acousto-optic modulator controls the phase and frequency of the 1760 nm laser implementing the rotation gates. The phase of the modulator is directly controlled by a synthesizer which supplies the radio-frequency signal to the modulator via an amplifier. The laser-qubit interaction time sets the rotation angle θ. Therefore, direct and precise control over the axis and angle of rotations is available. The optimal gate to apply repeatedly on the experiment is defined as:

$$R(\phi, \theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -ie^{-i\phi}\sin\frac{\theta}{2} \\ -ie^{i\phi}\sin\frac{\theta}{2} & \cos\frac{\theta}{2}. \end{bmatrix} \tag{A.17}$$

The time sequence shown in Fig. A.5 is controlled by a FPGA with a time jitter below 1 ns. The current version of the synthesizer controlling the phase of the laser is limited by the on-chip memory to 16 phase modulation steps thus limiting the layers to six, which is sufficient for the current discussion. Once the circuit runs over all the layers, the overlap between the final state of the qubit and a label state is measured. In case of binary classification, the final state is projected on to the label state $|0\rangle$ and the overlap is measured by observing spontaneously emitted photons while the qubit is excited by 493 nm laser. With a photon collection time of 2 ms, the bright state is clearly discriminated from the dark state. For any classification problem with more than two classes, two additional rotations are required before performing state discrimination. The overlap for each data point based on a threshold is designated a success (1) or failure (0) to be within a class.

The accuracy of the data re-uploading algorithm primarily relies on the the fidelity of individual single qubit rotation gate. The residual error in the gate operation is reflected in the accuracy of the classifier. In Eq. (A.17), the rotation angles are related to experimental values as $\phi = (\Delta/\hbar)t_{op} + \delta\phi$ and $\theta = (\Omega/\hbar)t_{op}$, where Δ is the laser detuning, t_{op} is the operation time of gates, Ω' is the modified Rabi frequency and $\delta\phi$ is the relative phase of the laser with respect to the qubit. The modified Rabi frequency is $\Omega' = \sqrt{\Omega_0^2 + \Delta^2}$ with Ω_0 denoting the resonant Rabi frequency. Furthermore, the resonant Rabi frequency is proportional to the square root of the intensity, I_0, at the ion position. Therefore each of the independent variables $\delta\phi$, t_{op}, Δ and I_0 contributes to the error in a rotation gate as follows:

$\delta\phi$ **Phase:** The synthesizer controls the radio-frequency phase of the modulator which determines the relative phase of the laser. Each synthesizer is synchronized to a rubidium atomic clock which is accurate to one part in 10^{10} and thus contributes negligibly to the phase error. The direct digital synthesizer is however triggered by the FPGA which has time jitter below 10 ns leading to phase noise on the qubit below 0.1% for a Rabi π time of 12 μs.

t_{op} **Interaction time:** The laser-qubit interaction time is determined by the FPGA, precise up to 1 ns. Therefore, due to the time jitter below 10 ns, its contribution to the accuracy is below 0.1%. Occasional collision with the residual background gas molecule during the interaction time leads to a projection to the state $|0\rangle$, thus losing the final state information and hence error in the classification. This error becomes usually smaller with larger statistics.

Δ **Laser-qubit detuning:** The detuning of the laser with respect to the qubit frequency modifies the Rabi frequency. The range of Rabi frequency fluctuation within the experimental time is quantified to about 10 min. To ensure that the accuracy is limited by systematic errors, statistical errors in the classification are measured by repeating the experiment, see Fig. A.6a. The error decreases from

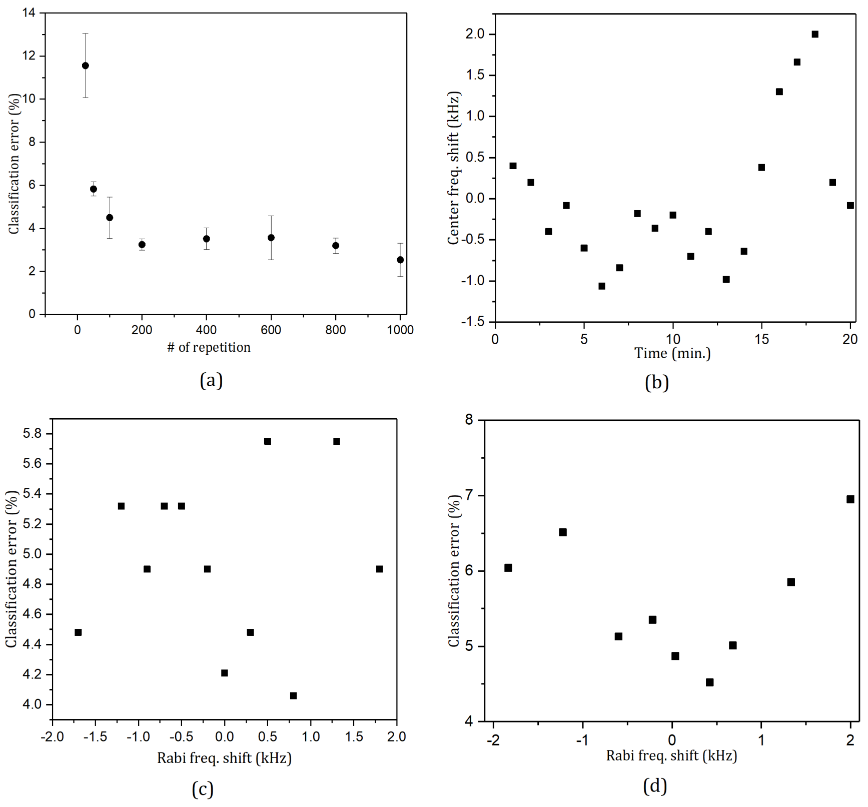


Figure A.6: *Systematic error analysis: All the results shown here are related to the binary classification of circle as in Fig. (3.18). The errors are classification error. (a) The classifier error as a function of the number of repeated experiments. The error bar at each point corresponds to 1 standard deviation of a number of repeat measurements for same number of repeated experiments under the same condition. The exact number of repeat measurements varies between 5 and 10. (b) Variation of the resonance frequency as a function of time. The range of Rabi frequency fluctuation within a typical experimental time of ≤ 10 minutes is about 2 kHz. (c) Error in binary classification of a circle feature with the variation of laser frequency detuning measured in terms of the Rabi frequency. The variation in the value of classification error is about 2% within the experimental time of ~ 10 min. (d) The same plot as in (c) but by varying the laser power measured in terms of the Rabi frequency.*

12% to about 4% for 100 repetitions and then enters a stationary regime limited by systematic errors. The fluctuation of the laser frequency with respect to the atomic resonance is captured over a time period of 20 min (twice the duration of an experiment) as plotted in Fig. A.6b. The random variation of the Rabi frequency over time is mostly caused by the magnetic field noise as the laser frequency drift was separately measured to be ≤ 5 kHz/24 hrs [Yum+17]. To minimize the impact of the residual magnetic field noise, electronic levels weakly sensitive to such noise are used. In addition, the detuning also indirectly influence the modified Rabi frequency. To check its influence, the Rabi frequency is varied as shown in Fig. A.6c, by varying the detuning within a 2 kHz range (as expected from the Fig. A.6b). The result shows below 5% accuracy for the classifier when operating for 10 min.

I_0 **Laser intensity:** The Rabi frequency is fixed by setting the power and frequency of the laser at the start of the experiment. Any change in the Rabi frequency during the experiment leads to error in applied qubit rotation angle. The intensity is influenced by two factors, the laser power noise and the laser beam pointing error. In the experiment, the laser beam is tightly focused on the ion by a high Numerical Aperture ~ 0.4 in-vacuum lens. In order to capture the influence of laser power variations on the classification error, the power is varied, see Fig. (A.6d). Thus it is seen that the influence of intensity noise accounts to 5% error in accuracy. To avoid the influence of Rabi frequency fluctuation within the experimental time of ~ 10 min, the Rabi frequency is reduced from 625 kHz to 80 kHz such that the absolute error also reduces. This leads to an overall error of only 2% on the classifier output.

A.6 Quantum circuits in NNPDF methodology

The latest implementation of the latest iteration of the NNPDF methodology is described in Ref. [CC19]. This implementation is very modular and one can seamlessly swap the `Tensorflow` based backend by any other provider. `Qibo`, which is also partially based on `Tensorflow` can be easily integrated with the NNPDF methodology.

As previously mentioned, all results in this section corresponds to the simulation of the quantum device on classical hardware. Such a simulation is very costly from a computational point of view which introduces a number of limitations that need to be addressed in order to produce results in reasonable time frames.

FK reduction: the definition of the quantum circuit depends on both the set of parameters θ and the value of the parton momentum fraction x (see Eq. (3.81)) which means the circuits needs to be simulated once per value of x. The union of all FK tables for all physical observables (following Eq. (3.88)) amounts to several thousand values of x. Since such a large number of evaluations of the quantum circuit is impracticable, a further approximation is introduced where each partial FK table is mapped to a fixed set of 200 nodes in the x-grid. This simplification introduces an error to the total χ^2 of the order of $\Delta\chi^2 = 0.14 \pm 0.01$

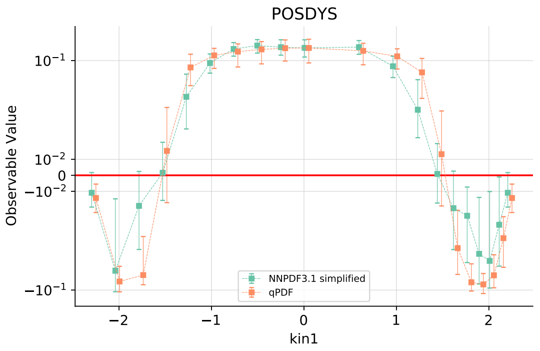


Figure A.7: *Predictions for a toy $s\bar{s}$ initiated Drell-Yan process with qPDF and a simplified version of NNPDF3.1 where the positivity constraint has been removed.*

when averaged over PDF members. This error on the cost function is however negligible for the accuracy reached in this work.

Positivity: in the fitting basis, as defined in section 3.5.1, the PDF cannot go negative. Physical predictions however are computed in the flavour basis [Bal+09] where the rotation between basis can make some results go negative. However, physical observables (differential or total cross sections) cannot be. This physical constraint is included in NNPDF3.1 via fake pathological datasets. These have not been implemented for qPDF as they correspond to a fine-tuning of the methodology which is beyond the scope of this work.

The removal of the positivity constraint from the fit introduces an unphysical distortion to the results as the PDF could produce negative predictions for physical predictions. Such results are unphysical because they would correspond to situations in which the probability of finding a particular phase space configuration is negative, which makes no sense. In Fig. A.7, the "negativity" between qPDF and a version of NNPDF3.1 with the positivity constraints removed is compared. We observe that both fits behave similarly, proving such unphysical results are a consequence of the removal of the constraint rather than a problem in the qPDF methodology.

Momentum Sum Rule: the PDFs as defined in Eq. (3.87) are normalized such that [Bal+15],

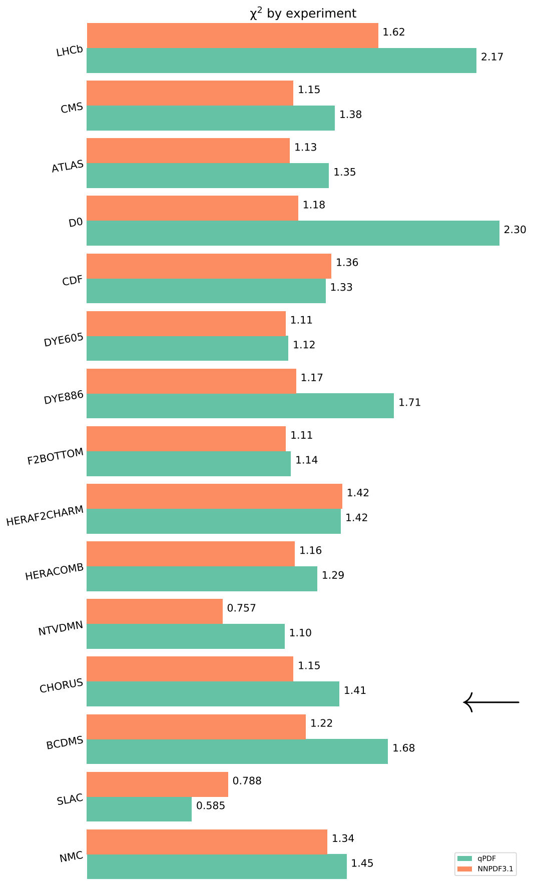
$$\frac{\displaystyle\int_0^1 dx\; x\, f_g(x, Q_0)}{1 - \displaystyle\int_0^1 dx x f_\Sigma(x, Q_0)} \simeq 1, \tag{A.18}$$

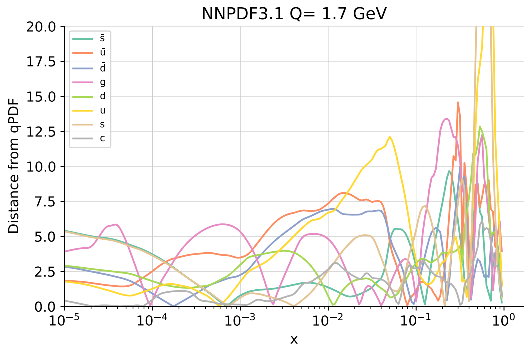
this equation is known as the momentum sum rule and it is imposed in `n3fit` through an integration over the whole range of x which is impracticable in this implementation for the reasons mentioned above. Instead, in qPDF these are only checked afterwards, finding a good agreement with the expected values (despite not being imposed at fitting time). Indeed, for qPDF the result for the average over all replicas is 1.01 ± 0.01, which is to be compared with the NNPDF3.1 result of 1.000 ± 0.001, where the constraint was imposed at fit time.

Extra results

With all ingredients implemented, it is possible to run a NNPDF3.1-like fit using the qPDF. As a base reference for the comparison, the NNPDF3.1 NNLO fit [Bal+17] is taken, which is the latest release by the NNPDF collaboration. The plots comparing the NNPDF sets with qPDF are then produced using a `reportengine` [Kas19] based internal NNPDF tool. The extra result here presented complement the one showed in the main text, in particular related to Fig. 3.28.

We can start by comparing the χ^2/N result for the datasets that have been considered in the fit, shown in Fig. A.8(b). One would expect a perfect fit when $\chi^2/N = 1$, however this is not the case even in the

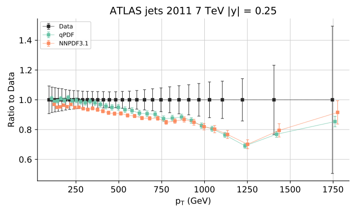




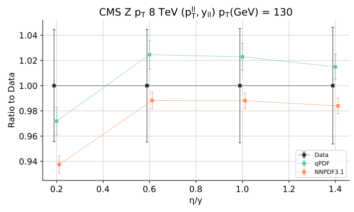
(a) Distance (as defined by Eq. (A.19)) between qPDF and NNPDF3.1. When the distance is kept under $d(f_i, r_i) = 10$ the two fits are 1-σ compatible. All partons except for u and s are below or around the 1-σ distance for the entire range considered. Note however, by comparing to Fig. 3.24 that the fits for both the u and s quarks are compatible in the most relevant regions for these particles.

$\longleftarrow$ (b) χ^2/N per experiment grouping. There is a deterioration of the goodness of the fit (measured by the χ^2) for some of the experiments for the central value. The goodness of the fit is very similar between the reference and qPDF for most of the experiments being considered.

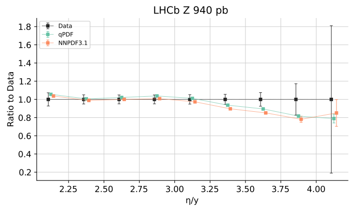
Figure A.8: *Collection of extra results for the qPDF method integrated in the NNPDF methodology as extracted from LHC data.*



(a) Atlas jets data differential in rapidity [Aad+15].



(b) CMS Z differential in rapidity for fixed value of p_T [Kha+15].



(c) LHCb, Z cross section differential in rapidity [Aai+12].

Figure A.9: *Theoretical predictions computed with the method describe in [BCH17] in order to compare the same prediction with three different PDF sets. Note that the predictions for the qPDF set is compatible with both the experimental measurements and the released PDF set. The parton-level calculation has been performed with the NLOjet++ [Nag02] and MCFM [CN19] tools.*

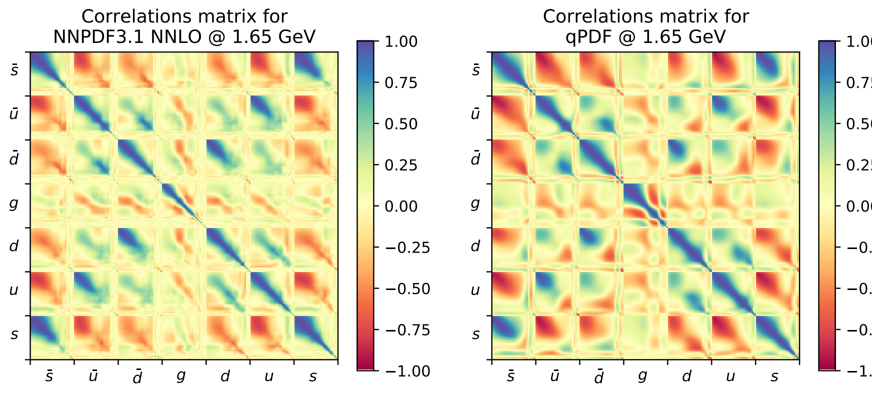


Figure A.10: *PDF correlation matrix for flavours in a grid of x points for NNPDF3.1 NNLO (left) and the qPDF (right).*

reference and it is due to a combination of missing higher order corrections (a lack of a better theory) or inconsistencies in the experimental results. The similarity on the phenomenological results obtained by both fitting methodologies as shown in Fig. A.8(b) is well understood as well by looking at the distance plots between the qPDF and the reference in Fig. A.8(a),

$$d^2(f_i, r_i) = \frac{\langle f_i \rangle - \langle r_i \rangle}{\frac{1}{N_f}\sigma(f_i)^2 + \frac{1}{N_r}\sigma(r_i)^2}, \tag{A.19}$$

where i is the flavour being considered and f and r corresponds to qPDF and the reference (NNPDF3.1) respectively. The central value is taken over the N replicas of the set, generally of the order of 100.

Indeed, for most partons the difference between both fits are under the 1-σ level (distance equal to 10 for 100 replicas) growing up to 2-σ for the

u and s quarks.

In Fig. A.9 it is shown specifically a comparison between the reference NNPDF3.1 and qPDF for selected datasets, and the LHAPDF-compatible PDF grid is provided. The accuracy of the qPDF central value is similar to that of NNPDF3.1. Furthermore, the error bars for the predictions of both PDF set overlap with the experimental error bars, and, in some cases, also among themselves.

Finally, in Fig. A.10the PDF correlations for NNPDF3.1 and qPDF replicas using Pearson's coefficient in a fixed grid of 100 points distributed logarithmically in $x = [10^{-4}, 1]$ are computed.

This leads to conclude that the methodology described in this paper can be used for regression problems to unknown functional forms such as the proton internal structure and produce results that are perfectly coherent, from a phenomenological point of view, with the state of the art. In addition it is believed that with adequate tuning one could achieve the same level of accuracy of the classical approach.

We finalize this section by showing phenomenological results where the LHAPDF grids produced with this approach are used for a full fixed order prediction. In summary going back circle to the master equation, i.e., computing numerically Eq. (3.88) with no approximations using state of the art tools.

B. Technical appendices for unary

B.1 Details for the Black-Scholes model

The Black-Scholes model for the evolution of an asset is based on the stochastic differential equation [BS73]

$$\mathrm{d}S_T = S_T\, r\, \mathrm{d}T + S_T\, \sigma\, \mathrm{d}W_T, \tag{B.1}$$

where r is the interest rate, σ is the volatility and W_T describes a Brownian process. Recall that a Brownian process W_T is a continuous stochastic evolution starting at $W_0 = 0$ and made of independent gaussian increments. To be specific, let $\mathcal{N}(\mu, \sigma_s)$ be a normal distribution with mean μ and standard deviation σ_s. Then, the increment related to two steps of the Brownian processes is $W_T - W_S \sim \mathcal{N}(0, T - S)$, for $T > S$.

The above differential equation can be solved analytically up to first order using Ito's lemma [Itô44]. The essential observation if that W_T is treated as an independent variable with the property that $(\mathrm{d}W_T)^2$ is of the order of $\mathrm{d}T$. Thus, the approximated derivative $\mathrm{d}S_T$ can be written as

$$\mathrm{d}S_T = \left(\frac{\partial S_T}{\partial T} + \frac{1}{2} \frac{\partial^2 S_T}{\partial W_T^2} \right) \mathrm{d}T + \frac{\partial S_T}{\partial W_T} \mathrm{d}W_T. \tag{B.2}$$

By direct comparison to Eq. (B.1), it is straightforward to see that

$$\frac{\partial S_T}{\partial W_T} = S_T\, \sigma, \tag{B.3}$$

$$\frac{\partial S_T}{\partial T} + \frac{1}{2} \frac{\partial^2 S_T}{\partial W_T^2} = S_T\, r. \tag{B.4}$$

Using the initial condition S_0 at $T = 0$, and the Ansatz

$$S_T = S_0 \exp\{(f(T) + g(W_T))\}, \tag{B.5}$$

the solution for the asset price turns out to be

$$S_T = S_0 e^{(r-\frac{\sigma^2}{2})T} e^{\sigma W_T} \sim S_0 e^{\mathcal{N}\left(\left(r-\frac{\sigma^2}{2}\right)T, \sigma\sqrt{T}\right)}. \tag{B.6}$$

This final result corresponds to a log-normal distribution.

B.1.1 European Option

An option is a contract where in its call/put form, the option holder can buy/sell an asset before a specific date or decline such a right. As a particular case, European options can be exercised only on the specified future date, and only depend on the price of the asset at that time. The previously agreed price that will be paid for the asset is called exercise price or *strike*. The day on which the option can be exercised is called *maturity date*.

A European option payoff is defined as

$$f(S_T, K) = \max(0, S_T - K), \tag{B.7}$$

where K is the strike price and T is the maturity date. An analytical solution exists for the payoff of this kind of options.

The expected payoff is given by

$$\begin{aligned} C(S_T, K) = \text{average}_{S_T \geq K}\,(S_T - K) = \int_{d_1}^{\infty} (S_T - K)\, dS_T = \\ = \int_{d_1}^{\infty} \frac{S_0}{\sqrt{2\pi}} e^{-\frac{\left(x-\left(r-\frac{\sigma^2}{2}\right)T\right)^2}{2\sigma^2 T}} e^{\frac{-x^2}{2}} dx, \end{aligned} \tag{B.8}$$

yielding the analytical solution

$$C(S_T, K) = S_0 \text{CDF}_{\mathcal{N}}(d_1) - K e^{-rT} \text{CDF}_{\mathcal{N}}(d_2), \tag{B.9}$$

with

$$d_1 = \frac{1}{\sigma\sqrt{t}} \left(\log \frac{S_0}{K} + \left(r + \frac{\sigma^2}{2} \right) T \right) \tag{B.10}$$

$$d_2 = d_1 - \sigma\sqrt{T} \tag{B.11}$$

$$\text{CDF}_{\mathcal{N}}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{\frac{-u^2}{2}} du. \tag{B.12}$$

This analytical development for the European option using the Black-Scholes model cannot be extended to some exotic options, like the American (the option can be exercised at any point) or the Asian (the final option price depends on a time average of the price until the maturity date). Therefore, in a general case the expected payoff return of an option cannot be obtained analytically, but rather performing a Monte Carlo simulation where many different scenarios are taken into consideration, and a global estimation is obtained.

B.2 Details for the Amplitude Distributor $\mathcal{D}$ in the unary basis

Figure 4.4 is considered. In the unary basis, every qubit represents the basis element in which the qubit is $|1\rangle$. Thus, the coefficient of every element depends on as many angles as partial-SWAP gates are needed to reach its corresponding qubit. Thus, the central qubits of the circuit will depend only on 2 angles, and the number of dependencies increases one by one as the gates move to the outer part of the circuit. The very last two qubits depend on the same angles. As the procedure goes by and moves away from the middle qubit, each qubit inherits the same angle dependency than the previous ones plus an additional rotation. Starting from the two edges, their coefficients verify the following ratios

$$\left|\frac{\psi_0}{\psi_1}\right|^2 = \tan^2(\theta_1/2) \tag{B.13}$$

$$\left|\frac{\psi_{n-1}}{\psi_{n-2}}\right|^2 = \tan^2(\theta_{n-1}/2). \tag{B.14}$$

Then $|\psi_i|^2 = p_i$, where $\{p_i\}$ is the target probability distribution of the asset prices at maturity. The next step corresponds to considering the qubits 1 and 2, as well as $n-3$, $n-2$. The relations for their coefficients must obey

$$\left|\frac{\psi_i}{\psi_{i+1}}\right|^2 = \cos^2(\theta_i/2)\tan^2(\theta_{i+1}/2) \tag{B.15}$$

$$\left|\frac{\psi_{n-1-i}}{\psi_{n-2-i}}\right|^2 = \cos^2(\theta_{n-i}/2)\tan^2(\theta_{n-1-i}/2). \tag{B.16}$$

Then, it is straightforward to back-substitute parameters step by step until we arrive to the central qubits. This procedure fixes all the angles for the partial-SWAP gates used in the amplitude distributor.

The exact algorithm to be followed can be also found in the provided code [RP20].

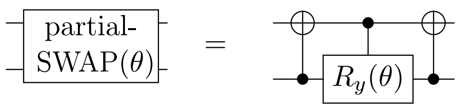
Once the exact solution for the angles is inserted into the circuit depicted in Fig. 4.4, the amplitude distributor algorithm is completed.

The quantum register then reads

$$|\Psi\rangle = \sum_{i=0}^{n-1} \sqrt{p_i}\, |i\rangle \,. \tag{B.17}$$

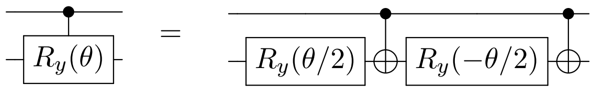
Note that describing a probability distribution with squared amplitudes of a quantum state allows for a free phase in every coefficient of the quantum circuit. For simplicity, we will set to zero all these relative phases by only operating with real valued partial-SWAP gates.

Let us turn our attention to the gates which are needed in the above circuit. Sharing probability between neighbor qubits can be achieved by introducing a two-qubit gate based on the SWAP and R_y operations. This variant on the SWAP gate performs a partial SWAP operation, where only a piece of the amplitude is transferred from one qubit to another. This operation preserves unarity, that is the state remains as a superposition of elements of the unary basis. This partial-SWAP, can be decomposed using CNOT as the basic entangling gate as



(B.18)

where the usual CNOT gate in the center of the conventional SWAP gate has been substituted by a controlled y-rotation, henceforth referred to as cR_y gate. In turn, the cR_y operation can be reworked as a combination of single-qubit gates and CNOT gates [Bar+95]:



(B.19)

This decomposition will come into play for the expected payoff calculation algorithm as well, albeit with angle ϕ in the payoff circuit.

For the purposes of this algorithm, both the CNOT and partial-iSWAP basis gates are analogous, but the direct modeling to partial-iSWAPs can economize the total number of required gates for the amplitude distributor. Partial-iSWAP gates can be used to decompose CNOT gates. More explicitly, a CNOT gate an be reproduced with two iSWAP gates, and 5 single qubit gates.

B.3 QAE

QAE is a general framework to estimate the probability of obtaining a certain outcome if measuring a given quantum state. This procedure allows to gain quantum advantage with respect to Monte Carlo samplings.

QAE is in general defined from an algorithm $\mathcal{A}$ such that

$$\mathcal{A}\left|0\right\rangle_n\left|0\right\rangle = \sqrt{1-a}\left|\psi_0\right\rangle_n\left|0\right\rangle + \sqrt{a}\left|\psi_1\right\rangle_n\left|1\right\rangle, \tag{B.20}$$

where the last qubit serves as an ancilla qubit and the states $\left|\psi_{0,1}\right\rangle_n$ can be non-orthogonal. The ancilla qubit is a flag which enables to identify the states as *good* ($\left|1\right\rangle$) or *bad* ($\left|0\right\rangle$). The state $\mathcal{A}\left|0\right\rangle_n\left|0\right\rangle$ can be directly sampled N times, and the estimate for probability of finding a good outcome will be $\bar{a}$, with

$$|a-\bar{a}| \sim \mathcal{O}(N^{-1/2}), \tag{B.21}$$

as dictated by the sampling error of a multinomial distribution.

However, QAE can improve this result. Let us first define the central operator for QAE [Bra+02]

$$\mathcal{Q} = -\mathcal{A}\mathcal{S}_0\mathcal{A}^\dagger\mathcal{S}_{\psi_0}, \tag{B.22}$$

where the operators $\mathcal{S}_0$ and $\mathcal{S}_{\psi_0}$ are inherited from Grover's search algorithm [Gro96], being

$$\mathcal{S}_0 = \mathbf{I} - 2\left|0\right\rangle_n\left\langle 0\right|_n \otimes \left|0\right\rangle\left\langle 0\right|, \tag{B.23}$$
$$\mathcal{S}_{\psi_0} = \mathbf{I} - 2\left|\psi_0\right\rangle_n\left\langle \psi_0\right|_n \otimes \left|0\right\rangle\left\langle 0\right|. \tag{B.24}$$

The $\mathcal{S}_0$ operator changes the sign of the $\left|0\right\rangle_n\left|0\right\rangle$ state, while $\mathcal{S}_{\psi_0}$ takes the role of an oracle and changes the sign of all bad outcomes.

B.3.1 QAE with QPE

The original QAE makes use of a QPE subroutine to study the operator $\mathcal{Q}$, with eigenvalues $e^{\pm i2\theta_a}$, with $a=\sin^2(\theta_a)$ [Bra+02]. The procedure of QPE is then applied to extract an integer number $y \in \{0,1,\ldots,2^m-1\}$ such that $\bar{\theta}_a = \pi y/2^m$ is an estimate of θ_a, with m the number of ancilla qubits. Recall that a QFT is required to perform QPE.

The value of $\bar{\theta}_a$ leads to an estimate of $\bar{a}$, such that

$$|a-\bar{a}| < \frac{2\pi\sqrt{a(1-a)}}{2^m} + \frac{\pi^2}{2^{2m}} \sim \mathcal{O}\left(\frac{\pi}{2^m}\right) \tag{B.25}$$

with probability at least $8/\pi^2 \approx 81\%$.

The original QAE procedure requires the implementation of QPE, which is highly resource demanding. Hence, the complexity of the circuit precludes its feasibility in the NISQ era.

Algorithm 1: Algorithm for QAE based on gaussian distribution of the measurements.

```
1  GaussianAmplitudeEstimation($N_{\text{shots}}$, $J$, $m_j$, $\alpha$)
2  $z \leftarrow \text{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)$
3  Ensure $m_0 = 0$
4  $a \leftarrow |\langle 1| \mathcal{A} |0\rangle|^2$ with $N_{\text{shots}}$ samples
5  $\theta_a^{(0)} \leftarrow \arcsin\sqrt{a}$ $\Delta\theta_a^{(0)} = \frac{z}{2\sqrt{N_{\text{shots}}}}$
6  for $j \leftarrow 1$to $J$ do
7      $a \leftarrow |\langle 1| Q^{m_j} A |0\rangle|^2$ with $N_{\text{shots}}$ samples
8      $\theta_{\text{array}} \leftarrow$ MultipleValuesArcsin($a, m_{j-1}$)
9      $\theta_a \leftarrow \min\left(|\theta_{\text{array}} - \theta_a^{(j-1)}|\right)$
10     $\Delta\theta_a \leftarrow \frac{z}{2(2m_j+1)\sqrt{N_{\text{shots}}}}$
11     $\theta_a^{(m_j)} \leftarrow \frac{\frac{\theta_a}{\Delta\theta_a^2} + \frac{\theta_a^{(j-1)}}{(\Delta\theta_a^{(j-1)})^2}}{\frac{1}{\Delta\theta_a^2} + \frac{1}{(\Delta\theta_a^{(j-1)})^2}}$
12     $\Delta\theta_a^{(m_j)} \leftarrow \left(\frac{1}{\Delta\theta_a^2} + \frac{1}{(\Delta\theta_a^{(j-1)})^2}\right)^{-1/2}$
13     $[a_j, \Delta a_j] \leftarrow [\sin^2\theta_a^j, \sin\left(2\theta_a^j\right)\Delta\theta_a^{(j)}]$
14 return $[a_j, \Delta a_j]$
```

Algorithm 2: Extracting multiple values for the arcsin, auxiliary function needed in Alg. 1.

```
1  MultipleValuesArcsin($a, m$)
2  $\theta_0 \leftarrow \arcsin\sqrt{a}$ // The value of $\theta_0$ is bounded between 0
       and $\pi/2$
3  The arcsin function has several solutions $\theta_{\text{array}} \leftarrow [0] * (2m+1)$
     $\theta_{\text{array}}[0] \leftarrow \theta_0$
4  for $k \leftarrow 1$to $m$ do
5      $\theta_{\text{array}}[2k-1] \leftarrow k\pi - \theta_0$
6      $\theta_{\text{array}}[2k] \leftarrow k\pi + \theta_0$
7  $\theta_{\text{array}} \leftarrow \theta_{\text{array}}/(2m+1)$
8  return $\theta_{\text{array}}$
```

B.3.2 IQAE

Here, a method is presented for obtaining the most probable value of a in an iterative fashion following similar methods as other QAE without QPE algorithms. We base this procedure in the theory of confidence intervals for a binomial distribution assuming normal distributions [Wal13].

A binomial distribution with probability a is considered, i. e. for every sample the chance of obtaining 1 is a, while the chance of obtaining 0 is $1 - a$. Then, if an estimate $\hat{a}$ of a was obtained using N samples, the true value of a lies in the interval

$$a = \hat{a} \pm \frac{\mathrm{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)\sqrt{\hat{a}(1-\hat{a})}}{2\sqrt{N}}, \tag{B.26}$$

with confidence $(1 - \alpha)$.

From this result QAE can construct an iterative algorithm returning the optimal value of a using QAE. A set of m_j for $j = 0, 1, 2, 3, \ldots$ is considered. For every m_j the probability of obtaining $|1\rangle$ is $\sin^2((2m_j + 1)\theta_a)$, where $a = \sin^2(\theta_a)$. In the θ space, for a given m the values and error of θ obtained are

$$\theta_a = \frac{1}{2m+1} \arcsin\left(\sqrt{a}\right) \qquad \Delta\theta_a = \frac{1}{2m+1} \frac{\mathrm{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)}{2\sqrt{N}}. \tag{B.27}$$

It is important to understand two main properties of Eq. (B.27). First, there are $2m + 1$ possible values for θ_a within the interval $\theta_a \in [0, \pi/2]$ as the $\sin^2(\cdot)$ function is π-periodical. For every new iteration it will be necessary to choose one of them. It is very important to set $m_j = 0$ at first because this case is the only one for which θ_a corresponds to the expected value for a. Otherwise, several possible values of a arise and it is not possible to tell which one is correct. Combining results for several values of m_j, it is possible to bound the uncertainty to be as small as desired.

The algorithm is based on the following statements. For a given collection of measurements and uncertainties $\{\theta_i, \Delta\theta_i\}$, the weighted average and uncertainty from the first j terms is

$$\tilde{\theta}_j = \frac{\sum_{i=0}^{j} \theta_i/\Delta\theta_i^2}{\sum_{i=0}^{j} 1/\Delta\theta_i^2} \qquad \Delta\tilde{\theta}_j = \left(\sum_{i=0}^{j} 1/\Delta\theta_i^2\right)^{-1/2}. \tag{B.28}$$

Notice also that this relation is recursive, as $\tilde{\theta}_{j+1}$ can be obtained by combining $\tilde{\theta}_j$ and θ_{j+1}. The same holds for uncertainties. Thus, the interpretation of this algorithm is that for every new step j a new term is added to the series $\{\theta, \Delta\theta\}$. The individual uncertainties decrease as $\sim ((2m + 1)^{-1})$, and the final global uncertainty is obtained as

$$\Delta\theta = \frac{\mathrm{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)}{\sqrt{N}} \left(\sum_{j=0}^{J} (2m_j + 1)^2\right)^{-1/2}, \tag{B.29}$$

where J denotes the last iteration performed. The full recipe for the algorithm is described in Algs. 1 and 2.

In the case of a linear selection of m_j, i. e. $m_j = j; j = (0, 1, 2, ..., J)$, the asymptotic behavior of this uncertainty is $\Delta\theta = \mathcal{O}(N^{-1/2}M^{-3/4})$, with M the sum of all m. For discovering it we just have to compute

$$\sum_{j=0}^{J}(j+1)^2 = 4\sum_{j=0}^{J} j^2 + 4\sum_{j=0}^{J} j + \sum_{j=0}^{J} 1. \tag{B.30}$$

We now take the identities $\sum_{j=0}^{J} j = J(J+1)/2 = M$ and $\sum_{j=0}^{J} j^2 = J(2J+1)(J+2)/6$. Then, it is direct to check that

$$\Delta\theta = \mathcal{O}(N^{-1/2}J^{-3/2}) = \mathcal{O}(N^{-1/2}M^{-3/4}). \tag{B.31}$$

This behavior already surpasses the tendency of the classical sampling, but does not reach the optimal QAE with QPE.

In the case of an exponential selection of m_j, i. e. $m_j = \{0\} \cup \{2^j\}; j = (0, 1, 2, ..., J)$ we can take the identities $\sum_{j=0}^{J} 2^j = 2^J - 1 = M$ and $\sum_{j=0}^{J} 2^{2j} = (2^{2J} - 1)/3$. Then it is direct to check that

$$\Delta\theta = \mathcal{O}(N^{-1/2}2^{-J}) = \mathcal{O}(N^{-1/2}M^{-1}). \tag{B.32}$$

Extension of QAE to error-mitigation techniques

The error-mitigation procedure proposed for the unary algorithm discards some of the algorithm instances to retain outcomes within the unary basis. This reduces the precision achieved in the algorithm with respect to the ones predicted in Eqs. (B.31) and (B.32) in order to maintain accuracy. This section provides some lower bounds on how many QAE iterations can be done while still reaching quantum advantage.

We will work now in the scheme where $m_j = j$. Let us assume that, in every iteration of QAE, only a fraction $\tilde{p}_j$ of the shots are retained. The equivalent version of Eq. (B.29) is now

$$\Delta\theta = \mathrm{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)\left(\sum_{j=0}^{J}(2m_j+1)^2 N\tilde{p}_j\right)^{-1/2}. \tag{B.33}$$

As more errors are bound to occur, $\tilde{p}_j$ decreases as m_j increases, we can state a bound for the accuracy as

$$\Delta\theta \leq \frac{\mathrm{CDF}_{\mathcal{N}}^{-1}(1-\alpha/2)}{\sqrt{N\tilde{p}_J}}\left(\sum_{j=0}^{J}(2m_j+1)^2\right)^{-1/2}, \tag{B.34}$$

since the precision is at least as good as the one obtained for the worst-case scenario. Comparing the trends, both in the linear and the exponential case, with the classical scaling, it is possible to see that quantum advantage is still achieved provided

$$\tilde{p}_J \geq M^{1-2\alpha}, \tag{B.35}$$

with $\alpha = 3/4$ in the linear case and $\alpha = 1$ in the exponential case. These quantities for the linear case correspond to the dashed lines in Figs. 4.13 and 4.15(b).

The probability of retaining a shot is at least the probability of having no errors in the circuit, considering that some double errors may lead to erroneous instances that belong even though to the unary basis. This zero-error probability in the worst case scenario, that is, at the last iteration of QAE, is written as

$$p_0 = \left((1 - p_e)^{an+b}\right)^{m_J}, \tag{B.36}$$

where p_e is the error of an individual gate, and a and b are related to the gate scaling, see Tab. 4.1 for the details. In principle, one can expand the calculation of p_0 by considering different kinds of errors for different gates, but for the sake of simplicity we will focus on this analysis. Rearranging together the results for Eqs. (B.31), (B.32) and (B.35) it is possible to see that quantum advantage is obtained if the individual gate errors is bounded by

$$p_e < 1 - m_J^{\frac{2-4\alpha}{(an+b)m_J}}. \tag{B.37}$$

C. Qibo

Qibo is an open-source software project to write and execute both quantum circuits and adiabatic quantum computing in a user-friendly manner [Eft+20a; Eft+20b]. In the global perspective, Qibo comes as a python library to join other such as qiskit [Ale+19a], cirq [Cir21], Forest [SCZ16], Qulacs [Suz+20a] or QCGPU [Kel18] among many others [Be18; Be20; BG16; Ce18; De07; De19; Fe18; JB20; Jon+19; Luo+19; Mey+20; Mic21; Mou+20; MS20; Pe17; SHT18; Ve19; We20; ZW17; ZYL15]. The effort of the Qibo project is coordinated by TII[1] and Qilimanjaro[2]. The current status of Qibo is that quantum algorithms can only be exactly simulated on classical hardware, but future plans forsee to extend the calculations on approximate classical methods such as the family of TNs [BB17], and on actual quantum devices.

The structure of Qibo is designed to be easy to use but extremely efficient to perform calculations. The final target is to make the catalogue of functionalities and the available applications increase with time. The high-level API receives the instructions from the user and allocates automatically all different ingredients through the code to be finally executed on a specific and optimized backend. The API can receive both simple instructions, such as gates and circuits, but also more ellaborated models, for example a VQE. In this chapter a shallow review on the capabilities of Qibo is covered.

[1]Quantum Research Center, Technology Innovation Institute, Abu Dhabi, United Arab Emirates

[2]Qilimanjaro Quantum Tech, Barcelona, Spain

C.1 Circuits

Quantum circuits are the main paradigm for implementing a quantum computing in `Qibo`. The circuits in `Qibo` are an abstract object capable to implement different tools normally used in theoretical descriptions.

Initialization

The circuit object is initialized by defining the number of qubits. Automatically, a quantum state object $|\psi\rangle$ with exactly 2^n complex coefficients is generated. `Qibo` also permits the use of a density matrix ρ, initially defined as $\rho = |\psi\rangle \langle\psi|$, with 4^n complex coefficients. Density matrices imply more costly calculations both in time and in storage memory since there are more numbers to manage. However, the use of density matrices allow represent mixed states, including those affected by noise.

Gates

Quantum operations are added to the circuit subsequently to modify the state. The available operations are of three kinds

- Gates: Standard unitary gates U are mainly used in `Qibo`. These operations modify the quantum state as $U\psi$, and the density matrix as $U\rho U^\dagger$. Gates can be defined to affect one or two qubits. `Qibo` allows to implement on any qubit any gate controlled by an arbitrary set of qubits. In case any gate is parameterized, the circuit incorporates a list of parameters that can be updated in subsequent executions.
- Channels: Quantum channels are the standard tool to introduce decoherence in a quantum circuit. Channels are defined as a set of unitary gates to be applied on a density matrix ρ with different probabilities. As an additional feature, the quantum state representing the circuit is transformed automatically into a density matrix to accomodate the quantum channel.
- Measurements: the measuring step is the only chance to retrieve information from the quantum circuits. In `Qibo`, measurements can be allocated at the end of the circuit to finish the computation, but also on any intermediate step. In the latter case, the circuit is modified accordingly to capture all the outcomes possibilities.

Hamiltonians

Hamiltonians serve the purpose of measuring quantities of interest in the quantum circuit. `Qibo` allows to define a hamiltonian as a full matrix by setting all components, and also from a symbolic description. In the case of quantum circuits, hamiltonians can be used to measured the expected value of quantum state, and also to compare it with the ground state. `Qibo` supports corresponding tools to manage those calculations.

Callbacks

It is often of interest to study the change of different quantities along the execution of a given circuit. To carry this task, Qibo implements the callbacks functionality allowing to extract those quantities when possible at any point of the circuit. Interesting and commonly used callbacks are expected values with respect to some hamiltonian, but entanglement entropy, overlap with respect to a target state or norm of the quantum state are also included.

Callbacks can only be used in simulation backends since any measurement on an actual device would destroy the computation. This is however a useful tool to develop proof-of-concept models to scale later.

Models

Different circuit models play the role of pre-defined architectures with specific functionalities for solving particular problems. The prominent examples of VQE [Per+14] and QAOA [FGG14] are included, but also less known examples as the qPDF [Pér+21a].

Optimization

Many models defined on quantum circuits depend on VQAs where some optimization method is required. Built-in classical optimizers are available in Qibo to be used in all methods. The optimizers are both gradient-descent [KB17; Nie15], quasi-newton [Byr+95; Vir+20], or genetic [Han06] algorithms.

For simulation backends there is an available option to parallelize the different pieces of optimization across different computation cores, improving the overall performance of the process.

C.2 Adiabatic computing

The adiabatic computing section of Qibo performs all necessary calculations to gradually modify a quantum state according to an adiabatic scheme [Far+00]. In adiabatic computing, the ground state of a known hamiltonian is taken as the starting line. Then, a time-dependent hamiltonian is triggered in such a way that the starting one is slowly modified into another problem hamiltonian. The ground state of the problem hamiltonian encodes the desired solution to a given problem. Then, the quantum state evolves according to the time-dependent Schrödinger equation. If the time evolution is slow enough, the overlap between the final evolved state and the ground state of the problem hamiltonian is close to 1.

Hamiltonian

Hamiltonians drive the time evolution in adiabatic quantum computing. Qibo uses a Trotter implementation [Pae+19] of the hamiltonians of interest to execute them more efficiently than if all information is conserved along the different steps.

Solver

Numerically solving the Schrödinger equation is far from being trivial. `Qibo` incorporates different solvers to perform the evolution in several manners. Those solvers include the exponential and trotterized exponential evolution, and Runge-Kutta methods.

C.3 Backends

Backends are the main strength of `Qibo` when competing against other similar libraries. The organization behind `Qibo` plans to add quantum machines and approximate simulation methods to the available backends. However, at the time of writing this thesis, `Qibo` can only be executed on exact classical simulators.

C.3.1 Classical Simulation - Hardware acceleration

`Qibo` was in origin designed to be easily used and to perform efficient computation. The most prominent feature for accomplishing this task is the capability of `Qibo` to adapt its management of computation to the environment. Depending essentially on the size of the quantum circuit to be simulated, the execution can be performed on several hardware simulations:

- Single-thread CPU: Central Processing Units (CPU) are the main component of a classical computer. They are used as general-purpose machines. For `Qibo`, this mode means that all calculation are passed to only one CPU core. It is convenient to use this mode for small circuits, up to 15 qubits, since the memory storage does not need to be extremely large, and the overhead of using other methods do not compensate the gain.
- Single-thread GPU: Graphical Processing Units (GPU) were developed for ML for their capability to perform linear algebra operations. This feature is now useful for simulating quantum circuits efficiently. In this mode, the CPU passes the data to GPU to perform the calculation. This translates into an overhead in the computation. Only for moderately large circuits this overhead is compensated by the more efficient calculation of a GPU. The ideal range is between 15 and 30 qubits. Over that size, current GPUs do not have enough memory to execute the calculation.
- Multi-thread CPU: this method is essentially useful for executing quantum circuits whose storage requirements exceed the capabilities of one CPU. The recommended range is for more than 15 qubits. The performance does not in general overcome a GPU.

- Multi-thread GPU: this is the most special mode, and exclusive of `Qibo`. Computational efforts are distributed accross many GPUs with a significant overhead. This method is only efficient for simulating large circuits, over 30 qubits.

QIBOJIT backend

Just-In-Time (JIT) compilation is a computational framework that optimizes the execution of a given algorithm in subsequent runs of it. In the case of `Qibo`, the JIT compilation is assisted by `cupy` [Oku+17] and `numba` [LPS15]

For using the JIT backend, custom operators optimized for this functioning scheme were developed and implemented in `Qibo`

QIBOTF backend

The first release of `Qibo` was built on `Tensorflow` [Mar+15]. The QIBOTF backend supports custom operations to be implemented using the general `Tensorflow` framework.

This backend support hardware management as inherited from `Tensorflow`. Some operations are implemented faster than in the original code, but in exchange some features, specially for automatic differentiation, were lost in the process.

TENSORFLOW and NUMPY backends

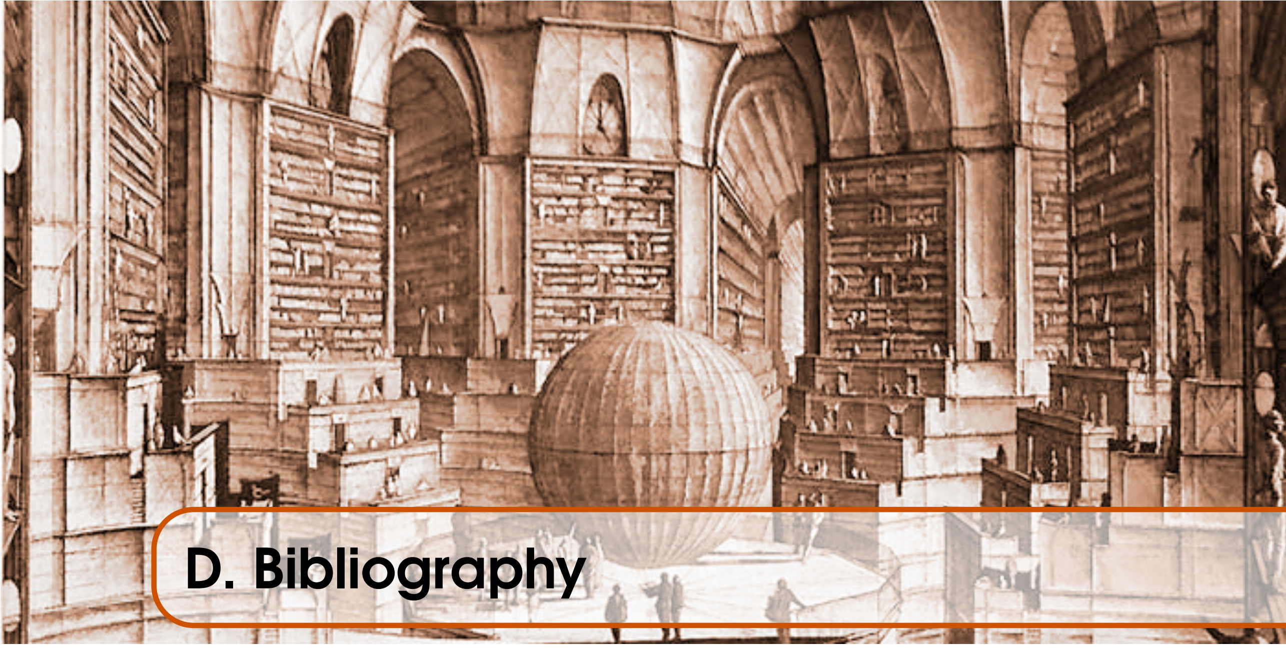
In both cases, the calculations are performed using the `einsum` method of the corresponding `Tensorflow` or `Numpy` package [Har+20]. All functionalities are taken from the parent libraries.

C.4 Examples

`Qibo` looks forward to creating a community of users contributing to the development of the open-source project. As a starting line of extended collaboration, several examples were developed by the creators of `Qibo`

- Examples inspired in the data re-uploading strategy explored in Ch. 3 [Pér+20a; Pér+21a].
- Examples on the unary strategy from Ch. 4 [Ram+21]
- Measuring the tangle of a three-qubit state [Pér+20b].
- The performance of a VQE when solving a condensed matter problem as in Ref. [Bra+20a]
- Examples of Grover's algorithm [Gro96] for a 3SAT problem [GJ90] and cryptographic applications [Ber+08].
- A quantum autoencoder [Bra21]
- The Quantum Singular Value Decomposer [BGL20]
- Adiabatic evolutions for a exact Cover problem [GJ90]
- Shor's factorization algorithm [Sho97] and a version requiring less qubits [Bea03]

The list of available examples is expected to grow in the short term.



D. Bibliography

El Universo (que otros llaman la Biblioteca) se compone de un número indefinido, tal vez infinito, de...

Jorge Luis Borges

Books

[AB09] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009. ISBN: 978-0521424264.

[Ash72] R. B. Ash. *Real Analysis and Probability*. Elsevier, 1972. ISBN: 978-0120652013. DOI: 10.1016/c2013-0-06164-6.

[DHS12] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley, 2012. ISBN: 978-1118586006.

[DS19] K. Du and M. Swamy. *Neural Networks and Statistical Learning*. Springer London, 2019. ISBN: 978-1447174523.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[GJ90] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. USA: W. H. Freeman & Co., 1990. ISBN: 978-0716710455. DOI: 10.5555/574848.

[Got97] D. Gottesman. *Stabilizer codes and quantum error correction.* California Institute of Technology, 1997. arXiv: quant-ph/9705052.

[Hel76] C. W. Helstrom. *Quantum detection and estimation theory / Carl W. Helstrom.* English. Academic Press New York, 1976. ISBN: 978-0123400503.

[HM85] F. Halzen and A. D. Martin. *Quark & Leptons: An Introductory Course In Modern Particle Physics.* John Wiley & Sons, 1985. ISBN: 978-0471887416.

[Hof92] N. Hoffmann. *Simulation Neuronaler Netze.* Vieweg+Teubner Verlag, 1992. ISBN: 978-3322832009. DOI: 10.1007/978-3-322-83200-9.

[Jai17] P. Jain. *Non-convex optimization for machine learning.* Hanover, Massachusetts: Now Publishers, 2017. ISBN: 978-1680833683.

[MCM13] R. Michalski, J. Carbonell, and T. Mitchell. *Machine Learning: An Artificial Intelligence Approach.* Symbolic Computation. Springer Berlin Heidelberg, 2013. ISBN: 978-3662124055.

[Mit+97] T. M. Mitchell et al. *Machine learning.* McGraw Hill, 1997. ISBN: 978-0070428077.

[NC10] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, 2010. ISBN: 978-1107002173. DOI: 10.1017/CBO9780511976667.

[Nie15] M. A. Nielsen. *Neural networks and deep learning.* Vol. 25. Determination press USA, 2015.

[NW06] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer New York, 2006. DOI: 10.1007/978-0-387-40065-5.

[Par17] P. M. Pardalos. *Non-convex multi-objective optimization.* Cham: Springer International Publishing Imprint Springer, 2017. ISBN: 978-3319610054.

[Pre+86] W. H. Press et al. *Numerical Recipes: The Art of Scientific Computing.* USA: Cambridge University Press, 1986. ISBN: 978-0521308119. DOI: 10.5555/6771.

[Rus10] S. Russell. *Artificial intelligence: a modern approach.* Upper Saddle River, New Jersey: Prentice Hall, 2010. ISBN: 978-0136042594.

[Spa05] J. C. Spall. *Introduction to stochastic search and optimization: estimation, simulation, and control.* Vol. 65. John Wiley & Sons, 2005.

[Sut13] I. Sutskever. *Training recurrent neural networks.* University of Toronto, Canada, 2013. ISBN: 978-0499220660. DOI: `10.5555/2604780`.

[Wei74] A. J. Weir. *General integration and measure.* Vol. 2. CUP Archive, 1974. ISBN: 978-0521297158.

Articles

[Aad+15] G. Aad et al. *Measurement of the inclusive jet cross-section in proton-proton collisions at $s = \sqrt{7}$ TeV using 4.5 fb^{-1} of data with the ATLAS detector.* Journal of High Energy Physics 2015.2 (Feb. 2015). DOI: `10.1007/jhep02(2015)153`.

[Aai+12] R. Aaij et al. *Inclusive W and Z production in the forward region at $\sqrt{7}$TeV.* Journal of High Energy Physics 2012.6 (June 2012). DOI: `10.1007/jhep06(2012)058`.

[AB08] D. Aharonov and M. Ben-Or. *Fault-Tolerant Quantum Computation with Constant Error Rate.* SIAM Journal on Computing 38.4 (Jan. 2008). DOI: `10.1137/s0097539799359385`.

[AHS85a] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. *A Learning Algorithm for Boltzmann Machines.* Cognitive Science 9.1 (Jan. 1985). DOI: `10.1207/s15516709cog0901_7`.

[AHS85b] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. *A Learning Algorithm for Boltzmann Machiness.* Cognitive Science 9.1 (Jan. 1985). DOI: `10.1207/s15516709cog0901_7`.

[Ale+19b] A. Alexandru et al. *Gluon field digitization for quantum computers.* Physical Review D 100.11 (Dec. 2019). DOI: `10.1103/physrevd.100.114501`.

[Alw+14] J. Alwall et al. *The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations.* Journal of High Energy Physics 2014.7 (July 2014). DOI: `10.1007/jhep07(2014)079`.

[Ama98] S.-i. Amari. *Natural Gradient Works Efficiently in Learning.* Neural Computation 10.2 (Feb. 1998). DOI: `10.1162/089976698300017746`.

[AR20] S. Aaronson and P. Rall. *Quantum Approximate Counting, Simplified. Symposium on Simplicity in Algorithms.* Society for Industrial and Applied Mathematics, (Jan. 2020), pp. 24–32. DOI: `10.1137/1.9781611976014.5`.

[Aru+19] F. Arute et al. *Quantum supremacy using a programmable superconducting processor.* Nature 574.7779 (Oct. 2019). DOI: 10.1038/s41586-019-1666-5.

[AS03] M. Atiyah and P. Sutcliffe. *Polyhedra in Physics, Chemistry and Geometry.* Milan Journal of Mathematics 71.1 (Sept. 2003). DOI: 10.1007/s00032-003-0014-1.

[Bab+18] R. Babbush et al. *Encoding Electronic Spectra in Quantum Circuits with Linear T Complexity.* Physical Review X 8.4 (Oct. 2018). DOI: 10.1103/physrevx.8.041015.

[Bal+09] R. D. Ball et al. *A determination of parton distributions with faithful uncertainty estimation.* Nuclear Physics B 809.1-2 (Mar. 2009). DOI: 10.1016/j.nuclphysb.2008.09.037.

[Bal+10] R. D. Ball et al. *A first unbiased global NLO determination of parton distributions and their uncertainties.* Nuclear Physics B 838.1-2 (Oct. 2010). DOI: 10.1016/j.nuclphysb.2010.05.008.

[Bal+15] R. D. Ball et al. *Parton distributions for the LHC run II.* Journal of High Energy Physics 2015.4 (Apr. 2015). DOI: 10.1007/jhep04(2015)040.

[Bal+17] R. D. Ball et al. *Parton distributions from high-precision collider data.* The European Physical Journal C 77.10 (Oct. 2017). DOI: 10.1140/epjc/s10052-017-5199-5.

[Ban29] S. Banach. *Sur les fonctionnelles linéaires II.* Studia Mathematica 1 (1929).

[Bar+18] P. K. Barkoutsos et al. *Quantum algorithms for electronic structure calculations: Particle-hole Hamiltonian and optimized wave-function expansions.* Physical Review A 98.2 (Aug. 2018). DOI: 10.1103/physreva.98.022322.

[Bar+95] A. Barenco et al. *Elementary gates for quantum computation.* Physical Review A 52.5 (Nov. 1995). DOI: 10.1103/physreva.52.3457.

[BCH17] V. Bertone, S. Carrazza, and N. P. Hartland. *APFELgrid: a high performance tool for parton density determinations.* Comput. Phys. Commun. 212 (2017). DOI: 10.1016/j.cpc.2016.10.006.

[Bea03] S. Beauregard. *Circuit for Shor's Algorithm Using 2n+3 Qubits.* Quantum Info. Comput. 3.2 (Mar. 2003). DOI: 10.5555/2011517.2011525.

[Ben+02] A. Ben-Hur et al. *Support Vector Clustering.* The Journal of Machine Learning Research 2 (Mar. 2002). DOI: 10.5555/944790.944807.

[Ben+92] C. H. Bennett et al. *Experimental quantum cryptography.* Journal of Cryptology 5.1 (Jan. 1992). DOI: 10.1007/bf00191318.

[Ben18] P. Bentley. *The end of Moore's Law: what happens next?* BBC Focus (May 2018).

[Bep+21] K. Bepari et al. *Towards a quantum computing algorithm for helicity amplitudes and parton showers.* Physical Review D 103.7 (Apr. 2021). DOI: 10.1103/physrevd.103.076020.

[Ber+08] D. J. Bernstein et al. *ChaCha, a variant of Salsa20.* Workshop record of SASC. Vol. 8. (2008), pp. 3–5.

[BG16] S. Bravyi and D. Gosset. *Improved Classical Simulation of Quantum Circuits Dominated by Clifford Gates.* Physical Review Letters 116 (June 2016). DOI: 10.1103/PhysRevLett.116.250501.

[BGL20] C. Bravo-Prieto, D. García-Martín, and J. I. Latorre. *Quantum singular value decomposer.* Physical Review A 101.6 (June 2020). DOI: 10.1103/physreva.101.062310.

[Bia+10] R. C. Bialczak et al. *Quantum process tomography of a universal entangling gate implemented with Josephson phase qubits.* Nature Physics 6.6 (Apr. 2010). DOI: 10.1038/nphys1639.

[Bra+02] G. Brassard et al. *Quantum amplitude amplification and estimation* (2002). DOI: 10.1090/conm/305/05215.

[Bra+20a] C. Bravo-Prieto et al. *Scaling of variational quantum circuit depth for condensed matter systems.* Quantum 4 (May 2020). DOI: 10.22331/q-2020-05-28-272.

[Bra21] C. Bravo-Prieto. *Quantum autoencoders with enhanced data encoding.* Machine Learning: Science and Technology 2.3 (July 2021). DOI: 10.1088/2632-2153/ac0616.

[Bro+11] K. R. Brown et al. *Single-qubit-gate error below* 10^{-4} *in a trapped ion.* Physical Review A 84.3 (Sept. 2011). DOI: 10.1103/physreva.84.030303.

[BS73] F. Black and M. Scholes. *The Pricing of Options and Corporate Liabilities.* Journal of Political Economy 81.3 (May 1973). DOI: 10.1086/260062.

[BV97] E. Bernstein and U. Vazirani. *Quantum complexity theory.* SIAM Journal on computing 26.5 (1997).

[Byr+95] R. H. Byrd et al. *A Limited Memory Algorithm for Bound Constrained Optimization.* SIAM Journal on Scientific Computing 16.5 (Sept. 1995). DOI: 10.1137/0916069.

[Cár+18] F. A. Cárdenas-López et al. *Multiqubit and multilevel quantum reinforcement learning with quantum technologies.* PLOS ONE 13.7 (July 2018). DOI: 10.1371/journal.pone.0200455.

[Car+19] G. Carleo et al. *Machine learning and the physical sciences.* Reviews of Modern Physics 91.4 (Dec. 2019). DOI: 10.1103/revmodphys.91.045002.

[Car+20] S. Carrazza et al. *PineAPPL: combining EW and QCD corrections for fast evaluation of LHC processes.* Journal of High Energy Physics 2020.12 (Dec. 2020). DOI: 10.1007/jhep12(2020)108.

[Car66] L. Carleson. *On convergence and growth of partial sums of Fourier series.* Acta Math. 116 (1966). DOI: 10.1007/BF02392815.

[CC19] S. Carrazza and J. Cruz-Martinez. *Towards a new generation of parton densities with deep learning models.* The European Physical Journal C 79.8 (Aug. 2019). DOI: 10.1140/epjc/s10052-019-7197-2.

[CCL19] I. Cong, S. Choi, and M. D. Lukin. *Quantum convolutional neural networks.* Nature Physics 15.12 (Aug. 2019). DOI: 10.1038/s41567-019-0648-8.

[Ce18] Z.-Y. Chen and *et al. 64-qubit quantum circuit simulation.* Science Bulletin 63.15 (Aug. 2018). DOI: 10.1016/j.scib.2018.06.007.

[CEB20] J. Cook, S. Eidenbenz, and A. Bartschi. *The Quantum Alternating Operator Ansatz on Maximum k-Vertex Cover.* 2020 IEEE International Conference on Quantum Computing and Engineering (QCE). IEEE, (Oct. 2020). DOI: 10.1109/qce49297.2020.00021.

[Cer+21a] M. Cerezo et al. *Sub-quantum Fisher information.* Quantum Science and Technology 6.3 (June 2021). DOI: 10.1088/2058-9565/abfbef.

[Cer+21b] M. Cerezo et al. *Cost function dependent barren plateaus in shallow parametrized quantum circuits.* Nature Communications 12.1 (Mar. 2021). DOI: 10.1038/s41467-021-21728-w.

[Cer18] A. Cervera-Lierta. *Exact Ising model simulation on a quantum computer.* Quantum 2 (Dec. 2018). DOI: 10.22331/q-2018-12-21-114.

[CFH20] A. Candido, S. Forte, and F. Hekhorn. *Can $\overline{MS}$ parton distributions be negative?* Journal of High Energy Physics 2020.11 (Nov. 2020). DOI: 10.1007/jhep11(2020)129.

[Che+20] S. Y.-C. Chen et al. *Variational Quantum Circuits for Deep Reinforcement Learning.* IEEE Access 8 (2020). DOI: 10.1109/access.2020.3010470.

[CHH02] M. Campbell, A. Hoane, and F.-h. Hsu. *Deep Blue.* Artificial Intelligence 134.1-2 (Jan. 2002). DOI: 10.1016/s0004-3702(01)00129-1.

[Cho+10] J. M. Chow et al. *Optimized driving of superconducting artificial atoms for improved single-qubit gates.* Phys. Rev. A 82 (Oct. 2010). DOI: 10.1103/PhysRevA.82.040305.

[CN19] J. Campbell and T. Neumann. *Precision phenomenology with MCFM.* Journal of High Energy Physics 2019.12 (Dec. 2019). DOI: 10.1007/jhep12(2019)034.

[Cor+98] D. G. Cory et al. *Experimental Quantum Error Correction.* Physical Review Letters 81.10 (Sept. 1998). DOI: 10.1103/physrevlett.81.2152.

[Coy+20] B. Coyle et al. *The Born supremacy: quantum advantage and training of an Ising Born machine.* npj Quantum Information 6.1 (July 2020). DOI: 10.1038/s41534-020-00288-9.

[CV95] C. Cortes and V. Vapnik. *Support-vector networks.* Machine Learning 20.3 (Sept. 1995). DOI: 10.1007/bf00994018.

[CWZ21] S. Cheng, L. Wang, and P. Zhang. *Supervised learning with projected entangled pair states.* Physical Review B 103.12 (Mar. 2021). DOI: 10.1103/physrevb.103.125117.

[Cyb89] G. Cybenko. *Approximation by superpositions of a sigmoidal function.* Mathematics of Control, Signals, and Systems 2.4 (Dec. 1989). DOI: 10.1007/bf02551274.

[De07] K. De Raedt and *et al. Massively parallel quantum computer simulator.* Computer Physics Communications 176.2 (2007). DOI: 10.1016/j.cpc.2006.08.007.

[De19] J. Doi and *et al. Quantum Computing Simulator on a Heterogenous HPC System.* Proceedings of the 16th ACM International Conference on Computing Frontiers. CF '19. Alghero, Italy: Association for Computing Machinery, (2019), pp. 85–93. ISBN: 9781450366854. DOI: 10.1145/3310273.3323053.

[Deh57] H. G. Dehmelt. *Slow Spin Relaxation of Optically Polarized Sodium Atoms.* Physical Review 105.5 (Mar. 1957). DOI: 10.1103/physrev.105.1487.

[Den12] L. Deng. *The mnist database of handwritten digit images for machine learning research.* IEEE Signal Processing Magazine 29.6 (2012).

[Deu85] D. Deutsch. *Quantum theory, the Church–Turing principle and the universal quantum computer.* Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences 400.1818 (1985).

[Dir29] P. G. L. Dirichlet. *Sur la convergence des séries trigonométriques qui servent à représenter une fonction arbitraire entre des limites données.* Journal für die reine und angewandte Mathematik 4 (1829).

[Dix+21] V. Dixit et al. *Training Restricted Boltzmann Machines With a D-Wave Quantum Annealer.* Frontiers in Physics 9 (June 2021). DOI: 10.3389/fphy.2021.589626.

[DK18] P.-L. Dallaire-Demers and N. Killoran. *Quantum generative adversarial networks.* Physical Review A 98.1 (July 2018). DOI: 10.1103/physreva.98.012324.

[DM20] T. Dutta and M. Mukherjee. *A single atom noise probe operating beyond the Heisenberg limit.* npj Quantum Information 6.1 (Jan. 2020). DOI: 10.1038/s41534-019-0234-z.

[DN06] C. M. Dawson and M. A. Nielsen. *The Solovay-Kitaev Algorithm.* Quantum Info. Comput. 6.1 (Jan. 2006). DOI: 10.5555/2011679.2011685.

[Don+08] D. Dong et al. *Quantum Reinforcement Learning.* IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 38.5 (Oct. 2008). DOI: 10.1109/tsmcb.2008.925743.

[DPS03] G. M. D'Ariano, M. G. Paris, and M. F. Sacchi. *Quantum Tomography. Advances in Imaging and Electron Physics.* Elsevier, (2003), pp. 205–308. DOI: 10.1016/s1076-5670(03)80065-4.

[DTB16] V. Dunjko, J. M. Taylor, and H. J. Briegel. *Quantum-Enhanced Machine Learning.* Physical Review Letters 117.13 (Sept. 2016). DOI: 10.1103/physrevlett.117.130501.

[DTB17] V. Dunjko, J. M. Taylor, and H. J. Briegel. *Advances in quantum reinforcement learning.* 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, (Oct. 2017). DOI: 10.1109/smc.2017.8122616.

[Du+20] Y. Du et al. *Expressive power of parametrized quantum circuits.* Physical Review Research 2.3 (July 2020). DOI: 10.1103/physrevresearch.2.033125.

[DVC00] W. Dür, G. Vidal, and J. I. Cirac. *Three qubits can be entangled in two inequivalent ways*. Physical Review A 62.6 (Nov. 2000). DOI: 10.1103/physreva.62.062314.

[Eas+21] P. Easom-Mccaldin et al. *On Depth, Robustness and Performance Using the Data Re-Uploading Single-Qubit Classifier*. IEEE Access 9 (2021). DOI: 10.1109/access.2021.3075492.

[EBL18] S. Endo, S. C. Benjamin, and Y. Li. *Practical Quantum Error Mitigation for Near-Future Applications*. Physical Review X 8.3 (July 2018). DOI: 10.1103/physrevx.8.031027.

[Eic68] M. Eichler. *A new proof of the Baker-Campbell-Hausdorff formula*. Journal of the Mathematical Society of Japan 20.1-2 (Apr. 1968). DOI: 10.2969/jmsj/02010023.

[Est02] V. Estivill-Castro. *Why so many clustering algorithms*. ACM SIGKDD Explorations Newsletter 4.1 (June 2002). DOI: 10.1145/568574.568575.

[Fe18] E. S. Fried and *et al. qTorch: The quantum tensor contraction handler*. PLOS ONE 13.12 (Dec. 2018). Ed. by I. Hen. DOI: 10.1371/journal.pone.0208510.

[Fey82] R. P. Feynman. *Simulating physics with computers*. International Journal of Theoretical Physics 21.6-7 (June 1982). DOI: 10.1007/bf02650179.

[Fey88] R. P. Feynman. *The Behavior of Hadron Collisions at Extreme Energies. Special Relativity and Quantum Theory*. Springer Netherlands, (1988), pp. 289–304. DOI: 10.1007/978-94-009-3051-3_25.

[FG20] L. Franken and B. Georgiev. *Explorations in Quantum Neural Networks with Intermediate Measurements*. ESANN. (2020), pp. 297–302.

[Fis22] R. Fisher. *On the mathematical foundations of theoretical statistics*. Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 222.594-604 (Jan. 1922). DOI: 10.1098/rsta.1922.0009.

[FMM+17] C. Figgatt, D. Maslov, C. Monroe, et al. *Complete 3-Qubit Grover search on a programmable quantum computer*. Nature Communications 8.1 (Dec. 2017). DOI: 10.1038/s41467-017-01904-7.

[Gar+20] B. T. Gard et al. *Efficient symmetry-preserving state preparation circuits for the variational quantum eigensolver algorithm.* npj Quantum Information 6.1 (Jan. 2020). DOI: 10.1038/s41534-019-0240-1.

[Gee+13] K. Geerlings et al. *Demonstrating a Driven Reset Protocol for a Superconducting Qubit.* Phys. Rev. Lett. 110 (Mar. 2013). DOI: 10.1103/PhysRevLett.110.120501.

[GLM08] V. Giovannetti, S. Lloyd, and L. Maccone. *Quantum Random Access Memory.* Physical Review Letters 100.16 (Apr. 2008). DOI: 10.1103/physrevlett.100.160501.

[Gra+09] A. Graves et al. *A Novel Connectionist System for Unconstrained Handwriting Recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence 31.5 (May 2009). DOI: 10.1109/tpami.2008.137.

[Gri+21] D. Grinko et al. *Iterative quantum amplitude estimation.* npj Quantum Information 7.1 (Mar. 2021). DOI: 10.1038/s41534-021-00379-1.

[Gro96] L. K. Grover. *A fast quantum mechanical algorithm for database search.* Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96. ACM Press, (1996). DOI: 10.1145/237814.237866.

[GT07] N. Gisin and R. Thew. *Quantum communication.* Nature Photonics 1.3 (Mar. 2007). DOI: 10.1038/nphoton.2007.22.

[GTN21] T. Goto, Q. H. Tran, and K. Nakajima. *Universal Approximation Property of Quantum Machine Learning Models in Quantum-Enhanced Feature Spaces.* Physical Review Letters 127.9 (Aug. 2021). DOI: 10.1103/physrevlett.127.090506.

[Had+19] S. Hadfield et al. *From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz.* Algorithms 12.2 (Feb. 2019). DOI: 10.3390/a12020034.

[Hah27] H. Hahn. *Über lineare Gleichungssysteme in linearen Räumen.* Journal für die reine und angewandte Mathematik 157 (1927).

[Hal15] B. C. Hall. *The Baker–Campbell–Hausdorff Formula and Its Consequences. Graduate Texts in Mathematics.* Springer International Publishing, (2015), pp. 109–137. DOI: 10.1007/978-3-319-13467-3_5.

[Han06] N. Hansen. *The CMA Evolution Strategy: A Comparing Review. Towards a New Evolutionary Computation.* Springer Berlin Heidelberg, (2006), pp. 75–102. DOI: `10.1007/3-540-32494-1_4`.

[Har+20] C. R. Harris et al. *Array programming with NumPy.* Nature 585.7825 (Sept. 2020). DOI: `10.1038/s41586-020-2649-2`.

[Hav+19] V. Havlíček et al. *Supervised learning with quantum-enhanced feature spaces.* Nature 567.7747 (Mar. 2019). DOI: `10.1038/s41586-019-0980-2`.

[HD21a] P. Huembeli and A. Dauphin. *Characterizing the loss landscape of variational quantum circuits.* Quantum Science and Technology 6.2 (Feb. 2021). DOI: `10.1088/2058-9565/abdbc9`.

[HD21b] P. Huembeli and A. Dauphin. *Characterizing the loss landscape of variational quantum circuits.* Quantum Science and Technology 6.2 (Feb. 2021). DOI: `10.1088/2058-9565/abdbc9`.

[Heb+17] M. Hebenstreit et al. *Compressed quantum computation using a remote five-qubit quantum computer.* Physical Review A 95.5 (May 2017). DOI: `10.1103/physreva.95.052339`.

[Hem+18] C. Hempel et al. *Quantum Chemistry Calculations on a Trapped-Ion Quantum Simulator.* Physical Review X 8.3 (July 2018). DOI: `10.1103/physrevx.8.031022`.

[HHL09] A. W. Harrow, A. Hassidim, and S. Lloyd. *Quantum Algorithm for Linear Systems of Equations.* Physical Review Letters 103.15 (Oct. 2009). DOI: `10.1103/physrevlett.103.150502`.

[HMS05] B. Hammer, A. Micheli, and A. Sperduti. *Universal Approximation Capability of Cascade Correlation for Structures.* Neural Computation 17.5 (May 2005). DOI: `10.1162/0899766053491878`.

[HN21] A. W. Harrow and J. C. Napp. *Low-Depth Gradient Measurements Can Improve Convergence in Variational Hybrid Quantum-Classical Algorithms.* Physical Review Letters 126.14 (Apr. 2021). DOI: `10.1103/physrevlett.126.140502`.

[Hol73] A. Holevo. *Bounds for the quantity of information transmitted by a quantum communication channel.* (1973), pp. 177–183.

[Hor+20] N. V. Horne et al. *Single-atom energy-conversion device with a quantum load.* npj Quantum Information 6.1 (May 2020). DOI: `10.1038/s41534-020-0264-6`.

[Hor91] K. Hornik. *Approximation capabilities of multilayer feedforward networks.* Neural Networks 4.2 (1991). DOI: `10.1016/0893-6080(91)90009-t`.

[HR96] S. Haroche and J.-M. Raimond. *Quantum Computing: Dream or Nightmare?* Physics Today 49.8 (Aug. 1996). DOI: 10.1063/1.881512.

[Hua+21a] H.-Y. Huang et al. *Power of data in quantum machine learning.* Nature Communications 12.1 (May 2021). DOI: 10.1038/s41467-021-22539-9.

[IOL07] S. Iblisdir, R. Orús, and J. I. Latorre. *Matrix product states algorithms and continuous systems.* Physical Review B 75.10 (Mar. 2007). DOI: 10.1103/physrevb.75.104305.

[Itô44] K. Itô. *Stochastic integral.* Proceedings of the Imperial Academy 20.8 (1944).

[JB20] T. Jones and S. Benjamin. *QuESTlink—Mathematica embiggened by a hardware-optimised quantum emulator.* Quantum Science and Technology 5.3 (May 2020). DOI: 10.1088/2058-9565/ab8506.

[Jer+21] S. Jerbi et al. *Quantum Enhancements for Deep Reinforcement Learning in Large Spaces.* PRX Quantum 2.1 (Feb. 2021). DOI: 10.1103/prxquantum.2.010328.

[Jon+19] T. Jones et al. *QuEST and High Performance Simulation of Quantum Computers.* Scientific Reports 9.1 (July 2019). DOI: 10.1038/s41598-019-47174-9.

[Jum+21] J. Jumper et al. *Highly accurate protein structure prediction with AlphaFold.* Nature (July 2021). DOI: 10.1038/s41586-021-03819-2.

[Kan+19] M.-S. Kang et al. *Implementation of SWAP test for two unknown states in photons via cross-Kerr nonlinearities under decoherence effect.* Scientific Reports 9.1 (Apr. 2019). DOI: 10.1038/s41598-019-42662-4.

[Kha+15] V. Khachatryan et al. *Measurement of the Z boson differential cross section in transverse momentum and rapidity in proton-proton collisions at 8 TeV.* Physics Letters B 749 (Oct. 2015). DOI: 10.1016/j.physletb.2015.07.065.

[Kit03] A. Kitaev. *Fault-tolerant quantum computation by anyons.* Annals of Physics 303.1 (Jan. 2003). DOI: 10.1016/s0003-4916(02)00018-0.

[KL51] S. Kullback and R. A. Leibler. *On Information and Sufficiency.* The Annals of Mathematical Statistics 22.1 (Mar. 1951). DOI: 10.1214/aoms/1177729694.

[KLM96] L. P. Kaelbling, M. L. Littman, and A. W. Moore. *Reinforcement Learning: A Survey.* Journal of Artificial Intelligence Research 4 (May 1996). DOI: 10.1613/jair.301.

[Kni+08] E. Knill et al. *Randomized benchmarking of quantum gates.* Physical Review A 77.1 (Jan. 2008). DOI: 10.1103/physreva.77.012307.

[Kra91] M. A. Kramer. *Nonlinear principal component analysis using autoassociative neural networks.* AIChE Journal 37.2 (Feb. 1991). DOI: 10.1002/aic.690370209.

[KV90] A. Kemna and A. Vorst. *A pricing method for options based on average asset values.* Journal of Banking & Finance 14.1 (Mar. 1990). DOI: 10.1016/0378-4266(90)90039-5.

[KW19] D. P. Kingma and M. Welling. *An Introduction to Variational Autoencoders.* Foundations and Trends® in Machine Learning 12.4 (2019). DOI: 10.1561/2200000056.

[Lam17] L. Lamata. *Basic protocols in quantum reinforcement learning with superconducting circuits.* Scientific Reports 7.1 (May 2017). DOI: 10.1038/s41598-017-01711-6.

[Lan95] R. Landauer. *Is quantum mechanics useful?* Philosophical Transactions of the Royal Society of London. Series A: Physical and Engineering Sciences 353.1703 (Dec. 1995). DOI: 10.1098/rsta.1995.0106.

[LAT21] Y. Liu, S. Arunachalam, and K. Temme. *A rigorous and robust quantum speed-up in supervised machine learning.* Nature Physics (July 2021). DOI: 10.1038/s41567-021-01287-z.

[LC17] G. H. Low and I. L. Chuang. *Optimal Hamiltonian Simulation by Quantum Signal Processing.* Physical Review Letters 118.1 (Jan. 2017). DOI: 10.1103/physrevlett.118.010501.

[LC19] G. H. Low and I. L. Chuang. *Hamiltonian Simulation by Qubitization.* Quantum 3 (July 2019). DOI: 10.22331/q-2019-07-12-163.

[Les+93] M. Leshno et al. *Multilayer feedforward networks with a nonpolynomial activation function can approximate any function.* Neural Networks 6.6 (Jan. 1993). DOI: 10.1016/s0893-6080(05)80131-5.

[LLa20] H. Lamm, S. Lawrence, and Y. Y. and. *Parton physics on a quantum computer.* Physical Review Research 2.1 (Mar. 2020). DOI: 10.1103/physrevresearch.2.013272.

[LMR14] S. Lloyd, M. Mohseni, and P. Rebentrost. *Quantum principal component analysis.* Nature Physics 10.9 (July 2014). DOI: 10.1038/nphys3029.

[LPS15] S. K. Lam, A. Pitrou, and S. Seibert. *Numba.* Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC - LLVM '15. ACM Press, (2015). DOI: `10.1145/2833157.2833162`.

[LS20] O. Lockwood and M. Si. *Reinforcement learning with quantum variational circuit.* Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment. Vol. 16. 1. (2020), pp. 245–251.

[LS21] O. Lockwood and M. Si. *Playing Atari with Hybrid Quantum-Classical Reinforcement Learning.* NeurIPS 2020 Workshop on Pre-registration in Machine Learning. PMLR. (2021), pp. 285–301.

[LW18a] J.-G. Liu and L. Wang. *Differentiable learning of quantum circuit Born machines.* Physical Review A 98.6 (Dec. 2018). DOI: `10.1103/physreva.98.062324`.

[LW18b] S. Lloyd and C. Weedbrook. *Quantum Generative Adversarial Learning.* Physical Review Letters 121.4 (July 2018). DOI: `10.1103/physrevlett.121.040502`.

[Mar+21] A. Martin et al. *Toward pricing financial derivatives with an IBM quantum computer.* Physical Review Research 3.1 (2021).

[Mas17] D. Maslov. *Basic circuit compilation techniques for an ion-trap quantum machine.* New Journal of Physics 19.2 (Feb. 2017). DOI: `10.1088/1367-2630/aa5e47`.

[MBE21] J. J. Meyer, J. Borregaard, and J. Eisert. *A variational toolbox for quantum multi-parameter estimation.* npj Quantum Information 7.1 (June 2021). DOI: `10.1038/s41534-021-00425-y`.

[McC+18] J. R. McClean et al. *Barren plateaus in quantum neural network training landscapes.* Nature Communications 9.1 (Nov. 2018). DOI: `10.1038/s41467-018-07090-4`.

[McK+17a] D. C. McKay et al. *Efficient Z gates for quantum computing.* Phys. Rev. A 96 (Aug. 2017). DOI: `10.1103/PhysRevA.96.022330`.

[McK+17b] D. C. McKay et al. *Efficient Z gates for quantum computing.* Physical Review A 96.2 (Aug. 2017). DOI: `10.1103/physreva.96.022330`.

[MGE11] E. Magesan, J. M. Gambetta, and J. Emerson. *Scalable and Robust Randomized Benchmarking of Quantum Processes.* Phys. Rev. Lett. 106 (May 2011). DOI: `10.1103/PhysRevLett.106.180504`.

[Mit+18] K. Mitarai et al. *Quantum circuit learning*. Physical Review A 98.3 (Sept. 2018). DOI: 10.1103/physreva.98.032309.

[Mol+18] N. Moll et al. *Quantum optimization using variational algorithms on near-term quantum devices*. Quantum Science and Technology 3.3 (June 2018). DOI: 10.1088/2058-9565/aab822.

[Mon15] A. Montanaro. *Quantum speedup of Monte Carlo methods*. Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 471.2181 (Sept. 2015). DOI: 10.1098/rspa.2015.0301.

[Moo65] G. E. Moore. *Cramming more components onto integrated circuits*. Electronics 38.8 (Apr. 1965). DOI: 10.1109/n-ssc.2006.4785860.

[Mot+09] F. Motzoi et al. *Simple Pulses for Elimination of Leakage in Weakly Nonlinear Qubits*. Phys. Rev. Lett. 103 (Sept. 2009). DOI: 10.1103/PhysRevLett.103.110501.

[MS20] M. Möller and M. Schalkers. *A Cross-Platform Programming Framework for Quantum-Accelerated Scientific Computing*. Computational Science – ICCS 2020. Ed. by V. V. Krzhizhanovskaya et al. Cham: Springer International Publishing, (2020), pp. 451–464. ISBN: 978-3-030-50433-5. DOI: 10.1007/978-3-030-50433-5_35.

[Nac+21] B. Nachman et al. *Quantum Algorithm for High Energy Physics Simulations*. Physical Review Letters 126.6 (Feb. 2021). DOI: 10.1103/physrevlett.126.062001.

[Nag02] Z. Nagy. *Three-Jet Cross Sections in Hadron-Hadron Collisions at Next-To-Leading Order*. Physical Review Letters 88.12 (Mar. 2002). DOI: 10.1103/physrevlett.88.122003.

[Nas84] S. G. Nash. *Newton-Type Minimization via the Lanczos Method*. SIAM Journal on Numerical Analysis 21.4 (Aug. 1984). DOI: 10.1137/0721052.

[Nat16] Nature Editorial. *The rise of quantum materials*. Nature Physics 12.2 (Feb. 2016). DOI: 10.1038/nphys3668.

[NCK+20] Y. Nam, J.-S. Chen, J. Kim, et al. *Ground-state energy estimation of the water molecule on a trapped-ion quantum computer*. npj Quantum Information 6.1 (Apr. 2020). DOI: 10.1038/s41534-020-0259-3.

[Nes83] Y. Nesterov. *A method for unconstrained convex minimization problem with the rate of convergence O (1/k^ 2)*. Doklady an ussr. Vol. 269. (1983), pp. 543–547.

[NM65] J. Nelder and R. Mead. *A Simplex Method for Function Minimization*. Comput. J. 7 (1965).

[NY21] K. Nakaji and N. Yamamoto. *Expressibility of the alternating layered ansatz for quantum computation*. Quantum 5 (Apr. 2021). DOI: 10.22331/q-2021-04-19-434.

[Oku+17] R. Okuta et al. *CuPy: A NumPy-Compatible Library for NVIDIA GPU Calculations*. Proceedings of Workshop on Machine Learning Systems (LearningSys) in The Thirty-first Annual Conference on Neural Information Processing Systems (NIPS). (2017).

[OML19a] R. Orús, S. Mugel, and E. Lizaso. *Forecasting financial crashes with quantum computing*. Physical Review A 99.6 (June 2019). DOI: 10.1103/physreva.99.060301.

[OML19b] R. Orús, S. Mugel, and E. Lizaso. *Quantum computing for finance: Overview and prospects*. Reviews in Physics 4 (Nov. 2019). DOI: 10.1016/j.revip.2019.100028.

[Orú14a] R. Orús. *A practical introduction to tensor networks: Matrix product states and projected entangled pair states*. Annals of Physics 349 (Oct. 2014). DOI: 10.1016/j.aop.2014.06.013.

[Orú14b] R. Orús. *Advances on tensor network theory: symmetries, fermions, entanglement, and holography*. The European Physical Journal B 87.11 (Nov. 2014). DOI: 10.1140/epjb/e2014-50502-9.

[Pae+19] S. Paeckel et al. *Time-evolution methods for matrix-product states*. Annals of Physics 411 (Dec. 2019). DOI: 10.1016/j.aop.2019.167998.

[Pai+11] H. Paik et al. *Observation of High Coherence in Josephson Junction Qubits Measured in a Three-Dimensional Circuit QED Architecture*. Phys. Rev. Lett. 107 (Dec. 2011). DOI: 10.1103/PhysRevLett.107.240501.

[Pea00] K. Pearson. *X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling*. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 50.302 (July 1900). DOI: 10.1080/14786440009463897.

[Ped+11] F. Pedregosa et al. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research 12 (2011).

[Pér+–] A. Pérez-Salinas et al. *Experimental optimization for single-qubit approximants*. Preparation (–).

[Per+14] A. Peruzzo et al. *A variational eigenvalue solver on a photonic quantum processor*. Nature Communications 5.1 (July 2014). DOI: `10.1038/ncomms5213`.

[Per+18] A. Perdomo-Ortiz et al. *Opportunities and challenges for quantum-assisted machine learning in near-term quantum computers*. Quantum Science and Technology 3.3 (June 2018). DOI: `10.1088/2058-9565/aab859`.

[Pér+20a] A. Pérez-Salinas et al. *Data re-uploading for a universal quantum classifier*. Quantum 4 (Feb. 2020). DOI: `10.22331/q-2020-02-06-226`.

[Pér+20b] A. Pérez-Salinas et al. *Measuring the Tangle of Three-Qubit States*. Entropy 22.4 (Apr. 2020). DOI: `10.3390/e22040436`.

[Pér+21a] A. Pérez-Salinas et al. *Determining the proton content with a quantum computer*. Physical Review D 103.3 (Feb. 2021). DOI: `10.1103/physrevd.103.034027`.

[Pér+21b] A. Pérez-Salinas et al. *One qubit as a universal approximant*. Physical Review A 104.1 (July 2021). DOI: `10.1103/physreva.104.012405`.

[Pou+18] D. Poulin et al. *Quantum Algorithm for Spectral Measurement with a Lower Gate Count*. Physical Review Letters 121.1 (July 2018). DOI: `10.1103/physrevlett.121.010501`.

[Pow64] M. J. D. Powell. *An efficient method for finding the minimum of a function of several variables without calculating derivatives*. The Computer Journal 7.2 (Feb. 1964). DOI: `10.1093/comjnl/7.2.155`.

[Pra15] M. L. de Prado. *Generalized Optimal Trading Trajectories: A Financial Quantum Computing Application*. SSRN Electronic Journal (2015). DOI: `10.2139/ssrn.2575184`.

[Pre18] J. Preskill. *Quantum Computing in the NISQ era and beyond*. Quantum 2 (Aug. 2018). DOI: `10.22331/q-2018-08-06-79`.

[PTP19] A. Pepper, N. Tischler, and G. J. Pryde. *Experimental Realization of a Quantum Autoencoder: The Compression of Qutrits via Machine Learning*. Physical Review Letters 122.6 (Feb. 2019). DOI: `10.1103/physrevlett.122.060501`.

[Qia99] N. Qian. *On the momentum term in gradient descent learning algorithms*. Neural Networks 12.1 (Jan. 1999). DOI: `10.1016/s0893-6080(98)00116-6`.

[Ram+21] S. Ramos-Calderer et al. *Quantum unary approach to option pricing*. Physical Review A 103.3 (Mar. 2021). DOI: `10.1103/physreva.103.032414`.

[RGB18] P. Rebentrost, B. Gupt, and T. R. Bromley. *Quantum computational finance: Monte Carlo pricing of financial derivatives.* Physical Review A 98.2 (Aug. 2018). DOI: 10.1103/physreva.98.022321.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning representations by back-propagating errors.* Nature 323.6088 (Oct. 1986). DOI: 10.1038/323533a0.

[Rie14] F. Riesz. *Démonstration nouvelle d'un théorème concernant les opérations fonctionnelles linéaires.* Annales scientifiques de l'École Normale Supérieure. Vol. 31. (1914), pp. 9–14.

[Rie67] B. Riemann. *Über die Darstellbarkeit einer Function durch eine trigonometrische Reihe.* Abhandlungen der Königlichen Gesellschaft der Wissenschaften zu Göttingen 13 (1867).

[RML14] P. Rebentrost, M. Mohseni, and S. Lloyd. *Quantum Support Vector Machine for Big Data Classification.* Physical Review Letters 113.13 (Sept. 2014). DOI: 10.1103/physrevlett.113.130503.

[ROA17] J. Romero, J. P. Olson, and A. Aspuru-Guzik. *Quantum autoencoders for efficient compression of quantum data.* Quantum Science and Technology 2.4 (Aug. 2017). DOI: 10.1088/2058-9565/aa8072.

[Ros+16] G. Rosenberg et al. *Solving the Optimal Trading Trajectory Problem Using a Quantum Annealer.* IEEE Journal of Selected Topics in Signal Processing 10.6 (Sept. 2016). DOI: 10.1109/jstsp.2016.2574703.

[Sav76] L. J. Savage. *On Rereading R. A. Fisher.* The Annals of Statistics 4.3 (May 1976). DOI: 10.1214/aos/1176343456.

[Sch+19] M. Schuld et al. *Evaluating analytic gradients on quantum hardware.* Physical Review A 99.3 (Mar. 2019). DOI: 10.1103/physreva.99.032331.

[Sch+20] M. Schuld et al. *Circuit-centric quantum classifiers.* Physical Review A 101.3 (Mar. 2020). DOI: 10.1103/physreva.101.032308.

[Sch98] J. Schmidhuber. *A General Method for Incremental Self-Improvement and Multi-Agent Learning.* Evolutionary Computation: Theory and Applications. Scientific Publ. Co., Singapore. In. (1998), pp. 81–123.

[Sho95] P. W. Shor. *Scheme for reducing decoherence in quantum computer memory.* Physical Review A 52.4 (Oct. 1995). DOI: 10.1103/physreva.52.r2493.

[Sho96] P. W. Shor. *Fault-tolerant quantum computation.* Proceedings of 37th Conference on Foundations of Computer Science. IEEE. (1996), pp. 56–65.

[Sho97] P. W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer.* SIAM Journal on Computing 26.5 (Oct. 1997). DOI: `10.1137/s0097539795293172`.

[SHT18] D. S. Steiger, T. Häner, and M. Troyer. *ProjectQ: an open source software framework for quantum computing.* Quantum 2 (Jan. 2018). DOI: `10.22331/q-2018-01-31-49`.

[Sil+16] D. Silver et al. *Mastering the game of Go with deep neural networks and tree search.* Nature 529.7587 (Jan. 2016). DOI: `10.1038/nature16961`.

[SJA19] S. Sim, P. D. Johnson, and A. Aspuru-Guzik. *Expressibility and Entangling Capability of Parameterized Quantum Circuits for Hybrid Quantum-Classical Algorithms.* Advanced Quantum Technologies 2.12 (Oct. 2019). DOI: `10.1002/qute.201900070`.

[Spa98] J. C. Spall. *An overview of the simultaneous perturbation method for efficient optimization.* Johns Hopkins apl technical digest 19.4 (1998).

[SS03] N. Schuch and J. Siewert. *Natural two-qubit gate for quantum computation using theXYinteraction.* Physical Review A 67.3 (Mar. 2003). DOI: `10.1103/physreva.67.032301`.

[SSM21] M. Schuld, R. Sweke, and J. J. Meyer. *Effect of data encoding on the expressive power of variational quantum-machine-learning models.* Physical Review A 103.3 (Mar. 2021). DOI: `10.1103/physreva.103.032430`.

[Sta+20] N. Stamatopoulos et al. *Option Pricing using Quantum Computers.* Quantum 4 (July 2020). DOI: `10.22331/q-2020-07-06-291`.

[Ste96] A. M. Steane. *Error Correcting Codes in Quantum Theory.* Physical Review Letters 77.5 (July 1996). DOI: `10.1103/physrevlett.77.793`.

[Sto+20] J. Stokes et al. *Quantum Natural Gradient.* Quantum 4 (May 2020). DOI: `10.22331/q-2020-05-25-269`.

[Suz+20b] Y. Suzuki et al. *Amplitude estimation without phase estimation.* Quantum Information Processing 19.2 (Jan. 2020). DOI: `10.1007/s11128-019-2565-2`.

[SW20] M. Steudtner and S. Wehner. *Estimating exact energies in quantum simulation without Toffoli gates.* Physical Review A 101.5 (May 2020). DOI: 10.1103/physreva.101.052329.

[Swe+20] R. Sweke et al. *Stochastic gradient descent for hybrid quantum-classical optimization.* Quantum 4 (Aug. 2020). DOI: 10.22331/q-2020-08-31-314.

[TBG17] K. Temme, S. Bravyi, and J. M. Gambetta. *Error Mitigation for Short-Depth Quantum Circuits.* Physical Review Letters 119.18 (Nov. 2017). DOI: 10.1103/physrevlett.119.180509.

[TG19] E. Torrontegui and J. J. García-Ripoll. *Unitary quantum perceptron as efficient universal approximator.* EPL (Europhysics Letters) 125.3 (Mar. 2019). DOI: 10.1209/0295-5075/125/30004.

[Tur38] A. M. Turing. *On Computable Numbers, with an Application to the Entscheidungsproblem. A Correction.* Proceedings of the London Mathematical Society s2-43.1 (1938). DOI: 10.1112/plms/s2-43.6.544.

[UB21] A. V. Uvarov and J. D. Biamonte. *On barren plateaus and cost function locality in variational quantum algorithms.* Journal of Physics A: Mathematical and Theoretical 54.24 (May 2021). DOI: 10.1088/1751-8121/abfac7.

[UBY20] A. Uvarov, J. D. Biamonte, and D. Yudin. *Variational quantum eigensolver for frustrated quantum systems.* Physical Review B 102.7 (Aug. 2020). DOI: 10.1103/physrevb.102.075104.

[Unr95] W. G. Unruh. *Maintaining coherence in quantum computers.* Physical Review A 51.2 (Feb. 1995). DOI: 10.1103/physreva.51.992.

[Val+20] M. Valueva et al. *Application of the residue number system to reduce hardware costs of the convolutional neural network implementation.* Mathematics and Computers in Simulation 177 (Nov. 2020). DOI: 10.1016/j.matcom.2020.04.031.

[Vaz02] U. V. Vazirani. *A survey of quantum complexity theory* (2002). DOI: 10.1090/psapm/058/1922899.

[VCL09] F. Verstraete, J. I. Cirac, and J. I. Latorre. *Quantum circuits for strongly correlated quantum systems.* Physical Review A 79.3 (Mar. 2009). DOI: 10.1103/physreva.79.032316.

[Ve19] B. Villalonga and *et al. A flexible high-performance simulator for verifying and benchmarking quantum circuits implemented on real hardware*. npj Quantum Information 5.1 (Oct. 2019). DOI: 10.1038/s41534-019-0196-1.

[Vid04] G. Vidal. *Efficient Simulation of One-Dimensional Quantum Many-Body Systems*. Physical Review Letters 93.4 (July 2004). DOI: 10.1103/physrevlett.93.040502.

[Vid07] G. Vidal. *Entanglement Renormalization*. Physical Review Letters 99.22 (Nov. 2007). DOI: 10.1103/physrevlett.99.220405.

[Vid08] G. Vidal. *Class of Quantum Many-Body States That Can Be Efficiently Simulated*. Physical Review Letters 101.11 (Sept. 2008). DOI: 10.1103/physrevlett.101.110501.

[Vir+20] P. Virtanen et al. *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*. Nature Methods 17 (2020). DOI: 10.1038/s41592-019-0686-2.

[VMC08] F. Verstraete, V. Murg, and J. Cirac. *Matrix product states, projected entangled pair states, and variational renormalization group methods for quantum spin systems*. Advances in Physics 57.2 (Mar. 2008). DOI: 10.1080/14789940801912366.

[VT20] F. J. G. Vidal and D. O. Theis. *Input Redundancy for Parameterized Quantum Circuits*. Frontiers in Physics 8 (Aug. 2020). DOI: 10.3389/fphy.2020.00297.

[VV18] A. Van-Brunt and M. Visser. *Explicit Baker–Campbell–Hausdorff Expansions*. Mathematics 6.8 (Aug. 2018). DOI: 10.3390/math6080135.

[Wal13] S. Wallis. *Binomial Confidence Intervals and Contingency Tests: Mathematical Fundamentals and the Evaluation of Alternative Methods*. Journal of Quantitative Linguistics 20.3 (July 2013). DOI: 10.1080/09296174.2013.799918.

[Wan+17] K. H. Wan et al. *Quantum generalisation of feedforward neural networks*. npj Quantum Information 3.1 (Sept. 2017). DOI: 10.1038/s41534-017-0032-4.

[Wan+20b] Z. Wang et al. *XY mixers: Analytical and numerical results for the quantum alternating operator ansatz*. Physical Review A 101.1 (Jan. 2020). DOI: 10.1103/physreva.101.012320.

[Wan+21a] P. Wang et al. *Single ion qubit with estimated coherence time exceeding one hour*. Nature Communications 12.1 (Jan. 2021). DOI: 10.1038/s41467-020-20330-w.

[WE19] S. Woerner and D. J. Egger. *Quantum risk analysis.* npj Quantum Information 5.1 (Feb. 2019). DOI: 10.1038/s41534-019-0130-6.

[WZ82] W. K. Wootters and W. H. Zurek. *A single quantum cannot be cloned.* Nature 299.5886 (Oct. 1982). DOI: 10.1038/299802a0.

[Yu+19] S. Yu et al. *Reconstruction of a Photonic Qubit State with Reinforcement Learning.* Advanced Quantum Technologies 2.7-8 (Mar. 2019). DOI: 10.1002/qute.201800074.

[Yum+17] D. Yum et al. *Optical barium ion qubit.* Journal of the Optical Society of America B 34.8 (July 2017). DOI: 10.1364/josab.34.001632.

[Zha+19] Z. Zhao et al. *Bayesian deep learning on a quantum computer.* Quantum Machine Intelligence 1.1-2 (May 2019). DOI: 10.1007/s42484-019-00004-7.

[Zho+20] H.-S. Zhong et al. *Quantum computational advantage using photons.* Science 370.6523 (2020). DOI: 10.1126/science.abe8770.

[Zhu+19] D. Zhu et al. *Training of quantum circuits on a hybrid quantum computer.* Science Advances 5.10 (Oct. 2019). DOI: 10.1126/sciadv.aaw9918.

[ZLW19] C. Zoufal, A. Lucchi, and S. Woerner. *Quantum Generative Adversarial Networks for learning and loading random distributions.* npj Quantum Information 5.1 (Nov. 2019). DOI: 10.1038/s41534-019-0223-2.

[ZLW21] C. Zoufal, A. Lucchi, and S. Woerner. *Variational quantum Boltzmann machines.* Quantum Machine Intelligence 3.1 (Feb. 2021). DOI: 10.1007/s42484-020-00033-7.

[ZYL15] P. Zhang, J. Yuan, and X. Lu. *Quantum Computer Simulation on Multi-GPU Incorporating Data Locality.* Algorithms and Architectures for Parallel Processing. Cham: Springer International Publishing, (2015), pp. 241–256. DOI: 10.1007/978-3-319-27119-4_17.

Preprints

[Alt01] M. V. Altaisky. *Quantum neural network.* (2001). arXiv: quant-ph/0107012 [quant-ph].

[Arr+20] A. Arrasmith et al. *Effect of barren plateaus on gradient-free optimization.* (2020). arXiv: 2011.12245 [quant-ph].

[BB17] J. Biamonte and V. Bergholm. *Tensor Networks in a Nutshell.* (2017). arXiv: `1708.00006 [quant-ph]`.

[Bec+20] J. L. Beckey et al. *Variational Quantum Algorithm for Estimating the Quantum Fisher Information.* (2020). arXiv: `2010.10488 [quant-ph]`.

[Bha+21] K. Bharti et al. *Noisy intermediate-scale quantum (NISQ) algorithms.* (2021). arXiv: `2101.08448 [quant-ph]`.

[BK21] L. Bittel and M. Kliesch. *Training variational quantum algorithms is NP-hard – even for logarithmically many qubits and free fermionic systems.* (2021). arXiv: `2101.07267 [quant-ph]`.

[Bra+20b] C. Bravo-Prieto et al. *Variational Quantum Linear Solver.* (2020). arXiv: `1909.05820 [quant-ph]`.

[Con+21] I. Convy et al. *Mutual Information Scaling for Tensor Network Machine Learning.* (2021). arXiv: `2103.00105 [quant-ph]`.

[Cra+19] D. Crawford et al. *Reinforcement Learning Using Quantum Boltzmann Machines.* (2019). arXiv: `1612.05695 [quant-ph]`.

[Cro+17] A. W. Cross et al. *Open Quantum Assembly Language.* (2017). arXiv: `1707.03429 [quant-ph]`.

[DB17] V. Dunjko and H. J. Briegel. *Machine learning & artificial intelligence in the quantum domain.* (2017). arXiv: `1709.02779 [quant-ph]`.

[Dut+21] T. Dutta et al. *Realization of an ion trap quantum classifier.* (2021). arXiv: `2106.14059 [quant-ph]`.

[Eft+20a] S. Efthymiou et al. *Qibo: a framework for quantum simulation with hardware acceleration.* (2020). arXiv: `2009.01845 [quant-ph]`.

[Egg+19] D. J. Egger et al. *Credit Risk Analysis using Quantum Computers.* (2019). arXiv: `1907.03044 [quant-ph]`.

[Far+00] E. Farhi et al. *Quantum Computation by Adiabatic Evolution.* (2000). arXiv: `quant-ph/0001106 [quant-ph]`.

[FC20] S. Forte and S. Carrazza. *Parton distribution functions.* (Aug. 2020). arXiv: `2008.12305 [hep-ph]`.

[FGG14] E. Farhi, J. Goldstone, and S. Gutmann. *A Quantum Approximate Optimization Algorithm.* (2014). arXiv: `1411.4028 [quant-ph]`.

[FN18] E. Farhi and H. Neven. *Classification with Quantum Neural Networks on Near Term Processors.* (2018). arXiv: `1802.06002 [quant-ph]`.

[Gac+21] J. Gacon et al. *Simultaneous Perturbation Stochastic Approximation of the Quantum Fisher Information.* (2021). arXiv: `2103.09232 [quant-ph]`.

[GL18] A. Garcia-Saez and J. I. Latorre. *Addressing hard classical problems with Adiabatically Assisted Variational Quantum Eigensolvers.* (2018). arXiv: `1806.02287 [quant-ph]`.

[Hol+21] Z. Holmes et al. *Connecting ansatz expressibility to gradient magnitudes and barren plateaus.* (2021). arXiv: `2101.02138 [quant-ph]`.

[Hua+21b] H.-Y. Huang et al. *Provably efficient machine learning for quantum many-body problems.* (2021). arXiv: `2106.12627 [quant-ph]`.

[Hub+21] T. Hubregtsen et al. *Single-component gradient rules for variational quantum algorithms.* (2021). arXiv: `2106.01388 [quant-ph]`.

[JDK+20] S. Johri, S. Debnath, I. Kerenidis, et al. *Nearest Centroid Classification on a Trapped Ion Quantum Computer.* (2020). arXiv: `2012.04145 [quant-ph]`.

[KB17] D. P. Kingma and J. Ba. *Adam: A Method for Stochastic Optimization.* (2017). arXiv: `1412.6980 [cs.LG]`.

[KBS21] J. M. Kübler, S. Buchholz, and B. Schölkopf. *The Inductive Bias of Quantum Kernels.* (2021). arXiv: `2106.03747 [quant-ph]`.

[Kel18] A. Kelly. *Simulating Quantum Computers Using OpenCL.* (2018). arXiv: `1805.00988 [quant-ph]`.

[KPS19] I. Kerenidis, A. Prakash, and D. Szilágyi. *Quantum Algorithms for Portfolio Optimization.* (2019). arXiv: `1908.08040 [math.OC]`.

[Llo+20] S. Lloyd et al. *Quantum embeddings for machine learning.* (2020). arXiv: `2001.03622 [quant-ph]`.

[LMR13] S. Lloyd, M. Mohseni, and P. Rebentrost. *Quantum algorithms for supervised and unsupervised machine learning.* (2013). arXiv: `1307.0411 [quant-ph]`.

[LW15] X. Li and X. Wu. *Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition.* (2015). arXiv: `1410.4281 [cs.CL]`.

[Mar+20] J. Martyn et al. *Entanglement and Tensor Networks for Supervised Image Classification.* (2020). arXiv: `2007.06082 [quant-ph]`.

[Mey21] J. J. Meyer. *Fisher Information in Noisy Intermediate-Scale Quantum Applications.* (2021). arXiv: 2103.15191 [quant-ph].

[Pes+20] A. Pesah et al. *Absence of Barren Plateaus in Quantum Convolutional Neural Networks.* (2020). arXiv: 2011.02966 [quant-ph].

[Pre21] J. Preskill. *Quantum computing 40 years later.* (2021). arXiv: 2106.10522 [quant-ph].

[RA19] J. Romero and A. Aspuru-Guzik. *Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions.* (2019). arXiv: 1901.00848 [quant-ph].

[RK19] S. Resch and U. R. Karpuzcu. *Quantum Computing: An Overview Across the System Stack.* (2019). arXiv: 1905.07240 [quant-ph].

[RL18] P. Rebentrost and S. Lloyd. *Quantum computational finance: quantum algorithm for portfolio optimization.* (2018). arXiv: 1811.03975 [quant-ph].

[RS20] J. Reyes and M. Stoudenmire. *A Multi-Scale Tensor Network Architecture for Classification and Regression.* (2020). arXiv: 2001.08286 [stat.ML].

[RTP+20] M. S. Rudolph, N. B. Toussaint, A. Perdomo-Ortiz, et al. *Generation of High-Resolution Handwritten Digits with an Ion-Trap Quantum Computer.* (2020). arXiv: 2012.03924 [quant-ph].

[Sch21] M. Schuld. *Supervised quantum machine learning models are kernel methods.* (2021). arXiv: 2101.11020 [quant-ph].

[SCZ16] R. S. Smith, M. J. Curtis, and W. J. Zeng. *A Practical Quantum Instruction Set Architecture.* (2016). arXiv: 1608.03355 [quant-ph].

[Shi+20] Y. Shingu et al. *Boltzmann machine learning with a variational quantum algorithm.* (2020). arXiv: 2007.00876 [quant-ph].

[SS17] E. M. Stoudenmire and D. J. Schwab. *Supervised Learning with Quantum-Inspired Tensor Networks.* (2017). arXiv: 1605.05775 [stat.ML].

[STC21] B. F. Schiffer, J. Tura, and J. I. Cirac. *Adiabatic Spectroscopy and a Variational Quantum Adiabatic Algorithm.* (2021). arXiv: 2103.01226 [quant-ph].

[Suz+20a] Y. Suzuki et al. *Qulacs: a fast and versatile quantum circuit simulator for research purpose.* (2020). arXiv: 2011.13524 [quant-ph].

[Tor+20] G. Torlai et al. *Quantum process tomography with unsupervised learning and tensor networks.* (2020). arXiv: 2006.02424 [quant-ph].

[VPB18] G. Verdon, J. Pye, and M. Broughton. *A Universal Training Algorithm for Quantum Deep Learning.* (2018). arXiv: 1806.09729 [quant-ph].

[Wan+20a] J. Wang et al. *Anomaly Detection with Tensor Networks.* (2020). arXiv: 2006.02516 [cs.LG].

[Wan+21b] S. Wang et al. *Noise-Induced Barren Plateaus in Variational Quantum Algorithms.* (2021). arXiv: 2007.14384 [quant-ph].

[Zei12] M. D. Zeiler. *ADADELTA: An Adaptive Learning Rate Method.* (2012). arXiv: 1212.5701 [cs.LG].

[Zha+20] K. Zhang et al. *Toward Trainability of Quantum Neural Networks.* (2020). arXiv: 2011.06258 [quant-ph].

Software and others

[Ahm19] S. Ahmed. *Data-reuploading classifer.* (2019). URL: https://pennylane.ai/qml/app/tutorial_data_reuploading_classifier.html.

[Ale+19a] G. Aleksandrowicz et al. *Qiskit: An Open-source Framework for Quantum Computing.* (2019). DOI: 10.5281/ZENODO.2562110. URL: http://zenodo.org/record/2562110.

[Ard16] M. A. Ardeh. *BenchmarkFcns Toolbox.* (2016). URL: http://www.benchmarkfcns.xyz/.

[Cir21] Cirq Developers. *Cirq.* (2021). DOI: 10.5281/ZENODO.4062499. URL: https://zenodo.org/record/4062499.

[Eft+20b] S. Efthymiou et al. *Quantum-TII/qibo: Qibo 0.1.1.* Version v0.1.1. (Oct. 2020). DOI: 10.5281/zenodo.4071702. URL: http://doi.org/10.5281/zenodo.4071702.

[Kas19] Z. Kassabov. *Reportengine: A framework for declarative data analysis.* Version v0.27. (Feb. 2019). DOI: 10.5281/zenodo.2571601. URL: http://doi.org/10.5281/zenodo.2571601.

[Mar+15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. (2015). URL: https://www.tensorflow.org/.

[nik+20] niko et al. *CMA-ES/pycma: r3.0.3.* (Apr. 2020). DOI: 10.5281/zenodo.3764210. URL: https://zenodo.org/record/3764210 (visited on 06/03/2021).

[NVF20] NVIDIA, P. Vingelmann, and F. H. Fitzek. *CUDA, release: 10.2.89.* (2020). URL: https://developer.nvidia.com/cuda-toolkit.

[Pek07] Pekaje. *Accuracy and precision.* (2007). URL: https://commons.wikimedia.org/w/index.php?curid=1862863.

[Pér19] A. Pérez-Salinas. *Quantum classifier with data re-uploading.* (2019). URL: https://github.com/AdrianPerezSalinas/universal_qlassifier.

[Pér21] A. Pérez-Salinas. *Universal-Approximator.* (2021). URL: https://github.com/UB-Quantic/Universal-Approximator.

[Qib20] Qibo Team. *Data-reuploading classifer.* (2020). URL: https://qibo.readthedocs.io/en/stable/tutorials/reuploading_classifier/README.html.

[RP20] S. Ramos-Calderer and A. Pérez-Salinas. *Quantum Finance.* (2020). URL: https://github.com/UB-Quantic/quantum-unary-option-pricing.

Acronyms

BCH	Baker-Campbell-Haussdorf
BM	Boltzmann Machines
BP	Barren Plateaus
CML	Classical Machine Learning
CPU	Central Processing Units
FfNN	Feedforward Neural Networks
FPGA	Field Programmable Gate Array
GPU	Graphical Processing Units
HEP	High Energy Physics
IQAE	Iterative Quantum Amplitude Estimation
JIT	Just-In-Time
LHC	Large Hadron Collider
ML	Machine Learning
NG	Natural Gradient
NISQ	Noisy Intermediate-Scale Quantum
NN	Neural Networks
PCA	Principal Component Analysis
PDF	Parton Distribution Functions
QAE	Quantum Amplitude Estimation
QAOA	Quantum Approximate Optimization Algorithm

QFT Quantum Fourier Transform
qGAN quantum Generative Adversarial Networks
QML Quantum Machine Learning
QPE Quantum Phase Estimation
QPU Quantum Processing Unit
QRAM Quantum Random Access Memory
SGD Stochastic Gradient Descent
SVC Support Vector Classifiers
TN Tensor Networks
UAT Universal Approximation Theorem
VQA Variational Quantum Algorithms
VQE Variational Quantum Eigensolver