2024 IEEE International Conference on Quantum Computing and Engineering (QCE)

Towards a Quantum Algorithm for the Incompressible Nonlinear Navier–Stokes Equations

Muralikrishnan Gopalakrishnan Meena*, Yu Zhang†, Weiwen Jiang‡,
Youzuo Lin§, Stefanie Günther¶, and Xinfeng Gao‖
*National Center for Computational Sciences, Oak Ridge National Laboratory, Oak Ridge, TN, USA
†Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM, USA
‡Department of Electrical and Computer Engineering, George Mason University, Fairfax, VA, USA
§School of Data Science and Society, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA
¶Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA, USA
‖Department of Mechanical and Aerospace Engineering, University of Virginia, Charlottesville, VA, USA
Corresponding authors: ‖x.gao@virginia.edu, *gopalakrishm@ornl.gov, †zhy@lanl.gov
* † These authors contributed equally

2024 IEEE International Conference on Quantum Computing and Engineering (QCE) | 979-8-3315-4137-8/24/$31.00 USG | DOI: 10.1109/QCE60285.2024.00083

***Abstract*—In this work, we present novel concepts for quantum algorithms to solve transient, nonlinear partial differential equations (PDEs). The challenge lies in how to effectively represent, encode, process, and evolve the nonlinear system of PDEs on quantum computers. We will discuss the new techniques using the incompressible Navier–Stokes equations as an example, because it represents the fundamental nonlinear feature and yet removes certain complexity in physics, allowing us to focus on the design of quantum algorithms. Previous attempts solving nonlinear PDEs in quantum computation have often involved storing multiple copies of solutions or employing linearizations. Neither is practical due to exponential scaling with evolution time or insufficient solution accuracy. We propose a new framework based on matrix product states (MPSs) and matrix product operators (MPOs), in addition to the Krylov subspace methods. For example, the solution variables of the Navier–Stokes equations are represented by MPSs, and the linear and nonlinear terms are processed by MPOs. The time evolution of the operators is attained by a fast-forwarding algorithm using Krylov subspace methods. Furthermore, we discuss various techniques for efficient encoding of MPSs, measurement reduction for MPOs, and use of tensor operations to treat multi-variate, multi-physics characteristics of Navier–Stokes.**

***Index Terms*—Nonlinear partial differential equations, Incompressible fluid dynamics, Tensor networks, Matrix product states, Matrix product operators, Krylov subspace methods**

I. Introduction

Partial differential equations (PDEs) are ubiquitous in science, governing the time evolution of systems in broad range of domains such as climate, transportation, biology, national security, and astrophysics to name a few. The main complexity of numerically solving these PDEs is when nonlinear terms are present. Analytically, for the full 3D Navier–Stokes equations, the existence of smooth and globally defined solutions is not known. Numerical methods can provide insight into how solutions might behave and offer evidence regarding the existence of singularities or the stability of solutions. Although significant progress has been made in solving 3D compressible Navier–Stokes equations with complex physics, such as turbulence by classical high-performance computing (HPC), accurate modeling and simulation of practical turbulent flows are still challenging and can be computationally prohibitive for high-Rynolds-number flows, even with exascale supercomputers [1]. For example, the total amount of computational resources used by turbulence-related projects on the Oak Ridge Leadership Computing Facility's Frontier exascale supercomputer in 2023 amounts to around 35–45% of the available resources.

With the advent of quantum computing, we are on the cusp of redesigning computational capabilities. By harnessing the power of quantum computers and algorithms, we aim to improve solution techniques to turbulence that are hard and intractable for conventional HPC. There have been many attempts in the past to solve linear PDEs using quantum computing, mainly using quantum linear solver algorithms (QLSA) [2], [3]. Linear versions of the Navier–Stokes equations have profound applications for modeling efforts, and quantum algorithms to solve many such problems have been demonstrated in the past [4]–[7], including hybrid approaches [8]. However, quantum algorithms that efficiently solve nonlinear PDEs are limited [9]. Most of the efforts for efficiently solving nonlinear versions of Navier–Stokes equations (and other nonlinear PDEs) have been focused on linearizing the nonlinear equations and using QLSA to solve the resulting set of ordinary differential equations (ODEs). For example, Carleman linearization can be used to solve an N-dimensional quadratic nonlinear ODE [10], [11]. Another approach is the use of Koopman von Neumann representation for the time evolution of nonlinear dynamical system [12]–[14].

Some efforts have demonstrated the capability to address nonlinear PDEs [9], [15]. Some works involve the use of quantum amplitude estimation algorithm (QAEA) to solve specific nonlinear PDEs [16], [17]. Another promising method uses variational quantum computing and tensor networks to calculate nonlinear terms for single-variable PDEs [18]. While the quantum nonlinear processing unit (QNPU) is a novel formulation to treat nonlinearities in PDEs, the system of PDEs (e.g., Navier–Stokes equations) adds new challenges.

DOI 10.1109/QCE60285.2024.00083

Alternate to the variational optimization procedure, Krylov subspace methods are promising to generate tensor operation circuits. Furthermore, the complex tensor operations can lead to large terms resulting from the operators, causing challenges in the measurement stages.

In this *New Ideas and Emergent Results* paper, we describe a quantum algorithm framework using tensor networks to solve the system of PDEs, specifically, the Navier–Stokes equations. For clarity, we describe the framework using the incompressible Navier–Stokes equations, providing a conceptual description of how the nonlinear terms in the PDEs can be effectively processed using tensor network-based algorithms. We also describe techniques to efficiently encode the flow states into tensor networks, address the multi-variate coupling in PDEs, and reduce measurement for tensor operations.

II. Incompressible Navier–Stokes equations

The dimensionless incompressible Navier–Stokes equations for continuity and momentum are given by

$$\nabla \cdot \boldsymbol{u} = 0, \tag{1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} = - \underbrace{\boldsymbol{u} \cdot \nabla \boldsymbol{u}}_{\text{nonlinear term}} \underbrace{- \nabla p + \frac{1}{Re} \nabla^2 \boldsymbol{u}}_{\text{linear terms}} + \boldsymbol{s}. \tag{2}$$

Here, $\boldsymbol{u} = (u, v, w)$ represents the velocity vector in the Cartesian coordinate system, $\boldsymbol{x} = (x, y, z)$, evolving in time, t. The pressure of the flow field is represented by p, Re is the Reynolds number[1] of the flow, and $\boldsymbol{s}$ represents any source terms or external forcing to the flow. For simplicity, $\boldsymbol{s}$ is neglected in our demonstration.

The pressure is solved using the Poisson equation obtained by taking the divergence of Eq. 2 and applying Eq. 1, giving

$$\nabla^2 p = - \underbrace{\nabla \cdot (\boldsymbol{u} \cdot \nabla \boldsymbol{u})}_{\text{nonlinear term}} \tag{3}$$

Classically, velocity and pressure are solved iteratively at each time step of the solver – for example, starting with an initial velocity field, pressure is solved using Eq. 3 and the solution is used to solve for velocity using Eq. 2. In the quantum algorithm, this coupling between velocity and pressure needs to be addressed carefully. In Eq. 2, $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$, is the nonlinear term and its accurate computation is critical, especially for highly turbulent flows. This is because understanding turbulence resides in comprehending the interscale correlations that influence turbulence structure, instability dynamics, and energy transfer across scales. Fig. 1 illustrates the interscale dynamics in a Taylor–Green vortex problem. Computation of the interscale correlations becomes more challenging for high-Reynolds-number flows where disparate scales lead to stronger nonlinearity. Next, we describe the quantum algorithms for evaluating the linear and nonlinear terms, highlighted in the momentum (Eq. 2) and pressure Poisson (Eq. 3) equations and for evolving the system of PDEs.

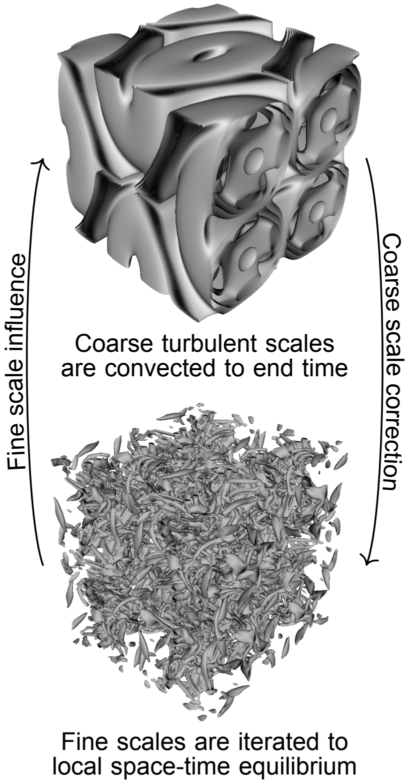
Fine scale influence
Coarse scale correction
Coarse turbulent scales
are convected to end time
Fine scales are iterated to
local space-time equilibrium

Fig. 1. Iso-surfaces of enstrophy in a Taylor-Green vortex problem showing the scale correlation [19].

III. Quantum algorithm

The main idea behind the algorithm is to represent the classical solution variables (e.g., the velocity vector) in quantum states by transferring Eq. 2 to a Hamiltonian evolution problem. There are three steps in this algorithm:

1) Quantum embedding of the flow variables - use matrix product state (MPS)
2) Quantum representation of the linear and nonlinear differential operations - use matrix product operation (MPO) and quantum Krylov subspace methods
3) Time evolution of the quantum operations - use variational fast-forwarding algorithms

Fig. 2 illustrates the steps from left to the right. That is prepare the initial state and the initial Hamiltonian, encode the variable states in MPSs, advance the states by MPOs, measure and analyze. In the following sections, we discuss the technical details.

A. Quantum embedding of flow variables using MPS

One of the most challenging aspects of translating quantum algorithms to solve classical problems in various domain applications is embedding the state information into quantum states. Inspired by tensor networks, we embed the flow states as matrix product states (MPSs) [20], [21].

The key idea is to decompose each component of the $\mathcal{D}$-dimensional velocity variable $\boldsymbol{u} = \sum_{\alpha}^{\mathcal{D}} u_\alpha(\boldsymbol{x}) \boldsymbol{e}_\alpha$ into a series

[1] A non-dimensional number representing the intensity of the flow to help identify the transition between laminar and turbulent flow behavior.

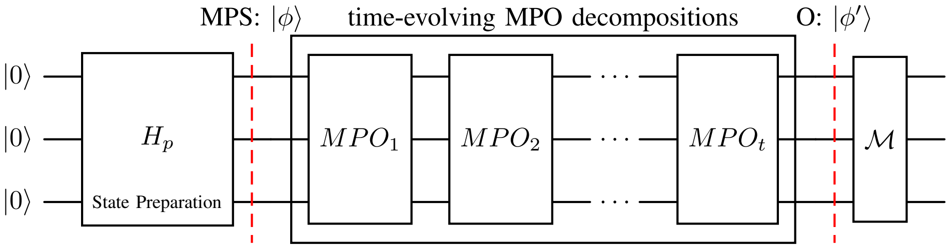

Fig. 2. Sample depiction of the tensor network-based quantum circuit representing the time evolution of the nonlinear equations. The circuit depths increase linearly, $\mathcal{O}(T)$, with time steps (T) in the conventional Trotterization procedure. In contrast, the fast-forwarding algorithm reduces the circuit depth to $\mathcal{O}(1)$ by approximating the time-evolving operator with a diagonal factorization.

of low-rank tensors. Suppose the PDE is discretized on a grid with $N = K^q$ grid points in each dimension (total number of grid points is $K^{\mathcal{D}q}$), where N is the classical registers and q the number of quantum registers. Thus, the velocity variable is approximated by $u_\alpha(\boldsymbol{x}) \simeq u_\alpha(x_1, \cdots, x_{\mathcal{D}q}) \equiv u_\alpha(\{x_i\})$. Here, $i = 1, 2, \ldots, \mathcal{D}q$ and x_i denote the position of the element in the mapped quantum state space, and its value ranges from 0 to $K-1$. Then we can decompose the function $u_\alpha(\{x_i\})$ into matrix product state (MPS) via the Schmidt decomposition,

$$u_\alpha(\boldsymbol{x}) = \sum_{j_1,\cdots,j_{N-1}}^{d} A^1_{j_1} A^2_{j_1,j_2} \cdots A^{\mathcal{D}q}_{j_{\mathcal{D}q-1}}, \tag{4}$$

where $\mathcal{D}$ is the number of dimensions, d is the bond dimension (or rank) that controls the quality of the tensor decomposition, and A^k is a third-order tensor. The same procedure can be used to embed the pressure, $p(\boldsymbol{x})$ (or any other variable). Herein, we use $u_p(\{x_i\})$ to denote the MPS embedding of pressure, considering it to be an additional decomposition. This leads to a total of $\hat{\mathcal{D}} = \mathcal{D} + 1$ variable decomposition.

The bond dimension is the dimension of the tensor that is used to perform tensor contraction with another tensor, and its value can be different for different tensors. For a full-rank decomposition, the value of d increases exponentially from the left to the middle of the tensor chain ($\min(j, \mathcal{D}q - j)$) and then decreases exponentially to the right-most tensor. Thus, the bond dimension is exponentially large for a full-rank representation of the original function. However, the number of non-zeros singular values in the decomposition is usually small compared to the full rank. Thus, we can impose a low-rank approximation of the original state by only retaining the d largest singular values in the decomposition at each bond. With the low-rank approximation, the original function of $K^{\mathcal{D}q}$ data points is replaced by a series of tensors with $\sim \mathcal{D}q(Kd^2)$ elements in total, leading to a significant reduction of data points. Furthermore, our recent work on "PermVQE" [22] algorithm demonstrated that quantum-information theory can be utilized to analyze the ranks of the tensors and optimize their ordering to reduce the bond dimension. We elaborate on some additional discussion for this efficient encoding in the Appendix section (A). Using the MPS framework, we can adaptively manage the computational resources needed to approximate solutions to the Navier–Stokes equations effectively.

Insight 1: Quantum-information-inspired efficient encoding of flow variables.

B. Quantum mapping of linear and nonlinear differential operators

Once the flow variables are embedded in the quantum state, we now map the classical differential operations (linear and nonlinear) to quantum operators. Below, we first elaborate on how the linear terms – the pressure gradient, ∇p, and the velocity diffusion, $\nabla^2 \boldsymbol{u}$, terms – are individually represented by matrix product operators (MPOs). Then, we show how the nonlinear convective term, $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$, can be mapped using the quantum Krylov subspace method.

Mapping the linear differential operators into MPOs: Once the MPS state is constructed, differential operators (first and second orders) can be easily mapped into corresponding matrix product operators (MPOs) [23]. Within the finite difference scheme, the differentiation (any order n) of a function $u_\alpha(x_i)$ at position x_i is turned into a linear combination of itself and its displacement $u_\alpha(x_i \pm n\delta x)$. Consequently, within the MPS representation, the differentiation operator is connected to the MPO that induces the state transition $u_\alpha(x) \to u_\alpha(x \pm n\delta x)$.

Let us consider the first and second order spatial deviates in central difference (with $\mathcal{O}(\delta x^2)$ error)

$$\frac{\partial u_\alpha(x_i)}{\partial x} \simeq \frac{u_\alpha(x_i + \delta x) - u_\alpha(x_i - \delta x)}{2\delta x} \tag{5}$$

$$\frac{\partial^2 u_\alpha(x_i)}{\partial x^2} \simeq \frac{u_\alpha(x_i + \delta x) + u_\alpha(x_i - \delta x) - 2u_\alpha(x_i)}{\delta x^2}. \tag{6}$$

The classical cost of applying these on the grid points is $\mathcal{O}(K^{\mathcal{D}q})$. Using the MPS embedding of the variables, we can use MPOs to implement the displacements in the grid as

$$\frac{\partial \left|u_\alpha(\{x_i\})\right\rangle}{\partial x} \simeq \frac{S^+ - S^-}{2\delta x} \left|u_\alpha(\{x_i\})\right\rangle \tag{7}$$

$$\frac{\partial^2 \left|u_\alpha(\{x_i\})\right\rangle}{\partial x^2} \simeq \frac{S^+ + S^- - 2}{\delta x^2} \left|u_\alpha(\{x_i\})\right\rangle. \tag{8}$$

Here, $S^{\pm}$ are ladder operators that enable the operation of state transitions in the quantum circuit

$$S^{+}\left|u_{\alpha}(\{x_i\})\right\rangle = \begin{cases} \left|u_{\alpha}(\{x_{i+1}\})\right\rangle, & u_{\alpha}(\{x_i\}) < K^{\mathcal{D}q} \\ 0, & \text{else} \end{cases} \tag{9}$$

$$S^{-} = \left(S^{+}\right)^{\dagger}. \tag{10}$$

The MPO implementations can be efficiently performed using a bond dimension of 3. Thus, the MPO implementation gives a computational cost reduction to $\mathcal{O}(\mathcal{D}d^3)$, leading to exponentially faster operations for representations with small bond dimension [23].

We note that for computing the finite difference operations in other dimensions, the ladder operator can also be used to perform displacements in the grid. The displacement step in Eq. 9, instead of being unity, will now be dependent on q and the particular dimension $\mathcal{D}$ where the finite difference operation is performed. Furthermore, the operations can be easily extended to incorporate higher order approximations.

Given the MPOs for the two linear terms and MPS being represented as the tensor product of individual components $\left|u_{\alpha}(\{x_i\}), u_p(\{x_i\})\right\rangle = \left|u_{\alpha}(\{x_i\})\right\rangle \otimes \left|u_p(\{x_i\})\right\rangle$, we can sum the results of the two MPOs to give the full linear MPO for the momentum equation as

$$\begin{aligned}\hat{L}\left|u_{\alpha}, u_p\right\rangle = -I_{\alpha} \otimes \frac{S_p^{+} - S_p^{-}}{2\delta x}\left|u_{\alpha}(\{x_i\}), u_p(\{x_i\})\right\rangle + \\ \frac{1}{Re}\frac{S_{\alpha}^{+} + S_{\alpha}^{-} - 2}{\delta x^2} \otimes I_p\left|u_{\alpha}(\{x_i\}), u_p(\{x_i\})\right\rangle.\end{aligned} \tag{11}$$

Where $S_p^{\pm}$ and $S_{\alpha}^{\pm}$ denote the ladder operators for pressure and velocity, respectively. And the number of qubits required to simulate the coupled multiphysics PDEs is doubled.

> **Insight 2:** Treatment of multi-variate, multi-physics flow characteristics using tensor products.

Mapping the nonlinear differential operator into MPOs. The critical challenge in solving the Navier–Stokes equations on quantum computers is the treatment of the nonlinear term $\boldsymbol{u} \cdot \nabla \boldsymbol{u}$ due to the linear nature of quantum state evolution. For a given circuit encoding the MPS u_{α}, the elementwise multiplication $u_{\alpha}\nabla u_{\alpha}$ can be represented by $u_{\alpha}(x_i)[O^{\nabla}u_{\alpha}](x_i)$, where O^{∇} represents the MPO for the gradient operator. Hence, following the quantum nonlinear processing unit (QNPU) concept proposed in Ref. [18], we can make a copy of the u_{α} circuit, apply the unitaries for O^{∇} operators, and then add a set of CNOT gates to obtain the element-wise multiplication between $u_{\alpha}(x_i)$ and $[O^{\nabla}u_{\alpha}](x_i)$. However, unlike the variational algorithm proposed in Ref. [18], our approach involves the use of a linear combination of unitaries (LCU) to solve u_{α} efficiently.

The evolution of the momentum equation Eq. 2 with a time step of $\delta\tau$ can be represented by a functional form $\boldsymbol{u}(t + \delta\tau) = e^{-\mathcal{L}\delta\tau}\boldsymbol{u}(t)$, where $e^{-\mathcal{L}\delta\tau}$ is a nonlinear and non-unitary evolution operator. Here, we map the scaled $\boldsymbol{u}'(t + \delta\tau) = c^{-1/2}\boldsymbol{u}(t + \delta\tau)$ into unitaries,

$$\boldsymbol{u}'(t + \delta\tau) = c^{-1/2}e^{-\mathcal{L}\delta\tau}\boldsymbol{u}(t) = e^{-i\delta\tau\hat{A}}\boldsymbol{u}(t),$$

where $\hat{A} = \sum c_i\hat{P}_i$ is Hermitian and expanded as sum of Pauli strings. Hence, the change in $\boldsymbol{u}'(t)$ after one step is

$$\Delta_0 = \frac{\boldsymbol{u}'(t + \delta\tau) - \boldsymbol{u}(t)}{\delta\tau} = -\boldsymbol{u} \cdot \nabla\boldsymbol{u} - \nabla p + \frac{1}{Re}\nabla^2\boldsymbol{u}.$$

At the same time, $\boldsymbol{u}'(t + \delta\tau)$ can be approximated from $\boldsymbol{u}(t)$ from a unitary operator $e^{-i\delta\tau\hat{A}} \simeq (1 - i\delta\tau\hat{A})$, leading to $\Delta = \frac{e^{-i\delta\tau\hat{A}}\boldsymbol{u}(t) - \boldsymbol{u}(t)}{\delta\tau} = -i\hat{A}\boldsymbol{u}(t)$ difference. Consequently, the coefficients $\boldsymbol{c} \equiv \{c_i\}$ can be obtained by minimizing the cost function $\mathcal{C} \equiv ||\Delta_0 - \Delta||$, which is given by

$$\mathcal{C} = \left\langle\Delta_0\right|\left.\Delta_0\right\rangle + \sum_i b_ic_i + \sum_{ij} c_ic_jS_{ij}. \tag{12}$$

Where $b_i = i\left\langle\boldsymbol{u}\right|\hat{P}_i^{\dagger}\left|\Delta_0\right\rangle + h.c.$ and $S_{ij} = \left\langle\boldsymbol{u}\right|\hat{P}_i^{\dagger}\hat{P}_j\left|\boldsymbol{u}\right\rangle$. Hence, the minimum of $\mathcal{C}$ is determined by the solution $(S^T + S)\mathbf{c} = \boldsymbol{b}$. Thus, abstaining the solution to $\boldsymbol{c}$, the evolution driven by the nonlinear term is readily simulated by a unitary evolution via

$$e^{-i\hat{A}\delta\tau}\left|u_{\alpha}\right\rangle = e^{-i(\sum_i c_i\hat{P}_i)\delta\tau}\left|u_{\alpha}\right\rangle, \tag{13}$$

which avoids the Carleman linearization [10] of nonlinear equations.

> **Insight 3:** Linear combination of unitaries can be applied to create the Quantum Nonlinear Processing Unit (QNPU) circuits, which serve as the backbone for a wide range of real-world problems, including complex computational fluid dynamics (CFD) problems.

C. *Fast-forwarding algorithms for time evolution*

Let us consider the state representation of the flow variables in terms MPS, the initial state, $\left|u_{\alpha}(\{x_i\}, t_0)\right\rangle$, and the Hamiltonian, $\hat{H}$, for the time evolution operation determined by the mapping of the differential operations into MPOs. Now we can represent the time evolution of the states using spectral-step method [24] as

$$\left|u_{\alpha}(\{x_i\}, t)\right\rangle = e^{i\hat{H}t}\left|u_{\alpha}(\{x_i\}, t_0)\right\rangle. \tag{14}$$

If the Hamiltonian is a combination of multiple MPOs, we can rely on the Trotter–Suzuki expansion to split the operation and obtain an approximate solution [23]. In discrete time, the above formulation becomes

$$\left|u_{\alpha}(\{x_i\}, t_0 + M)\right\rangle = \left(e^{i\hat{H}\Delta t}\right)^M\left|u_{\alpha}(\{x_i\}, t_0)\right\rangle, \tag{15}$$

where Δt is the discrete time-step and M is the total number of time steps to evolve the states. A sample depiction of this is shown in Fig. 2.

The above Trotterization procedure involves multiple MPO operations (of the linear and nonlinear operators), and thus, the circuit depth increases linearly with respect to the time steps, M. This can result in computationally expensive and error prone circuits. However, a strategy can be introduced to ensure that the depth of the circuit remains constant while accurately integrating the Hamiltonian. For example, by combining Trotter-Suzuki decomposition, MPOs, variational approaches, and parallelization, it is feasible to achieve a scalable and efficient implementation of Hamiltonians in quantum circuits. Specifically, to alleviate the linear growth of the circuit depth, we propose to use quantum Krylov subspace methods [25] to fast-forward the time-dependent simulation of fluid flows. The variational fast-forwarding (VFF) algorithm is a typical algorithm that could further reduce the circuit depth by variationally searching for an approximate diagonalization of the time evolution unitaries [26],

$$(e^{i\hat{H}\Delta t})^M \simeq WD(\Delta t)^M W^\dagger = WD(M\Delta t)W^\dagger, \qquad (16)$$

where the unitary with a single time step $U(\Delta t) = e^{i\hat{H}\Delta t}$ is approximated by a diagonal factorization $WD(\Delta t)W^\dagger$. The diagonalization enables a constant circuit depth with respect to the time steps M. While the fast-forwarding technique does provide a great alternative to reduce the circuit depth, it is to be noted that diagonalizing the nonlinear operators can pose challenges and require in-depth research.

Though the VFF algorithm could reduce the circuit depth, it results in an additional optimization procedure. Similar to the widely used VQE algorithms, variational methods usually suffer from optimization problems due to the high nonlinearity of the cost function and stochastic errors (due to decoherence of qubits, control error, readout errors, etc.) [27]. And the situation becomes even worse when multiple channels are included in the optimization. Alternatively, the Krylov subspace method can be employed to bypass the optimization problem.

In linear algebra, an order$-r$ Krylov subspace, generated by a matrix H and a reference vector b, is the linear subspace spanned by the images of b under the first r powers of H [28]. Such subspace is denoted as $\mathcal{K}_r(H,b) \equiv \{b, Hb, H^2b, \cdots, H^{r-1}b\}$, containing all rth-order polynomials of H applied to b. As r gets large, any function of H, $f(H)$, is well approximated by an rth-order polynomial. Choosing $f(H)$ to be a spectral projection, we see that $\mathcal{K}_r$ contains approximates of all eigenvectors for which $|b\rangle$ has nonzero overlap. Hence, the Krylov subspace has been widely used in numerical algorithms to find the solutions to a high-dimensional matrix, such as the generalized minimal residual method (GMRES), Davidson, and quasi-minimal residual (QMR) algorithms [29]. In this context, we leverage the recently developed Quantum Davidson algorithms to obtain diagonalization in Eq. 16.

The last stage of the algorithm is to measure the resulting states from the final MPO. The extreme complexity of the multidimensional fluid flow system representation will lead to a large number of terms in the MPO. This leads to a profound impact on the algorithm's efficiency at the measurement stage. The MPO can result in multiple groups of commutable terms – identifying and coupling these terms using graph theory can help to significantly reduce the measurement cost [30].

Insight 4: Measurement reduction technique for MPOs is possible due to multiple commutable terms, which is important to deploying the proposed algorithm to current noisy intermediate-scale quantum (NISQ) quantum computers and near-term early fault-tolerant quantum computers (EFTQC).

IV. Outlook and Concluding Remarks

We present a conceptual description of a quantum algorithm to treat the nonlinear aspects of PDEs. The incompressible Navier–Stokes equations are used to formulate the algorithm. Taking advantage of tensor networks, the algorithm uses matrix product states (MPSs) to encode the flow variables. We describe quantum-information-inspired efficient encoding techniques for the MPSs. The linear differential terms, approximated using finite difference, are mapped using matrix product operation (MPO). The MPO induces state transitions using ladder operations, and these can be generalized for multi-dimensional and higher order finite difference approximations. The multivariate characteristics of the incompressible Navier–Stokes equations (velocity and pressure) are treated using tensor product contractions. The nonlinear terms are mapped using the quantum nonlinear processing units (QNPUs) [18]. We use LCU instead of variational optimization for creating the QNPU circuits. Given the Hamiltonian expression of the states as combinations of multiple MPOs for the linear and nonlinear terms, fast-forwarding using Krylov subspace methods can be used to simulate the time evolution. The proposed algorithm provides a scalable way to handle complex nonequilibrium phenomena. Nevertheless, for practical considerations, the scalablility of the proposed algorithm needs to be demonstrated for the problem size ranging from small to large, including parameters, such as the depth and qubit counts of MPS/MPO circuits for the nonlinear operators, the structure or physics involved in the problem, the optimization techniques (e.g., for gate optimization, parallelization, and variational approaches for circuits), and the capabilities of the quantum hardware. This will be an immediate follow-up study.

Toward the demonstration of quantum utility for the proposed quantum algorithm on noisy intermediate-scale quantum (NISQ) quantum computers or early fault-tolerant quantum computers (EFTQC), we note that measurement reduction can be an important technique for MPOs to boost the algorithm's efficiency. In addition to the optimization of measurement, achieving the quantum utility also calls for a stable state obtained by time-evolving MPO decompositions. In addition to improving hardware qubit fidelity, one potential near-term remedy at the software layer is to apply the circuit compression techniques [31] to reduce the circuit length according to the on-the-fly quantum noise presented on the quantum device right before running the circuit.

Acknowledgment

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the US

Department of Energy (DOE) under Contract No. DE-AC05-00OR22725. This work was partly supported by the National Science Foundation through awards OAC-2311949 and DMS-2403552-Amendment-ID005. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, LLNL-PROC-867553. Y.Z. acknowledges the support from the US DOE, Office of Science, Basic Energy Sciences, Chemical Sciences, Geosciences, and Biosciences Division under Triad National Security, LLC ("Triad") contract Grant 89233218CNA000001 (FWP: LANLECF7). This manuscript has been authored in part by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US DOE. The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript or allow others to do so for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

References

[1] M. R. Norman, D. A. Bader, C. Eldred, W. M. Hannah, B. R. Hillman, C. R. Jones, J. M. Lee, L. R. Leung, I. Lyngaas, K. G. Pressel, S. Sreepathi, M. A. Taylor, and X. Yuan, "Unprecedented cloud resolution in a GPU-enabled full-physics atmospheric climate simulation on OLCF's summit supercomputer," *International Journal of High Performance Computing Applications*, vol. 36(1), pp. 93–105, 2021.

[2] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical Review Letters*, vol. 103, no. 15, p. 150502, 2009.

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

[4] J. Yepez, "Quantum lattice-gas model for computational fluid dynamics," *Physical Review E*, vol. 63, no. 4, p. 046702, 2001.

[5] Y. Cao, A. Papageorgiou, I. Petras, J. Traub, and S. Kais, "Quantum algorithm and circuit design solving the Poisson equation," *New Journal of Physics*, vol. 15, no. 1, p. 013021, 2013.

[6] B. N. Todorova and R. Steijl, "Quantum algorithm for the collisionless Boltzmann equation," *Journal of Computational Physics*, vol. 409, p. 109347, 2020.

[7] L. Budinski, "Quantum algorithm for the advection–diffusion equation simulated with the lattice Boltzmann method," *Quantum Information Processing*, vol. 20, no. 2, p. 57, 2021.

[8] S. S. Bharadwaj and K. R. Sreenivasan, "Hybrid quantum algorithms for flow problems," *Proceedings of the National Academy of Sciences*, vol. 120, no. 49, p. e2311014120, 2023.

[9] S. K. Leyton and T. J. Osborne, "A quantum algorithm to solve nonlinear differential equations," *arXiv preprint arXiv:0812.4423*, 2008.

[10] J.-P. Liu, H. Ø. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, and A. M. Childs, "Efficient quantum algorithm for dissipative nonlinear differential equations," *Proceedings of the National Academy of Sciences*, vol. 118, no. 35, p. e2026805118, 2021.

[11] W. Itani, K. R. Sreenivasan, and S. Succi, "Quantum algorithm for lattice Boltzmann (QALB) simulation of incompressible fluids with a nonlinear collision term," *Physics of Fluids*, vol. 36, no. 1, 2024.

[12] Y. T. Lin, R. B. Lowrie, D. Aslangil, Y. Subaşı, and A. T. Sornborger, "Koopman von Neumann mechanics and the Koopman representation: A perspective on solving nonlinear dynamical systems with quantum computers," *arXiv preprint arXiv:2202.02188*, 2022.

[13] D. Giannakis, A. Ourmazd, P. Pfeffer, J. Schumacher, and J. Slawinska, "Embedding classical dynamics in a quantum computer," *Physical Review A*, vol. 105, no. 5, p. 052404, 2022.

[14] I. Joseph, "Koopman–von Neumann approach to quantum simulation of nonlinear classical dynamics," *Physical Review Research*, vol. 2, p. 043102, 2020.

[15] R. Steijl, "Quantum algorithms for nonlinear equations in fluid mechanics," *Quantum Computing and Communications*, 2020.

[16] F. Gaitan, "Finding flows of a Navier–Stokes fluid through quantum computing," *npj Quantum Information*, vol. 6, no. 1, p. 61, 2020.

[17] F. Oz, O. San, and K. Kara, "An efficient quantum partial differential equation solver with chebyshev points," *Scientific Reports*, vol. 13, no. 1, p. 7767, 2023.

[18] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, "Variational quantum algorithms for nonlinear problems," *Physical Review A*, vol. 101, no. 1, p. 010301, 2020.

[19] S. M. Guzik, J. Christopher, S. Walters, X. Gao, J. B. Schroder, and R. D. Falgout, "On the use of a multigrid-reduction-in-time algorithm for multiscale convergence of turbulence simulations," *Computers & Fluids*, vol. 261, 2023.

[20] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of Physics*, vol. 326, no. 1, pp. 96–192, 2011.

[21] R. Orús, "A practical introduction to tensor networks: Matrix product states and projected entangled pair states," *Annals of Physics*, vol. 349, pp. 117–158, 2014.

[22] N. V. Tkachenko, J. Sud, Y. Zhang, S. Tretiak, P. M. Anisimov, A. T. Arrasmith, P. J. Coles, L. Cincio, and P. A. Dub, "Correlation-informed permutation of qubits for reducing ansatz depth in the variational quantum eigensolver," *PRX Quantum*, vol. 2, no. 2, p. 020337, 2021.

[23] J. J. García-Ripoll, "Quantum-inspired algorithms for multivariate analysis: from interpolation to partial differential equations," *Quantum*, vol. 5, p. 431, 2021.

[24] J. Weideman and B. M. Herbst, "Split-step methods for the solution of the nonlinear Schrödinger equation," *SIAM Journal on Numerical Analysis*, vol. 23, no. 3, pp. 485–507, 1986.

[25] N. V. Tkachenko, L. Cincio, A. I. Boldyrev, S. Tretiak, P. A. Dub, and Y. Zhang, "Quantum davidson algorithm for excited states," *Quantum Science and Technology*, vol. 9, no. 3, p. 035012, 2024.

[26] C. Cirstoiu, Z. Holmes, J. Iosue, L. Cincio, P. J. Coles, and A. Sornborger, "Variational fast forwarding for quantum simulation beyond the coherence time," *npj Quantum Information*, vol. 6, no. 1, p. 82, 2020.

[27] L. Bittel and M. Kliesch, "Training variational quantum algorithms is NP-hard," *Physical Review Letters*, vol. 127, no. 12, p. 120502, 2021.

[28] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *Journal of Research of the National Bureau of Standards*, vol. 45, no. 5, pp. 255–282, 1950.

[29] Y. Saad, *Iterative methods for sparse linear systems*. SIAM, 2003.

[30] T.-C. Yen, V. Verteletskyi, and A. F. Izmaylov, "Measuring all compatible operators in one series of single-qubit measurements using unitary transformations," *Journal of Chemical Theory and Computation*, vol. 16, no. 4, pp. 2400–2409, 2020.

[31] Z. Hu, Y. Lin, Q. Guan, and W. Jiang, "Battle against fluctuating quantum noise: Compression-aided framework to enable robust quantum neural network," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2023, pp. 1–6.

[32] J. Rissler, R. M. Noack, and S. R. White, "Measuring orbital interaction using quantum information theory," *Chemical Physics*, vol. 323, no. 2-3, pp. 519–531, 2006.

Appendix

Quantum information inspired efficient encoding of MPS states: While the MPS representation using Schmidt decomposition enables exponentially large expressibility capabilities to significantly reduce the data points, the operation as such has exponential cost of $\mathcal{O}(K^{3Dq/2})$ if considering the full-rank approximation. We can leverage quantum-information-inspired techniques to analyze the ranks of the tensors and optimize the ordering of tensors to reduce the bond dimension and, consequently, circuit depths. Recently, we proposed an approach named "PermVQE" [22] to reduce circuit depth. The PermVQE method permutes qubits to solve for the qubit Hamiltonian that maximally localizes correlations in the ground state. The choice of permutations is based on mutual information, a measure of interaction between electrons and/or holes in spin-orbitals [32]. Encoding strongly entangled spin-orbitals into proximal qubits on a quantum chip naturally reduces the circuit depth needed to prepare the ground state. In particular, we demonstrated the beneficial effect of qubit permutations in building a fermionic–adaptive derivative assembled pseudo-Trotter ansatz on a linear qubit connectivity architecture with nearly a twofold reduction in the number of CNOT gates. Here, we propose to leverage the same idea to efficiently reduce the CNOT gates for encoding the MPS on quantum circuits.

After obtaining the initial MPO and MPS, the entanglement map that reflects the correlations among different qubits will be constructed. For this purpose, we calculate the quantum mutual information for all pairs of qubits, which provides a measure of the total correlation. Previous results obtained for the orbital ordering problem in the Density Matrix Renormalization Group (DMRG) method in classical quantum chemistry calculations [32] showed that the quantum mutual information is a reliable parameter to quantify the correlation between two particles.

The quantum mutual information between qubits i and j is defined as follows:

$$I_{ij} = \frac{1}{2}(S_i + S_j - S_{ij})(1 - \delta_{ij}) \,, \tag{17}$$

S_i and S_{ij} are the single-qubit and two-qubit von Neumann entropies, respectively. The Kronecker δ sets all diagonal elements I_{ii} to zero. The single-qubit von Neumann entropy S_i is given by:

$$S_i = -\sum_{\alpha} \lambda_i^{(\alpha)} \log \lambda_i^{(\alpha)} \,, \tag{18}$$

and S_{ij} is defined analogously. Here, λ_i are the normalized singular values associated with the bond. Based on the mutual information values for each pair of qubits, we build an $n \times n$ matrix. This matrix $\mathbf{I} = \{I_{ij}\}$, called the entanglement map, is useful to illustrate the amount and the length scale of the correlations in the approximate function.

To quantify the amount of long-range correlations, we introduce a cost function as follows. For a NISQ device with a given connectivity, let d_{ij} denote the distance between qubits i and j, which can be precisely defined as the number of edges in the shortest path through the connectivity graph between these qubits. Alternatively, one can view d_{ij} as the minimum number of swap gates (plus one) needed to make qubits i and j nearest neighbors. Then, for any qubit connectivity, a cost function can be defined as $C(\mathbf{I}) = \sum_{i<j} f(d_{ij}) I_{ij}$, where $f(\cdot)$ is a monotonously increasing function of d_{ij}. Then, we can define a permutation P as a bijection from the set of qubit indices to itself. The action of P will affect the entanglement map $\mathbf{I}$, and our strategy is to find the best permutation P that minimizes the cost function, i.e., $P_{opt} = \arg\min_P C(P\mathbf{I}P^{-1})$. We will use Spectral Graph Algorithm or Heuristic permutation search with random sampling to optimize the permutations.