On the intersection of quantum computing and computational structural mechanics
With a focus on near-term quantum computing

Tosti Balducci, G.B.L.

DOI
10.4233/uuid:7a0fd2a9-d401-4d76-8eed-37ca092a235d

Publication date
2025

Document Version
Final published version

Citation (APA)
Tosti Balducci, G. B. L. (2025). *On the intersection of quantum computing and computational structural mechanics: With a focus on near-term quantum computing*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:7a0fd2a9-d401-4d76-8eed-37ca092a235d

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

On the intersection of quantum computing and computational structural mechanics

With a focus on near-term quantum computing

On the intersection of quantum computing and computational structural mechanics

With a focus on near-term quantum computing

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus,
prof. dr. ir. T.H.J.J. van der Hagen,
Chair of the Board for Doctorates
to be defended publicly on
Friday 3, October 2025 at 12:30 o'clock

by

Giorgio TOSTI BALDUCCI

Master of Science in Aerospace engineering, Delft University of Technology, The Netherlands
born in Grosseto, Italy

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Prof. dr. ir. R. De Breuker,	Delft University of Technology, *promotor*
Dr. rer. nat. M. Möller,	Delft University of Technology, *promotor*
Dr. B. Chen,	Delft University of Technology, *copromotor*
Independent members:	
Prof. dr. ir. L.J. Sluys	Delft University of Technology
Prof. dr. ir. C. Vuik	Delft University of Technology
Dr. R. Steijl	University of Glasgow (UK)
Prof. dr. O. Kyriienko	University of Sheffield (UK)
Dr. ir. M. I. Gerritsma,	Delft University of Technology, reserve member

Keywords: quantum computing, structural mechanics, machine learning
Printed by: Ipskamp Printing
Cover by: Ipskamp Printing

ISBN 978-94-6473-885-8

An electronic copy of this dissertation is available at
https://repository.tudelft.nl/.

When you read, don't just consider what the author thinks, consider what you think.

Tom Schulman, *Dead Poets Society*: The Screenplay

CONTENTS

Summary ix

Samenvatting xi

1. Introduction 1
1.1. Scientific background of the thesis . . . 3
1.1.1. Quantum algorithms for PDEs in structural mechanics 4
1.1.2. Solving the Poisson 1D problem with a variational quantum linear solver . . . 5
1.1.3. Classifying failure states of open-hole composite panels using classical- and quantum-kernel based support vector machines . . . 5
1.2. Research problem . . . 6
1.3. Reseach aim and research questions . . . 7
1.4. Significance of the research . . . 8
1.5. Limitations of the research . . . 9
1.6. Thesis outline . . . 10

2. Review of quantum PDE methods for structures 13
2.1. Main concepts of quantum PDE solving . . . 15
2.1.1. Quantum state preparation . . . 15
2.1.2. Hamiltonian simulation . . . 18
2.1.3. Quantum linear solvers . . . 19
2.1.4. Amplitude amplification and amplitude estimation . . . 20
2.1.5. Variational quantum algorithms . . . 25
2.1.6. Measurement . . . 28
2.2. Fundamental PDEs in structural mechanics . . . 29
2.2.1. Mechanical linear equilibrium . . . 29
2.2.2. Mechanical nonlinear equilibrium . . . 30
2.2.3. Thermal equilibrium . . . 30
2.3. Quantum algorithms for linear PDEs . . . 31
2.3.1. Poisson equation . . . 31
2.3.2. Heat equation . . . 46
2.3.3. Wave equation . . . 54
2.4. Quantum algorithms for nonlinear PDEs . . . 58
2.5. Discussion . . . 60

3. VQLS for Poisson 1D **71**
3.1. Background . . . 73
3.1.1. The VQLS algorithm . . . 73
3.2. 1D Poisson Problem and Matrix Decompositions . . . 76
3.2.1. Pauli decomposition . . . 76
3.2.2. Decomposition with multi-qubit gates . . . 78
3.3. Results . . . 80
3.4. Discussion . . . 83

4. Composite failure predictions with classical and quantum SVM **89**
4.1. Introduction . . . 90
4.2. Machine learning problem . . . 91
4.3. Methodology . . . 94
4.4. Results . . . 98
4.5. Conclusion . . . 103

5. Conclusions **109**
5.1. Conributions to the existing literature . . . 109
5.2. Answers to the research questions . . . 110
5.3. Recommendations for future work . . . 113

A. Kernels - Open hole specimen features and finite element model details **119**
A.1. Geometry and material properties . . . 119
A.2. Details of the FE models . . . 120

B. Kernels - Support vector machines, kernel methods and KTA **123**
B.1. Primal SVM . . . 123
B.2. Dual SVM and kernels . . . 124
B.3. Kernel-target alignment . . . 125

C. Kernels - Comparison on further classification metrics **129**

Acknowledgements **131**

Curriculum Vitæ **133**

List of Publications **135**

SUMMARY

Quantum computation encodes and operates on data in a radically different way than classical logic. This difference allows researchers in nearly all applied sciences to explore how quantum-accelerated computing could enhance the efficiency of their most computationally demanding tasks.

This thesis studies the introduction of quantum-accellerated computing in structural mechanics, a field that traditionally leverages computational techniques at both industrial and research scales. The scope is limited to practical and mostly near-term quantum computing. Therefore, the algorithms analyzed or proposed are evaluated for their runtime as end-to-end routines and for their ability to run on near-term quantum devices.

Given the vastness of the application domain, the research was developed in three separate threads. The first is a review of quantum algorithms applicable to partial differential equations (PDEs) in structural mechanics. The second is an application of a variational quantum algorithm for linear systems of equations to the discrete Poisson equation, while the last studies how quantum machine learning, specifically quantum kernel methods, discriminates damage-inducing loading states in a composite plate with a cutout.

Each of the three parts reveals the potentials and limitations of practical quantum-accelerated computing. The PDE review questions the end-to-end advantage of fault-tolerant quantum algorithms and highlights the need to specialize near-term alternatives to problems in mechanics. The work on the variational quantum linear solver emphasizes the matrix decomposition bottleneck and proposes a method to factorize the discrete Poisson equation matrix. Finally, the work on kernel methods for damage identification shows how heuristically selected and trained quantum kernels reach scores comparable to classical best-practice kernels, but also hints at the fact that potential quantum advantage requires more systematic kernel optimization and retaining performance when scaling quantum systems beyond classical simulation.

SAMENVATTING

Quantumcomputing codeert en werkt met data op een radicaal andere manier dan klassieke logica. Dit verschil stelt onderzoekers in bijna alle toegepaste wetenschappen in staat om te onderzoeken hoe quantum-accelerated computing de efficiëntie van hun meest computationeel veeleisende taken kan verbeteren.

Dit proefschrift bestudeert de introductie van quantum-accelerated computing in structurele mechanica, een vakgebied dat traditioneel computationele technieken op zowel industriële als onderzoeksschaal benut. De reikwijdte is beperkt tot praktische en voornamelijk op korte termijn quantumcomputing. Daarom worden de geanalyseerde of voorgestelde algoritmen geëvalueerd op hun runtime als end-to-end routines en op hun vermogen om op korte termijn quantumapparaten te draaien.

Gezien de omvang van het toepassingsgebied werd het onderzoek ontwikkeld in drie afzonderlijke threads. De eerste is een review van quantumalgoritmen die van toepassing zijn op partiële differentiaalvergelijkingen (PDE's) in structurele mechanica. De tweede is een toepassing van een variationeel kwantumalgoritme voor lineaire vergelijkingssystemen op de discrete Poisson-vergelijking, terwijl de laatste onderzoekt hoe kwantummachinelearning, met name kwantumkernelmethoden, schadeinducerende laadtoestanden in een samengestelde plaat met een uitsparing onderscheidt.

Elk van de drie onderwerpen onthult het potentieel en de beperkingen van praktisch kwantumversneld computergebruik. De PDE-review stelt het end-to-endvoordeel van fouttolerante kwantumalgoritmen ter discussie en benadrukt de noodzaak om alternatieven op korte termijn voor problemen in de mechanica te specialiseren. Het werk aan de Variational Quantum Linear Solver benadrukt de knelpunten van matrixontleding en stelt een methode voor om de discrete Poisson-vergelijkingsmatrix te factoriseren. Ten slotte laat het werk aan kernels voor schade-identificatie zien hoe heuristisch geselecteerde en getrainde kwantumkernels scores bereiken die vergelijkbaar zijn met klassieke best-practice kernels, maar hint ook naar het feit dat potentieel kwantumvoordeel meer systematische kerneloptimalisatie en behoud van prestaties vereist bij het schalen van kwantumsystemen voorbij klassieke simulatie.

1

INTRODUCTION

1

Quantum computing promises to disrupt many scientific fields by solving problems in useful times that would be intractable with today's largest supercomputers. From the initial vision of simulating molecules with controllable quantum systems [1], the applications of quantum computing proliferated over the last three decades and included data encryption, optimization, machine learning, and other domains. In general, the idea of *quantum-accellerated computing* came to be, for which computational sub-problems that are classically challenging can be outsourced to a quantum computer that can solve them efficiently.

Structural engineers have long been using accelerators to alleviate the bottlenecks of their numerical pipelines. The most obvious example is perhaps parallel computing, which splits the workload of nonsequential tasks between multiple processors and speeds up tasks with relatively low data transfer overhead. For example, many finite element codes incorporate parallel linear solvers into their linear algebra libraries [2] and with the recent introduction of neural networks for material and structural modeling, graphics processing units (GPUs) also become relevant to computational mechanics.

Therefore, it is not surprising that researchers in the field started looking at quantum computers as a new and potentially better-performing accellerator. The strongest appeal lies in those algorithms that can exponentially improve the runtime of their classical counterparts. For example, by solving large linear systems of an wing optimization problem using the HHL algorithm [3], which is exponentially faster than classical linear solvers, one might wish to speed up the overall optimization process exponentially.

Regrettably, yet intriguingly, the situation is not as straightforward as simply integrating a textbook quantum solver with existing industrial software. In reality, there are caveats all the way down the computational line. Even if a quantum algorithm offers a theoretical speed-up on best practice classical solvers, is it also possible to prepare quantum states and read out results also as efficiently? And if using hybrid algorithms, more suitable for the technology readiness level (TRL) of current near-term intermediate scale quantum (NISQ) devices [4], is the advantage retained and to what extent? Finally, does hardware noise completely pollute the computation when running the compiled program on an actual quantum processing unit (QPU)?

These are fundamental questions that are under the spotlight of the applied quantum computing community. The research reported here did not aim to answer them in general or in any specific field. What this research did was to critically explore some of the possible applications of quantum computing to computational structural mechanics, while keeping the above practical matters in mind.

The rest of this Introduction offers more details on what this research

involves and looks into its objectives, significance, and limitations. Section 1.1 gives concise background on the main concepts of the remaining chapters. Section 1.2 summarizes the research problem and Section 1.3 specifies the objectives and research questions. Section 1.4 explains the significance of this PhD work and Section 1.5 its limitations. Finally, Section 1.6 outlines the structure of the rest of the dissertation.

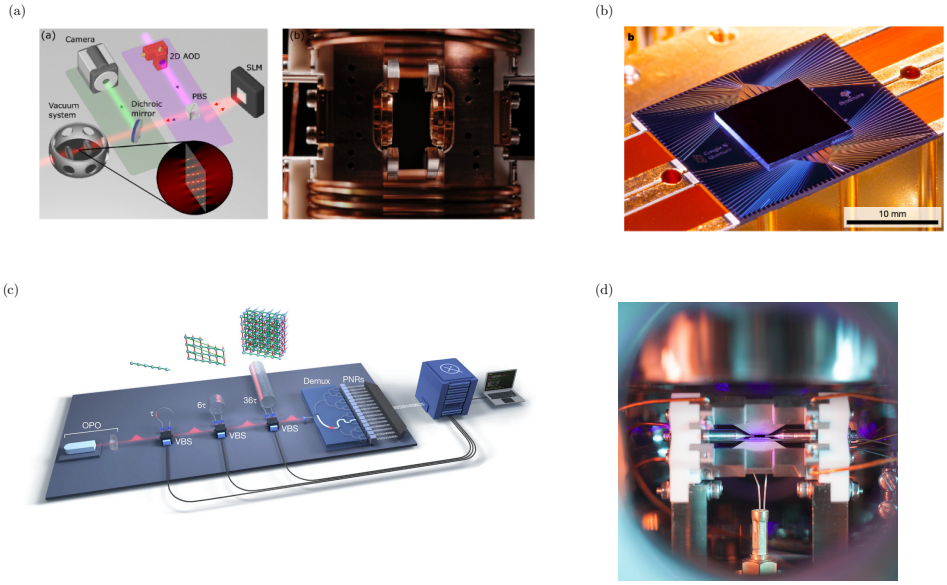

Figure 1.1.: The technology-readiness level (TRL) of current quantum computers is currently comparable to that of early starge classical computers. As in the 1950s the units of computations could be mechanical relays, vacuum tubes and transistors, nowadays quantum computation happens with (a) neutral atoms [5], (b) superconducting qubits [6], (c) photons [7], (d) trapped ions [8] and more. There is not a clear winner yet and whether only one or multiple technologies will survive the race will be decided by matters like scalability, noise level, and error correction options.

1.1. SCIENTIFIC BACKGROUND OF THE THESIS

The main body of this manuscript describes the three main research areas of this work. They are a review of quantum algorithms for partial differential equations (PDEs) in structural mechanics, an algorithm to solve the discretized Poisson equation using the variational quantum

linear solver (VQLS) and an ultimate failure classification algorithm based on classical and quantum kernels.

1.1.1. QUANTUM ALGORITHMS FOR PDES IN STRUCTURAL MECHANICS

The mechanical deformation of solids and structures can be described by means of partial differential equations. Some notorious linear models are presented in every introductory course in structural mechanics. For example, the linear elasticity equation describes the behaviour of generic three dimensional solids undergoing small deformations. Also, structural elements can be represented by differential equations, such as the Euler-Bernoulli and Timoschenko models for thin and thick beans and the Kirchhoff model and Reissner-Mindlin equations for plates [9]. Nonlinear differential equations also describe many fundamental phenomena in material and structural mechanics. Examples of this are large deformation, hyperplasticity, plasticity, damage and contact mechanics [10].

Because exact solutions only exist for a few textbook problems, engineers mostly resort to techniques that deliver reliable approximate solutions. Numerical methods such as finite elements, finite differences, and finite volumes, originated decades ago. They have known several algorithmic improvements and cleverly developed around new technologies such as parallel processing. However, the need has always existed to cut the computational costs of the solvers, in line with the increasing modeling demands.

Many quantum algorithms seem to offer a breakthrough, since their complexity can be exponentially lower than even current state-of-the-art classical techniques. For example, inverting the stiffness matrix of a finite element model with millions of degrees of freedom would require either long execution times or high computing power for classical machines, but would be almost instantaneous for a quantum computer [3]. However, matrix inversion is only one part of the computation and one should not think of solving a PDE end-to-end with a quantum algorithm, but rather of having a *quantum primitive* in an otherwise classical process. The total cost must then be computed from the point of preparing the quantum states to that of measuring the quantities of interest.

Even without considering the theoretical barriers, quantum computers are not currently fault-tolerant machines and cannot yet implement complexity-superior algorithms. However, as will be discussed in Chapter 2, there are ways to solve PDEs with NISQ devices via hybrid quantum-classical routines, which may still offer an advantage for specific instances. Interesting strategies leverage circuit learning [11] and quantum annealing [12], which is a form of analog quantum

computing and provides the ability to solve different linear and nonlinear equations. However, yet to be fully developed is the adaptation of these near-term routines to specific application domains.

1.1.2. SOLVING THE POISSON 1D PROBLEM WITH A VARIATIONAL QUANTUM LINEAR SOLVER

Numerical approximations of PDEs often result in one or multiple linear systems of equations. In many interesting cases, these linear systems are sparse, meaning that only a few degrees of freedom are coupled, allowing for efficient classical and quantum methods. A prototypical example is the Poisson equation, which is discretized to a matrix with only one band of nonzero elements and to a tri-diagonal linear system in one dimension.

Because quantum hardware immaturity does not allow one to successfully run the logarithmic-time quantum linear solvers, a hybrid quantum-classical solution was developed to better match the technological readiness of NISQ machines. The Variational Quantum Linear Solver (VQLS) [13] maps the problem of solving the linear system into that of finding the ground state of a Hamiltonian, which corresponds, up to the solution vector's norm, to the original linear system's solution.

VQLS was demonstrated on fabricated examples, where the linear system's matrix is built from a polynomial number of unitary operators. Our intention was instead to deal with a mechanical problem (the 1D Poisson problem) that was not built a priori for VQLS and deal with its decomposition in unitary terms.

The main challenge with arbitrary matrices is to decompose them into just a few operators. A straightforward decomposition uses what is known as *Pauli operators*. However, it will be shown that the Poisson equation decomposes in a number of Pauli operators that scales exponentially in the number of qubits. Therefore, a major challenge of applying VQLS to practical problems is to reduce as much as possible the number of terms in the decomposition of the linear system's matrix.

1.1.3. CLASSIFYING FAILURE STATES OF OPEN-HOLE COMPOSITE PANELS USING CLASSICAL- AND QUANTUM-KERNEL BASED SUPPORT VECTOR MACHINES

Machine learning is believed to be one of the disciplines that can demonstrate quantum advantage in the NISQ era [14]. In fact, systems with relatively few qubits can be a highly expressive map of classical data [15] and achieve superior learning in tasks such as regression or classification.

In addition to mapping data, quantum computers can also natively calculate similarities or *inner products* between these mapped states

[16]. The combination of these two features transforms even small quantum circuits into effective *kernels* [17], and enables a form of quantum machine learning (QML) that is potentially advantageous and amenable to quantum hardware.

The last part of this research identifies a nontrivial binary classification problem and compared classical and quantum kernel-based machine learning models to solve it. More specifically, the task is to classify intact and failed open hole composite specimens subject to two-dimensional planar loading. Traditional approaches to solving this problem use either semi-empirical formulas or finite-element models. The first ones are generally used to predict quantities of interest, such as the stresses for first ply failure. They are analytical predictors, but they make strong assumptions on the failure modes involved and may need expensive experimental campaigns to determine certain parameters. On the other hand, finite-element models can capture all sort of damage modes and the interaction between them, but at the cost of large computational times.

In our approach, we rely on a dataset extracted from finite element analyses, but only on an offline training phase. The labelled loading states obtained are input into a Support Vector Machine (SVM) [18] algorithm and kernel functions to determine the separating boundary between different classes. The resulting surrogate classifier is then a fast evaluator of loading states on an open-hole composite plate.

1.2. RESEARCH PROBLEM

Throughout the duration of this thesis, quantum computing received increasing attention from different applied sciences communities. For instance in fluid mechanics, both fault-tolerant and near-term quantum algorithms were applied to prototypical examples in the field [19]. However, the literature at the intersection of quantum computation and structural mechanics is more sporadic. Thus, this thesis aims to be one of the first exploratory works of quantum computational mechanics.

Because differential equations are ubiquitous in mechanics, they are a natural starting point for testing quantum computing methods. The literature on PDE solving is consolidated in both quantum algorithms and computational solid mechanics, but there is almost a void in between. Therefore, there is a need to review solvers on one side and problems on the other and critically assess the possibility, advantage, and limitations of quantum-accellerated PDE solvers for solid mechanics applications.

A common ground to both mechanics and quantum computing research is linear algebra and, in particular, solving linear systems of equations. While there is no shortage of fault-tolerant quantum linear solvers, near-term options are mostly limited to one variational approach. The variational quantum linear solver cleverly reformulates

the linear system problem into finding the ground state of a related Hamiltonian. However, the decomposition of this Hamiltonian is non-trivial. Even when dealing with the didascalic case of the discretized 1D Poisson equation, the scaling of the decomposition can become a bottleneck. This makes it critical to find efficient decompositions to PDE-derived linear systems, else any hope of computational advantage of the variational linear solver is compromised.

Beyond differential equations and linear systems, quantum machine learning (QML) is another gateway for short-term applications in structural mechanics. There is already a consistent literature on material modeling with data-based function approximators, but this is mainly focused on neural networks [20]. On the other hand, quantum computers are native kernel approximators [17], suggesting that QML is easily integrated with kernel-based methods such as support vector machines.

But how do we apply quantum machine learning to computational solid mechanics? First, we evaluate the interest of a problem or see if it can be expressed in a form that is amenable for kernel methods. Then there is the issue of limited resources. High-dimensional inputs, such as image data without compression cannot fit the quantum registers available today, so the input must be either low-dimensional or compressed in some way. Finally, there is no consensus on a default kernel function that has good performance over a range of problems, contrary to what happens with the Radial Basis Functions (RBF) kernel in classical machine learning. Thus, many different embeddings must be explored get the required expressivity compared to classical kernels with respect to different classification metrics.

1.3. RESEACH AIM AND RESEARCH QUESTIONS

The overarching aim of this research is to identify and show potential applications of quantum computing to structural mechanics and critically evaluate them for advantage and near-term feasibility. The novelty of the subject suggests against a more restricted aim and in favor of exploratory rather than exploitative research.

That being said, based on the discussion in Section 1.1, we can introduce the three main research questions of this thesis.

> *Which fault-tolerant and near-term quantum algorithms can solve (parts of) which PDEs in structural mechanics? What can be said about their speedup with respect to classical solvers?*
>
> *Are decompositions of the one-dimensional Poisson matrix that scales polynomially in the number of qubits?*
>
> *Up to what accuracy can quantum-kernel based support vector machines classify failed open-hole composite specimens,*

when trained on numerical simulations data? How does that performance compare to state-of-the-art classical kernels?

1.4. SIGNIFICANCE OF THE RESEARCH

This thesis helps practicioners and future researchers of applied mechanics to determine the interest and applicability of quantum algorithms in their domains.

On the one hand, it says something about the expectations that mechanical engineering should have about both fully mature and near-term quantum technology. By reading the review in the next chapter, those approaching quantum computing from the computational mechanics side get an idea of whether fault-tolerant quantum advantage is in fact end-to-end for their problem or if it only speeds up the quantum primitive. In the same way, the review helps formulating PDE-related problems that are amenable for quantum computation, for instance by favoring sparse input arrays rather than dense ones and by suggesting against use cases that need to read out the full solution vector.

The bulk of the thesis concerns however near-term quantum computation, which is more tangible, but does not have guarantees about runtimes and scaling with the problem's dimension. The review helps identifying the different hybrid solutions to PDE solving and offers insights on their generality and possible pittfalls. One of the approaches, the 1D Poisson equation solver via the VQLS algorithm has a dedicated chapter, since it is illustrative of how a variational quantum algorithm finds a lowerbound in its runtime when applied to a simple classical problem. The same work suggests a solution and talks about an even better approach proposed in literature [21]. The utility of these workarounds is not just related to the specific case of the 1D Poisson equation, but to how to decompose non-trivial Hamiltonian terms efficiently for implementation on quantum software programs and potentially hardware.

This thesis also touches upon applied quantum machine learning. Thanks to the fact that Hamiltonian encoding-based quantum embeddings are universal function approximators [17], QML allows to experiment with hybrid quantum computing not just for toy problems, but for interesting applications. Specifically, we solve the problem of simulating the failure envelope of open-hole composite laminates given the strain loads applied. The novelty of this approach is not only relative to quantum computing; rather, it is probably the inaugural instance of a surrogate (i.e., data-driven) failure criterion. Perhaps it will inspire future work, that can even use data of mixed sources, experimental and numerical to define failure envelopes of substructures or new materials. From the perspective of applied quantum algorithms, it shows how to integrate quantum kernels in a learning pipeline for material mechanics

classification problems. There are not many other cases in the literature that bring QML so close to real-world applications[1]. It can also be added that failure envelope characterization may be a good candidate for near-term implementation in quantum hardware. In fact, the input feature vector is low-dimensional (at most nine components of the strain tensor), which means that no feature reduction is needed and that more quantum resources can be devoted to increase the feature map expressivity [24].

1.5. LIMITATIONS OF THE RESEARCH

This research did not aim to make any general statement about whether quantum computers are or ever will be advantageous to structural mechanics. Nonetheless, this theme has consistently served as the foundation of the work.

Critically reviewing PDE algorithms allowed us to translate *known* limitations of near-term and fault-tolerant methods into a perspective on structural mechanics. No definitive conclusions can be drawn about the effectiveness of quantum PDE solvers universally, since research in this area is still in its very early stages. Furthermore, there might have been advances in the field since the review was conducted, which are therefore not included in the analysis.

In solving the 1D Poisson problem with the VQLS method, we found a new technique to decompose the Hamiltonian in less than half the terms required by the Pauli basis. Although this new decomposition holds for any number of points in the spatial grid, it does not extend to the generic Poisson equation of N. A similar work conducted shortly after our own discovered a more general decomposition with only logarithmic scaling [21]. Furthermore, we provided numerical examples of solving the 1D Poisson equation with VQLS and our new decomposition, but we did not study the training part of the algorithm. Experimental evidence of trainability, scaling with register size and depth of the *ansatz* circuit is left for future work.

About the composite failure identification project, the numerical findings do not claim any generality. They hold for the specific specimen, input type, finite element analysis settings, sampling and labeling criterion used. Some of the methods however are novel and can be applied more generally. In particular, the idea of treating failure identification as a binary classification problem can be used in other data-based failure envelope characterization cases. Furthermore, the radial sampling strategy we propose to sample the input space with incremental-iterative analyses can also be applied to other tasks where data is drawn from finite element simulations.

[1] Even though exceptions exist such as [22] and [23]

1.6. THESIS OUTLINE

The remaining of the manuscript presents the three main works of this thesis and the concluding remarks. Chapter 2 presents the review of quantum algorithms for PDEs in structural mechanics. Chapter 3 discusses how to solve the 1D Poisson problem via the variational quantum linear solvers, while Chapter 4 deals with open hole composite failure classification with classical and quantum kernel-based support vector machines. Chapter 5 offers the final conclusions and recommendations for future work.

REFERENCES

[1] R. P. Feynman. “Simulating physics with computers”. In: *Int. J. Theor. Phys.* 21.6 (June 1982), pp. 467–488.

[2] V. Kumar, K. Chandan, K. V. Nagaraja, and M. V. Reddy. “Heat Conduction with Krylov Subspace Method Using FEniCSx”. In: *Energies* 15.21 (Oct. 2022), p. 8077.

[3] A. W. Harrow, A. Hassidim, and S. Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Phys. Rev. Lett.* 103.15 (Oct. 2009), p. 150502.

[4] J. Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (Aug. 2018), p. 79.

[5] L. Henriet *et al.* “Quantum computing with neutral atoms”. In: *Quantum* 4 (Sept. 2020), p. 327.

[6] F. Arute *et al.* “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574 (Oct. 2019), pp. 505–510.

[7] L. S. Madsen *et al.* “Quantum computational advantage with a programmable photonic processor”. In: *Nature* 606 (June 2022), pp. 75–81.

[8] *Ion trap quantum computing*. [Online; accessed 26. Jun. 2024]. June 2024.

[9] A. P. Boresi and R. J. Schmidt. *Advanced Mechanics of Materials, 6th Edition*. Hoboken, NJ, USA: Wiley, Nov. 2002. isbn: 978-0-471-43881-6.

[10] G. A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Hoboken, NJ, USA: Wiley, Apr. 2000. isbn: 978-0-471-82319-3.

[11] O. Kyriienko, A. E. Paine, and V. E. Elfving. “Solving nonlinear differential equations with differentiable quantum circuits”. In: *Physical Review A* 103.5 (May 2021).

[12] J. C. Criado and M. Spannowsky. “Qade: solving differential equations on quantum annealers”. In: *Quantum Science and Technology* 8.1 (2022), p. 015021.

[13] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles. “Variational Quantum Linear Solver”. In: *Quantum* 7 (Nov. 2023), p. 1188.

[14] Y. Gujju, A. Matsuo, and R. Raymond. “Quantum machine learning on near-term quantum devices: Current state of supervised and unsupervised techniques for real-world applications”. In: *Phys. Rev. Appl.* 21.6 (June 2024), p. 067001.

[15] M. Schuld and F. Petruccione. *Machine Learning with Quantum Computers*. Springer International Publishing, 2021. isbn: 9783030830984.

[16] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. “Quantum Fingerprinting”. In: *Phys. Rev. Lett.* 87.16 (Sept. 2001), p. 167902.

[17] M. Schuld. “Supervised quantum machine learning models are kernel methods”. In: *arXiv:2101.11020* (Jan. 2021).

[18] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, Mar. 2000. isbn: 9780511801389.

[19] S. Succi, W. Itani, K. Sreenivasan, and R. Steijl. “Quantum computing for fluids: Where do we stand?” In: *Europhys. Lett.* 144.1 (Oct. 2023), p. 10001.

[20] X. Liu, S. Tian, F. Tao, and W. Yu. “A review of artificial neural networks in the constitutive modeling of composite materials”. In: *Composites Part B* 224 (Nov. 2021), p. 109152.

[21] H.-L. Liu *et al.* “Variational quantum algorithm for the Poisson equation”. In: *Phys. Rev. A* 104.2 (Aug. 2021), p. 022418.

[22] A. Miroszewski *et al.* “Detecting Clouds in Multispectral Satellite Images Using Quantum-Kernel Support Vector Machines”. In: *arXiv:2302.08270* (Feb. 2023).

[23] Z. Kaseb, M. Moller, G. T. Balducci, P. Palensky, and P. P. Vergara. “Quantum Neural Networks for Power Flow Analysis”. In: *arXiv:2311.06293* (Nov. 2023).

[24] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre. “Data re-uploading for a universal quantum classifier”. In: *Quantum* 4 (Feb. 2020), p. 226.

2

REVIEW OF QUANTUM ALGORITHMS FOR PARTIAL DIFFERENTIAL EQUATIONS IN STRUCTURAL MECHANICS

Structural mechanics is commonly modeled by (systems of) partial differential equations (PDEs). Except for very simple cases where analytical solutions exist, the use of numerical methods is required to find approximate solutions. However, for many problems of practical interest, the computational cost of classical numerical solvers running on classical, that is, silicon-based computer hardware, becomes prohibitive.

Quantum computing, though still in its infancy, holds the promise of enabling a new generation of algorithms that can execute the most cost-demanding parts of PDE solvers up to exponentially faster than classical methods, at least theoretically. Also, increasing research and availability of quantum computing hardware spurs the hope of scientists and engineers to start using quantum computers for solving PDE problems much faster than classically possible.

This work reviews the contributions that deal with the application of quantum algorithms to solve PDEs in structural mechanics. The aim is not only to discuss the theoretical possibility and extent of advantage for a given PDE, boundary conditions and input/output to the solver, but also to examine the hardware requirements of the methods proposed in literature.

Parts of this chapter have been published in Frontiers in Mechanical Engineering **8**, 914241 (2022) [1].

2

Structural mechanics is often modeled by means of partial differential equations (PDEs). However, it is only for a small set of simple problems, domains and boundary condition that an analytical solution is known. In all other cases, engineers must rely on numerical techniques to obtain an approximate solution.

Given the importance of PDEs, research on convergence and accuracy of numerical solvers has dominated in the past decades. Nevertheless, the computational demand is still high whenever the domain is large with respect to the physics' scale or when nonlinearities require a high level of detail of the numerical solution.

Potentially, quantum computing can revolutionize the field of numerical computational mechanics, thanks to its promise of unprecedented theoretical speed-up. For instance, the Harrow, Hassidim, Lloyd (HHL) algorithm [2] can solve sparse linear systems exponentially faster than any classical method. At a first glance, it may seem that HHL could be applied to the discretized Poisson equation and exponentially reduce the runtime to obtain the displacement field of a loaded structure.

As tantalizing as they sound, almost all promises of quantum advantage are currently bound to theory and simulations and likely will be for the next few decades. What has yet to catch up with the algorithms is the hardware, which is still far from the technological requirements for practical quantum advantage.

This limitation prompted a separate branch of research, which abandoned the idea of proving quantum speed-up, but focused on algorithms that take into account the limitations of current hardware. These are mostly hybrid methods, that use a classical computer in combination with a quantum one, assigned to solve classically hard tasks. As with the theoretical speed-up algorithms, these hybrid methods also found application in the field of numerical PDE solving.

Given the importance and richness of the field, this review wants to be the first work to present and compare different quantum PDE solvers. Specifically, this work is aimed at the applied mechanics community and thus limits its focus to equations in structural mechanics. However, it must be pointed out that the field of quantum algorithms for PDEs spans many other branches of science, such as fluid mechanics [3–9], finance [10], model discovery [11] and cosmology [12], which may also benefit from specialized reviews.

This Chapter is structured as follows. Section 2.1 presents some standard concepts of quantum computing and explains the quantum algorithms at the root of PDE-solving. Section 2.3 reviews the literature on quantum algorithms for linear PDEs, while Section 2.4 is devoted to nonlinear PDEs. Finally, Section 2.5 provides concluding remarks and discusses the possible future prospects of quantum computation applied to structural problems.

2.1. MAIN CONCEPTS OF QUANTUM PDE SOLVING

At the time of writing, almost no quantum algorithm applies to all possible combinations of partial differential equations, boundary conditions, discretizations, etc. However, one can trace out a generic PDE-solving workflow, as in Figure 2.1. Here, it is important to notice that quantum computation takes place only in the so-called *quantum primitive*, i.e. an algorithm that acts on quantum states by means of quantum operators. Also, the quantum primitive does not solve the PDE alone, but rather its discrete form, which generally is the computational bottleneck in classical logic.

Furthermore, in contrast to a classical algorithm, a quantum primitive does not operate in the same environment in which the data is stored and read-out. In other words, there is a state-preparation step prior to quantum computation, where classical data is encoded as a quantum state, as well as a measurement step afterwards, that provides output in classical form.

A similar workflow was previously proposed [13], where the author points out that a discretized PDE can be mapped either to a Schrödinger equation or to one or more linear system(s). In the first case, one may use Hamiltonian simulation to obtain the solution as a quantum state, while linear systems could be solved by the HHL algorithm or other quantum linear solvers. However, this classification concerns only linear PDEs and does not consider algorithms that are not fully quantum (i.e. all computations from quantum-form data to quantum-form solution are done on a quantum computer) or quantum annealing.

This section gives the necessary background on the 'quantum-block' of PDE solving, that concerns the last three steps in Figure 2.1.

2.1.1. QUANTUM STATE PREPARATION

The problem of quantum state preparation arises every time classical data need to be encoded as a quantum state, and it is by no means restricted to quantum PDE solvers. Consider the task of encoding an initial condition $u_0(x)$ in a quantum register. First, space-discretization transforms $u_0(x)$ into a discrete array of gridpoint values $\mathbf{u}_0$ of N entries. Then, this vector is normalized to turn it into the suitable quantum state

$$|u_0\rangle = \frac{\mathbf{u}_0}{\|\mathbf{u}_0\|_2}. \tag{2.1}$$

The vector on the left-hand side of Equation (2.1) is called *ket vector*, according to the so-called *braket* notation [14].

Furthermore, $|u_0\rangle$ can be written as a vector of N complex amplitudes

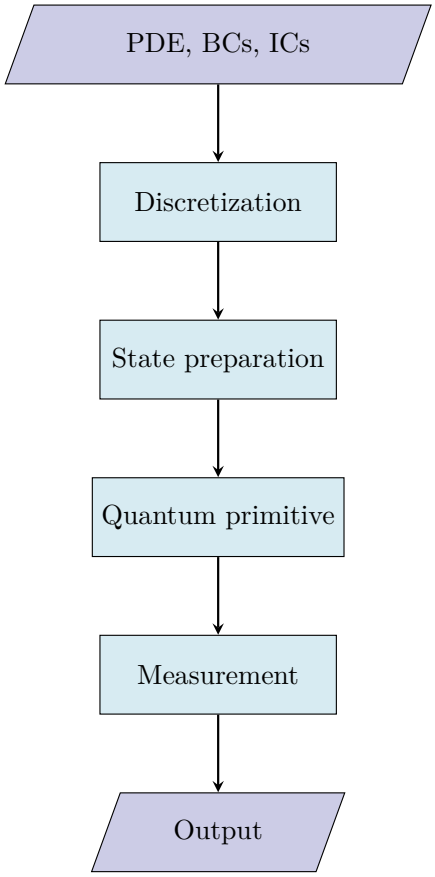

Figure 2.1.: Workflow of the process of solving partial differential equations with a quantum primitive. The input is given, in the general case, by the PDE together with its boundary (BCs) and initial (ICs) conditions. After the differential problem has been discretized with schemes such as finite elements or finite differences, the input data is encoded into a quantum state (state preparation step). The quantum primitive then produces the solution to the discrete problem as a quantum state. However, the information in this state cannot be accessed, as this is generally in quantum superposition. The quantum PDE solver also needs to measure the state after the quantum primitives as many times as required by the user-defined solution precision. Depending of the application, such measurement are postprocessed to provide the final output.

defined with respect to a chosen basis, i.e.

$$|u_0\rangle = \sum_{j=0}^{N-1} u_{0,j} \left|b_j\right\rangle,$$

where $u_{0,j} \in \mathbb{C}$ are the amplitudes and $\left|b_j\right\rangle$ are the basis states. Since many quantum computer models work with binary physical systems (qubits), the basis $\left\{\left|b_j\right\rangle\right\}_{j=0}^{N-1}$ is defined as the set of all possible states of $n = \lceil \log_2(N) \rceil$ such systems ($\log_2$ will be subsituted by log in the following, for simplicity of notation). By labeling the states of a qubit as 0 and 1, the basis states are written as

$$|00\ldots00\rangle, |00\ldots01\rangle, |00\ldots10\rangle, \ldots, |11\ldots11\rangle,$$

which constitute the so-called *computational basis*. This is an orthonormal basis, since, given two basis vectors b_i and b_j

$$\left\langle b_i \middle| b_j \right\rangle = \delta_{ij}, \qquad i,j \in \{0,1,\ldots N-1\},$$

where δ_{ij} is the Dirac delta.

One should keep in mind that the computational basis is just one of the infinitely many possible bases in the $\mathbb{C}^N$ space and that any other binary orthonormal set of vectors form an equally valid basis for the same space.

Furthermore, the braket notation implies a unit norm vector, that is

$$\sum_{j=0}^{N-1} u_{0,j}^2 = 1. \tag{2.2}$$

Equivalently, Equation (2.2) states that the squared amplitudes of a ket vector can be seen as probability amplitudes, modelling the fact that a n-qubits system will collapse to one of the N basis states $|b_m\rangle$ with probability $u_{0,m}^2$ upon measurement.

Having access to the amplitues, the problem of state preparation becomes to evolve an initial state up to the state $|u_0\rangle$. This can be expressed in formulas,

$$|u_0\rangle = U|0\rangle^{\otimes n},$$

where U is a unitary operator representing the quantum state evolution and $|0\rangle^{\otimes n}$ is a conventional initial state where all qubits are in state $|0\rangle$. From now on, the initial state $|0\rangle^{\otimes n}$ will mostly be represented simply as $|0\rangle$, except when the number of qubits is not immediately evident from the context.

Clearly, the cost of implementing U influences the overall cost of solving the differential problem. In fact, preparing a generic quantum

state over n qubits requires $O(N)$ quantum gates [14], where the 'big O' notation represents the asymptotic upper bound on the amount of computational resources. Therefore, even if the quantum primitive runs in $O(\log(N))$ time, the task of numerically evolving the initial condition on a quantum comptuer might still take $O(N)$ time overall. Fortunately, several probability distributions can be prepared efficiently (i.e. in logarithmic time). Interesting cases for numerical analysis are polynomials defined over a regular grid or locally supported functions in the case of the finite element method (FEM) [15].

2.1.2. HAMILTONIAN SIMULATION

One of the initial motivations for quantum computers was to simulate quantum systems that are difficult to simulate classically [16]. Generally speaking, one aims at obtaining the evolution of a quantum state $|\psi\rangle$, which is described by the Schrödinger equation

$$\frac{\mathrm{d}}{\mathrm{d}t}|\psi\rangle = -iH|\psi\rangle, \tag{2.3}$$

where H is the system's Hamiltionian, i.e. a Hermitian matrix. For the sake of explanation, the Hamiltonian is considered here as time-invariant.

Hamiltonian simulation consists of evolving an initial quantum state $|\psi_0\rangle$ according to Equation (2.3) on a quantum computer. Interestingly, certain partial differential equations can be rewritten as Schrödinger equations after a semi-discretization in space [13, 17, 18]. For this reason, quantum algorithms for Hamiltonian simulation can be used as quantum primitives to solve PDEs.

Equation (2.3) has the exact solution

$$|\psi\rangle = e^{-iHt}|\psi_0\rangle, \tag{2.4}$$

where e^{-iHt} is a unitary operator, due to the fact that H is Hermitian. Therefore, e^{-iHt} is a valid quantum operation that could be applied to $|\psi_0\rangle$. However, the exponential of an arbitrary Hamiltonian does not correspond, in general, to any known quantum circuit. In other words, the Hamiltonian H is difficult to simulate directly.

However, one can see H as a sum of Hermitian matrices that are efficient to simulate. In that case

$$H = \sum_i c_i H_i, \tag{2.5}$$

and $|\psi\rangle = e^{-i\sum_i c_i H_i t}|\psi_0\rangle$. The number of terms in Equation (2.5) determines the complexity of the Hamiltonian simulation. In fact, the generic Hamiltonian decomposes in a number of known evolution

operators that scales exponentially with the dimensions of the physical systems. Equation (2.5) cannot always be rewritten as a product of exponentials, as the H_i generally do not commute, i.e.

$$\left[H_i, H_j\right] = H_i H_j - H_j H_i \neq 0,$$

for some i and j. Anyway, by dividing the evolution time in r subsequent time intervals $\Delta t = t/r$, the single H_i Hamiltonians can be evolved separately over Δt. This evolution over the sub-interval is then repeated r times. In the end, the overall operator is an approximation of the actual evolution of the total Hamiltonian H over t as stated by the Trotter's formula [19]:

$$e^{-iHt} = \left(\prod_i e^{-ic_i H_i t/r}\right)^r + O\left((\Delta t)^2\right), \tag{2.6}$$

The importance of H and H_i being easy to simulate is clear from the Hamiltonian simulation runtime using product formulas. This is $O(f(n)t/\varepsilon)$ [14], where n is the number of qubits and ε is the desired Hamiltonian simulation error. While $f(n) = \text{poly}(n)$ for simulatable Hamiltonians, $f(n) = \exp(n)$ in the general case, making for an exponential difference in the runtime.

In quantum physics, many natural systems are described by the so-called *local Hamiltonians*, which are operators that act non-trivially on a few qubits and are known to be efficient to simulate, [14]. However and most importantly for PDEs, it was shown that also sparse Hamiltonians can be simulated in $O(\text{poly}(n))$ time, [20].

Finally, we remark that recent work exponentially reduced the dependency on precision [21, 22] and also achieved an optimal dependency on the sparsity [22]. Most recently, an approach based on qubitization achieved an additive lower bound with respect to t and $1/\varepsilon$ [23].

2.1.3. QUANTUM LINEAR SOLVERS

Many PDE discretization techniques result in one or more linear systems of equations and the runtime required to solve them drives the runtime of the whole PDE algorithm.

After appropriate discretization, the PDE assumes the familiar form $A\mathbf{x} = \mathbf{b}$, which is a linear system in the N-dimensional space, where N is the number of unknowns, which can correspond to the number of gridpoints, particles, or harmonic basis functions, depending on the method of discretization used.

If A is positive definite, the fastest general-purpose classical linear system solver is the conjugate gradient method [24]. This algorithm has asymptotic time complexity

$$O\left(Ns\sqrt{\kappa}\log(1/\varepsilon)\right), \tag{2.7}$$

where s is the matrix sparsity, κ is the conditioning number and ε is the desired error for a certain precision metric.

What motivates the study of linear solvers beyond classical computation is the conjecture, due to complexity arguments, that classical algorithms cannot invert A in time $O(\log(N))$, [2]. Harrow, Hassidim and Lloyd were the first ones to prove that, instead, a quantum algorithm based on the $O(\log(N))$ sparse Hamiltonian simulation and phase estimation could solve linear systems exponentially faster in N, [2]. In particular the time complexity of the HHL algorithm is

$$O\left(s^2\kappa^2 \log(N)/\varepsilon\right). \tag{2.8}$$

However, the HHL algorithm and subsequent quantum linear solvers, do not solve $A\mathbf{x} = \mathbf{b}$, but rather

$$A|x\rangle = |b\rangle, \tag{2.9}$$

known as *quantum linear system problem* (QLSP). Furthermore, HHL and other quantum linear system algorithms (QLSAs) prepare the solution $|x\rangle = A^{-1}|b\rangle$ assuming that $|b\rangle$ has already been prepared and that $|x\rangle$ or an observable of it can be measured efficiently. However, the cost of these operations can (and generally does) trump that of solving for $|x\rangle$.

The complexity of solving the QLSP was improved by later contributions. In terms of conditioning number, Ambainis *et al.* reduced the dependency from $O(\kappa^2)$ to $O(\kappa)$ using variable time amplitude amplification (VTAA) for post-selecting the solution, [25]. Also concerning the conditioning number, Clader *et al.* showed how to include a preconditioner in quantum linear solvers to deal with ill-conditioned matrices, [26]. Later, an approach based on linear combination of unitaries was proposed to replace the phase estimation step, reducing the dependence on precision to $O(\text{poly}\log(1/\varepsilon))$, [27]. Furthermore, an approach inspired by the adiabatic principle was also developed, achieving similar complexity compared to Ambainis' work on VTAA, [28, 29].

Finally, several authors recently proposed to solve the QLSP with variational quantum algorithms (VQAs), [29–33], which will be discussed later. These algorithms are appealing since they are suitable for implementation on near-term devices and require only shallow quantum circuits. However, VQAs are heuristics, which means that the number of iterations to convergence is a-priori unknown.

2.1.4. AMPLITUDE AMPLIFICATION AND AMPLITUDE ESTIMATION

Amplitude amplification is an algorithm based on Grover's database search, [34, 35] that iteratively promotes the probability of one or more amplitudes of a quantum state. The problem of database searching

consists in finding a target bitsring m in a database of $N = 2^n$ entries, each of which is a n-bit string, such as

$$x = x_0 x_1 \dots x_n, \qquad x_i \in \{0, 1\}.$$

A function f marks the target bitstring m as follows

$$f(x) = \begin{cases} 1 & \text{if } x = m \\ 0 & \text{otherwise} \end{cases}.$$

Classical search methods require one to check the bitstring individually and therefore require in general $O(N)$ queris, $N-1$ of them in the worst-case scenario. On the other hand, Grover's algorithm works with all of the database entries in superposition and progressively promotes the amplitude of the target state $|m\rangle$. A popular starting state is the unbiased uniform superposition

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle,$$

where each j represents one of the N bitstrings.

In the quantum search algorithm, one applies the *Grover's operator*

$$G = -(\mathbb{1} - 2\,|\psi_0\rangle\langle\psi_0|)(\mathbb{1} - 2\,|m\rangle\langle m|) \tag{2.10}$$

k times, where $|\phi\rangle\langle\phi|$ is the projector onto the state $|\phi\rangle$. $k = O\left(\sqrt{N}\right)$ iterations are needed to prepare $|m\rangle$ with high probability [14], making the quantum search algorithm quadratically faster than classical searches. This quadratic speed-up can be understood intuitively by thinking that each time Grover's operator changes the amplitudes of the superposition, the probabilities change quadratically as much.

Of course, the target state $|m\rangle$ is generally unkwnown, and G must implement an *oracle* for f,

$$O_f\,|x\rangle\,|0\rangle = \begin{cases} |x\rangle\,|1\rangle & \text{if } \; x = m \\ |x\rangle\,|0\rangle & \text{otherwise} \end{cases},$$

where the notation implies a tensor product between neighbouring ket vectors.

The specifics of the oracle depend on the different problems and go beyond the scope of this discussion. It suffices to say that they generally entail a unitary contolled by a $n = \lceil \log N \rceil$ register, such that the unitary gets applied only when such register is in state $|m\rangle$ (or is in a superposition with nonzero probability of measuring $|m\rangle$).

Grover's algorithm was extended to multiple target states with quantum amplitude amplification (QAA), [36]. The Hilbert space

spanned by the database is divided here into a 'good' subspace, containing the target states and a complementary 'bad' subspace. Therefore, the Grover's operator becomes

$$G = -(\mathbb{1} - 2\,|\psi_0\rangle\langle\psi_0|)\left(\mathbb{1} - 2P_g\right), \tag{2.11}$$

where P_g is the projector onto the 'good' subspace. Thanks to Grover's search, a state in the 'good' subspace can be prepared quadratically faster than by using any classical search routine.

QAA can be used in those quantum algorithms that require post-selection. This is a procedure used as the last step of a quantum computation, where a quantum register contains the solution state with finite probability and it is entangled to an ancilla qubit. Measuring the ancilla in one of its two states will collapse the register to the solution state. Therefore post-selecting requires to repeat the entire quantum routine until the correct measurement of the ancilla is obtained. Taking quantum linear solvers as an example, the state before post-selection is

$$|\text{trash}\rangle\,|0\rangle + |x\rangle\,|1\rangle\,,$$

where $|x\rangle$ is the solution of the QLSP and $|\text{trash}\rangle$ represents the rest of the Hilbert space. In this case QAA amplifies the probabilities of the states that have the ancilla register equal to 1.

Also related to Grover's search is quantum amplitude estimation (QAE), which consists in performing quantum phase estimation on the Grover's operator. Because the action of G on a state $|\psi\rangle$ is a rotation of an angle 2θ in the plane defined by the components of $|\psi\rangle$ in the 'good' and 'bad' subspaces (see Figure 2.2),

$$G = \begin{bmatrix} \cos 2\theta & -\sin 2\theta \\ \sin 2\theta & \cos 2\theta \end{bmatrix} = e^{iY2\theta}, \tag{2.12}$$

where Y is the Pauli-Y operator,

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},$$

applying quantum phase estimation on G gives an approximation of the inner product of $|\psi\rangle$ with the 'good' subspace. i.e.

$$\left|\psi_g\right|^2 = \langle\text{good}|\psi\rangle = \sin^2\theta.$$

Quantum amplitude estimation can be useful to estimate the integral of a PDE solution over a certain region S, i.e. $\int_S u(x)\,\mathrm{d}, x$ [37]. As it will be clear in the following section, scalar quantitites of this type are generally the output of quantum routines for PDEs.

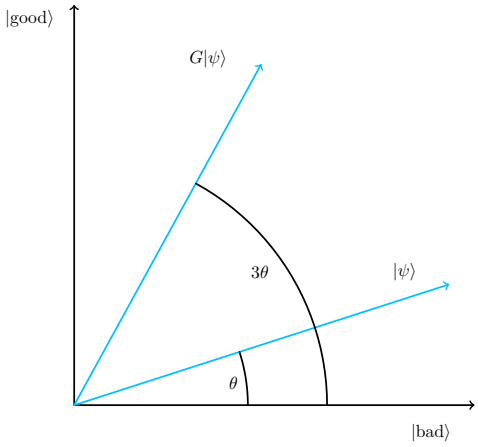

Figure 2.2.: Effect of one Grover iteration used in quantum amplitude amplification. The axes represent the components of $|\psi_0\rangle$ in the 'good' subspace and in the complementary 'bad' subspace.

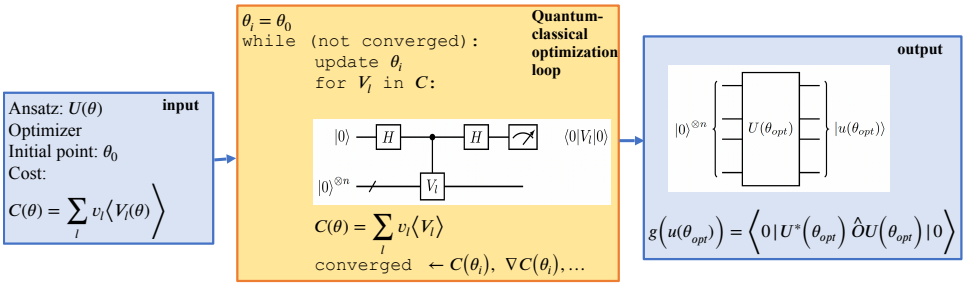

Figure 2.3.: Working principle of variational quantum algorithms. The inputs are the hyperparameters, such as ansatz, optimizer and type of cost function and the initial value of the ansatz parameter. The cost is shown here as a linear combination of expectation values of unitaries, although other expressions are possible. In the quantum-classical optimization loop, the classical computer is in charge of updating the parameters $\boldsymbol{\theta}$ according to the optimization logic, while the quantum computer evaluates the cost function terms. At the end of the optimization, the optimal parameter set $\boldsymbol{\theta}_{\text{opt}}$ can be used to reconstruct the approximate solution as a wavefunction ($|u(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|0\rangle^{\otimes n}$) or in a larger circuit to obtain a scalar function of the solution $g(u)$.

2.1.5. VARIATIONAL QUANTUM ALGORITHMS

Variational quantum algorithms (VQAs) are the main class of methods conceived to run on NISQ devices. Thanks to low quantum hardware requirements, variational quantum computing can work with noisy *physical qubits*, as opposed to fault-tolerant quantum computing (FTQC) which require fully error-corrected *logical qubits*.

Besides being suitable for near-term applications, VQAs are thought to be candidates for quantum advantage, in areas such as quantum chemistry, nuclear physics, but also optimization and machine learning, [38].

The small circuit depth is achieved via a hybrid strategy internal to an optimization process. At every iteration, a cost function is evaluated by the quantum computer and fed to the classical one, which updates the trainable parameters according to an optimization algorithm. Figure 2.3 shows schematically the working principle of VQAs.

VQAs have been reviewed in a dedicated work [38], illustrating their main concepts, applications and future prospects. What follows provides a succinct overview of the main elements of these algorithms.

ANSATZ

An ansatz is a parametrized unitary $U(\boldsymbol{\theta})$ that encodes the tentative solution of the problem. The parameters $\boldsymbol{\theta}$ are trained with an optimization routine, such that $U(\boldsymbol{\theta}_{\text{opt}})$ prepares the approximate solution of the original problem.

Though many families of ansatze exist, a first important distinction is between *problem-inspired* ansatze, which incorporate information of the problem, and the *problem-agnostic* ansatze. The prototypical problem agnostic ansatz is the *hardware-efficient ansatz*, [39], which optimizes the use and distribution of gates according to the hardware specifications. On the other hand, examples of problem-inspired ansatze are the Unitary Coupled Clustered ansatz, [40] and the Quantum Alternating Operator Ansatz, [41, 42].

COST FUNCTION

The cost function $C(\boldsymbol{\theta})$ is a metric of how far the tentative solution at the current iteration is from the actual solution of the problem. In VQAs, the cost function is computed by measuring different observables of the ansatz state $U(\boldsymbol{\theta})|0\rangle$. The type and number of observables can be either an algorithm design choice or depend on the problem.

Some of the requirements for a good cost function apply to quantum as well as classical methods. For example, the minimum of $C(\boldsymbol{\theta})$ must coincide with the solution of the problem and decreasing values of the cost function should correspond to better approximations of the solution. However, a good cost function for VQAs also needs to be

hard to evaluate classically, in order to retain the possibility of quantum advantage, [38].

COST FUNTION GRADIENT

Because of finite sampling and noise in the hardware and measurements, approximating the cost function's derivative with finite differences can lead to imprecise descent trajectories or even divergence of the optimization process. Luckily, the so-called *parameter-shit rule* (PSR) allows to compute gradients analytically using two cost function evaluations, similar to what happens in finite differences. PSR computes the derivative of the cost C with respect to the parameter θ_l

$$\frac{\partial C}{\partial \theta_l} = \frac{1}{2\sin\alpha}\left(C(\boldsymbol{\theta}^+) - C(\boldsymbol{\theta}^-)\right), \tag{2.13}$$

where $\boldsymbol{\theta}^\pm = \theta \pm \alpha \mathbf{e}_l$, $\mathbf{e}_l$ is the vector with 1 in the l^{th} entry and 0 in all others and α is the magnitude of the shift.

Although Equation (2.13) looks similar to the finite difference formula, the two differ by the term $\frac{1}{2\sin\alpha}$. When the shift is $\alpha = \pi/2$, the difference in statistical error between finite differences and PSR is maximum, [43]

OPTIMIZER

As mentioned, cost functions of VQAs are often complex due their noisy nature or due to flat regions known *barren plateaus*, [44, 45].

A main classification of VQA optimizers is between gradient-based and gradient-free algorithms. In principle, any classical optimizer can be used in VQAs, but the noisy character of the cost function makes some choices better than others in avoiding stability and convergence issues. Popular gradient-based techniques are stochastic gradient descent (SGD) variants, such as Adam, [46] or natural gradient descent, [47]. On the other hand, the simultaneous perturbation stochastic approximation (SPSA) method is probably the most popular gradient-free technique, [48].

QUANTUM ANNEALING

Quantum annealing (QA) is an algorithmic approach utilized for addressing combinatorial optimization (CO) problems. These problems require the exploration of a finite set of variables or choices with the objective of identifying the 'optimal' solution based on a specified metric of merit. Notably, a significant number of combinatorial optimization problems are classified as NP-hard. Due to their inherent complexity, CO problems are typically approached with approximate solutions, and the primary challenge lies in generating accurate approximations within a short (polynomial) timeframe.

QA transforms the combinatorial optimization problem to that of finding the ground state of the Ising Hamiltonian [49]

$$H_{\text{Ising}} = \sum_{i}^{n} H_i q_i + \sum_{(ij)} J_{ij} q_i q_j, \tag{2.14}$$

where q_i, q_j represent the binary values associated to the qubits, for instance, the different spins of the quantum particles. Also, $H_i \in \mathbb{R}$ is the local field at site i and $J_{ij} \in \mathbb{R}$ is the interaction strength between qubits i and j.

In order to reach to the ground state of H_{Ising}, the quantum annealing process starts from an initial Hamiltonian H_0, whose ground state is known and easy to prepare. A common choice is the transverse magnetic field

$$H_0 = -\sum_{i}^{n} X_i, \tag{2.15}$$

X_i being the Pauli-X operator acting on the i^{th} qubit, where

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

From here, the system Hamiltonian is slowly changed according to an evolution law, such that

$$H(t) = (1 - f(t))H_0 + f(t)H_{\text{Ising}} \qquad \text{for} \quad t \in [0, T]. \tag{2.16}$$

Here, T is the total evolution time, $f(t) \in [0, 1]$ and $f(0) = 0$, $f(T) = 1$. If $H(t)$, known as the *total Hamiltonian*, is changed slowly enough, then the adiabatic principle ensures that the system's configuration is at the ground state of $H(t)$ at all times, [50]. At the end of adiabatic evolution, the system will therefore be in the ground state of the Ising Hamiltonian, which corresponds to the approximate solution of the combinatorial optimization problem.

However, quantum annealers operate with spin systems, where each spin s_i is a binary variable equal to ± 1. Thus, QA programmers need to transform the CO problem to a binary optimization one. In D-Wave machines, which are the state-of-the-art quantum annealers, the problem is given as input in the Quadratic Unconstrained Binary Optimization (QUBO) form, that is

$$\begin{aligned} \text{minimize} \quad & \mathbf{q}^\top Q \mathbf{q} \\ \text{with} \quad & \mathbf{q} \in \{0, 1\}^n, \end{aligned} \tag{2.17}$$

where $Q \in \mathbb{R}^{n \times n}$. The QUBO problem in Equation (2.17) can be turned into the Ising Hamiltonian ground state problem by using the mapping $s_i = 2q_i - 1$.

2.1.6. MEASUREMENT

At the end of the quantum primitive the problem's solution is encoded as a quantum state, which is a unit vector $|u\rangle \in \mathbb{C}^N$, whose information must be accessed through measurement. However, measuring the entire state vector requires $O(N)$ measurements, breaking any speed-up given by the quantum primitive.

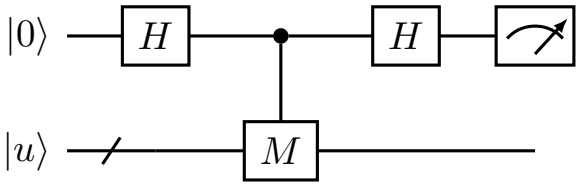

Figure 2.4.: Hadamard Test circuit for computing $\langle u|M|u\rangle$. In contrast to standard observable measurements, M needs to be unitary.

A scenario where measurement might be performed in polynomial time is when to obtain a scalar function $g(u)$ of the solution. This function is often taken as the expectation value of an observable M, i.e.

$$g(u) \propto \langle M\rangle := \langle u|M|u\rangle\,.$$

A popular way to compute expectation values is the *Hadamard Test*, [51], which assumes that M is a unitary operator. As shown in Figure 2.4, M is controlled by an ancilla qubit in uniform superposition, which is later measured in the Pauli-X basis. By solving the circuit in Figure 2.4, one obtains

$$\mathrm{Re}\{\langle M\rangle\} = \frac{1}{2}\left(\Pr(0) - \Pr(1)\right), \tag{2.18}$$

where $\Pr(0)$ and $\Pr(1)$ are the probabilities of the ancilla qubit collapsing to $|0\rangle$ and $|1\rangle$ respectively.

Also computable in polynomial time is the overlap of the solution state $|u\rangle$ with a reference state $|u_{\mathrm{ref}}\rangle$. This can be accomplished using the *SWAP Test*, [52], which is a Hadamard test where M is equal to the *SWAP* gate. If $|\psi_1\rangle$ and $|\psi_2\rangle$ are the states of two qubits, the *SWAP* gate acts between them as

$$SWAP\,|\psi_1\rangle\,|\psi_2\rangle = |\psi_2\rangle\,|\psi_1\rangle\,.$$

In the SWAP Test, the *SWAP* operator is actually generalized between the quantum registers containing $|u\rangle$ and $|u_{\mathrm{ref}}\rangle$. The overlap between the two states is given by the probabilities of the ancilla qubit as

$$\mathrm{Re}\{\langle u|u_{\mathrm{ref}}\rangle\} = \frac{1}{2}\left(\Pr(0) - \Pr(1)\right). \tag{2.19}$$

In practice, all expectation values are determined statistically as averages. The number of samples N_m required to approximate $\langle M \rangle$ up to precision ε is given by the Chernoff-Hoeffding inequality, for which

$$\Pr\left(\left|\hat{M} - \langle M \rangle\right| \geq \varepsilon\right) \leq 2e^{-2N_m\varepsilon^2}, \tag{2.20}$$

where $\hat{M} = 1/N_m \sum_{i=1}^{N_m} M_i$ is the average of the measurements and $M_i \in \{0, 1\}$. Therefore, $O(1/\varepsilon^2)$ measurements are necessary to esimtate $\langle M \rangle$ up to precision ε.

2.2. FUNDAMENTAL PDES IN STRUCTURAL MECHANICS

To frame the following discussion, we present here the equations of mechanical and thermal equilibrium of a generic three-dimensional solid. Even though such an abstract model is rarely used in practice, it allows us to derive the equilibrium of actual materials and structural elements. Throughout the rest of the chapter, we will show how the PDEs reviewed relate to the generic equilibrium of a solid, in order to understand the level of modeling complexity that current quantum methods deal with.

For the remainder of the section, we will consider the equilibrium of a generic three-dimensional solid of volume Ω and surface $\partial\Omega$. Although not specifically treated, all the following PDEs should be complemented by an appropriate set of boundary conditions in order to completely define the differential problem.

2.2.1. MECHANICAL LINEAR EQUILIBRIUM

Under the assumption of small deformations, the generic three-dimensional solid is in static equilibrium if

$$\nabla \cdot_{\mathbf{x}} \tau - \mathbf{b} = 0, \qquad \mathbf{x} \in \Omega \subseteq \mathbb{R}^3, \tag{2.21}$$

where $\nabla\cdot$ is the divergence operator and $\mathbf{x}$ are the coordinates of the deformed configuration of the solid. For small deformations, the deformed configuration coincides with the reference one. Furthermore, $\mathbf{b}$ are the body forces acting on the solid, while τ is the second-order Cauchy stress tensor, which relates to the strain tensor ϵ via the consitutive law of the material,

$$\tau = C(\mathbf{x})\,\epsilon. \tag{2.22}$$

The fourth-order constitutive tensor $C(\mathbf{x})$ can be a function of the space coordinates for inhomogeneous materials. Finally, the kinematics of small deformations relate the strain tensor ϵ to the displacement field $u(\mathbf{x})$ as

$$\epsilon = \frac{1}{2}\left(\nabla_{\mathbf{x}}\mathbf{u} + \nabla_{\mathbf{x}}^{\top}\mathbf{u}\right). \tag{2.23}$$

Equation (2.21) describe the static equilibrium of a solid. By introducing the accelleration of the body, we obtain the linear dynamic equilibrium equations,

$$\rho(\mathbf{x})\frac{\partial^2 \mathbf{u}}{\partial t^2} + \nabla \cdot_{\mathbf{x}} \tau - \mathbf{b} = 0, \qquad \mathbf{x} \in \Omega \subseteq \mathbb{R}^3, \tag{2.24}$$

where $\rho(\mathbf{x})$ is the density of the material.

2.2.2. MECHANICAL NONLINEAR EQUILIBRIUM

In presence of large deformations, the reference configuration $\mathbf{X}$ and the deformed configuration $\mathbf{x}$ do not coincide. One way to relate the two quantities is through the deformation gradient,

$$F = \frac{\partial x}{\partial X} = I + \frac{\partial u}{\partial X}. \tag{2.25}$$

The deformation gradient also relates the tensor of stresses in the reference configuration P, or 1^{st} Piola-Kirchhoff stress tensor, to the Cauchy stress tensor τ,

$$P = \det(F)\,\tau F^{-\top}. \tag{2.26}$$

Thanks to these quantities, one can write the balance of a solid undergoing large deformations in its reference configuration and finally obtain the following (dynamic) equilibrium equation,

$$\rho(\mathbf{X})\frac{\partial^2 \mathbf{u}}{\partial t^2} + \nabla \cdot_{\mathbf{X}} P - \mathbf{b_0} = 0, \qquad \mathbf{X} \in \Omega_0 \subseteq \mathbb{R}^3, \tag{2.27}$$

where $\mathbf{b_0} = \det(F)\,\mathbf{b}$ is the vector of body forces written in the reference configuration. The nonlinearity of Equation (2.27) in the displacement $\mathbf{u}$ is given by the expression of the 1^{st} Piola-Kirchhoff stress tensor in Equation (2.26).

Alternatively to large deformations, nonlinear mechanical behaviour can also be due to material nonlinearities. In theses cases, which include hyperelasticity, plasticity and damage, the consititutive tensor C in Equation (2.22) depends, in some fashion, on the displacement $\mathbf{u}$.

2.2.3. THERMAL EQUILIBRIUM

Similarly to the balance of mechanical energy, one can write the balance of thermal energy of a solid and obtain the following PDE,

$$\rho c \frac{\partial T}{\partial t} - \kappa \nabla^2_{\mathbf{x}} T - Q(\mathbf{x}, t) = 0. \tag{2.28}$$

The thermal equilibrium equation describes how the temperature T of the solid evolves in the presence of conductive heat exchange and

possible internal heat sources $Q(\mathbf{x}, t)$. The material quantities ρ, c and κ are respectively the density, specific heat and coefficient of thermal conductivity of the material and can be in general a function of $\mathbf{x}$ and t.

Equation (2.28) can become nonlinear if one of the material properties depends on the temeprature T or with a nonlinear $Q(x, t)$ term. Common cases are those where the thermal conductivity depends on T, such as changes of phase or degradation of the surface.

2.3. QUANTUM ALGORITHMS FOR LINEAR PDES

The bulk of the literature that we gathered on quantum solvers for linear PDEs focuses on the Poisson, heat and wave equations. The different techniques are analyzed in the following sections.

2.3.1. POISSON EQUATION

Let $u(\mathbf{x}) \in \mathbb{R}$ be the unknown solution function, where $\mathbf{x} \in \Omega \subseteq \mathbb{R}^d$, d is the number of spatial dimensions and $f(x) \in \mathbb{R}$ is a known source term. Then the Poisson equation in Cartesian coordinates reads

$$-\left(\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2}\right) = -\nabla^2 \mathbf{u} = f(\mathbf{x}), \qquad \mathbf{x} = [x_1, \ldots, x_d]^\top \in \Omega. \tag{2.29}$$

Equation (2.29) must be complemented with Dirichlet, Neumann or Robin conditions on the boundary $\delta\Omega$ in order to find a unique solution.

In mechanics, the Poisson equation describes the static deformations of thin elements such as rods and membranes over their axis or plane. In fact, for $d \leq 3$ and purely volumetric deformations, that is, when the consitutive tensor C in Equation (2.22) reduces to a constant, the linear elastic equilibrium Equation (2.21) takes the form of Equation (2.29).

Assuming a hypercubic domain, the standard technique to solve Equation (2.29) numerically is to discretize the domain in N grid points in each of the d spatial dimensions and either approximate the Laplacian with a central finite difference (FD) scheme or employ a finite element (FE) approximation. Both approaches lead to a discrete equation, or *Discrete Poisson Equation*, that is a linear systems of $(N-2)^d \times (N-2)^d$ equations of the form

$$A\mathbf{u} = \mathbf{f}, \tag{2.30}$$

where $u_i = u(x^{(i)})$ is the solution at grid point $x^{(i)}$ and $f_i = f(x^{(i)})$ for finite differences or $\mathbf{f} = \int_\Omega f(x)\varphi_i(x)$ for finite elements with basis functions $\varphi_i(x)$. Equation (2.30) can be solved classically using direct or iterative solvers. However, all classical linear solvers have a cost upper bound by a polynomial in N ($O(\text{poly}(N))$ complexity). Even the fastest iterative solvers such as conjugate gradient, [24] or multigrid techniques, [53] require $O(N)$ iterations.

HHL AND PRECONDITIONING

QLSAs such as HHL (cfr Section 2.1) seem natural candidates to solve the discrete Poisson problem exponentially faster than classical methods. Several authors worked on this concept. Clader et al. were among the first to apply an improved version of HHL to a finite difference discretization of the stationary Maxwell equations, [26]. These form a system of PDEs that are not of the Poisson type but still elliptical and can be reduced to a linear system such as Equation (2.30) upon discretization.

As mentioned, Clader et al. did not use the standard HHL, but a modified version of it. Most importantly, they noticed that the $O(\kappa^2)$ in the cost of HHL nullifies the exponential speed-up in N, if $\kappa = O(N)$, which is the case for matrices induced by a finite element discretization of second-order elliptical boundary value problems, for which $\kappa = O(N^{2/d})$, [54]. Therefore, Clader et al. proposed a quantum algorithm to reproduce the Sparse Preconditioner Approximate Inverse (SPAI), [55] operator. This technique uses a matrix P to achieve nearly optimal preconditioning, i.e. $PA \approx I$, where I is the identity matrix. At the same time, P is such that PA preserves the sparsity of A. If s is the sparsity of A in Equation (2.30) and the preconditioner P has similar sparsity, then applying the P such that

$$PA\mathbf{u} = P\mathbf{f}, \tag{2.31}$$

can be done in $O(s^2)$ queries to an oracle accessing PA and $O(s^3)$ runtime, [26]. Then, if the SPAI procedure is applied succesfully, the conditioning will be independent of N or $O(\log(N))$, eliminating the polynomial scaling, that would still be present even in improved QLSAs such as [25]. If s and $1/\varepsilon$ are also constant or logarithmic in N, the complexity of Clader's method would be $O(\text{poly}\log(N))$, which is a true exponential speed-up. Unfortunately, in most if not all discretization schemes, $1/\varepsilon = O(N)$, as will be discussed later in this section.

Further than conditioning, Clader et al. discussed two other caveats of HHL, [56], namely the state preparation and the measurement problems. For state prepatation, they proposed to use an oracle able to return $f_j \in \mathbb{R}$ and $\phi_j \in \mathbb{R}$ as superposition states, such that

$$|f\rangle = \sum_{j=0}^{N-1} f_j e^{i\phi_j} |j\rangle . \tag{2.32}$$

However, even though one would query this oracle a constant number of times to produce $|f\rangle$, it could be that the same oracle would compile to an exponential number of native gates, effectively only rephrasing the state preparation problem.

About measurements, [26] provide examples of classical quantities that can be extracted once $|x\rangle$ has been prepared with the HHL

algorithm. In particular they show how to compute $\langle r|x\rangle$, for an arbitrary state $|r\rangle$ and $\langle j|x\rangle$, i.e. the j-th component of $|x\rangle$.

As mentioned, [26] also describe how the preconditioned HHL can be applied to Maxwell's equations, and, if a scalar solution $\langle r|x\rangle$, such as the scattering cross section, is required, then the speed-up is exponential in N. However, Clader's method is problem-agnostic and the Maxwell discrete problem is seen more as a linear system for benchmarking their solver, rather than a discretized PDE. Moreover, all speed-up results are based on the existence and the efficiency of oracles, which are black-box gates. In the particular case of Maxwell's equations, the authors only mention that an oracle implementing A and $\mathbf{f}$ would be 'efficient', but do not elaborate on the number of submodules, gates, etc [26]. Furthermore, the complexity analysis in [26] is incomplete, as later noticed by Montanaro and Pallister, [15]. In fact the preconditioned HHL still retains a $O(\text{poly}(1/\varepsilon))$ factor and since for local FE schemes $N = O\left((1/\varepsilon)^{\alpha}\right)$, with $\alpha \in (0,\, 1)$, this primitive still does not achieve an exponential speed-up for elliptical problems.

DIAGONALIZATION WITH QUANTUM FOURIER TRANSFORM

While still using a QLSA, Cao et al. focused on the problem rather than the solver, [57]. In their work, they treated the d-dimensional Poisson equation (Equation (2.30)) and noticed that every classical linear solver requires at least

$$O\left(\sqrt{\kappa}\left(\frac{1}{\varepsilon}\right)^{\alpha d}\log\left(\frac{1}{\varepsilon}\right)\right) \tag{2.33}$$

time, therefore suffering from the curse of dimensionality. Using the HHL algorithm, Cao *et al.* resolved the exponential dependency on d, by preparing the solution $|x\rangle$ (as a quantum state) with

$$\max\left\{d, \log\left(\frac{1}{\varepsilon}\right)\right\}\left(\log\left(\frac{1}{\varepsilon}\right)\right)^{3} \tag{2.34}$$

quantum operations. Noticeably, both Equations (2.33) and (2.34) are completely expressed in terms of $1/\varepsilon$ and d, thus not hiding any dependencies between complexity terms. In addition, the authors described the quantum circuits used from submodules to gates.

Comparing Equation (2.34) that of the HHL (Equation (2.8)), one notices that the $O(1/\varepsilon)$ term in the cost of Cao's algorithm is replaced by a logarithmic dependency. The exponential reduction does not derive from modifications to the linear solver, but from the specific structure of the Poisson matrix. This can be understood starting from the 1-dimensional case, defining A_P^1 as the 1D Poisson matrix. This is an instance of the class of Toeplitz matrices, all of which can be

diagonalized by the sine transform S, [58], as

$$A_P^1 = S\Lambda S^\dagger, \tag{2.35}$$

where $\Lambda = \mathrm{diag}(\lambda_i)$, $\lambda_i = 4\sin^2\left(\frac{i\pi}{2N}\right)$ and $S_{ij} = \sqrt{\frac{2}{N}}\sin\left(\frac{ij\pi}{N}\right)$ are the components of the discrete sine transform. Cao *et al.* proved that estimating the eigenvalues of Equation (2.35) up to precision ε requires $O(\log(1/\varepsilon))^3$ quantum operations. Moving to multiple dimensions, the d-dimensional Poisson matrix A_P^d can be written as

$$A_P^d = A_P^1 \otimes I \otimes \cdots \otimes I + I \otimes A_P^1 \otimes I \otimes \cdots \otimes I + \cdots + I \otimes \cdots \otimes I \otimes A_P^1 \tag{2.36}$$

and its Hamiltonian simulation is

$$e^{iA_P^d t} = e^{iA_P^1 t} \otimes \cdots \otimes e^{iA_P^1 t}. \tag{2.37}$$

Therefore, simulating A_P^d requires time $O\left(d\log^3(1/\varepsilon)\right)$, that is exponentially better than vanilla HHL in the solution precision.

Equation (2.34) should not mislead the reader in thinking that the cost of Cao's method is independent of the conditioning number. The runtime refers to a single run of the quantum circuit. $O(\kappa^2)$ runs are still required to post-select the solution state based on measurements of the ancilla qubit. If $\kappa = O(N^{d/2})$, the exponential advantage is lost, but Clader's preconditioning technique, [26], may be used to achieve constant or logarithmic dependency of the conditioning with respect to N.

Wang *et al.* further built on Cao's work, [59]. In particular, they presented a fully modular circuit of this algorithm, defining every module in its components, down to known quantum operations (such as addition and subtraction) and gates. They noticed that several steps in Cao's algorithm could be reduced to the evaluation of trascendental function. For this sake, they developed the so-called *quantum function-value binary expansion* (qFBE) algorithm, [60], which allows to replace more expensive power operations with arithmetic ones.

Moreover, [59] showed how to reduce the complexity of the controlled rotation operation in Cao's method. In the latter, after the quantum phase estimation step is performed, the eigenvalue register $|\lambda_j\rangle$ is used to compute the reciprocal state $|1/\lambda_j\rangle$ using a quantum version of Newton's method. Then, the controlled rotation $C{-}R_y(\theta_j)$ step is used to encode amplitude $1/\lambda_j$, if the rotation angle is taken as $\theta_j = \arcsin(1/\lambda_j)$. Cao et al. used a quantum implementation of the bisection method, to iteratively evaluate the arc sine function, requiring $O(\log^4(1/\varepsilon))$ operations for a small-dimension, high-precision ($d < \log(1/\varepsilon)$) problem. However, Wang et al. bypassed the computation of the reciprocal state

completely and noticed that the angles for the controlled rotation could be estimated with minimum error through the following relation

$$\theta_j = \omega_j \pi; \quad \omega_j \approx \frac{\operatorname{arccot}(\lambda_j)}{\pi}. \tag{2.38}$$

Also, the arc-cotangent function could be estimated through the qFBE algorithm in $O(\log^3(1/\varepsilon))$ operations.

Overall, Wang's Poisson solver produces the solution state $|x\rangle$ in $O(\kappa d \log^3(1/\varepsilon))$ operations, where the usual consideration about the conditioning number applies. Comparing this scaling with that of Cao's algorithm (cfr Equation (2.34)), Wang's method is polynomially more efficient for low dimensional and high precision problems, which are usually the most interesting ones in structural analysis.

ADAPTIVE ORDER SCHEME AND SPECTRAL METHOD

Cao [57] and Wang [59] approximated the Laplacian in Equation (2.29) three points centered finite differences for all grid sizes. However, Childs *et al.* recently remarked that fixed finite difference, finite element and finite volume schemes require $O(\text{poly}(1/\varepsilon))$ time to bring the approximate solution $|\tilde{u}\rangle$, ε-close to the actual solution on the grid [61]. In fact, fixed schemes produce matrices with $\kappa = O(\text{poly}(1/\varepsilon))$ and all quantum linear solvers have polynomial time in the conditioning number.

Childs et al. accounted for this issue, by solving Poisson and general second order elliptical boundary value problems with two different numerical approximations, namely an adaptive order FD approximation and a spectral method, [61]. The adaptive FD approach is used to solve the d-dimensional Poisson problem with periodic boundary conditions.

The authors showed that the conditioning number κ of the order-k Laplacian with periodic boundary conditions is $O(N^2)$ if $k \leq \left(6/\pi^2\right)^{1/3} N^{2/3}$. Then, by assuming that the error due to the finite difference discretization and due to the quantum linear system algorithm are of the same order of magnitude, a relationship can be established between N, k, d and $1/\varepsilon$. Choosing $k = \left(6/\pi^2\right)^{1/3} N^{2/3}$ is found to be optimal in terms of runtime, [61] and N is then automatically determined to ensure the total error is upper bound as $O(\varepsilon)$:

$$N = \Theta\left(d^{3/2} \log^{3/2}\left(\left|\frac{\partial^{2k+1} u}{\partial x^{2k+1}}\right| / \varepsilon\right)\right). \tag{2.39}$$

By substituting Equation (2.39) into the runtime of the complexity-optimal QLSA solver of [27], the solution of the second order elliptical problem with periodic BCs is found in

$$O(d^{3/2} \text{poly}\left(\log(d), \log(1/\varepsilon)\right)) \tag{2.40}$$

runtime, meant as number of gate operations, which is polynomial in d (i.e. no curse of dimensionality) and has optimal dependency in $1/\varepsilon$.

Furthermore, the authors show that the same runtime holds for Dirichlet or Neumann homogeneous boundary conditions. To achieve this, they use the method of images to extend the domain and symmetrize the solution u according to the BCs, [61].

The second algorithm proposed in [61] uses the spectral method. In this case, the solution is approximated globally and the discretized Laplacian is non-sparse. To obviate to this problem, two variations of the quantum Fourier transform, namely the *quantum shifted Fourier transform* (QSFT) and the *quantum cosine transform* (QCT) are used to induce sparsity. The algorithm makes use of an oracle that is queried

$$d^2 \mathrm{poly}\left(\log\left(1/\varepsilon\right)\right) \tag{2.41}$$

times for second-order elliptic problems with non-homogeneous Dirichlet boundary conditions and

$$d \mathrm{poly}\left(\log\left(d\right), \log\left(1/\varepsilon\right)\right) \tag{2.42}$$

times for the Poisson problem with homogeneous Dirichlet BCs.

FULL COMPLEXITY ANALYSIS

All previous techniques are based on quantum linear system algorithms and demonstrate times that are exponentially better than classical solvers. Yet it is important to understand what output is prepared in such time and what input provides the starting point. To begin with, all algorithms for solving linear systems $A\mathbf{x} = \mathbf{b}$ require that the right hand side vector is given as an input in normalized form, i.e. $|b\rangle$. In classical computation, keeping the input in the computer's main memory is common practice, since random access memories (RAMs) can store arrays of double-precision values over a long time and enable repeated readout. In terms of output, classical linear solvers produce the entire solution vector $\mathbf{x}$ and therefore the full discrete solution.

In quantum computers, information is encoded and processed as quantum states. Therefore, any computation must prepare the input states before the quantum primitive and measure the quantum register after the primitive. For example, the HHL algorithm requires the right-hand side vector $\mathbf{b}$ to be provided as a quantum state $|b\rangle$. If this state had been previously *stored* in a quantum RAM (QRAM), then the cost of state preparation would not fall on the algorithm. However, even though QRAM models exist and can theoretically create quantum superposition states in $O(\log(N))$ time, [62, 63], it is uncertain whether they can be physically built and if they actually offer an advantage if a parallel computer with the same amount of resources is available, [64]. On the other hand, $|b\rangle$ may be *prepared* through a sequence of

unitary operations. Nonetheless, the task of preparing a generic state is an $O(N)$ problem, [14], which would eliminate any possible exponential savings.

The full complexity of a QLSA for differential problems, from input encoding to output readout was discussed by Montanaro and Pallister, [15]. Specifically, they considered discretized elliptical PDEs using the finite element method, using piecewise polynomial basis functions.

Montanaro and Pallister assumed the final output to be the inner product $\langle r|u\rangle$. The state $|r\rangle$ is the quantum state representation of a known function $r(x)$ defined over the domain Ω

$$|r\rangle = \frac{1}{\sqrt{\sum_{i=1}^{N} \langle \varphi_i|r\rangle^2}} \sum_{i=1}^{N} \langle \varphi_i|r\rangle \, |i\rangle \,, \tag{2.43}$$

where φ_i are the finite element basis functions and the vectors $|i\rangle$ form a basis for $\mathbb{C}^N$.

The main findings of [15] are summarized in Table 2.1. The classical linear solver used is the conjugate gradient method, while the quantum solver is that in [27], which has logarithmic dependency on $1/\varepsilon$. The basis functions used for the results in Table 2.1 are the linear 'hat' functions

$$\varphi_i(x) = \begin{cases} \frac{1}{h}(x - x_{i-1}) & \text{if} \quad x \in [x_{i-1},\, x_i] \\ \frac{1}{h}(x_{i+1} - x) & \text{if} \quad x \in [x_i,\, x_{i+1}] \\ 0 & \text{elsewhere} \end{cases}, \tag{2.44}$$

defined over a uniform grid $x_i \in [x_0, x_1, \ldots, x_{N-1}]$ with equidistant spacing h. The paper also presents the complexities for discertizations with polynomial shape function of order p. The complexity analysis was performed both without preconditioning and with optimal preconditioning (i.e. $PA = I$), where a realistic preconditioning case lies between these two extremes.

The most important result in Table 2.1 is that, for fixed dimension d, no exponential quantum speed-up can be achieved, regardless of preconditioning. The reason is that to compute $\langle r|x\rangle$ up to precision ε requires $O(1/\sqrt{\varepsilon})$ repetitions of the QLSA operator while doing quantum amplitude amplification, [15].

It may seem that, if d was allowed to vary, the quantum solver would be exponentially faster (no curse of dimensionality). However, the authors warn that the $\tilde{O}$ notation hides terms that are independent of $1/\varepsilon$ but can vary with d, making it hard to say what kind of speed-up is achievable for variable dimensions.

The runtimes in Table 2.1 scale with $1/\varepsilon$, $|u|_l$ $\|u\|_l$, where the last two quantities are respectively the Sobolev l-seminorm and l-norm of the analytical solution u, that depend on the derivatives of u up to the l-th order. Therefore for high PDE dimensions d, there can be a

Table 2.1.: Complexity results in [15]. The problem is a second order elliptical PDE, discretized with the finite element method, using piece-wise linear shape functions.

Algorithm	No preconditioning	Optimal preconditioning
classical (conjugate gradient)	$\tilde{O}\left(\left(\frac{\lvert u\rvert_2}{\varepsilon}\right)^{(d+1)/2}\right)$	$\tilde{O}\left(\left(\frac{\lvert u\rvert_2}{\varepsilon}\right)^{d/2}\right)$
quantum [27]	$\tilde{O}\left(\lVert u\rVert \frac{\lvert u\rvert_2^2}{\varepsilon^3} + \lVert u\rVert_1 \frac{\lvert u\rvert_2}{\varepsilon^2}\right)$	$\tilde{O}\left(\frac{\lVert u\rVert_1}{\varepsilon}\right)$

consistent polynomial speed-up for high precision problems and large second derivatives of the solution $u(x)$. These considerations extend to higher-order finite elements, with the difference that higher-order Sobolev seminorms appear in the runtime. In classical mechanics, however, elliptical problems reach at most $d = 3$, limiting the achievable polynomial speed-up.

To understand why, even considering a $\log(1/\varepsilon)$-scaling algorithm, Montanaro and Pallister found that quantum linear solvers cannot provide exponential speed-up, one can answer the questions identified by Aaronson in his 'fine print' for quantum linear algebra, [56].

1. *Can* $|b\rangle$ *be prepared in time* $O(\log(1/\varepsilon))$*, starting from* $\mathbf{b}$*?* In general, this is a hard problem, but if $f(x)$ in Equation (2.29) is a polynomial or a function supported only on a few elements, the state $|b\rangle$ can indeed be prepared in time $O(\log(1/\varepsilon))$, [15].

2. *Is there an algorithm for accessing the elements of* A *in time* $O(\log(1/\varepsilon))$*?* Yes. Quantum linear solvers require a sparse matrix and a sparse access to the matrix. The second point means that having an algorithm that, given a row index r and another index i, returns the column index and value of the i -th nonzero element in A. Finite element matrices satisfy both these requirements, since they are sparse and, if the mesh is regular, sparse access can be obtained by knowledge of the element's degrees of freedom and by the connectivity matrix.

3. *Is is possible to apply efficient pre-conditioning to* A*?* Yes. One way is the quantum-SPAI technique proposed by Clader et al., [26].

4. *Is it possible to measure the output in time* $O(\log(1/\varepsilon))$*?* Not in general. Especially, it is not the case for estimating properties such as $\langle r|u\rangle$, where distinguishing between two quantum states that are ε-close to each other require $O(1/\sqrt{\varepsilon})$ queries to the QLSA operator, [15].

Therefore, measurement generally is the computational bottleneck in the pipeline, even though particular properties, such as periodicity, can be verified in logarithmic complexity, [15].

NISQ SOLVERS

The algorithms discussed previously require fault-tolerant hardware. For instance, Wang could demonstrate his algorithm only to solve a minimal problem of a 4×4 Poisson matrix, [59], on the Sunway TaihuLight supercomputer, acting as a quantum simulator, [65]. Interestingly, the authors also provide qubit and gate counts for this implementation, declaring 38 qubits and 800 gates, among which *TOFFOLI*s and *SWAP*s, that must be further compiled and mapped to actual hardware connectivity. Therefore, circuits of this size require quantum volumes beyond near-term capabilities by orders of magnitude. Consequently, other authors recently looked at the possibility to solve the Poisson equation with near-term techniques.

Wang *et al.* also proposed a way to solve the one-dimensional Poisson problem in NISQ, using circuits of $O(\text{poly}\log(1/\varepsilon))$ operations, [66]. Their main idea was that Hamiltonian simulation can be bypassed if one is able to directly encode the inverse eigenvalues of the matrix A in the quantum state amplitudes.

The Poisson problem with homogeneous Dirichlet boundary conditions results in a matrix whose eigenvalues have an analytical expression, i.e.

$$\lambda_j = 4N^2 \sin^2\left(\frac{j\pi}{2N}\right), \tag{2.45}$$

where j ranges from 1 to N-2.

Wang *et al.* noticed that this Poisson matrix is a Cartan matrix. The eigenvalues of Cartan matrices are also sines and they follow the following product relation, [67]

$$2^{2^{n+1}-2} \prod_{j=1}^{2^n-1} \sin^2\left(\frac{j\pi}{2^{n+1}}\right) = 2^n, \tag{2.46}$$

where $n = \log(N)$.

Therefore, the sine terms in the product are equal to the eigenvalues in Equation (2.45) up to a constant term. Consequently, $1/\lambda_j$ can be written as a product of all other eigenvalues λ_k, for $k \neq j$.

However, implementing Equation (2.46) requires $O(N)$ qubits, since every inverse eigenvalue depends on $2^n - 1$ others. The authors anyway show that the periodicity of the discrete sines and some trigonometric relations allow to compute $1/\lambda_j$ from Equation (2.46) as product of only $n-2$ sine terms. This allows the algorithm to be implemented in $3n$ qubits. In terms of implementation, the sine expressions in

Equation (2.46) can be performed straightforwardly as controlled R_y rotations.

Wang et al. also shows the circuit that implements their algorithm, which requires $\frac{5}{3}n^3$ one and two qubit gates. They also mention that parallelization of the controlled R_y operations can further reduce the gate count to $10n^2$, by adding $n-2$ ancillary qubits. For a few ($n < 10$) qubit problems the required number of gates may fit the specifications of NISQ devices.

Despite proposing a polynomial-time primitive with low gates count, [66] does not elaborate on state preparation and measurement. Another shortcoming of the method is that it is limited to the one-dimensional Poisson problem with homogeneous Dirichlet boundary conditions. The associated tridiagonal matrix can be inverted analytically, eliminating the need for such a specialized numerical technique [68]. Unfortunately, the authors do not present extensions of their method to higher dimensions or different boundary conditions.

Another approach amenable to NISQ devices is to use variational quantum algorithms to invert the 1D Poisson linear system, [69]. In this case, a variational state $|x\rangle = U(\boldsymbol{\theta})|0\rangle$ is prepared by an ansatz $U(\boldsymbol{\theta})$ (see Section 2.1.5) and the loss functions needs to be zero when $|x\rangle = A^{-1}|b\rangle$. A possible choice for such a loss function is

$$\mathcal{L}(\boldsymbol{\theta}) = \langle x(\boldsymbol{\theta})|A^\dagger A|x(\boldsymbol{\theta})\rangle - |\langle b|A|x(\boldsymbol{\theta})\rangle|^2. \tag{2.47}$$

In the Poisson case the matrix A is symmetric, therefore $A^\dagger A = A^2$ and Equation (2.47) becomes

$$\mathcal{L}(\boldsymbol{\theta}) = \langle x(\boldsymbol{\theta})|A^2|x(\boldsymbol{\theta})\rangle - |\langle b|A|x(\boldsymbol{\theta})\rangle|^2. \tag{2.48}$$

A necessary condition for advantage is that the terms in Equation (2.48) can be efficiently evaluated on a quantum computer and are hard to evaluate classically. It was proved that the latter requirement is satisfied for loss functions such as Equation (2.47), [30]. On the other hand, efficient quantum evaluation is possible only if the following two necessary conditions are met

1. A and A^2 can be decomposed in $O(\text{poly}\log(N))$ operators O_k
2. These operators are observables and have a simple tensor-product form.

Notice that we will talk here about observables as Hermitian operators, whose eigenvectors form an orthonormal basis for measurement.

Liu et al. show that A and A^2 for the 1-dimensional Poisson matrix meet both requirements, [69]. The matrix A has the following block structure

$$A_n = \begin{bmatrix} A_{n-1} & D_{n-1} \\ D_{n-1}^T & A_{n-1} \end{bmatrix}, \tag{2.49}$$

where A_n is the Poisson matrix for n qubits and

$$D_{n-1} = \begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & & \\ -1 & 0 & \dots & 0 \end{bmatrix}. \tag{2.50}$$

For example, the 1 and 2 qubit cases are

$$A_1 = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} = 2I - \sigma_+ - \sigma_-;$$

$$A_2 = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}$$
$$= I \otimes A_1 - \sigma_+ \otimes \sigma_- - \sigma_- \otimes \sigma_+,$$

where $\sigma_+ = |0\rangle\langle 1|$ and $\sigma_- = |1\rangle\langle 0|$. By applying repeated tensor products of A_{n-1} with the identity and adding the center off-diagonal terms for the n^{th} case, A_n can be written as a sum of $2m + 1$ terms.

The matrix A^2 can instead be split into the following two submatrices

$$A_n^2 = \begin{bmatrix} 5 & -4 & 1 & & & 0 \\ -4 & 6 & -4 & 1 & & \\ 1 & \ddots & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & 1 \\ & & 1 & -4 & 6 & -4 \\ 0 & & & 1 & -4 & 5 \end{bmatrix}$$
$$= \begin{bmatrix} 6 & -4 & 1 & & & 0 \\ -4 & 6 & -4 & 1 & & \\ 1 & \ddots & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & \ddots & 1 \\ & & 1 & -4 & 6 & -4 \\ 0 & & & 1 & -4 & 6 \end{bmatrix} - \begin{bmatrix} 1 & & & & & \\ & 0 & & & & \\ & & \ddots & & & \\ & & & 0 & & \\ & & & & 0 & \\ & & & & & 1 \end{bmatrix} \tag{2.51}$$
$$= B_n - C_n.$$

The matrix B_n is decomposed in the same fashion as A_n, while C_n is the sum of just two terms

$$C_n = \underbrace{\sigma_+\sigma_- \otimes \cdots \otimes \sigma_+\sigma_-}_{n} + \underbrace{\sigma_-\sigma_+ \otimes \cdots \otimes \sigma_-\sigma_+}_{n} \tag{2.52}$$

Overall, A_n^2 can be decomposed in $4n+1$ terms. Since both A and A^2 are decomposed in $O(\log(N))$ operators, the first requirement for

advantage is satisfied. This holds also for the d-dimensional case. In fact, Equation (2.36) states that the d-dimensional Poisson equation is the sum of tensor products of the identity tensor with the one-dimensional matrix. Therefore, the d-dimensional Poisson problem with Dirichlet boundary conditions can be handled with the same operators as in the $d = 1$ case and specifically with $d(2n+1)$ of them for A and $(d(2n+1))^2$ for A^2. In a similar fashion, the authors show that the Neumann, Robin, and mixed boundary conditions cases with $d = 1$ also have efficient decompositions.

To fullfill the second necessary condition for an efficient protocol, the operators A_n and B_n need to be mapped to Hermitian operators. Liu et al. achieve this by mapping them in the higher dimensional space of Bell states. For instance, in the case of a 2×2 linear system, one can build

$$\begin{aligned} O_{11} &= \begin{bmatrix} 0 & \sigma_+ \\ \sigma_+^\dagger & 0 \end{bmatrix} = \left|\varphi_{11}^+\right\rangle\!\left\langle\varphi_{11}^+\right| - \left|\varphi_{11}^-\right\rangle\!\left\langle\varphi_{11}^-\right|, \\ O_{12} &= \begin{bmatrix} 0 & \sigma_- \\ \sigma_-^\dagger & 0 \end{bmatrix} = \left|\varphi_{12}^+\right\rangle\!\left\langle\varphi_{12}^+\right| - \left|\varphi_{12}^-\right\rangle\!\left\langle\varphi_{12}^-\right|, \end{aligned} \tag{2.53}$$

where $\left|\varphi_{11}^\pm\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$, $\left|\varphi_{12}^\pm\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)$ are Bell states. By also defining $|0,1\rangle$ and $|0,i1\rangle$ as the following 1-qubit states

$$\begin{aligned} |0,1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \\ |0,i1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle), \end{aligned} \tag{2.54}$$

it is possible to evaluate the terms appearing in Equation (2.48), such as

$$\begin{aligned} \langle x(\boldsymbol{\theta})|\sigma_+|x(\boldsymbol{\theta})\rangle &= \langle 0,1|\langle x(\boldsymbol{\theta})|O_{11}|0,1\rangle|x(\boldsymbol{\theta})\rangle - i\,\langle 0,i1|\langle x(\boldsymbol{\theta})|O_{11}|0,i1\rangle|x(\boldsymbol{\theta})\rangle, \\ \langle x(\boldsymbol{\theta})|\sigma_-|x(\boldsymbol{\theta})\rangle &= \langle 0,1|\langle x(\boldsymbol{\theta})|O_{12}|0,1\rangle|x(\boldsymbol{\theta})\rangle - i\,\langle 0,i1|\langle x(\boldsymbol{\theta})|O_{12}|0,i1\rangle|x(\boldsymbol{\theta})\rangle. \end{aligned} \tag{2.55}$$

For higher matrix dimensions, measuring the expectation values of A_m and B_m requires operators similar to those in Equation (2.53), but whose eigenvalues are entangled states in more than two qubits.

Figure 2.5 shows that the measurement circuit in [69] is shallow and made by only one and two qubits gates. However, this circuit assumes full qubits connectivity, which is generally not available in current hardware.

A final remark refers to the ansatz $U(\boldsymbol{\theta})$ chosen by Liu et al. for their simulations. This is the quantum alternating operator ansatz (QAOA), [41, 42], which consists of a layered circuit, each layer having only two parameters, corresponding to the evolution times of the mixer and driver Hamiltonians. The two Hamiltonians were chosen so that their gate depth grows only linearly with the number of qubits. Furthermore,

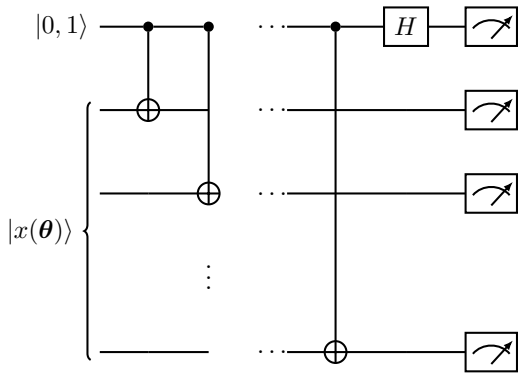

Figure 2.5.: Circuit for measuring the state $|0,1\rangle|x(\boldsymbol{\theta})\rangle$ in the Bell basis [69].

the results obtained on a quantum simulator show that the number of QAOA layers only needs to increase linearly with the number of qubits for fixed solution fidelity, resulting in a circuit that is overall suitable for NISQ hardware.

QUANTUM ANNEALING

Srivastava and Sundararaghavan demonstrated the use of a quantum annealer to solve differential equations, [70]. The motivation behind their work was that the functional minimization form of the differential problem can be written in terms of the discrete solution and solved as a combinatorial optimization problem. More precisely, the problem becomes a binary graph-coloring problem, which is NP-hard, [71].

As seen in Section 2.1.5, a quantum annealer finds the ground state of the Ising Hamiltonian, hereby reported for clarity

$$H(\boldsymbol{q}) = \sum_{i=1}^{n} H_i q_i + \sum_{(i,j)} J_{ij} q_i q_j.$$

The spin variables q_i encode the values of the discrete problem variables, while H_i and J_{ij} depend on the problem data and boundary conditions.

For instance, consider the 1D Laplace problem with Dirichlet boundary conditions

$$\begin{cases} \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} = 0, \\ u(0) = u_0, \\ u(L) = u_L \end{cases} \tag{2.56}$$

where L is the length of the domain. The associated energy potential is

$$\Pi(u) = \int_0^L \frac{1}{2}\left(\frac{du}{dx}\right)^2 dx. \tag{2.57}$$

2

One can replace $u(x)$ with the discrete solution

$$u(x) \approx \sum_{i=1}^{N} \varphi_i(x) a_i, \tag{2.58}$$

where $\varphi_i(x)$ can be taken as the linear finite element shape functions in Equation (2.44). In particular, finite element approximations are local, which fits well the fact that quantum annealers have limited local connectivity.

Considering only two elements (3 nodes) on the unit domain and substituting Equation (2.58) in (2.57) leads to the discrete functional

$$\Pi(a_0, a_1, a_2) = \mathbf{a}_1^T \mathbf{s}_1 + \mathbf{a}_2^T \mathbf{s}_2, \tag{2.59}$$

where

$$\begin{aligned}
\mathbf{a}_1 &= [a_0^2, a_1^2, a_0 a_1, a_0, a_1]^T \\
\mathbf{a}_2 &= [a_1^2, a_2^2, a_1 a_2, a_1, a_2]^T. \\
\mathbf{s}_1 &= \mathbf{s}_2 = [1, 1, -2, 0, 0]^T
\end{aligned}$$

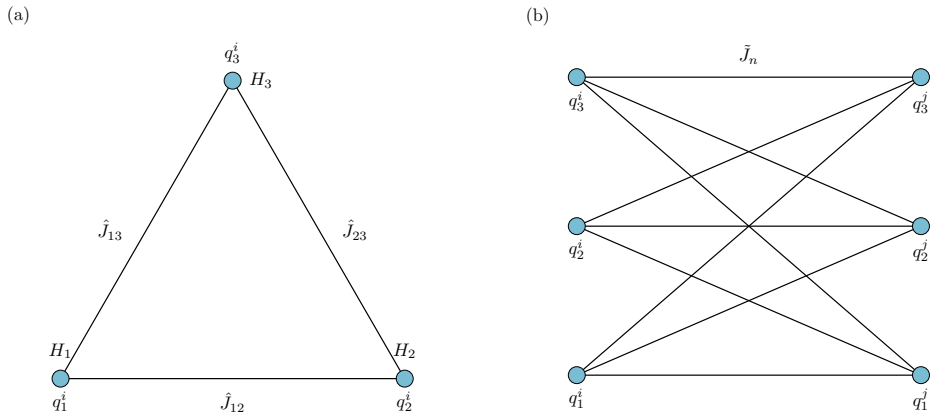

Figure 2.6.: Generic node graph (a) and element graph (b) for the one dimensional Laplace equation in [70] with the corresponding weigths of the Ising Hamiltonian.

A central idea in [70] is to associate every element with a local graph, repeat it throughout the annealer graph, and form global connections

according to the connectivity of finite elements. Every node i is associated with 3 qubits q_j^i, which in turn are assigned to 3 different values v_{i_j} that the nodal solution can assume. Since $q_j^i = \pm 1$,

$$a_i = \sum_{j=1}^{3} v_{i_j} \frac{q_i^j + 1}{2}. \tag{2.60}$$

In this way, a_i can, in principle, assume 9 possible values. However, some of these values lead to invalid solutions and need to be penalized when writing the discrete potential.

The image on the left of Figure 2.6 shows the node graph used in [70]. This has associated H_l and $\hat{J}_{lk}$ weights, which contibute to the Ising Hamiltonian. The aim of the nodal weights is to promote the boundary conditions as well as feasible values of the solution of the nodes. In case the PDE was not homogeneous, H_l and $\hat{J}_{lk}$ would also account for the right hand side.

The right image shows instead the element graph for a single element, characterized by the matrix $\tilde{J}_n$. In the case of a 1D domain, the connection is between two adjacent nodes, each characterized by 3 qubits, therefore the element weight matrix $\tilde{J}^n$ is characterized by 9 linear equations

$$\sum_{k=1}^{3}\sum_{l=1}^{3} (\tilde{J}^n)_{kl} q_k^i q_l^j = \boldsymbol{\alpha}_n (a_i, a_j)^T \mathbf{s}_n, \tag{2.61}$$

where i and j are the nodes connected by element n.

Once H_l, $\hat{J}_{lk}$ and $(\tilde{J}^n)_{kl}$ are defined for every node and element, the weights in the transverse Ising Hamiltonian are defined and the quantum annealer can search for its ground state.

However, if a higher precision is required while spanning the same range of possible values, then more than 3 qubits would be required per node. In a realistic FE model with thousands of nodes this would require a high number of qubits and a high degree of connectivity, which may be beyond hardware capacity. Therefore Srivastava and Sundararaghavan proposed the so-called 'box algorithm'. The main idea is to have the values v_{i_j} centered around a value u_i^c with distance r, thus

$$v_{i_j} = u_i^c + r(j-2) \tag{2.62}$$

In this way, the nodal values are spaced around u_i^c with radius r and for N nodes, all possible admissible values will be distributed on the surface of a N-dimensional box. The procedure consists in doing subsequent annealings until the desired precision r is met. After every annealing, u_i^c and r are allowed to vary according to the following logic

1. If $u_i^c \pm r$ for a certain i has lower potential than u_i^c, then the center is moved to that point

2. If u_i^c is the point of minimum in the box, then r is reduced.

The procedure continues until r is below a threshold defined by the user-defined precision of the solution. It can be shown that this is a convergent process [70].

The graph-coloring and box algorithm was benchmarked against two truss problems, described by the equation

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(EA\frac{\mathrm{d}u}{\mathrm{d}x}\right)+f(x)=0, \tag{2.63}$$

where $E(x)$ and $A(x)$ are respectively the distributed Young's modulus and area, while $f(x)$ is the distributed load over the truss. The first example is a truss with a discontinous cross section at half length, while the second is a tapered truss with distributed load. Both examples converge to the numerical FE solution using up to 6 elements. The authors also mention that a finer discretization would likely require a two-point version of the box technique to better exploit the connectivity of the annealer.

2.3.2. HEAT EQUATION

The heat equation is another widely applied mathematical model in solid mechanics, which can be used to predict temperature profiles and heat concentrations.

Let the spatial domain be $\Omega \subseteq \mathbb{R}^d$, and the temporal domain be $I \subseteq \mathbb{R}$. The solution $u(\mathbf{x}, t)$ satisfies the equation, in Cartesian coordinates,

$$\frac{\partial u}{\partial t}=\alpha\left(\frac{\partial^2 u}{\partial x_1^2}+\cdots+\frac{\partial^2 u}{\partial x_d^2}\right)=\alpha\nabla^2 u, \qquad \mathbf{x}=[x_1,\ldots,x_d]^\top\in\Omega, \quad t\in[0,T], \tag{2.64}$$

where α is the thermal diffusivity of the material. As usual, the differential problem is well posed once complemented with boundary and initial conditions.

For $d \leq 3$, the heat equation can be derived from Equation (2.28), when no internal reaction terms $Q(x, t)$ are present. In that case,

$$\alpha=\frac{\kappa}{\rho c}. \tag{2.65}$$

COMPARISON OF CLASSICAL AND GATE-BASED QUANTUM METHODS

A comprehensive study on quantum solvers for the heat equation was conducted by Linden et al., [37]. Specifically, the authors compared 5 classical and 5 quantum methods in terms of their theoretical runtimes

to approximate the temperature integrated over a region $S \subseteq \Omega$, up to precision ε with 99% success probability of the algorithm, i.e.

$$\left| \tilde{H} - \int_S u(\mathbf{x}, t) \mathrm{d}\mathbf{x} \right| \leq \varepsilon \tag{2.66}$$

where $\tilde{H}$ is the approximated temperature integral.

The discretization in time consists in first-order forward finite differences, while second-order centered finite differences are used for the space grid. This scheme is also called *forward time centered space* or FTCS, for short. Therefore **??** becomes the finite difference equation

$$\begin{aligned} &\frac{\tilde{u}(\mathbf{x}, t + \Delta t) - \tilde{u}(\mathbf{x}, t)}{\Delta t} \\ &= \frac{\alpha}{(\Delta x)^2} \sum_{i=1}^{d} \tilde{u}(x_1, \ldots, x_i + \Delta x_i, \ldots, x_d, t) \\ &- 2\tilde{u}(x_1, \ldots, x_i, \ldots, x_d, t) \\ &+ \tilde{u}(x_1, \ldots, x_i - \Delta x_i, \ldots, x_d, t), \end{aligned} \tag{2.67}$$

where $\Delta t = T/M$, M being the number of time intervals, while $\Delta x = L/N$, assuming equal length L for each dimension and division in N intervals. Furthermore, $\tilde{u}$ is the approximation of u due to the finite difference discretization.

By grouping the values of $\tilde{u}$ at the same time step, Equation (2.67) can be rewritten as

$$\begin{aligned} \tilde{u}(\mathbf{x}, t + \Delta t) &= \left(1 - \frac{2d\alpha\Delta t}{(\Delta x)^2}\right) \tilde{u}(\mathbf{x}, t) \\ &+ \frac{\alpha\Delta t}{(\Delta x)^2} \sum_{i=1}^{d} \tilde{u}(x_1, \ldots, x_i + \Delta x_i, \ldots, x_d, t) \\ &+ \tilde{u}(x_1, \ldots, x_i - \Delta x_i, \ldots, x_d, t). \end{aligned} \tag{2.68}$$

Also, by defining $\tilde{\mathbf{u}}_i$ as the solution at time step t_k, $k = \{1, \ldots M\}$, Equation (2.68) can be written in vector form as

$$\tilde{\mathbf{u}}_{k+1} = \mathcal{L}\tilde{\mathbf{u}}_k, \tag{2.69}$$

where $\mathcal{L}$ is the linear operator on the right-hand side of Equation (2.68).

The classical solvers studied by Linden for this problem are the following.

1. *Single linear system approach with conjugate gradient.* Equation (2.69) can be seen as a unique linear system, for the solution at

subsequent times. In fact, if $\mathbf{u} = [\mathbf{u}_1, \dots \mathbf{u}_M]^T$, $\mathbf{f} = [\mathcal{L}\mathbf{u}_0, \mathbf{0}, \dots, \mathbf{0}]^T$ and $\mathbf{u_0}$ is the initial condition in discrete space, then

$$A\mathbf{u} = \mathbf{f}, \tag{2.70}$$

where

$$A = \begin{pmatrix} I & 0 & & \\ -\mathcal{L} & I & & \\ & \ddots & \ddots & \\ & & -\mathcal{L} & I \end{pmatrix}. \tag{2.71}$$

The conjugate gradient method is assumed in [37] to solve the sparse linear system in Equation (2.70) in linear time, even though variations of this method such as the biconjugate gradient stabilized method (BiCGSTAB), [72], would be more efficient, while accounting for the asymmetry of A.

2. *Time-stepping from initial condition.* This is a matrix-vector multiplication problem. In fact, Equation (2.69) can be expanded up to $\mathbf{u_0}$ as

$$\tilde{\mathbf{u}}_\mathbf{k} = \mathcal{L}^k \mathbf{u_0}. \tag{2.72}$$

Then, the approximate solution at time $t_k = k\Delta t$ is obtained by k successive multiplications of $\mathcal{L}$ to $\mathbf{u_0}$.

3. *Time-stepping using the Fast Fourier Transform (FFT).* For the FTCS scheme, the matrix $\mathcal{L}$ in d dimensions has the expression [37]

$$\mathcal{L} = I_N^{\otimes d} + \frac{\alpha \Delta t}{(\Delta x)^2} \sum_{j=1}^{d} I_N^{\otimes (j-1)} \otimes H \otimes I_N^{\otimes (d-j)}, \tag{2.73}$$

where I_N is the identity matrix of dimension N. Also,

$$H = \begin{bmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & 1 & \ddots & \ddots & \\ & & \ddots & \ddots & \\ 1 & & & 1 & -2 \end{bmatrix}, \tag{2.74}$$

which is a circulant matrix and can therefore be diagonalized by the Discrete Fourier Transform F. Since the circulant matrices operate on different dimensions, the matrix $\mathcal{L}$ is diagonalized by the tensor product of F, i.e. $F^{\otimes d}$. Furthermore, H has eigenvalues $\lambda_j = -4 \sin^2\left(\frac{j\pi}{N}\right)$, which can be used in combination with Equation (2.73) to compute the eigenvalues of $\mathcal{L}$. Therefore,

$$\mathcal{L} = \left(F^{\otimes d}\right)^{-1} \Lambda F^{\otimes d}, \tag{2.75}$$

where Λ is the diagonal matrix with the eigenvalues of $\mathcal{L}$ on the diagonal. Therefore the time stepping equation Equation (2.72) becomes

$$\tilde{\mathbf{u}}_k = \left(F^{\otimes d}\right)^{-1} \Lambda^k F^{\otimes d} \mathbf{u}_0. \tag{2.76}$$

Numerically, this consists in doing the Fast Fourier Transform (FFT) of $\mathbf{u}_0$, multiplying the resulting vector by the k-th powers of the eigenvalues of $\mathcal{L}$ and finally performing an inverse FFT.

4. *Random walk.* Equation (2.68) can be seen in terms of stochastic quantities. In fact, introducing $s = \frac{\alpha \Delta t}{(\Delta x)^2}$, this equation can be rewritten as

$$\begin{aligned} \tilde{u}(\mathbf{x}, t+\Delta t) &= (1-2ds)\,\tilde{u}(\mathbf{x}, t) \\ &+ \sum_{i=1}^{d} s\tilde{u}(x_1, \ldots, x_i + \Delta x_i, \ldots, x_d, t) \\ &+ s\tilde{u}(x_1, \ldots, x_i - \Delta x_i, \ldots, x_d, t). \end{aligned} \tag{2.77}$$

If $s \leq 1/(2d)$, Equation (2.77) can be thought of as a stochastic process. In fact, the temperature $\tilde{u}$ at each time is determined by those at the preceding time step on the surrounding d-dimensional lattice with probabilities determined by s. In this sense the FTCS equation is a random walk, where the position is the approximate temperature $\tilde{u}$.

5. *Fast random walk.* The standard random walk samples for all m time steps, each sample requiring $O(d\log(N))$ time, for a total time of $O(Md\log(N))$. A speed-up can be achieved with respect to this standard technique. First, sample from the intial distribution in $O(d\log(N))$ time and then compute the number of steps in each dimension d and the number of positive/negative increments in every dimension, by sampling two binomial distributions in time $O(\log(M))$, [37]. The improved runtime is $O(d(\log(M)+\log(N)))$.

For their comparison, Linden *et al.* discussed how quantum subroutines can speed-up the classical numerical algorithms for the heat equation. In the case of quantum algorithms, the final solution state $\left|\tilde{\tilde{u}}\right\rangle$ approximates $|\tilde{u}\rangle$, that is the quantum state representation of $\mathbf{u}$ of the FTCS equation. Starting from $\left|\tilde{\tilde{u}}\right\rangle$, the approximate integrated temperature $\tilde{H}$ in region $S \subseteq \Omega$ can be calculated up to precision ε using numerical quadrature, i.e.

$$\left| \int_S u(\mathbf{x}, t)\mathrm{d}\mathbf{x} - (\Delta x)^d \sum_{\mathbf{x} \in G \cap S} \widetilde{\|\tilde{u}\|}_2 w(\mathbf{x}) \left\langle \mathbf{x}, t \middle| \tilde{\tilde{u}} \right\rangle \right| \leq \varepsilon, \tag{2.78}$$

where G is the d-dimensional grid in the domain Ω, $\widetilde{\|\tilde{u}\|}_2$ is the numerically estimated norm of $\tilde{u}$ and $w(\mathbf{x})$ are weights that depend on the specific numerical quadrature scheme.

The 5 quantum algorithms considered in [37] the following.

1. *Quantum linear solver.* A quantum algorithm for linear systems can be used to solve $A\mathbf{u} = \mathbf{r}$, with the matrix A defined in Equation (2.71). Linden *et al.* utilized the algorithm in [73], which is logarithmically dependent on precision. Even though this method requires the matrix to be Hermitian, it can be applied to the FTCS-discretized heat equation by solving for

$$\begin{bmatrix} 0 & A \\ A^\dagger & 0 \end{bmatrix} \begin{bmatrix} 0 \\ \mathbf{u} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}$$

2. *Fast forwarded quantum walk method.* Quantum walks are a form of random walks that can be performed with unitary operations on quantum states, [74]. The first results in quantum walks showed how these could simulate random walks quadratically faster in the limit of the number of time steps. Later, it was proved that quantum walks can still retain quadratic speed-up even at intermediate times, thanks to fast forwarding [75].

3. *Coherent diagonalization.* The operator $\mathcal{L}^k$ in Equation (2.72) can be diagonalized exponentially faster using QFT instead of FFT. Ahead of measuring, one wants to reproduce the state

$$|u_k\rangle = \mathcal{L}^k |u_0\rangle , \tag{2.79}$$

where it is assumed that $|u_0\rangle$ can be prepared [37]. If Φ is the Quantum Fourier Transform (QFT) operator, then

$$|u_k\rangle = \Phi^\dagger \Lambda^k \Phi |u_0\rangle , \tag{2.80}$$

where Λ is the diagonal matrix of the eigenvalues. Therefore $|u_k\rangle$ can be prepared following these steps

   a) Prepare $|u_0\rangle$.

   b) Apply QFT.

   c) Apply the Λ^k operator. This is not unitary in general, but it can be implemented by using an ancilla qubit, applying a rotation controlled on the $\Phi|u_0\rangle$ state, measuring and post-selecting, [37].

   d) Apply the inverse QFT.

4. *Random walk with amplitude estimation.* The random walk technique can be accelerated quadratically, thanks to amplitude estimation. In general, approximating $\int_S u(\mathbf{x})\mathrm{d}\mathbf{x}$ requires $O(1/\varepsilon^2)$

repetitions of the classical random walk due to the Chernoff bound. However, if a boolean function $f(s)$ exists such that

$$f(s) = \begin{cases} 0 & \text{if } \mathbf{x} \notin S \\ 1 & \text{if } \mathbf{x} \in S \end{cases} \tag{2.81}$$

and if f can be encoded as an oracle, then quantum amplitude estimation can estimate $\Pr(f(s) = 1)$ in $O(1/\varepsilon)$ time. This can be used, in turn, to compute $\int_S u(\mathbf{x})\mathrm{d}\mathbf{x}$.

5. *Fast random walk with amplitude estimation.* In the same way as in standard random walks, amplitude estimation can be applied to the fast random walk technique (see classical methods).

Table 2.2 shows the time complexities of all the classical and quantum methods analyzed in [37]. In general, the classical FFT diagonalization technique scores the best complexity for the one-dimensional heat equation, while fast random walks with quantum amplitude estimation have the lowest runtime for $d \leq 2$. However, both of these techniques work only for hyper-rectangular domains, for which the heat equation has an analytical solution in terms of Fourier components, [76]. Still, the amplitudes of these modes are integrals, often computed numerically. Depending on the initial condition $\mathbf{u}_0$ and its Fourier decomposition, one may need to estimate a high number of integrals, in which case the methods for rectangular regions may still be meaningful.

On the other hand, quantum algorithms can still be faster even on generic domains. In fact, except for $d = 1$ where the classical time-stepping technique has lowest runtime, standard random walks with quantum amplitude estimation are as fast ($d = 2$) or slightly faster ($d = 3$).

However, Table 2.2 also shows that no quantum exponential speed-up is possible. This happens even when some of the underlying quantum subroutines are 'exponentially faster' than their classical counterparts, as in the case of linear solvers. However, as showed by Montanaro for elliptical problems discretized by FEM, obtaining a scalar quantity requires $O(\mathrm{poly}(1/\varepsilon))$ samples of the final quantum state, [15].

Finally, Table 2.2 shows that methods using a quantum algorithms for linear system are never faster than the best classical algorithm for $d < 5$, be it for rectangular or generic domains. Thus, one should be aware of this limitation, if aiming for a speed-up to solve the heat equation in a 3- or lower-dimensional space.

QUANTUM ANNEALING

Pollachini et al. proposed using quantum annealing in a quantum-classical routine, in order to solve the heat equation, [77].

Table 2.2.: Runtime comparison of classical and quantum methods for solving the FTCS heat equation, [37]. All runtimes are expressed in terms of terms of spatial dimensions d and error ε on the estimated temperature integral. The $\tilde{O}$ notation hides polylogarithmic factors in the complexity. Adapted from [37].

	Method	Domain	$d = 1$	$d = 2$	$d = 3$	$d \geq 4$
Classical	Single linear system	General	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2.5})$	$\tilde{O}(\varepsilon^{-3})$	$\tilde{O}(\varepsilon^{-d/2-1.5})$
	Time stepping	General	$\tilde{O}(\varepsilon^{-1.5})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2.5})$	$\tilde{O}(\varepsilon^{-d/2-1})$
	Time stepping + FFT	Hypercube	$\tilde{O}(\varepsilon^{-0.5})$	$\tilde{O}(\varepsilon^{-1})$	$\tilde{O}(\varepsilon^{-1.5})$	$\tilde{O}(\varepsilon^{-d/2})$
	Random walk	General	$\tilde{O}(\varepsilon^{-3})$	$\tilde{O}(\varepsilon^{-3})$	$\tilde{O}(\varepsilon^{-3})$	$\tilde{O}(\varepsilon^{-3})$
	Fast random walk	Hypercube	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$
Quantum	Single linear system	General	$\tilde{O}(\varepsilon^{-2.5})$	$\tilde{O}(\varepsilon^{-2.5})$	$\tilde{O}(\varepsilon^{-2.75})$	$\tilde{O}(\varepsilon^{-d/4-2})$
	FFWD Quantum walk	General	$\tilde{O}(\varepsilon^{-1.75})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2.25})$	$\tilde{O}(\varepsilon^{-d/4-1.5})$
	Coherent diagonalisation	Hypercube	$\tilde{O}(\varepsilon^{-1.25})$	$\tilde{O}(\varepsilon^{-1.5})$	$\tilde{O}(\varepsilon^{-1.75})$	$\tilde{O}(\varepsilon^{-d/4-1})$
	Random walk + AE	General	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$	$\tilde{O}(\varepsilon^{-2})$
	Fast random walk + AE	Hypercube	$\tilde{O}(\varepsilon^{-1})$	$\tilde{O}(\varepsilon^{-1})$	$\tilde{O}(\varepsilon^{-1})$	$\tilde{O}(\varepsilon^{-1})$

The equation considered is

$$k\left(\frac{\partial^2 u}{\partial x_1^2}+\frac{\partial^2 u}{\partial x_2^2}\right)+f(x_1,x_2)=0, \tag{2.82}$$

which describes the steady state of the temperature distribution on a 2D domain with forcing term $f(x_1,x_2)$. Equation (2.82) is actually the Poisson equation rather than the heat equation, where $u(x_1,x_2)$ is the equilibrium temperature and $f(x_1,x_2)$ is a distributed heat flux. Dirichlet conditions were used at the boundary.

Equation (2.82) is discretized using centered finite differences in the usual way and the problem reduces to solving a linear system $A\mathbf{u}=\mathbf{f}$, where $f_i=f(x_1^{(i)},x_2^{(i)})\ \forall i\in G$, where G is the space grid.

The following quadratic Hamiltonian can be written,

$$H(\mathbf{u})=(A\mathbf{u}-\mathbf{f})^{\mathsf{T}}(A\mathbf{u}-\mathbf{f}) \tag{2.83}$$

whose ground state corresponding to the solution of the linear system. Also, u_i can be restricted to the range $[-d_i,2c_i-d_i)$ using the following mapping

$$u_i=-d_i+c_i\sum_{r=0}^{R-1}q_r^i/2^r, \tag{2.84}$$

where d_i and c_i are user-defined real numbers and q_r^i are binary digits. In this way, u_i is associated to the binary string q_r^i and the precision can be tuned by choosing d_i and c_i.

Substituting Equation (2.84) in Equation (2.83), results in an Ising Hamiltonian

$$H(\mathbf{q})=\sum_{r=0}^{R-1}\sum_{i=0}^{N-1}H_r^i q_r^i+\sum_{r,s=0}^{R-1}\sum_{i,j=0}^{N-1}J_{rs}^{ij}q_r^i q_s^j, \tag{2.85}$$

where N is the number of nodes in the grid. After annealing the system, the inverse of the mapping in Equation (2.84) allows to reconstruct the solution.

Pollachini et al. also provided a strategy to keep their algorithm hardware-feasible even for large problem dimensions. They proposed to use the iterative block Gauss-Seidel method, which consists in iteratively solving D blocks of dimension N/D instead of one N-dimensional linear system. For instance, taking $D=2$

$$\begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix}\begin{bmatrix}\mathbf{u}_1\\ \mathbf{u}_2\end{bmatrix}=\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\end{bmatrix}. \tag{2.86}$$

Equation (2.86) can be solved iteratively, by making an initial guess for

$\mathbf{u}_2$ as $\mathbf{u}_2^{(0)}$. Then, at time step $k+1$,

$$\begin{cases} A_{11}\mathbf{u}_1^{(k+1)} & = \mathbf{f}_1 - A_{12}\mathbf{u}_2^{(k)} \\ A_{22}\mathbf{u}_2^{(k+1)} & = \mathbf{f}_2 - A_{21}\mathbf{u}_1^{(k)} \end{cases}. \tag{2.87}$$

The quantum annealer takes care of solving each lower-dimensional linear systems in Equation (2.87) and Gauss-Seidel iterations are repeated until convergence.

Pollachini *et al.* ran their algorithm on both DWave 2000Q and DWave Advantage quantum annealers, [78]. The source term $f(x_1, x_2)$ was taken randomly and Equation (2.82) was discretized on a 11×11 grid, corresponding to 9 internal points and a linear system with 81 unknowns, which was at the time one of the largest linear systems solved (at least partially) on quantum hardware.

One of the issues that arises in the computations is a flattening of the error curve for increasing iterations of the block Gauss-Seidel solver. This was attributed to saturating the floating-point precision achievable by a fixed number of qubits R. Indeed the authors fixed this issue by progressively shrinking the d_i range in Equation (2.84), matching the same convergence curve as the classical Gauss-Seidel algorithm. However, increasing the number of qubits R per interval did not benefit the solution's precision, likely due to increasing hardware noise.

Despite the approach of Pollachini et al. being hardware-ready and verified, it is unclear whether it may provide an advantage. Furthermore, their method can only be applied to the steady-state problem. However, an alternative to solve the time-dependent within quantum annealing might be to semi-discretize in space and then use the algorithm for systems of linear ODEs proposed in [79].

2.3.3. WAVE EQUATION

A third classical PDE is the wave equation, which describes the propagation of a perturbation through a medium. In Cartesian coordinates, the wave equation reads

The wave equation is written, in Cartesian coordinates, as

$$\frac{\partial^2 u}{\partial t^2} = c^2 \sum_{i=1}^{d} \frac{\partial^2 u}{\partial x_i^2}, \qquad \mathbf{x} = [x_1, \dots, x_d]^\top \in \Omega, \quad t \in [0, T], \tag{2.88}$$

where c is the wave propagation speed in the medium. Additionally, two boundary and two initial conditions fully specify the problem.

If the Poisson equation describes the volumetric deformations of rods, membranes and solids, the wave equation models the isotropic propagation of waves, at constant speed, in the same elements. Assuming $d \leq 3$, the wave equation can be derived from Equation (2.24)

by assuming volumetric deformations and the absence of body forces. In that case, given k the axial or membranal or volumetric stiffness, then

$$c^2 = \frac{k}{\rho}. \tag{2.89}$$

Except for a few instances, where an analytical solution can be found through separation of variables or using the method of characteristics, the FDM and FEM are generally used to find an approximate solution of the wave equation. For instance, in the case of FDM, the Laplacian on the right hand side of Equation (2.88) is discretized with centered finite differences and then a Runge-Kutta scheme allows to find the solution at subsequent time steps, starting from the initial conditions

$$\begin{cases} u(\mathbf{x}, 0) & = u_0(\mathbf{x}) \\ \frac{\partial u}{\partial t}\Big|_{t=0} & = \dot{u}_0(\mathbf{x}) \end{cases}. \tag{2.90}$$

The exact runtimes of these classical methods depend on the order of discretization r of the Laplacian, the choice of the time-stepping technique, etc. However, their asymptotic behavior is bounded from below as $\Omega\left[T\text{poly}\,(1/\varepsilon)^d\right]$, showing the curse of dimensionality already seen for the Poisson and heat equations.

Nevertheless, replacing classical linear algebra subroutines with quantum algorithms can remove the exponential dependency on d also when solving the wave equation. This was proved by Costa et al., [17], who showed how to turn Equation (2.88) into a Schrödinger equation and solve it using Hamiltonian simulation.

For the sake of explanation, let the wave equation be one dimensional and take $c = 1$. As usual, the domain can be reduced to a grid of spacing Δx on which the Laplacian on the right hand side of Equation (2.88) can be approximated. The number of gridpoints used for the finite difference approximation determines the *order* r of the discrete Laplacian $L^{(r)}$ and the discretization error $\|\frac{1}{(\Delta x)^2}L_i^{(r)} - \nabla^2(x^{(i)})\|$, which scales as $O((\Delta x)^r)$. For instance, the standard centered difference scheme uses $r = 2$, such that

$$\frac{1}{(\Delta x)^2}L^{(2)}u(x^{(i)}) = \frac{u(x^{(i+1)}, t) - 2u(x^{(i)}, t) + u(x^{(i-1)}, t)}{(\Delta x)^2}. \tag{2.91}$$

Furthermore, if one sees the grid that discretizes Ω as a graph $G_{\Delta x}$ of $|V|$ vertices $x^{(i)}$ and $|E|$ edges $x^{(i+1)} - x^{(i)}$, the discrete Laplacian can be thought of as a matrix $L^{(r)}(G_{\Delta x})$ defined on this graph.

Keeping $r = 2$, Equation (2.88) becomes

$$\frac{\partial^2 \mathbf{u}}{\partial t^2} = \frac{1}{(\Delta x)^2}L^{(2)}\mathbf{u}, \tag{2.92}$$

where $\mathbf{u} = [u_1, u_2, \ldots, u_N]$ and N is the number of vertices.

Assume that a matrix B exists such that $BB^\dagger = L$. One can then write

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\mathbf{u}_V \\ \mathbf{u}_E\end{bmatrix} = -\frac{i}{\Delta x}\begin{bmatrix}0 & B \\ B^\dagger & 0\end{bmatrix}\begin{bmatrix}\mathbf{u}_V \\ \mathbf{u}_E\end{bmatrix}, \tag{2.93}$$

where $\mathbf{u}_V = \mathbf{u}$ and $\mathbf{u}_E$ are additional variables associated to the edges of the graph G. We notice that Equation (2.93) is equivalent to the mixed formulation of the finite element method, where momentum and position figure as the unknowns.

Deriving Equation (2.93) with respect to time, one obtains

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\begin{bmatrix}\mathbf{u}_V \\ \mathbf{u}_E\end{bmatrix} = -\frac{1}{(\Delta x)^2}\begin{bmatrix}BB^\dagger & 0 \\ 0 & B^\dagger B\end{bmatrix}\begin{bmatrix}\mathbf{u}_V \\ \mathbf{u}_E\end{bmatrix}, \tag{2.94}$$

which shows that, if $BB^\dagger = L$, then $\mathbf{u}_V$ both evolves according to the Schrödinger equation (Equation (2.93)) and it is the solution of the original wave equation.

For an order 2 Laplacian, the B matrix is the graph *signed incidence matrix*. If one assigns random orientations to the edges of graph $G_{\Delta x}$, then

$$B_{ij} = \begin{cases} \sqrt{W_{ij}} & \text{if edge } j \text{ self-loops in vertex } i \\ \sqrt{W_{ij}} & \text{if vertex } i \text{ is a source of edge } j \\ -\sqrt{W_{ij}} & \text{if vertex } i \text{ is a sink of edge } j \\ 0 & \text{otherwise} \end{cases} \tag{2.95}$$

where W_{ij} are weights assigned to the edges of the graph. For instance, in case of the second order Laplacian, the graph is unweigthed ($W_{ij} = 1\ \forall i, j$).

If the Laplacian has order $r > 2$, the graph theoretical interpretation of B is not as straightforward. Yet, [17] discusses a general algebraic procedure to determine the incidence matrices for these higher order Laplacians and provides the entries for B and $L^{(r)}$ up to order 10.

In order to solve Equation (2.93), the Hamiltonian simulation can be performed from an initial state, in order to determine $\mathbf{u}_V$ at time t. The authors employ the algorithm in [80] for sparse Hamiltonian simulation, that is optimal with respect sparsity, error and simulation time.

It is shown that Hamiltonian simulation for time $t = T$ requires a number of gates that is $\tilde{O}\left(Td^2\,(T/\varepsilon)^{1/r}\right)$ and that the initial state can be prepared in time $\tilde{O}\left((r/2+1)\,d^{5/2}l\,(T/\varepsilon)^{1/r}\right)$, where l is a characteristic domain dimension. Most importantly, these runtimes show no exponential dependency on the dimension d, even though they do not include the time required to sample the output. Still, unless the full Hilbert space needs to be sampled, the measurement step would not reintroduce the curse of dimensionality, but just a $O(1/\varepsilon)$ factor.

Thus, as seen for the Poisson and heat equations, the speed-up with respect to classical numerical solvers is exponential for variable dimensions, but at most polynomial if the dimension is fixed. However, it is interesting to that the homogeneous wave equation can be interpreted as a Schrödinger equation and solved via Hamiltonian simulation. The same authors of [17] notice that if Equation (2.88) was treated as a second-order ODE, rather than a Schrödinger equation on an extended Hilbert space, it could be solved using the algorithm of [81], but that would result in a quadratic slowdown.

The work of Costa had an important follow-up in Suau et al., [18], where the authors studied the implementation, number of gates and actual runtime of Costa's wave equation solver. As a benchmark problem, they took the simplest case of a 1-dimensional wave equation with homogeneous Dirichlet boundary conditions on the end points. However, this implementation slightly deviates from the original wave equation solver, since the authors replaced the optimal-complexity Hamiltonian simulation algorithm in [80] with the more common Lie-Trotter-Suzuki (LTS) product formula, [82, 83].

The number of gates and runtimes required for implementing the wave equation algorithm was counted according to the following gate set

$$\{U_1(\lambda), U_2(\lambda,\phi), U_3(\lambda,\phi,\theta), CNOT\}, \tag{2.96}$$

where

$$U_3(\lambda,\phi,\theta) = \begin{bmatrix} \cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right) \\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i\lambda+\phi}\cos\left(\frac{\theta}{2}\right) \end{bmatrix} \tag{2.97}$$

$$U_2(\lambda,\phi) = U_3\left(\lambda,\phi,\frac{\pi}{2}\right) \tag{2.98}$$

$$U_1(\lambda) = U_3(\lambda,0,0) \tag{2.99}$$

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.100}$$

The runtime is obtained by converting the Hamiltonian simulation circuit to the gates in Equation (2.96) and using the gate execution times provided by the manufacturer, [84]. The first interesting point is that the circuit in [18] represents one of the few instances of a quantum PDE algorithm specified in terms of gates that can be efficiently compiled into hardware.

The gate counts of Suau's algorithm match the asymptotic (big-O) behavior of the wave equation solver for variable error, simulation time and number of gridpoints. Most interesting however, are the constants hidden in the big-O scaling, that vary between10^5 and 10^8. This results

in extremely high gate counts even just for solving the simplest possible instance of the wave equation. For example, given a moderately interesting grid size of 10^6 nodes, the quantum wave equation solver requires 10^{17} gates. The total runtime estimated for such a problem size is almost 1000 calendar years, [18]. In terms of number of qubits, the solver requires roughly 70 logical qubits, each consisting of 1000 to 10000 physical qubits, thus vastly overshooting the NISQ hardware specifications.

2.4. QUANTUM ALGORITHMS FOR NONLINEAR PDES

Solvers for nonlinear problems require high computational resources. The non linearities in the PDEs may be due material properties, such as hysteresis, plasticity or damage, but also arise in case of large displacements or in the presence of contact. Whatever the cause, classical numerical methods are generally iterative and require to solve large linear system of equations in possibly many iterations.

Quantum algorithms for nonlinear PDEs are scarce up to present date, and no work focuses specifically on structural mechanics. However, techniques to solve generic (or quasi-generic) nonlinear PDEs were proposed, [85, 86]. Both approaches consist in variationally training a parametrized circuit and on using a hybrid stratregy, whereby the quantum computer estimates the cost function terms and the classical one implements the optimization update. Nevertheless, the two methodologies significantly differ in their approaches to encoding the nonlinearities and computing the cost function.

The quantum circuit in [85] is represented in Figure 2.7. The encoding of nonlinear terms happens via a *quantum nonlinear processing unit* (QNPU), which is a circuit meant to compute nonlinear functions of polynomial form that appear in the cost function. These can be written as

$$u^{(1)*} \prod_{j=1}^{r} O_j \, u^{(j)}, \tag{2.101}$$

where the terms $u^{(i)}$ are copies of the solution function and $u^{(1)*}$ represents the complex conjugate of $u^{(1)}$. The terms O_j are different linear operators that are applied on different copies.

By normalizing $u^{(j)}$ as $\left|u^{(j)}\right\rangle$, the solution can be prepared by a parametrized ansatz, i.e. $\left|u^{(j)}\right\rangle = U(\boldsymbol{\theta})|0\rangle$. The ansatz at a given optimization step can be used as many times as the number of solution copies required by Equation (2.101) and the QNPU circuit applies the operators O_j and performs point-by-point multiplication depending on the specific nonlinear terms in the PDE.

The introduction of repeated input and the QNPU increases the depth of the quantum circuit with respect to VQAs for linear problems. The

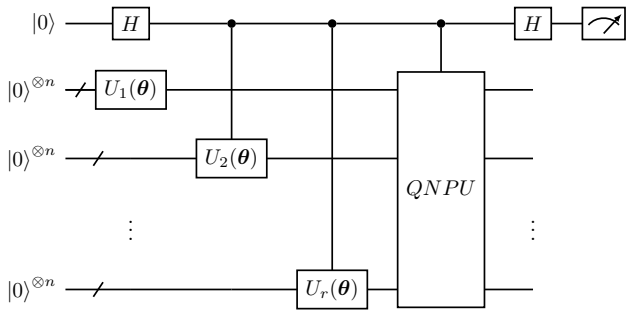

Figure 2.7.: Scheme of the variational circuit in [85]. The quantum nonlinear processing unit (QNPU) takes as input r copies of the solution vector, generated by the ansatz $U(\boldsymbol{\theta})$. Then, the QNPU applies the operators O_j in Equation (2.101) as quantum operators. The circuit's output comes from the measurements of the ancilla qubit and corresponds to the required cost function term.

authors solve this problem by encoding the quantum ansatz and the operators O_j as matrix product states (MPS) of the bond dimension χ, resulting in an overall depth of $O(\text{poly}(\chi, n))$. In this way, the number of ansatz parameters is also polynomial in n and χ [85], preventing any curse of dimensionality.

The integration of multiple inputs with a QNPU provides an effective approach for replicating nonlinearities. Nonetheless, the implementation of the QNPU block is inherently contingent upon the specific problem and may prove challenging for complex nonlinear expressions.

Another work leveraged the ideas of *quantum feature mapping* and exact differentiation of quantum circuits [87, 88] to variationally solve generic nonlinear differential problems [86], i.e.

$$F\left[\left\{\frac{\mathrm{d}^m u_n}{\mathrm{d}x^m}\right\}_{m,n}, \{u_n(x)\}_n, x\right] = 0, \tag{2.102}$$

The concept of feature map originates from the machine learning literature and consists of embedding the data into the model as parameters. In terms of quantum circuits, this means that x can be mapped to a 2^n-dimensional space with a unitary operator $U_\xi(x)$ parametrized through a nonlinear function $\xi(x)$. A straightforward instance of quantum feature map is

$$U_\xi(x) = \bigotimes_{i=1}^{n} R_{Y,i}(\xi(x)), \tag{2.103}$$

where $R_{Y,i}(\phi) = e^{-i\frac{\phi}{2}Y_i}$ and Y_i is the Pauli Y gate applied to qubit i. A common choice is $\xi(x) = \alpha\arcsin(x)$, which means that x will be encoded in quantum amplitudes that are polynomials up to order 2^n in $\left\{1, x, \sqrt{1-x^2}\right\}$, [86].

An ansatz is U_θ used to confine the search of the PDE solution to a feasible subspace. Overall, the parametrized quantum state whose amplitudes embed the tentative solution is

$$\left|u_{\xi,\theta}(x)\right\rangle = U_\xi(x)\, U_\theta \left|0\right\rangle. \tag{2.104}$$

In order to map between $\left|u_{\xi,\theta}(x)\right\rangle$ and $u(x)$, one needs also to specify an observable $\hat{C}$, such that

$$u(x) = \left\langle u_{\xi,\theta}(x)\middle|\hat{C}\middle|u_{\xi,\theta}(x)\right\rangle. \tag{2.105}$$

Once the parameters $\boldsymbol{\theta}$ have been optimized, the approximate solution at any point x can be reconstruced by simply measuring the expectation value of $\hat{C}$ under the state $\left|u_{\xi,\theta}(x)\right\rangle$. This has the obvious advantage of not having to sample the entire 2^n-dimensional Hilbert space, as it is necessary with quantum algorithms based on amplitude encoding.

Training U_θ requires to minimize a loss function $L_\theta = L_\theta\left(\frac{\mathrm{d}^m u}{\mathrm{d}x^m}, u, x\right)$ with respect to the parameters $\boldsymbol{\theta}$. Since the feature map is parametrized on x, the analytical derivative $\frac{\mathrm{d}^m u}{\mathrm{d}x^m}$ can be calculated by applying parameter shifting m times [43, 89]. For $m = 1$

$$\frac{\mathrm{d}u}{\mathrm{d}x} = \frac{1}{2}\sum_j \left(\left\langle u^+_{\xi,j,\theta}\middle|\hat{C}\middle|u^+_{\xi,j,\theta}\right\rangle - \left\langle u^-_{\xi,j,\theta}\middle|\hat{C}\middle|u^-_{\xi,j,\theta}\right\rangle\right), \tag{2.106}$$

where '+' and '-' symbolically represent the positive/negative x-shift and the sum is among all the parametrized gates composing the feature map. Suitable choices for L_θ can be the residual in Equation (2.102) or the mean squared difference with respect to the exact solution, if this is available.

2.5. DISCUSSION

The previous sections reviewed the literature of partial differential equations pertinent to structural mechanics. The analysis was divided into linear and nonlinear PDEs, which is a standard classification for differential problems. The first group includes the works related to Poisson, heat and wave equations, while the second one deals with the methods to solve general nonlinear problems.

Linear problems can be solved using all different quantum paradigms, i.e. full gate-based, hybrid quantum computing and quantum annealing. In terms of the full-quantum gate-based primitives, such as quantum

linear solvers, quantum Hamiltonian simulation etc, the Poisson, heat and wave equations can be solved with these quantum algorithms and inherit their complexities. However, the different character of the equations and the specific discretization determine which quantum routines are applicable and the extent of the advantage. For instance, quantum linear solvers are applicable to every linear PDE, both stationary and time-dependent, if the latter are written as a single linear system spanning multiple time steps. Nevertheless, the Poisson equation (on rectangular domains) with periodic boundary conditions is a favourite candidate for QLSAs, since the finite difference approximation of the Laplacian results in a circulant matrix that is diagonalized by the QFT, [57, 59, 61, 66]. This allows to do Hamiltonian simulation in the QLSA solving the Poisson equation exponentially faster than with non-circulant matrices.

On the other hand, heat and wave equations benefit more from different quantum subroutines. One hint to this is the evolutionary character of both equations, which means that linear system dimensions scale multiplicatively with respect to time grid size. Also, the time dependency seems to suggest the approach of semi-discretizing in space and then solving systems of ODEs with some Hamiltonian simulation algorithm. Indeed, this approach is ideal for the wave equation where Hamiltonian simulation solves the related graph problem in the higher dimensional space, [17]. On the other hand, the heat equation does not benefit as much from quantum ODE solvers and the useful analysis of [37] proves that minimum runtimes are achieved when accelerating a classical method (classical random walks) with amplitude amplification.

Of course, all previous considerations hold for quantum subroutines that require error-corrected hardware. Near-term quantum techniques for linear PDEs exist as quantum annealing and variational quantum algorithms, even though the efforts in this sense are in their infancy. For PDEs in structural mechanics, only two quantum annealing algorithms have been proposed, for elliptical FE problems [70] and for the stationary heat equation [77]. A first gap in this branch of literature are quantum annealing algorithms for evolutionary problems, such as heat and wave equations.

Also VQAs have just recently been applied to linear PDEs. The general literature on variational quantum computing is vast, but their use in PDEs has been limited to generic nonlinear problems, [85, 86]. In particular, there is still a lack of works for specific PDEs, even though such specialization is critical. For instance, [69] showed the relevance of the Poisson matrix in the context of VQAs. In fact, the discrete Poisson matrix can be decomposed in a poly-logarithmic number of observables, which is a necessary condition for advantage of variational PDE solvers.

The matter of choosing a quantum primitiv for nonlinear problems is not straightforward because all quantum operations are ultimately

linear. The ways forward seem ultimately to linearize the equation and apply existing quantum techniques or solve the PDE variationally. Although there is research on linearization of nonlinear ODEs and use of QLSAs at each step [90], all works on nonlinear PDEs so far have relied only on variational quantum computing. As seems likely, this paradigm together with quantum annealing will likely be the only quantum alternative viable in the near term to solve PDEs, linear and non-linear alike. However, there is currently a literature gap on fault-tolerant quantum computing for nonlinear PDEs.

A final remark concerns the extent of the overlap between quantum PDE and structural mechanics literature. To the best of our knowledge, only a single work has been published on the subject, [70]. In fact, the vast majority of literature either focuses on textbook PDEs (Poisson, heat and wave) on hypercubic domains and standard boundary conditions or provides a framework for generic nonlinear problems, that ultimately leaves the choice of critical hyperparameters to the user, [85, 86]. Of course, both sides of the spectrum project onto structural mechanics, but quantum computation still has to be tested against more complex problems in the field. It should be noted that other disciplines that are not traditional targets of quantum avantage, such as fluid mechanics, [91] and finance, [92], have explored several quantum algorithms to solve problems in their domain. The hope is that the structural mechanics community will also increase efforts in this direction.

REFERENCES

[1] G. Tosti Balducci, B. Chen, M. Möller, M. Gerritsma, and R. De Breuker. “Review and perspectives in quantum computing for partial differential equations in structural mechanics”. In: *Front. Mech. Eng.* 8 (Sept. 2022), p. 914241.

[2] A. W. Harrow, A. Hassidim, and S. Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Phys. Rev. Lett.* 103.15 (Oct. 2009), p. 150502.

[3] A. Mezzacapo, M. Sanz, L. Lamata, I. L. Egusquiza, S. Succi, and E. Solano. “Quantum Simulator for Transport Phenomena in Fluid Flows”. In: *Scientific Reports* 5.1 (Aug. 2015).

[4] R. Steijl and G. N. Barakos. “Parallel evaluation of quantum algorithms for computational fluid dynamics”. In: *Computers & Fluids* 173 (Sept. 2018), pp. 22–28.

[5] B. N. Todorova and R. Steijl. “Quantum algorithm for the collisionless Boltzmann equation”. In: *Journal of Computational Physics* 409 (May 2020), p. 109347.

[6] F. Gaitan. “Finding flows of a Navier–Stokes fluid through quantum computing”. In: *npj Quantum Information* 6.1 (July 2020).

[7] F. Gaitan. “Finding Solutions of the Navier-Stokes Equations through Quantum Computing—Recent Progress, a Generalization, and Next Steps Forward”. In: *Advanced Quantum Technologies* (Sept. 2021), p. 2100055.

[8] L. Budinski. “Quantum algorithm for the advection-diffusion equation simulated with the lattice Boltzmann method”. In: *Quantum Information Processing* 20.2 (Feb. 2021).

[9] B. Ljubomir. “Quantum algorithm for the Navier–Stokes equations by using the streamfunction-vorticity formulation and the lattice Boltzmann method”. In: *Int. J. Quantum Inform.* 20.02 (Feb. 2022), p. 2150039.

[10] F. Fontanela, A. Jacquier, and M. Oumgari. *A Quantum algorithm for linear PDEs arising in Finance*. 2021. arXiv: 1912.02753 [q-fin.CP].

[11] N. Heim, A. Ghosh, O. Kyriienko, and V. E. Elfving. *Quantum Model-Discovery*. 2021. arXiv: 2111.06376 [quant-ph].

[12] P. Mocz and A. Szasz. “Toward cosmological simulations of dark matter on quantum computers”. In: *Astrophys. J.* 910.1 (Mar. 2021), p. 29.

[13] A. Pesah. “Quantum Algorithms for Solving Partial Differential Equations”. In: *unpublished* (2020).

[14] M. A. Nielsen and I. L. Chuang. *Quantum computation and quantum information*. 10th anniversary ed. Cambridge ; New York: Cambridge University Press, 2010. isbn: 9781107002173.

[15] A. Montanaro and S. Pallister. “Quantum algorithms and the finite element method”. In: *Physical Review A* 93.3 (Mar. 2016).

[16] R. P. Feynman. “Simulating physics with computers”. In: *Int. J. Theor. Phys.* 21.6 (June 1982), pp. 467–488.

[17] P. C. S. Costa, S. Jordan, and A. Ostrander. “Quantum algorithm for simulating the wave equation”. In: *Physical Review A* 99.1 (Jan. 2019).

[18] A. Suau, G. Staffelbach, and H. Calandra. “Practical Quantum Computing: Solving the Wave Equation Using a Quantum Approach”. In: *ACM Transactions on Quantum Computing* 2.1 (Apr. 2021), pp. 1–35.

[19] H. F. Trotter. “On the product of semi-groups of operators”. In: *Proceedings of the American Mathematical Society* 10.4 (Aug. 1959), pp. 545–551.

[20] D. Aharonov and A. Ta-Shma. “Adiabatic Quantum State Generation and Statistical Zero Knowledge”. In: *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*. STOC ’03. Association for Computing Machinery, 2003, pp. 20–29.

[21] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma. “Exponential Improvement in Precision for Simulating Sparse Hamiltonians”. In: *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*. Association for Computing Machinery, 2014, pp. 283–292.

[22] D. W. Berry, A. M. Childs, R. Cleve, R. Kothari, and R. D. Somma. “Simulating Hamiltonian Dynamics with a Truncated Taylor Series”. In: *Phys. Rev. Lett.* 114 (9 Mar. 2015), p. 090502.

[23] G. H. Low and I. L. Chuang. “Hamiltonian Simulation by Qubitization”. In: *Quantum* 3 (July 2019), p. 163.

[24] J. R. Shewchuk. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep. USA, 1994.

[25] A. Ambainis. “Variable time amplitude amplification and quantum algorithms for linear algebra problems”. In: *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*. Vol. 14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2012, pp. 636–647.

[26] B. D. Clader, B. C. Jacobs, and C. R. Sprouse. “Preconditioned Quantum Linear System Algorithm”. In: *Phys. Rev. Lett.* 110.25 (June 2013), p. 250504.

[27] A. M. Childs, R. Kothari, and R. D. Somma. “Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision”. In: *SIAM Journal on Computing* 46.6 (Jan. 2017), pp. 1920–1950.

[28] Y. Subaşı, R. D. Somma, and D. Orsucci. “Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing”. In: *Physical Review Letters* 122.6 (Feb. 2019).

[29] D. An and L. Lin. “Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm”. In: *ACM Transactions on Quantum Computing* 3.2 (Mar. 2022), pp. 1–28.

[30] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles. “Variational Quantum Linear Solver”. In: *Quantum* 7 (Nov. 2023), p. 1188.

[31] X. Xu, J. Sun, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan. “Variational algorithms for linear algebra”. In: *Science Bulletin* 66.21 (Nov. 2021), pp. 2181–2188.

[32] H.-Y. Huang, K. Bharti, and P. Rebentrost. “Near-term quantum algorithms for linear systems of equations with regression loss functions”. In: *New J. Phys.* 23.11 (Nov. 2021), p. 113021.

[33] H. Patil, Y. Wang, and P. S. Krstić. “Variational quantum linear solver with a dynamic ansatz”. In: *Physical Review A* 105.1 (Jan. 2022).

[34] L. K. Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC ’96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219.

[35] L. K. Grover. “Quantum Mechanics Helps in Searching for a Needle in a Haystack”. In: *Phys. Rev. Lett.* 79.2 (July 1997), pp. 325–328.

[36] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. *Quantum amplitude amplification and estimation*. 2002.

[37] N. Linden, A. Montanaro, and C. Shao. “Quantum vs. Classical Algorithms for Solving the Heat Equation”. In: *Commun. Math. Phys.* 395.2 (Oct. 2022), pp. 601–641.

[38] M. Cerezo *et al.* “Variational quantum algorithms”. In: *Nature Reviews Physics* 3.9 (Aug. 2021), pp. 625–644.

[39] A. Kandala *et al.* “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets”. In: *Nature* 549.7671 (Sept. 2017), pp. 242–246.

[40] A. G. Taube and R. J. Bartlett. “New perspectives on unitary coupled-cluster theory”. In: *International Journal of Quantum Chemistry* 106.15 (2006), pp. 3393–3401.

[41] E. Farhi, J. Goldstone, and S. Gutmann. *A Quantum Approximate Optimization Algorithm*. 2014. arXiv: 1411.4028 [quant-ph].

[42] S. Hadfield, Z. Wang, B. O’Gorman, E. Rieffel, D. Venturelli, and R. Biswas. “From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz”. In: *Algorithms* 12.2 (Feb. 2019), p. 34.

[43] A. Mari, T. R. Bromley, and N. Killoran. “Estimating the gradient and higher-order derivatives on quantum hardware”. In: *Physical Review A* 103.1 (Jan. 2021).

[44] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. “Barren plateaus in quantum neural network training landscapes”. In: *Nature Communications* 9.1 (Nov. 2018).

[45] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles. “Cost function dependent barren plateaus in shallow parametrized quantum circuits”. In: *Nature Communications* 12.1 (Mar. 2021).

[46] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015.

[47] J. Stokes, J. Izaac, N. Killoran, and G. Carleo. “Quantum Natural Gradient”. In: *Quantum* 4 (May 2020), p. 269.

[48] J. Spall. “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation”. In: *IEEE Transactions on Automatic Control* 37.3 (1992), pp. 332–341.

[49] C. C. McGeoch. “Theory versus practice in annealing-based quantum computing”. In: *Theoretical Computer Science* 816 (May 2020), pp. 169–183.

[50] T. Kato. “On the Adiabatic Theorem of Quantum Mechanics”. In: *Journal of the Physical Society of Japan* 5.6 (Nov. 1950), pp. 435–439.

[51] D. Aharonov, V. Jones, and Z. Landau. “A polynomial quantum algorithm for approximating the Jones polynomial”. In: *Algorithmica* 55.3 (2009), pp. 395–421.

[52] H. Buhrman, R. Cleve, J. Watrous, and R. de Wolf. “Quantum Fingerprinting”. In: *Phys. Rev. Lett.* 87 (16 Sept. 2001), p. 167902.

[53] G. H. Golub and C. F. van Loan. *Matrix Computations*. Fourth. JHU Press, 2013. isbn: 1421407949 9781421407944.

[54] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods*. Springer New York, 2008. isbn: 9780387759340.

[55] M. Benzi and M. Tûma. “A comparative study of sparse approximate inverse preconditioners”. In: *Applied Numerical Mathematics* 30.2-3 (June 1999), pp. 305–340.

[56] S. Aaronson. “Read the fine print”. In: *Nature Physics* 11 (Apr. 2015), pp. 291–293.

[57] Y. Cao, A. Papageorgiou, I. Petras, J. Traub, and S. Kais. “Quantum algorithm and circuit design solving the Poisson equation”. In: *New Journal of Physics* 15.1 (2013), p. 013021.

[58] F. D. Benedetto. “Preconditioning of Block Toeplitz Matrices by Sine Transforms”. In: *SIAM Journal on Scientific Computing* 18.2 (Mar. 1997), pp. 499–515.

[59] S. Wang, Z. Wang, W. Li, L. Fan, Z. Wei, and Y. Gu. “Quantum fast Poisson solver: the algorithm and complete and modular circuit design”. In: *Quantum Information Processing* 19.6 (Apr. 2020).

[60] S. Wang *et al.* “Quantum circuits design for evaluating transcendental functions based on a function-value binary expansion method”. In: *Quantum Information Processing* 19.10 (Sept. 2020).

[61] A. M. Childs, J.-P. Liu, and A. Ostrander. “High-precision quantum algorithms for partial differential equations”. In: *Quantum* 5 (Nov. 2021), p. 574.

[62] V. Giovannetti, S. Lloyd, and L. Maccone. “Quantum Random Access Memory”. In: *Physical Review Letters* 100.16 (Apr. 2008).

[63] V. Giovannetti, S. Lloyd, and L. Maccone. “Architectures for a quantum random access memory”. In: *Physical Review A* 78.5 (Nov. 2008).

[64] C. Ciliberto *et al.* “Quantum machine learning: a classical perspective”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 474.2209 (Jan. 2018), p. 20170551.

[65] Z.-Y. Chen, Q. Zhou, C. Xue, X. Yang, G.-C. Guo, and G.-P. Guo. “64-qubit quantum circuit simulation”. In: *Science Bulletin* 63.15 (Aug. 2018), pp. 964–971.

[66] S. Wang *et al.* “A quantum Poisson solver implementable on NISQ devices”. In: *arXiv:2005.00256* (2020).

[67] P. A. Damianou. “A Beautiful Sine Formula”. In: *The American Mathematical Monthly* 121.2 (2014), p. 120.

[68] Y. Huang and W. F. McColl. “Analytical inversion of general tridiagonal matrices”. In: *Journal of Physics A: Mathematical and General* 30.22 (Nov. 1997), pp. 7919–7933.

[69] H.-L. Liu *et al.* “Variational quantum algorithm for the Poisson equation”. In: *Phys. Rev. A* 104.2 (Aug. 2021), p. 022418.

[70] S. Srivastava and V. Sundararaghavan. “Box algorithm for the solution of differential equations on a quantum annealer”. In: *Phys. Rev. A* 99.5 (May 2019), p. 052355.

[71] “Introduction to Graph Coloring”. In: *Graph Coloring Problems*. John Wiley & Sons, Ltd, 1994. Chap. 1, pp. 1–30. isbn: 9781118032497.

[72] H. A. van der Vorst. “Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (1992), pp. 631–644. eprint: https://doi.org/10.1137/0913035.

[73] S. Chakraborty, A. Gilyén, and S. Jeffery. “The Power of Block-Encoded Matrix Powers: Improved Regression Techniques via Faster Hamiltonian Simulation”. In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by C. Baier, I. Chatzigiannakis, P. Flocchini, and S. Leonardi. Vol. 132. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 33:1–33:14.

[74] A. Ambainis. *Quantum walks and their algorithmic applications*. 2004. arXiv: quant-ph/0403120 [quant-ph].

[75] S. Apers and A. Sarlette. *Quantum Fast-Forwarding: Markov Chains and Graph Property Testing*. 2019. arXiv: 1804.02321 [quant-ph].

[76] R. Haberman. *Applied partial differential equations with fourier series and boundary value problems*. Harlow: Pearson, 2014. isbn: 129203985X.

[77] G. G. Pollachini, J. P. L. C. Salazar, C. B. D. Góes, T. O. Maciel, and E. I. Duzzioni. “Hybrid classical-quantum approach to solve the heat equation using quantum annealers”. In: *Physical Review A* 104.3 (Sept. 2021).

[78] *D-Wave System Documentation*. [Online; accessed 28. Jun. 2024]. June 2022.

[79] B. Zanger, C. B. Mendl, M. Schulz, and M. Schreiber. “Quantum Algorithms for Solving Ordinary Differential Equations via Classical Integration Methods”. In: *Quantum* 5 (July 2021), p. 502.

[80] D. W. Berry, A. M. Childs, and R. Kothari. “Hamiltonian Simulation with Nearly Optimal Dependence on all Parameters”. In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science* (Oct. 2015).

[81] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang. “Quantum Algorithm for Linear Differential Equations with Exponentially Improved Dependence on Precision”. In: *Commun. Math. Phys.* 356.3 (Dec. 2017), pp. 1057–1081.

[82] S. Lloyd. “Universal Quantum Simulators”. In: *Science* 273.5278 (1996), pp. 1073–1078.

[83] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders. “Efficient Quantum Algorithms for Simulating Sparse Hamiltonians”. In: *Communications in Mathematical Physics* 270.2 (Dec. 2006), pp. 359–371.

[84] *Melbourne hardware operation execution time*. Online; accessed 2022-01-28. 2019.

[85] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch. “Variational quantum algorithms for nonlinear problems”. In: *Physical Review A* 101.1 (Jan. 2020).

[86] O. Kyriienko, A. E. Paine, and V. E. Elfving. “Solving nonlinear differential equations with differentiable quantum circuits”. In: *Physical Review A* 103.5 (May 2021).

[87] K. Mitarai, M. Negoro, M. Kitagawa, and K. Fujii. “Quantum circuit learning”. In: *Physical Review A* 98.3 (Sept. 2018).

[88] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin. “General parameter-shift rules for quantum gradients”. In: *Quantum* 6 (Mar. 2022), p. 677. eprint: 2107.12390v3.

[89] G. E. Crooks. *Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition*. 2019. arXiv: 1905.13311 [quant-ph].

[90] J.-P. Liu, H. Ø. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, and A. M. Childs. “Efficient quantum algorithm for dissipative nonlinear differential equations”. In: *Proceedings of the National Academy of Sciences* 118.35 (Aug. 2021).

[91] S. Succi, W. Itani, K. Sreenivasan, and R. Steijl. “Quantum computing for fluids: Where do we stand?” In: *Europhysics Letters* 144.1 (Oct. 2023), p. 10001.

[92] D. Herman *et al.* “Quantum computing for finance”. In: *Nature Reviews Physics* 5.8 (July 2023), pp. 450–465.

3

A VARIATIONAL QUANTUM LINEAR SOLVER FOR THE 1D POISSON PROBLEM

Different hybrid quantum-classical algorithms have been proposed to solve linear systems of equations on near-term quantum devices. These methods map the linear system problem to a Hamiltonian evolution one, where the ground state is proportional to the linear system's solution. The free parameters of a tryout or ansatz *quantum circuit are updated classically, while state preparation and measurement occur on a quantum device. To retain a potential advantage, variational linear solvers assume that the coefficient matrix can be decomposed into a logarithmic number of efficiently measurable operators with respect to the linear system size. However, the standard decomposition in Pauli operators for generic matrices shows a linear scaling. In this work, we show how to solve a specific instance, namely the Poisson 1D matrix with Dirichlet boundary conditions with a variational quantum linear solver (VQLS) and we find a nontrivial decomposition based on both Pauli strings and multi-controlled gates. The resulting number of operators is roughly half that obtained by a full Pauli decomposition for large linear system dimensions. We further discuss the trade-off between the number of terms and the near-term implementability of the quantum circuits that the proposed decomposition requires. Finally, we present the first simulated and real-hardware results obtained by solving the Poisson 1D problem with VQLS also using our novel decomposition.*

Current quantum computers are a tool in search for applications. Even though several quantum algorithms have deep practical implications and offer up to exponential speed-ups, their promises are so far purely theoretical. Good examples of this are Shor's algorithm for period finding [1], and Grover's for database search, [2],.

The reason why no implementations at useful scales exist for this algorithms is that quantum hardware is still far from its maturity. One of today's challenges is to find combinations of problems and algorithms that can run on NISQ devices with limited error. One way to proceed is to choose problems that are classically hard but easy for quantum computers, regardless of the practical interest of the problem itself, [3]. A second case way is to assess the performance of algorithms for early quantum hardware in solving a relevant problem. Offloading part of the computation to a classical processor, variational quantum algorithms (VQA), [4], have been identified as a one of the prominent near-term ways to solve problems in different disciplines, such as quantum chemistry, [5–7] and quantum machine learning, [8].

Certain variants of variational quantum routines were developed specifically for solving linear systems of equations, [9–13], which are encountered in many different scientific and engineering disciplines. Unlike the seminal work of Harrow, Hassiddim and Lloyd on solving linear systems on an ideal quantum device, [14], these algorithms do not promise an exponential speed-up, but, due to the lower hardware requirements, they were used to solve linear systems with up to 1024 equations on a quantum device, [11].

However, previous works focused on problems where the coefficient matrix is composed of unitaries acting on few qubits, that can be straightforwardly encoded as quantum circuit. The open question is whether variational linear solvers can efficiently solve problems that are closer to real-world applications. In this work, we contribute to the answer by solving a basic example in numerical analysis, which is the 1D Poisson problem with Dirichlet boundary conditions, with the algorithm known as Variational Quantum Linear Solver (VQLS), [11].

More specifically, we discuss the decomposition the 1D Poisson coefficient matrix, resulting from a uniform grid, finite-difference discetization of the 1D Possion problem. This decomposition is crucial to variationally simulate the Hamiltonian of the VQLS problem, whose ground state corresponds to the linear system's solution. We show that the standard decomposition in Pauli operators translates in 2^n terms, where n is the number of qubits, and we propose an alternative decomposition in $2^{n-1} + n$ terms, which almost halves the number of terms for large dimensions. We compare the Pauli and novel decompositions in terms of hardware fesibility and show results both on simulator and on quantum hardware.

The rest of the chapter is structured as follows. Section 3.1 briefly

explains the main concepts behind quantum linear systems and VQLS. Following, Section 3.2 presents the problem and the standard and proposed decompositions for the coefficient matrix, and compare the two in terms of number of operators and ease of implementation on hardware. Section 3.3 shows the results of solving the Poisson 1D linear system with VQLS on both simulator and IBM quantum hardware. Finally, Section 3.4 discusses the findings and draws the final conclusions.

3.1. BACKGROUND

Let A be a square complex $N \times N$ matrix and $\boldsymbol{b}$ a complex N-dimensional vector. The aim is to find the complex vector $\boldsymbol{u}$ such that

$$A\boldsymbol{u} = \boldsymbol{b}. \tag{3.1}$$

In general, the form in Equation (3.1) is not suitable for quantum computation, because $\boldsymbol{u}$ and $\boldsymbol{b}$ may not have unit norm and thus are not valid quantum states. Nevertheless, one can solve the analogous problem

$$A\,|u\rangle = |b\rangle\,, \tag{3.2}$$

where $|u\rangle = \boldsymbol{u}/\|u\|$ and $|b\rangle = \boldsymbol{b}/\|b\|$. Eq. Equation (3.2) is known as the *Quantum Linear System Problem* (QLSP).

Following [11], one requirement to solve Equation (3.2) variationally is to express A as a sum of unitary operators. This means that the matrix A must be decomposed as

$$A = \sum_{l=1}^{L} c_l A_l, \tag{3.3}$$

where $c_l \in \mathbb{C}$, A_l are unitary $N \times N$ complex matrices and L is the total number of operators. Furthermore, the state $|b\rangle$ needs to be prepared efficiently.

As explained later in the paper, the problem of reducing the number of unitary components of A is crucial for the success and the efficiency of the variational quantum linear solver.

3.1.1. THE VQLS ALGORITHM

The VQLS algorithm, [11], finds an approximate solution $\left|x_f\right\rangle$ to the generic QLSP. It works by minimizing a cost function that represents the distance between the states $|\psi\rangle = A\,|u\rangle$ and $|b\rangle$. The optimization parameters $\boldsymbol{\theta}$ determine the tentative solution $|u\rangle$ at every iteration as

$$|u(\boldsymbol{\theta})\rangle = V(\boldsymbol{\theta})\,|0\rangle\,, \tag{3.4}$$

where $V(\boldsymbol{\theta})$ is a parametrized quantum circuit known as *ansatz*.

VQLS is a hybrid optimization algorithm, where the quantum part prepares the state $|u(\boldsymbol{\theta})\rangle$ and evaluates the cost function, while the classical routine updates $\boldsymbol{\theta}$ following an optimization logic, for example, gradient descent.

COST FUNCTION

Solving Equation (3.2) can be seen as finding the ground state of the Hamiltonian

$$H_G = A^\dagger(1 - |b\rangle\langle b|)A. \tag{3.5}$$

Therefore, the cost function of VQLS can be seen as the expectation of H_G with respect to trial state $|u(\boldsymbol{\theta})\rangle$,

$$C_G = \langle u(\boldsymbol{\theta})|H_G|u(\boldsymbol{\theta})\rangle\,. \tag{3.6}$$

In this form, however, C_G has a minimum not only if $A|u\rangle \propto |b\rangle$, but also if the norm of $\boldsymbol{\psi} = A|u\rangle$ vanishes. Therefore, one can simply normalize $\boldsymbol{\psi}$ as $|\psi\rangle = \boldsymbol{\psi}/\|\psi\|$ to find

$$C_G = 1 - |\langle b|\psi\rangle|^2, \tag{3.7}$$

As $A|u\rangle \to |b\rangle$, $\langle b|\psi\rangle \to 1$ and C_G vanishes, making Equation (3.7) a suitable cost function for the QLSP.

What motivates the use of a quantum computer in VQLS is the hardness of classically evaluating a cost function such as Equation (3.7). In particular, evaluating Equation (3.7) to within precision $\delta = 1/\text{poly}(n)$ is a DQC1-hard[1] problem, [11], and therefore impossible to solve efficiently in classical logic, [16, 17].

In order to evaluate C_G, we need to introduce the expansion in Equation (3.3). Remembering that $\boldsymbol{\psi} = A|u\rangle$, we can rewrite

$$|\langle b|\psi\rangle|^2 = \frac{\langle u|A^\dagger|b\rangle\langle b|A|u\rangle}{\langle u|A^\dagger A|u\rangle}.$$

Then, expanding A, we get

$$C_G = 1 - \frac{\sum_{l=1}^{L}\sum_{l'=1}^{L}\langle u|A_l^\dagger|b\rangle\langle b|A_{l'}|u\rangle}{\sum_{l=1}^{L}\sum_{l'=1}^{L}\langle u|A_l^\dagger A_{l'}|u\rangle}, \tag{3.8}$$

where we notice that a total of $2L^2$ expectation values must be computed to evaluate C_G, where L is the total number of terms in the decomposition of A (Equation (3.3)).

[1] DQC1 is the Deterministic Quantum Computing with 1 Clean Qubit (DQC1) complexity class, which consists of probems that can be solved in the 1-clean-qubit model of computation in polynomial time [15]

ANSATZ

In principle any $|u(\boldsymbol{\theta})\rangle$ can be prepared by any parametrized quantum circuit. We give here just a high level overview of the possible choices.

In literature, different ansatz types exist, but a main distinction is between the hardware efficient ansatze (HEA) [18], and the Quantum Alternating Operator Ansatze (QAOA), [19]. The first ones use only a few gates that are native or easily implementable on the quantum device, in order to introduce as little noise as possible. However, these ansatze are problem-agnostic, because they do not relate to the input of the problem. On the other hand, the QAOA is inspired by the adiabatic principle and therefore needs a final Hamiltonian that is aware of the original problem.

Indicatively, QAOAs can be good choices if some knowledge about the solution of the problem is available, because this can be encoded in the final Hamiltonian. However, the register size and circuit depth required to prepare the ansatz might be unavailable on NISQ hardware, at least for problem sizes that are not classically simulatable.

OPTIMIZATION ROUTINE

In principle any classical optimizer can minimize Equation (3.8) and there are no general rules to favor a specific choice. Furthermore, VQAs can also leverage gradient-based optimizers, because the different flavours of the parameter shift rule, [20], allow to compute derivatives of any order.

Here, we do not discuss extensively the benefits of different optimizers applied to VQAs. Good resources exist that overview the field, [21]. However, two considerations are worth mentioning, that can steer the optimizer's choice. Firstly, VQA loss landscapes are in general highly nonconvex, [22], and simple gradient descent easily falls into local minima. Gradient-based optmizers with momentum, such as Adam, [23], or gradient-free methods are more advisable for that reason. Also, gradients of the VQA loss can vanish for certain choices of ansatz, [24], loss function, [25], and due to noise [26]. Quantum-information based optimizers such as the quantum natural gradient descent method, [27], can help navigating these barren plateaus of the loss landscape, at the cost of more quantum resources.

3.2. 1D POISSON PROBLEM AND MATRIX DECOMPOSITIONS

The 1D Poisson problem with Dirichlet boundary conditions is defined as

$$\begin{aligned} -\frac{\mathrm{d}^2 u}{\mathrm{d}x^2} &= f(x), \\ u(x_0) &= u_0, \\ u(x_L) &= u_L, \end{aligned} \tag{3.9}$$

where $u(x)$ is the solution function, $f(x)$ is a load vector and u_0, u_L are respectively the known values of the solution at the boundary.

The problem can be reduced to a linear system of the form $A\boldsymbol{u} = \boldsymbol{b}$ by discretizing the domain with a uniform-grid centered finite difference approximation

$$\frac{\mathrm{d}^2 u}{\mathrm{d}x^2} \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{2(\Delta x)^2}, \tag{3.10}$$

where Δx is the resolution of the space discretization, and by sampling $f(x)$ at the internal points.

The resulting matrix is tridiagonal with constant coefficients $\alpha, \beta \in \mathbb{R}$ on the main and off-diagonals, that is

$$A = \begin{bmatrix} \alpha & \beta & & & & & & \\ \beta & \ddots & \ddots & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & \beta & \alpha & \beta & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \beta & \\ & & & & & \beta & \alpha & \end{bmatrix}. \tag{3.11}$$

The exact value of α and β does not matter for the present discussion.

3.2.1. PAULI DECOMPOSITION

Generally speaking, every $N \times N$ Hermitian matrix can be seen as a linear combination of N^2 *Pauli strings*, [28], where each Pauli string acts on $n = \log_2(N)$ qubits. Therefore, the Poisson matrix in Equation (3.11) can be expressed as

$$A = \sum_{i=1}^{N^2} c_i P_i, \tag{3.12}$$

$$P_i = \sigma_0^i \otimes \sigma_1^i \otimes \cdots \otimes \sigma_{n-1}^i, \tag{3.13}$$

where $\sigma_j^i \in \{X, Y, Z, I\}$ are Pauli operators.

Since all Pauli strings are unitary, Equation (3.12) can be used to compute the terms of the VQLS cost function.

We show now that tridiagonal matrices do not require all the terms in the Pauli alphabet of generic Hermitian matrices.

Theorem 1. *Tridiagonal matrices such as Equation (3.11) can be decomposed in 2^n Pauli strings, rather than 2^{2n}.*

Heuristic Proof. Without loss of generality, we assume that $\alpha = 2$, $\beta = 1$. The $n = 1$ case is trivial, since

$$A_1 = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} = 2I_0 - X_0. \tag{3.14}$$

For $n = 2$, we start by taking the tensor product of I with the terms for $n = 1$, which gives

$$\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 \\ 0 & 0 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} = I_1(2I_0 - X_0),$$

where the Pauli operator I_i acts on qubit i and tensor products are implied between operators acting on different qubits.

To add the missing off-diagonal elements, we first notice that

$$X_1X_0 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

We can correct the 2 spurious elements by including

$$Y_1Y_0 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{bmatrix},$$

so that, for $n = 2$ we get

$$A_2 = I_1(2I_0 - X_0) - \frac{1}{2}X_1X_0 + \frac{1}{2}Y_1Y_0. \tag{3.15}$$

This mechanism holds more generally for $n \geq 2$. Each time, the tensor product of I with the expansion for $n-1$ adds 2^{n-1} terms. Furthermore, the term $X_{n-1} \dots X_0$ fills the main antidiagonal of the matrix and finally, the spurious $2^{n-1} - 1$ nonzero couples of terms on the main antidiagonal are removed by as many Pauli strings that are either $Y_{n-1} \dots Y_0$ or combinations of X and Y gates on different qubits. □

3.2.2. DECOMPOSITION WITH MULTI-QUBIT GATES

The set of Pauli strings is not the only option to decompose Equation (3.11). One class of quantum gates that cannot be replicated by Pauli strings are multi-qubit gates, like SWAP or controlled gates. We use these multi-qubit gates to formulate a decomposition that, for large dimensions, almost halves the number of operators with respect to using solely the Pauli strings basis.

In particular, we make use of the following unitary operator, which we name *center-switch*. This gate 'switches' the most significant qubit to a state and all other qubits to the opposite state, that is

$$\begin{aligned} \mathrm{CS}\,|01\ldots1\rangle &= \mathrm{CS}\,|10\ldots0\rangle \\ \mathrm{CS}\,|10\ldots0\rangle &= \mathrm{CS}\,|01\ldots1\rangle \end{aligned} \tag{3.16}$$

. In matrix form,

$$\mathrm{CS} = \begin{bmatrix} 1 & & & & & & & \\ & 1 & & & & & & \\ & & \ddots & & & & & \\ & & & 1 & & & & \\ & & & & 1 & & & \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{bmatrix}. \tag{3.17}$$

For $n = 2$ qubits, the center-switch corresponds to the familiar SWAP gate. The matrix form shows that the CS gate contains the off-diagonal elements without any extra spurious ones on the antidiagonal, opposite than the $X_{n-1}\ldots X_0$ operator in the Pauli decomposition.

Table 3.1 compares both terms and number of terms in the Pauli and CS-based decompositions up to $n = 4$. Evidently, compensating for the extra diagonal elements introduced by the center-switch requires less additional terms than correcting the full antidiagonal of the Poisson matrix.

COMPILATION OF THE CENTER-SWITCH GATE

We can show, by using basic permutation theory, that the CS gate corresponds to a sequence of multicontrolled gates. First, we recall the concept of *transposition*, that is simply a permutation between two variables, i.e.

$$(a,\, b): \quad a \to b \quad \text{and} \quad b \to a. \tag{3.18}$$

By introducing a third variable p, $(a,\, b)$ can be expanded as a product of three transpositions, thus

$$(a,\, b) = (a,\, p)\,(p,\, b)\,(a,\, p). \tag{3.19}$$

Table 3.1.: Comparison between Pauli and center-switch decompositions of the Poisson 1D matrix. The notation $\mathrm{CS}_{(i-j)}$ indicates a center-switch gate acting on the register between qubit i and j.

n	N	Pauli decomposition	Decomposition with center-switch
2	4	I_1I_0, I_1X_0, X_1X_0, Y_1Y_0	$\mathrm{SWAP}_{(1-0)}$, I_1I_0, Z_1Z_0, I_1X_0
3	8	$I_2I_1I_0$, $I_2I_1X_0$, $I_2X_1X_0$, $I_2Y_1Y_0$, $X_2X_1X_0$, $X_2Y_1Y_0$, $Y_2X_1Y_0$, $Y_2Y_1X_0$	$I_2\mathrm{SWAP}_{(1-0)}$, $I_2I_1I_0$, $I_2Z_1Z_0$, $I_2I_1X_0$ $\mathrm{CS}_{(2-0)}$, $Z_2I_1Z_0$, $Z_2Z_1I_0$
4	16	$I_3I_2I_1I_0$, $I_3I_2I_1X_0$, $I_3I_2X_1X_0$, $I_3I_2Y_1Y_0$, $I_3X_2X_1X_0$, $I_3X_2Y_1Y_0$, $I_3Y_2X_1Y_0$, $I_3Y_2Y_1X_0$ $X_3X_2X_1X_0$, $X_3X_2Y_1Y_0$, $X_3Y_2X_1Y_0$, $X_3Y_2Y_1X_0$, $Y_3X_2X_1Y_0$, $Y_3X_2Y_1X_0$, $Y_3Y_2X_1X_0$, $Y_3Y_2Y_1Y_0$	$I_3I_2I_1I_0$, $I_3I_2Z_1Z_0$, $I_3I_2I_1X_0$, $I_3I_2\mathrm{SWAP}_{(1-0)}$, $I_3\mathrm{CS}_{(2-0)}$, $I_3Z_2I_1Z_0$, $I_3Z_2Z_1I_0$, $\mathrm{CS}_{(3-0)}$, $Z_3I_2I_1Z_0$, $Z_3I_2Z_1I_0$, $Z_3Z_2I_1I_0$, $Z_3Z_2Z_1Z_0$
n_{terms}		2^n	$2^{n-1}+n$

In particular, considering transpositions between bitsrings, we see that

$$\mathrm{CS} = (011, 100), \tag{3.20}$$

where bitstrings are written in little-endian notation. Furthermore, the transposition in Equation (3.20) can be expanded twice by using Equation (3.19) and two intermediate bitstrings, e.g., $p_1 = 001$ and $p_2 = 000$, such that

$$\begin{aligned}\mathrm{CS} &= (011,\ 100) = (011,\ 001)(001,\ 100)(011,\ 001) \\ &= (011,\ 001)(001,\ 000)(000,\ 100)(001,\ 000)(011,\ 001).\end{aligned} \tag{3.21}$$

The gates corresponding to the transposition in the last line of Equation (3.21) are Toffoli gates, which have two control qubits and one target. For instance, the $(011,\ 001)$ term corresponds to a Toffoli gate whose control qubits are the most and least significant ones and whose target qubit is the middle one. The middle qubit is flipped when the first control is in state 0 and the second one is in state 1. Figure 3.1 shows the circuit implementing Equation (3.21).

When $n > 3$, the procedure to compile CS is the same, but the gates implementing the single bitflip permutations are multicontrolled gates, which can be further compiled in single and 2 qubits gates, following known recipes, [29].

We notice therefore that the CS gate, once compiled, has a high cost in terms of hardware resouces. In fact, every CS ultimately results in a large number of noisy two qubit gates, and it requires almost full register connectivity. Thus, even though the CS-based decomposition saves almost half the number of operators for large number of qubits, it is less ameanable than the Pauli decomposition for implementation on NISQ devices.

3.3. RESULTS

We used the VQLS algorithm with the CS-based decomposition of the tridiagonal matrix to solve the following Poisson 1D problem

$$\begin{aligned}-\frac{\mathrm{d}^2 u}{\mathrm{d}x^2} &= \text{const}, \\ u(x_0) &= 0, \\ u(x_L) &= 0,\end{aligned} \tag{3.22}$$

where the value of the constant load was chosen such that the normalized discrete load vector can be prepared with a Hadamard transform, i.e.

$$|b\rangle = H^{\otimes n}\left|0^n\right\rangle = \frac{1}{\sqrt{2^n}}[1 \ldots 1]^{\mathrm{T}}. \tag{3.23}$$

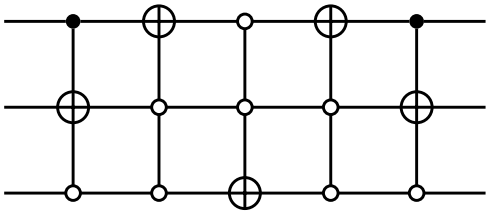

Figure 3.1.: Center-switch gate corresponding to the (011, 100) permutation. Each one of the gates is a Toffoli gate, which has two control and one target qubits. An empty cirlce at the control qubit means that the qubit must be in state 0 to activate the gate and a full circle means that it must be in state 1.

The trial state vector was prepared with an ansatz consisting of a single $R_Y(\theta)$ gate per qubit, therefore

$$|u(\boldsymbol{\theta})\rangle = R_Y(\theta_0) \otimes \cdots \otimes R_Y(\theta_{n-1}) \left|0^n\right\rangle. \tag{3.24}$$

The cost function terms in the numerator and denominator of Equation (3.8) were evaluated using the Hadamard Test circuit, [30]. Also, quantum circuit composition and evaluation was done using the Qiskit library, [31]. Two different backends were tested. The IBM QASM simulator, where circuits were sampled 1×10^4 times, and IBM quantum device `ibmq_athens`, [32], which only allowed for 8192 shots. Finally, cost function minimization was done using the COBYLA algorithm, [33].

Figure 3.2 shows the VQLS cost history and fidelity of the solution for $n = 1$ and $n = 2$, that is, for a 2×2 and 4×4 linear system respectively. The fidelity is calculated as

$$F = \left|\langle u(\boldsymbol{\theta}_f)|u_0\rangle\right|, \tag{3.25}$$

where $\left|u(\boldsymbol{\theta}_f)\right\rangle$ is the final VQLS solution and $|u_0\rangle$ is the solution of the QLSP calculated via Gaussian elimination.

The $n = 1$ case, shows that the run with the simulator converges to near-zero cost after 10 iterations. In the hardware runs, 5 in this case, the cost also decreases and assesses close to a constant value after a similar number of iterations. However, the final value is negative, in disagreement with the fact that, analytically, Equation (3.7) ranges between 0 and 1. This discrepancy is probably related to the limited amount of shots that could be performed on the quantum device, which is a source of systematic error. Despite the offset in the final cost, the hardware runs converge and achieve high fidelity for this simple

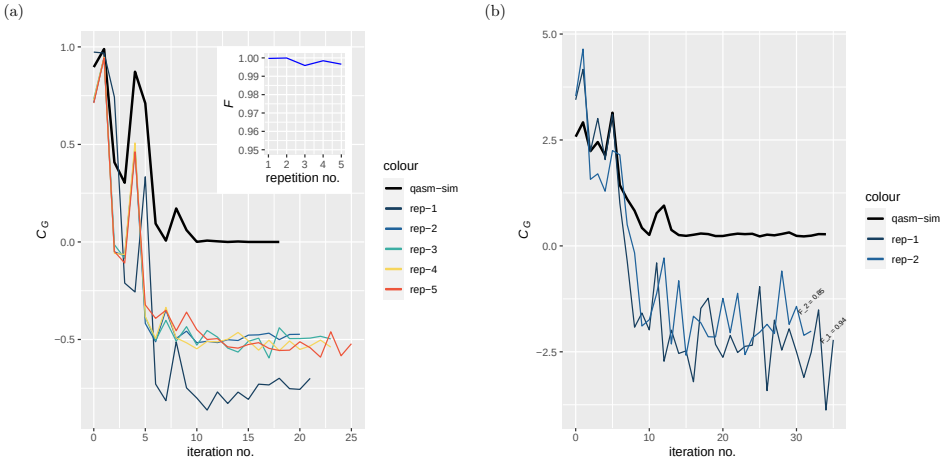

Figure 3.2.: Cost history and final solution fidelity obtained by solving the discretized Equation (3.22) with the VQLS on both simulator and quantum hardware. (a) $n = 1$ (2×2 linear system), 5 VQLS repetitions on quantum hardware (b) $n = 2$ (4×4 linear system), 2 VQLS repetitions on quantum hardware. The fidelity at the end of the quantum runs for $n = 2$ are annotated next to the corresponding curves.

example, as demonstrated by the inset of Figure 3.2 (a). Similar noise resilience has been observed before in variational quantum algorithms, [34].

The $n = 2$ case is more interesting, since it is the first system size where the Pauli and CS decompositions differ. We used this time the nonnormalized version of the cost function, which is Equation (3.8) without the term in the denominator. Even though there could be local minima where $A|u\rangle \approx 0$ in the non-normalized cost landscape, this loss is more regular than the normalized one, because it avoids the fractional term. Also in the $n = 2$ case, both simulator and hardware runs converge, although we still notice the final cost value discrepancy due to sampling noise. We also observe that, in the simulated case, the cost does not vanish completely, likely because the ansatz is not expressive enough to represent the exact solution.

3.4. DISCUSSION

In this chapter, we showed how to solve the 1D Poisson problem with the variational quantum linear solver taken from [11]. Furthermore, we proposed a way to decompose the matrix of the discrete problem, using what we call the 'center-switch' gate. We showed that, with these gates, roughly half of the terms are needed to decompose the Poisson 1D matrix, compared to a decomposition in the Pauli basis. Finally, we demonstrated the method for 1 and 2 qubits both on simulator and hardware and approached in both cases the target solution with high fidelity.

It is important to clarify that the scalings we demonstrated hold only for a uniform discretization of Equation (3.9). Nonuniform meshes would require, in general, one or more correcting terms for each of the $3N-2$ entries of the 1D Poisson matrix, making both Pauli and CS-based decompositions polynomially worse than in the uniform mesh case. Furthermore, nonuniform meshes spread-out the range of the eigenvalues of the Poisson matrix, and thus increase its conditioning number. This is expected to translate into roughly a linear reduction in the convergence rate of the VQLS solver [11].

The methods and results shown here can be improved and extended in multiple ways. First, we remark that a parallel work, published shortly after this research took place, demonstrated an efficient decomposition technique, using only $2\log(n)+1$ terms, [35], that can also be extended to the general Poisson equation on N dimensions. From the perspective of efficiency and generalization, this new method is more favorable than the one we propose. Still, there are other important aspects to be considered, such as trainability. The method in [35] decomposes the Hamiltonian H_G into observables, rather than A in unitary terms. However, having a decomposition for A is more flexible, because it

happens prior to the choice of a Hamiltonian, and therefore prior to the definition of the cost function. In particular, if the observables are global, i.e. the full qubit register is measured, the cost gradients vanish even with a shallow ansatz, [25]. One could experiment both with our decomposition and a local cost function and with the global Hamiltonian decomposition in [35] and monitor the emergence of barren plateaus.

Further research should also consider other linear systems. In particular, space-time discretization of PDEs always result in sparse linear systems with a lot of structure for which there could also be efficient decompositions. Maybe, some decomposition rules could also be generalized not just to PDEs but to numerical schemes, for instance Runge-Kutta methods of a given order.

Finally, more attention should be dedicated to the other VQA components of the algorithm. One improvement would be to introduce differentiability, with gradient-based oprimizers and paramer shift rules, [20]. Another one would be to explore different ansatze and figure out their tradeoff between expressivity and trainability, [24].

REFERENCES

[1] P. W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509.

[2] L. K. Grover. “A fast quantum mechanical algorithm for database search”. In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC ’96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 212–219.

[3] F. Arute *et al.* “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574 (Oct. 2019), pp. 505–510.

[4] J. R. McClean, J. Romero, R. Babbush, and A. Aspuru-Guzik. “The theory of variational hybrid quantum-classical algorithms”. In: *New J. Phys.* 18.2 (Feb. 2016), p. 023023.

[5] A. Peruzzo *et al.* “A variational eigenvalue solver on a photonic quantum processor”. In: *Nat. Commun.* 5.4213 (July 2014), pp. 1–7.

[6] P. J. J. O’Malley *et al.* “Scalable Quantum Simulation of Molecular Energies”. In: *Phys. Rev. X* 6.3 (July 2016), p. 031007.

[7] T. Jones, S. Endo, S. McArdle, X. Yuan, and S. C. Benjamin. “Variational quantum algorithms for discovering Hamiltonian spectra”. In: *Phys. Rev. A* 99.6 (June 2019), p. 062304.

[8] M. Schuld and F. Petruccione. *Machine Learning with Quantum Computers*. Springer International Publishing, 2021. isbn: 9783030830984.

[9] H.-Y. Huang, K. Bharti, and P. Rebentrost. “Near-term quantum algorithms for linear systems of equations with regression loss functions”. In: *New J. Phys.* 23.11 (Nov. 2021), p. 113021.

[10] X. Xu, J. Sun, S. Endo, Y. Li, S. C. Benjamin, and X. Yuan. “Variational algorithms for linear algebra”. In: *Science Bulletin* 66.21 (Nov. 2021), pp. 2181–2188.

[11] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles. “Variational Quantum Linear Solver”. In: *Quantum* 7 (Nov. 2023), p. 1188.

[12] D. An and L. Lin. “Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm”. In: *ACM Transactions on Quantum Computing* 3.2 (Mar. 2022), pp. 1–28.

[13] B. Wu, M. Ray, L. Zhao, X. Sun, and P. Rebentrost. “Quantum-classical algorithms for skewed linear systems with an optimized Hadamard test”. In: *Phys. Rev. A* 103.4 (Apr. 2021), p. 042422.

[14] A. W. Harrow, A. Hassidim, and S. Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Phys. Rev. Lett.* 103.15 (Oct. 2009), p. 150502.

[15] E. Knill and R. Laflamme. “Power of One Bit of Quantum Information”. In: *Phys. Rev. Lett.* 81.25 (Dec. 1998), pp. 5672–5675.

[16] T. Morimae. “Hardness of classically sampling the one-clean-qubit model with constant total variation distance error”. In: *Phys. Rev. A* 96.4 (Oct. 2017), p. 040302.

[17] K. Fujii, H. Kobayashi, T. Morimae, H. Nishimura, S. Tamate, and S. Tani. “Impossibility of Classically Simulating One-Clean-Qubit Model with Multiplicative Error”. In: *Physical Review Letters* 120.20 (May 2018).

[18] A. Kandala *et al.* “Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets”. In: *Nature* 549.7671 (Sept. 2017), pp. 242–246.

[19] S. Hadfield, Z. Wang, B. O’Gorman, E. Rieffel, D. Venturelli, and R. Biswas. “From the Quantum Approximate Optimization Algorithm to a Quantum Alternating Operator Ansatz”. In: *Algorithms* 12.2 (Feb. 2019), p. 34.

[20] D. Wierichs, J. Izaac, C. Wang, and C. Y.-Y. Lin. “General parameter-shift rules for quantum gradients”. In: *Quantum* 6 (Mar. 2022), p. 677. eprint: 2107.12390v3.

[21] M. Cerezo *et al.* “Variational quantum algorithms”. In: *Nature Reviews Physics* 3.9 (Aug. 2021), pp. 625–644.

[22] P. Huembeli and A. Dauphin. “Characterizing the loss landscape of variational quantum circuits”. In: *Quantum Science and Technology* 6.2 (Feb. 2021), p. 025011.

[23] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015.

[24] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. “Barren plateaus in quantum neural network training landscapes”. In: *Nature Communications* 9.1 (Nov. 2018).

[25] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles. “Cost function dependent barren plateaus in shallow parametrized quantum circuits”. In: *Nature Communications* 12.1 (Mar. 2021).

[26] S. Wang *et al.* “Noise-induced barren plateaus in variational quantum algorithms”. In: *Nat. Commun.* 12.6961 (Nov. 2021), pp. 1–11.

[27] J. Stokes, J. Izaac, N. Killoran, and G. Carleo. “Quantum Natural Gradient”. In: *Quantum* 4 (May 2020), p. 269.

[28] M. A. Nielsen and I. L. Chuang. *Quantum computation and quantum information*. 10th anniversary ed. Cambridge ; New York: Cambridge University Press, 2010. isbn: 9781107002173.

[29] A. Barenco *et al.* “Elementary gates for quantum computation”. In: *Physical Review A* 52.5 (Nov. 1995), pp. 3457–3467.

[30] D. Aharonov, V. Jones, and Z. Landau. “A polynomial quantum algorithm for approximating the Jones polynomial”. In: *Algorithmica* 55.3 (2009), pp. 395–421.

[31] A. Javadi-Abhari *et al. Quantum computing with Qiskit*. 2024. doi: 10.48550/arXiv.2405.08810. arXiv: 2405.08810 [quant-ph].

[32] *IBM Quantum*. 2024.

[33] M. J. D. Powell. “A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation”. In: *Advances in Optimization and Numerical Analysis*. Dordrecht, The Netherlands: Springer, 1994, pp. 51–67.

[34] E. Fontana, N. Fitzpatrick, D. M. Ramo, R. Duncan, and I. Rungger. “Evaluating the noise resilience of variational quantum algorithms”. In: *Phys. Rev. A* 104.2 (Aug. 2021), p. 022403.

[35] H.-L. Liu *et al.* “Variational quantum algorithm for the Poisson equation”. In: *Phys. Rev. A* 104.2 (Aug. 2021), p. 022418.

4

COMPOSITE FAILURE PREDICTION VIA CLASSICAL AND QUANTUM KERNEL CLASSIFICATION

Modeling open hole failure of composites is a complex task, consisting in a highly nonlinear response with interacting failure modes. Numerical modeling of this phenomenon has traditionally been based on the finite element method, but requires to tradeoff between high fidelity and computational cost. To mitigate this shortcoming, recent work has leveraged machine learning to predict the strength of open hole composite specimens. Here, we also propose using data-based models but to tackle open hole composite failure from a classification point of view. More specifically, we show how to train surrogate models to learn the ultimate failure envelope of an open hole composite plate under in-plane loading. To achieve this, we solve the classification problem via support vector machine (SVM) and test different classifiers by changing the SVM kernel function. The flexibility of kernel-based SVM also allows us to integrate the recently developed quantum kernels in our algorithm and compare them with the standard radial basis function (RBF) kernel. Finally, thanks to kernel-target alignment optimization, we tune the free parameters of all kernels to best separate safe and failure-inducing loading states. The results show classification accuracies higher than 90% for RBF, especially after alignment, followed closely by the quantum kernel classifiers.

Parts of this chapter are available at *arXiv:2405.02903* [1] and were sent for publication to the *Computers and Structures* journal.

4.1. INTRODUCTION

Modern aviation industry makes wide use of composite materials, thanks to their lightweight and favorable mechanical properties. Frequently, aeronautical structural elements are often not textbook flat composite panels, but tailored components with complex mechanical responses. For instance, composite panels often show cutouts in order to allow fastening or lightening the structure or even for allowing the passage of wiring or cables. However, the presence of holes in a composite plate induces stress concentrations that can initiate damage which can propagate into intricate failure mechanisms involving different modes.

Models for open hole composite failure have developed in different directions. On the one hand, semi-empirical models were proposed to predict the allowables of these structures, such as ultimate stength, and their statistical distribution with respect to hole geometry, loading conditions, stacking sequence, ply thickness, etc. Early attempts required experimental properties from testing both the unnotched and notched laminate, [2], while later models removed the need of directly testing the open hole laminate, [3, 4] or just required the ply properties, [5]. Despite being fast to evaluate and suitable for preliminary design, semi-empirical models can make large errors when extensive delaminations propagate from the notch, as it happens with ply-scaled laminates.

Finite Element (FE) simulations allow for improved modeling of open hole laminates failure. Open hole tension (OHT) has been extensively studied numerically both for capturing the in-plane, [6] and thickness size effects, [7–9] on the ultimate strength and for reproducing the different failure modes and their interactions, [10, 11] with increasing detail. Furthermore, FE simulations managed to quite accurately predict open-hole compression (OHC), even though still struggling to predict the precise kink band formation, [12–14]. However, the accuracy offered by FE models generally comes at the price of high computational costs, possibly making them unfeasible when many design iterations are required.

Therefore, there is a practical need for computationally efficient yet accurate models that can simulate open hole composite laminates. A possibility is offerred by machine learning surrogates, which have been employed in composite design and optimization, [15–17], constitutive law modeling and multiscale analyses (see [18] for a comprehensive review) and damage characterisation, [19, 20]. Concerning open-hole composite failure, Furtado *et al.* proposed a methodology to define allowables using four different machine learning models, [21]. Their methodology was applied to open-hole tensile strength prediction for different dimensions, layups and material properties. While their methods are demonstrated on data generated analytically, [5], the authors suggest using high fidelity finite element analyses for training,

potentially providing accurate data-based models.

Similarly, in this work we propose a machine learning surrogate for open hole composites, which is accurate and efficient in inference. Differently from [21] however, the approach we suggest is not to have a fast predictor of allowables, but a classifier for ultimate failure of open hole composite laminates. More precisely, our trained model takes a loading state as input, such as the far field homogenized plane strain components and returns a binary valuable (±1) as output, depending on whether the load applied is lower or higher than the notched laminate strength. In this sense, the surrogate acts as a data-based generalized failure criterion which predicts at the structural component level, rather than at the material level.

This Chapter also aims at comparing classical and quantum computation for a classification problem in composite mechanics. To do this, we train the machine learning surrogate using *kernel*-based support vector machines (SVMs), [22], where the kernel function can be computed both in classical and quantum logic. As it will be clear in the next sections, quantum computation offers a way to encode information into exponentially large Hilbert spaces and to define an inner product in this spaces, effectively generating a kernel. This allows to explore the generalization potential of quantum machine learning, while leaving the SVM optimization to well-established classical quadratic optimization algorithms.

The rest of this Chapter is structured as follows. Section 4.2 describes the machine learning problem, by defining the input, the data sampling strategy and the labeling criterion. Section 4.3 briefly introduces the SVM dual problem, the Radial Basis Functions (RBF) kernels and the quantum kernels. More details about these methods are available in the appendices. Finally, Section 4.4 presents the classification results for all kernels and Section 4.5 outlines conclusions and future work.

All data and code used in this chapter are made publicly available (see [23], [24] respectively).

4.2. MACHINE LEARNING PROBLEM

Our method was applied to predict failure of an open hole composite specimen similar in geometry and material properties to the one experimentally tested in [25]. The specimen was modeled and meshed with the Abaqus finite element code, [26], and it was loaded with different combinations of axial and shear strains and constrained with periodic boundary conditions. All the details of the specimen properties and of the finite element analyses are left to Appendix A.

The input of our surrogate models are homogenized far field strains $\boldsymbol{\varepsilon} = [\varepsilon_{11}, \varepsilon_{22}, \gamma_{12}]^{\mathsf{T}}$, which derive from enforcing periodic boundary conditions on opposite faces of the plate. The displacements of the

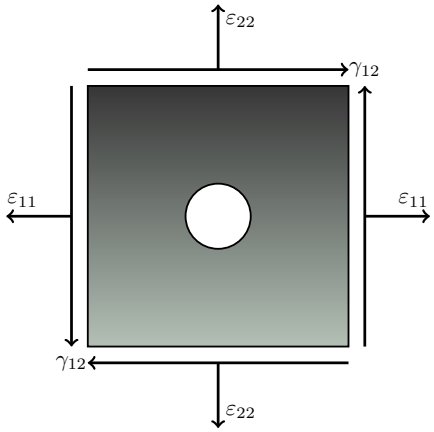

Figure 4.1.: Geometry of the open hole composite plate and homogenous strain loads components.

left/right and top/bottom faces respectively can be linked through some reference degrees of freedom

$$\begin{aligned} U_1 &= u_1^R - u_1^L \\ U_2 &= u_2^T - u_2^B \\ U_3 &= u_2^R - u_2^L \\ U_4 &= u_1^T - u_1^B, \end{aligned} \tag{4.1}$$

where directions 1 and 2 are the horizontal and vertical directions in Figure 4.1. The homogenized strains are then obtained as

$$\begin{aligned} \varepsilon_{11} &= \frac{U_1}{D_1} \\ \varepsilon_{22} &= \frac{U_2}{D_2} \\ \gamma_{12} &= \frac{U_3}{D_1} + \frac{U_4}{D_2}, \end{aligned} \tag{4.2}$$

where D_1 and D_2 are the planar dimensions of the plate.

As mentioned, the input space was sampled through nonlinear incremental-iterative finite element analyses. Figure 4.2 illustrates the sampling strategy used in this work in the simplified case of two-dimensional input. We refer to this technique as *radial sampling*, due to the fact that the design of experiments (DoE) does not directly

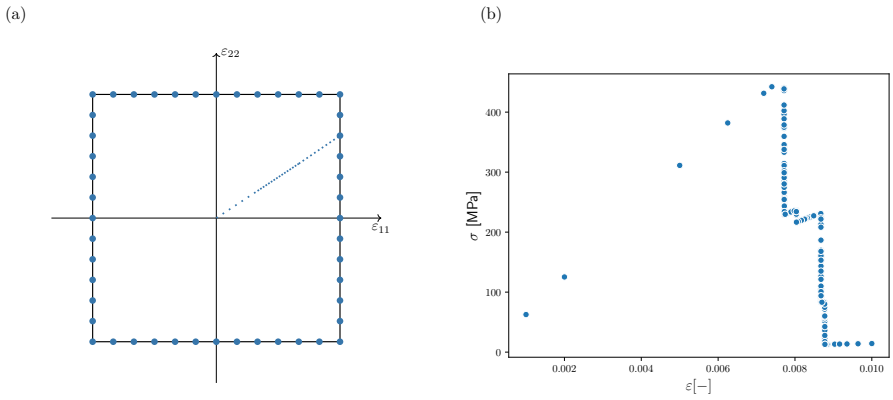

Figure 4.2.: (a) Radial strategy for sampling the far-field strain space. The bigger dots on the edge of the $(\varepsilon_{11}, \varepsilon_{22})$ volume represent the final applied load in different nonlinear FE incremental-iterative analyses. The arrow represents a loading path with the load increments unevenly distributed. (b) Load-displacement curve corresponding to the loading path in (a).

affect all the points in this input space, but only the ones on the boundary. On the other hand, all the intermediate points are generated internally by the FE solver and they correspond to the homogenized strain values at every time increment. The user maintains control of the inner samples values, by the choice of initial, minimum and maximum time steps. For this work, we chose the sampling space to be the hypercube $\left[-10^{-2}, -2\right]^{\otimes 3}$ in $\mathbb{R}^3$, meaning that all three components of the applied strains vector have the same bounds.

We argue that radial sampling is the most efficient sampling choice when the experiments providing the data are finite element nonlinear incremental analyses or even lab tests on coupons. In both cases, the user parameters are the final load applied on the coupon, while the experiment's output is a load-displacement curve with intermediate stress-strain readings. As Figure 4.2 demonstrates, radial sampling covers the input space extensively by performing experiments only on the boundary of the input space. More typical techniques, such as Latin Hypercube Sampling (LHS), would either miss certain regions of the input space or require experiments over the whole interior.

Each strain sample was assigned a label based on an ultimate failure criterion. In particular, we defined failure by the loss of stiffness of the laminate for given a user-defined threshold.

From the results of the FE analyses with periodic boundary conditions,

one obtains the reaction forces F_1, F_2, F_3 and F_4 conjugate to the degrees of freedom in Equation (4.1). These provide the homogenized stresses, which can then be derived via the Hill-Mandel principle of energy balance as

$$\begin{aligned}\sigma_{11} &= \frac{F_1}{tD_2}\\ \sigma_{22} &= \frac{F_2}{tD_1}\\ \sigma_{12} &= \frac{F_3U_3 + F_4U_4}{\gamma_{12}tD_1D_2},\end{aligned} \tag{4.3}$$

where t is the thickness of the plate.

The laminate stiffness in the two axial directions and in shear can thus be defined at every timestep t as

$$\begin{aligned}E_1^{(t)} &= \frac{\sigma_{11}^{(t)}}{\varepsilon_{11}^{(t)}}\\ E_2^{(t)} &= \frac{\sigma_{22}^{(t)}}{\varepsilon_{22}^{(t)}}\\ G_{12}^{(t)} &= \frac{\sigma_{12}^{(t)}}{\varepsilon_{12}^{(t)}}.\end{aligned} \tag{4.4}$$

The stiffness degradation d_S is defined as the minimum ratio between the instantaneous stiffness and the corresponding stiffness measure in the linear elastic region,

$$d_S^{(t)} = \min\left\{\frac{E_1^{(t)}}{E_1^{(0)}}, \frac{E_2^{(t)}}{E_2^{(0)}}, \frac{G_{12}^{(t)}}{G_{12}^{(0)}}\right\} \tag{4.5}$$

Therefore, given M the total number of samples, every sample $\boldsymbol{\varepsilon}^{(m)}$ $(m = 1, \dots, M)$ is assinged a label $y^{(m)} = -1$ if $d^{(m)} < \bar{d}_S$ and $y^{(m)} = +1$ otherwise.

4.3. METHODOLOGY

As already mentioned, we solve the ultimate failure binary classification problem using the SVM algorithm, [22]. Intuitively, given samples belonging to two different classes, the SVM algorithm finds the hyperplane with the highest margin between the classes, in order to increase the confidence of labeling further unseen samples. In

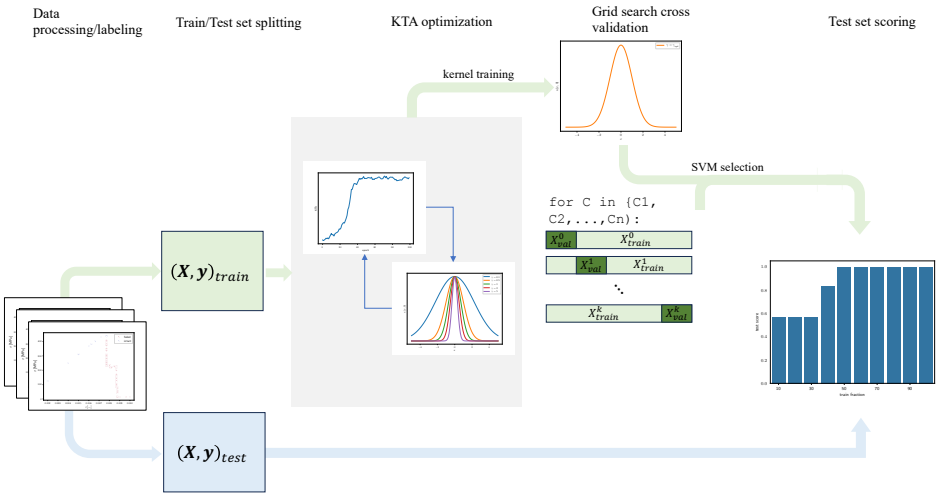

Figure 4.3.: The methodology used in this work. The dataset is generated by nonlinear finite element analyses, then labeled and split into training and testing sets. The training set is used first to train the kernel, by optimizing the kernel-target alignment (KTA), then in a grid search cross-validation to find the best slack penalty C of the SVM. With all the hyperparameters fixed, the SVM is trained for increasing dataset sizes and the classification accuracy is evaluated on the testing set.

particular, the SVM translates to a quadratic optimization problem, which can be written in dual form as

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{m=1}^{M} \alpha^{(m)} - \frac{1}{2} \sum_{m,m'=1}^{M} y^{(m)} y^{(m')} \alpha^{(m)} \alpha^{(m')} \kappa\left(\boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m')}\right) \\ \text{s.t.} \quad & 0 \leq \alpha^{(m)} \leq C, \quad m = 1, \ldots, M \\ & \sum_{m=1}^{M} \alpha^{(m)} y^{(m)} = 0, \end{aligned} \tag{4.6}$$

where $\boldsymbol{x}^{(m)} = \boldsymbol{\varepsilon}^{(m)}$, $y^{(m)} = \pm 1$ are the labels, respectively non-failed and failed and $\alpha^{(m)}$ are the Lagrange multipliers. The slack penalty C is a hyperparameter, which trades off the amount of constraint violations of the samples and the magnitude of the margin with respect to the hyperplane. Finally, the kernel function $\kappa(\cdot, \cdot)$ is a similarity metric between two samples in a higher-dimensional feature space. More details on the SVM algorithm are left to Appendix B.

The performance of the dual SVM depends on the choice of its hyperparameters, namely the kernel function κ and slack penalty C. To restrict the search space, the kernel function is generally parametrized via one or more parameters $\boldsymbol{\theta}$ and the standard practice is to perform a grid-search cross-validation procedure in the $(\boldsymbol{\theta}, C)$ space. In this work, we use instead a mixed procedure, where the kernel function is determined by optimizing the kernel-target alignment (KTA, [27] and the slack penalty is found by grid search cross-validation. The overall methodology is illustrated in Figure 4.3, where we refer to the two steps as *kernel training* and *SVM selection*. Once the SVM has been fully determined, it can be trained by solving Equation (4.6) and its learning ability can be measured as the accuracy on unseen test data, for different training dataset sizes.

We compare one classical and two quantum kernels. The classical kernel is the radial basis function (RBF) kernel, defined as

$$\kappa_{\text{RBF}}\left(\boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m')}\right) = \exp\left(-\gamma \|\boldsymbol{x}^{(m)} - \boldsymbol{x}^{(m')}\|^2\right). \tag{4.7}$$

RBF is a powerful kernel which corresponds to a feature map in an infinite-dimensional feature space, [28]. It induces a Gaussian similarity function, whose width is controlled by the hyperparameter γ.

On the other hand, the quantum kernel is defined via a *quantum embedding*, which is constructed via data-depending unitary transformations $U(\boldsymbol{x})$ that prepare the quantum state

$$|\psi(\boldsymbol{x})\rangle = U(\boldsymbol{x})|0\rangle. \tag{4.8}$$

Given two samples $\boldsymbol{x}^{(m)}$ and $\boldsymbol{x}^{(m')}$, the quantum kernel is simply the

inner product

$$\kappa_Q\left(\boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m')}\right) = \left|\langle\psi\left(\boldsymbol{x}^{(m)}\right), \psi\left(\boldsymbol{x}^{(m')}\right)\rangle\right|^2. \tag{4.9}$$

Figure 4.4 shows the generic quantum embedding and the two specific ones used in this work, which are the hardware efficient embedding (HE2), [29] and the instantaneous quantum polynomial (IQP), [30] one. To have more expressive feature mapping, either the *width* or the *depth* of the quantum embedding can be increased. The first one is the number of qubits, which can be even higher than the number of features in the dataset, by cyclically re-encoding the features to generate a highly nonlinear and potentially better separable feature space. Meanwhile, the embedding's depth can be increased by repeating a base data-encoding block, such as IQP and HE2. Even in this case, re-encoding of the features may lead to a higher expressivity of the overall feature map, [31].

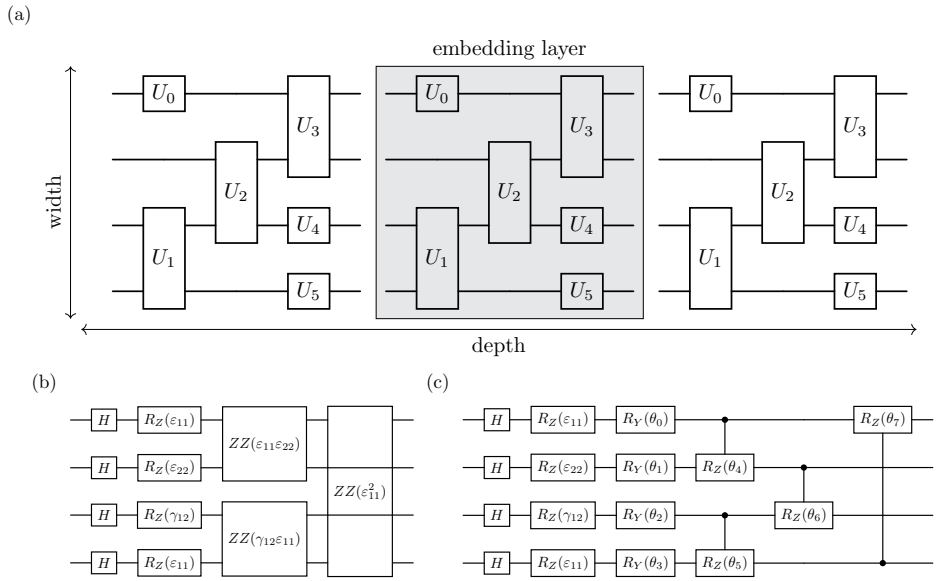

Figure 4.4.: Quantum embeddings. (a) Generic quantum embedding made of one- and two-qubit gates. The *width* of the embedding is the number of qubits, while the *depth* is the number of layers, which is a minimal block of gates. (b) IQP quantum embedding layer, which is parameter free, but encodes products of features as *ZZ* interactions. (c) HE2 quantum embedding layer, parametrized by $[\theta_0, \theta_1, \ldots, \theta_{P-1}]$.

4.4. RESULTS

We tested our machine learning models on a dataset of 1960 labelled strain vectors $\boldsymbol{\varepsilon}^{(m)}$, which we obtained by uniformly sampling the homogeneous strain/stress pairs from the FE simulations of the open-hole composite specimen. The input homogeneous strains in both normal and shear directions were varied between -10^4 and 10^4 microstrains and a stiffness degradation threshold of 0.9 was used to discriminate non-failed and failed loading states.

Both classical- and quantum-kernel SVMs were implemented using different Python libraries. We used PyTorch for training the RBF kernel and PennyLane for the quantum kernels. These libraries implement automatic differentiation (AD), which allows to optimize the KTA with gradient-based methods. We also used JAX together with PennyLane to just-in-time compile the quantum kernel functions. Concerning the classification problem, we employed the SVM and grid-search cross validation routines available from the Scikit-Learn Python package.

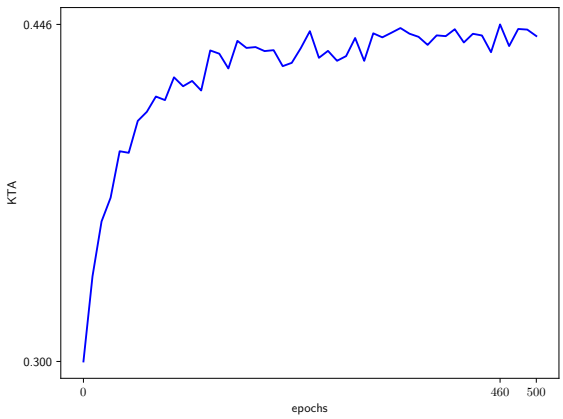

Figure 4.5.: Training history of the RBF kernel's KTA.

The KTA of both RBF and quantum kernels was maximized the Adam optimizer, [32]. Figure 4.5 shows the kernel alignment training of the RBF kernel. Figure 4.6 presents instead the KTAs before and after training for nine different quantum kernels with HE2 embedding. It can be seen that increasing width and depth of these kernels generally improves their KTA. A higher number of qubits means that the strain features are mapped in a higher dimensional space, which can favor separability of the classes. On the other hand, increasing the depth benefits the kernel alignment, since it results in more expressive feature maps. Also, every additional layer of the HE2 embedding doubles the number of free parameters, explaining why optimization of deeper kernels mostly leads to higher gains in KTA. However, the advantage

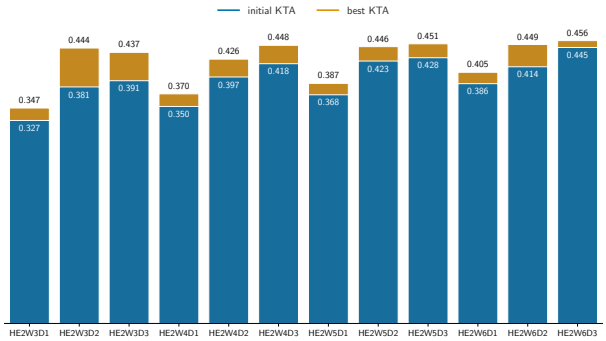

Figure 4.6.: KTA of quantum kernels before and after training for 9 different quantum embeddings. The basic layer for all embeddings is the HE2, which has trainable parameters. The tag WXDY indicates width and depth of the embedding.

of increasing quantum encoding resources does not scale uniformly. Already with 6 qubits and 3 HE2 layers, the optimization only modestly improves the KTA, likely due to the vanishing KTA gradients, [33].

To find the hyperparameter C that guarantees the highest off-training accuracy of the SVM algorithm, we used grid-search cross validation for the kernels considered. The validation accuracy values are reported in Figure 4.7 for multiple values of C and γ. We observe that kernels with $\gamma \leq 10$ achieve the higest scores, with the highest-KTA γ scoring first for the whole range of C values. Furthermore, the accuracy of the maximally-aligned RBF kernel increases monotonically with C, which suggests the usefulness of maximizing the KTA, but also that the class boundary in this feature space is densely populated and still requires a tight margin.

The same analysis was performed for all the quantum kernels considered, where we wanted to take into account the effect on accuracy of different embeddings and of maximizing the kernel-target alignment. The results are reported in Figure 4.8, which shows accuracies roughly between 67% and 87% for all embeddings with different values of C. Except for IQP case, increasing C leads to higher accuracies, hinting to the need of a tight bound when mapping with these embeddings, similar to the RBF kernel. Unfortunately, for high values of C, the optimization of the dual SVM failed to converge for some of the quantum kernels, likely due to numerical ill-conditioning. This presumably prevented the quantum kernel classifiers from even better separating failed instances, as suggested by the monotonic increasing test accuracies with C, at least for the HE2 kernels. Furthermore, Figure 4.8 shows that the scores improve when more embedding

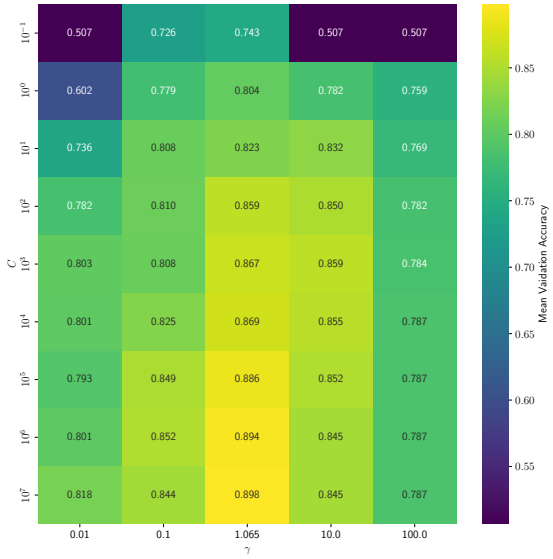

Figure 4.7.: Grid-search cross validation for the RBF kernel. γ=1.065 corresponds to the kernel with highest kernel-target alignment.

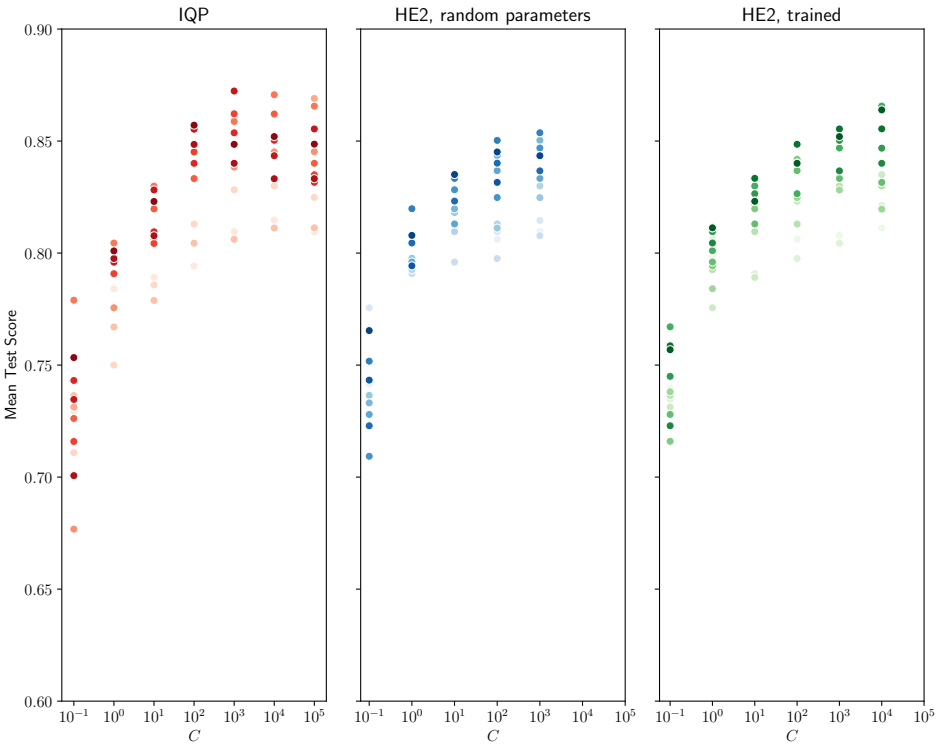

Figure 4.8.: Summary of the grid-search cross validation results for different quantum kernels. The three figures correspond respectively to the IQP embeddding, the HE2 kernel with untrained parameter and HE2 with trained parameters. Different shades of the same color correspond to different depths and widths. From lighter to darker, the points correspond to embeddings of increasing widths and of increasing depth per fixed width.

resources (number of qubits and layers) are added, especially in the case of KTA-optimized HE2 kernel.

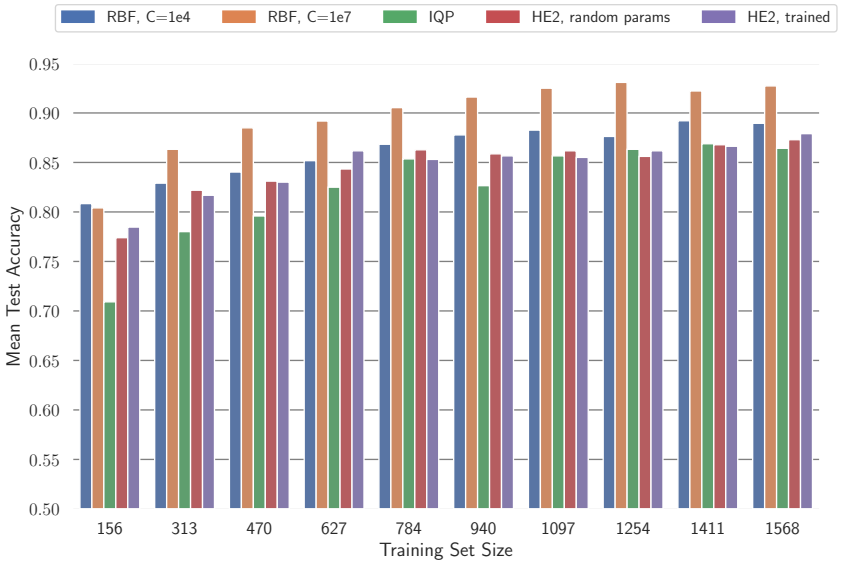

Figure 4.9.: Test data classification accuracy of classical and quantum kernels for increasing training set size. The RBF kernel-SVMs were trained with both $C = 10^7$ and $C = 10^4$, as the latter is the highest value of C for which the quantum-kernel SVMs could still be fitted. Quantum-kernel classifiers were trained instead with the embedding arhitecture and C value ensuring highest accuracy during grid-search cross validation.

Classical and quantum kernels are finally compared in Figure 4.9, which shows how 5 different models classify a test set of strain loading data when fitted on progressively larger training sets. A similar comparison on additional classification metrics can be found in Appendix C. The RBF kernel achieves 80% accuracy with just 10% the total training set size, and with $C = 10^7$ it reaches over 90% with just half the training points. In comparison, all quantum kernel classifiers are at least 5% less accurate than the best RBF-kernel SVM. However, especially for HE2 embeddings, the scores are similar to the $C = 10^4$ RBF case, suggesting that RBF and HE2 kernels separate the non-failed and failed classes to a similar extent. Changing the embedding from HE2 to IQP, there is a drop in accuracy for small training set sizes, while the performance is similar when more than half the training set is used. On the other hand, the effect of training the kernel is less visible at this stage, reflecting the fact that the accuracies obtained during grid-search cross validation are

alike for untrained and trained HE2.

4.5. CONCLUSION

In this Chapter, we proposed a methodology to build a binary classifier from finite element analyses data for the particular case of an open hole composite specimen. We studied the case of in-plane strain loading of the specimen where the objective is to correctly label strain combinations that lead to ultimate failure.

From a design of experiment point of view, we demonstrated a radial sampling strategy technique, where the choice of which simulations to make to cover the input space takes into account the incremental-iterative nature of the nonlinear FE method. We then proposed a labelling criterion of homoegenized strain-stress pairs based on residual in-plane stiffness.

For classification of the labelled data, we used the kernel-based SVMs, which also allowed us to compare the performance of the recently proposed quantum kernels against the more traditional RBF. Furthermore, we optimized the kernel-target alignment to improve class separbility of both RBF and the HE2 embedding kernel.

For all the kernel examined, the corresponding SVMs separate non-failed and failed loading states with good accuracy. The RBF-based model classify more accurately than SVM with quantum kernels, although this likely happens due to numerical ill-conditioning in the current quantum SVM implementation. These numerical issues can likely be fixed by studying the dual SVM problem for the problematic instances, which could be investigated in future work.

Regarding kernel alignment, optimizing the KTA is shown to be powerful for RBF, since the SVM for the trained kernel outperforms the other RBF-based models in terms of accuracy. Aligning quantum kernels for this dataset also helps them to better separate the two classes, but for simple architectures the improvement is moderate, while more complex embeddings only reach the scores of the more simple ones after they have been aligned. Furthermore, one should remember that optimizing quantum kernels is almost always more computationally involved than for RBF, as the formers can have highly parametrized embeddings, while RBF is completely defined by the single parameter γ.

Extensions of this work can go in many directions. From the point of view of the problem, it would be interesting to increase the number of degrees of freedom, by allowing the notch radius or the lamination sequence to also change. The latter could be written in terms of lamination parameters, [34] to have a continuous representation.

In terms of algorithms, both classical and quantum kernels can be explored further. RBF is the most popular choice for classical kernels, but certainly not the only one. Due to Mercer's condition, any function which

defines a positive semi-definite kernel matrix is a valid kernel function, [28]. Obviously, the design space is vast, but automated procedures help reduce the search for instance by exploring combinations of only a fixed set of standard kernel functions.

On the other hand, the freedom of designing and parametrizing quantum embedding circuit also makes the choice of a quantum kernel nontrivial. Within the limits of classical simulation of quantum circuits, one could experiment with increasing number of qubits or different layering strategies, for instance the one proposed in [35] for the task of satellite image classification. From an optimization point of view, a recent technique has been proposed to maximize the quantum kernel alignment KTA and solving the SVM in a single optmization loop, [36], which would of course greatly reduce the computational cost. Nevertheless, to truly understand a potential competitiveness of quantum kernels, it is probably most important to remove layers of simulation and study the effects of statistical and hardware noise on SVM convergence and accuracy.

REFERENCES

[1] G. Tosti Balducci, B. Chen, M. Möller, M. Gerritsma, and R. D. Breuker. *Predicting Open-Hole Laminates Failure Using Support Vector Machines With Classical and Quantum Kernels*. 2024. arXiv: 2405.02903 [cs.CE].

[2] J. Whitney and R. Nuismer. "Stress Fracture Criteria for Laminated Composites Containing Stress Concentrations". In: *Journal of Composite Materials* 8.3 (July 1974), pp. 253–265.

[3] P. Camanho, G. Erçin, G. Catalanotti, S. Mahdi, and P. Linde. "A finite fracture mechanics model for the prediction of the open-hole strength of composite laminates". In: *Composites Part A: Applied Science and Manufacturing* 43.8 (Aug. 2012), pp. 1219–1225.

[4] G. Catalanotti, R. M. Salgado, and P. P. Camanho. "On the Stress Intensity Factor of cracks emanating from circular and elliptical holes in orthotropic plates". In: *Engineering Fracture Mechanics* 252 (July 2021), p. 107805.

[5] C. Furtado, A. Arteiro, M. Bessa, B. Wardle, and P. Camanho. "Prediction of size effects in open-hole laminates using only the Young's modulus, the strength, and the R-curve of the 0° ply". In: *Composites Part A: Applied Science and Manufacturing* 101 (2017), pp. 306–317.

[6] P. Camanho, P. Maimí, and C. Dávila. "Prediction of size effects in notched laminates using continuum damage mechanics". In: *Composites Science and Technology* 67.13 (Oct. 2007), pp. 2715–2727.

[7] S. Hallett, B. Green, W. Jiang, and M. Wisnom. "An experimental and numerical investigation into the damage mechanisms in notched composites". In: *Composites Part A: Applied Science and Manufacturing* 40.5 (May 2009), pp. 613–624.

[8] F. P. van der Meer, L. J. Sluys, S. R. Hallett, and M. R. Wisnom. "Computational modeling of complex failure mechanisms in laminates". In: *Journal of Composite Materials* 46.5 (Sept. 2011), pp. 603–623.

[9] B. Chen, T. Tay, P. Baiz, and S. Pinho. "Numerical analysis of size effects on open-hole tensile composite laminates". In: *Composites Part A: Applied Science and Manufacturing* 47 (Apr. 2013), pp. 52–62.

[10] F. van der Meer, C. Oliver, and L. Sluys. “Computational analysis of progressive failure in a notched laminate including shear nonlinearity and fiber failure”. In: *Composites Science and Technology* 70.4 (Apr. 2010), pp. 692–700.

[11] B. Chen, T. Tay, S. Pinho, and V. Tan. “Modelling the tensile failure of composites with the floating node method”. In: *Computer Methods in Applied Mechanics and Engineering* 308 (Aug. 2016), pp. 414–442.

[12] C. Soutis, N. A. Fleck, and P. A. Smith. “Failure Prediction Technique for Compression Loaded Carbon Fibre-Epoxy Laminate with Open Holes”. In: *Journal of Composite Materials* 25.11 (Nov. 1991), pp. 1476–1498.

[13] Z. Su, T. Tay, M. Ridha, and B. Chen. “Progressive damage modeling of open-hole composite laminates under compression”. In: *Composite Structures* 122 (2015), pp. 507–517.

[14] R. Higuchi, S. Warabi, A. Yoshimura, T. Nagashima, T. Yokozeki, and T. Okabe. “Experimental and numerical study on progressive damage and failure in composite laminates during open-hole compression tests”. In: *Composites Part A: Applied Science and Manufacturing* 145 (2021), p. 106300.

[15] C. Bisagni and L. Lanzi. “Post-buckling optimisation of composite stiffened panels using neural networks”. In: *Composite Structures* 58.2 (Nov. 2002), pp. 237–247.

[16] M. Bessa and S. Pellegrino. “Design of ultra-thin shell structures in the stochastic post-buckling range using Bayesian machine learning and optimization”. In: *International Journal of Solids and Structures* 139-140 (May 2018), pp. 174–188.

[17] Z. Zhang, Z. Zhang, F. Di Caprio, and G. X. Gu. “Machine learning for accelerating the design process of double-double composite structures”. In: *Compos. Struct.* 285 (Apr. 2022), p. 115233.

[18] X. Liu, S. Tian, F. Tao, and W. Yu. “A review of artificial neural networks in the constitutive modeling of composite materials”. In: *Composites Part B* 224 (Nov. 2021), p. 109152.

[19] N. Zobeiry, J. Reiner, and R. Vaziri. “Theory-guided machine learning for damage characterization of composites”. In: *Composite Structures* 246 (Aug. 2020), p. 112407.

[20] J. Reiner, R. Vaziri, and N. Zobeiry. “Machine learning assisted characterisation and simulation of compressive damage in composite laminates”. In: *Compos. Struct.* 273 (Oct. 2021), p. 114290.

[21] C. Furtado *et al.* “A methodology to generate design allowables of composite laminates using machine learning”. In: *International Journal of Solids and Structures* 233 (Dec. 2021), p. 111095.

[22] C. Cortes and V. Vapnik. “Support-vector networks”. In: *Mach. Learn.* 20.3 (Sept. 1995), pp. 273–297.

[23] B. Chen and G. T. Balducci. *Nonlinear responses of metals and composites*. Dec. 2022. doi: 10.5281/zenodo.7409612.

[24] G. Tosti Balducci. *oh-comp-kernels: A Python code for the prediction of the open-hole tensile strength of composite laminates using kernel methods*. https://github.com/debrevitatevitae/oh-comp-kernels. 2023.

[25] B. Green, M. Wisnom, and S. Hallett. “An experimental investigation into the tensile strength scaling of notched composites”. In: *Composites Part A: Applied Science and Manufacturing* 38.3 (Mar. 2007), pp. 867–878.

[26] Dassault Systèmes. *ABAQUS Finite Element Analysis Software*. 2022.

[27] T. Wang, D. Zhao, and S. Tian. “An overview of kernel alignment and its applications”. In: *Artif. Intell. Rev.* 43.2 (Feb. 2015), pp. 179–192.

[28] B. Schoelkopf and A. J. Smola. *Learning with kernels*. Adaptive Computation and Machine Learning Series. London, England: MIT Press, June 2019.

[29] T. Hubregtsen, D. Wierichs, E. Gil-Fuster, P.-J. H. S. Derks, P. K. Faehrmann, and J. J. Meyer. “Training quantum embedding kernels on near-term quantum computers”. In: *Phys. Rev. A* 106.4 (Oct. 2022), p. 042431.

[30] O. Kyriienko and E. B. Magnusson. *Unsupervised quantum machine learning for fraud detection*. 2022. arXiv: 2208.01203 [quant-ph].

[31] M. Schuld, R. Sweke, and J. J. Meyer. “Effect of data encoding on the expressive power of variational quantum-machine-learning models”. In: *Phys. Rev. A* 103.3 (Mar. 2021), p. 032430.

[32] D. P. Kingma and J. Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015.

[33] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. “Barren plateaus in quantum neural network training landscapes”. In: *Nature Communications* 9.1 (Nov. 2018).

[34] S. Setoodeh, M. M. Abdalla, and Z. Gürdal. “Design of variable-stiffness laminates using lamination parameters”. In: *Composites Part B: Engineering* 37 (4-5 June 2006), pp. 301–309.

[35] A. Miroszewski *et al.* “Detecting Clouds in Multispectral Satellite Images Using Quantum-Kernel Support Vector Machines”. In: *arXiv:2302.08270* (Feb. 2023).

[36] G. Gentinetta, D. Sutter, C. Zoufal, B. Fuller, and S. Woerner. “Quantum Kernel Alignment with Stochastic Gradient Descent”. In: *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Vol. 01. 2023, pp. 256–262.

5

CONCLUSIONS

This thesis explored the intersection between quantum computing and computational structural mechanics. It started by reviewing the literature on quantum techniques for solving partial differential equations (PDEs) in structural mechanics, where it analized not just the runtimes of the quantum primitives, but the full time complexity from data encoding to readout and the hardware feasibility. Then, it moved to a hybrid quantum classical algorithm, the Variational Quantum Linear Solver and used it to solve a prototypical PDE in mechanics, namely the Poisson problem in one dimension. We discussed the linear scaling of the common Pauli decomposition for the Poisson matrix and proposed an alternative which is still linear, but with half the number of terms. Finally, this thesis touched upon quantum machine learning, specifically, quantum kernel methods. It characterized the binary classification problem of labelling loading states of composite specimens and compared state-of-the-art classical kernels and multiple quantum kernels in their ability to separate the label sets.

The rest of this chapter retrospects on the results achieved and knowledge gaps filled, it answers the research questions presented in Chapter 1 and it discusses future lines of research and possible extensions of this work.

5.1. CONRIBUTIONS TO THE EXISTING LITERATURE

The general contribution of this work has been to bring some of the methods of quantum computing into structural mechanics. To the best of the author's knowledge, this is the first work of considerable extent that analysed and demonstrated the use of quantum solvers in different structural mechanics applications. Only maybe a few standalone papers in literature seem to have so far covered the subject, [1].

We provided a review of quantum methods for linear and nonlinear PDEs in the domain of structural mechanics. At the time of publishing

this study, no other review of quantum PDE solvers existed, although recently a similar overview for fluid dynamics was conducted, [2]. The analysis we performed was a critical one, especially on the subject of speedups. In this respect, our work is different compared to the bulk of quantum algorithms research papers for applications, which state the runtime of their quantum primitives, but hide the complexity of quantum data upload and readout.

Applying the VQLS algorithm to the 1D Poisson problem revealed the decomposition scaling bottleneck. Shortly after the time that our work took place, other efforts looked at variational approaches to solve the Poisson problem, even for higher dimensions, [3, 4]. One strategy proposed a technique to reduce the VQLS cost terms for the Poisson problem, [3]. The approach of these researchers resulted in logarithmic reduction, against our more modest halving of the number of terms, thanks to them focusing on the structure of the problem's Hamiltonian, rather than the A matrix. While we know now that the 1D Poisson global Hamiltonian is more efficiently decomposable than A itself, one should keep in mind that global (i.e. applied to the full register) VQA Hamiltonians suffer from barren plateaus, [5]. On the other hand, saving *matrix* decomposition terms, as we proposed, leaves the flexibility of choosing a local Hamiltonian in the VQA recipe, potentially saving trainability.

The final work we presented is a study on composite failure identification with classical and quantum kernel methods. This is probably also one of the first applications of quantum machine learning to a real-world problem. Some aspects of the specific learning problem itself are likely new themselves. Our work seems to be the first one to propose a surrogate failure envelope by solving a binary classification problem. Also novel is the radial sampling technique we proposed to avoid repetitions of experiments when sampling the input space with nonlinear finite element analyses.

5.2. ANSWERS TO THE RESEARCH QUESTIONS

Based on the results of this research, we can answer the research questions posed in Chapter 1.

The first question treats the subject of speedup of quantum PDE solvers, compared to classical ones.

> *Which fault-tolerant and near-term quantum algorithms can solve (parts of) which PDEs in structural mechanics? What can be said about their speedup with respect to classical solvers?*

Chapter 2 reviewed several quantum and hybrid-quantum PDE algorithms in order to answer this question.

Both fault-tolerant and NISQ techniques exist for linear PDEs, in particular the Poisson, heat and wave equations. For nonlinear PDEs, the quantum algorithms proposed so far are general and not specialized to single PDEs.

Fault-tolerant routines have either polynomial or exponential speedups, but only at the quantum primitive level. Generic state preparation and/or readout re-introduces the polynomial scaling. However, there are special input states and measurable quantities that only have logarithmic complexity. It is still an important open question whether this narrow 'goldilocks zone' of state preparations, quantum primitive and readouts contains useful use-cases.

On the other hand are NISQ approaches, whose runtime cannot be determined independently of the specifc application. However, there can be necessary conditions, as in the case of VQLS, where a logarithmic runtime is possible only with a logarithmic decomposition of the linear system's matrix.

A promising subset of NISQ PDE methods train quantum circuits to learn the PDE solution [6–9]. These are essentially the quantum model variant of the famous Physics-Informed Neural Networks [10–12]. The main motivation behind using quantum models is their higher expressivity compared to, e.g., classical neural networks with similar number of parameters. Nevertheless, quantum advantage on practical use-cases has not yet been proven using these techniques. Even just establishing that circuit learning achieves quantum advantage in an application is no easy endeavor, as a final answer will be based on experimental evidence and will have to motivate matters such as expressivity, trainability, sensitivity to noise and quantum hardware overhead, among others.

The second research question targets the Variational Quantum Linear Solver (VQLS), an algorithm proposed as a quantum routine for solving PDEs. A necessary condition for advantage of the VQLS is that the input matrix is decomposable in $\log(n)$ terms, n being the number of qubits representing the normalized solution of the linear system. In Chapter 3 we studied the possibility to logarithmically decompose the discretized Poisson 1D equation.

> *Are there* $\log(n)$ *decompositions of the one-dimensional Poisson matrix?*

Our work did not find one such decomposition. Even though we determined a decomposition using multi-qubit gates, we could only nearly halve the number of terms resulting from the standard Pauli decomposition. Furthermore, the multi-qubit gates we introduce are complex multi-controlled gates, which act on the full register[1]. Even

[1]Assuming a register size equal to $\log(N)$, where N is the linear system's size.

assuming full-connectivity, which is not yet available in digital quantum hardware, these gates compile to a large number of noisyt two-qubit gates, which pollute the loss evaluations.

A legitimate question is whether it is worth to run a hybrid quantum algorithm to solve the 1D Poisson equation, which is a trivial classical problem with an analytical solution. Of course, there is no practical relevance in this task, as an application. However, the aim of this study has not been to pursue practical quantum advantage, but rather to assess performance and bottlenecks of a near-term quantum algorithms, applied to a toy, yet classical problem. Quite intuitively, this work showed that the *simplicity* of a problem depends on the logic of the method that we use to solve it. Specifically to our case, even though specific matrix structures derived from quantum physics Hamiltonians might be promising candidates for quantum advantage with a variational linear solver, there is no physical intuition on why the same algorithm should outperform even simple classical linear systems.

The final research question delas a more realistic scenario and concerns the performance of classical and near-term quantum machine learning to classify safe and failure-inducing loads applied to a open-hole composite plate.

> *To what accuracy can quantum-kernel based support vector machines classify failed open-hole composite specimens, when trained on numerical simulations data? How does that performance compare against that of state-of-the-art classical kernels?*

Chapter 4 showed that the quantum-kernel SVMs classify open-hole composite ultimate failure with performance comparable to classical-kernel SVMs (radial basis function kernel), with the quantum kernels slightly under-performing. RBF scored 93% accuracy on a test dataset and the best quantum kernel got 87% test loading states correctly classified. We encountered, however, a convergence issue in running quantum-kernel SVMs, which did not allow to ultimately determine the performance of these models.

Methodologically, we considered two different circuit architectures for the quantum kernels, HEA, which has trainable parameters and IQP, which is non trainable. Both architectures are modular, which easily allowed to increase the register size and number of basic layers. We also tuned the hyperparameters of the algorithm, those belonging to the kernels, using kernel-target alignment optimization and the SVM penalty strength C via grid-search cross validation. The performance of different trained and untrained quantum kernels and the RBF were compared on a set of unseen loading conditions.

Besides classification accuracy, we detected differences between RBF and quantum models when optimizing the kernel-target alignment. On

the one hand, RBF clearly benefitted from the higher alignment, as demonstrated by the higher validation scores of the aligned model with respect to non-aligned ones. Interepreting trainability for quantum models was more subtle. Ultimately, the scores on the test set of untrained and trained HE2 kernels were comparable, regardless of the size of the dataset used in training. Though this might be due to the high penalty strength that the final models shared, we noticed in general an insensitivity of quantum kernels to alignment optimization, at least for more complex datasets like the open-hole composite failure one.

5.3. RECOMMENDATIONS FOR FUTURE WORK

We end our discussion by presenting some directions for future work for each of the topics touched on in this thesis.

Quantum computers are still under the radar as PDE solving accellerators. Being in the noisy era of quantum hardware, near-term algorithms will likely resort to variational techniques. A promising instance we identified in Chapter 2 is the differentiable quantum circuits (DQC) protocol, originally proposed in [6], which is one of the few quantum algorithms that can be applied to nonlinear differential equations and that easily implements differentiation. From the time of writing our review work, this method has been extended to trainable frequency feature maps, [8], specialized to harmonic functions, [13] and was successfully applied in fields such as weather modeling, [9].

There is, however, not yet a consensus on which might be the 'killer' problem for DQC. This is partly due to the fact that the technique is still young and has not yet been applied to many domains of science and engineering, structural mechanics among them. More in general, there is a lack of knowledge about which characteristics of a PDE would make it a good fit for DQC and perhaps a better fit than for classical machine learning models or even established numerical methods. How to find the right application for DQC advantage? We argue that this will be a problem whose dynamics can be captured by a quantum model complex enough to not be classically simulatable and whose expressivity scales monotonically with increasing resources, e.g. width and depth of the circuit. From expressivity analysis, we know that quantum learnable models can be expressed as series of terms in different bases, for instance Fourier modes, [14], or Chebyshev polynomials, [6]. The bandwidth of the available modes depends on the number of encoding operations and the accessible modes, i.e. those with nonzero coefficient, depends on the variational part (ansatz). Therefore, problems with propagation of finite-spectrum signals and having finite-spectrum dynamics, i.e. without discontinuities, might be a good fit for DQC. For instance, in structural engineering, DQC could find application in nonlinear structural dynamics.

Yet, research in quantum PDE solvers should not be confined to variational routines. Advances in quantum technologies are opening the next quantum computing era, known as ISQ (intermediate scale quantum), [15]. Thanks to the implementation of error-correcting codes, upcoming quantum computers will offer a few, but error-compensated qubits. Some recent works proposed to solve PDEs in latent space, [7], or using linearization and a quantum iterative linear system solvers, [16]. In contrast to more standard matrix-inversion or Hamiltonian-simulation based solvers, these techniques have lower hardware requirements and may be running on machines with a small register of error-corrected qubits. Nevertheless, what is still missing is an analysis of end-to-end complexity from classical data encoding to readout of the solution or a measure of interest. Future efforts on quantum PDE solving should perhaps narrow their scope from general primitives for differential equations to the specific problems and discuss whether end-to-end advantage exists for those instances.

Keeping the classical step of discretizing a PDE, one can use the VQLS algorithm to solve the resulting linear system. This strategy was used in Chapter 3 to solve the 1D Poisson problem. Even though VQLS is a NISQ algorithm and [17] suggests that the cost function evaluation is classically hard, it is still unclear if VQLS can ouperform established classical algorithms in some use-cases. If we exclude the encoding and readout bottlenecks, the time complexity of VQLS is lower-bound in general by the number of unitary terms that decompose the matrix A. For random matrices, this scaling alone is worse than the number of steps in Gaussian elimination, [18]. Furthermore, even when $O(\log N)$ or $O(1)$ decompositions were found, [3, 4], the resulting Hamiltonian is global, which means that interesting problem sizes will have a flat loss landscape and that they will be difficult to optimize.

Therefore, future research on VQLS should focus on this combination of challenges. On the one hand, one should discover interesting differential problems whose matrices are efficiently decomposable. Of course, this would be more valuable if the decomposition accomodated different boundary conditions, as in [3] and [4]. At the same time though, novel approaches should be critical about trainability and seek local problem Hamiltonians, which do not suffer from exponentially vanishing gradients, [19]. Finally, since VQLS and fault-tolerant methods alike provide the full solution vector as a wavefunction, future literature should define solution metrics that are efficiently measurable.

Chapter 4 covered the subject of damage classification with classical and quantum kernels. The conclusions in Section 4.5 already point to several future research directions, such as the investigation of the lower trainability of quantum kernels, the extension of the input space to, e.g., the lamination sequence of the composite plate and the study of different quantum kernel structures. We provide some additional

recommendations here.

A general advice is to extend the quantum kernel selection procedure so that it is less arbitatry. In Chapter 4 we considered only two layered architectures (HEA and IQP), but an automated approach such as [20] can extend the search and potentially find better mappings to separate loading states based on failure.

However, more important would be aligning certain algorithmic choices with the specific task of identifying damage. A first adjustment would be to move from a binary to a continuous damage output of the machine learning model. For example, damage can be represented by the decision function of the SVM,

$$y(x) = \sum_{i \in \mathrm{SV}} \alpha_i y_i \kappa(x_i, x) + b.$$

where $y(x)$ is the signed distance of the state x from the decision boundary. An even better alternative would be to use Gaussian processes (GPs) instead of SVMs, since the damage would then be modeled as a probability distribution and bounded in $[0, 1]$, as it happens with continuous damage models.

Finally, quantum feature map selection could be made more problem-aware by including the symmetries of the composite plate and loading configuration. In our case, we considered a quasi-isotropic plate, whose response is symmetric with respect to the normal loading direction and sign of the shear deformation. Building the quantum circuit and choosing a parametrization based on these invariances will likely provide a higher class separation for comparable resources, [21, 22].

REFERENCES

[1] S. Srivastava and V. Sundararaghavan. “Box algorithm for the solution of differential equations on a quantum annealer”. In: *Phys. Rev. A* 99.5 (May 2019), p. 052355.

[2] S. Succi, W. Itani, K. Sreenivasan, and R. Steijl. “Quantum computing for fluids: Where do we stand?” In: *Europhys. Lett.* 144.1 (Oct. 2023), p. 10001.

[3] H.-L. Liu *et al.* “Variational quantum algorithm for the Poisson equation”. In: *Phys. Rev. A* 104.2 (Aug. 2021), p. 022418.

[4] Y. Sato, R. Kondo, S. Koide, H. Takamatsu, and N. Imoto. “Variational quantum algorithm based on the minimum potential energy for solving the Poisson equation”. In: *Phys. Rev. A* 104.5 (Nov. 2021), p. 052409.

[5] J. R. McClean, S. Boixo, V. N. Smelyanskiy, R. Babbush, and H. Neven. “Barren plateaus in quantum neural network training landscapes”. In: *Nature Communications* 9.1 (Nov. 2018).

[6] O. Kyriienko, A. E. Paine, and V. E. Elfving. “Solving nonlinear differential equations with differentiable quantum circuits”. In: *Physical Review A* 103.5 (May 2021).

[7] A. E. Paine, V. E. Elfving, and O. Kyriienko. “Physics-Informed Quantum Machine Learning: Solving nonlinear differential equations in latent spaces without costly grid evaluations”. In: *arXiv preprint arXiv:2308.01827* (2023).

[8] B. Jaderberg, A. A. Gentile, Y. A. Berrada, E. Shishenina, and V. E. Elfving. “Let quantum neural networks choose their own frequencies”. In: *Physical Review A* 109.4 (2024), p. 042421.

[9] B. Jaderberg *et al.* “Potential of quantum scientific machine learning applied to weather modeling”. In: *Phys. Rev. A* 110 (5 Nov. 2024), p. 052423.

[10] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations”. In: *arXiv preprint arXiv:1711.10561* (2017).

[11] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations”. In: *arXiv preprint arXiv:1711.10566* (2017).

[12] M. Raissi, P. Perdikaris, and G. E. Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378 (2019), pp. 686–707.

[13] A. Ghosh *et al. Harmonic (Quantum) Neural Networks*. 2023. arXiv: 2212.07462 [cs.LG].

[14] M. Schuld, R. Sweke, and J. J. Meyer. “Effect of data encoding on the expressive power of variational quantum-machine-learning models”. In: *Phys. Rev. A* 103.3 (Mar. 2021), p. 032430.

[15] *From NISQ to ISQ*. https://pennylane.ai/blog/2023/06/from-nisq-to-isq. Accessed: 2024-11-16.

[16] C. A. Williams, A. A. Gentile, V. E. Elfving, D. Berger, and O. Kyriienko. *Quantum Iterative Methods for Solving Differential Equations with Application to Computational Fluid Dynamics*. 2024. arXiv: 2404.08605 [quant-ph].

[17] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, and P. J. Coles. “Variational Quantum Linear Solver”. In: *Quantum* 7 (Nov. 2023), p. 1188.

[18] G. Turati, A. Marruzzo, M. F. Dacrema, and P. Cremonesi. “An Empirical Analysis on the Effectiveness of the Variational Quantum Linear Solver”. In: (Sept. 2024).

[19] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, and P. J. Coles. “Cost function dependent barren plateaus in shallow parametrized quantum circuits”. In: *Nature Communications* 12.1 (Mar. 2021).

[20] M. Incudini, D. L. Bosco, F. Martini, M. Grossi, G. Serra, and A. D. Pierro. “Automatic and Effective Discovery of Quantum Kernels”. In: *IEEE Transactions on Emerging Topics in Computational Intelligence* (2024), pp. 1–10.

[21] J. J. Meyer *et al.* “Exploiting Symmetry in Variational Quantum Machine Learning”. In: *PRX Quantum* 4 (1 Mar. 2023), p. 010328.

[22] Q. T. Nguyen *et al.* “Theory for Equivariant Quantum Neural Networks”. In: *PRX Quantum* 5 (2 May 2024), p. 020328.

A

KERNELS - OPEN HOLE SPECIMEN FEATURES AND FINITE ELEMENT MODEL DETAILS

A.1. GEOMETRY AND MATERIAL PROPERTIES

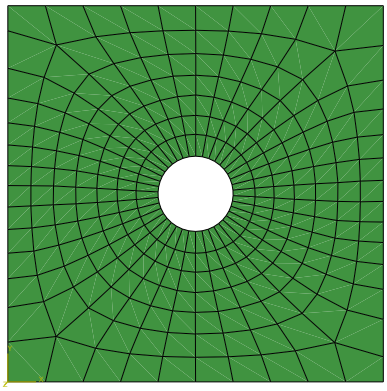

Figure A.1.: Finite element mesh of the composite specimen in Chapter 4.

The plate's hole has a 6 mm diameter and the in-plane dimensions are both 5 times the hole diameter. The ply material is IM7/8552 prepreg (carbon fibres and epoxy matrix) and each ply has $t = 0.125$ mm thickness. We considered the lamination sequence $[45/90/-45/0]_S$ for a total of 8 plies and 1 mm laminate thickness.

A.2. DETAILS OF THE FE MODELS

All finite element models were produced with the Abaqus finite element code [1]. In addition, a Python code was used to automatically generate different FE models for each strain loading combination [2, 3].

The meshed part is illustrated in Figure A.1, which shows that a radial mesh was obtained by seeding the hole edge 4 times as much as the outer edges. Since no delaminations were expected due to the absence of ply blocks, the elements were chosen to be S4 shells elements of the Abaqus Standard Element Library whose in-plane and bending behaviour are described by the classical lamination theory (CLT), once the stacking sequence and ply thicknesses are specified [4].

Damage initiation was modeled with the Hashin criterion, while damage evolution was represented in a smeared crack fashion. For this purpose, the cohesive law available in Abaqus was employed to model the stiffness degradation due to matrix and fiber tensile and compressive failure [4].

REFERENCES

[1] Dassault Systèmes. *ABAQUS Finite Element Analysis Software*. 2022.

[2] T. Gulikers. *Computational framework to implement an artificial neural network based constitutive model in Abaqus for mesh coarsening*. https://github.com/tgulikers/ABAQUS_ANN_constitutive_model. 2018.

[3] B. Chen. *Training ANNs with FEM data on Open Hole Composite plate - for summer school 2022 in Delft*. https://github.com/BoyangChenFEM/Summer2022. 2022.

[4] Dassault Systèmes. *ABAQUS Analysis User Manual*. 2022.

B

KERNELS - SUPPORT VECTOR MACHINES, KERNEL METHODS AND KTA

B.1. PRIMAL SVM

The SVM is the linear decision model

$$y = \boldsymbol{w}^\top \boldsymbol{x} + b \tag{B.1}$$

which assigns labels through the sign function

$$\operatorname{sgn}(y) = \operatorname{sgn}\left(\boldsymbol{w}^\top \boldsymbol{x} + b\right). \tag{B.2}$$

In Equations (B.1) and (B.2), $\boldsymbol{w}$ is the vector normal to the decision hyperplane and b is the intercept.

The optimal hyperplane is found by maximizing the *geometric margin* of the dataset, which can be proved to be

$$\gamma^* = \frac{1}{\|\boldsymbol{w}\|}. \tag{B.3}$$

By minimizing the squared norm $\|\boldsymbol{w}\|^2$ one obtaines the primal optimization problem of the SVM,

$$\begin{aligned} \min_{\boldsymbol{w}, b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{s.t.} \quad & y^{(m)}\left(\boldsymbol{w}^\top \boldsymbol{x}^{(m)} + b\right) \geq 1 \quad m = 1, \ldots, M, \end{aligned} \tag{B.4}$$

where m identifies the sample and M is the total number of training samples.

Equation (B.4) enforces exact separability, which can lead to overfitting. A way to improve generalization is the so-called *soft margin*

SVM, which modifies Equation (B.4) by introducing the constraints slack variables $\xi^{(m)}$ and the penalty constant C,

$$\begin{aligned} \min_{\boldsymbol{w}, b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{m=1}^{M}\xi^{(m)} \\ \text{s.t.} \quad & y^{(m)}\left(\boldsymbol{w}^\top \boldsymbol{\varepsilon}^{(m)} + b\right) \geq 1 - \xi^{(m)} \quad m = 1, \ldots, M \\ & \xi^{(m)} \geq 0. \end{aligned} \tag{B.5}$$

B.2. DUAL SVM AND KERNELS

By introducing the Lagrange multipliers $\alpha^{(m)}$ and $\beta^{(m)}$, one can write the Lagrangian of the SVM optimization problem,

$$\begin{aligned} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = & \frac{1}{2}\langle \boldsymbol{w}, \boldsymbol{w}\rangle + C\sum_{m=1}^{M}\xi^{(m)} \\ & - \sum_{m=1}^{M}\alpha^{(m)}\left(y^{(m)}\left(\boldsymbol{w}^\top \boldsymbol{x}^{(m)} + b\right) - 1 + \xi^{(m)}\right) - \sum_{m=1}^{M}\beta^{(m)}\xi^{(m)}. \end{aligned} \tag{B.6}$$

The *dual* soft-margin SVM is obtained by setting all the derivatives of the Lagrangian in Equation (B.6) equal to zero,

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{m=1}^{M}\alpha^{(m)} - \frac{1}{2}\sum_{m,m'=1}^{M} y^{(m)}y^{(m')}\alpha^{(m)}\alpha^{(m')}\langle \boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m')}\rangle \\ \text{s.t.} \quad & 0 \leq \alpha^{(m)} + \beta^{(m)} \leq C, \quad m = 1, \ldots, M \\ & \sum_{m=1}^{M}\alpha^{(m)}y^{(m)} = 0. \end{aligned} \tag{B.7}$$

Equation (B.7) is still a linear model in the original feature space. However, by introducing a feature map

$$\phi : \boldsymbol{x} \longrightarrow \phi(\boldsymbol{x}) \tag{B.8}$$

we can map the features nonlinearly and potentially to a manifold where they are more easily separable. Furthermore, replacing $\boldsymbol{x}$ with $\phi(\boldsymbol{x})$ in Equation (B.7), we see that the mapped features only appear in the inner product

$$\kappa\left(\boldsymbol{x}^{(m)}, \boldsymbol{x}^{(m')}\right) = \langle \phi(\boldsymbol{x}^{(m)}), \phi(\boldsymbol{x}^{(m')})\rangle, \tag{B.9}$$

which is known as the *kernel* of the feature map. The advantage of having only inner product of features (*kernel trick*) is the possibility of classifying in nonlinear feature spaces without having to compute the feature map explicitly.

The kernels mostly used in machine learning are the polynomial, Gaussian and sigmoid kernels

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \begin{cases} (\gamma \boldsymbol{x}^\top \boldsymbol{x}' + c_0)^d & \text{(polynomial)} \\ \exp\left(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|^2\right) & \text{(Gaussian)} \\ \tanh\left(\gamma \boldsymbol{x}^\top \boldsymbol{x}' + c_0\right) & \text{(sigmoid)} \end{cases} \tag{B.10}$$

B.3. KERNEL-TARGET ALIGNMENT

The *alignment* between two kernels is defined as

$$\begin{aligned} A\left(K^{(1)}, K^{(2)}\right) &= \frac{\langle K^{(1)}, K^{(2)} \rangle_F}{\sqrt{\langle K^{(1)}, K^{(1)} \rangle_F \langle K^{(2)}, K^{(2)} \rangle_F}} \\ &= \frac{\langle K^{(1)}, K^{(2)} \rangle_F}{\|K^{(1)}\|_F \|K^{(2)}\|_F}, \end{aligned} \tag{B.11}$$

where K is the *kernel matrix*, obtained by taking the kernel of all pairs of features, and

$$\langle K^{(1)}, K^{(2)} \rangle_F = \operatorname{tr}\left(K^{(1)\top} K^{(2)}\right).$$

The alignment between two kernels is always lesser or equal to 1, where 1 corresponds to perfect alignment.

Assume a kernel $\kappa_{\boldsymbol{\theta}}$, parametrized by $\boldsymbol{\theta}$ and define the *target kernel* matrix as

$$K^* = \mathbf{y}\mathbf{y}^\top. \tag{B.12}$$

The *kernel-target alignment* (KTA) of $\kappa_{\boldsymbol{\theta}}$ is the alignment between the chosen kernel and the target,

$$\begin{aligned} A\left(K_{\boldsymbol{\theta}}, K^*\right) &= \frac{\langle K_{\boldsymbol{\theta}}, K^* \rangle_F}{\|K_{\boldsymbol{\theta}}\|_F \|K^*\|_F} \\ &= \frac{\langle K_{\boldsymbol{\theta}}, K^* \rangle_F}{M \|K_{\boldsymbol{\theta}}\|_F}, \end{aligned} \tag{B.13}$$

where $K_{\boldsymbol{\theta}}$ is the kernel matrix of $\kappa_{\boldsymbol{\theta}}$.

The KTA enjoys theoretical properties such as concentration around its expected value and generalisation [1] and therefore it is indicative of the ability of a kernel to separate classes of data.

REFERENCES

[1] T. Wang, D. Zhao, and S. Tian. “An overview of kernel alignment and its applications”. In: *Artif. Intell. Rev.* 43.2 (Feb. 2015), pp. 179–192.

C

KERNELS - COMPARISON ON FURTHER CLASSIFICATION METRICS

N_{train}	Accuracy	Jaccard Index	Precision	Recall	Specificity
RBF kernel					
156	0.694	0.627	0.784	0.759	0.795
313	0.750	0.708	0.835	0.824	0.838
470	0.788	0.751	0.886	0.832	0.895
627	0.788	0.750	0.893	0.825	0.903
784	0.838	0.813	0.939	0.859	0.944
940	0.824	0.797	0.915	0.861	0.921
1097	0.848	0.827	0.938	0.874	0.943
1254	0.878	0.866	0.943	0.913	0.945
1411	0.873	0.860	0.937	0.912	0.939
1568	0.882	0.869	0.955	0.906	0.958
HE2W6D3 kernel					
156	0.699	0.626	0.804	0.739	0.823
313	0.731	0.675	0.835	0.780	0.846
470	0.756	0.708	0.858	0.802	0.869
627	0.779	0.741	0.878	0.826	0.887
784	0.795	0.762	0.892	0.839	0.900
940	0.794	0.757	0.897	0.829	0.907
1097	0.805	0.773	0.902	0.844	0.910
1254	0.797	0.765	0.884	0.849	0.891
1411	0.814	0.785	0.911	0.851	0.917
1568	0.818	0.790	0.912	0.855	0.919

Table C.1.: Comparison of five different classification scores between the RBF-SVM and the trained HE2W6D3 quantum kernel SVM. The classifiers were trained with increasing fractions of the training dataset. Notice that the RBF-SVM problem used $C = 10^7$, while the HE2W6D3 quantum kernel SVM used $C = 10^4$, which is the highest C values before the occurrence of convergence issues.

ACKNOWLEDGEMENTS

Starting and finishing this Ph.D. trajectory would have not been possible without the support of many.

I start by thanking my supervisor Boyang Chen, who, in a café in Delft back in 2019, pitched me the idea of looking into quantum computing for speeding up the finite element method. Although we knew barely anything of the subject, we humbly started learning and finally concretized a research proposal. Boyang pushed the idea in the department and met both interest and skepticism.

Fortunately, my promotor Roeland de Breuker, was one of the interested ones. He supported the project and fought to find the necessary funding. I deeply thank him for that.

My other promotor, Matthias Möller, joined us soon after the start of the project. Even though quantum computing was, at the time, mostly a subject for physicists and theoretical computer scientists, Matthias had an eye for realistic, near-term and application-oriented QC. This made him the right person at the right time for our team, and I wish to thank him for taking the bet with us.

I also thank Marc Gerritsma, who was not officially, but effectively part of my Ph.D. team. Marc jumped on the 'quantum train' with us in 2020 and brought two bold M.Sc. students along for the ride, Enrico and Thomas. After they graduated, Marc continued following my trajectory with interest and dispensed precious advice.

My Ph.D. was a bumpy road. Some fallbacks were enough to undermine my confidence and question my career choices. I have my family to thank if I kept walking this path until the end.

I thank my *Liefie*, Mayon, who is the love of my life. She accepted this half-disillusioned Ph.D. student in her life and, with her kindness and patience, helped him to the finishing line.

I thank my Mom and Dad for always supporting me and for knowing me well enough to give me a (gentle) kick in the lower back when I needed it. I dedicate this thesis and this milestone to them, for giving me the chance to become who I am today.

A special 'thank you' goes to Feike and Anita, my parents in law. By having me as their guest in Leutingewolde, they kept me away from distractions and allowed me to write my thesis in one big push. I will always cherish how those days really brought us together.

I thank Alberto and Sergio, who are my paranymphs during the defence, and who are like brothers to me. We shared together the joys,

the intensity and the exhaustion of many climbs, mountain-hikes and via ferratas. Hearing of the ‘mud slide’ or of the *Große Kinigat* ferrata still makes our backs shiver and our eyes sparkle.

Finally, I thank my colleagues of AS(C)M, especially Kevin and Luc, for all the good times together. What would a Ph.D. be without a beer at the *Atmosfeer*, while cursing the academic struggles?

CURRICULUM VITæ

Giorgio Tosti Balducci

17-10-1992	Born in Grosseto, Italy.

EDUCATION

2006–2011	Grammar School Liceo Classico Ricasoli, Grosseto (IT)
2012–2016	BSc in Aerospace Engeineering Politecnico di Milano
2016–2019	MSc in Aerospace Structures and Materials Technische Universiteit Delft
2019–2025	PhD. Aerospace Engineering Technische Universiteit Delft *Thesis:* About the intersection of quantum computing and computational structural mechanics *Promotors:* Prof. dr. ir. R. De Breuker & Dr. rer. nat. M. Möller

INDUSTRY WORK

2017–2018	ATG Europe, Noordwijk (NL)
2024	Pasqal, Amsterdam (NL)

LIST OF PUBLICATIONS

5. G. Tosti Balducci, B. Chen, M. Möller, M. Gerritsma, and R. De Breuker. “Review and perspectives in quantum computing for partial differential equations in structural mechanics”. In: *Front. Mech. Eng.* 8 (Sept. 2022), p. 914241

4. G. Tosti Balducci, B. Chen, M. Möller, M. Gerritsma, and R. D. Breuker. *Predicting Open-Hole Laminates Failure Using Support Vector Machines With Classical and Quantum Kernels*. 2024. arXiv: 2405.02903 [cs.CE]. url: https://arxiv.org/abs/2405.02903

3. G. Tosti Balducci, B. Chen, M. Möller, and R. D. Breuker. *Solving 1D Poisson problem with a Variational Quantum Linear Solver*. 2024. arXiv: 2412.04938 [cs.CE]. url: https://arxiv.org/abs/2412.04938

2. Z. Kaseb, M. Möller, G. Tosti Balducci, P. Palensky, and P. P. Vergara. “Quantum neural networks for power flow analysis”. In: *Electric Power Systems Research* 235 (2024), p. 110677. issn: 0378-7796

1. G. Tosti Balducci and B. Chen. “Overcoming the cohesive zone limit in the modelling of composites delamination with TUBA cohesive elements”. In: *Composites Part A: Applied Science and Manufacturing* 185 (2024), p. 108356. issn: 1359-835X