An Exploration on Quantum Computing for Solving 1-Dimensional Non-Linear Viscous Burgers' Equation

Arnau Moreno Sánchez

Lund University

Spring 2026

Supervisor: Rixin Yu

Supervisors from RISE: Erdzan Hodzic, Anastasiia Andriievska

Department of Energy Sciences

ISRN: LUTMDN/TMHP-26/5705-SE

ISSN: 0282-1990

Contents

Acknowledgements

I would firstly like to thank my supervisor Rixin for his advice, feedback and support along this journey.

I would also like to express my gratitude to Erdzan and Anastasiia from RISE, as their continuous support and help were vital to achieving this thesis.

This would not have been possible without the constant support of my companion Miss Hu (aka integer overflow) and the fantastic friends I had the pleasure to spend time with during these two years: Diego, Pedro, Edu, Uljei, Veronica, Alexandros, Pepe... And I do not forget the friends that already had to come back home.

Per acabar, vull donar les gràcies als meus pares, l'Enric i la Glòria, i a la meva germana, la Berta, per donar-me aquesta oportunitat. Sense el seu suport i la seva ajuda, ara no seria aquí vivint aquestes experiències tan meravelloses.

An Exploration on Quantum Computing for Solving 1-Dimensional Non-Linear Viscous Burgers' Equation

Arnau Moreno Sánchez

July 3, 2026

Abstract

As the energy sector transitions towards more sustainable forms of energy production, classical Computational Fluid Dynamics (CFD) increasingly struggles with the bottlenecks of simulating high complexity turbulent flows. While quantum computing (QC) theoretically offers an exponential speedup for large-scale linear systems, current Noisy Intermediate-Scale Quantum (NISQ) hardware lacks the error correction required for deep and complex circuits that mathematically ideal algorithms require. Furthermore, the inherent linearity of quantum mechanics seriously impedes the solving non-linear partial differential equations (PDEs), which are critical for fluid modeling. To bridge this gap, this thesis evaluates the Variational Quantum Linear Solver (VQLS), a shallow-circuit, hybrid algorithm, applied to the 1D viscous Burgers' equation while utilizing the Cole-Hopf transformation to analytically linearize the equation. In both simulated and actual hardware, the VQLS algorithm is implemented across increasingly scaled topographies (2, 4, and 8 qubits). This work intends to identify the limitations of implementing quantum fluid dynamics in near-term quantum hardware.
Keywords: *CFD, QC, NISQ, VQLS, Burgers' Equation, Cole-Hopf Transformation*

I. INTRODUCTION

In the current world, the transition towards a more sustainable form of producing energy has become fundamental. The objective of turning away from fossil fuels hides different challenges that the engineering field has been trying to solve for a long time. A really important field of study is fluid dynamics, where systems are optimized whether for getting the most power from wind turbines, simulating complex fluid flows across tubes or analyzing the airflow behavior around an aircraft [1]. The optimal design of these systems relies entirely on proper fluid control. The currently most popular approach for this kind of design is through Computer Fluid Dynamics (CFD), which allows to numerically solve the Navier-Stokes equations and other related non-linear partial differential equations (PDEs) that are fundamental for studying the behavior of fluids.

However, as systems become more and more complex, the classical CFD approach is getting closer towards a bottleneck. In order to simulate high-complexity, chaotic, and turbulent systems, the simulation domain needs to be discretized into massive grids with multiple dimensions, especially for 3D simulations [2]. As the resolution of simulations becomes larger, the computational cost increases exponentially. As a consequence, simulating highly-complex 3D turbulent flows, even on supercomputers, can take several days.

In order to surpass this barrier, quantum computing (QC) offers a highly promising framework when it comes to designing these complex systems. By mapping classical data into quantum qubits, there is an exponential increase in speed when solving large-scale linear systems of equations thanks to the ability of manipulating probability amplitudes simultaneously [3]. Instead of solving node-by-node as classical computer would do, a QC is able to reduce CFD matrices into easily solvable quantum states, thus theoretically providing a massive speed-up when it comes to simulate these large energy systems.

Yet, the gap between the theoretical quantum physics and the current usable hardware is still too large. The current era in quantum computing is the Noisy Intermediate-Scale Quantum era (NISQ), where QC have around 50-250 stable qubits. However, the noise between these qubits considerably affects the performance of quantum algorithms [4]. This limitation makes the usage of mathematically ideal algorithms such as Harrow-Hassidim-Lloyd (HHL)

and Quantum Singular Value Transformation (QSVT) impossible to run nowadays as they require deep quantum circuits with extreme error correction [5, 6]. This is where alternative hybrid algorithms such as the Variational Quantum Linear Solver (VQLS) [6] take the advantage. VQLS replaces deep, pure-quantum circuits, with shallow circuits that employ hybrid machine learning optimizers in order to save qubits.

The objective of this thesis is to evaluate the viability of the VQLS algorithm when it comes to non-linear fluid dynamics, in this specific case, the 1D viscous Burgers' equation. As quantum mechanics is inherently linear, solving non-linear PDEs still remains a challenge [7]. To go around this problem, Cole-Hopf Transformation and its reverse form are used to linearize the equation, allowing it to be mapped into the VQLS architecture. Moreover, the algorithm is going to be tested in different qubit topographies (2, 4 and 8 qubits) in both simulated and real hardware environments. This allows to track the convergence, hardware scaling capabilities and noise robustness of the algorithm for non-linear cases.

II. BACKGROUND

In order to properly understand how quantum mechanics can be utilized to simulate fluid dynamics, it is important to properly understand the differences between ordinary classical computing and quantum information processing. While classical computers work by using a series of logical gates that are deterministic, quantum computers operate by using the principles of quantum mechanics: linear algebra and probability.

II.A. Fundementals of Quantum Computing

II.A.i. Qubits and Superposition

Classical computers work by storing the information and data into bits, which use binary in order to give a deterministic state of either 0 or 1, whereas in quantum computers, the information is stored in quantum bits, also known as qubits. Qubits are the fundamental unit of information in quantum computing, which can be seen as 2-state system, such as an atom with 2 energy levels [8].

Qubits differ from bits as they are not restricted to a single binary value; qubits can be

both 0 and 1 at the same time as long as they are not measured. This is because qubits are in a mathematical state of what is called superposition [8], which means that it is described as a linear combination of two quantum states $|0\rangle$ and $|1\rangle$, represented in Dirac notation or "braket".

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

When looking at a single qubit, its state can be represented as $|\psi\rangle$, as seen in equation (2). Here, α and β represent the probability amplitudes of measuring each of the two different states with the form of complex numbers. In other words, it can be said that the qubit is not both 0 and 1 at the same time, but is instead in a state represented by a probability distribution. The distribution is said to "collapse" into one of the states $|0\rangle$ or $|1\rangle$ after measurement.

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2}$$

The squared magnitudes of these probability amplitudes, $|\alpha|^2$ and $|\beta|^2$, are the exact probabilities of collapsing into 0 or 1 when measuring the system. Because quantum mechanics is probabilistic, both of the probabilities must add up to 100% as bound by the following: $|\alpha|^2 + |\beta|^2 = 1$.

Quantum circuits can also contain multiple qubits. These different qubits can be represented individually, or together in the same ket by applying the tensor product between both. Observe the following example with two $|0\rangle$ states in equation (3):

$$|0\rangle \otimes |0\rangle = |00\rangle \tag{3}$$

II.A.ii. Entanglement and Exponential State Space

When having multiple qubits, a quantum computer is able to make the different qubits interact with each other through a phenomenon called entanglement. Entangled qubits are capable of communicating between each other as they become mathematically linked by

making use of certain quantum gates [8, 9]. It can be said that a state is entangled if it cannot be factored into the tensor product of individual qubits as the measurement of a qubit determines the state of the qubit that is entangled with it.

A classical computer with N bits is able to represent one state at a time out of 2^N possibilities. On the other hand, a quantum computer with N entangled qubits is able to represent all the 2^N probability amplitudes at the same time, thus creating a large multidimensional vector space [8]. If the cases with 2, 4 and 8 qubits are observed, the state spaces would be 4, 16 and 256 dimensions respectively. Moreover, the usage of quantum logic gates enables the quantum processor to process the information of all dimensions in parallel as the qubits are then linked with each other. This phenomenon of processing all the amplitudes at the same time is the key that enables quantum speed-up when compared to classical computing.

II.A.iii. Quantum Gates, Bloch Sphere and Circuits

In order to perform the necessary computations, the probability amplitudes of each qubit must be manipulated. Classical computers make use of logical gates (e.g., NOT, AND, OR, XOR...) that can be irreversible in many cases. This means that the information is lost during the operation; it is not possible to reverse the operation back to its input state. On the other hand, quantum computers take advantage of quantum gates, which are represented by a series of unitary matrices ($U^\dagger U = I$) that assure reversibility, a critical condition required in quantum computation [8, 9].

A good way to properly understand the manipulations performed by quantum gates is through the Bloch Sphere (Figure I). It is an intuitive way to visualize the quantum state of a single qubit by representing it as a unit vector inside a sphere [9]. It is important to mention that, apart from the two quantum states $|0\rangle$ and $|1\rangle$, there exist other states of superposition that contain both real and imaginary probability amplitudes: the $|\pm i\rangle$ (4) and the $|\pm\rangle$ (5) states. Those points in the sphere represent all the possible superposition states for all possible values of complex numbers [9]. Therefore, the Bloch Sphere has three axes, being Z the axis where both $|0\rangle$ and $|1\rangle$ are represented, and X and Y where the $|\pm\rangle$ and $|\pm i\rangle$ superposition states sit. The position of the state in the sphere can be represented by making use of the "relative phase" form as seen in equation (6), where θ dictates the

probability of measuring 0 or 1, and ϕ dictates the position around the equator, i.e. the phase. Because measurement probabilities rely on the absolute square of the amplitudes, the global phase mathematically cancels out and has no observable physical effect. The relative phase ϕ, however, dictates quantum interference and is therefore preserved as the azimuthal angle on the Bloch sphere. Changing ϕ does not alter the probabilities of measuring 0 or 1, thus, if ψ is at the equator, the probability is 50% for each one.

$$|\pm i\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm i|1\rangle) \tag{4}$$

$$|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \tag{5}$$

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle \tag{6}$$

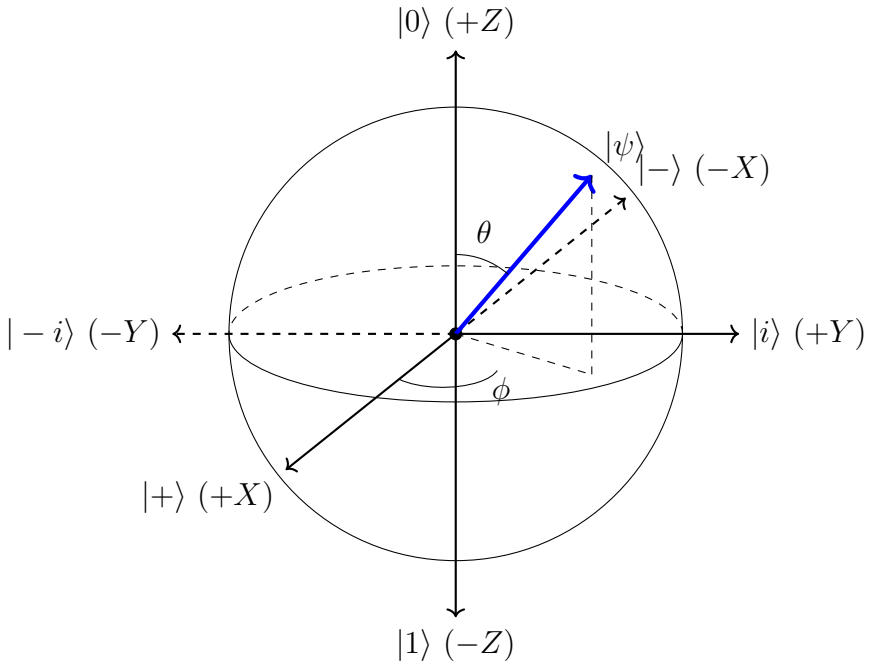


Figure I: Geometric representation of a single qubit state $|\psi\rangle$ on the Bloch Sphere.

Some examples of quantum gates are Pauli gates (X, Y, Z), which are able to rotate the qubit 180^{o} around the X, Y and Z axis of the Bloch Sphere, as seen in equation (7). The Controlled-NOT gate, or CNOT, operates on two qubits, a control and target qubit; see

equation (8). It flips the target qubit if and only if the control qubit is $|1\rangle$. It is a gate of great importance as it acts as a mechanism for creating quantum entanglement. In addition to the Pauli gates, there exists a customizable version of them, which are the rotation gates. See equation (9) for an example of the R_y gate, which allows to rotate the qubit to any arbitrary angle θ around the Y axis.

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{7}$$

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{8}$$

$$R_y(\theta) = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \tag{9}$$

Looking at Figure II, a simple 2 qubit example with some Pauli gates and a CNOT gate applied to both qubits can be observed. Quantum circuits consist of coherent operations applied to a quantum register. The circuit represents each qubit as a wire where quantum logical gates are applied, and it is read from left to right. At the farthest left, the initial state of the qubit is shown; in this case, both qubits start at state $|0\rangle$. Along the length of the quantum circuit, the different stages of transformation are signaled with $|\psi_i\rangle$, marking the three different states from start to finish. Finally, the state $|\psi_2\rangle$ marks the final state for this case, which is represented in the circuit with a measurement box.

When seeing the mathematical development of this example in the next series of equations, it can be observed how the quantum gates are applied to each qubit in order of appearance.

The initial state $|\psi_0\rangle$ is represented as a 4×1 column vector, both qubits starting at ground state $|0\rangle$:

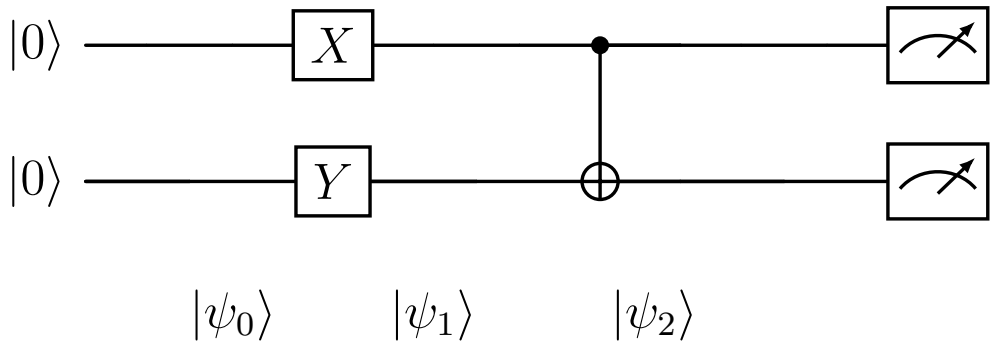


Figure II: Two-qubit quantum circuit demonstrating Pauli-X and Y gates alongside a CNOT entangling gate.

$$|\psi_0\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \left.\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}\right\} \begin{matrix} |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{matrix} \tag{10}$$

Next, the X and Y gates are applied via the tensor product to form a 4×4 operator matrix, which is multiplied by the state vector to find $|\psi_1\rangle$:

$$|\psi_1\rangle = (X \otimes Y)|\psi_0\rangle = \left[\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \right] \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \\ 0 & -i & 0 & 0 \\ i & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ i \end{pmatrix} = i|11\rangle \tag{11}$$

Finally, the CNOT gate acts on $|\psi_1\rangle$:

$$|\psi_2\rangle = \text{CNOT}|\psi_1\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ i \\ 0 \end{pmatrix} = i|10\rangle \tag{12}$$

II.A.iv. Measurement and Hardware Bottleneck

Once the system runs through the quantum circuit, a distribution of probabilities for each of the resulting states at the end of the circuit is given. However, according to the laws of physics, when a quantum system is measured, the property of superposition automatically disappears and the system collapses into a single state [9]. The complex multidimensional quantum vector collapses into a single string of binary digits. This resulting state will be dictated by the probability distribution that ends up at the final step of measuring but, even though some specific state has a much larger probability than another, other not so correct states can be given as the ending result.

Because of this inherent nature of quantum systems, in order to reconstruct complex fluid dynamics distributions, it is necessary to perform a large number of runs through the quantum circuit. These repeated executions are also known as "shots", and they are necessary in order to ensure that the best possible state output is given. Moreover, the current era of quantum processors is the Noisy Intermediate-Scale Quantum Era, which has some important bottlenecks and limitations that should be considered [4]. Qubits in the present era's processors are highly sensitive to thermal interference and electromagnetic noise. This phenomenon can cause changes in the probability amplitude values, thus altering the expected results. In consequence, deep circuits are more susceptible to interference as the amount of qubits interacting with each other is higher, as well as the longer time of interaction. Current quantum processors face a challenge in the form of a trade-off between qubit isolation versus interaction, as qubits require strong isolation to give accurate results, but at the same time they must be able to interact with each other to ensure proper entanglement.

II.B. CFD vs. QC and Quantum Algorithms for Linear Systems

In order to solve large-scale linear systems of equations, which constitutes the core of CFD, some of the most popular quantum algorithms that are currently known rely on solving an $Ax = b$ equation, given a matrix A and a vector b. Some approaches on classical CFD also predict fluid flow by solving the Navier-Stokes equations with this sort of linear system solving[10], which is critically limited by computational cost when performing highly complex

3D calculations as the grid resolution increases. Classical computers are able to solve these problems in a polynomial scale of time whereas quantum computers can theoretically solve them in logarithmic time, thus providing an exponential advantage [3, 11]. However, this speed up is only achieved in computationally difficult cases such as fully developed turbulence at large Reynolds number [11], where classical CPUs really struggle. Because of these possible advantages in the near future, the development of a scalable quantum computer capable of solving these complex cases could potentially save millions of dollars in the aeronautics and energy industry [10].

Another known characteristic about QC is its inherent linearity as it works according to the strictly linear laws of quantum mechanics. However, many governing equations in the fields of fluid dynamics, weather or gravity are non-linear. To tackle this issue, researchers have found ingenious ways to bring these non-linear equations into a linear framework, such as Carleman Linearization [12], Hamiltonian simulation [13], or by alternative variational algorithmic methods [14]. For the case covered in this thesis, the 1D viscous Burgers' equation can be linearized in a simple way by using the Cole-Hopf Transformation, which eliminates the non-linear convective term of the equation and transforms it into the classic 1D heat equation.

The development of Quantum Linear Systems Algorithms (QLSA) has been a topic of investigation among the experts in this field for years, and has progressed through different mathematical generations. Here are some of the most recognized examples of these kind of algorithms:

II.B.i. Harrow-Hassidim-Lloyd (HHL)

Proposed in 2009, HHL algorithm [15] was the first one that could prove a exponential speed up of quantum computers against classical supercomputers when it comes to solving linear systems. This algorithm works by encoding the vector b into a quantum state, then extracting the eigenvalues (scaling factor λ so that $Av = \lambda v$) of matrix A into the quantum register by using a technique called Quantum Phase Estimation (QPE). This allows to invert those eigenvalues so that it resembles a matrix inversion to calculate $A^{-1}|b\rangle$.

Even though HHL was a massive breakthrough in quantum research, its requirement

on the number of qubits makes it unusable on current hardware (for large and complex problems). HHL needs a massive number of completely error-corrected qubits as it is an algorithm that requires a deep circuit due to the necessity of needing ancillary qubits to store quantum information. Using this algorithm in current NISQ computers would cause decoherence before the end of the calculation.

II.B.ii. Quantum Singular Value Transformation (QSVT)

QSVT [16] represents a modern and more optimal quantum framework for algorithms that was presented in 2019. It works by embedding the classical matrix A into a bigger matrix, through a process called block-encoding, to make it unitary. This process eliminates the need of using QPE as specific phase shifts are applied directly to the matrix A by going back and forth, thus "inverting" it directly just by changing its singular values.

Even though this framework presents advantages in terms of optimization compared to HHL, calculating the phase shift the angles that are needed requires heavy preprocessing. Moreover, the circuit needed is still too deep for today's hardware.

II.B.iii. Variational Quantum Linear Solver (VQLS)

Due to the hardware-related issues present in the previous algorithms, the quantum industry has heavily focused on finding alternatives that are more feasible for the current technology while waiting for better capabilities. In particular, hybrid classical-quantum algorithms have gained more importance.

Proposed in 2023, the VQLS algorithm [6] tackles the depth requirement issue by, instead of performing a pure-quantum mathematical inversion of matrix A, using a hybrid approach that adjusts the parameters of the quantum gates in the circuit until a cost function is reduced as low as possible. These parameters are tweaked by using an optimizer working on classical computing that iteratively adjusts them until the solution vector x is represented accurately. This algorithm architecture makes only use of simple R_y and $CNOT$ quantum gates and does not require extra qubits to store information, therefore being significantly shallower than the other two algorithms mentioned previously. However, by not being purely quantum as it requires the use of classical computational resources, the calculation speed is significantly

slower than both HHL and QSVT. Because of the simplicity and more realistic application of this algorithm in the current available hardware, this thesis focuses in evaluating the viability, scalability and noise constraints of this particular algorithm applied to non-linear fluid dynamics.

A significant advantage of this VQLS implementation is its exclusive reliance on R_y rotations and CNOT gates. Because the linearized Burgers' equation only makes use of strictly real-valued amplitudes (no complex fluid velocities), the R_y gate provides sufficient rotational expressivity across the real plane without needing complex phase gates. Paired with CNOT gates to generate entanglement, this drastically reduces the hardware noise overhead.

III. EXPERIMENTAL DESIGN

III.A. Physical Model: Burgers' Equation

In order to evaluate the performance of NISQ hardware when it comes to solving partial differential equations, this experiment utilizes the viscous Burgers' equation as its baseline. Becoming widely popular in 1948 and named after Johannes Martinus Burgers, the equation is a fundamental non-linear partial differential equation that describes the mathematical modeling of fluid flow and turbulence in CFD, as it captures the non-linearity of convective and diffusive effects. It is commonly used to represent shockwave formation [17].

The standard 1D viscous Burgers' equation (13) is defined by:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = v\frac{\partial^2 u}{\partial x^2} \tag{13}$$

with u being the fluid velocity, x the spatial coordinate, t time, and v the kinematic viscosity of the fluid.

III.A.i. Linearization

Because of the already mentioned necessity of fitting into the linearity requirements of quantum mechanics, it is necessary to linearize the Burgers' equation as it has a non-linear convective term, $u(\frac{\partial u}{\partial x})$, that cannot be treated by the quantum environment. This is done through

the application of the Cole-Hopf Transformation, which linearizes the non-linear Burgers' equation by converting it into the linear heat equation.

The Cole-Hopf Transformation (14) is defined by:

$$u(x,t) = -2\nu \frac{\partial}{\partial x} \ln \phi(x,t) = -\frac{2\nu}{\phi} \frac{\partial \phi}{\partial x} \tag{14}$$

When substituting the transformation into the original equation, it is then transformed into the classical linear heat equation (15):

$$\frac{\partial \phi}{\partial t} = \nu \frac{\partial^2 \phi}{\partial x^2} \tag{15}$$

III.A.ii. Discretization

After linearizing the model into the heat equation, it is necessary to prepare the inputs for the algorithm. As computers are not able to understand continuous space, the heat equation needs to be discretized. Finite Difference Method (FDM) is used in order to do so, as it is one of the common and established numerical methods for discretizing PDEs [17]. With Central Differencing the spatial part of the equation is discretized in equation (16), and the temporal discretization is done through Backward Euler in equation (17).

$$\left.\frac{\partial^2 \phi}{\partial x^2}\right|_i^{m+1} \approx \frac{\phi_{i-1}^{m+1} - 2\phi_i^{m+1} + \phi_{i+1}^{m+1}}{(\Delta x)^2} \tag{16}$$

$$\left.\frac{\partial \phi}{\partial t}\right|_i^{m+1} \approx \frac{\phi_i^{m+1} - \phi_i^m}{\Delta t} \tag{17}$$

Substituting both terms back in the original equation (18) and gathering the constant values into a dimensionless constant r in equation (19), it is possible then to rearrange the equation by putting the unknown next time-step terms to the left and the known current time-step term to the right in equation (20).

$$\frac{\phi_i^{m+1} - \phi_i^m}{\Delta t} = \nu \left(\frac{\phi_{i-1}^{m+1} - 2\phi_i^{m+1} + \phi_{i+1}^{m+1}}{(\Delta x)^2} \right) \tag{18}$$

$$r = \frac{\nu \Delta t}{(\Delta x)^2} \tag{19}$$

$$-r\phi_{i-1}^{m+1} + (1+2r)\phi_i^{m+1} - r\phi_{i+1}^{m+1} = \phi_i^m \tag{20}$$

This equation provides the diagonal and off-diagonal values of the diagonal-dominant sparse matrix A, where the main $1+2r$ diagonal represents the thermal inertia of the node, and the off-diagonals $-r$ represent the diffusion. For the case of this experiment, an $r = 0.2$ is used with a kinematic viscosity of $\nu = 0.1$. Furthermore, to optimize the matrix behavior in the quantum environment, an artificial stabilization term $\epsilon = 0.1$ has been added to the main diagonal so the matrix condition number is lowered, thus improving the performance of the optimizer. On the other hand, vector b is a Gaussian Bell curve that represents the known previous state that acts as the initial state, a simple initial condition to test the algorithm performance. From this, it is possible to fill in the terms of the equation to be solved by the algorithm, $Ax = b$, where x is the updated point of the fluid grid that is wanted to be found as seen in equation (21). The amount of spatial nodes is going to be determined by the number of qubits in the circuit, $N = 2^n$, where n is the amount of qubits. Therefore, vectors b and x are going to have a 2^n length, and matrix A a square shape of $2^n \times 2^n$. The discretized matrix A forms a well-posed linear system due to its strict diagonal dominance. With the addition of the artificial stabilization term ($\epsilon = 0.1$), the absolute value of the main diagonal exceeds the sum of the absolute values of the off-diagonal elements ($|1.5| > |-0.2| + |-0.2|$). This property mathematically guarantees that the matrix has a unique solution while having a low condition number to guarantee numerical stability.

$$\underbrace{\begin{pmatrix} (1+2r)+\epsilon & -r & 0 & \cdots & 0 \\ -r & (1+2r)+\epsilon & -r & \cdots & 0 \\ 0 & -r & (1+2r)+\epsilon & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & -r & (1+2r)+\epsilon \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} \phi_1^{m+1} \\ \phi_2^{m+1} \\ \phi_3^{m+1} \\ \vdots \\ \phi_{N-1}^{m+1} \end{pmatrix}}_{x} = \underbrace{\begin{pmatrix} \phi_1^{m} \\ \phi_2^{m} \\ \phi_3^{m} \\ \vdots \\ \phi_{N-1}^{m} \end{pmatrix}}_{b} \tag{21}$$

For the time advancement of this system, the initial condition at time t_0 is defined as a Gaussian distribution, giving an initial heat state of $\phi(x, t_0) = \exp(-3x^2)$ across the spatial domain, which is then normalized and mapped into the initial boundary vector b. The discretization matrix A is strictly truncated at the spatial limits as the simulation assumes "ghost nodes" on the left and right end of the spatial boundary from the matrix A, which have a value really close to 0, thus applying an "approximated" Dirichlet boundary condition right outside the physical domain. The solution is then advanced forward in time over a discrete interval $t_1 - t_0 = \Delta t$, allowing the algorithm to compute the updated state vector x representing $\phi(x, t_1)$. As the three different qubit versions use the same value of r for simplicity, the time step is different for each case, being the highest in the 2-qubit version (fastest time advancement) and the lowest in the 8-qubit version as having more qubits decreases Δx, thus making Δt decrease exponentially in order to comply with equation (19). It is important to point out that the utilization of a larger value of r, even though it would enlarge the time step Δt, would as well increase the matrix's condition number, which would clearly decrease the optimizer performance when navigating through the cost function landscape. Observe in Figure III how the top row depicts the initial conditions at t_0, showing the linearized Gaussian heat state ϕ (right) and the corresponding physical fluid velocity u (left) after delinearization. The bottom row displays the time-advanced states at t_1, where the heat state curve is visibly flattened due to thermal diffusion, and the fluid velocity produces sharp vertical gradients at the spatial limits caused by the fluid exiting the truncated boundaries.

III.B. Quantum Representation and Scalability

Considering that the 1D spatial fluid is discretized into N points, it can be said that the fluid heat state at each point i at a specific time step for n qubits can be represented by a classical column vector $\vec{\phi} = [\phi_0, \phi_1, \ldots, \phi_{N-1}]^T$ formed by all the amplitudes that is directly related to the base quantum state of the system of n qubits (where $N = 2^n$). By using what is known as amplitude encoding, the vector is directly mapped onto the amplitudes of a quantum state

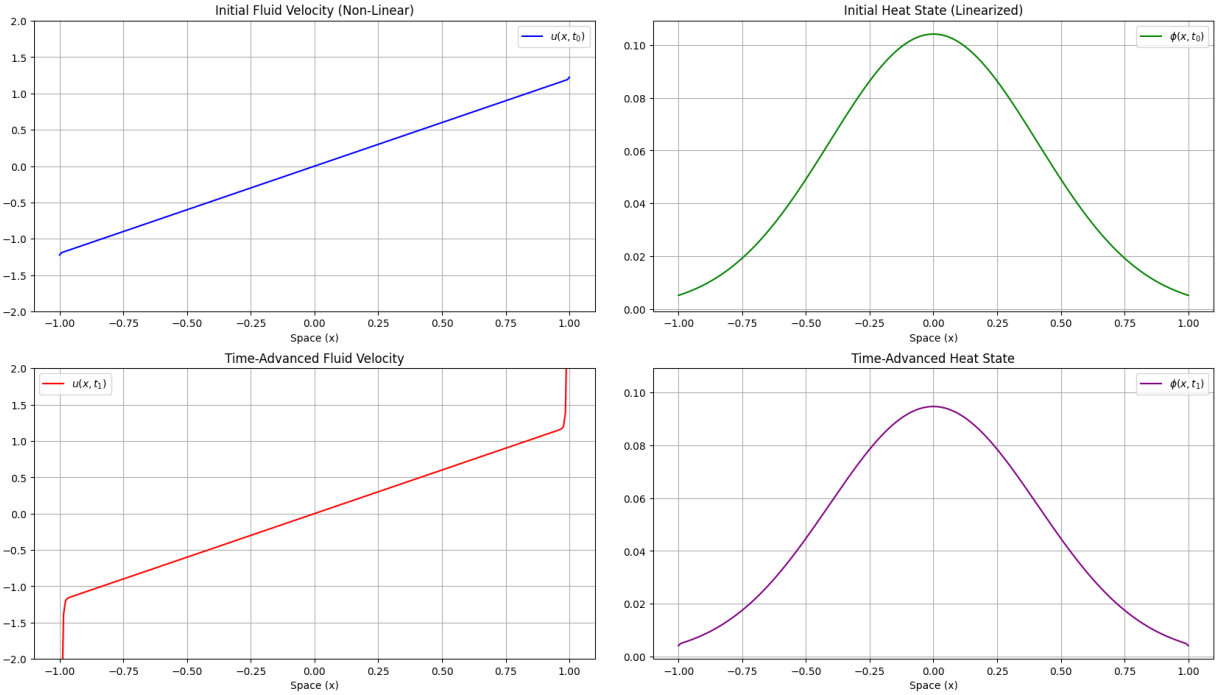


Figure III: Classical reference states for the spatial grid ($N = 256$).

Note: The spikes observed at the boundaries of the time-advanced velocity reference (bottom-left) are a numerical artifact. These are produced due to the steep gradient that is divided by a close to 0 ϕ value during inverse Cole-Hopf.

vector $|\psi\rangle$, as seen in equation (22).

$$|\psi\rangle = \frac{1}{||\vec{\phi}||} \sum_{i=0}^{2^n-1} \phi_i |i\rangle \tag{22}$$

where $||\vec{\phi}||$ is the normalization constant, defined by the 2-norm $||\vec{\phi}|| = \sqrt{\sum_{i=0}^{N-1} |\phi_i|^2}$, which ensures that the sum of all squared probabilities is equal to 1. Observing this equation, it is demonstrated that, in order to double the number of spatial grid points, it is only necessary to add a single qubit. This fact, thus, shows the main quantum advantage over classical computing.

In order to evaluate the performance of the VQLS algorithm for current hardware, the experiment scales across three different circuit sizes. This scaling process is going to be helpful in understanding how the algorithm behaves as more qubits are added, which in turn adds more depth and layers to the circuit. Taking into account the addition of qubits and usage of more quantum gates, it can be expected that the results could vary significantly for each case due to hardware noise and decoherence. The circuit architecture chosen for the experiment follows the guidelines of recent literature [5], where the circuit consist of a series of R_y columns followed by a $CNOT$ column directly afterwards, which both form a single layer. These allow to give the sufficient expressivity and entanglement to the qubits so the angles can be adjusted properly to fit the result. The scale of the ansatz increases with the number of qubits following the pattern of *Number of Layers* $= N - 1$ for the majority of cases (not applicable to the 2-qubit version because it would have not enough expressivity), which gives the best trade-off between qubit interaction and interference.

III.B.i. 2-Qubit Circuit

In this first set-up, the circuit consists of 2 qubits that will generate a 4 point spatial grid. Even though the spatial grid is way too coarse for a practical fluid analysis case, this configuration serves the purpose of verifying the circuit architecture. Because the circuit is shallow and has only 4 states, it is expected not to be significantly affected by hardware noise.

The 2-qubit architecture used for the experiment consists of 2 layers with a total of 4 parameters, as seen in Figure IV:

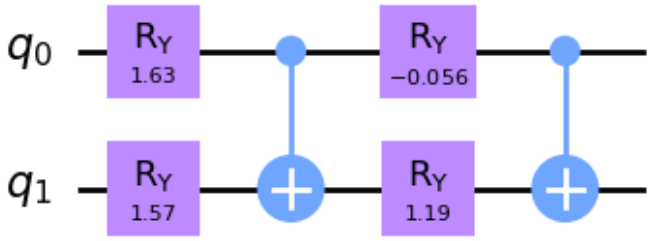


Figure IV: 2-Qubit Circuit architecture.

Note: The qubits q_i in this circuit collectively encode the amplitude representation of the transformed heat state variable ϕ.

The sequential composition of the matrices applied to the initial ground state $|00\rangle$ to encode the fluid state of this 2-qubit circuit can be explicitly formulated as in equation (23):

$$|\phi\rangle_{4\times1} = \underbrace{\underbrace{\underbrace{\underbrace{\text{CNOT}_{4\times4} \cdot \Big[R_y(-0.056) \otimes R_y(1.19)\Big]_{4\times4}}_{|\psi_3\rangle} \cdot \text{CNOT}_{4\times4} \cdot \Big[R_y(1.63) \otimes R_y(1.57)\Big]_{4\times4}}_{|\psi_2\rangle} \cdot \underbrace{\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}_{4\times1}}_{|\psi_0\rangle}}_{|\psi_1\rangle}}_{|\phi\rangle} \tag{23}$$

An expanded view of $|\psi_1\rangle$ with its full matrices can be observed in (24) for better clarity. Let $c_1 = \cos(0.815)$, $s_1 = \sin(0.815)$, $c_2 = \cos(0.785)$, and $s_2 = \sin(0.785)$:

$$
\begin{aligned}
|\psi_1\rangle &= [R_y(1.63) \otimes R_y(1.57)]\,|\psi_0\rangle \\
&= \left[\begin{pmatrix} c_1 & -s_1 \\ s_1 & c_1 \end{pmatrix} \otimes \begin{pmatrix} c_2 & -s_2 \\ s_2 & c_2 \end{pmatrix} \right] \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} c_1c_2 & -c_1s_2 & -s_1c_2 & s_1s_2 \\ c_1s_2 & c_1c_2 & -s_1s_2 & -s_1c_2 \\ s_1c_2 & -s_1s_2 & c_1c_2 & -c_1s_2 \\ s_1s_2 & s_1c_2 & c_1s_2 & c_1c_2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} c_1c_2 \\ c_1s_2 \\ s_1c_2 \\ s_1s_2 \end{pmatrix}
\end{aligned}
\tag{24}
$$

III.B.ii. 4-Qubit Circuit

For this second set-up, the amount of qubits is doubled up to 4, which creates a 16 point spatial grid. This significant increase in grid size makes the matrix operators more complex as the circuit depth also increases. Because of this, at this stage there could potentially be signs of decoherence and noise that can affect the fidelity of the results.

The 4-qubit architecture used for the experiment consists of 3 layers with a total of 12 parameters, as seen in Figure V:

III.B.iii. 8-Qubit Circuit

The final set-up for this experiment consists in a 8 qubit circuit, which translates into a 256 point spatial grid. This step is a significant increase in matrix complexity, amount of

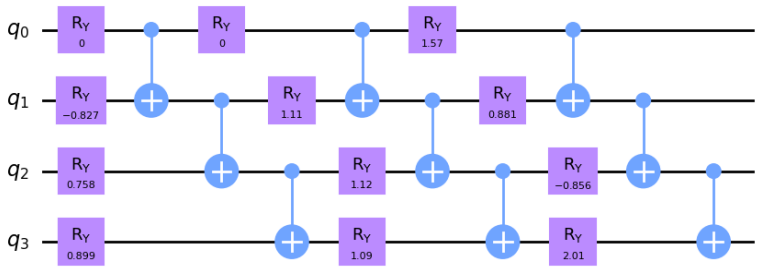


Figure V: 4-Qubit Circuit architecture.

Note: The qubits q_i in this circuit collectively encode the amplitude representation of the transformed heat state variable ϕ.

quantum gates, and circuit depth, which would probably push the limits of the current NISQ generation. Although the resolution of the grid could capture significant fluid behaviors, such large 256×256 matrices would be considerably affected by the known bottlenecks of NISQ machines such as decoherence and noise.

The 8-qubit architecture used for the experiment consists of 7 layers with a total of 56 parameters, as seen in Figure VI:

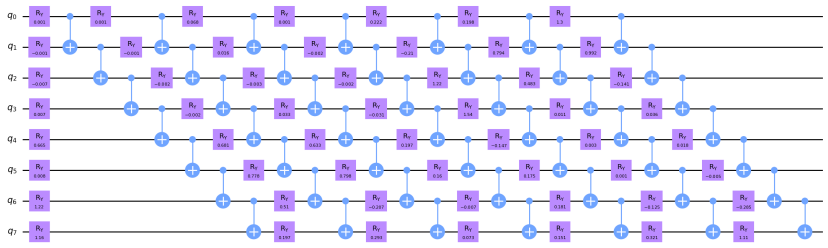


Figure VI: 8-Qubit Circuit architecture.

Note: The qubits q_i in this circuit collectively encode the amplitude representation of the transformed heat state variable ϕ.

III.C. Simulator and Hardware Implementation

In order to test these three architectures, a simulation environment is needed. Qiskit is a popular SDK written in Python that is widely used among researchers for quantum computing. Developed by IBM, it allows users to create quantum programs, execute them on different kinds of simulators, and even run them on real IBM quantum computers via remote access.

Initially, the three configurations of qubits have been tested on a statevector extraction simulator in order to observe how the algorithm performs in an ideal noiseless situation. Even though this technique is not realistic on current hardware, it gives a helpful feedback regarding the code's ability to be executed.

Then, to assess a performance that is more aligned with reality, the algorithm with the three qubit configurations has been implemented on Qiskit's AerSimulator, which is a local statevector simulator that provides a baseline to test the performance of the algorithm in conditions that imitate a real quantum computer due to its noise model that imitates the real IBM Fez quantum machine. Because of the current limitations of real hardware, it is not possible to extract the exact solution vector directly from the quantum circuit, as it has been tested in recent literature [5]. To tackle this issue, the algorithm runs a series of shots (Quantum State Tomography), as would be done on real hardware, so that the resulting probability distribution can properly return an accurate representation of the fluid.

Moreover, this approach also allows the algorithm to be executed on real quantum machines. By making use of the IBM Quantum Platform and its Open Plan with 10 free minutes of monthly runtime, the 2, 4 and 8 qubit versions have been run on real IBM quantum computers to see how the performance differs from that observed in simulated environments. However, executing a true hybrid QPU-CPU VQLS loop is unfeasible under the runtime limit as thousands of iterations would be required. In order to bypass this limitation, the code consists of two distinct phases: a first optimization phase that is run locally on the CPU using a simulated backend to calculate the angle parameters with the COBYLA optimizer [6], and a second phase that uses the previously calculated parameters to evaluate the circuit in the IBM quantum machine. The available quantum machines in the Open Plan are IBM Kingston, IBM Fez, and IBM Marrakesh, which all of them are IBM's Heron r2 model with 156 qubits. As the code was set to choose between the machine with the lowest queue time, the QPUs used for the experiment were IBM Fez for the 2Q and 8Q versions, and IBM Kingston for the 4Q version.

III.D. VQLS Algorithm Development

VQLS is the algorithm of choice to solve the discretized Burgers' equation system $Ax = b$, as it is specifically designed to be executed on current-era NISQ machines. Unlike the other known algorithms like HHL which require deep circuits, VQLS is a hybrid variational algorithm that is able to circumvent matrix inversion, as it is a non-unitary operation, by using a parameterized quantum ansatz that iteratively approximates the solution state through an optimization process [6].

Here, the steps of the algorithmic process are outlined:

III.D.i. Normalization and State Preparation

In the quantum framework, it is first necessary to normalize the vector b and the unknown solution vector x. Vector b, which might include physical noise from its measurement, is encoded to the quantum state, as seen in equation (25), where $||b||$ is the magnitude of the vector.

$$|b\rangle = \frac{b}{||b||} \tag{25}$$

As quantum mechanics is governed by probability, this is necessary to ensure all the amplitudes in the vector sum up to 1.

III.D.ii. LCU Decomposition

As quantum computers require the matrix A to be unitary, it is necessary to transform the current hermitian matrix into a unitary one so that reversibility is accomplished in the ansatz. To do so, Linear Combination of Unitaries (LCU) is used. Any square matrix can be decomposed and rewritten as a weighted sum of Pauli matrices (I, X, Y, Z) in equation (26), where c_k are the coefficients and P_k are the unitary Pauli strings.

$$A = \sum_{k=1}^{4^n} c_k P_k \tag{26}$$

The CPU runs the Hilbert-Schmidt's inner product (27) on all possible combinations of

Pauli strings for the particular case (e.g. II, IX, IY, ZZ...) and evaluates the result for each one, with n being the number of qubits, and Tr being the trace of the matrix of the resulting $A \cdot P_k$ matrix (sum of numbers on the main diagonal).

$$c_k = \frac{1}{2^n}\mathrm{Tr}(A \cdot P_k) \tag{27}$$

After evaluation, only the Pauli strings with a trace different than 0 are kept in the weighted sum of Pauli matrices. It is also important to point out that the amount of Pauli strings is going to increase exponentially as the number of qubits is increased, thus requiring for computational cost.

The reason behind utilizing LCU decomposition in this specific fluid dynamics framework is closely tied to the structure of the resulting FDM matrix A. Because the discretization of the heat equation yields a matrix that is sparse, real-symmetric, and tridiagonally dominant, the Hilbert-Schmidt inner product mathematically forces the vast majority of the Pauli string coefficients (c_k) to zero. Consequently, this ensures that the expectation values can be computed efficiently on near-term hardware.

III.D.iii. Cost Function

In order to guide the optimization loop, it is necessary to design a cost function $C(\theta)$ that is able to evaluate to 0 when the trial state $|x\rangle$ is proportional to the true solution of the linear system $Ax = b$, as the calculated angles θ are varied by the optimizer.

At first, local cost functions that rely on calculating the square of the measured amplitudes to obtain the probabilities were considered. However, this pathway did not provide accurate enough results for the 4 and 8 qubit version due to losing the negative and complex phases when recovering the amplitudes from the measured probabilities ($\sqrt{P_i}$), making the optimizer navigate wrongly.

To tackle this issue, the global cost function used is able to preserve all the phase information by avoiding the observation of the amplitudes [6]. The cost function is defined as in equation (28):

$$C_G(\theta) = 1 - \frac{\langle x(\theta)|O_{\text{num}}|x(\theta)\rangle}{\langle x(\theta)|O_{\text{den}}|x(\theta)\rangle} = 1 - \frac{|\langle b|A|x\rangle|^2}{\langle x(\theta)|A^2|x(\theta)\rangle} = 1 - \frac{\langle x(\theta)|A|b\rangle\langle b|A|x(\theta)\rangle}{\langle x(\theta)|A^\dagger A|x(\theta)\rangle} \tag{28}$$

In this cost function, both numerator and denominator are restructured into Hermitian observables, which are matrices that represent a physical quantity that is wanted to be measured. An Hermitian matrix can be rewritten as a sum of Pauli matrices, the same as it has been done in the LCU decomposition step. The numerator can be algebraically rewritten as the expectation value of the Hermitian observable $\langle x|A|b\rangle\langle b|A|x\rangle = |\langle b|A|x\rangle|^2 = \langle x|O_{\text{num}}|x\rangle$. The squaring occurs algebraically before any measurement takes place, so all phase information is captured through quantum interference rather than through amplitude observation. The result is a single expectation value that indicates how much the quantum state $|x\rangle$ relates to the observable O_{num}, as observed in equation (29).

$$\langle x|O_{num}|x\rangle = \underbrace{\langle x|}_{(1\times N)} \cdot \underbrace{O_{num}}_{(N\times N)} \cdot \underbrace{|x\rangle}_{(N\times 1)} = \underbrace{\text{scalar}}_{(1\times 1)} \tag{29}$$

The denominator $O_{\text{den}} = A^2$ normalizes the cost function by measuring the squared norm of the transformed state $A|x\rangle$, making it scale-invariant, as only the direction of the vector matters.

Both observables are Hermitian and are decomposed into Pauli strings via the LCU decomposition, allowing the Estimator to evaluate each expectation value without measuring the amplitudes. This requires only two circuit evaluations per optimizer iteration—one for O_{num} and one for O_{den}—with no ancillary qubits or Hadamard test circuits, making the approach fully compatible with real IBM quantum hardware.

III.D.iv. Optimization Loop

VQLS algorithm operates within a continuous hybrid QPU-CPU loop. At each iteration, CPU and QPU work together with the objective of minimizing the cost function. A classical optimizer COBYLA (Constrained Optimization BY Linear Approximations) is utilized to update the parameters in the R_y gates. COBYLA is a gradient-free optimizer that is very

resilient against hardware noise, which makes it optimal for NISQ hardware applications.

The iterative process follows these steps:

1. A first random batch of parameters (angles) are generated and sent to the QPU.
2. QPU runs the ansatz circuit with those angles, rotating the qubits into a trial state $|x(\theta)\rangle$.
3. QPU examines the circuit and performs a shot-count of measurements. The measurements are then used to evaluate the expectation values of the cost function.
4. Cost function returns a scalar value on the CPU and it is evaluated by COBYLA.
5. COBYLA calculates a new set of parameters based on the current cost to lower the cost function in the next iteration.

III.D.v. Solution Extraction and Back To Non-Linear

Once the COBYLA optimizer manages to converge the cost function to the minimum, the final parameters θ are stored. The quantum circuit is then prepared to extract the final quantum state whose probability amplitudes represent the normalized solution to the heat equation.

Because the final quantum state is normalized so that all the probabilities sum up to 1, the resulting amplitudes must be extracted from the quantum processor and de-normalized. In the noiseless simulation, the extraction is achieved through direct statevector extraction (solution array is extracted from the CPU's RAM), whereas on physical hardware, it requires quantum state tomography (circuit collapses and the state is measured a large number of times). Once the solution vector is extracted, it is then multiplied by the scaling factor (derived from the initial boundary vector $||b||$ and the matrix's condition number) in order to restore the physical magnitude of the heat state.

Once the final vector is extracted, it is necessary to convert the heat equation back to the non-linear domain to obtain the velocity u of the fluid. To do so, the inverse of the previously introduced Cole-Hopf Transformation (30) is used. The transformation relates the fluid velocity $u(x,t)$ to the heat state $\phi(x,t)$ through the following:

$$u(x,t) = -2\nu \frac{1}{\phi} \frac{\partial \phi}{\partial x} \tag{30}$$

Finally, the continuous spatial derivative is approximated by using classical numerical differentiation methods (central difference). For a spatial grid with step size Δx, the velocity of the fluid u_i at each point of the grid i is calculated as in equation (31):

$$u_i = -2\nu \left(\frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x \cdot \phi_i} \right) \tag{31}$$

III.E. Evaluation Metrics

In order to assess the performance of the algorithm, a set of evaluation metrics have been used. For the three kinds of code tested, the execution time was observed to observe how feasible they are to run in each case, and their global cost value to see that the cost approaches 0 as much as possible.

Each of the runs also include their respective graphs of comparison between the classical and quantum result of both heat state and fluid velocity once it is transformed back into the non-linear Burgers' equation. These graphs include measurements of the evolution of the cost function across the optimization process until convergence.

Both heat and velocity states include the mean absolute error (MAE) to see the average error across the whole spatial domain. Moreover, in the statevector extraction case, a noise robustness analysis has been performed. On this analysis, a synthetic noise ranging $5\% - 40\%$ is injected to the initial boundary vector to observe how the algorithm is able to handle noise in the applied data. The noise robustness analysis has not been performed on the AerSimulator and IBM hardware versions because of computational cost.

IV. RESULTS

In this chapter, the results of the different scenarios for the VQLS algorithm applied to the linearized Burgers' equation are presented. In order to fully observe the capabilities of the algorithm with its three different qubit configurations for a spatial fluid grid of $N = 4, 16, 256$ (2, 4 and 8 qubits), VQLS was evaluated across three different execution environments:

1. Statevector Simulation

2. AerSimulator (FakeFez)

3. IBM Quantum Hardware

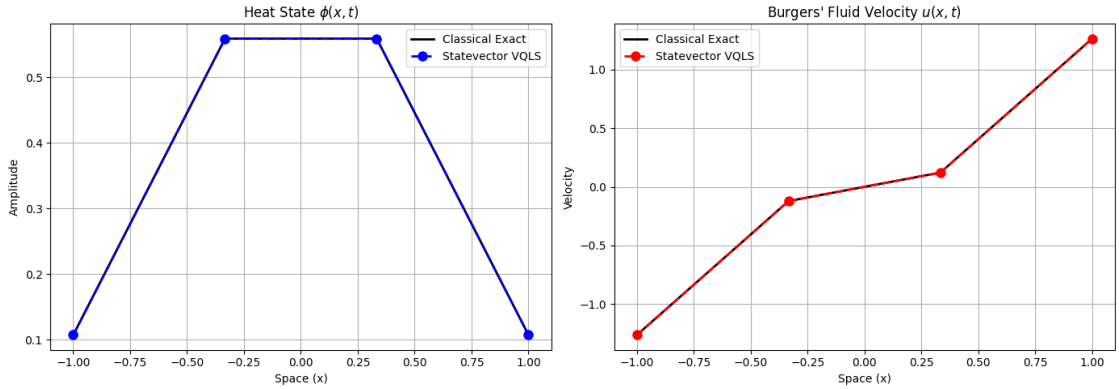
IV.A. Statevector Simulation

Statevector simulation serves as a mathematical baseline so that the algorithm can be tested before applying noise models or even executing it on physical quantum hardware. As statevector extraction is an ideal way of getting the solution vector without needing to measure the circuit, it does not require to take a large number of shots; it directly extracts the solution. However, this makes this method unrealistic for current hardware.

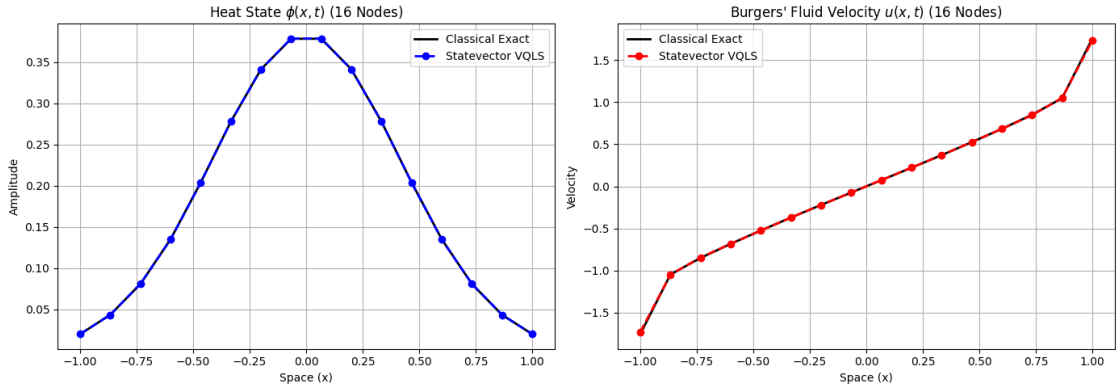
As seen in the convergence graphs in Figure VIII, COBYLA is able to reduce the cost function at a steady descent grade for both 2 and 4-qubit systems. However, the 8-qubit circuit struggles in minimizing the cost as it reaches a barren plateau at around 1.000 iterations. This means that, despite statevector extraction being a noiseless ideal environment, COBYLA is unable to navigate through a high-dimensional landscape. The algorithm complexity is significantly increased at 8 qubits, which means that the limit of expressivity is reached.

Consequently, the reconstructed fluid profiles for heat state and velocity in Figure VII matched the exact classical solution for the 2 and 4-qubit cases but produced high-frequency spikes across the curve of the 8-qubit circuit due to the lack of expressivity. In Table I, it can be observed that the heat state MAE gives very low errors, even though the 8-qubit circuit produces spikes in the profile. In terms of velocity, the MAE remains low for the 2 and 4-qubit versions, whereas the spikes on the 8-qubit versions massively amplifies the velocity error, as can be observed in the velocity profile for that case. After all, it can be said that the algorithmic formulation remained mathematically robust across the three versions.

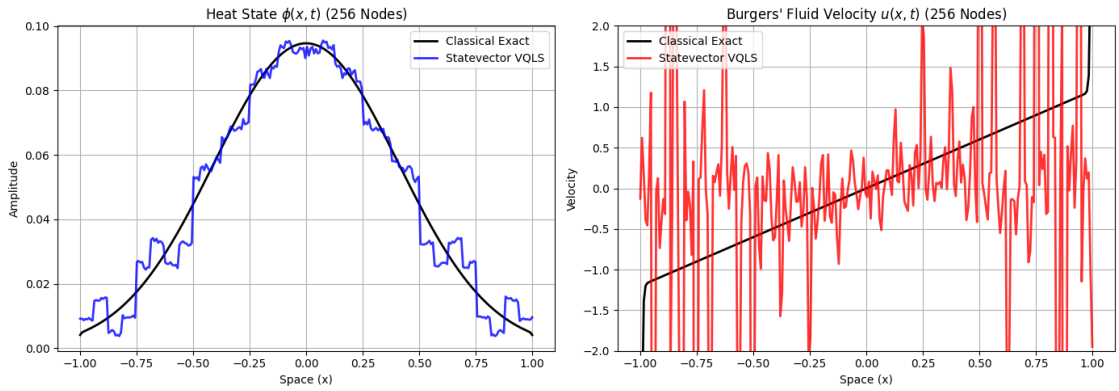
For this statevector scenario, a noise robustness analysis (Table II) has been performed to observe the theoretical capability of this algorithm to handle disturbances in the input data. The results demonstrate that the algorithm running with COBYLA is able to handle noise in the boundary vector quite acceptably until a certain level of noise for the 2 and 4-qubit cir-



(a) 2-Qubit System (4-Node Grid)

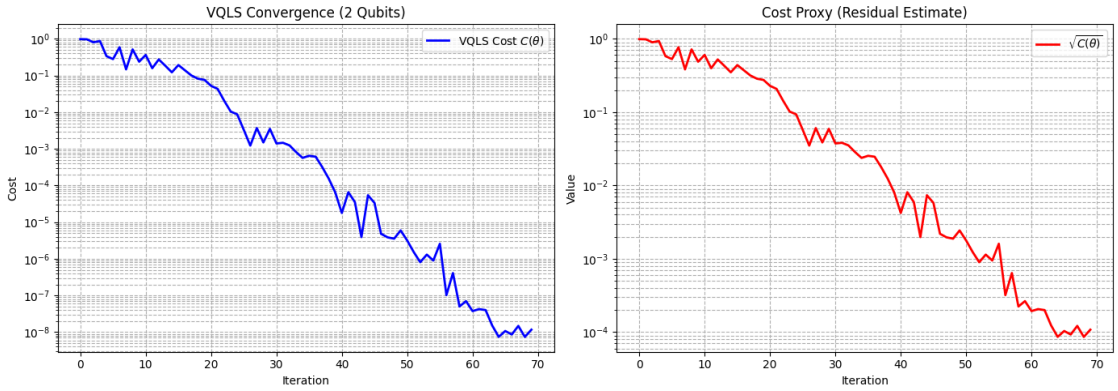


(b) 4-Qubit System (16-Node Grid)

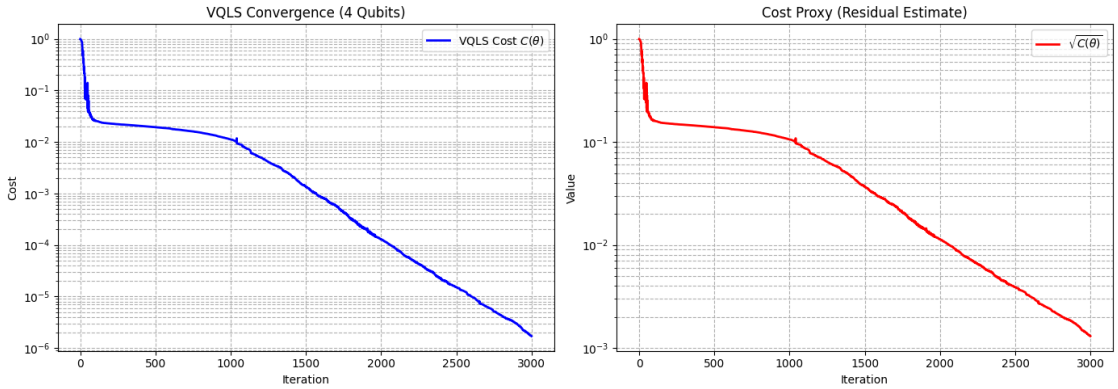


(c) 8-Qubit System (256-Node Grid)

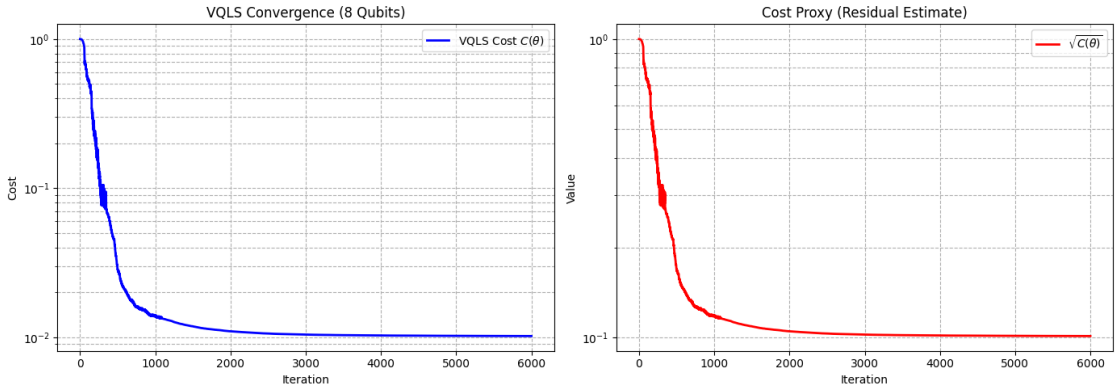
Figure VII: **Reconstructed Fluid Heat and Velocity Profiles from Statevector Simulator.** A spatial comparison of the exact classical solution versus the quantum-derived results across (a) 4-node, (b) 16-node, and (c) 256-node grids using noiseless ideal statevector simulation. While the 2-qubit and 4-qubit systems achieve near-perfect overlap, the 8-qubit system (c) displays severe high-frequency instability in the velocity domain. COBYLA is not able to surpass the barren plateau in the 8-qubit version.



(a) 2-Qubit System (4-Node Grid)



(b) 4-Qubit System (16-Node Grid)



(c) 8-Qubit System (256-Node Grid)

Figure VIII: **VQLS Optimization Convergence from Statevector Simulator.** Algorithmic training progress of the global cost function across scaled grid sizes using statevector extraction. The 2-qubit (a) and 4-qubit (b) systems converge smoothly to highly precise global minima ($< 10^{-5}$). However, the 8-qubit system (c) exhibits a pronounced barren plateau, stalling near a cost of 10^{-2} despite 6,000 iterations. This indicates that scaling to 256 nodes introduces significant algorithmic complexity, reaching the limit of expressivity.

Qubits	Final Cost (C_G)	Exec. Time (s)	MAE (Heat)	MAE (Velocity)
2 qubits	0.0000	0.2	0.0000	0.0001
4 qubits	0.0000	15.0	0.0002	0.0049
8 qubits	0.0102	813.6	0.0045	1.6688

Table I: **Statevector Simulator Execution Metrics.** Performance and accuracy results for the VQLS algorithm executed on statevector simulator. Note the degradation in velocity accuracy at the 8-qubit scale.

cuits. The 8-qubit version struggles even at low noise levels because of its complexity in terms of qubit communication. As the heat state error increases, the velocity error reaches a massive value because the de-linearization amplifies it, as was expected. This shows that VQLS paired with the COBYLA optimizer is a robust algorithm for shallow circuit configurations (2 and 4 qubits for this case).

Noise	Heat State Error (%)			Velocity Error (%)		
Variance	2-Qubit	4-Qubit	8-Qubit	2-Qubit	4-Qubit	8-Qubit
5%	1.81	2.88	19.22	6.99	40.37	3628.62
10%	6.06	9.60	19.52	67.08	99.93	1792.53
15%	9.62	8.28	17.86	137.08	160.09	8614.40
20%	10.19	9.46	18.37	133.57	134.88	9928.98
25%	7.31	14.42	21.19	26.51	68.72	17426.02
30%	15.87	15.17	25.67	554.80	65.09	920.35
35%	29.89	12.98	19.96	192.11	173.49	14977.83
40%	31.62	28.39	24.05	151.30	613.05	9889.74

Table II: **Noise Robustness Analysis Across Scaled Statevector Simulations.** Synthetic Gaussian noise (ranging from 5% to 40% variance) was injected into the initial classical boundary vector $|b\rangle$. While the linear heat state fidelity degrades at a relatively moderate and expected rate, the non-linear fluid velocity error explodes massively, especially for the 8-qubit case.

IV.B. AerSimulator (FakeFez)

To bridge the gap between theory and physical reality, the three versions of the algorithm have been tested on Qiskit's AerSimulator utilizing the FakeFez noise model, which replicates the properties and architecture of IBM Fez (one of the three available IBM quantum computers

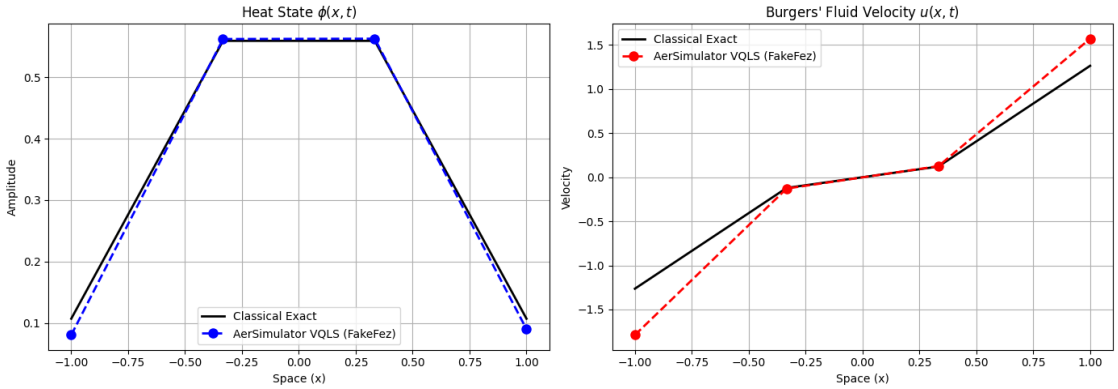
usable on the free tier). FakeFez uses system snapshots of the real QC to replicate its behavior and errors. The execution has been performed, in contrast to the statevector environment, with measurement shots, as it would be done on a real quantum machine. The 2-qubit circuit has been measured 8.000 times with 1.000 COBYLA iterations, and the 4 and 8-qubit versions have been measured 10.000 times with 3.000 and 6.000 cost reducing iterations respectively. Although the number of shots for the two larger circuits might not provide the accuracy that a larger number would, its computational cost and time required would be excessively large as simulators with noise models take much longer time to compute.

The introduction of noise in the simulation altered the results significantly compared to the statevector ideal simulation. The simulated noise and the probability distribution resulting from the measurement shots lessen the accuracy that was seen before. In the convergence graphs in Figure X, the initial rapid descent is still present as seen before. However, the optimizer quickly encounters a noisy barren plateau, which is caused by the noise present in the FakeFez simulator, which makes the optimizer stop after a really short number of iterations as COBYLA gets confused by statistical and shot noise and keeps trying to surpass the plateau with no effect. This premature convergence affects the accuracy of the reconstruction results significantly.

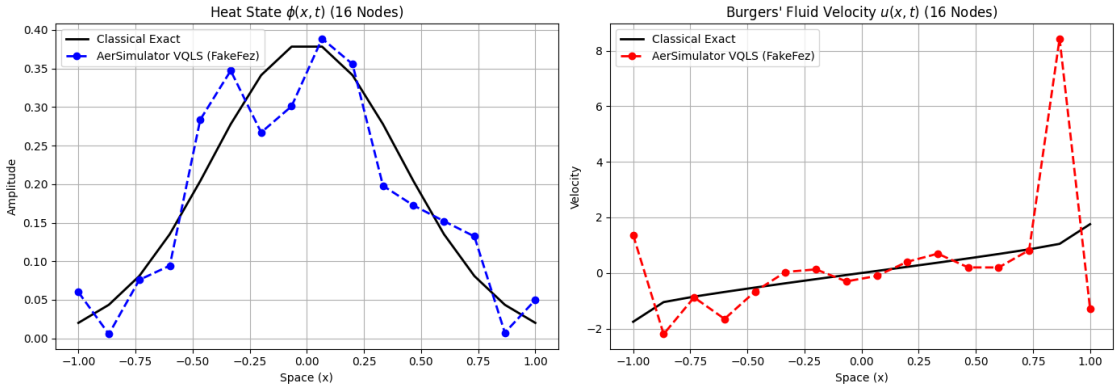
As the cost function cannot be further reduced below 10^{-1}, the fluid reconstruction results are considerably affected (Figure IX). While the 2-qubit version is able to reconstruct the heat state effectively, when the depth of the circuit is increased, the results become more inaccurate with spikes that make the fluid reconstruction fluctuate. Even though the results in Table III show relatively low MAE for the linear heat state, the non-linear derivative step causes distinct spikes that are especially obvious in the 8-qubit reconstruction, which experienced severe degradation due to its depth and complexity. As observed, the execution time rises exponentially when the number of qubits is increased, reaching several hours of runtime in the case of 8 qubits.

IV.C. IBM Quantum Hardware

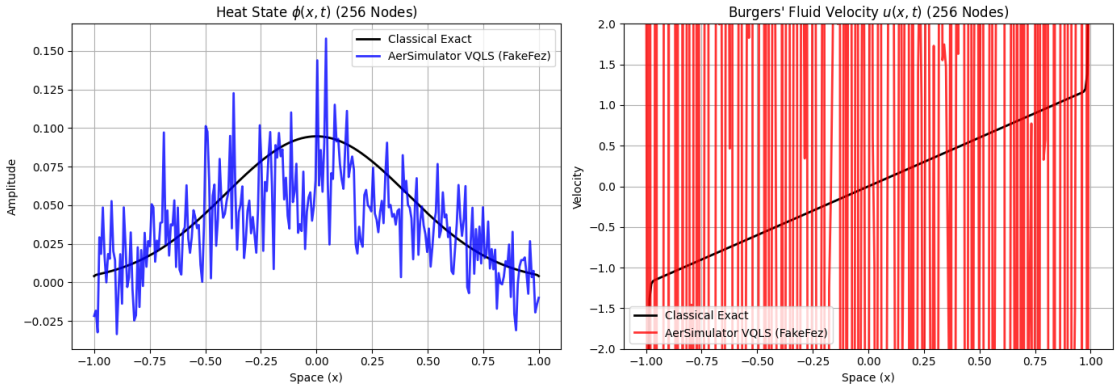
Executing the algorithm on physical quantum hardware from IBM exposed really interesting insights. Even though NISQ hardware is subject to issues such as analog drift, qubit inter-



(a) 2-Qubit System (4-Node Grid)

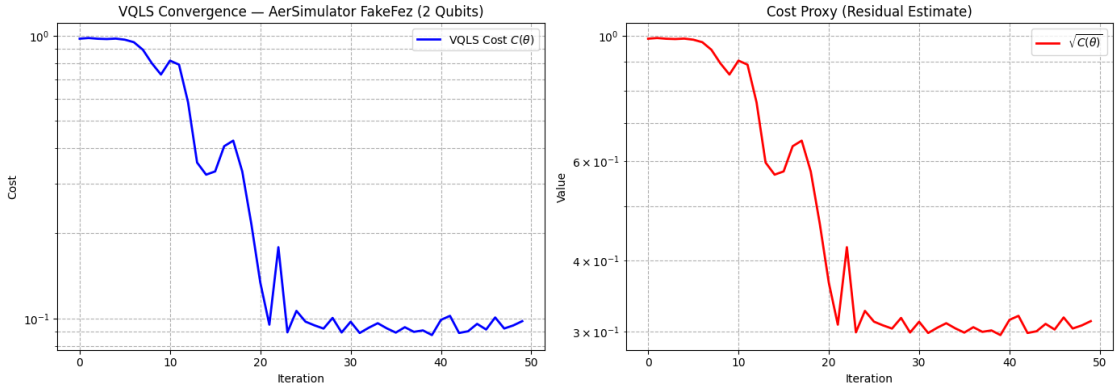


(b) 4-Qubit System (16-Node Grid)

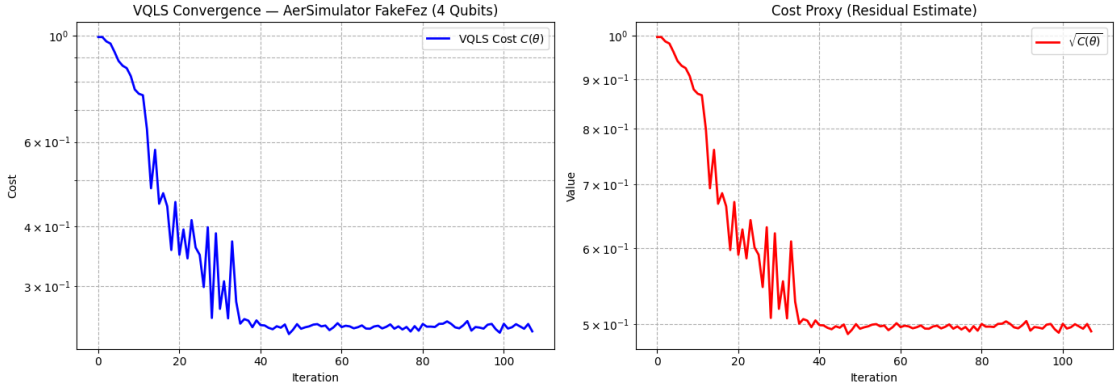


(c) 8-Qubit System (256-Node Grid)

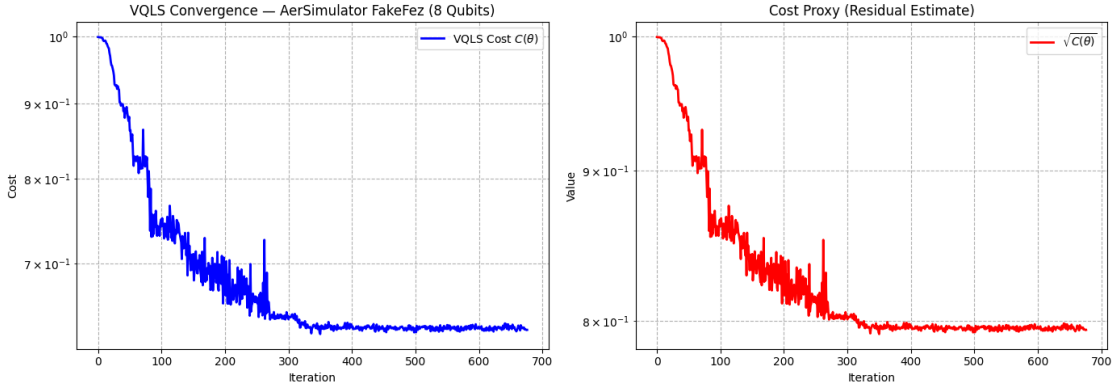
Figure IX: **Reconstructed Fluid Heat and Velocity Profiles from AerSimulator FakeFez.** A spatial comparison of the exact classical solution versus the quantum-derived results utilizing the IBM FakeFez noise model. (a) The 2-qubit system maintains the general wave profile despite minor amplitude dampening. (b) The 4-qubit and (c) 8-qubit systems clearly demonstrate the vulnerability of the inverse Hopf-Cole transformation, amplifying the error on the velocity with severe spikes on the 8-qubit circuit.



(a) 2-Qubit System (4-Node Grid)



(b) 4-Qubit System (16-Node Grid, 10,000 Shots)



(c) 8-Qubit System (256-Node Grid)

Figure X: **VQLS Optimization Convergence from AerSimulator FakeFez.** Algorithmic training progress of the global cost function utilizing the IBM FakeFez noise model. In the three circuits, the cost rapidly descents and stalls when it hits a noisy barren plateau, as COBYLA cannot get through statistical hardware noise. The cost reduction gets worse when the depth of the circuit is increased.

Qubits	**Final Cost (C_G)**	**Exec. Time (s)**	**MAE (Heat)**	**MAE (Velocity)**
2 qubits	0.0873	244.9	0.0125	0.2112
4 qubits	0.2385	610.2	0.0434	1.1466
8 qubits	0.6268	21900.1	0.0216	22.1043

Table III: **AerSimulator (FakeFez) Execution Metrics.** Performance and accuracy results for the VQLS algorithm utilizing IBM's `FakeFez` noise model. The 8-qubit configuration highlights the dual bottlenecks of the NISQ era: the classical simulation time scales exponentially (exceeding 6 hours), and the unmitigated simulated hardware noise drives the optimization into a barren plateau ($C_G \approx 0.62$). This elevated cost completely destabilizes the inverse Hopf-Cole transformation, resulting in a catastrophic physical velocity error.

ference, and limited qubit connectivity, it is observed that the results for the three different configurations are fairly reasonable and acceptable. The execution has been performed with a number of measurement shots that fitted into the time limit of 10 available minutes: 8.000 shots for the 2-qubit circuit, and 4.000 shots for the 4 and 8-qubit versions. COBYLA was set on 1.000 iterations for the 2-qubit, 3.000 for the 4-qubit, and 6.000 for the 8-qubit version.

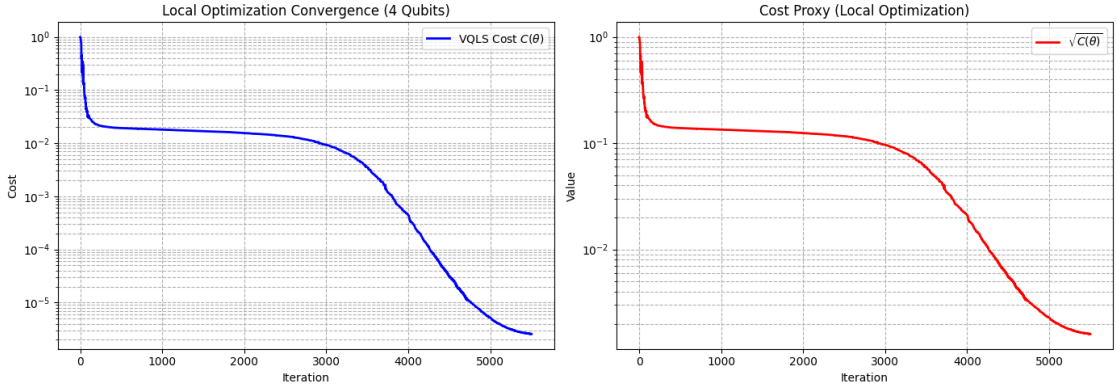
When looking at the convergence plots in Figure XII, it can be seen that the cost for the 2-qubit circuit is able to approach 0 steadily in a short amount of iterations (before 100 iterations). On the other hand, for the 4 and 8-qubit circuits, COBYLA reduces the cost drastically during the first iterations but reaches a barren plateau as, being a gradient-free optimizer, it struggles when navigating through a large number of dimensions.

Regarding the fluid heat and velocity reconstruction (Figure XI), all three cases present an accurate reconstruction of the heat state, although the 8-qubit circuit contains a significant amount of noise in form of spikes across the profile. This highlights that the 8 qubits are possibly not fully connected in the IBM quantum machine, reaching quantum decoherence that makes some information to be lost. This noise in the heat state is directly translated to the velocity profile, which generate massive spikes, especially in the 8-qubit version. The 2-qubit circuit gives a really accurate profile after delinearization, and the 4-qubit generates large spikes on the limits of the boundary condition where the error is higher. In general, as seen in Table IV, the MAE for the heat state maintains really acceptable values for the three cases, whereas for the velocity, the error is highly amplified for the 4 (because of the spikes on the edges) and 8-qubit cases. The execution time increases exponentially as the number

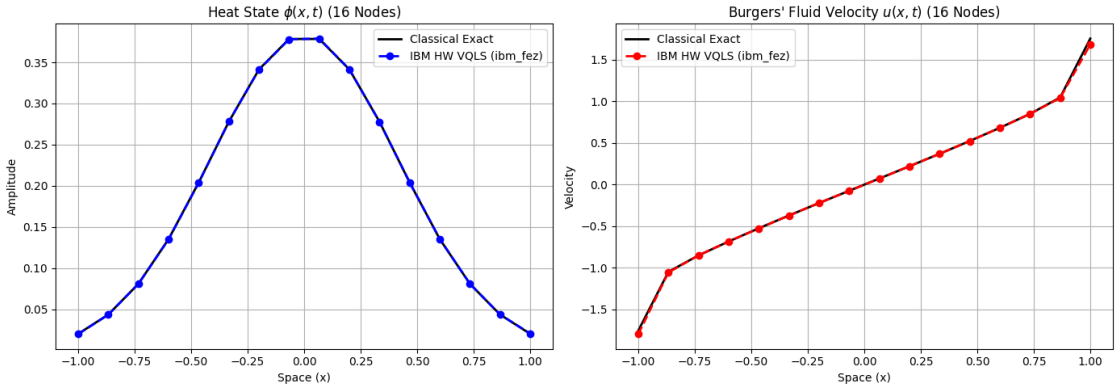
of qubits is increased further above the 4-qubit threshold in both local CPU optimization and QPU evaluation.

Qubits	**Cost** (C_G)	**CPU Time**	**QPU Time**	**MAE (H)**	**MAE (V)**
2 qubits	0.0000	0.29 s	32.30 s	0.0000	0.0001
4 qubits	0.0093	17.85 s	57.80 s	0.0113	0.4426
8 qubits	0.0099	881.54 s	419.76 s	0.0044	1.6233

Table IV: **IBM Quantum Hardware Execution Metrics.** Performance and accuracy results for the VQLS algorithm executed on IBM hardware. Note the degradation in accuracy at the 8-qubit scale, highlighting the vulnerability of the non-linear velocity transformation.



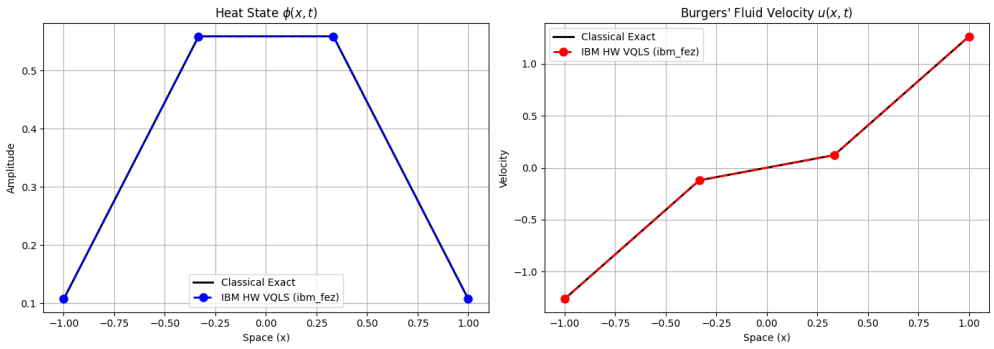
(a) VQLS Optimization Convergence



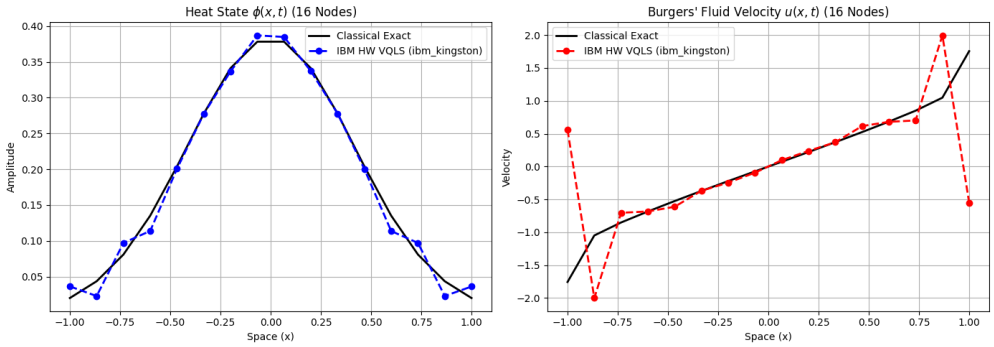
(b) Reconstructed Heat State and Fluid Velocity

Figure XIII: **IBM Hardware Performance on a 4-Qubit System (8.000 Shots, 6.000 COBYLA Iterations).** (a) The algorithmic training progress of the global cost function over 6.000 iterations. (b) The resulting profile for heat state and velocity. Note the increased accuracy in both heat state and velocity after the cost is further decreased by COBYLA.

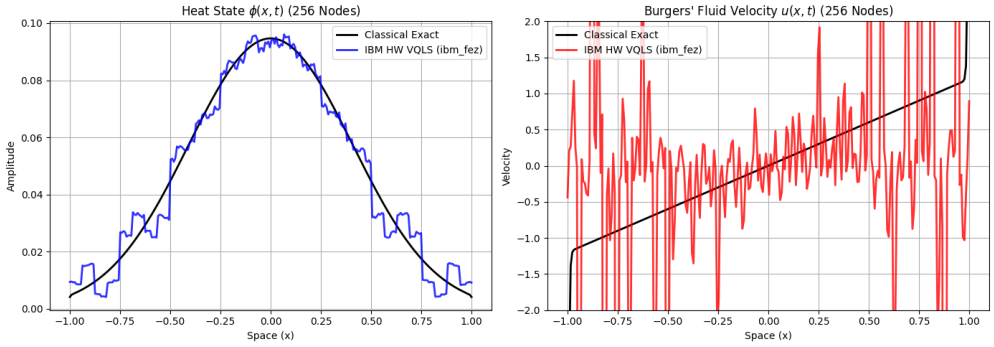
Moreover, as the cost barren plateau for the 4-qubit circuit seemed to start decreasing



(a) 2-Qubit System (4-Node Grid)

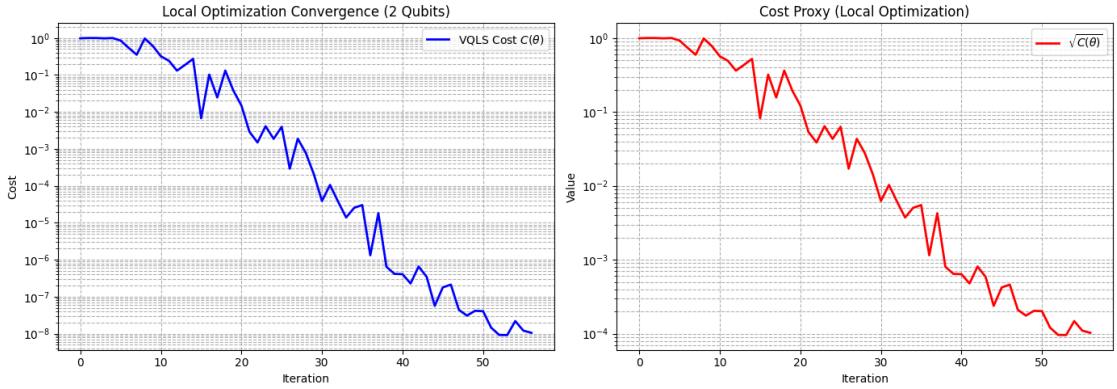


(b) 4-Qubit System (16-Node Grid)

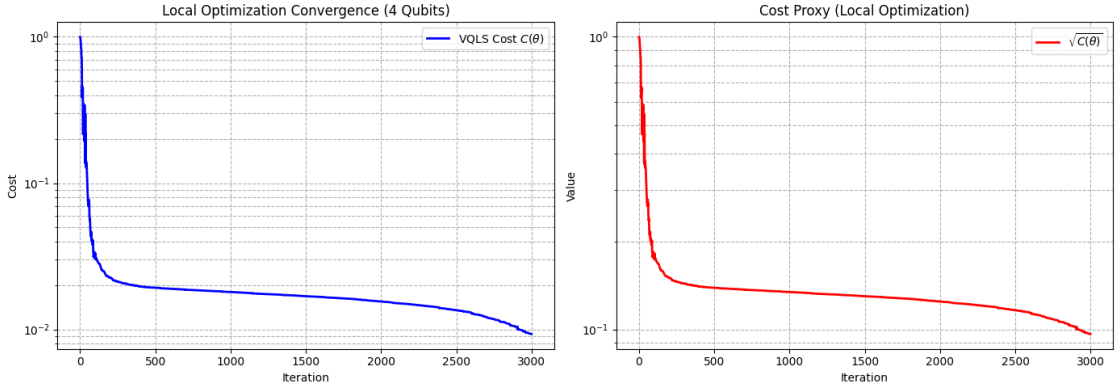


(c) 8-Qubit System (256-Node Grid)

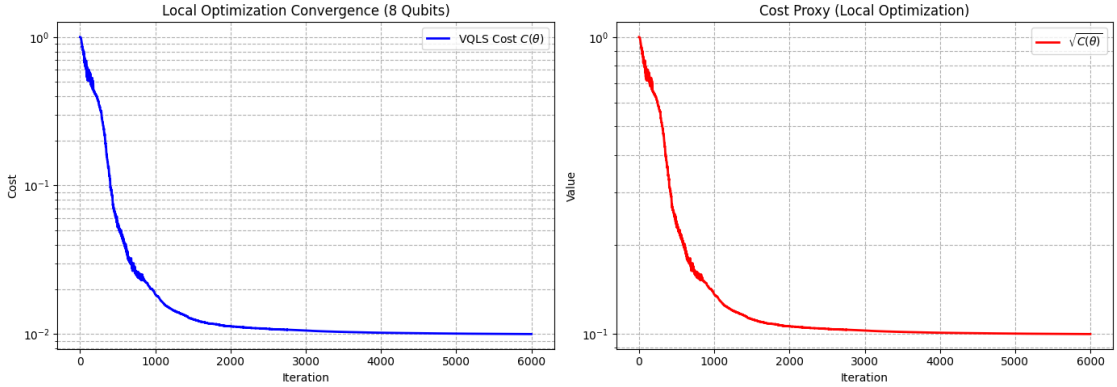
Figure XI: **Reconstructed Fluid Heat and Velocity Profiles from IBM Hardware.** A spatial comparison of the exact classical solution to the viscous Burgers' equation versus the derived fluid velocities across (a) 4-node, (b) 16-node, and (c) 256-node grids. Note the degradation of physical accuracy and the emergence of severe velocity spikes in (c) as the physical circuit depth exceed the hardware coherence limits.



(a) 2-Qubit System (4-Node Grid)



(b) 4-Qubit System (16-Node Grid)



(c) 8-Qubit System (256-Node Grid)

Figure XII: **VQLS Optimization Convergence from IBM Hardware.** Algorithmic training progress of the COBYLA optimizer for the (a) 2-qubit, (b) 4-qubit, and (c) 8-qubit hardware configurations. Note that as the system scales to 8 qubits (c), the optimizer experiences a more pronounced barren plateau and a higher final convergence floor, highlighting the importance of hardware noise and gate errors in deeper circuits.

around the end of the 3.000 iterations, the circuit was tested again increasing the COBYLA iterations up to 6.000 and the amount of shots doubled to 8.000 to observe if the results are better. As observed in Figure XIII, COBYLA is now able to reduce the cost significantly after passing over the barren plateau, resulting in much better accuracy in the heat state reconstruction that translates perfectly to the velocity. In consequence, it can be said that the 4-qubit version is capable of providing extraordinary results on physical hardware if the optimizer is able to reduce the cost to a sufficient level (close to 10^{-6}).

When comparing these results to the ones obtained from the FakeFez simulator, it is surprising to see how up-to-date quantum hardware is able to outperform noisy simulators. IBM quantum hardware automatically applies several error mitigation techniques "behind the curtain", which are not present in Qiskit's AerSimulator, that help achieve a significantly lower cost than a noisy simulator and, therefore, more accurate reconstruction results. These techniques include Zero Noise Extrapolation, dynamical decoupling, and measurement error mitigation [18, 19]. Error mitigation techniques are exclusive to IBM physical quantum hardware and are not replicated on local simulators. This highlights that simulators should be understood as a conservative lower bound on achievable accuracy for scalable cases rather than a faithful replica of real QC performance, as there is a gap in accuracy.

V. CONCLUSION

This thesis evaluated the viability of the Variational Quantum Linear Solver (VQLS) for solving the 1D viscous Burgers' equation, a fundamental non-linear partial differential equation used to model fluid flow in Computational Fluid Dynamics (CFD). To bypass the inherent linearity of quantum mechanics, the Cole-Hopf transformation was successfully utilized to linearize the convective term into the standard heat equation, allowing the system to be discretized and mapped into a quantum framework. The hybrid QPU-CPU algorithm was evaluated across increasingly scaled topographies of 2, 4, and 8 qubits (representing 4, 16, and 256 spatial nodes) across three execution environments: an ideal statevector simulator, a noisy AerSimulator, and physical IBM quantum hardware.

The ideal statevector simulations confirmed that VQLS, paired with the gradient-free COBYLA optimizer, is mathematically robust and highly accurate for shallow circuits of 2 and 4 qubits. However, when scaling the algorithm to 8 qubits, even though the reconstructed heat state mimics the classical solution shape, it revealed significant algorithmic bottlenecks; the optimizer encountered a barren plateau and struggled to navigate the high-dimensional landscape, indicating that the chosen circuit ansatz reached its limit of expressivity even in the absence of physical noise. Therefore, a larger circuit with extended expressivity might be theoretically beneficial for better expressivity. However, this introduces a tradeoff as adding more parameters would increase the computational overhead for the CPU and force the optimizer to navigate through an even more highly-dimensional landscape, thus resulting in more barren plateaus. Crucially, the noise robustness analysis demonstrated a severe vulnerability when bridging quantum algorithms with non-linear physics: the inverse Cole-Hopf transformation acts as a massive error amplifier. Even minor deviations in the linear heat state result in catastrophic, high-frequency spikes in the reconstructed fluid velocity profile, especially for the 8-qubit case.

When transitioning to a realistic conditions environment, the simulated noise in the AerSimulator FakeFez caused premature algorithmic convergence, generating highly inaccurate velocity profiles for deeper circuits and pushing the classical simulation time past 6 hours for the 8-qubit configuration. Surprisingly, the algorithm performed significantly better on

actual IBM quantum hardware than on the noisy simulator. This outperformance was driven by IBM's "behind the curtain" error mitigation techniques that allowed the 4-qubit physical system to generate highly accurate fluid profiles when granted sufficient measurement shots and optimization iterations. The short amount of shots limited by the runtime access to IBM's free tier has been enough for executing the two larger circuits with acceptable accuracy, even though the execution of these circuits with a large shot amount remains for future research. Nevertheless, the 8-qubit hardware execution still suffered from massive velocity spikes, likely due to deep circuit decoherence and limited qubit connectivity on the physical chip.

Ultimately, while quantum computing theoretically offers an exponential advantage for solving the large-scale linear systems inherent to CFD , current Noisy Intermediate-Scale Quantum (NISQ) devices face severe practical limitations. The VQLS framework is capable of solving shallow-circuit fluid models, but scaling to the high resolutions required for practical industrial CFD is currently unfeasible. The compounding challenges of hardware decoherence, algorithmic barren plateaus, and the extreme amplification of quantum noise during non-linear mathematical mappings heavily restrict near-term applicability. For quantum fluid dynamics to realize its potential, future advancements must focus on the development of fault-tolerant logical qubits, highly expressive ansätze, and noise-resilient mathematical frameworks for non-linear equations. It can be concluded that, although the shallow versions of the circuits are fully able to reconstruct the heat state with fidelity, the non-linear transformation of this particular case faces severe errors when the heat state is not perfectly replicated. For future work, it would be of great interest to test the VQLS algorithm with other non-linear equations and different linearization techniques that are not specific to the particular case of study. Due to the limitations imposed by the limited run-time, only one time step has been simulated; with more access to quantum hardware, further time steps could be simulated in order to observe more critical phenomena such as shockwave formation.

Declaration of AI Usage

During the preparation of this thesis, generative artificial intelligence (Claude) was used to assist with code debugging, structural refinement, and LaTeX typesetting. Following the use of these tools, the author thoroughly reviewed and edited the generated content and assumes full, personal responsibility for the final content, scientific findings, and integrity of this document. No AI was used to generate or alter the physical fluid dynamics data, quantum algorithmic results, or core mathematical proofs.

References

[1] Andreas Bayerstadler, Guillaume Becquin, Julia Binder, Thierry Botter, Hans Ehm, Thomas Ehmer, Marvin Erdmann, Norbert Gaus, Philipp Harbach, Maximilian Hess, Johannes Klepsch, Martin Leib, Sebastian Luber, Andre Luckow, Maximilian Mansky, Wolfgang Mauerer, Florian Neukart, Christoph Niedermeier, Lilly Palackal, Ruben Pfeiffer, Carsten Polenz, Johanna Sepulveda, Tammo Sievers, Brian Standen, Michael Streif, Thomas Strohm, Clemens Utschig-Utschig, Daniel Volz, Horst Weiss, and Fabian Winter. Industry quantum computing applications. *EPJ Quantum Technology*, 8(1), 12 2021. ISSN 21960763. doi: 10.1140/epjqt/s40507-021-00114-x.

[2] Paul Kieckhefen, Swantje Pietsch, Maksym Dosta, and Stefan Heinrich. Possibilities and Limits of Computational Fluid Dynamics-Discrete Element Method Simulations in Process Engineering: A Review of Recent Advancements and Future Trends. 00:19, 2026. doi: 10.1146/annurev-chembioeng. URL https://doi.org/10.1146/annurev-chembioeng-.

[3] Claudio Sanavio and Sauro Succi. Quantum computing for simulation of fluid dynamics. 1 2024. doi: 10.5772/intechopen.1005242. URL http://arxiv.org/abs/2404.01302http://dx.doi.org/10.5772/intechopen.1005242.

[4] John Preskill. Quantum Computing in the NISQ era and beyond. 7 2018. doi: 10.22331/

q-2018-08-06-79. URL http://arxiv.org/abs/1801.00862http://dx.doi.org/10.22331/q-2018-08-06-79.

[5] Amir Hossein Salehi Shayegan. Quantum linear solvers for scientific computing: a comparison of VQLS, HHL and quantum annealing on time-fractional diffusion problems. *Scientific Reports*, 16(1):10278, 2 2026. ISSN 2045-2322. doi: 10.1038/s41598-026-40910-y. URL https://www.nature.com/articles/s41598-026-40910-y.

[6] Carlos Bravo-Prieto, Ryan LaRose, M. Cerezo, Yiğit Subaşı, Lukasz Cincio, and Patrick J. Coles. Variational Quantum Linear Solver. *Quantum*, 7, 2023. ISSN 2521327X. doi: 10.22331/q-2023-11-22-1188. URL http://dx.doi.org/10.22331/q-2023-11-22-1188.

[7] Felix Tennie, Sylvain Laizet, Seth Lloyd, and Luca Magri. Quantum Computing for nonlinear differential equations and turbulence. 6 2024. URL http://arxiv.org/abs/2406.04826.

[8] Andrew Steane. Quantum computing. *Reports on Progress in Physics*, 61(2):117–173, 2 1998. ISSN 0034-4885. doi: 10.1088/0034-4885/61/2/002. URL https://iopscience.iop.org/article/10.1088/0034-4885/61/2/002.

[9] Colin P. Williams. *Explorations in Quantum Computing*. Springer London, London, 2011. ISBN 978-1-84628-886-9. doi: 10.1007/978-1-84628-887-6. URL https://link.springer.com/book/10.1007/978-1-84628-887-6.

[10] Madhava Syamlal, Carter Copen, Masashi Takahashi, and Benjamin Hall. Computational Fluid Dynamics on Quantum Computers. 7 2024. URL http://arxiv.org/abs/2406.18749.

[11] Frank Gaitan. Finding flows of a Navier–Stokes fluid through quantum computing. *npj Quantum Information*, 6(1), 12 2020. ISSN 20566387. doi: 10.1038/s41534-020-00291-0.

[12] Hari Krovi. Improved quantum algorithms for linear and nonlinear differential equations. *Quantum*, 7:913, 2 2023. ISSN 2521-327X. doi: 10.22331/q-2023-02-02-913.

[13] Shi Jin, Nana Liu, and Yue Yu. Quantum simulation of partial differential equations via Schrodingerisation: technical details. 12 2022. doi: 10.1103/PhysRevA.108.032603.

[14] Albert J. Pool, Alejandro D. Somoza, Conor Mc Keever, Michael Lubasch, and Birger Horstmann. Nonlinear dynamics as a ground-state solution on quantum computers. *Physical Review Research*, 6(3):033257, 9 2024. ISSN 2643-1564. doi: 10.1103/PhysRevResearch.6.033257.

[15] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum Algorithm for Linear Systems of Equations. *Physical Review Letters*, 103(15):150502, 10 2009. ISSN 0031-9007. doi: 10.1103/PhysRevLett.103.150502.

[16] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 193–204, New York, NY, USA, 6 2019. ACM. ISBN 9781450367059. doi: 10.1145/3313276.3316366.

[17] Mayur P Bonkile, Ashish Awasthi, C Lakshmi, Vijitha Mukundan, and V S Aswin. A systematic literature review of Burgers' equation with recent advances. *Pramana*, 90(6):69, 6 2018. ISSN 0304-4289. doi: 10.1007/s12043-018-1559-4.

[18] Kristan Temme, Sergey Bravyi, and Jay M. Gambetta. Error mitigation for short-depth quantum circuits. 11 2017. doi: 10.1103/PhysRevLett.119.180509. URL http://arxiv.org/abs/1612.02058http://dx.doi.org/10.1103/PhysRevLett.119.180509.

[19] Siyuan Niu and Aida Todri-Sanial. Effects of Dynamical Decoupling and Pulse-level Optimizations on IBM Quantum Computers. 9 2022. doi: 10.1109/TQE.2022.3203153. URL http://arxiv.org/abs/2204.01471http://dx.doi.org/10.1109/TQE.2022.3203153.

Appendix

The implemented codes can be found in the following GitHub repository.