Commun. Comput. Phys.
doi: 10.4208/cicp.OA-2022-0081

Vol. **33**, No. 2, pp. 425-451
February 2023

Quantum Implementation of Numerical Methods for Convection-Diffusion Equations: Toward Computational Fluid Dynamics

Bofeng Liu[1,2,†], Lixing Zhu[1,3,†,*], Zixuan Yang[1,3,*] and Guowei He[1,2,3]

[1] *LNM, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China.*
[2] *Department of Modern Mechanics, University of Science and Technology of China, Hefei 230027, China.*
[3] *School of Engineering Sciences, University of Chinese Academy of Sciences, Beijing 100049, China.*

Received 15 March 2022; Accepted (in revised version) 8 November 2022

Abstract. We present quantum numerical methods for the typical initial boundary value problems (IBVPs) of convection-diffusion equations in fluid dynamics. The IBVP is discretized into a series of linear systems via finite difference methods and explicit time marching schemes. To solve these discrete systems in quantum computers, we design a series of quantum circuits, including four stages of encoding, amplification, adding source terms, and incorporating boundary conditions. In the encoding stage, the initial condition is encoded in the amplitudes of quantum registers as a state vector to take advantage of quantum algorithms in space complexity. In the following three stages, the discrete differential operators in classical computing are converted into unitary evolutions to satisfy the postulate in quantum systems. The related arithmetic calculations in quantum amplitudes are also realized to sum up the increments from these stages. The proposed quantum algorithm is implemented within the open-source quantum computing framework Qiskit [2]. By simulating one-dimensional transient problems, including the Helmholtz equation, the Burgers' equation, and Navier-Stokes equations, we demonstrate the capability of quantum computers in fluid dynamics.

AMS subject classifications: 68Q12, 76M20

Key words: Quantum computing, partial differential equations, computational fluid dynamics, finite difference, finite element.

[†]These authors contributed equally to this work.
[*]Corresponding author. *Email addresses:* `zlx@imech.ac.cn` (L. Zhu), `yangzx@imech.ac.cn` (Z. Yang)

1 Introduction

Fluid mechanics is one of the earliest disciplines that widely bring in numerical simulations. Von Neumann and Charney [24] were beginning to use the first programmable digital computer ENIAC for meteorology as early as the 1940s. Nevertheless, the scales of simulation in computational fluid dynamics (CFD) without any modeling are still far away from industrial-strength problems, since the computational cost exponentially dependents on the Reynolds number (Re) [48]. Slotnick et al. [41] suggested that a potential paradigm shift driven by cutting-edge computing technologies, including quantum computing, may fundamentally change the situation. Motivated by the reaching point of quantum supremacy [37] in experimental quantum computing [3], we observe a flourishing development of quantum numerical methods or quantum simulations in researches and engineering practices with various physical contexts, including fluid dynamics. In the present work, our focus is the realization of the numerical methods for partial differential equations (PDEs) governing the fluid dynamics system.

The following introduction of the previous efforts on quantum solvers of PDEs inevitably involves some discussions on the categories of quantum computing hardware since some of the quantum numerical procedures are better considered as different architectures rather than algorithms, as suggested by Kendon et al. [25]. One category of quantum computing hardware is the so-called quantum analog computing [25] or analog quantum simulator [19], which uses a controllable quantum system to investigate another much more complex system. Although it is relatively feasible for implementation, the universality of quantum simulators relies on finding a corresponding Hamiltonian, which is nontrivial for fluid dynamics or other classical systems. Specific analog quantum hardware based on quantum annealing (QA) [23] is most likely to become commercially available [33] amongst many prototypes of quantum computing systems. QA algorithm utilizes the quantum-mechanical fluctuation to tunnel through the cost barrier between local minima, and thus it is suitable for optimization problems. Ray et al. [39] converted a one-dimensional (1D) laminar channel flow problem into a quadratic unconstrained binary optimization problem via the least square method. Srivastava and Sundararaghavan [42] constructed a graph representation of the functional of a 1D elastic bar via Ising Hamiltonian on a D-Wave machine. Both attempts [39, 42] directly adopted steady-state elliptical differential equations to an discretely equivalent form that is admitted to D-Wave hardware [8]. However, the PDEs in fluid dynamics are most often non-elliptic, as our discussion later on in Section 2.1. Therefore, the QA machines are very likely to be merely used for certain sub-process rather than for the entire solution process. Zanger et al. [50] proposed a QA-based integrator for ordinary differential equations (ODE), where a heuristic minor-embedding algorithm proposed by Cai et al. [9] is employed to make the connection locally condensed. Knudsen and Mendl [26] constructed a variational continuous-variable quantum algorithm [5] to integrate an ODE. An adiabatic quantum algorithm for solving a Hermitian linear system is proposed by Subasi et al. [45], and this algorithm is experimentally implemented and tested by Wen et al. [46].

Another family of quantum computing hardware is digital quantum computers (DQCs). Unlike analog or continuous variable quantum computers, where a Hilbert space with infinite dimensions is essentially formulated by a Hamiltonian, the DQC employs quantum bits (qubits) as building blocks for its registers and spans a finite-dimensional Hilbert space. In the DQC architecture, a predefined operation, namely a quantum gate [4] that applies to one or multiple qubits, is the basic unit in a quantum circuit. Quantum gates can be systematically combined to construct complex operations. Furthermore, the Solovay-Kitaev theorem has shown that the depth and complexity of such construction is bounded [14]. Within the universal quantum computing protocol, many quantum algorithms are proposed and proven to be superior to their classic counterparts in the sense of complexity by exploiting quantum interference. A comprehensive review of quantum algorithms and circuits for basic algebraic operations, which builds the foundations of DQC-based algorithms, is presented by Childs and van Dam [13].

Via spatial discretization methods, a boundary value problem (BVP) can be discretized into a system of linear equations (LE), which is the essential ingredient in numerical methods for PDEs. In the context of fluid mechanics, the continuity equation at the limit of zero Mach number (Ma) becomes the constraint of incompressibility, which yields a Poisson equation. Steijl [43, 44] proposed a quantum-classical hybrid computing framework where the quantum Fourier transform (QFT) is involved in solving the Poisson equation in the solution process of the vortex-in-cell method. The QFT [21] shows advantages in both the complexity and required number of gates as compared to classical Fourier transform. However, the spectral method where a Fourier series is used as the basis function in spatial interpolation cannot deal with problems with complex-boundary geometries. Harrow, Hassidim, and Lloyd (HHL) firstly proposed a quantum solver for Hermitian matrices [22]. Cao et al. [10] presented the circuit design with four qubits for a 2×2 linear system. The HHL algorithm and its derivatives are further applied to Poisson equations with more flexible discretization methods, including finite elements [34] and finite differences [11]. The quantum linear solver is further refined by Berry et al. [7] and Xin et al. [47] with lower bounds of complexity. Regarding IBVPs, besides the spatial discretization, it is also necessary to apply a time marching scheme. Gaitan [18] suggested that a quantum time integrator based on the quantum amplitude estimation algorithm could be a potential candidate. Apart from solving the Navier-Stokes equations, there are some other attempts [30] in modeling a fluid dynamic system with the Lattice Boltzmann method (LBM) in DQCs.

All the aforementioned quantum algorithms involved in the solution process of PDEs are primarily suitable for Hermitian linear system. However, in the context of fluid mechanics, an anti-Hermitian nonlinear system is more common, especially in viscous flows with high convective velocity. Fillion-Gourdeau and Lorin [17] exhibited a quantum algorithm for a linear hyperbolic system with polynomial costs with respect to the input size. Much effort was exerted on the quantum algorithms for the nonlinear system in the literature. Leyton and Osborne proposed algorithms for nonlinear transformation and time marching in their preprint [28]. The resulting quantum algorithm for nonlinear systems

requires resources exponentially increasing with the integration time. Lloyd et al. [31] proposed a quantum algorithm for a nonlinear Schrödinger equation with quadratic scaling in the integration time. Recently, Liu et al. [29] utilized Carleman linearization to convert the nonlinearity into a truncated series of linear problems, which leads to an algorithm that quadratically depends on the integration time. As an alternative path, Lubasch et al. [32] proposed variational quantum algorithm for nonlinear problems and demonstrate the algorithm in a steady-state Schrödinger equation.

In this paper, we present the quantum algorithm and its implementation in DQCs for IBVPs of PDEs in fluid mechanics. Unlike most previous attempts that only move part of the numerical procedures to quantum computing, we aim to solve PDEs fully within a quantum computer. The rest of the paper is arranged as follows. In Section 2, we summarize the PDEs in fluid mechanics. Section 3 illustrates the proposed quantum algorithms for the generic form of the PDEs, followed by some numerical tests in Section 4. Finally, conclusions and discussions are drawn in Section 5.

2 Mathematical models in CFD

2.1 Governing equations and differential operators

The well-established Navier-Stokes equations [16] that consist of conservation laws of mass, momentum, and energy describe the fluid motion from the perspective of continuum mechanics. The primitive variables (i.e., the density, the velocity, and the temperature) or the conservative variables (i.e., the density, the momentum, and the total energy) are involved in the governing equations. Moreover, in more complicated fluid-related simulations (e.g., two-phase flows and combustion), one or more passive or active scalar fields (e.g., location of interface and concentration) are introduced to the system of governing equations to represent other topological, physical, and chemical evolutions. In the derivation of the governing equations in the Eulerian coordinates, the transformation of an arbitrary-shaped control volume V from Lagrangian coordinates regarding a unknown generic field ϕ is addressed by Reynolds' transport theorem, which results in the so-called material derivative or convective derivative [16], *viz.*

$$\frac{D\phi(\boldsymbol{x},t)}{Dt} \equiv \partial_t \phi + \underbrace{\boldsymbol{u}\cdot\nabla}_{\text{Convection}} \phi, \tag{2.1}$$

where D/Dt represents the material derivative, $\boldsymbol{u}$ is the local velocity, and ∂_t represents the temporal partial derivative in the Eulerian reference frame. Eq. (2.1) depicts the rate of change of ϕ in the infinitesimal control volume, and therefore, the convection operator exits in almost every governing equation in CFD.

The rate of change in the control volume is balanced by some volume integrals and surface integrals of the generic fluxes, which is further recast as a volume integral via

divergence theorem as follows.

$$\int_{S=\partial V} \boldsymbol{F}(\phi)\cdot\boldsymbol{n}\, dS = \int_V \nabla\cdot\boldsymbol{F}(\phi)\, dV, \tag{2.2}$$

where S is the surface of the control volume V and $\boldsymbol{F}(\phi)$ is the generic fluxes. To close the governing system, it is necessary to involve constitutive models to explicitly express the flux with respect to the unknown field. The transport of mass through diffusion is governed by Fick's law, where the flux of concentration linearly depends on its gradient. In an incompressible Newtonian flow, the deviatoric part of the total stress is linearly dependent on the symmetric gradient of the velocity. The most common model for heat conduction is Fourier's law, where the heat flux is defined as the temperature gradient with a constant coefficient. In summary, the generic flux most regularly holds a linear relation with the gradient. Combined with the divergence operation on the right-hand-side (RHS) of Eq. (2.2), the integrand $\nabla\cdot\boldsymbol{F}$ becomes the Laplacian or diffusion $\nabla\cdot\nu\nabla\phi$, which is the other significant differential operators. The coefficient ν owns different interpretations in various physical contexts, like the viscosity in momentum conservation or the heat diffusivity in energy conservation.

With the above brief discussion on the derivation of the Navier-Stokes equations, it is understood that the two most significant differential operators are: i) the convective operator $\boldsymbol{u}\cdot\nabla$, and ii) the diffusion operator $\nabla\cdot\nu\nabla$. The former one is the consequence of the Eulerian reference frame, which is a common setting in the mathematical models for fluids. The latter one perceives different physics in various contexts. In the fluid motions, the viscous effect essentially represents the frictional force between molecules or atoms in the microscope, and its macroscopic contribution is the resistance of the shear motion. In thermal dynamics, the heat conduction transfers energy through microscopic collision and movement of particles. In mass transfer, the diffusion of a passive or active scalar field comes from the random walk of particles.

2.2 Discretizations and matrix Forms

2.2.1 Non-conservative Form

Following the aforementioned discussion, we present the discretization of an unsteady convection-diffusion equation which includes the two representative differential operators in the mathematical models of fluid dynamics. In a unit 1D domain $\Omega=[0,1]$, a scalar solution field ϕ is governed by the following convection-diffusion equation in advective form.

$$\partial_t\phi + u(x)\phi_{,x} = \nu\phi_{,xx} + f(x) \quad \text{in } \Omega\times[0,T], \tag{2.3}$$

where $u(x)$ is the prescribed velocity, ν is the viscosity, and f is the source term. Specifically, if the velocity is the solution field, Eq. (2.3) becomes the celebrated 1D viscous Burgers' equation. For the well-posedness of the problem, we apply Dirichlet boundary conditions on both ends of the 1D domain. The initial condition ϕ_0 and boundary

conditions are

$$\phi(x,0)=\phi_0(x) \quad \text{in } \Omega\times\{0\}, \tag{2.4}$$

$$\phi=g(x) \quad \text{on } \Gamma=\partial\Omega\times[0,T]. \tag{2.5}$$

To develop a numerical scheme for solving Eq. (2.3), we uniformly decompose the computational domain Ω into m segments with $m+1$ grid points. Consequently, the domain is discretized into a set of the grid points $\Omega^h=\{x_0,x_1,\cdots,x_m\}$, the spatial coordinates of the grid points are $x_i=i/m$ with $i\in\{0,1,\cdots,m\}$ being the spatial index, and the uniform grid spacing is $\Delta x=1/m$. By utilizing the second-order central difference of first- and second- order derivatives in space and forward Euler scheme for time advancement, we arrive at the following fully discrete form of Eq. (2.3)

$$\frac{\phi_i^{n+1}-\phi_i^n}{\Delta t}+\frac{u_i}{2\Delta x}(\phi_{i+1}^n-\phi_{i-1}^n)=\frac{\nu}{\Delta x^2}(\phi_{i+1}^n+\phi_{i-1}^n-2\phi_i^n)+f_i, \tag{2.6}$$

where $f_i=f(x_i)$, the viscosity ν is a constant in the 1D domain, $\Delta t=t_{n+1}-t_n$ is the time increment, and the superscript "n" of ϕ^n indicate the time spot of the unknown field $\phi(x,t_n)$. The fully discrete form given by Eq. (2.6) can be rewritten in a matrix form as

$$\boldsymbol{\Phi}^{n+1}=\underbrace{\boldsymbol{A}\boldsymbol{\Phi}^n}_{\text{Amplification}}\quad\underbrace{+\boldsymbol{S}}_{\text{Source}}\quad\underbrace{+\boldsymbol{B}}_{\text{BCs}}, \tag{2.7}$$

where $\boldsymbol{\Phi}=[\phi_1,\phi_2,\cdots,\phi_{m-1}]^T$ is the discrete solution fields excluding both ends with Dirichlet boundary condition, $\boldsymbol{A}$ is the classical amplification matrix, and $\boldsymbol{S}=\Delta t[f_1,f_2,\cdots,f_{m-1}]^T$ is the discrete source vector. Specifically, $\boldsymbol{A}$ covers the contribution from the convection and diffusion operators with the following form,

$$\boldsymbol{A}=\boldsymbol{I}_{(m-1)}-\frac{\Delta t}{2\Delta x}\boldsymbol{C}_{\text{ADV}}+\frac{\nu\Delta t}{\Delta x^2}\boldsymbol{D}, \tag{2.8}$$

where $\boldsymbol{I}$ is an identity matrix; $\boldsymbol{C}$ and $\boldsymbol{D}$ represent the differential algebraic matrix of convection and diffusion, respectively. Matrices $\boldsymbol{C}$ and $\boldsymbol{D}$ are defined as

$$\boldsymbol{C}_{\text{ADV}}=\begin{bmatrix}u_1 & & & \\ & u_2 & & \\ & & \ddots & \\ & & & u_{m-1}\end{bmatrix}\begin{bmatrix}0 & 1 & & \\ -1 & 0 & 1 & \\ & & \ddots & \\ & & -1 & 0\end{bmatrix} \quad\text{and}\quad \boldsymbol{D}=\begin{bmatrix}-2 & 1 & & \\ 1 & -2 & 1 & \\ & & \ddots & \\ & & 1 & -2\end{bmatrix}. \tag{2.9}$$

Specifically, $\boldsymbol{C}$ is a gradient matrix left multiplied by a diagonal matrix to account for the inhomogeneous convective velocity. The size of the linear system is $(m-1)\times(m-1)$ since the solution fields at both ends are determined by the boundary condition. Their influence on the interior points of the computational domain is presented by vector $\boldsymbol{B}_{\text{ADV}}$,

which consists of the contribution from both differential operators to enforce the Dirichlet boundary condition, *viz.*

$$\boldsymbol{B}_{\mathrm{ADV}} = -\frac{\Delta t}{2\Delta x}\begin{bmatrix} u_1 g_0 \\ \mathbf{0}_{(m-3)} \\ u_{m-1} g_m \end{bmatrix} + \frac{\nu \Delta t}{\Delta x^2}\begin{bmatrix} g_0 \\ \mathbf{0}_{(m-3)} \\ g_m \end{bmatrix}, \tag{2.10}$$

where $\mathbf{0}_{(m-3)}$ is the zero vector with $(m-3)$ components.

2.2.2 Conservative form

The convection term $\boldsymbol{u}\cdot\nabla\phi$ arising from the material derivative in Eq. (2.1), is one of the most notorious terms in a typical hyperbolic system in fluid mechanics. Since this term is the source of many issues that are widely concerned in the CFD community, including the conservation properties of the discrete system, stability in high Re number, and the closure problem in turbulence modeling, many alternative representations are proposed within various contexts of discretization procedures. Most of them are derived via tensor identities and the conservation law of mass (or the divergence-free constraint in incompressible flows). These alternative forms are equivalent to each other in the continuum but, in principle, not equivalent in the discrete function space. Morinishi [35] has identified that only two forms are independent amongst all the alternative forms of the convection term, including advective, divergence, skew-symmetric, and rotational forms. Thus, rather than summarize all possible numerical treatments of the convection term, we describe the detailed procedures to implement a typical conservative form (i.e., divergence form [20]) to illustrate the flexibility of the proposed quantum algorithm in practical problems.

By replacing the advective form of the convection term in Eq. (2.3) with the divergence form (i.e., $u(x)\phi_{,x} \to (u(x)\phi)_{,t}$), the convection-diffusion equation becomes

$$\partial_t\phi + (u(x)\phi)_{,x} = \nu\phi_{,xx} + f(x) \quad \text{in } \Omega\times[0,T]. \tag{2.11}$$

Note that Eq. (2.11) is only equivalent with Eq. (2.3) when the prescribed velocity field is divergence-free. By applying the identical discretization strategy as that in Section 2.2.1, the fully discrete form is

$$\frac{\phi_i^{n+1}-\phi_i^n}{\Delta t} + \frac{1}{2\Delta x}(u_{i+1}^n\phi_{i+1}^n - u_{i-1}^n\phi_{i-1}^n) = \frac{\nu}{\Delta x^2}(\phi_{i+1}^n + \phi_{i-1}^n - 2\phi_i^n) + f_i. \tag{2.12}$$

The matrix form defined in Eq. (2.7) still holds except for the convection term in divergence form, which becomes

$$\boldsymbol{C}_{\mathrm{DIV}} = \begin{bmatrix} 0 & 1 & & \\ -1 & 0 & 1 & \\ & & \ddots & \\ & & -1 & 0 \end{bmatrix}\begin{bmatrix} u_1 & & & \\ & u_2 & & \\ & & \ddots & \\ & & & u_{m-1} \end{bmatrix}. \tag{2.13}$$

The corresponding vector for enforcement of Dirichlet boundary conditions becomes

$$\boldsymbol{B}_{\mathrm{DIV}} = -\frac{\Delta t}{2\Delta x}\begin{bmatrix} u_0 g_0 \\ \mathbf{0}_{(m-3)} \\ u_m g_m \end{bmatrix} + \frac{\nu \Delta t}{\Delta x^2}\begin{bmatrix} g_0 \\ \mathbf{0}_{(m-3)} \\ g_m \end{bmatrix}. \tag{2.14}$$

To close the discussion on the governing equation and discretization, we note that although the forward Euler method is undoubtedly an unsophisticated choice from the palette of numerical methods, it indeed serves our purpose as an early validation of quantum computing in CFD as forward time central space (FTCS) represents a wide potential of various discretization strategies. It can be readily replaced by a large family of explicit time marching schemes.

3 Quantum algorithms

The fully discrete formulation given by Eq. (2.7) holds the classic form of amplification matrix for stability analysis. With the assumptions that the cell Peclet number $\mathrm{Pe} = u\Delta x/2\nu$ is sufficiently small and the Courant-Friedrichs-Lewy (CFL) condition is satisfied for both convection and diffusion operators, the simulation is robust in classical computers, as will be shown later. The scope of this work, however, is not the analysis of numerical methods but rather the realization in quantum computing. Specifically, the implementation of arithmetic and linear algebraic operations in Eq. (2.7) in DQCs via standard quantum gates remains a challenge. The present study aims at addressing this issue and providing a general guideline on the use of quantum computers for numerical simulations of fluid flows.

Each single-step time-marching process in quantum computers consists of four stages, namely, encoding, amplifying, applying source terms, and incorporating boundary conditions, as shown in Fig. 1. The schematic diagram shown in Fig. 1 only depicts the skeleton of the algorithm and does not represent the actual usage of ancilla qubits.

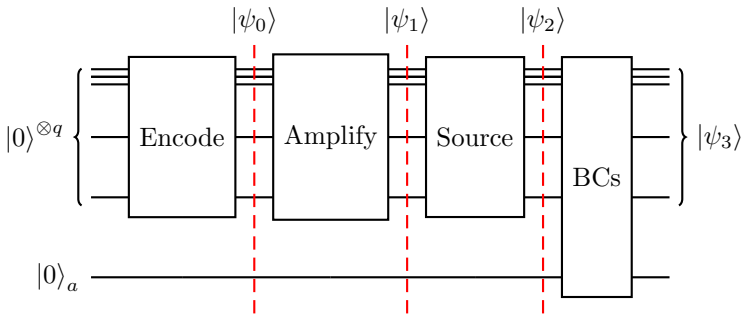

Figure 1: Overall stages of the proposed quantum algorithm for a single-step of explicit time-marching in an IBVP problem: encoding (Section 3.1), amplifying (Section 3.2.1), applying source terms (Section 3.2.2), and incorporating boundary conditions (Section 3.2.3). The ancilla qubits are denoted with a subscript $|\bullet\rangle_a$.

The quantum circuits for different stages are delineated in the following subsections, with related postulates and principles in quantum computing systems.

3.1 Encoding and initialization

The basic unit of quantum computing hardware is a quantum bit, which is also known as the qubit. Unlike the classic bit or binary digit that only takes the value of either 0 or 1, a qubit defines a Hilbert space $\mathcal{H}(\mathbb{C}^2)$ spanned by two basis states (i.e., $|0\rangle$ and $|1\rangle$). The state vector $|\psi\rangle=[\alpha,\beta]^T$ of the qubit is a linear combination of these two states, *viz.*

$$|\psi\rangle=\alpha|0\rangle+\beta|1\rangle=\alpha\begin{bmatrix}1\\0\end{bmatrix}+\beta\begin{bmatrix}0\\1\end{bmatrix}=\begin{bmatrix}\alpha\\\beta\end{bmatrix}, \tag{3.1}$$

where $|\alpha|^2$ and $|\beta|^2$ can be interpreted as the probabilities of the qubit resting in either state when being measured. This stochastic interpretation reflects a postulate in quantum mechanics that any quantum system is described by its state vector, which is a unit vector in the system's state space [36]. Eq. (3.1) leads to the superiority of a qubit that it stores two real or complex numbers in a vector form. With an increasing number of qubits n, the dimension of the Hilbert space increases exponentially as $\mathcal{H}(\mathbb{C}^{2^n})$. One of the state-of-the-art direct numerical simulations (DNS) on forced isotropic turbulence utilizes a grid of 8192^3 [49] to resolve the flow physics to the Kolmogorov scale, of which the memory requirement almost hits the limitation of classical high-performance computers. Via the spanned Hilbert space, its velocity field only takes 41 qubits to store in quantum computers. The exponential growth of the storage capacity of qubits brings in the most alluring attribute of quantum computing, that is, the guaranteed acceleration of space complexity of the quantum algorithm.

To take advantage of quantum computing in the space complexity, we need to encode the solution field in the vector space represented by a quantum register that consists of multiple qubits. The quantum-equivalent operation of initializing vector variables in classical computing is to prepare a specific state vector of the quantum register. To obtain the quantum state representing an arbitrary vector (i.e., the initial condition), universal quantum gates are the prerequisite and are considered at the early development of quantum computing [15]. In practice, we start with a quantum register where all qubits are $|0\rangle$ and apply a series of unitary operators to prepare the desired state vector $|\phi_1\rangle$ in Fig. 1 as follows

$$|0\rangle^{\otimes(q+1)}\xrightarrow{\text{Encode}}|\psi_0\rangle=\frac{1}{\sqrt{\|\boldsymbol{\Phi}^n\|^2+\|\boldsymbol{S}\|^2}}\left(|0\rangle\otimes\sum_{k=1}^{m-1}\Phi_k^n|k\rangle+|1\rangle\otimes\sum_{k=1}^{m-1}S_k|k\rangle\right), \tag{3.2}$$

where $q=\log_2(m-1)$ is the number of qubits used to store the solution vector $\boldsymbol{\Phi}^n$ in Eq. (2.7). With one additional qubit, the spanned vector space can also store the discrete source vector $\boldsymbol{S}$. In other words, with $\log_2(m-1)+1$ qubits, the state vector $|\psi_0\rangle$ stores

both the normalized current solution field $\boldsymbol{\Phi}_n$ and the source terms $\boldsymbol{S}$. A straightforward method for state preparation introduced in [6] is achieved by a series of controlled rotation operators about the y axis (C-R_y gate). The angles in the C-R_y gates are determined by a system of nonlinear equations. Shende et al. [40] proposed a more efficient quantum circuit that achieves the initialization with an upper bound of complexity (i.e., number of CNOT gates). Since the arrangement of the quantum circuits is case-specific, the comparisons of the state vectors prepared via both algorithms are conducted within the numerical tests in Section 4.

3.2 Arithmetic of quantum amplitudes

According to the matrix form of the discretized equation given by Eq. (2.7), the three stages after the encoding of the state vector incorporate two linear algebraic operations, including matrix-vector multiplications and vector or component addition. After the solution field is encoded in the spanned Hilbert space of the quantum register (i.e., a state vector of amplitudes) as in Eq. (3.2), quantum circuits need to be designed to manipulate the encoded state vector $|\phi_0\rangle$ for the realization of the linear algebraic operations. The manipulation of the state vector in a quantum system follows another postulate in quantum mechanics that the evolution of a closed quantum system is described by a unitary transformation. Nevertheless, most of the classical system, like fluid mechanics, involves non-unitary operations. In the following content, we present how to accomplish those non-unitary operations via the combination of unitary ones.

3.2.1 Matrix-vector multiplication

After the encoding stage introduced in Section 3.1, we need to evolve the encoded state vector $|\psi_0\rangle$ to the state vector $|\psi_1\rangle$ representing the amplified current solution $\boldsymbol{A}\boldsymbol{\Phi}^n$. As mentioned above, this manipulation needs to be achieved by a unitary matrix or a linear combination of several unitary matrices so that the indicated evolution is admissible in a quantum system. From Eq. (2.8), it is understood that the amplification matrix $\boldsymbol{A}$ is the linear combination of three parts: the identity matrix $\boldsymbol{I}$, the convection matrix $\boldsymbol{C}$ and the diffusion matrix $\boldsymbol{D}$.

The discrete Laplace operator $\boldsymbol{D}$ is a symmetric Toeplitz matrix due to the use of a uniform grid. In a more generic case of structured mesh with uneven grid spacing, the discrete Laplace operator loses the attribute of the Toeplitz matrix but preserves the symmetry. Following the transformation suggested by Xin et al. [47] for linear systems, we first normalize $\boldsymbol{D}$ and construct a unitary matrix from the discrete Laplace operator as

$$\boldsymbol{D}^u = \frac{1}{\|\boldsymbol{D}\|}\boldsymbol{D} + \mathrm{i}\sqrt{\boldsymbol{I} - \frac{1}{\|\boldsymbol{D}\|^2}\boldsymbol{D}\cdot\boldsymbol{D}}, \tag{3.3}$$

where $\mathrm{i}=\sqrt{-1}$ is the imaginary unit, $\|\boldsymbol{D}\|$ is the Frobenious norm of the matrix, and $\sqrt{(\bullet)}$ is the matrix square root. It can be shown that as long as $\boldsymbol{D}$ is symmetric, $\boldsymbol{D}^u$ is a unitary

matrix. Moreover, the real part of $\boldsymbol{D}^u$ remains identical as the normalized $\boldsymbol{D}$. Technically one may need to eliminate the effect of the imaginary part via its conjugate counterpart as $\boldsymbol{D}\boldsymbol{\Phi}^n/\|\boldsymbol{D}\| = \boldsymbol{D}^u\boldsymbol{\Phi}^n + (\boldsymbol{D}^u)^*\boldsymbol{\Phi}^n$. In practice, we can simply ignore the imaginary part in the state vector.

The discrete convection operator $\boldsymbol{C}$ is slightly trickier for implementation in the quantum circuits. Even with the homogeneous velocity field (i.e., $u(x)=u$), we are left with a skew-symmetric matrix. Therefore, the construction of unitary matrix shown in Eq. (3.3) is not applicable for the convection operator $\boldsymbol{C}$. To overcome this issue, we decompose the convection operator as follows,

$$\boldsymbol{C}_{\mathrm{ADV}}\boldsymbol{\Phi}_n = \underbrace{\begin{bmatrix} u_1 & & & \\ & u_2 & & \\ & & \ddots & \\ & & & u_{m-1} \end{bmatrix}}_{\mathrm{diag}(\boldsymbol{u})} \left(\begin{bmatrix} \phi_2 \\ \phi_3 \\ \vdots \\ \phi_{m-1} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 \\ \phi_1 \\ \vdots \\ \phi_{m-3} \\ \phi_{m-2} \end{bmatrix} \right), \tag{3.4}$$

$$\boldsymbol{C}_{\mathrm{DIV}}\boldsymbol{\Phi}_n = \begin{bmatrix} u_2 & & & & \\ & u_3 & & & \\ & & \ddots & & \\ & & & u_{m-1} & \\ & & & & 0 \end{bmatrix} \begin{bmatrix} \phi_2 \\ \phi_3 \\ \vdots \\ \phi_{m-1} \\ 0 \end{bmatrix} - \begin{bmatrix} 0 & & & & \\ & u_1 & & & \\ & & \ddots & & \\ & & & u_{m-3} & \\ & & & & u_{m-2} \end{bmatrix} \begin{bmatrix} 0 \\ \phi_1 \\ \vdots \\ \phi_{m-3} \\ \phi_{m-2} \end{bmatrix}, \tag{3.5}$$

where the dimension of the linear system is $m-1$. Unlike the common practice of implementing a non-unitary matrix via decomposition into a linear combination of multiple unitary matrices, we rearrange the vectors on the RHS of both forms in Eqs. (3.4) and (3.5) to accommodate for the numerical differences. Such re-arrangement requires auxiliary qubits for extra storage in the state vector and extra work in the encoding stage, while reducing the multiplier into a diagonal matrix. The resulting diagonal matrix $\mathrm{diag}(\boldsymbol{u})$ can be converted into a unitary matrix following the transformation given in Eq. (3.3), and the vector addition is addressed in Section 3.2.2.

An alternative approach to implement the amplification matrix $\boldsymbol{A}$ in quantum circuits is based on the singular value decomposition (SVD), where $\boldsymbol{A}$ is factorized into the following form:

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^*. \tag{3.6}$$

Both $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices, the superscript $*$ indicates the conjugate transpose, and $\boldsymbol{\Sigma}$ is a diagonal matrix comprising the singular values of the square matrix $\boldsymbol{A}$. In particular, since the singular values are the square roots of the eigenvalues of the normal matrix $\boldsymbol{A}^*\boldsymbol{A}$ or $\boldsymbol{A}\boldsymbol{A}^*$, $\boldsymbol{\Sigma}$ is a real diagonal matrix. Therefore, the unitary transformation in Eq. (3.3) is applicable to $\boldsymbol{\Sigma}$, which yields the unitarized amplification matrix $\boldsymbol{A}^u$ as follows

$$\boldsymbol{A}^u = \boldsymbol{U}\boldsymbol{\Sigma}^u\boldsymbol{V}^*, \quad \text{where} \quad \boldsymbol{\Sigma} \xrightarrow{Eq.\ (3.3)} \boldsymbol{\Sigma}^u. \tag{3.7}$$

It is worth noticing that the resulting matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ from SVD are real and orthogonal when $\boldsymbol{A}$ is a real square matrix as is the case in simulations of fluid dynamics. Therefore, the real part of the normalized amplification matrix is preserved in the unitarization process as $\Re(\boldsymbol{A}^u)=\boldsymbol{A}/\|\boldsymbol{A}\|$. Compared to specific treatments shown in Eq. (3.3) for symmetric Toeplitz matrices and Eqs. (3.4) and (3.5) for anti-symmetric matrices, the SVD-based unitary process in Eq. (3.7) is capable of unitarizing a real amplification matrix discretized from a general fluid dynamics system.

The last component in the amplification matrix is an identity matrix $\boldsymbol{I}$, which is unitary by default. In the implementation on the Qiskit simulator, we treat $\boldsymbol{I}$, $\boldsymbol{C}_{\text{ADV}}$ or $\boldsymbol{C}_{\text{DIV}}$, and $\boldsymbol{D}$ with coefficients together and embed their unitary counterparts $\boldsymbol{A}^u$ into quantum circuit after unitary transformation defined in Eq. (3.3)

$$|\psi_0\rangle \xrightarrow{\boldsymbol{A}^u} |\psi_1\rangle = \frac{1}{\sqrt{\|\boldsymbol{A}\boldsymbol{\Phi}^n\|^2+\|\boldsymbol{S}\|^2}}\left(|0\rangle\otimes\sum_{k=1}^{m-1}(\boldsymbol{A}\boldsymbol{\Phi}^n)_k|k\rangle+|1\rangle\otimes\sum_{k=1}^{m-1}S_k|k\rangle\right). \tag{3.8}$$

The contribution from the convection operator is accounted for separately with additional vector addition, and the details of the quantum realization are presented in Section 4.2.

3.2.2 Vector addition

Following the encoding introduced in Section 3.1, we need $q=\log_2(m-1)$ qubits to store the entire solution vector $\boldsymbol{\Phi}^n$ in Eq. (2.7). As shown in Fig. 2, the quantum state $|\psi_1\rangle$ encoded with the matrix-vector multiplication results $\boldsymbol{A}\boldsymbol{\Phi}^n$ and the source vector $\boldsymbol{S}$ is manipulated by a Hadamard gate to calculate $\boldsymbol{A}\boldsymbol{\Phi}^n+\boldsymbol{S}$ as follows,

$$|\psi_1\rangle \xrightarrow{\mathbf{H}} |\psi_2\rangle = \frac{1}{\sqrt{2}\sqrt{\|\boldsymbol{A}\boldsymbol{\Phi}^n\|^2+\|\boldsymbol{S}\|^2}}\left(|0\rangle\otimes\sum_{k=1}^{m-1}(\boldsymbol{A}\boldsymbol{\Phi}^n+\boldsymbol{S})_k|k\rangle+|1\rangle\otimes\sum_{k=1}^{m-1}(\boldsymbol{A}\boldsymbol{\Phi}^n-\boldsymbol{S})_k|k\rangle\right), \tag{3.9}$$

where the subscript $(\bullet)_k$ denotes the k^{th} component of the vector, and $|k\rangle$ denotes the k^{th} basis in $\mathcal{H}(\mathbb{C}^{2^q})$. Both the amplified converged solution $\boldsymbol{A}\boldsymbol{\Phi}^n$ and the source term $\boldsymbol{S}$ have the dimension of $m-1$. Therefore, the quantum state $|\psi_1\rangle=[\boldsymbol{A}\boldsymbol{\Phi}^n,\boldsymbol{S}]^T$ before the vector addition is stored in a quantum register with a total of $q+1$ qubits. Via a Hadamard gate $\mathbf{H}$ on the highest-order qubit, two vectors are superposed, resulting in an equivalent operation as addition.

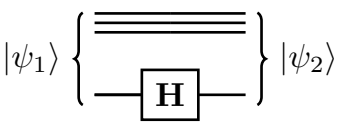

Figure 2: The quantum circuit for vector addition in the stage of adding source terms.

3.2.3 Component addition

The fully discrete matrix form of the governing equation given by Eq. (2.7) converts the incorporation of the boundary condition as an correction vector $\boldsymbol{B}$. By adding the correction vector $\boldsymbol{B}$ and the source vector $\boldsymbol{S}$ together, one can merge the stages of adding source terms and incorporating boundary conditions to arrive at the state vector $|\psi_3\rangle$ right after the operations of the matrix-vector multiplication and the vector addition. Nevertheless, for the problems with a local source, embedding the source term and boundary conditions into vectors with the same length as the linear system increases the space complexity of quantum algorithms. To economically add the correction vector of boundary conditions or local source terms into the amplified solution field, we hereby present the quantum circuit for component addition. We introduce an ancilla qubit to the quantum circuits for the purpose of the element-wise operation. With two controlled-NOT (CNOT) gates, two components in the state vector that are meant to be summed are exposed, and the following Hadamard gate superposes these two components, as shown in Fig. 3. The evolution of the state vector in the stage of incorporating boundary conditions is:

$$
\begin{aligned}
|\psi_2^*\rangle &= \frac{1}{\sqrt{\|\boldsymbol{\Phi}_2^n\|^2+B^2}}|0\rangle\otimes\left(\sum_{k=1}^{m-1}(\boldsymbol{\Phi}_2^n)_k|k\rangle+B|m\rangle\right)\xrightarrow{\text{CNOT, }\mathbf{H}}\\
|\psi_3^*\rangle &= \frac{1}{\sqrt{2\|\boldsymbol{\Phi}_2^n\|^2+2B^2}}\left[|0\rangle\otimes\left(\sum_{k=1}^{m-2}(\boldsymbol{\Phi}_2^n)_k|k\rangle+((\boldsymbol{\Phi}_2^n)_{m-1}+B)|m-1\rangle\right)\right.\\
&\quad\left.+|1\rangle\otimes\left(\sum_{k=1}^{m-2}(\boldsymbol{\Phi}_2^n)_k|k\rangle+((\boldsymbol{\Phi}_2^n)_{m-1}-B)|m-1\rangle\right)\right],
\end{aligned}
\tag{3.10}
$$

where $\boldsymbol{\Phi}_2^n=\boldsymbol{A}\boldsymbol{\Phi}^n+\boldsymbol{S}$ indicates the state vector after the stage of applying the source term and B denotes the correction at the $(m-1)$th component due to boundary condition.

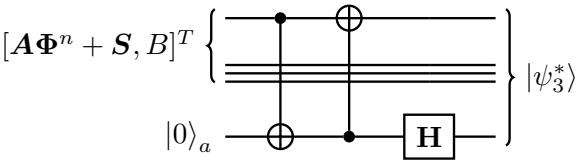

Figure 3: The quantum circuit for component addition in the stage of incorporating boundary conditions.

4 Numerical tests

In this section, we evaluate the proposed quantum algorithm using a series of problems with manufactured solutions for the convenience of validation. These 1D problems cover a wide range of governing equations, including the Helmholtz equation and the inviscid

Burgers' equation. Also presented is the case of a convergent-divergent nozzle that is governed by the compressible Navier-Stokes equations. The proposed quantum algorithm is implemented in Qiskit [2], an open-source software development kit for quantum computing, which allows us to simulate a DQC in the level of quantum circuits. The employed "statevector" backend enables the perfect readout of the quantum state vector without any noise, which allows us to validate the results stage by stage. In other words, we assume there exists an *oracle* that is able to efficiently embed the current solution field from the state vector into the next-step quantum circuits for nonlinear problems. Such oracle is also assumed in the discussion of quantum time integrator for Navier-Stokes equations [18] and the communication between classical and quantum computers in the hybrid algorithm [30]. It is worth noticing that the purpose of the present numerical tests is to exhibit the potential of quantum computers to simulate fluid dynamics. Thus it is critical to validate whether these algorithms can produce replicas of classical numerical procedures. The stability of the algorithms in mathematically non-smooth problems, like shockwaves, is beyond the scope of this work.

4.1 Helmholtz equation

In a 1D open-bounded domain $\Omega=(0,1)$, we have a Helmholtz equation of an unknown scalar field ϕ with homogeneous Dirichlet boundary conditions at both ends as

$$\partial_t\phi=\nu\nabla\cdot\nabla\phi \quad \text{in } \Omega\times[0,T], \tag{4.1}$$

$$\phi=0 \quad \text{on } \Gamma\times[0,T]. \tag{4.2}$$

With the applied boundary conditions, the steady-state solution of Eq. (4.1) is simply zero across the entire domain. The computational domain is evenly divided into 17 cells (or 18 nodes). Only 16 internal nodes are considered in the linear system, which needs 4 qubits to represent the discrete solution field. The initial condition is a sinusoidal wave function $\phi_0=\phi(x,0)=\sin(2\pi x)$. To initialize the computation, we need to prepare the state vector with the discrete initial condition $\Phi_i=\phi_0(x_i)$. Two quantum circuits following the two different methods introduced in Section 3.1 are implemented and tested. Fig. 4 presents the quantum circuits following the method in [6] where only C-R_y gates are used and corresponding angles are determined by a series of nonlinear equations. In Fig. 5, the quantum circuits making use of the method in [40] is presented, where only two-qubit gates (i.e., CNOT, R_y and R_z) are utilized. Both circuits in Figs. 4 and 5 are implemented in Qiskit and applied to the quantum register with all zero states, namely, $|0\rangle^{\otimes 4}$. The initialized quantum state vectors from both circuits are plotted in Fig. 6. We notice that Shende's algorithm [40] provides a more accurate representation of the initial condition. Consequently, this method is adopted in all the numerical tests in this work.

Due to the homogeneity in the PDE and the boundary condition, the corresponding terms $\boldsymbol{S}$ and $\boldsymbol{B}$ in the FTCS discretizations in Eq. (2.7) are dropped and so is the convection matrix $\boldsymbol{C}_{\text{onv}}$. Consequently, the FTCS scheme for this problem is as simple as a

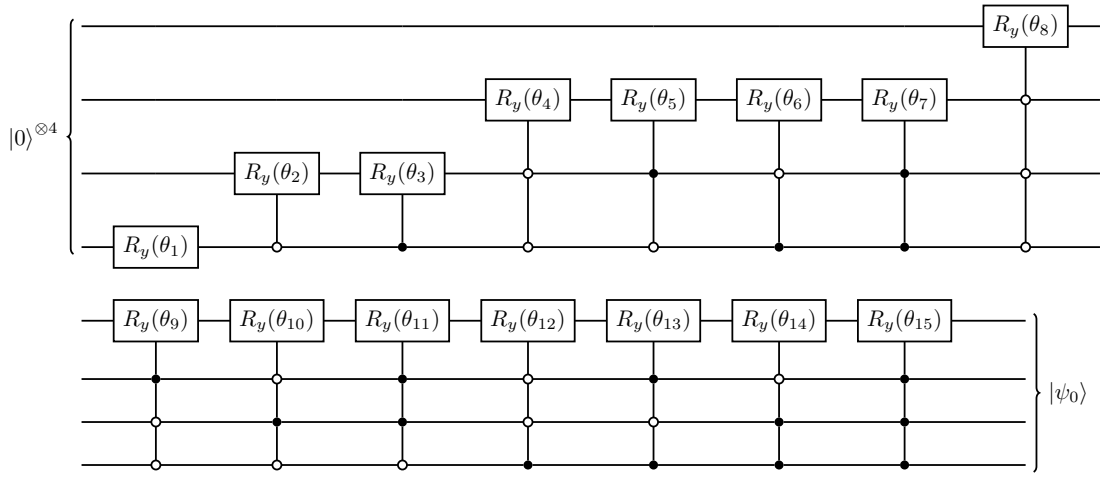

Figure 4: Quantum circuits for the state preparation in [6]. The angles in R_y gates are: $\theta_1=4.76$, $\theta_2=1.43$, $\theta_3=1.63$, $\theta_4=2.09$, $\theta_5=0.839$, $\theta_6=3.98$, $\theta_7=5.24$, $\theta_8=2.26$, $\theta_9=1.68$, $\theta_{10}=1.38$, $\theta_{11}=0.672$, $\theta_{12}=2.47$, $\theta_{13}=1.76$, $\theta_{14}=1.46$, $\theta_{15}=0.984$.

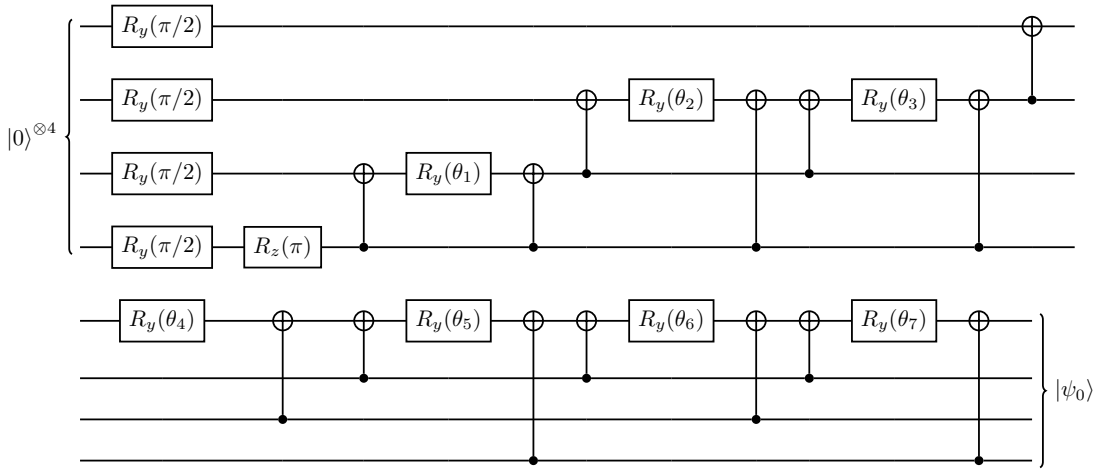

Figure 5: Quantum circuits for the state preparation in [40]. The angles in R_y gates are: $\theta_1=-0.119$, $\theta_2=0.632$, $\theta_3=-0.0991$, $\theta_4=2.98$, $\theta_5=0.444$, $\theta_6=-0.0583$, $\theta_7=-0.098$.

matrix-vector multiplication $\boldsymbol{\Phi}^{n+1}=\boldsymbol{A}\boldsymbol{\Phi}^n$, where $\boldsymbol{A}=\boldsymbol{I}-\nu\Delta t/\Delta x^2\boldsymbol{D}_{\text{iff}}$. Via Eq. (3.3), we convert the amplification matrix $\boldsymbol{A}$ into its unitary counterpart $\boldsymbol{A}^u$, of which the real part is essentially the normalized $\boldsymbol{A}$. To illustrate the implementations in quantum computers, we present the quantum circuits for a case with 5 cells (4 inner nodes encoded using 2 qubits) in Fig. 7. The state vector $|\psi_1\rangle$ encoded in a 4-qubit register is evolved by $\boldsymbol{A}^u$ and

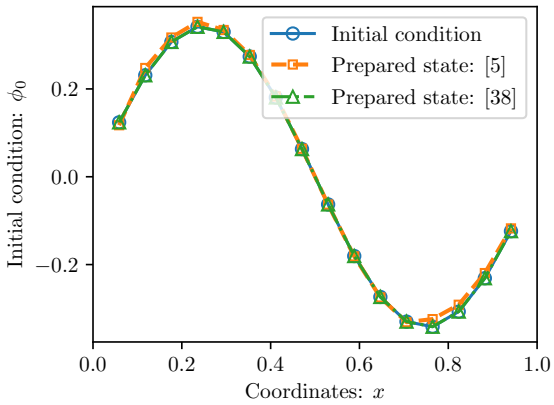

Figure 6: The initialized state vectors via methods in [6, 40].

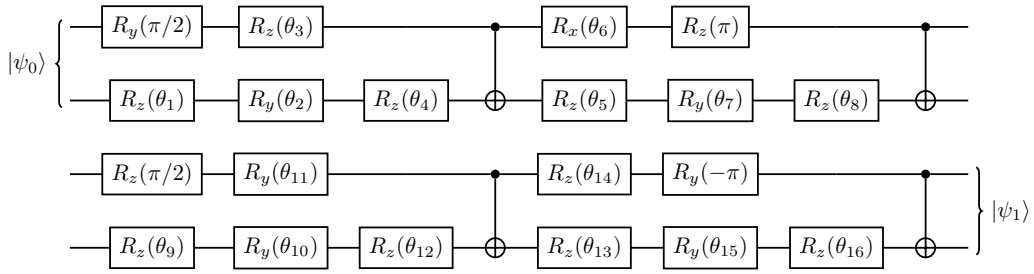

Figure 7: The quantum circuit of the amplifying stage for Helmholtz equation discretized by four inner grid points: Two qubits are utilized to encode the solution field and the time step is $\Delta t = 0.2\Delta x^2/\nu$. The angles in R_y and R_z gates are: $\theta_1 = 2.97$, $\theta_2 = 0.403$, $\theta_3 = 1.33$, $\theta_4 = 1.75$, $\theta_5 = 0.0356$, $\theta_6 = -0.239$, $\theta_7 = 1.49$, $\theta_8 = -1.97$, $\theta_9 = -0.16$, $\theta_{10} = 1.94$, $\theta_{11} = -0.0674$, $\theta_{12} = 1.14$, $\theta_{13} = -3.08$, $\theta_{14} = -1.81$, $\theta_{15} = 1.54$, $\theta_{16} = 0.396$.

the resulting state vector $|\psi_2\rangle$ is

$$|\psi_1\rangle = \frac{1}{\|\mathbf{\Phi}^n\|}\sum_{k=0}^{15}\Phi_k^n|k\rangle \quad \xrightarrow{A^u} \quad |\psi_2\rangle = \frac{1}{\|\mathbf{\Phi}^n\|}\sum_{k=0}^{15}\left(\frac{1}{\|\boldsymbol{A}\|}\Phi_k^{n+1} + i(\Im[\boldsymbol{A}^u]\mathbf{\Phi}^n)_k|k\rangle\right), \tag{4.3}$$

where $|k\rangle$ denotes the k^{th} one among 16 bases. To recover the time-evolved solution $\mathbf{\Phi}^{n+1}$, we need the L_2 norm of the amplification matrix and solution field. The former one is an a priori knowledge for a linear problem after discretization, while the latter one requires measurement or equivalent operations after each time-marching step. We implement the proposed algorithm in Qiskit and simulate it with "statevector" backend where quantum circuits are ideal without any error in readout and encoding. The classical algorithm shown in Eq. (2.6) is also implemented as the baselines of solutions. The time increment is uniformly set as $\Delta t = 0.1\Delta x^2/\nu$. The results from the quantum simulator are compared with those from a classical computer in Fig. 8, where solution fields at various snapshots

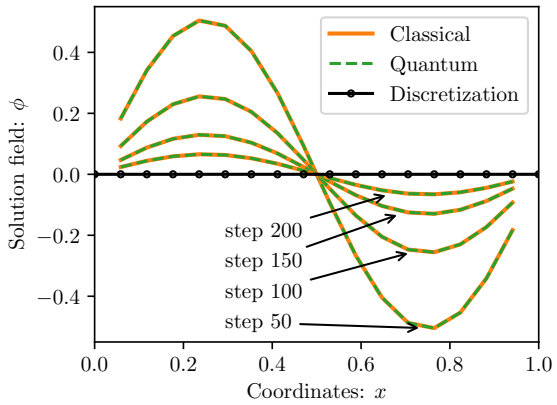

Figure 8: Comparisons of the solution field of 1D Helmholtz equation between quantum and classical implementations after 50, 100, 150, and 200 timesteps.

are plotted. In all instantaneous solution fields drawn in Fig. 8, it is observed that those obtained from the proposed realization in quantum simulator perfectly match the results from the classical computer. This observation indicates that the quantum implementation replicates the time-evolving solution fields from classical computers in a transient diffusion problem.

4.2 Inviscid Burgers' equation

Next, we evaluate the proposed quantum numerical method for the following inviscid 1D Burgers' equation.

$$\partial_t u + u u_{,x} = 0 \quad \text{in } \Omega \times [0,T], \tag{4.4}$$

where the same Dirichlet boundary conditions as Eq. (4.2) are applied to the 1D domain $\Omega = [0,1]$. While Eq. (4.4) takes the standard convective form, an equivalent substitute in conservative form can be written as,

$$\partial_t u + \frac{1}{2}(u^2)_{,x} = 0 \quad \text{in } \Omega \times [0,T]. \tag{4.5}$$

Burgers simplifies the Navier-Stokes equations to this 1D form to investigate the properties of turbulence [27]. The 1D domain is uniformly discretized into 15 cells. The initial condition takes the form of a sinusoidal wave with unit wavelength. As compared to the Helmholtz equation in Section 4.1, not only does the primary differential operator of a Burgers' equation becomes the convection operator, resulting in a hyperbolic system, but the problem also becomes nonlinear since the solution field is the convection velocity itself. We wish to point out that the validation here is particularly about the realization of quantum algorithms presented in Section 3.2.1 for matrix-vector multiplication rather than the usual discussion on the stability and shock-capturing for numerical schemes.

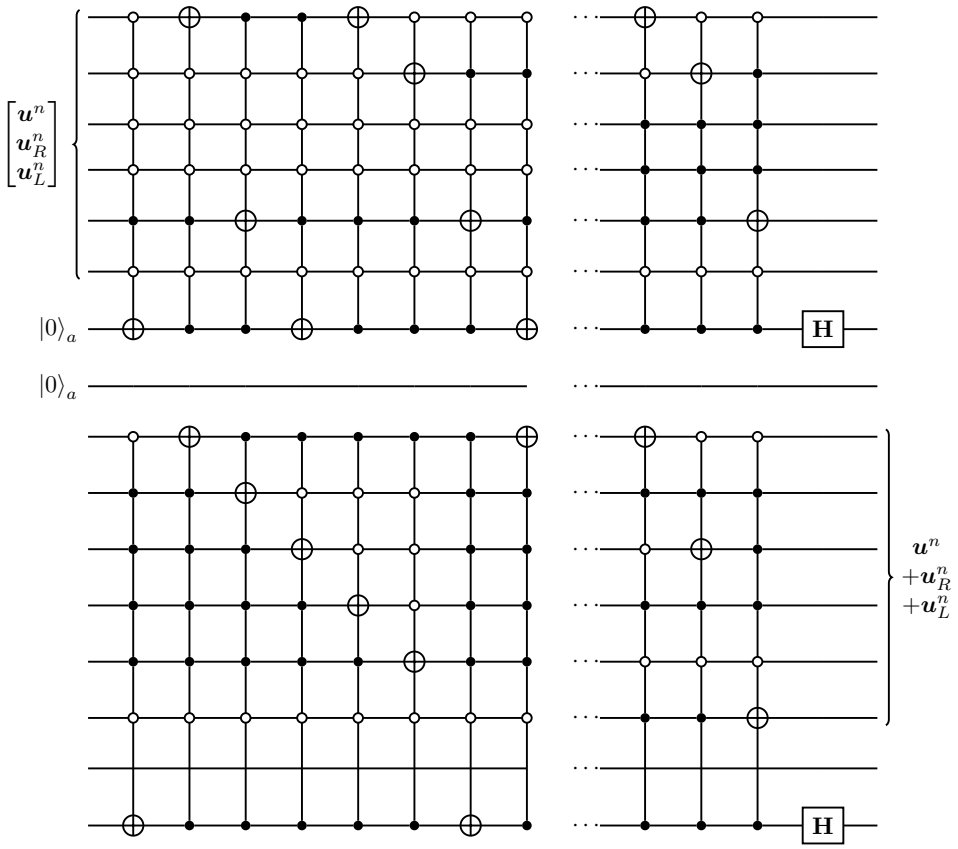

Figure 9: Quantum circuits for the twice vector additions in Eq. (4.6).

As discussed in Section 3.2.1, due to the inhomogeneity in the velocity field and the anti-Hermitian matrix discretized from the convection operator, it is not straightforward to convert the nonlinear convection into a unitary operation directly. Therefore, we encode multiple copies of $\boldsymbol{\Phi}^n$ into the quantum register to calculate different components related to the discretization shown in Eqs. (2.7) and (3.4) and sum them up following the strategies presented in Section 3.2.1. The abstract form of the present algorithm, including one matrix-vector multiplication and two vector addition, is expressed as

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \boldsymbol{u}_R^n + \boldsymbol{u}_L^n \quad \text{and} \quad \begin{bmatrix} \boldsymbol{u}^n \\ \boldsymbol{u}_R^n \\ \boldsymbol{u}_L^n \end{bmatrix} = \tilde{\boldsymbol{A}} \begin{bmatrix} \boldsymbol{u}^n \\ [u_2, u_3, \cdots, u_{m-1}, 0]^T \\ [0, u_1, \cdots, u_{m-3}, u_{m-2}]^T \end{bmatrix}, \tag{4.6}$$

where $\tilde{\boldsymbol{A}}$ is the enriched amplification matrix. Following discrete forms defined in

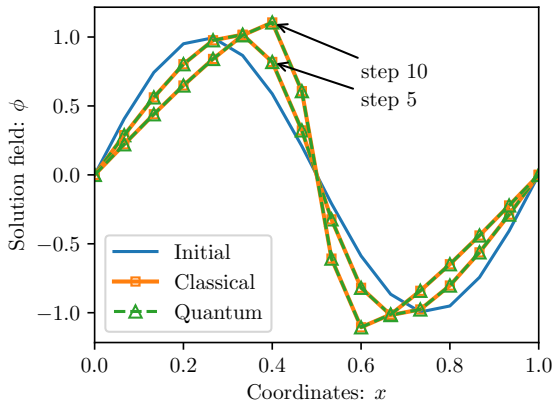

Figure 10: Comparison of the solution field of 1D inviscid Burgers' equation in the advective form between quantum and classical implementations after five and ten steps. The time increment is $\Delta t = 0.2\Delta x$. The initial condition is shown using the solid line.

Eqs. (3.4) and (3.5) for the convection operator, $\tilde{\boldsymbol{A}}$ is adapted to as

$$\tilde{\boldsymbol{A}}_{\mathrm{ADV}} = \begin{bmatrix} \boldsymbol{I} & \\ & \frac{\Delta t}{2\Delta x}\mathrm{diag}(\boldsymbol{u}^n) \\ & & -\frac{\Delta t}{2\Delta x}\mathrm{diag}(\boldsymbol{u}^n) \end{bmatrix}, \tag{4.7}$$

$$\tilde{\boldsymbol{A}}_{\mathrm{ADV}} = \begin{bmatrix} \boldsymbol{I} & \\ & \frac{\Delta t}{4\Delta x}\mathrm{diag}([u_2,u_3,\cdots,u_{m-1},0]) \\ & & -\frac{\Delta t}{4\Delta x}\mathrm{diag}([0,u_1,\cdots,u_{m-3},u_{m-2}]) \end{bmatrix}. \tag{4.8}$$

Since $\tilde{\boldsymbol{A}}$ is a diagonal matrix, it is convenient to implement it in quantum circuits after being converted to a unitary matrix using Eq. (3.3). In this test case, to delineate the implementation of component addition in Section 3.2.3, we illustrate the corresponding quantum circuits in Fig. 9, where two ancilla qubits are utilized for twice vector additions.

The simulations in both the quantum simulator and classical computer are stopped before the shockwave happens due to the lack of shock-capturing terms in the present discretization. The solution fields obtained from quantum and classical computing in the advective form after five and ten time-steps are plotted in Fig. 10, together with the initial condition. As shown in Fig. 10, the proposed quantum algorithm reproduces the evolution history of solution fields as classical computing for the convection-dominant problems. In the meanwhile, we also implement the quantum circuits for the divergence form and present the solution field obtained from the quantum simulator after ten-step time marching in Fig. 11. It is clear that the divergence form is also successfully implemented in the quantum simulator, where the output reproduce the results in a classical computer. Also, the divergence form suppress the overshoot phenomenon as compared to the advective form.

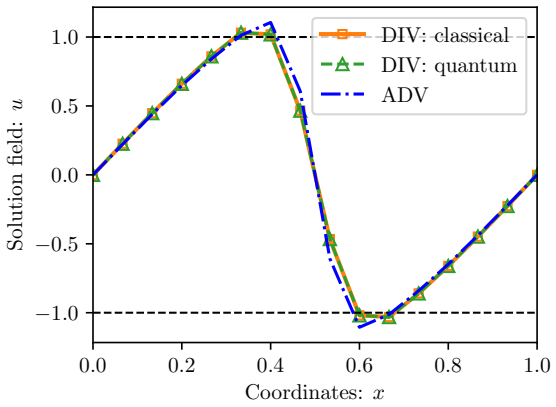

Figure 11: Comparison of the solution field of 1D inviscid Burgers' equation between advective and divergence forms after ten steps. The time increment is identical as Fig. 10.

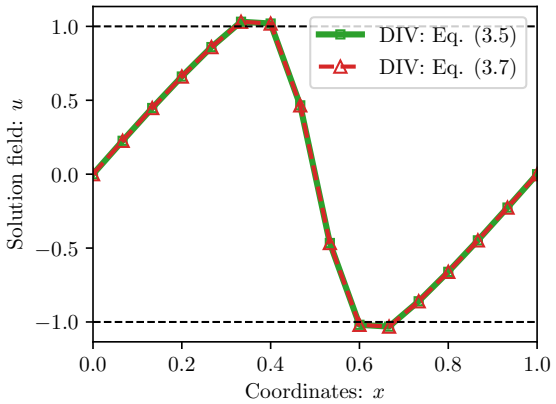

Figure 12: Comparison of the solution field of 1D inviscid Burgers' equation: two quantum realizations of matrix-vector multiplication

The SVD-based quantum matrix arithmetic presented in Section 3.2.1 is also utilized for the inviscid Burgers' equation with the conservative form where the amplification matrix becomes $\boldsymbol{A}=\boldsymbol{I}-(\Delta t/2\Delta x)\boldsymbol{C}_{\mathrm{DIV}}$. Via the unitarization process introduced in Eq. (3.7), we rearrange the quantum circuits of the amplifying stage in the simulator. After ten time-steps of simulation, the obtained solution field is plotted in Fig. 12 alongside the results based on the vector addition in Eq. (3.5). It is clear that both approaches for matrix-vector multiplication lead to identical results. However, the vector-addition-based quantum implementation costs more qubits since one needs to prepare multiple copies of the solution field. The number of copies is determined by order of spatial interpolation p as $2p-1$. The SVD-based implementation is free of such linear dependence in spatial

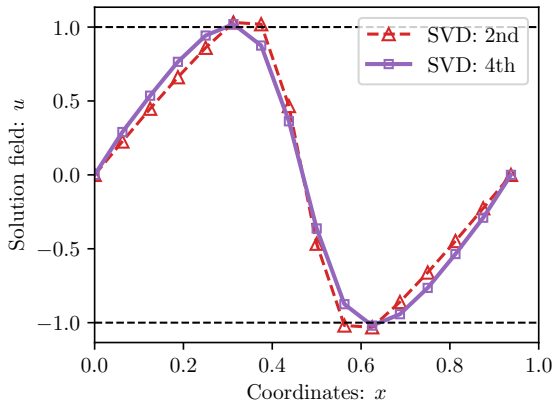

Figure 13: Comparison of the solution field of 1D inviscid Burgers' equation: SVD-based matrix-vector multiplication with quadratic and quartic center difference

complexity. To further illustrate the advantages of the SVD-based implementation, we extend the spatial interpolation from quadratic center differences presented in Eq. (2.12) to quartic center differences. The quantum circuits are rearranged accordingly for the amplification matrix with larger bandwidth, while the number of qubits remains the same. The simulated result with a higher-order stencil is plotted in Fig. 13, where the overshoot phenomenon is further suppressed as compared to the case with a lower-order stencil.

4.3 Navier-Stokes equations

In this test case, a convergent-divergent nozzle (also known as de Laval nozzle) is considered. The variation of the cross-section area of the nozzle is sufficiently slow, yielding a quasi-1D problem. Here we prescribe the area $A=1+2.2(x-1.5)^2$ as the test case in Gaitan [18] for a quantum time integrator. The quasi-1D compressible Navier-Stokes equations are:

$$\partial_t(\rho A)+\partial_x(\rho A u)=0, \tag{4.9}$$

$$\partial_t(\rho A u)+\partial_x\left(\rho A u^2+\frac{pA}{\gamma}\right)=\frac{p}{\gamma}\partial_x A, \tag{4.10}$$

$$\partial_t(\rho A E)+\partial_x(\rho A u E+pAu)=0, \tag{4.11}$$

where $\rho(x,t)$, $u(x,t)$, and $T(x,t)$ are the density, velocity and temperature fields, respectively, which are also referred to as primitive variables; $A(x)$ is the prescribed varying area of the cross section of the nozzle; $\gamma=1.4$ is the heat capacity ratio; $E=e/(\gamma-1)+\gamma u^2/2$ is the total energy; the internal energy $e=RT/(1-\gamma)$. To apply the proposed quantum algorithm, we rewrite the quasi-1D compressible Navier-Stokes equa-

tions in Eq. (4.11) based on conservative variables as follows.

$$\partial_t \boldsymbol{U} + \partial_x \boldsymbol{F}(\boldsymbol{U}) = \boldsymbol{S}(\boldsymbol{U}), \tag{4.12}$$

$$\boldsymbol{U} = \begin{bmatrix} \rho A \\ \rho A u \\ \rho A E \end{bmatrix}, \quad \boldsymbol{F} = \begin{bmatrix} U_2 \\ \frac{U_2^2}{U_1} + \frac{\gamma-1}{\gamma}\left(U_3 - \frac{\gamma U_2^2}{2U_1}\right) \\ \frac{\gamma U_2 U_3}{U_1} - \frac{\gamma(\gamma-1)U_2^3}{2U_1^2} \end{bmatrix}, \quad \boldsymbol{S} = \begin{bmatrix} 0 \\ \frac{\gamma-1}{\gamma}\left(U_3 - \frac{\gamma U_2^2}{2U_1}\right)\frac{\partial_x A}{A} \\ 0 \end{bmatrix}, \tag{4.13}$$

where $\boldsymbol{U}$ is the vector of the conservative variables including mass, momentum, and total energy, $\boldsymbol{F}$ is the flux vector, and $\boldsymbol{S}$ is the source vector.

The 1D computational domain $\Omega = [0,3]$ is uniformly divided into 128 cells. 11 qubits are used to encode the solutions and fluxes of the last time-step together with the source vector. The initial conditions for temperature and density are,

$$\rho_0 = \begin{cases} 1.0, & x \le 1.5, \\ -\frac{1}{3}x + 1.0, & x > 1.5, \end{cases} \quad \text{and} \quad T_0 = \begin{cases} -\frac{1}{3}x + 1.0, & x \le 1.5, \\ 0.5, & x > 1.5, \end{cases} \tag{4.14}$$

and the initial velocity $u_0(x)$ is determined by the governing equations. The time increment is set as $\Delta t = 0.2\Delta x$ to ensure the stability. Both classical and quantum computing are conducted for 2000 time-steps to make the discrete systems evolve into the nearly steady state. We depict the converged quantum solutions and the steady-state exact solutions derived by Anderson [1] in Fig. 14. The converged solution obtained from the quantum simulator is very close to the exact solutions, which validates the capacity of our quantum implementation. Moreover, profiles of primitive variables at three different snapshots from both classical and quantum computing are plotted in Fig. 14. It is observed that the results obtained from classical and quantum computing match well with each other, which indicates the consistency of the presenting quantum implementations.

5 Conclusions

In this paper, we have presented a numerical procedure to conduct simulations of fluid dynamics problems using a quantum computer. A sample convection-diffusion equation, as a simplified version of the Navier-Stokes equations, is discretized using the classical FTCS scheme. We propose a multiple-stage numerical procedure for the realization of a single-step time-marching scheme in DQC, including encoding, amplification, applying source terms, and incorporating boundary conditions. To accommodate the postulates in a quantum-mechanical system, the solution fields are encoded via of the quantum state vector and the operations on the solution fields need to be recast into unitary operations. In particular, three linear algebra procedures, including matrix-vector multiplication, vector addition, and component addition, are converted into unitary transformations and therefore become admissible in a quantum computer.

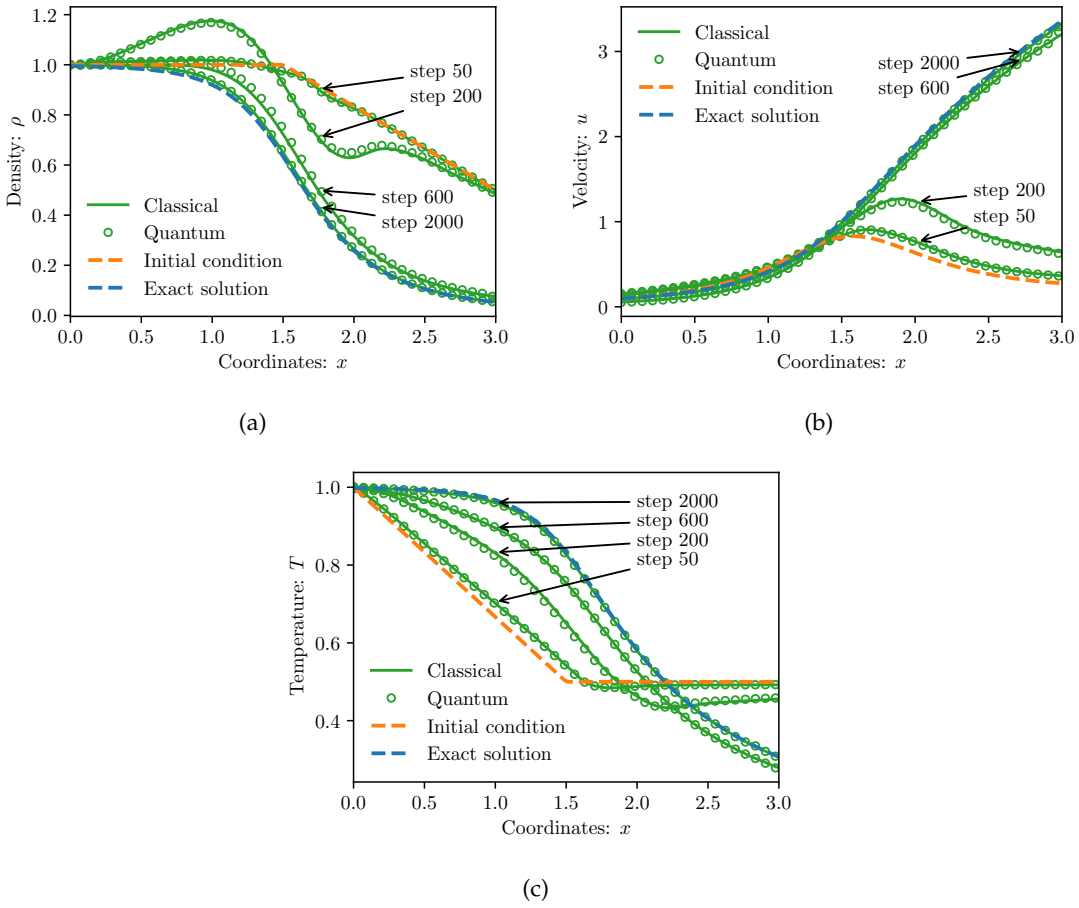

Figure 14: Comparisons of the solution fields in a de Laval nozzle obtained from quantum and classical computing at various snapshots: (a) density, (b) velocity, and (c) temperature.

Though the dimension of the vector space spanned by qubits increases exponentially, the space complexity of the proposed quantum numerical procedure is not necessarily superior to classical algorithms. Due to the no-cloning theorem, it is impossible to efficiently (i.e., less than the logarithmic scale) reproduce a state vector with identical probability magnitudes to be recycled during time marching in nonlinear problems, where the amplification matrix $\boldsymbol{A}$ depends on the solution field from last time-step $\boldsymbol{u}^n$. As pointed out in [28], one needs to generate enough number of copies to ensure there is at least one valid copy of $\boldsymbol{u}^n$ stored in the probability magnitudes of the state vector at $T=N\Delta t$, where N is the total number of time-steps. The required number of copies in the present method is in the scale of $\mathcal{O}(2^n)$. The total space complexity becomes $\mathcal{O}(2^N \log m)$. For any practical fluid mechanics problem, N will be large enough to eliminate the direct space complexity from encoding in the state vector spanned by qubits.

To validate the proposed quantum numerical method, we implement it in a quantum computing simulator for a series of problems possessing an increasing degree of complexity. The first test case is a 1D Helmholtz equation with homogeneous boundary conditions. In this problem, we delineate the quantum circuits to encode the discrete initial condition into quantum amplitudes and validate our implementation for diffusion equations. The second test case is a transient problem of 1D inviscid Burgers' equation to further validate the proposed quantum implementation for nonlinear convection equations. Finally, we extend the application of the proposed quantum numerical procedure to a nonlinear coupled system of hyperbolic equations–quasi-1D compressible Navier-Stokes equations. A hypersonic de Laval nozzle with analytical solutions is used as the benchmark problem. In all three test cases, the results attained from quantum computing are identical to those attained from classical computing. These tests comprehensively validate the realization of the proposed method in ideal quantum circuits.

Quantum solvers for hyperbolic PDEs are relatively rare among all the preliminary investigations on the quantum computing for Navier-Stokes equations or their simplified derivatives, as summarized in Childs et al. [12]. This work addresses the quantum implementations for 1D nonlinear hyperbolic PDEs via classical central differences and explicit time marching for the brevity. It is worth noticing that the spatial discretization and time marching schemes can be straightforwardly replaced by more sophisticated options to largely improve the accuracy and stability of the numerical procedures. Moreover, the proposed quantum implementation for the 1D problem can be further extended to multidimensional problems. The proposed quantum numerical procedure is validated in a simulator for perfect quantum circuits, while the number of qubits in the cutting edge hardware is reaching the concept of the so-called Noisy Intermediate-Scale Quantum (NISQ) computers [38]. To make quantum computational fluid dynamics feasible in academic and industrial practices, it is critical to propose algorithms with superiority in time complexity and with inherent error-resistant capabilities in the future.

Acknowledgments

This work is supported by NSFC Basic Science Center Program for "Multiscale Problems in Nonlinear Mechanics" (Grant No. 11988102) and National Natural Science Foundation of China (Grant No. 12202454).

References

[1] J. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill Book Company, 6 edition, Mar. 2016.

[2] M. S. Anis, H. Abraham, AduOffei, R. Agarwal, G. Agliardi, M. Aharoni, I. Y. Akhalwaya, G. Aleksandrowicz, T. Alexander, M. Amy, S. Anagolum, E. Arbel, A. Asfaw, A. Athalye, A. Avkhadiev, C. Azaustre, P. BHOLE, A. Banerjee, S. Banerjee, ⋯, and M. ÄŒepulkovskis. Qiskit: An Open-source Framework for Quantum Computing, 2021.

[3] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, ⋯, and J. M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, 2019.

[4] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter. Elementary gates for quantum computation. *Phys. Rev. A*, 52(5):3457–3467, 1995.

[5] J. R. Basani and A. Bhattacherjee. Continuous-variable deep quantum neural networks for flexible learning of structured classical information. *Commun. Comput. Phys.*, 30(4):1216–1231, 2021.

[6] G. Benenti, G. Casati, D. Rossini, and G. Strini. *Principles of Quantum Computation and Information*. World Scientific, 2018.

[7] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang. Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Commun. Math. Phys.*, 356(3):1057–1081, 2017.

[8] S. Boixo, T. F. Rønnow, S. V. Isakov, Z. Wang, D. Wecker, D. A. Lidar, J. M. Martinis, and M. Troyer. Evidence for quantum annealing with more than one hundred qubits. *Nature Phys.*, 10(3):218–224, 2014.

[9] J. Cai, W. G. Macready, and A. Roy. A practical heuristic for finding graph minors. *arXiv:1406.2741 [quant-ph]*, 2014.

[10] Y. Cao, A. Daskin, S. Frankel, and S. Kais. Quantum circuit design for solving linear systems of equations. *Molecular Physics*, 110(15-16):1675–1680, 2012.

[11] Y. Cao, A. Papageorgiou, I. Petras, J. Traub, and S. Kais. Quantum algorithm and circuit design solving the Poisson equation. *New J. Phys.*, 15(1):013021, 2013.

[12] A. M. Childs, J.-P. Liu, and A. Ostrander. High-precision quantum algorithms for partial differential equations. *Quantum*, 5:574, 2021.

[13] A. M. Childs and W. van Dam. Quantum algorithms for algebraic problems. *Rev. Mod. Phys.*, 82(1):1–52, 2010.

[14] C. M. Dawson and M. A. Nielsen. The Solovay-Kitaev algorithm. *Quantum Info. Comput.*, 6(1):81–95, 2006.

[15] D. E. Deutsch, A. Barenco, and A. Ekert. Universality in quantum computation. *Proc. R. Soc. Lond. A*, 449(1937):669–677, 1995.

[16] C. R. Doering and J. D. Gibbon. *Applied Analysis of the Navier-Stokes Equations*. Cambridge University Press, 1995.

[17] F. Fillion-Gourdeau and E. Lorin. Simple digital quantum algorithm for symmetric first-order linear hyperbolic systems. *Numer. Algor.*, 82(3):1009–1045, 2019.

[18] F. Gaitan. Finding flows of a Navier-Stokes fluid through quantum computing. *npj Quantum Information*, 6(1):1–6, 2020.

[19] I. Georgescu, S. Ashhab, and F. Nori. Quantum simulation. *Rev. Mod. Phys.*, 86(1):153–185, 2014.

[20] P. M. Gresho and R. L. Sani. *Incompressible Flow and the Finite Element Method. Volume 1: Advection-Diffusion and Isothermal Laminar Flow*, volume 1. John Wiley & Sons, Ltd., New York, NY (United States), 1998.

[21] L. Hales and S. Hallgren. An improved quantum Fourier transform algorithm and applications. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 515–525, 2000. ISSN: 0272-5428.

[22] A. W. Harrow, A. Hassidim, and S. Lloyd. Quantum algorithm for linear systems of equa-

tions. *Phys. Rev. Lett.*, 103(15):150502, 2009.
[23] T.-J. Hsu, F. Jin, C. Seidel, F. Neukart, H. De Raedt, and K. Michielsen. Quantum annealing with anneal path control: Application to 2-SAT problems with known energy landscapes. *Commun. Comput. Phys.*, 26(3):928–946, 2019.
[24] J. Hunt. Lewis Fry Richardson and his contributions to mathematics, meteorology, and models of conflict. *Annu. Rev. Fluid Mech.*, 30(1):xiii–xxxvi, 1998.
[25] V. M. Kendon, K. Nemoto, and W. J. Munro. Quantum analogue computing. *Philos. Trans. Royal Soc. A*, 368(1924):3609–3620, 2010.
[26] M. Knudsen and C. B. Mendl. Solving differential equations via continuous-variable quantum computers. *arXiv:2012.12220 [quant-ph]*, 2020.
[27] R. H. Kraichnan. Lagrangian-history statistical theory for Burgers' equation. *Phys. Fluids*, 11(2):265, 1968.
[28] S. K. Leyton and T. J. Osborne. A quantum algorithm to solve nonlinear differential equations. *arXiv:0812.4423 [quant-ph]*, 2008.
[29] J.-P. Liu, H. Ã. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, and A. M. Childs. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proc. Natl. Acad. Sci. USA*, 118(35):e2026805118, 2021.
[30] B. Ljubomir. Quantum algorithm for the Navier-Stokes equations by using the stream function-vorticity formulation and the lattice Boltzmann method. *International Journal of Quantum Information*, 2022.
[31] S. Lloyd, G. De Palma, C. Gokler, B. Kiani, Z.-W. Liu, M. Marvian, F. Tennie, and T. Palmer. Quantum algorithm for nonlinear differential equations. *arXiv:2011.06571 [nlin, physics:quant-ph]*, 2020.
[32] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch. Variational quantum algorithms for nonlinear problems. *Phys. Rev. A*, 101(1):010301, Jan. 2020.
[33] E. R. MacQuarrie, C. Simon, S. Simmons, and E. Maine. The emerging commercial landscape of quantum computing. *Nat Rev Phys*, 2(11):596–598, 2020.
[34] A. Montanaro and S. Pallister. Quantum algorithms and the finite element method. *Phys. Rev. A*, 93(3):032324, 2016.
[35] Y. Morinishi, T. Lund, O. Vasilyev, and P. Moin. Fully conservative higher order finite difference schemes for incompressible flow. *J. Comput. Phys.*, 143(1):90–124, 1998.
[36] M. A. Nielsen and I. L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.
[37] J. Preskill. Quantum computing and the entanglement frontier. *arXiv:1203.5813 [cond-mat, physics:quant-ph]*, 2012.
[38] J. Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.
[39] N. Ray, T. Banerjee, B. Nadiga, and S. Karra. Towards solving the Navier-Stokes equation on quantum computers. *arXiv:1904.09033 [physics]*, 2019.
[40] V. Shende, S. Bullock, and I. Markov. Synthesis of quantum-logic circuits. *IEEE Trans. Comput-Aided Des. Integr. Circuits Syst.*, 25(6):1000–1010, 2006.
[41] J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, and D. J. Mavriplis. CFD vision 2030 study: A path to revolutionary computational aerosciences. Technical Report NASA/CR–2014-218178, NASA, 2014.
[42] S. Srivastava and V. Sundararaghavan. Box algorithm for the solution of differential equations on a quantum annealer. *Phys. Rev. A*, 99(5):052355, 2019.
[43] R. Steijl. Quantum Algorithms for Fluid Simulations. In F. Bulnes, V. N. Stavrou, O. Morozov, and A. V. Bourdine, editors, *Advances in Quantum Communication and Information*. Inte-

chOpen, 2020.
[44] R. Steijl and G. N. Barakos. Parallel evaluation of quantum algorithms for computational fluid dynamics. *Comput. Fluids*, 173:22–28, 2018.
[45] Y. Subaşı, R. D. Somma, and D. Orsucci. Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Physical Review Letters*, 122(6):060504, 2019.
[46] J. Wen, X. Kong, S. Wei, B. Wang, T. Xin, and G. Long. Experimental realization of quantum algorithms for a linear system inspired by adiabatic quantum computing. *Phys. Rev. A*, 99(1):012320, 2019.
[47] T. Xin, S. Wei, J. Cui, J. Xiao, I. Arrazola, L. Lamata, X. Kong, D. Lu, E. Solano, and G. Long. Quantum algorithm for solving linear differential equations: Theory and experiment. *Phys. Rev. A*, 101(3):032307, 2020.
[48] X. I. A. Yang and K. P. Griffin. Grid-point and time-step requirements for direct numerical simulation and large-eddy simulation. *Phys. Fluids*, 33(1):015108, 2021.
[49] P. K. Yeung, K. R. Sreenivasan, and S. B. Pope. Effects of finite spatial and temporal resolution in direct numerical simulations of incompressible isotropic turbulence. *Phys. Rev. Fluids*, 3(6):064603, 2018.
[50] B. Zanger, C. B. Mendl, M. Schulz, and M. Schreiber. Quantum algorithms for solving ordinary differential equations via classical integration methods. *Quantum*, 5:502, 2021.