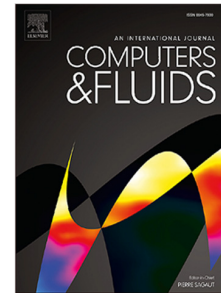# Journal Pre-proof

Incompressible Navier–Stokes solve on noisy quantum hardware via a
hybrid quantum–classical scheme

Zhixin Song, Robert Deaton, Bryan Gard, Spencer H. Bryngelson

Please cite this article as: Z. Song, R. Deaton, B. Gard et al., Incompressible Navier–Stokes solve
on noisy quantum hardware via a hybrid quantum–classical scheme. *Computers and Fluids* (2024),
doi: https://doi.org/10.1016/j.compfluid.2024.106507.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the
addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive
version of record. This version will undergo additional copyediting, typesetting and review before it
is published in its final form, but we are providing this version to give early visibility of the article.
Please note that, during the production process, errors may be discovered which could affect the
content, and all legal disclaimers that apply to the journal pertain.

Incompressible Navier–Stokes solve on
noisy quantum hardware via a hybrid quantum–classical scheme

Zhixin Song[1], Robert Deaton[1,2], Bryan Gard[2], Spencer H. Bryngelson[3,4,5]

[1]School of Physics, Georgia Institute of Technology, Atlanta, GA 30332, USA

[2]CIPHER, Georgia Tech Research Institute, Atlanta, GA 30332, USA

[3]School of Computational Science & Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

[4]Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

[5]George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

Abstract

Partial differential equation solvers are required to solve the Navier–Stokes equations for fluid flow. Recently, algorithms have been proposed to simulate fluid dynamics on quantum computers. Fault-tolerant quantum devices might enable exponential speedups over algorithms on classical computers. However, current and foreseeable quantum hardware introduce noise into computations, requiring algorithms that make judicious use of quantum resources: shallower circuit depths and fewer qubits. Under these restrictions, variational algorithms are more appropriate and robust. This work presents a hybrid quantum–classical algorithm for the incompressible Navier–Stokes equations. A classical device performs nonlinear computations, and a quantum one uses a variational solver for the pressure Poisson equation. A lid-driven cavity problem benchmarks the method. We verify the algorithm via noise-free simulation and test it on noisy IBM superconducting quantum hardware. Results show that high-fidelity results can be achieved via this approach, even on current quantum devices. Multigrid preconditioning of the Poisson problem helps avoid local minima and reduces resource requirements for the quantum device. A quantum state readout technique called HTree is used for the first time on a physical problem. Htree is appropriate for real-valued problems and achieves linear complexity in the qubit count, making the Navier–Stokes solve further tractable on current quantum devices. We compare the quantum resources required for near-term and fault-tolerant solvers to determine quantum hardware requirements for fluid simulations with complexity improvements.

1 Introduction

With the rapid development of hardware platforms and algorithms [1–5], quantum computing has gained enormous attention due to its potential to solve many large and complex problems faster than classical methods. The ability of quantum computers to access an exponentially large Hilbert space and exploit unique quantum properties such as superposition and entanglement are at the heart of many current appealing quantum algorithms. Quantum computing algorithms already promise theoretical speed-up for integer factoring [6], unstructured database search [7], and many other problems of practical interest [8]. Numerous quantum PDE solvers have been proposed based on quantum linear system algorithms. The Harrow–Hassidim–Lloyd (HHL) algorithm [9] claims an exponential speed-up for solving linear systems compared to classical iterative methods such as conjugate gradient [10], regardless of the cost of state preparation and readout the quantum solution encoded in complex amplitudes. HHL and its later improvements [11–14] have been successfully adopted to solve various linear PDEs [15–20] and aid in the solution of nonlinear ones [21–24].

Quantum algorithms for solving computational fluid dynamics (CFD) problems have also gained

---

Email: shb@gatech.edu (Spencer H. Bryngelson)

Code available at https://github.com/comp-physics/NISQ-Quantum-CFD

much attention due to their nonlinear and non-Hermitian nature [25]. Some of these works focus on longer-term lattice-based methods, including the lattice Boltzmann method (LBM) and lattice gases [26, 26–34], including linearization techniques for handling advective terms [35–37] or otherwise treating nonlinearities [38]. Li et al. [24] claims a potential quantum advantage for fluid simulation when applying a PDE solver based on quantum linear system algorithms [17] to Carleman-linearized LBM. Although they are likewise long-term algorithms, methods based on oracles have also been proposed [39] and later developed into quantum circuits for implementation [40]. Oz et al. [41] adopts the quantum PDE algorithm from [39] to solve 1D Burgers' equation and Basu et al. [42] further extend similar approaches to compute dispersal of submarine volcanic tephra. Perhaps closest to the work presented here is that of Lapworth [23], who proposed a hybrid quantum-classical algorithm for solving incompressible Navier–Stokes equations based on the SIMPLE (Semi-Implicit Method for Pressure Linked Equations) algorithm [43] and an HHL linear solver.

The current noisy-intermediate scale quantum (NISQ) hardware capability [44] has qubits with meaningfully high error rates. NISQ methods for solving fluid dynamics problems on actual hardware are limited, and we focus on them here. Most PDE solves suitable to the NISQ-era use variational strategies [5, 45], including the use of quantum algorithms for pressure Poisson solves in the spectral [46, 47] and physical domain [23, 48].

These variational quantum algorithms use shallow parametrized quantum circuits to prepare a general quantum state, combined with classical optimizers and a problem-tailored cost function to (iteratively) prepare the solution. Among them, the Variational Quantum Linear Solver (VQLS) [49] is a near-term solution, in some cases, for solving linear systems. Combined with linearization techniques, one can adopt VQLS to solve simple fluid dynamics problems [50–52]. To address general nonlinear problems using variational algorithms, Lubasch et al. [53] propose the quantum nonlinear processing unit to evaluate nonlinear cost functions given multiple trial states. Based on that, Jaksch et al. [54] develop the variational quantum computational fluid dynamics algorithm to solve the viscous Burgers' equation.

This work proposes a hybrid quantum–classical computing scheme for solving incompressible Navier–Stokes equations on NISQ hardware. One can combine classical pressure projection methods and variational quantum algorithms to solve the pressure Poisson equation. Unlike previous approaches that focus on fault-tolerant quantum computers, we perform the calculation on IBM's superconducting quantum processors to solve a 2D flow problem. Our results show that preconditioning for the linear system can improve the trainability and convergence speed of variational quantum algorithms. The hybrid algorithm requires a quantum state readout for each simulation time step.

We, in part, address the well-known readout problem [55] via a new and efficient quantum state reconstruction method for the real-valued statevector with *linear* time complexity [56]. Conventional methods such as Quantum State Tomography (QST) [57] reconstruct the full quantum state as a complex-valued density matrix at the cost of projecting it into an exponential number of Pauli bases. This improved method instead leverages the fact that many scientific and engineering problems involve only real-valued states. Then, one separates the readout procedure into amplitude estimation by measuring in the standard computational basis and determining the relative signs between each amplitude. This approximate readout method, discussed further in section 3.3, simplifies the measurement and can be applied to a broad range of applications beyond PDE solves. We introduce quantum decoherence noise into the hybrid algorithm for the time-dependent PDE solve. This is a critical component for designing NISQ-type quantum CFD algorithms and has seen little attention in the literature [58, 59].

This paper begins with section 2, reviewing classical CFD solvers for incompressible Navier–Stokes equations, including Chorin's projection method [60] and the SIMPLE algorithm. We introduce the hybrid quantum–classical scheme for incompressible fluid simulations on NISQ devices in section 3. Section 4 summarizes the simulation and hardware results for a 2D lid-driven cavity flow. We discuss quantum decoherence noise modeling and the error mitigation and suppression methods we adopt in our hybrid approach. We further conduct resource estimation to analyze the hardware requirement for solving the same problem with the HHL algorithm. Section 5 discusses the presented method's limitations and possible extensions.

2 Background

2.1 The incompressible Navier–Stokes equations

The incompressible Navier–Stokes equations are a set of coupled nonlinear PDEs that describe the conservation of mass and momentum. In dimensionless form, the mass conservation equation is

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

and the momentum conservation equation is

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla p + \frac{1}{\mathrm{Re}}\nabla^2 \boldsymbol{u}, \tag{2}$$

where $\boldsymbol{u} = (u, v)$ is the flow velocity vector, ρ is the density, p is the pressure, and Re is the Reynolds number (ratio of inertial to viscous effects).

2.2 Pressure projection methods for solving incompressible flow

The main difficulty of solving the incompressible Navier–Stokes equations is often the coupling of velocity and pressure under incompressibility constraint (1). To transform the governing equations and avoid the pressure-velocity coupling, derived quantities such as the stream function and vorticity could be used instead of the primitive variables $\{u, v, p\}$. However, this approach poses difficulties in geometry modeling and setting the boundary conditions for 3D problems.

Two numerical schemes for solving viscous incompressible flow equations are pressure-based projection methods and artificial compressibility [61]. The former approach decouples the velocity and pressure calculation using a fractional step or pseudo-time-stepping. Chorin's projection method [60] is a well-known pressure projection method, which follows as

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^n}{\Delta t} = -\frac{1}{\rho}\nabla p^{n+1} - (\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \mathrm{Re}\nabla^2 \boldsymbol{u}^n, \tag{3}$$

with explicit Euler time discretization and $\boldsymbol{u}^n$ the velocity at n^{th} time step. The time-step Δt is chosen following the CFL criterion: $\Delta t \left(\sum_k \|\boldsymbol{u}\|_{\max}/\Delta x_k\right) \leq 1$, where Δx_k is the grid spacing in each spatial dimension k. The first step is solving for an intermediate velocity $\boldsymbol{u}^*$ using the discretized momentum equation without the pressure gradient term ($\nabla p^{n+1} = 0$):

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \mathrm{Re}\nabla^2 \boldsymbol{u}^n. \tag{4}$$

This velocity field $\boldsymbol{u}^*$ is usually not divergence-free.

3
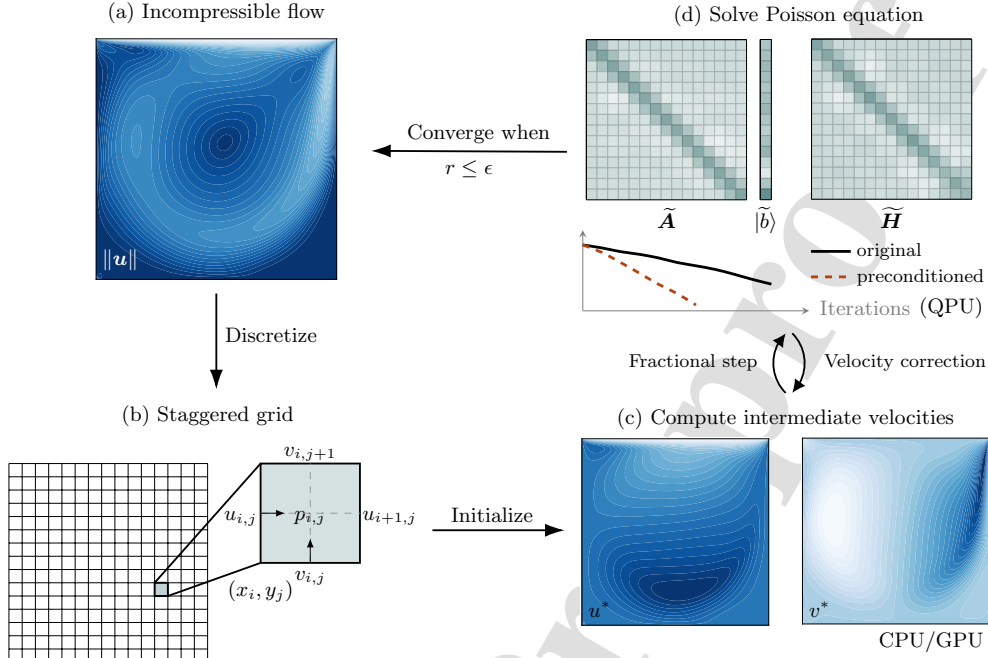
Figure 1: We show a schematic of the hybrid quantum–classical scheme for solving incompressible flow problems. (a) Velocity contour of the 2D lid-driven cavity flow at Re = 1000. (b) A staggered grid allays the well-known checkerboard pressure problem. The pressure is stored at the cell center, and the velocities are stored at the cell interfaces. (c) Calculate intermediate velocities and proceed to the classical computer's next fractional time step. (d) Solve the Poisson equation with proper preconditioning on a quantum computer and obtain velocity corrections. We iterate until the convergence criterion reaches the threshold ϵ.

In the second step, the pressure Poisson equation is derived by imposing incompressibility constraint (1) on the velocity at the next time-step

$$\nabla^2 p^{n+1} = \frac{\rho}{\Delta t} \nabla \cdot \boldsymbol{u}^*. \tag{5}$$

This projection step is achieved by solving the Poisson equation and is often more computationally expensive than other steps. Once we solve the Poisson problem, we correct the velocity from $\boldsymbol{u}^*$ to the next time step as

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \frac{\Delta t}{\rho} \nabla p^{n+1}. \tag{6}$$

We iterate the above steps until the residual r converges to a tolerance threshold, here chosen to be $\epsilon = 10^{-5}$, so our results are insensitive to this particular choice. Residual measures how far the current solution is from satisfying the discrete form of the governing equations and is widely used as a convergence criterion in CFD. For steady-state problems, one could also monitor the root mean

4

square difference between two consecutive time steps

$$r_{\mathrm{RMS}} = \frac{\sqrt{\sum_{ij} |\phi_{i,j}^{n+1} - \phi_{i,j}^n|^2}}{\sqrt{\sum_{ij} |\phi_{i,j}^n|^2}}, \tag{7}$$

where the monitored physical quantity ϕ are the velocities (u, v) or the pressure p.

Different pressure projection schemes exist, such as the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) algorithm [43]. Compared to Chorin's projection method, the SIMPLE algorithm uses semi-implicit time-stepping and the velocity correction in (4) accounts for the pressure solution from the last step. Both methods can be integrated into our proposed hybrid quantum–classical scheme, see Lapworth [23] for further discussion.

Another common approach is artificial compressibility or pseudo-compressibility methods [62], where one adds a time-derivative of the pressure term to the continuity equation (1) as

$$\frac{\partial p}{\partial t} + \beta \, \nabla \cdot \boldsymbol{u} = 0, \tag{8}$$

where β is the artificial compressibility. Such a formulation relaxes the strict requirement to satisfy mass conservation in each step, and the implicit schemes developed for compressible flows can be directly implemented. However, this method is computationally expensive for transient problems due to dual time-stepping since the pressure field has to go through one complete steady-state iteration cycle in each time step.

2.3 Quantum solutions for the pressure Poisson equation

This study focuses on pressure projection schemes. We transform the pressure Poisson equation into a large-scale system of linear algebraic equations with spatial discretization

$$\boldsymbol{A}|p\rangle = |b\rangle, \tag{9}$$

where $\boldsymbol{A}$ is a sparse Laplacian matrix that only depends on boundary conditions and discrete formats (finite difference or finite volume method), $|p\rangle$ is a vector composed of pressures on all discrete grids, and $|b\rangle$ is a vector containing intermediate velocity divergences and boundary conditions. We assume the mesh is Cartesian when using the staggered grid in fig. 1 (b), and one can further extend our method by adopting Rhie–Chow interpolation [63].

Among the family of variational quantum algorithms, the Variational Quantum Eigensolver (VQE) [64] and Variational Quantum Linear Solver (VQLS) [49] are promising candidates for solving linear systems on NISQ devices. These two approaches both solve the quantum linear system problem $\boldsymbol{A}|x\rangle = |b\rangle$ by encoding the solution as the ground-state $|\psi_g\rangle = |x\rangle$ of a problem Hamiltonian $\boldsymbol{H}$. VQE was created to determine the ground-state energy of chemistry molecules based on the Rayleigh–Ritz variational principle [65] and later adopted to solve Poisson equations [66, 67]. Given the Hamiltonian mapped into the Pauli basis

$$\boldsymbol{H} = \sum_{l=1}^{L_h} \alpha_l \boldsymbol{H}_l, \tag{10}$$

where each Hermitian matrix $\boldsymbol{H}_l$ is decomposed in the format of N-qubit Pauli strings $\boldsymbol{H}_l \in \langle P_1 \otimes P_2 \cdots \otimes P_N : P_\ell \in \{I, X, Y, Z\} \rangle$ and each coefficient α_l is a real number, VQE iteratively

5
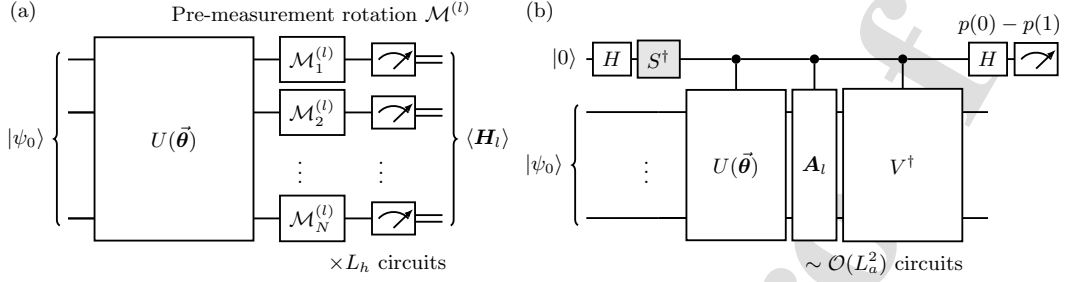
Figure 2: Quantum circuit structure for the (a) VQE and (b) VQLS algorithms.

prepares the solution with a parameterized quantum circuit (PQC) $U(\vec{\boldsymbol{\theta}})$ by minimizing the expectation value $\sum_l \alpha_l \langle 0|U^\dagger(\vec{\boldsymbol{\theta}}) \boldsymbol{H}_l U(\vec{\boldsymbol{\theta}})|0\rangle = \sum_l \alpha_l \langle \boldsymbol{H}_l\rangle$ as the energy measure.

Figure 2 (a) illustrates the VQE quantum circuit to measure a single term $\langle \boldsymbol{H}_l\rangle$. To compute the total energy inside each iteration, VQE executes L_h circuits. Thus, the total runtime can be estimated as $t_{\text{VQE}} = N_{\text{itr.}} L_h t_{\text{circ.}}$, where $N_{\text{itr.}}$ is the number of iterations for VQE to reach ground state and $t_{\text{circ.}}$ is the hardware runtime for each circuit. The pre-measurement rotation gates in Pauli basis $\mathcal{M}^{(l)} \in \{H, HS^\dagger\}$ change the measurement basis for each qubit from computational Z-basis to X and Y, respectively. H (Hadamard gate) and $S^\dagger$ ($-\pi/2$ phase gate) are two basic quantum gate operations. For example, the pre-measurement rotation gate sequence reads as $\mathcal{M}_1^{(l)} \otimes \mathcal{M}_2^{(l)} \otimes \mathcal{M}_3^{(l)} = I \otimes H \otimes HS^\dagger$ given $\boldsymbol{H}_l = Z \otimes X \otimes Y$. Notice these pre-measurement rotations only consist of single-qubit gates when $\boldsymbol{H}_l$ is decomposed in the Pauli basis. Decomposing $\boldsymbol{H}_l$ into another basis, such as $\{\sigma_+, \sigma_-\}$, where $\sigma_\pm = (X \pm iY)/2$, could lead to a more efficient decomposition such that $L_h = \mathcal{O}(\text{poly}(N))$ comparing to the worst case scenario for Pauli decomposition of a dense matrix $L_h = \mathcal{O}(4^N)$. Consequently, the pre-measurement rotation circuit is more complex and contains multi-qubit gates [66]. Last, one computes the expectation value $\langle \boldsymbol{H}_l\rangle$ by measuring in the standard Z-basis and post-process the probabilities

$$\langle \boldsymbol{H}_l\rangle = \sum_{z \in \{0,1\}^N} (-1)^{w(z)} p(z), \tag{11}$$

where $p(z)$ is the probability of being in the state $|z\rangle = |z_1 z_2 \cdots z_N\rangle$ with $z_i \in \{0,1\}$ and $w(z)$ is the Hamming weight of the binary string z.

VQLS inherits the same variational principle, though includes explicit construction of a N-qubit circuit V to load $|b\rangle = V|0\rangle^{\otimes N}$ and additional quantum subroutines such as the Hadamard test to estimate the cost function $C(\vec{\boldsymbol{\theta}}) = 1 - |\langle b|\Psi\rangle|^2$, where $|\Psi\rangle = \boldsymbol{A}|\psi\rangle/\sqrt{\langle \psi|\boldsymbol{A}^\dagger \boldsymbol{A}|\psi\rangle}$ and $|\psi\rangle = U(\vec{\boldsymbol{\theta}})|0\rangle^{\otimes N}$ is the quantum iterative solution prepared using the same parameterized quantum circuit $U(\vec{\boldsymbol{\theta}})$ as in VQE. Here, $|\langle b|\Psi\rangle|^2$ is the cosine similarity between two vectors $|b\rangle$ and $|\Psi\rangle$. Hence, the cost function has an operational meaning similar to the absolute residual $\||b\rangle - \boldsymbol{A}|\psi\rangle\|$. It is assumed that matrix $\boldsymbol{A}$ is decomposed into a linear combination of unitaries (LCU) $\boldsymbol{A} = \sum_{l=1}^{L_a} c_l \boldsymbol{A}_l$, where each $\boldsymbol{A}_l$ is a unitary matrix and c_l is a complex number. The VQLS cost function follows as

$$C(\vec{\boldsymbol{\theta}}) = 1 - \frac{\sum_{l,l'} c_l c_{l'}^* \langle 0|U^\dagger(\vec{\boldsymbol{\theta}}) \boldsymbol{A}_{l'}^\dagger V|0\rangle \langle 0|V^\dagger \boldsymbol{A}_l U(\vec{\boldsymbol{\theta}})|0\rangle}{\sum_{l,l'} c_l c_{l'}^* \langle 0|U^\dagger(\vec{\boldsymbol{\theta}}) \boldsymbol{A}_{l'}^\dagger \boldsymbol{A}_l U(\vec{\boldsymbol{\theta}})|0\rangle}. \tag{12}$$

6

Last, one repeats the Hadamard test $\mathcal{O}(L_a^2)$ times to compute the real and imaginary parts of the above terms.

The VQLS circuit for computing a single numerator term $\mathrm{Re}\langle 0|V^\dagger \boldsymbol{A}_l U(\vec{\boldsymbol{\theta}})|0\rangle$ is illustrated in fig. 2 (b) given the initial state $|\psi_0\rangle = |0\rangle^{\otimes N}$. This circuit represents the Hadamard test using one ancilla qubit initialized in $|0\rangle$ and a controlled unitary sandwiched with two Hadamard gates. Then, one can calculate $\mathrm{Re}\langle \psi_0|V^\dagger \boldsymbol{A}_l U(\vec{\boldsymbol{\theta}})|\psi_0\rangle = p(0) - p(1)$, where $p(0)$ and $p(1)$ are the probabilities of measuring 0 and 1 in the ancilla qubit. The shaded extra $S^\dagger$ gate appears only when estimating imaginary parts. To evaluate denominator terms $\langle 0|U^\dagger(\vec{\boldsymbol{\theta}})\boldsymbol{A}_{l'}^\dagger \boldsymbol{A}_l U(\vec{\boldsymbol{\theta}})|0\rangle$ in (12), one needs to modify the circuit by applying $U(\vec{\boldsymbol{\theta}})$ in the non-controlled way and replacing $V^\dagger$ circuit block with $\boldsymbol{A}_{l'}^\dagger$. Although VQLS presents as a natural solver for (9), it requires meaningfully more quantum resources than VQE on NISQ devices (due to extra controlled gates in Hadamard test) for general engineering problems [51]. For example, the VQLS circuit runtime for our problem of interest exceeds the coherence time of NISQ hardware. Section 4.4 provide a detailed quantum resource estimation comparison between these variation techniques.

3 Hybrid quantum–classical Scheme

3.1 Overview

We propose a hybrid quantum–classical scheme for solving the incompressible Navier–Stokes problem. This scheme combines the projection method with the VQE linear system algorithm. Figure 1 illustrates our workflow. We compute the intermediate velocity field $\boldsymbol{u}^*$ on a classical computer. We form the discretized Poisson equation and encode the solution as the ground state of an effective Hamiltonian $\boldsymbol{H}$ as

$$\boldsymbol{H} = \boldsymbol{A}^\dagger(I - |\widehat{b}\rangle\langle\widehat{b}|)\boldsymbol{A}, \tag{13}$$

where $\boldsymbol{A}^\dagger$ is the Hermitian transpose of the $\boldsymbol{A}$ operator and $|\widehat{b}\rangle = |b\rangle/\|b\|$. We use a parameterized quantum circuit $U(\vec{\boldsymbol{\theta}})$ to iteratively prepare the ground state of this Hamiltonian $|\psi(\vec{\boldsymbol{\theta}})\rangle = U(\vec{\boldsymbol{\theta}})|\psi_0\rangle$. By optimizing over the parameters $\vec{\boldsymbol{\theta}} = (\theta_1, \ldots, \theta_m)$ to minimize the cost function

$$\underset{\vec{\boldsymbol{\theta}}}{\arg\min}\, C(\vec{\boldsymbol{\theta}}) = \underset{\vec{\boldsymbol{\theta}}}{\arg\min}\, \langle\psi(\vec{\boldsymbol{\theta}})|\boldsymbol{H}|\psi(\vec{\boldsymbol{\theta}})\rangle, \tag{14}$$

we obtain an approximate solution $|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle \approx |\psi_g\rangle$ when $C(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}}) \leq \gamma$. γ is the termination condition for the optimization loop similar to the residual tolerance in classical iterative linear solvers. The other common termination condition for VQE is specifying the maximum number of iterations.

To retrieve the pressure solution $|p\rangle$, we require a normalization factor $\mathcal{N}_p$ to scale the quantum solution such that $\mathcal{N}_p|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle \approx |p\rangle$. The normalization is computed via the ratio of the largest element of $\boldsymbol{A}|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle$ and $|\widehat{b}\rangle$. The cost function $C(\vec{\boldsymbol{\theta}})$ has an operational meaning of distance measure between the exact and prepared solutions. One can show the following lower bound holds in general [49],

$$C(\vec{\boldsymbol{\theta}}) \geq \frac{1}{4\kappa^2}\mathrm{Tr}^2\left[\left|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle\langle\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\right| - |\psi_g\rangle\langle\psi_g|\right], \tag{15}$$

where $\mathrm{Tr}[\cdot]$ is the trace distance and $\kappa(\boldsymbol{A}) := \|\boldsymbol{A}\|\,\|\boldsymbol{A}^{-1}\|$ is the matrix condition number. The

7
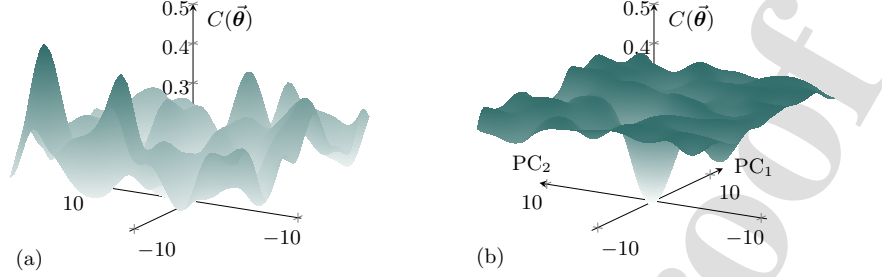
Figure 3: Comparison of the (a) original optimization landscape and (b) preconditioned landscape. The preconditioned landscape (b) has fewer local minima and thus exhibits better trainability. The global minima are located at the center $(0,0)$ in this illustration. PC_j is the j-th principal component.

parametrized circuit $U(\vec{\boldsymbol{\theta}})$ is decomposed as a chain of unitary operators

$$U(\vec{\boldsymbol{\theta}}) = \prod_{\ell=1}^{L_{\text{cir.}}} U_\ell(\theta_\ell) W_\ell, \tag{16}$$

where $U_\ell(\theta_\ell) = \exp(-i\theta_\ell/2 P_\ell)$ with the Pauli operator $P_\ell \in \{X, Y, Z\}$ and W_ℓ denotes a fixed (non-parameterized) operator such as two-qubit gates that provide entanglement.

3.2 Preconditioning the linear system

Preconditioning techniques are widely used in iterative sparse linear system solvers. The common practice is finding a preconditioner $\boldsymbol{M}^{-1}$ and applying it to the original linear system as

$$\underbrace{\boldsymbol{M}^{-1}\boldsymbol{A}}_{\widetilde{\boldsymbol{A}}} |x\rangle = \underbrace{\boldsymbol{M}^{-1}|b\rangle}_{|\widetilde{b}\rangle}, \tag{17}$$

such that the matrix condition number of the newly assembled matrix is well behaved $\kappa(\widetilde{\boldsymbol{A}}) \ll \kappa(\boldsymbol{A})$. This technique reduces computational complexity since classical sparse linear solvers typically have a $\sqrt{\kappa}$ dependence, compared to the κ^2 dependence for the HHL algorithm and κ dependence for more advanced quantum linear systems algorithms based on discrete adiabatic theorem [68].

Clader et al. [69] introduced preconditioning to quantum linear system algorithms via a sparse approximate inverse (SPAI) method, where $\boldsymbol{M}^{-1}$ is constructed by solving a least-squares procedure for each row of the matrix $\boldsymbol{A}$ such that $\|\boldsymbol{M}^{-1}\boldsymbol{A} - I\|_F^2$ is minimized. For NISQ linear solvers, Hosaka et al. [70] showed that using the incomplete LU factorization as a preconditioning technique can aid VQLS by preparing a higher fidelity solution with fewer iterations in the optimization loop. The preconditioning involves an LU factorization of $\boldsymbol{A}$ (lower and upper triangular): $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$. One can then drop certain elements in $\boldsymbol{L}$ and $\boldsymbol{U}$ based on the sparsity pattern of $\boldsymbol{A}$ to construct the preconditioner as $\boldsymbol{M} = \widetilde{\boldsymbol{L}}\widetilde{\boldsymbol{U}} \approx \boldsymbol{A}$.

The present hybrid quantum–classical scheme sees an improvement over ILU via a smoothed aggregation algebraic multigrid (AMG) preconditioner. The AMG used herein constructs a hierarchy V-cycle of progressively coarser grids, where smooth error components are associated with low-frequency eigenmodes of matrix $\boldsymbol{A}$ that are otherwise challenging to eliminate on the fine grid. The procedure groups fine-grid points into aggregates to form the coarse grid and then applies a smoother to improve the interpolation between the fine and coarse grids. Coarse-level operators are

8

created using Galerkin projection. An appropriate multigrid method accelerates the linear solver convergence so long as the operator is elliptic positive definite. With AMG preconditioning the effective Hamiltonian is

$$\widetilde{\boldsymbol{H}} = \widetilde{\boldsymbol{A}}^{\dagger}(I - |\widetilde{b}\rangle\langle\widetilde{b}|)\widetilde{\boldsymbol{A}}. \tag{18}$$

Multigrid methods have been recently adopted to variational ansatz design to improve solution quality [71] and enhance trainability [58].

Here, we use AMG preconditioning to improve the trainability of variational algorithms by reducing the local traps illustrated in fig. 3. In fig. 3, the optimization landscape is visualized based on principal component analysis (PCA) of a successful optimization trajectory using the ORQVIZ package [72]. The scale of the landscapes is normalized by a factor of $\|\boldsymbol{H}\|$ and $\|\widetilde{\boldsymbol{H}}\|$ for visualization purposes, but not repeated for the actual simulations. We only need to marshal the preconditioner once when using Chorin's projection method as $\boldsymbol{A}$ is the same for all time steps. For (semi-)implicit time-stepping tools, such as the SIMPLE algorithm, preconditioning can introduce overhead as $\boldsymbol{M}^{-1}$ must be determined separately for each pseudo-time-step.

3.3 Efficient quantum state readout

As discussed in section 1, an efficient readout method is key for our (or perhaps any current) hybrid quantum–classical algorithm. We introduce the Hadamard tree method (HTree) to address this bottleneck [56]. Standard quantum state tomography (QST) reconstructs the density matrix ρ of arbitrary quantum state with complex amplitudes. So, QST requires collecting $\mathcal{O}(3^N)$ expectation values by measuring each qubit with the Pauli basis $\{X, Y, Z\}$, where N is the number of qubits involved. Reconstructing a physical quantum state satisfying $\mathrm{Tr}[\rho] = 1$ requires a maximum likelihood post-processing technique. HTree instead reconstructs real-valued quantum states $|\psi\rangle = \sum_j \psi_j |j\rangle, \psi_j \in \mathbb{R}$ by sampling the magnitude of state amplitude $|\psi_j|$ according to measurement distribution $|\psi_j|^2$. Hadamard gates on the $(N-k)$-th qubits determine the relative sign between amplitudes ψ_j and ψ_{j+2^k}. We compare the runtime- between these two readout methods on real quantum hardware and summarize the results in table 1. We use the Qiskit Experiments package to implement QST [73].

Method	Qubits: 2	3	4	5	6
QST	3	10	93	310	939
HTree	11	12	13	14	17

Table 1: Runtime (seconds) comparison between two readout methods on 27-qubit `ibmq_kolkata`.

4 Results

4.1 CFD benchmark

Figure 4 shows the 2D lid-driven cavity problem. We consider a square domain with an edge length L, and the upper lid moves with velocity U in the horizontal (x) direction. The other three walls are solid and entail a Dirichlet boundary condition on velocity $\boldsymbol{u}\big|_{\partial\Omega} = 0$. The fluid is initially quiescent with $u = v = 0$. We consider a viscous fluid with kinematic viscosity ν and seek solutions to the incompressible Navier–Stokes equations (1) and (2). Solutions are characterized by the Reynolds number $\mathrm{Re} = UL/\nu$.
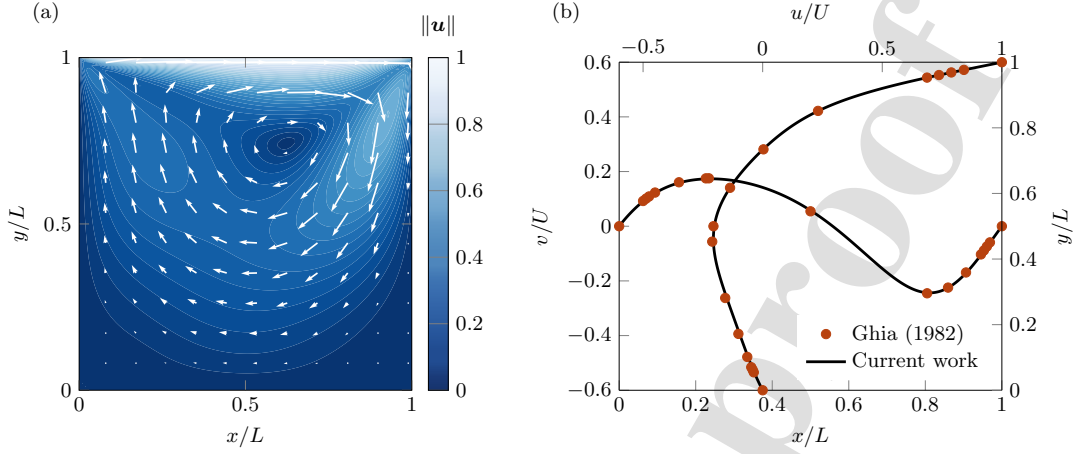
Figure 4: Lid-driven cavity flow at Re = 100 over a 100×100 grid. (a) Velocity magnitude $\|\boldsymbol{u}\| = \sqrt{u^2 + v^2}$ at $t = 10$. (b) Benchmark results and a comparison against a reference solution. The horizontal curve is y-direction velocity v along the horizontal line through the geometric center of the cavity at $y = 0.5$. The vertical curve compares the x-direction velocity u along the vertical line through the geometric center of the cavity at $x = 0.5$ with the reference values of Ghia et al. [74]. The convergence criterion is $r_{\mathrm{RMS}} \leq 10^{-5}$.

We discretize the domain via a uniform staggered grid with n_k grid points in each spatial dimension k. The computational domain thus comprises n_k^2 equally spaced points with grid spacing $\Delta x_k = L/(n_k + 1)$. For a scale resolved simulation, the number of grid points n_k can be determined by resolving the Kolmogorov length scale η as $L/\eta \sim \mathcal{O}(\mathrm{Re}^{3/4})$. We conduct simulations at Re = 100 and verify the solution against the established benchmark of Ghia et al. [74].

4.2 Simulation results

In this section, we conduct numerical simulations that apply the hybrid quantum–classical CFD solver to simulate lid-driven cavity flow. The parameterized quantum circuits (PQC) $U(\vec{\boldsymbol{\theta}})$, also called the ansatz, used in this study is a specific type of hardware-efficient ansatz (HEA) [75], which consists of the gate set $\mathcal{S}_{\mathrm{gate}} \in \{R_y(\theta_\ell), \mathrm{CNOT}\}$. We provide a more detailed analysis to design PQC given hardware resource constraints in Appendix A. All simulations, including optimization loops, are conducted using Qiskit [76] as an open-source toolkit for quantum computing.

We first solve the pressure Poisson problem for Re = 100 at $t = 6$ using the Qiskit statevector simulator. The preconditioning effect is analyzed by repeating the same experiment multiple times with random initial parameters $\theta_\ell \in [-\pi, \pi]$. We report the convergence history of VQE to prepare the ground state solution with fidelity $F \geq 99\%$ in fig. 5. The state fidelity F is a cosine distance measure between the true ground-state $|\psi_g\rangle$ and prepared state $|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle$

$$F\left(|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle, |\psi_g\rangle\right) = \left|\langle\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})|\psi_g\rangle\right|^2. \tag{19}$$

Among the different choices of optimizers, we use the L-BFGS-B optimizer [77], which we find to consistently outperform other optimizers, like COBYLA and Adam, for statevector simulation.

Figure 5 show that a proper preconditioner can reduce VQE iterations. We observe the correlation between cost value and other standard distance measure metrics, which include the state fidelity F
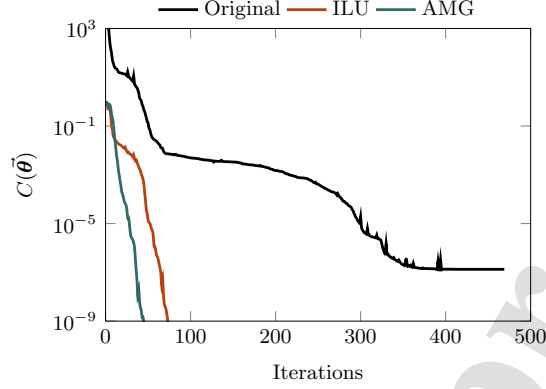
Figure 5: Preconditioning helps VQE converge faster to solve the linear system on a 5×5 mesh. The data are averaged from five VQE runs with random initialization. The PQC is chosen to be the real-amplitude ansatz with five repetitions. The black line shows results without preconditioning. ILU is an incomplete LU factorization and AMG is algebraic multigrid. The number of iterations reported here could differ from other literature recordings of the number of function evaluations. For the L-BFGS-B optimizer, each iteration requires about 30 evaluations to approximate gradient information.

as $C(\vec{\boldsymbol{\theta}}) \approx 1 - F$ and the ℓ_2 norm as $C(\vec{\boldsymbol{\theta}}) \approx \||\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle - |\psi_g\rangle\|_2^2$ when the AMG preconditioner is applied. The preconditioner aids in the preparation of a higher-fidelity solution by enlarging the spectral gap

$$\Delta(\boldsymbol{H}) = \log_{10} \frac{|\lambda_1(\boldsymbol{H})|}{|\lambda_0(\boldsymbol{H})|}, \tag{20}$$

where λ_0 is the smallest eigenvalue of $\boldsymbol{H}$ and λ_1 is the second smallest eigenvalue. A large spectral gap means more space for the classical optimizer to explore the solution once $C(\vec{\boldsymbol{\theta}}) \leq \lambda_1$. We further analyze the effect of preconditioners at different mesh sizes in table 2. The AMG preconditioner consistently creates a larger spectral gap and keeps the matrix condition number low.

Mesh	N_{VQE}	Δt	$\kappa(\boldsymbol{A})$	$\kappa(\widetilde{\boldsymbol{A}}_{\mathrm{ILU}})$	$\kappa(\widetilde{\boldsymbol{A}}_{\mathrm{AMG}})$	$\Delta(\boldsymbol{H})$	$\Delta(\widetilde{\boldsymbol{H}}_{\mathrm{ILU}})$	$\Delta(\widetilde{\boldsymbol{H}}_{\mathrm{AMG}})$
5×5	4	0.2	4.43×10^2	8.01	**1.13**	11.35	14.89	**15.37**
9×9	6	0.1	3.98×10^3	51.51	**1.33**	10.32	13.83	**15.28**
17×17	8	0.05	3.26×10^4	2.94×10^2	**2.46**	9.97	12.17	**15.26**
33×33	10	0.01	2.62×10^5	1.53×10^3	**7.35**	9.12	10.75	**14.89**
65×65	12	0.005	2.10×10^6	7.56×10^3	**33.68**	7.70	9.89	**14.21**

Table 2: Scaling analysis of two preconditioners with different mesh sizes at $\mathrm{Re} = 100$ and $t = 6$, where N_{VQE} is the number of qubits required for VQE to prepare the solution and Δt is the time step size.

Next, we will consider more realistic cases, including finite sampling noise and quantum decoherence noise. We repeat the same experiment on the Qiskit QASM simulator and switch to the COBYLA optimizer due to its noise resilience performance. Figure 6 shows the convergence history. In the noise-free simulation, VQE can converge to an optimal cost $C(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})$ of accuracy up to 10^{-4} and a state fidelity of 99.99% under 700 iterations. If we let the optimizer keep going until 1600 iterations, it will converge to $C(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}}) = 4 \times 10^{-8}$ with state infidelity $1 - F = 4.14 \times 10^{-8}$. When the state fidelity is higher than $F \geq 99.99\%$, we will report the state infidelity $1 - F$ instead. The

11
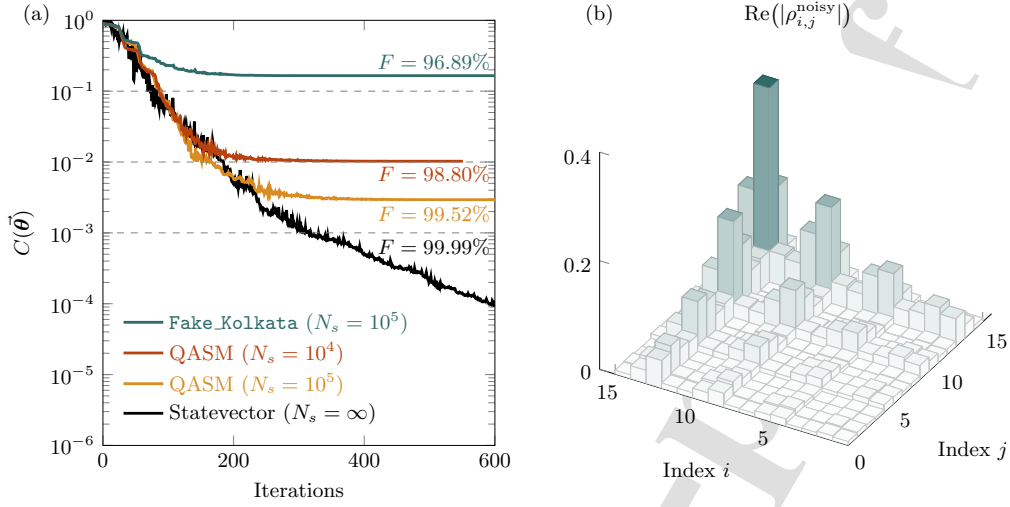
Figure 6: (a) Effect of quantum/classical noise on solving the 4-qubit Poisson problem at $t = 6$ using VQE. (b) Tomography result of the state $\rho_{\mathrm{noisy}} = \mathcal{M}_{\mathrm{HTree}}\big(|\psi(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}})\rangle\big)$ prepared by noisy training on `Fake_Kolkata`, where $\mathcal{M}_{\mathrm{HTree}}(\cdot)$ represent the efficient readout method discussed in section 3.3.

sampling-based simulation limits this accuracy to $C(\vec{\boldsymbol{\theta}}_{\mathrm{opt.}}) \geq 1/\sqrt{N_s}$, where N_s is the shot number. Due to the probabilistic nature of quantum computing, one needs to collect many samples to recover the statistical information stored in the quantum state. This limitation is common for variational algorithms due to the accuracy required for the prepared ground state, though one can mitigate this via more samples. We last consider quantum noise from a real quantum device, the 27-qubit `ibmq_kolkata` with past calibration data. This experiment results in a solution with state fidelity 96.89% compared to the true ground state solution. We repeat the same noisy VQE experiment multiple times and it typically leads to $3\% \sim 5\%$ relative forward errors for solving the linear system at a single time-step.

4.2.1 Noise modeling on quantum computer

Two main categories of quantum noise can be modeled at the circuit level, ignoring all hardware defects: coherent error and decoherence error. The former can be caused by miscalibration of the hardware and leads to an undesired over-rotation applied to the circuit $U(\vec{\boldsymbol{\theta}}) \rightarrow U(\vec{\boldsymbol{\theta}} + \delta\vec{\boldsymbol{\theta}})$, where $\delta\vec{\boldsymbol{\theta}}$ sampled from a Gaussian distribution. Notably, variational quantum algorithms are insensitive to such coherent errors by design [45]. Unwanted interactions between the quantum system and the environment cause decoherence errors. These errors are the most problematic on NISQ hardware and could lead to training issues for VQE. Thus, we focus on modeling them in this section. Decoherence errors can no longer be described as unitary quantum gates and are typically treated as quantum channels in the operator-sum representation [78]

$$\mathcal{N}(\rho) = \sum_k E_k \rho E_k^{\dagger}, \tag{21}$$

where $\{E_k\}$ are Kraus operators satisfying the completeness condition $\sum_k E_k^{\dagger} E_k = I$ and $\rho = |\psi\rangle\langle\psi|$ is the density matrix of a quantum state. Typical decoherence errors include the single qubit thermal relaxation error $\mathcal{N}_{\mathrm{TR}}(\cdot)$ describing the effects of energy dissipation (T_1 process) and dephasing

12
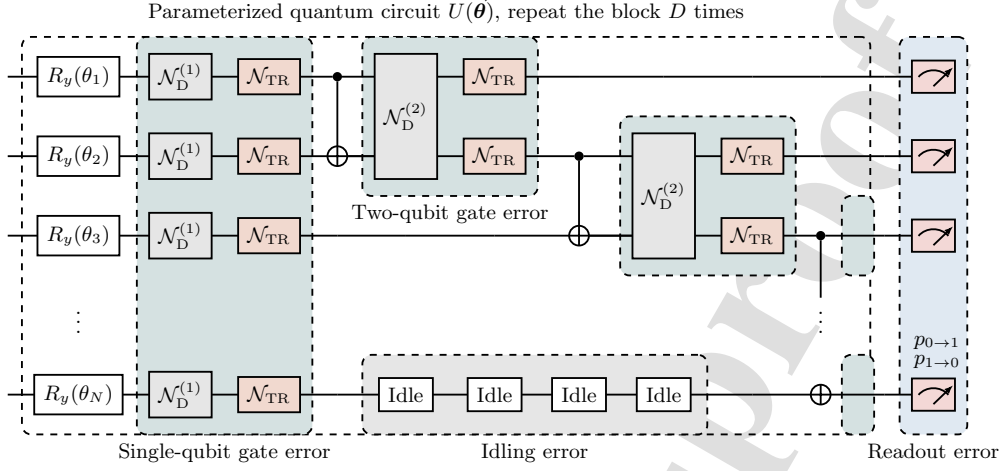
Figure 7: Modeling quantum decoherence noise on a generic circuit structure. This work does not model the idling error shown in the shaded gray area. Executing two-qubit gates adjacent to idling qubits could lead to crosstalk and performance degradation. One can adopt dynamical decoupling (DD) as an error suppression technique to tackle idling errors on hardware.

(associated with T_2 process) caused by interaction with the environment at low system temperature

$$\mathcal{N}_{\mathrm{TR}}(\rho) = (1 - p_Z - p_{\mathrm{reset}})\rho + p_Z Z\rho Z^{\dagger} + p_{\mathrm{reset}}\left((|0\rangle\langle 0|)\rho(|0\rangle\langle 0|)^{\dagger} + (|0\rangle\langle 1|)\rho(|0\rangle\langle 1|)^{\dagger}\right), \qquad (22)$$

where p_Z is the probability of dephasing and p_{reset} stands for the probability of relaxation by projecting to the $|0\rangle$ state. The other common decoherence error is the N-qubit depolarizing channel as a white noise by symmetrically implementing Pauli noises

$$\mathcal{N}_{\mathrm{D}}^{(N)}(\rho) = (1 - p_{\mathrm{D}})\rho + \frac{p_{\mathrm{D}}}{4^N - 1}\sum_{i,j,k,\cdots,n=0}^{3} \left(P_i \otimes P_j \otimes \cdots P_n\right)\rho\left(P_i \otimes P_j \otimes \cdots P_n\right)^{\dagger}, \qquad (23)$$

where p_{D} stands for the probability of such an error happening, and each P_{ℓ} is chosen from the Pauli group as $P_0 = I, P_1 = X, P_2 = Y, P_3 = Z$ such that there are $4^N - 1$ terms in the summation.

Given a generic circuit in fig. 7, our noise model consists of single- and two-qubit gate errors and single-qubit readout errors. The single-qubit gate errors $\mathcal{N}_{1\mathrm{q}}(\rho) = \mathcal{N}_{\mathrm{TR}} \circ \mathcal{N}_{\mathrm{D}}^{(1)}(\rho)$ are composed of a single-qubit depolarizing error channel and a single-qubit thermal relaxation error. Two-qubit gate errors $\mathcal{N}_{2\mathrm{q}}(\rho) = (\mathcal{N}_{\mathrm{TR}} \otimes \mathcal{N}_{\mathrm{TR}}) \circ \mathcal{N}_{\mathrm{D}}^{(2)}(\rho)$ consist of a two-qubit depolarizing error and a single-qubit thermal relaxation error on both qubits. Single-qubit readout error $\mathcal{N}_{\mathrm{readout}}$ flips the classical bit value based on experimentally determined probabilities, $\mathrm{Error}_{\mathrm{mea.}} = (p_{0\to 1} + p_{1\to 0})/2$. $p_{0\to 1}$ stands for the probability of preparing a series of $|0\rangle$ states on hardware but measured as $|1\rangle$. To determine the probability of errors in $\mathcal{N}_{\mathrm{TR}}$, we use the gate operation time t_{g} and hardware T_1/T_2 coherence time to estimate $p_{\mathrm{reset}} = 1 - e^{-t_{\mathrm{g}}/T_1}$ and $p_Z = (1 - p_{\mathrm{reset}})(1 - e^{-t_{\mathrm{g}}/T_2 + t_{\mathrm{g}}/T_1})/2$. These data are included in the backend properties through calibrations (see table 3) and used to build a fake backend such as Fake_Kolkata in fig. 6. The probability of the depolarizing error p_{D} is set such that the combined average gate infidelity matches the value from backend properties $\mathrm{Error}_{1\mathrm{q}}$ ($\mathrm{Error}_{2\mathrm{q}}$).
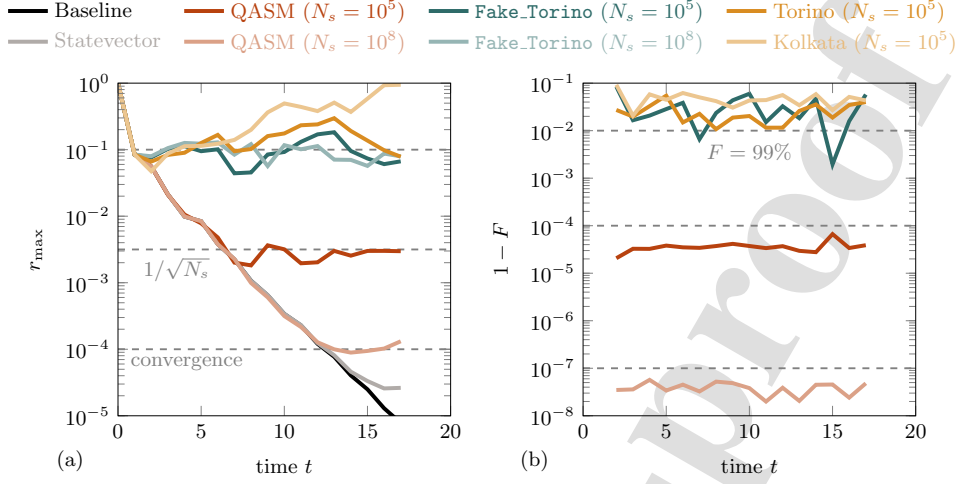
Figure 8: (a) Convergence history of the hybrid algorithm on simulators and hardware. (b) State infidelity as a quality measure of the hybrid solver in each time step.

4.3 Quantum hardware results

We study the proposed hybrid quantum–classical scheme on IBM's superconducting quantum hardware. Considering the noisy nature of quantum hardware, we relax the convergence criterion of the lid-driven cavity flow problem to $r_{\max} \leq 10^{-4}$, where

$$r_{\max} = \max_{\phi \in \{u,v\}} \left| \phi_{i,j}^{n+1} - \phi_{i,j}^{n} \right|. \tag{24}$$

This maximal residual is strictly larger than the root mean square difference of (7). We are interested in two questions: (i) What is the minimum tolerable noise to reach the steady-state solution $r_{\max} \leq 10^{-4}$ for the lid-driven cavity benchmark using the proposed hybrid scheme, and (ii) whether current NISQ hardware meets the requirement. To address (i), we run VQE on a statevector simulator to obtain optimal parameters $\vec{\theta}_{\mathrm{opt.}}$ and readout the state $|\psi_{\mathrm{noisy}}\rangle$ from real hardware or noisy simulators using HTree readout method. One could, in principle, conduct end-to-end hardware experiments by including VQE on-device training. However, this will introduce extra error sources as the hardware noise behavior drifts with time, and on-device training requires prohibitively long simulation times.

We present the results with a 4-qubit test case in fig. 8. The calibration data of IBM's quantum hardware is recorded in table 3. The statevector simulation and QASM simulation with a large shot number $N_s = 10^8$ meet the loose convergence criterion $r_{\max} \leq 10^{-4}$. The QASM simulation with $N_s = 10^5$ shots shows an accuracy limit near $1/\sqrt{N_s}$. This conclusion holds for fault-tolerant quantum linear system algorithms, including HHL. Without quantum noise, the solver still needs many shots to converge. As a reference value, 10^5 shots take about 43 seconds on the 133-qubit IBM Heron processor `ibm_torino` and 19 seconds on the 27-qubit Falcon processor `ibmq_kolkata`. The difference is caused by the measurement time $t_{\mathrm{mea.}}$ in two generations of IBM quantum processors as listed in table 3.

To address the second question (ii) raised at the beginning of this section, we test the hybrid

14

Backend	Qubits	T_1 (µs)	T_2 (µs)	Error$_{1q}$	Error$_{2q}$	Error$_{mea.}$	t_{2q} (ns)	$t_{mea.}$ (ns)
`ibmq_kolkata`	27	103	57	2.55×10^{-4}	8.93×10^{-3}	1.22×10^{-2}	452	640
`ibm_sherbrooke`	127	263	184	2.17×10^{-4}	7.61×10^{-3}	1.29×10^{-2}	533	1244
`ibm_torino`	133	168	132	3.08×10^{-4}	3.40×10^{-3}	1.89×10^{-2}	124	1560

Table 3: Calibration data of the IBM quantum computers used in this study. The data are averaged among all the qubits.

algorithm on a noisy simulator and hardware with a moderate shot number $N_s = 10^5$. The top two curves in fig. 8 (a) are real hardware results. `ibmq_kolkata` is an older processor with higher two-qubit gate error rates, so we observe an accumulation of error in the maximal residual r_{max}. While on the latest 133-qubit processor `ibm_torino`, we observe a lower error rate but still do not reach convergence. The noisy simulator results on `Fake_Torino` assembles the behavior of real hardware where the detailed noise modeling is described in section 4.2.1. The maximal residual consistently stays on the level $r_{max} \approx 10^{-1}$. In the presence of quantum noise, increasing the shot number from $N_s = 10^5$ to $N_s = 10^8$ does not help the hybrid solver converge. From the perspective of quantum state tomography, the HTree method achieves good fidelity $95\% \sim 99\%$ on real hardware as shown on fig. 8 (b). We use dynamical decoupling sequences and measurement error mitigation to suppress the effect of quantum noise. However, the QASM ($N_s = 10^8$) simulation result shows that a converging CFD solution requires at least $1 - F \leq 10^{-7}$. The above results indicate that the current NISQ hardware is still too noisy to implement explicit time-stepping numerical schemes for solving time-dependent PDE problems.

4.4 Resource estimation & complexity analysis

This section compares the quantum resources required for three quantum linear systems: VQE, VQLS, and HHL. We determine the most suitable one for NISQ hardware based on qubits required, 2-qubit gate counts, circuit depth, and estimated runtime on the 127-qubit IBM Eagle processor `ibm_sherbrooke`. Table 4 summarizes the results. We further compare the runtime per iteration to classical iterative linear solvers such as conjugate gradient (CG) and GMRES. This estimation assumes no parallelization for both classical and quantum solvers and does not consider any communication bottlenecks when submitting quantum jobs to IBM's cloud service. We manually transpile the quantum circuits using the highest optimization level in Qiskit.

As mentioned in section 2.3, VQLS would require more quantum resources than VQE on NISQ hardware. It requires one ancillary qubit to perform the Hadamard test but introduces a deeper circuit mainly due to the controlled application of the parametrized quantum circuit $U(\vec{\theta})$ which composes 54.3% of the estimated runtime on `ibm_sherbrooke` and 306 2-qubit gates after transpilation. This leads to each VQLS circuit runtime almost $50\times$ longer than the VQE circuit and beyond the hardware coherence time. Such an runtime overhead can be mitigated by adopting the Hadamard overlap test as proposed in the original VQLS paper [49], where we can significantly reduce circuit runtime at the cost of doubling the qubits used. The other problematic part is the scaling of number of VQLS circuits $N_{cir.} = \mathcal{O}(L_a^2)$ needs to evaluate in each iteration, where L_a is the number of LCU (or Pauli) terms given $\boldsymbol{A} = \sum_{l=1}^{L_a} c_l \boldsymbol{A}_l$. Notice the total VQLS runtime for each iteration $N_{cir.}$ can already be compatible to the total runtime of HHL algorithm. HHL is known to be a resource-intensive fault-tolerant algorithm which requires much more ancillary qubits, and most of the 2-qubit gate counts come from implementing the $\exp(\boldsymbol{A}t)$ time evolution operator. The results indicate VQLS is only suitable for NISQ hardware when the $m \times m$ matrix $\boldsymbol{A}$ has an efficient decomposition such as $L_a = \mathcal{O}(\text{poly}(\log_2 m))$. In the original VQLS paper [49], the authors consider the Ising-inspired linear system problems satisfying this condition and leading to an empirical

15

Solver	Regime	Qubits	2q gates	Circuit depth	$N_{\text{cir.}}$	$t_{\text{cir.}}$ (µs)	t_{total} (µs)
CG	Classical	/	/	/	/	/	22.2
GMRES	Classical	/	/	/	/	/	16.8
VQE	NISQ	4	15	59	136	**6.3**	**856.8**
VQLS	NISQ	5	504	2512	484	291.2	1.4×10^5
HHL	FT	12	3.7×10^5	1.7×10^6	1	2.4×10^5	2.4×10^5

Table 4: Resource estimation for solving a 16×16 linear system on the 127-qubits IBM Eagle processor ibm_sherbrooke. $t_{\text{cir.}}$ stands for the estimated quantum circuit runtime on this NISQ hardware. The coherence time is $T_2 \approx 184$ µs. The computation returns meaningful results when $t_{\text{cir.}} \leq T_2$. $N_{\text{cir.}}$ stands for the number of circuits to evaluate the cost in each iteration. Since HHL is not an iterative solver, $N_{\text{cir.}} = 1$. t_{total} represents the total runtime on quantum hardware for each iteration of VQE and VQLS. For quantum solvers, $t_{\text{total}} \approx N_{\text{cir.}} t_{\text{cir.}}$. For classical iterative solver, t_{total} is averaged among 100 iterations.

scaling of $\mathcal{O}\left((\log_2(m))^{8.5} \kappa \log(1/\epsilon)\right)$ through numerical simulations [79]. Compared to the time complexity of CG for positive definite matrices at each iteration $\mathcal{O}(ms\sqrt{\kappa} \log(1/\epsilon))$ and GMRES for general matrices $\mathcal{O}\left(m^2\right)$, VQLS can be advantageous for very large and sparse system m. However, it is unlikely the same scaling holds for general engineering problems through our observation. From table 4, the physical runtime required for VQLS and HHL both exceed the NISQ hardware limitation. Although obtaining better gate counts using more advanced transpilation methods for fault-tolerant algorithms like HHL is possible, the conclusions here still hold. We conclude that VQE is the most suitable one for NISQ hardware judging from estimated runtime.

Similar to VQLS, it is difficult to directly give the time complexity of VQE due to its heuristic nature. As discussed in section 2.3, one can estimate the total runtime of VQE as $t_{\text{VQE}} = N_{\text{itr.}} L_h t_{\text{circ.}}$, given the Hamiltonian in the form of Pauli decomposition $\boldsymbol{H} = \sum_{l=1}^{L_h} \alpha_l \boldsymbol{H}_l$. Since the construction of effective Hamiltonian involves non-trivial $|b\rangle$, $\boldsymbol{H}$ is usually a dense Hermitian matrix and leads to approximately $L_h = \mathcal{O}(m^2)$. Thus, VQE exhibits no hope to achieve any quantum advantage on solving linear systems over classical iterative solvers unless some novel preconditioner could effectively improve the sparsity of $\boldsymbol{H}$ such that $L_h = \mathcal{O}(\text{poly}(\log_2 m))$ while maintaining a low condition number κ. Eventually, we would expect running fault-tolerant quantum linear system algorithms with optimal time complexity $\mathcal{O}(\kappa \log(m/\epsilon))$ on matured hardware with quantum error correction.

5 Conclusion

We propose and test a hybrid quantum–classical algorithm for solving the incompressible Navier–Stokes equations on NISQ hardware. The algorithm combines the classical pressure projection method and quantum linear system algorithms. We use an efficient state reconstruction method to address the well-known quantum readout problem, largely unexplored in previous art. Preconditioning techniques can improve the trainability of variational quantum linear system algorithms. Although VQLS is widely considered a promising candidate for solving linear systems on NISQ hardware, our resource estimation indicates it would require non-affordable quantum resources unless the matrix $\boldsymbol{A}$ has an efficient Pauli or LCU decomposition. Our results indicate that current quantum hardware is too noisy to solve time-dependent PDE problems with explicit time-stepping. Besides quantum linear system algorithms, another promising approach for solving PDEs on a quantum computer is via Hamiltonian simulation [80] targeting to run on error-corrected quantum hardware and allowing deep quantum circuits. Compared to our hybrid NISQ algorithm, one clear advantage

of Hamiltonian simulation and other fault-tolerant quantum PDE algorithms [39, 40] is they do not require repeated encoding and readout during the computation and hence further control the accumulation of error caused by noisy measurement. With recent hardware demonstrations for linear PDEs [81, 82], Hamiltonian simulation could, eventually, provide a different approach for solving the Navier–Stokes equations.

Declaration of competing interests

The authors declare no conflicts of interest.

Data availability

Acknowledgements

References

[1] Y. Kim, A. Eddins, S. Anand, K. X. Wei, E. Van Den Berg, S. Rosenblatt, H. Nayfeh, Y. Wu, M. Zaletel, K. Temme, et al., Evidence for the utility of quantum computing before fault tolerance, Nature **618** (2023) 500–505.

[2] Google Quantum AI, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614** (2023) 676–681.

[3] S. Moses, C. Baldwin, M. Allman, R. Ancona, L. Ascarrunz, C. Barnes, J. Bartolotta, B. Bjork, P. Blanchard, M. Bohn, et al., A race track trapped-ion quantum processor, Physical Review X **13** (2023) 041052.

[4] D. Bluvstein, S. J. Evered, A. A. Geim, S. H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter, et al., Logical quantum processor based on reconfigurable atom arrays, Nature **626** (2024) 58–65.

[5] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, et al., Noisy intermediate-scale quantum algorithms, Reviews of Modern Physics **94** (2022) 015004.

[6] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Review **41** (1999) 303–332.

[7] L. K. Grover, A fast quantum mechanical algorithm for database search, in: Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, 1996, pp. 212–219.

[8] A. M. Dalzell, S. McArdle, M. Berta, P. Bienias, C.-F. Chen, A. Gilyén, C. T. Hann, M. J. Kastoryano, E. T. Khabiboulline, A. Kubica, et al., Quantum algorithms: A survey of applications and end-to-end complexities, arXiv preprint arXiv:2310.03011 (2023).

[9] A. W. Harrow, A. Hassidim, S. Lloyd, Quantum algorithm for linear systems of equations, Physical Review Letters **103** (2009) 150502.

[10] M. R. Hestenes, E. Stiefel, Methods of conjugate gradients for solving linear systems, Journal of Research of the National Bureau of Standards **49** (1952).

[11] A. M. Childs, R. Kothari, R. D. Somma, Quantum algorithm for systems of linear equations with exponentially improved dependence on precision, SIAM Journal on Computing **46** (2017) 1920–1950.

[12] L. Wossnig, Z. Zhao, A. Prakash, Quantum linear system algorithm for dense matrices, Physical Review Letters **120** (2018) 050502.

[13] Y. Subaşı, R. D. Somma, D. Orsucci, Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing, Physical Review Letters **122** (2019) 060504.

[14] D. An, L. Lin, Quantum linear system solver based on time-optimal adiabatic quantum computing and quantum approximate optimization algorithm, ACM Transactions on Quantum Computing **3** (2022) 1–28.

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

[16] A. Montanaro, S. Pallister, Quantum algorithms and the finite element method, Physical Review A **93** (2016) 032324.

[17] D. W. Berry, A. M. Childs, A. Ostrander, G. Wang, Quantum algorithm for linear differential equations with exponentially improved dependence on precision, Communications in Mathematical Physics **356** (2017) 1057–1081.

[18] S. Wang, Z. Wang, W. Li, L. Fan, Z. Wei, Y. Gu, Quantum fast Poisson solver: The algorithm and complete and modular circuit design, Quantum Information Processing **19** (2020) 1–25.

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

[20] N. Linden, A. Montanaro, C. Shao, Quantum vs. classical algorithms for solving the heat equation, Communications in Mathematical Physics **395** (2022) 601–641.

[21] J.-P. Liu, H. Ø. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, A. M. Childs, Efficient quantum algorithm for dissipative nonlinear differential equations, Proceedings of the National Academy of Sciences **118** (2021) e2026805118.

18

[22] S. Jin, N. Liu, Quantum algorithms for computing observables of nonlinear partial differential equations, arXiv preprint arXiv:2202.07834 (2022).

[23] L. Lapworth, A hybrid quantum-classical CFD methodology with benchmark HHL solutions, arXiv preprint arXiv:2206.00419 (2022).

[24] X. Li, X. Yin, N. Wiebe, J. Chun, G. K. Schenter, M. S. Cheung, J. Mülmenstädt, Potential quantum advantage for simulation of fluid dynamics, arXiv preprint arXiv:2303.16550 (2023).

[25] S. Succi, W. Itani, K. Sreenivasan, R. Steijl, Quantum computing for fluids: Where do we stand?, Europhysics Letters **144** (2023) 10001.

[26] B. Ljubomir, Quantum algorithm for the Navier–Stokes equations by using the streamfunction-vorticity formulation and the lattice Boltzmann method, International Journal of Quantum Information **20** (2022) 2150039.

[27] L. Budinski, Quantum algorithm for the advection-diffusion equation simulated with the lattice Boltzmann method, Quantum Information Processing **20** (2021) 57.

[28] J. Yepez, Type-II quantum computers, International Journal of Modern Physics C **12** (2001) 1273–1284.

[29] J. Yepez, Quantum lattice-gas model for the Burgers equation, Journal of Statistical Physics **107** (2002) 203–224.

[30] M. M. Micci, J. Yepez, Measurement-based quantum lattice gas model of fluid dynamics in 2+1 dimensions, Physical Review E **92** (2015) 033302.

[31] S. Kocherla, Z. Song, F. E. Chrit, B. Gard, E. F. Dumitrescu, A. Alexeev, S. H. Bryngelson, Fully quantum algorithm for lattice Boltzmann methods with application to partial differential equations, arXiv preprint arXiv:2305.07148 (2023).

[32] S. Kocherla, A. Adams, Z. Song, A. Alexeev, S. H. Bryngelson, A two-circuit approach to reducing quantum resources for the quantum lattice Boltzmann method, arXiv preprint arXiv:2401.12248 (2024).

[33] B. N. Todorova, R. Steijl, Quantum algorithm for the collisionless Boltzmann equation, Journal of Computational Physics **409** (2020) 109347.

[34] W. Itani, K. R. Sreenivasan, S. Succi, Quantum algorithm for lattice Boltzmann (QALB) simulation of incompressible fluids with a nonlinear collision term, Physics of Fluids **36** (2024).

[35] W. Itani, S. Succi, Analysis of Carleman linearization of lattice Boltzmann, Fluids **7** (2022) 24.

[36] C. Sanavio, S. Succi, Lattice Boltzmann–Carleman quantum algorithm and circuit for fluid flows at moderate Reynolds number, AVS Quantum Science **6** (2024).

[37] C. Sanavio, R. Scatamacchia, C. de Falco, S. Succi, Three Carleman routes to the quantum simulation of classical fluids, Physics of Fluids **36** (2024).

[38] R. Steijl, Quantum algorithms for nonlinear equations in fluid mechanics, Quantum Computing and Communications (2020).

[39] F. Gaitan, Finding flows of a Navier–Stokes fluid through quantum computing, npj Quantum Information **6** (2020) 61.

[40] F. Gaitan, Circuit implementation of oracles used in a quantum algorithm for solving nonlinear partial differential equations, Physical Review A **109** (2024) 032604.

[41] F. Oz, R. K. Vuppala, K. Kara, F. Gaitan, Solving burgers' equation with quantum computing, Quantum Information Processing **21** (2022) 30.

[42] B. Basu, S. Gurusamy, F. Gaitan, A quantum algorithm for computing dispersal of submarine volcanic tephra, Physics of Fluids **36** (2024).

[43] S. Pantankar, D. Spalding, A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows, International Journal of Heat and Mass Transfer **15** (1972) 1787–1806.

[44] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2** (2018) 79.

[45] M. Cerezo, A. Arrasmith, R. Babbush, S. C. Benjamin, S. Endo, K. Fujii, J. R. McClean, K. Mitarai, X. Yuan, L. Cincio, et al., Variational quantum algorithms, Nature Reviews Physics **3** (2021) 625–644.

[46] R. Steijl, G. N. Barakos, Parallel evaluation of quantum algorithms for computational fluid dynamics, Computers & Fluids **173** (2018) 22–28.

[47] K. Griffin, S. Jain, T. Flint, W. Chan, Investigations of quantum algorithms for direct numerical simulation of the Navier–Stokes equations, Center for Turbulence Research Annual Research Briefs **347** (2019).

[48] S. S. Bharadwaj, K. R. Sreenivasan, Hybrid quantum algorithms for flow problems, Proceedings of the National Academy of Sciences **120** (2023) e2311014120.

[49] C. Bravo-Prieto, R. LaRose, M. Cerezo, Y. Subasi, L. Cincio, P. J. Coles, Variational quantum linear solver, Quantum **7** (2023) 1188.

[50] R. Demirdjian, D. Gunlycke, C. A. Reynolds, J. D. Doyle, S. Tafur, Variational quantum solutions to the advection–diffusion equation for applications in fluid dynamics, Quantum Information Processing **21** (2022) 322.

[51] M. Ali, M. Kabel, Performance study of variational quantum algorithms for solving the Poisson equation on a quantum computer, Physical Review Applied **20** (2023) 014054.

[52] Y. Liu, Z. Chen, C. Shu, P. Rebentrost, Y. Liu, S. Chew, B. Khoo, Y. Cui, A variational quantum algorithm-based numerical method for solving potential and Stokes flows, Ocean Engineering **292** (2024) 116494.

[53] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, D. Jaksch, Variational quantum algorithms for nonlinear problems, Physical Review A **101** (2020) 010301.

[54] D. Jaksch, P. Givi, A. J. Daley, T. Rung, Variational quantum algorithms for computational fluid dynamics, AIAA Journal **61** (2023) 1885–1894.

[55] S. Aaronson, Read the fine print, Nature Physics **11** (2015) 291–293.

[56] N. Renaud, HTree, https://github.com/QuantumApplicationLab/qiskit_tomography_toolbox, 2024. Accessed: 2024-03-09.

[57] J. A. Smolin, J. M. Gambetta, G. Smith, Efficient method for computing the maximum-likelihood quantum state from measurements with additive Gaussian noise, Physical Review Letters **108** (2012) 070502.

[58] A. J. Pool, A. D. Somoza, C. Mc Keever, M. Lubasch, B. Horstmann, Nonlinear dynamics as a ground-state solution on quantum computers, Physical Review Research **6** (2024) 033257.

[59] S. S. Bharadwaj, K. R. Sreenivasan, Compact quantum algorithms that can potentially maintain quantum advantage for solving time-dependent differential equations, arXiv preprint arXiv:2405.09767 (2024).

[60] A. J. Chorin, Numerical solution of the Navier–Stokes equations, Mathematics of Computation **22** (1968) 745–762.

[61] D. Kwak, C. Kiris, C. S. Kim, Computational challenges of viscous incompressible flows, Computers & Fluids **34** (2005) 283–299.

[62] A. J. Chorin, A numerical method for solving incompressible viscous flow problems, Journal of Computational Physics **135** (1997) 118–125.

[63] C. M. Rhie, W.-L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation, AIAA journal **21** (1983) 1525–1532.

[64] J. Tilly, H. Chen, S. Cao, D. Picozzi, K. Setia, Y. Li, E. Grant, L. Wossnig, I. Rungger, G. H. Booth, et al., The variational quantum eigensolver: A review of methods and best practices, Physics Reports **986** (2022) 1–128.

[65] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, J. L. O'Brien, A variational eigenvalue solver on a photonic quantum processor, Nature Communications **5** (2014) 4213.

[66] H.-L. Liu, Y.-S. Wu, L.-C. Wan, S.-J. Pan, S.-J. Qin, F. Gao, Q.-Y. Wen, Variational quantum algorithm for the Poisson equation, Physical Review A **104** (2021) 022418.

[67] Y. Sato, R. Kondo, S. Koide, H. Takamatsu, N. Imoto, Variational quantum algorithm based on the minimum potential energy for solving the Poisson equation, Physical Review A **104** (2021) 052409.

[68] P. C. Costa, D. An, Y. R. Sanders, Y. Su, R. Babbush, D. W. Berry, Optimal scaling quantum linear-systems solver via discrete adiabatic theorem, PRX quantum **3** (2022) 040303.

[69] B. D. Clader, B. C. Jacobs, C. R. Sprouse, Preconditioned quantum linear system algorithm, Physical Review Letters **110** (2013) 250504.

[70] A. Hosaka, K. Yanagisawa, S. Koshikawa, I. Kudo, X. Alifu, T. Yoshida, Preconditioning for a variational quantum linear solver, arXiv preprint arXiv:2312.15657 (2023).

[71] C. M. Keller, S. Eidenbenz, A. Bärtschi, D. O'Malley, J. Golden, S. Misra, Hierarchical multigrid ansatz for variational quantum algorithms, arXiv preprint arXiv:2312.15048 (2023).

[72] M. S. Rudolph, S. Sim, A. Raza, M. Stechly, J. R. McClean, E. R. Anschuetz, L. Serrano, A. Perdomo-Ortiz, ORQVIZ: Visualizing high-dimensional landscapes in variational quantum algorithms, arXiv preprint arXiv:2111.04695 (2021).

[73] N. Kanazawa, D. J. Egger, Y. Ben-Haim, H. Zhang, W. E. Shanks, G. Aleksandrowicz, C. J. Wood, Qiskit experiments: A Python package to characterize and calibrate quantum computers, Journal of Open Source Software **8** (2023) 5329.

[74] U. Ghia, K. N. Ghia, C. Shin, High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, Journal of Computational Physics **48** (1982) 387–411.

[75] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549** (2017) 242–246.

[76] Qiskit contributors, Qiskit: An open-source framework for quantum computing, 2023.

[77] R. H. Byrd, P. Lu, J. Nocedal, C. Zhu, A limited memory algorithm for bound constrained optimization, SIAM Journal on Scientific Computing **16** (1995) 1190–1208.

[78] M. A. Nielsen, I. L. Chuang, Quantum computation and quantum information, volume 2, Cambridge university press Cambridge, 2001.

[79] H. Patil, Y. Wang, P. S. Krstić, Variational quantum linear solver with a dynamic ansatz, Physical Review A **105** (2022) 012423.

[80] S. Lloyd, Universal quantum simulators, Science **273** (1996) 1073–1078.

[81] Y. Sato, R. Kondo, I. Hamamura, T. Onodera, N. Yamamoto, Hamiltonian simulation for time-evolving partial differential equation by scalable quantum circuits, arXiv preprint arXiv:2402.18398 (2024).

[82] L. Wright, C. M. Keever, J. T. First, R. Johnston, J. Tillay, S. Chaney, M. Rosenkranz, M. Lubasch, Noisy intermediate-scale quantum simulation of the one-dimensional wave equation, arXiv preprint arXiv:2402.19247 (2024).

[83] S. Sim, P. D. Johnson, A. Aspuru-Guzik, Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum–classical algorithms, Advanced Quantum Technologies **2** (2019) 1900070.

[84] K. Nakaji, N. Yamamoto, Expressibility of the alternating layered ansatz for quantum computation, Quantum **5** (2021) 434.

[85] M. Cerezo, A. Sone, T. Volkoff, L. Cincio, P. J. Coles, Cost function dependent barren plateaus in shallow parametrized quantum circuits, Nature Communications **12** (2021) 1791.

[86] J. Y. Araz, M. Spannowsky, Classical versus quantum: Comparing tensor-network-based quantum circuits on large hadron collider data, Physical Review A **106** (2022) 062423.

[87] G. Li, Y. Ding, Y. Xie, Tackling the qubit mapping problem for NISQ-era quantum devices, in: Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, 2019, pp. 1001–1014.

[88] S. Kullback, R. A. Leibler, On information and sufficiency, The Annals of Mathematical Statistics **22** (1951) 79–86.

[89] D. A. Meyer, N. R. Wallach, Global entanglement in multiparticle systems, Journal of Mathematical Physics **43** (2002) 4273–4278.

[90] V. Tripathi, H. Chen, M. Khezri, K.-W. Yip, E. Levenson-Falk, D. A. Lidar, Suppression of crosstalk in superconducting qubits using dynamical decoupling, Physical Review Applied **18** (2022) 024068.

[91] P. Das, S. Tannu, S. Dangwal, M. Qureshi, ADAPT: Mitigating idling errors in qubits via adaptive dynamical decoupling, in: MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture, 2021, pp. 950–962.

[92] S. Niu, A. Todri-Sanial, Effects of dynamical decoupling and pulse-level optimizations on ibm quantum computers, IEEE Transactions on Quantum Engineering **3** (2022) 1–10.

[93] N. Ezzell, B. Pokharel, L. Tewala, G. Quiroz, D. A. Lidar, Dynamical decoupling for superconducting qubits: A performance survey, Physical Review Applied **20** (2023) 064027.

[94] H. R. Grimsley, S. E. Economou, E. Barnes, N. J. Mayhall, An adaptive variational algorithm for exact molecular simulations on a quantum computer, Nature Communications **10** (2019) 3007.

[95] H. Wang, Y. Ding, J. Gu, Y. Lin, D. Z. Pan, F. T. Chong, S. Han, QuantumNAS: Noise-adaptive search for robust quantum circuits, in: 2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA), IEEE, 2022, pp. 692–708.

[96] Z. Liang, J. Cheng, H. Ren, H. Wang, F. Hua, Z. Song, Y. Ding, F. T. Chong, S. Han, X. Qian, et al., NAPA: Intermediate-level variational native-pulse ansatz for variational quantum algorithms, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2024).

Appendix A Quantum circuit and hardware co-design

We use different parameterized quantum circuit (PQC) designs for $U(\vec{\boldsymbol{\theta}})$ to prepare the ground-state solution of the pressure Poisson equation on quantum hardware. We consider four design candidates (illustrated in fig. A.9) widely used in the literature of variational quantum algorithms [75, 83–86] and evaluate them in terms of simulation performance and hardware resource cost.

The Hardware-Efficient Ansatz (HEA) [75] was originally proposed for quantum hardware with nearest-neighbor connectivity (such as that of IBM's and Google's superconducting quantum processors) where 2-qubit gates are only applied on adjacent qubits to avoid additional SWAP gates introduced in transpilation [87]. The Real-Amplitude Ansatz (RAA) is a variant of HEA where the general single qubit rotation layer is limited to $R_y(\theta)$ gate and only generates real-valued states. Alternating Layered Ansatz (ALT) is composed of blocks forming local 2-designs and proven to be trainable for circuit depth $D \in \mathcal{O}(\log(N))$ when combined with a local cost function $C_l(\vec{\boldsymbol{\theta}})$ [85]. Finally, the Tensor Network Ansatz (TEN) is a family of circuits inspired by classical Tensor Network architectures such as Matrix Product States (MPS), Tree Tensor Networks (TTN) and Multi-scale Entanglement Renormalisation Ansatz (MERA)[86]. For simplicity, we consider a tensor product ansatz here to probe if entanglement is necessary to find the ground state.
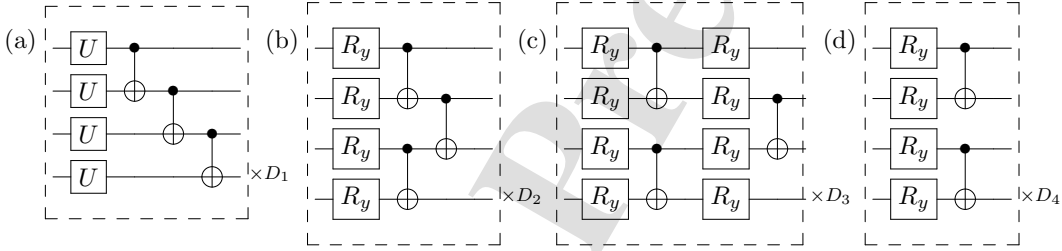
Figure A.9: The ansatze considered in this study. After repeating D_j times of each block, a final rotation layer enhances expressibility, though it is not illustrated further here. (a) HEA uses $U = R_z(\theta_j)R_y(\phi_j)R_z(\varphi_j)$ as a general rotation on the Bloch sphere along with non-parameterized adjacent CNOT gates providing entanglement to prepare the solution state; RAA uses $U = R_y(\theta_j)$ in the rotation layer to reduce the trainable parameters. (b) RAA with a pairwise entanglement layer to reduce the circuit duration of one CNOT gate per block. (c) Alternating Layered Ansatz (ALT). (d) Tensor Network Ansatz (TEN) prepares tensor product states and is expected to have a weaker entangling capability.

Ansatze	Avg. $\bar{F}$	Expr. ↓	Ent. ↑	N_{2q}	$D_{\text{trans.}}$ (DD)	$t_{\text{circ.}}$ (µs)	Coherence
HEA	99.99%	**0.0003**	**0.8149**	15	47 (68)	5.76	10.08%
RAA	99.99%	0.1899	0.7428	15	35 (56)	5.76	10.08%
RAA-p	99.27%	0.1895	0.7469	15	34 (54)	5.40	**9.45**%
ALT	83.82%	0.2349	0.5843	8	29 (34)	2.88	5.05%
TEN	53.04%	0.6350	0.3314	10	29 (30)	2.72	4.85%

Table A.5: Summary of a 4-qubit circuit design case study where each PQC candidate contains 24 trainable parameters except for HEA $N_{\text{param}} = 72$. The expressibility and entanglement capability are sampled from 10^3 random samples. The average entanglement for the true ground state $|\psi_g\rangle$ across different simulation times is 0.46. The transpiled circuit depth with dynamical decoupling (DD) sequence is included. Digital DD sequence generally increases transpiled circuit depth but not actual hardware runtime $t_{\text{circ.}}$. Coherence is calculated as $t_{\text{circ.}}/T_2$, where the average qubit lift-time $T_2 \approx 57.15$ µs for `ibmq_kolkata`.

24
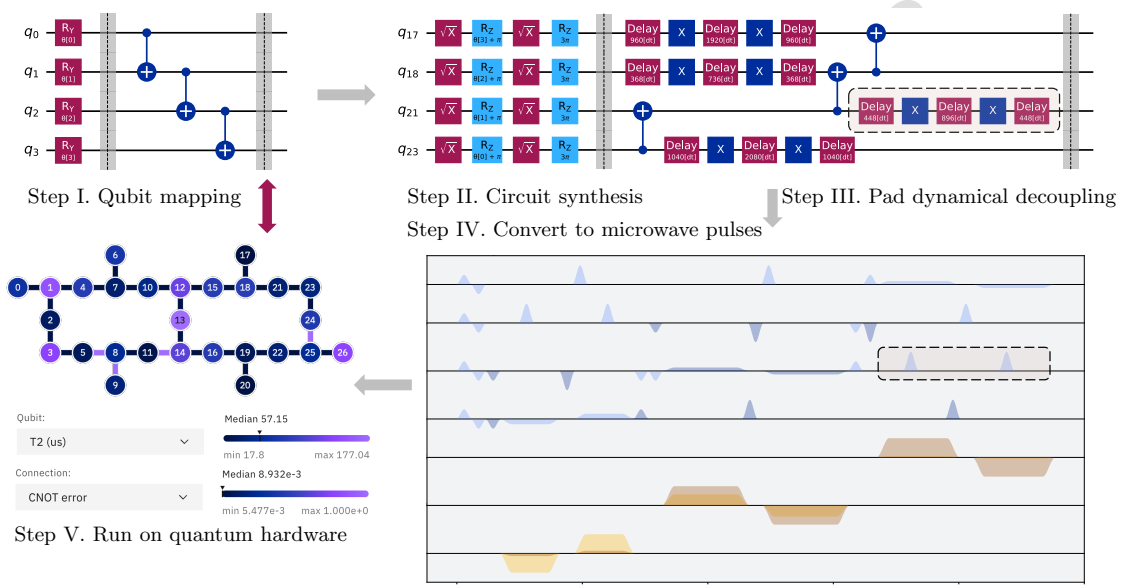
Figure A.10: Visualization of the transpilation process of 4-qubit RAA ($D_1 = 1$) on 27-qubit `ibmq_kolkata`.

To evaluate each circuit candidate performance, we randomly sample $K = 10$ pre-conditioned Hamiltonian $\{\widetilde{\boldsymbol{H}}(t)\}_{i=1}^K$ at different times of the simulation with $\Delta t = 1$. We then randomly initiate the parameters of VQE and solve for ground-state preparation with the L-BFGS-B optimizer [77] on the Qiskit statevector simulator.

We calculate the expressibility and the entanglement capability [83] as auxiliary measures for each ansatz candidate. The ansatz expressibility describes its capability to sample uniformly from the Hilbert space and approximate arbitrary states. It can be defined as

$$\text{Expr} = \mathcal{D}_{\text{KL}} \left(\widehat{P}_{\text{PQC}}(F_{\vec{\boldsymbol{\theta}}}) \| P_{\text{Haar}} \left(F \right) \right), \tag{A.1}$$

where the Kullback–Leibler (KL) divergence [88] measures the distance between $\widehat{P}_{\text{PQC}}(F_{\vec{\boldsymbol{\theta}}})$ distribution of state fidelities between two randomly sampled parametrized states, and $P_{\text{Haar}}(F)$ is the state fidelity distribution for the ensemble of Haar random states. Expr closer to zero indicates more expressivity for an ansatz. Following the original definition [83], we calculate the entanglement capability as the average Meyer–Wallach Q-measure [89] from an ensemble of randomly sampled states

$$\text{Ent} = \frac{1}{|S|} \sum_{\theta_j \in S} Q\big(|\psi(\theta_j)\rangle\big) = \frac{2}{|S|} \sum_{\theta_j \in S} \left(1 - \frac{1}{n} \sum_{k=1}^{n} \text{Tr} \left[\rho_k^2(\theta_j) \right] \right), \tag{A.2}$$

where ρ_k is the reduced density matrix of the k-th qubit and $|S|$ represents the size of samples. Notice $Q = 0$ for any product state and $Q = 1$ for the GHZ state $|\psi\rangle_{\text{GHZ}} = (|0\rangle^{\otimes n} + |1\rangle^{\otimes n})/\sqrt{2}$. We further record the number of 2-qubit gates involved $N_{2\text{q}}$, transpiled circuit depth $D_{\text{trans.}}$ and hardware runtime $t_{\text{circ.}}$ on 27-qubit `ibmq_kolkata`. The circuits are transpiled using the highest optimization level and adopt XX dynamical decoupling sequence [90–93] to suppress decoherence and crosstalk during the idling time of the qubits, as illustrated in fig. A.10.

25

We report a $N = 4$ case study in table A.5. The random seeds for parameter generation, simulator, and optimizer are fixed to ensure a fair comparison of averaged state fidelity $\bar{F}$. One can conclude that the real-amplitude ansatz (RAA) outperforms all the other candidates in terms of fidelity and resource burden. In this case, switching from the linear entanglement structure via RAA pairwise only reduces 0.6% of the coherence time. The difference can be as high as 5% (≈ 2.8 µs) at $N = 10$ with $D_2 = 11$. HEA has higher expressibility and entanglement capability but achieves the same average fidelity with 3 times more parameters. The results from TEN emphasize that quantum entanglement is a necessary resource in general state preparation. To further enhance performance, one can consider strategies such as ADAPT-VQE [94] to design problem-tailored ansatze, noise-adaptive circuit design [95] to cope with hardware noise, and pulse-level ansatze [96] to suppress decoherence due to much shorter duration on hardware.

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: