Hybrid Quantum Algorithms for flow problems

February 1, 2025 11:55 AM

QC is very resourceful when it comes to fluid dynamics.

-> We use a QC simulator called QFlowS to demonstrate this

Significance

-> Quantum Computing (QC) has the advantages of speed & storage over classical computing, but is based on a linear paradigm.

-> Most problems are non-linear, thus, a viable QC algorithm must include a suitable preparation to convert a nonlinear problem, into a linear one. & Once done mechanically via quantum computation, is spat out in classical terms

- because non-linear problems must be converted to linear problems, this offsets the quantum advantage

-> Thus, when solving Poiseuille & Couette flows we introduce an in-house quantum simulator, dubbed Quantum Flow Simulator (QFlowS) -> which is used for computational fluid dynamics

↳ This Method in turn both highlights the limitations of QC whilst preserving quantum advantage

When solving a non-linear flow by QC, we usually solve an equivalent ∞ dimensional linear system via linear embedding

-> We do this by simulating two well-known flows using QflowS

-> Through Simulation, it shows us previously unseen, full-gate lvl implementation of a hybrid-high precision Quantum Linear Systems Algorithm(s) (QLSA(s)) for simulating such 2-flows as low as Reynolds numbers.

The utility of the simulation comes from its error estimates & power law scaling that relates T_0, (a parameter crucial to Hamiltonian simulations), to the condition # K of the simulation matrix.

-> This allows for the prediction of an optimal scaling parameter for accurate eigenvalue estimation.

To further speed up the process, while preserving accuracy, we include:

a) the functional form; (the sparse quantum state preparation) &

b) in situ* quantum postprocessing tool for computing nonlinear fctns of the velocity field

means: in its original position or place; Latin for "in place" or "on sight"

Simulating non-linear physical systems like turbulent flows & climate physics is extremely demanding, exceeding current supercomputing capabilities. Direct Numerical Simulations (DNS) provide detailed insights, but are limited by computational power. Thus, advancing fundamental theories & large-scale simulations requires a major leap in computing technology.

One such potential candidate is Quantum Computing (QC) as it offers potential speedups over classical methods, but remains in its early "Noisy Intermediate Scale Quantum" (NISQ) era. While QC has been applied in fields like finance & cryptography, its use in solving non linear PDEs, such as those in fluid dynamics, is limited. This work explores Quantum Computation of Fluid Dynamics (QCFD), but challenges arise due to quantum mechanics' inherent linearity, requiring approximations that introduce truncation errors, restricting applications to weakly non-linear problems.

Thus, the ability to solve high dimensional linear systems in an end-to-end manner (meaning an algorithm that efficiently prepares a quantum state, processes it, & outputs a result by measurement while retaining all or some net quantum advantage) whilst capturing the flow physics is crucial to simulating non-linear flow problems.* Our goal here is to present the steps in solving simple & idealized problems, inclusive of providing estimates of scaling & errors involved.

To achieve our previously stated goal,* we use a high-performance quantum simulator called QFlowS (Quantum Flow Simulator), designed specifically to simulate fluid flows. Built on a C++ platform, it offers both QC & CFD tools in one place. With QFlowS, we implement a modified version of the class of algorithms, now termed as an equivalent quantum linear systems algorithm(s) (QLSA). Which under certain caveats, these algorithms solve a linear system of eqns given by the matrix inversion problem $A\vec{x}=\vec{b}$, w̄ an up to exponential speedup in comparison to known classical algorithms.

In recent years, efforts using various quantum & quantum-inspired methods have aimed to solve linear & nonlinear PDEs. However, most remain theoretical, lacking quantum simulations, flow field analysis, & accurate error estimates.

In particular, a full-gate level quantum simulation is implemented on QFlowS to solve the unsteady Poiseulle & Couette flow problems. It employs both the Harrow-Hassidim-Lyod (HHL) algorithm & its LCU-based (linear combination of unitaries) variant while introducing new quantum state preparation & postprocessing (QPP) protocols to compute nonlinear flow properties, such as viscous dissipation. This end-to-end approach helps mitigate limitations in quantum fluid simulations, which is more feasible for small systems on near-term NISQ machines, though still far from realistic large scale applications.

Linear Flow Problems

Consider the well known 1D unsteady Poiseuille & Couette flows, which model microchannel & lubricant flows around bearings. The proposed framework extends to linear advection-diffusion (w̄ constant advection velocity), & generally to elliptic & parabolic PDEs, including Laplace, Poisson & heat eqns. Under certain boundary conditions, these flows have exact analytical solns, making them ideal for testing quantum solvers.

Previous studies have explored similar quantum implementations on QC & estimated theoretical upper bounds of their theoretical complexities. The general form of the PDEs considered are given by the momentum & mass conservation relations:

$$\frac{\partial\vec{u}}{\partial t}+C\cdot\nabla\vec{u}=\frac{1}{Re}\nabla^2\vec{u}-\nabla\vec{p} \quad (1)$$

$$\nabla\cdot\vec{u}=0 \quad (2)$$

Where the vector, $\vec{u}$, is a 3-dimensional vector, $\vec{u}=(u,v,w)$ representing the velocity field, C is the constant advection velocity (which is zero in the fully developed state of flows), $\vec{p}$ is the pressure field, $Re=UD/\nu$ is the Reynolds number, U is the characteristic velocity, ν is the kinematic viscosity, & D is the separation btwn the boundaries.

However, the fully developed 1D reduces to:

$$\frac{\partial u}{\partial t}=\frac{1}{Re}\cdot\frac{\partial^2 u}{\partial y}-\frac{\partial P}{\partial x} \quad (3)$$

*No-Slip Conditions

-> a boundary condition that states "a fluid layer in contact w̄ a solid boundary must have the same velocity as (that) the solid boundary."

where velocity only varies only along y (also known as the wall-normal direction), & the pressure gradient $\frac{\partial P}{\partial x}$ is a constant. The boundary conditions are no-slip* w̄ $u(0,t)=u(D,t)=0$ for the Poiseuille flow & $u(0,t)=0$ & $u(D,t)=1$ for the Couette flow. The initial condition for the temporal evolution is set to be a uniform flow $u(y,0)=u_{in}=1$. While simple for classical CFD, this problem serves as a crucial starting pt to demonstrate the feasibility of quantum algorithms for computational fluid dynamics.

Hybrid Quantum-Classical Numerical Setup

To solve Eq. 3 by QLSA, it must first be recast as a linear system of eqns. This can be done through finite difference discretization of space & time, employing a second-order central difference scheme for the Laplacian & forward & backward Euler (FE & BE) schemes to discretize time. This results in three possible matrix equations:

-> Iterative Backward Euler (BE): $A_{be1}\vec{u}=b_{be1}$ (solved iteratively @ each time step until convergence).

-> One-shot Backward Euler (BE): $A_{be2}\vec{u}=b_{be2}$ (solving for all time steps @ once).

-> One-shot Forward Euler (FE): $A_{fe}\vec{u}=b_{fe}$ (solving for all time steps @ once).

A hybrid quantum-classical approach is developed through the following, where matrix preconditioning & computation of key parameters (such as rotation angles for quantum state preparation & Hamiltonian simulation time) are handled classically. The QC then loads the prepared input states & solves the system using QLSA.

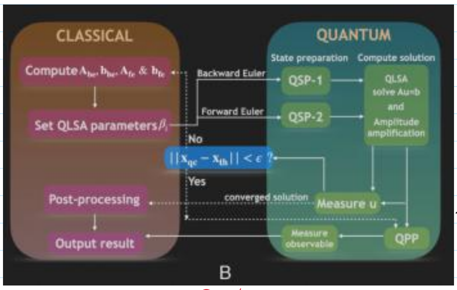
- One-shot Forward Euler (FE): ... solving total time steps at once).

A hybrid quantum-classical approach is developed through the following, where matrix preconditioning & computation of key parameters (such as rotation angles for quantum state preparation & Hamiltonian simulation time) are handled classically. The QC then loads the prepared input states & solves the system using QLSA.

- For IBE, the system is solved iteratively @ each time step w/ classical verification until residual error $\leq 10^{-6}$.
- For one-shot BE & FE, the solution is computed for all time steps simultaneously.

Challenges & Quantum Limitations

Although one-shot BE can be adapted for quantum advantage, state preparation & measurement remain major bottlenecks. Measuring the quantum state & re-preparing it for the next time step incurs an $O(N_q)$ cost, which nullifies the quantum speedup, making it no better than classical solvers (not including additional quantum errors).



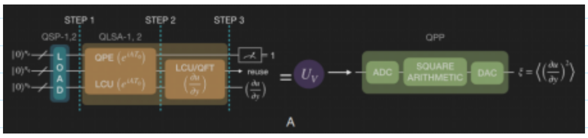


Fig A shows the modified QLSA circuit w/ QSP, forming U_V that is employed recursively in the QPP protocol for computing viscous dissipation by combination of quantum analog-digital converters (QADCs).
Fig B. Shows the working flowchart of the hybrid quantum-classical algorithm.

Solution Extraction & Postprocessing

The computed solution can be:

① Classically post-processed after quantum measurement, allowing validation & circuit refinement

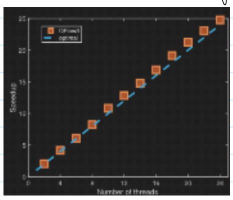
② Post-processed in situ using the Quantum Postprocessing (QPP) protocol, which directly extracts key observables, such as non-linear fcts of the velocity field, while minimizing quantum state measurements.

By computing nonlinear fcts directly on the quantum device, QPP reduces measurement noise & preserves quantum advantage as much as possible, making this approach a plausible alternative to classical solvers for small systems.

QFlowS

Quantum solvers Qiskit (IBM), Quipper & QuEST, are optimized for running quantum simulations, making it hard to input subroutines & custom data structures for CFD calculations. For the classical approach, we have ANSYS & OpenFOAM to preform CFD calculations. However, since we are more interested in having a single software for CFD, while preserving quantum advantage, We use a software called QFlowS. QFlowS is a high-performance quantum toolkit based on a C++ Core & designed to be used both independently or as part of a software package.

- QFlowS has the current capability of 30+ qubit simulations w/ custom quantum circuits
- It also has several built-in gates & quantum circuits which can be used while also being able to probe different quantum state metrics (such as the norm, density matrix & entanglement).
- Most importantly, QFlow includes the CFD tools needed to set up flow problems, which makes it versatile for QCFD simulations.



QFlowS is currently in the process of being optimized for performance on supercomputers. Not only is QFlowS is not only near-optimal, but superoptimal in terms of scaling w/ increasing # of threads up to 24. Superoptimality arises when the quantum circuit is sparse, which in turn causes lesser quantum entanglement. All single circuit layer operations are distributed over many worker threads, whose cache size exceeds the size of the quantum state subspace being handled, thus making the optimal & QFlow graphs almost parralell.

QLSA Quantum Linear Solver Algorithm (QLSA)

The QLSA is a fundamental quantum approach for solving linear systems of eqns of the form $A\vec{x} = \vec{b}$, w/ the Harrow-Hassidim-Lloyd (HLL) algorithm being one of its first implementations. The work here explores two variations of QLSA & their improvements, focusing on applications in Quantum Computation of Fluid Dynamics (QCFD).

① QLSA-1: The HHL Algorithm

-> The HHL algorithm is the first quantum protocol for solving linear eqns & is the basis for what is referred to here as QLSA-1.

Key Properties of HHL (QLSA-1)

-> For a Hermitian & nonsingular matrix => $A \in \mathbb{C}^{2^n \times 2^n}$, & a solⁿ vector => $b \in \mathbb{C}^{2^n}$ $(N = 2^n)$:

- Given oracles to prepare A & b in $O(\text{poly} \log(N))$ time.
- The algorithm computes a solⁿ x s.t. $\|\langle x \rangle - \langle A^{-1} b \rangle\| \leq \varepsilon$.
- Complexity is $O(\text{polylog}(N) s^2 \kappa^2 / \varepsilon)$, where:
  - ↳ κ (condition #): Determines numerical stability; a large κ leads to slower convergence.
  - ↳ s (sparsity of A): Sparse matrices yeild faster computations
  - ↳ ε (Precision requirements): Higher precision increases computational cost

Advantages & Limitations of QLSA-1

② QLSA-2: Linear Combination of Unitaries (LCU) Approach

-> To address QLSA-1's limitations, later methods leveraged the Linear Combination of Unitaries (LCU) techique, forming a more refined QLSA-2.

Key Properties of QLSA-2 (LCU-based method)

- Also works for a Hermitian & invertible A, w/ a given b
- Given oracles to prepare A & b in $O(\text{poly} \log(N))$ time.
- Computes x such that $\|\langle x \rangle - \langle A^{-1} b \rangle\| \leq \varepsilon$ in $O(\text{poly} \log(N\varepsilon) \kappa)$ time.
- Improved complexity: Reduces error scaling from $\text{poly}(1/\varepsilon)$ to $\text{poly}(\log(1/\varepsilon))$, significantly improving precision.

Advantages & Limitations of QLSA-2

☑ Better error handling than QSLA-1

☑ More efficient for well-conditioned systems (low κ)

↳ s (sparsity of A): sparse matrices yield faster computations
↳ ε (Precision requirements): Higher precision increases computational cost

Advantages & Limitations of QLSA-1

- ☑ Exponential speedup over classical methods, provided the matrix is well-conditioned (small K) & sparse
- ⚠ Caveats: High error complexity (scaling w/ poly(1/ε)), limiting practical performance

- ☑ Better error handling than QSLA-1
- ☑ More efficient for well-conditioned systems (low K)
- ⚠ Still relies on efficient quantum preparation, which remains challenging.

③ Implementation in QCFD: Efficient LCU Strategy & End-to-End Setup

This work implements modified algorithms from both QLSA-1 & QLSA-2 by developing a custom LCU decomposition strategy to make quantum fluid simulations feasible

Preparation of Key Components for Quantum Computation

→ To construct an end-to-end method in Quantum Computation of Fluid Dynamics (QCFD), it is essential to prepare:

- Right-hand side vectors: $\{b_{be1}, b_{be2}, b_{se}\}$.
- System matrices: $\{A_{be1}, A_{be2}, A_{se}\}$.

These elements enable post processing of the quantum solution, $\vec{u}$, while ensuring that quantum state preparation aligns w/ the solver's requirements.

Key Takeaways

- QLSA-1 (HHL based) offers exponential speedup, but suffers from high error complexity
- QLSA-2 (LCU-based) significantly reduces error scaling, making it more efficient for high-precision computations
- Efficient state preparation is crucial: w/out effective loading of input data into quantum states, quantum speedup is lost
- End-to-end hybrid method in QCFD: This work ensures a complete pipeline for quantum fluid simulations by optimizing LCU strategies for real-world application

Quantum State Preparation (QSP)

→ Quantum state Preparation (QSP) is essential for encoding $b_{be1}, b_{be2}, \& \ b_{se}$. Two different methods (QSP-1 & QSP-2) implemented are a crucial step for achieving quantum speedup, offering subexponential circuit depth complexity.

QSP-1: Iterative Backward Euler (BE) State Preparation

This method is applied in the case of iterative BE, where b_{be1} must be prepared @ every time step, making it a fully dense vector w/ sparsity: $s \sim O(N_{el})$

→ This method is computationally expensive cause of this

Log-Concave Distribution: For Poiseuille & Couette flows under specific initial conditions, the state @ each time step forms a discrete log-concave distribution (e.g. $\frac{\partial^2 \log(b)}{\partial y^2} < 0$ for $V_t \geq 0$)

$$u(y,t) = \sum_{k=1}^{\infty}\left[\frac{2(1-(-1)^k)}{k\pi}\left(1+\frac{\partial p}{\partial x}\frac{Re}{(k\pi)^2}\right)\sin\left(\frac{k\pi y}{D}\right)e^{\frac{-t}{Re}\left(\frac{k\pi}{D}\right)^2}\right]-\frac{Re}{2}\frac{\partial p}{\partial x}y(1-y). \quad [4]$$

Analytical Validation: The Sol^n $u(y,t)$ can be derived from eqn (4) which validates the preparation method.

Key Considerations for QSP-1

- ☑ Allows for recursive quantum state preparation & measurement
- ☑ Even if the exact analytical sol^n is unknown, the initial condition is the only required information for state preparation
- ☑ Helps analyze whether BE achieves quantum advantage in simulations
- ☑ QSP-1 provides flexibility for real-world fluid simulations where log-concave initial conditions are common
- ⚠ For arbitrary state vectors, this method incurs exponential circuit depth costs
- ⚠ Measuring all qubits in the register @ every time step requires $O(N_{el})$ operations, potentially negating quantum speedup

QSP-1 is more useful for small qubit systems where iterative state updates are needed.

QSP-1.alt Recursive State w/ Measurement

A modified QSP-1 method where all qubits are measured @ each time step incurring $O(N_{el})$ operations which compromises quantum speedup, but introduces measurement errors

Key Considerations

- ☑ Efficient state preparation w/ logarithmic complexity in smaller systems
- ⚠ Frequent qubit measurements hinder speedup in larger simulations

For QSP-1 (Iterative BE): Recursive preparation w/ potential quantum advantage for smaller systems

For QSP-2 (One-Shot BE & FE): Sparse state preparation protocol; useful for efficiently encoding large, sparse matrices

QSP-2: One-Shot State Preparation for BE & FE

This method applies to one-shot methods (BE & FE), where b_{be2} & b_{se} need to be encoded only once for all time steps

Used for Backward Euler (BE) & Forward Euler (FE)

This method applies to one-shot methods (BE & FE), where b_{be2} & b_{se} need to be encoded only once for all time steps

Used for Backward Euler (BE) & Forward Euler (FE)

Sparsity Advantage: Since b_{be2} & b_{se}, are typically sparse, they fall into a lower complexity regime:

-> Since all other quantum registers are initially set to zero, the sparsity of b is $O(N_{el}/m)$, making this a highly sparse problem w/ complexity

-> The vector b in this case is typically larger, scalling as $O(m + pN_y)$ [as defined in SI Appendix]   So $\sim O(N_{el}^{m})$

Since we are encoding in a sparse state, a sparse state preparation protocol is used, coined QSP-2, it has a few advantages:

- Has subexponential-optimal circuit depth, scaling only polynomially w/ vector size (less abnomallies)
- Constructs binary decision trees to encode quantum states efficiently, reducing computational costs

Final Takeaways & Comparison

Method	Best for	Complexity	Limitations
QSP-1 (Iterative BE)	Small, log-concave distributions	$O(log N_{el})$	Measurement errors reduce speedup
QSP-1 (Recursive State Prep.)	Small systems with frequent updates	$O(N_{el})$	Slower for larger problems
QSP-2 (One-Shot Methods)	Large vectors, sparse data	Polynomial in vector size	Tree complexity depends on problem structure

Key Considerations for QSP-2

- ✓ Efficient for one-shot state preparation methods w/ larger vector sizes; reduces coding overhead
- ✓ Uses a binary decision tree approach which reduces state preparation complexity
- ⚠ Depending on the structure of tree used, The complexity can fluxuate, which can lead to implementation challenges, requiring careful optimization