A Hybrid Quantum Solver for the 1D Burgers' Equation

Chenan Wei

August 9, 2025

1 Introduction

The simulation of fluid dynamics presents a significant challenge for classical computation. This work focuses on the 1D viscous Burgers' equation as a prototypical model for Computational Fluid Dynamics (CFD):

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} \tag{1}$$

While simpler than the full Navier-Stokes system, it retains the essential nonlinear convective and dissipative viscous terms, making it an ideal testbed for novel computational approaches. We propose a hybrid quantum-classical algorithm that leverages quantum simulation for the time-evolution step, inspired by recent work in the field [1, 2].

2 Hybrid QTN-HSE Framework

The algorithm employs a hybrid framework combining a **Quantum Tensor Network (QTN)** inspired classical component with a **Hamiltonian Simulation Engine (HSE)** quantum component.

2.1 Hamiltonian Simulation Engine (HSE - Quantum Core)

The core of the algorithm lies in mapping the Burgers' equation to a Schrödinger-like form, as developed by Meng & Yang [1]. The time evolution of the system's quantum state vector, which encodes the fluid velocity, is then performed on a quantum simulator.

2.2 Quantum Tensor Networks (QTN) and State Compression

Before and after the quantum evolution, classical processing is required. This includes encoding the classical velocity field into the quantum state's phase and decoding it back. The framework is inspired by QTN methods which use classical representations like **Matrix Product States (MPS)** to efficiently handle quantum states [2].

An n-qubit quantum state $|\psi\rangle$ resides in a Hilbert space of dimension 2^n. Representing this state requires storing 2^n complex amplitudes, which is classically intractable for even modest n. An MPS decomposes the state's coefficient tensor into a product of n smaller tensors:

$$C_{s_1 s_2 \ldots s_n} = \sum_{\alpha_0,\ldots,\alpha_n} M^{[s_1]}_{\alpha_0,\alpha_1} M^{[s_2]}_{\alpha_1,\alpha_2} \ldots M^{[s_n]}_{\alpha_{n-1},\alpha_n} \tag{2}$$

This decomposition, shown in Figure 1, provides an exponential compression, requiring only $O(n \cdot 2 \cdot \chi^2)$ parameters, where χ is the bond dimension. The conversion from a dense state vector to an MPS is performed classically via a series of Singular Value Decompositions (SVDs). This compression is a key feature of the classical part of our hybrid loop.

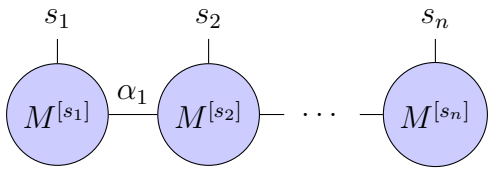

Figure 1: Tensor network diagram for a Matrix Product State (MPS).

2.3 Hybrid Workflow

The workflow for each time step is as follows:

1. **Encode and Prepare**: The classical velocity field $u(x)$ is encoded into the phase of a quantum state $|\psi\rangle$. This dense state vector is then used to prepare the state on the quantum computer.

2. **Evolve (Quantum Step)**: The state $|\psi\rangle$ is evolved for a time step Δt using the HSE, governed by the equation $|\psi(t + \Delta t)\rangle = e^{-iH\Delta t} |\psi(t)\rangle$.

3. **Decode**: The evolved state $|\psi(t + \Delta t)\rangle$ is read out, and the new classical velocity field is decoded from its phase.

4. **Update & Repeat**: The Hamiltonian H is updated with the new velocity field, and the loop repeats for the next time step.

3 PDE Mapping: Burgers' Equation to Schrödinger Form

To simulate this on a quantum computer, we map it to the Schrödinger equation. The velocity field $u(x, t)$ is encoded into the phase S of a complex wavefunction $\psi = Ae^{iS/\nu}$ such that $u(x) = \frac{\partial S}{\partial x}$.

This mapping results in a Schrödinger-like equation $i\frac{\partial|\psi\rangle}{\partial t} = H|\psi\rangle$, where the Hamiltonian H is diagonal in the computational (position) basis:

$$H = \frac{1}{2\nu}u(x)^2 - \frac{\partial u}{\partial x} \tag{3}$$

Because the Hamiltonian is constructed from the velocity field $u(x)$ at each time step, it is state-dependent, necessitating the hybrid classical-quantum loop.

4 Gate Decomposition & Quantum Circuit

4.1 State Preparation from Matrix Product States

While the current code uses the black-box 'qml.StatePrep' for simplicity, a scalable approach relies on preparing the state from its MPS form. The classical MPS description is used to construct a

quantum circuit that prepares the state on an n-qubit register. The unitaries U_k are derived from the MPS tensors $M^{[s_k]}$. Each tensor $M^{[s_k]}_{\alpha_{k-1},\alpha_k}$ is reshaped into a matrix M_k of size $(2\alpha_{k-1} \times \alpha_k)$. A QR decomposition of this matrix, $M_k = Q_k R_k$, yields an isometry Q_k (a matrix satisfying $Q_k^\dagger Q_k = I$). This isometry can be extended to a full unitary U_k that acts on qubit k and an ancilla register representing the bond index. The resulting circuit structure is shown in Figure 2.

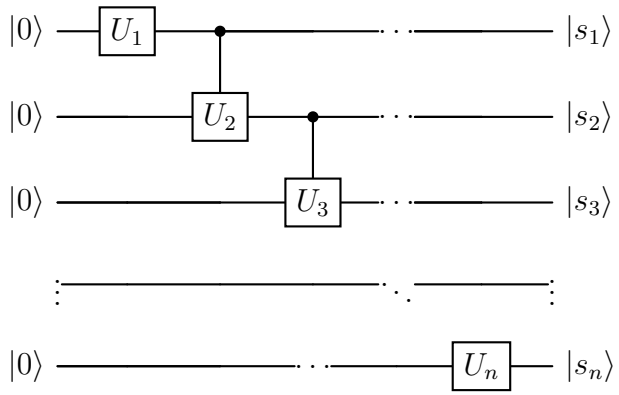

Figure 2: General structure of a quantum circuit for preparing a Matrix Product State.

4.2 Time Evolution Circuit

For a general Hamiltonian with non-commuting kinetic ($\hat{H}_K$) and potential ($\hat{H}_V$) terms, the evolution is implemented using a Trotter-Suzuki decomposition. The circuit for a single Trotter step is shown in Figure 3. This involves transforming to the Fourier basis using a Quantum Fourier Transform (QFT) to apply the kinetic term.

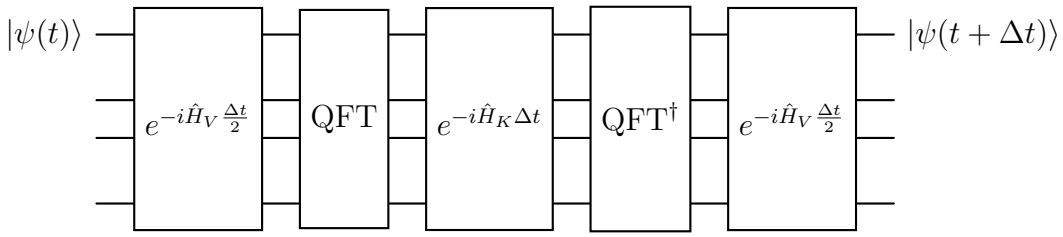

Figure 3: General quantum circuit for a single Trotter step of the HSE time evolution.

Note on Implementation: In the provided Python code, the derived Hamiltonian (Eq. 2) is found to be entirely diagonal in the position basis. This is a significant simplification, as it means the QFT and Trotter steps are not required. The entire evolution $e^{-iH\Delta t}$ can be implemented as a single diagonal unitary gate, dramatically reducing circuit depth.

5 Validation & Benchmark

The hybrid solver was benchmarked against a classical spectral solver for the 1D shock tube problem over 100 time steps ($T = 0.01$ s). The final velocity profiles are shown in Figure 4.

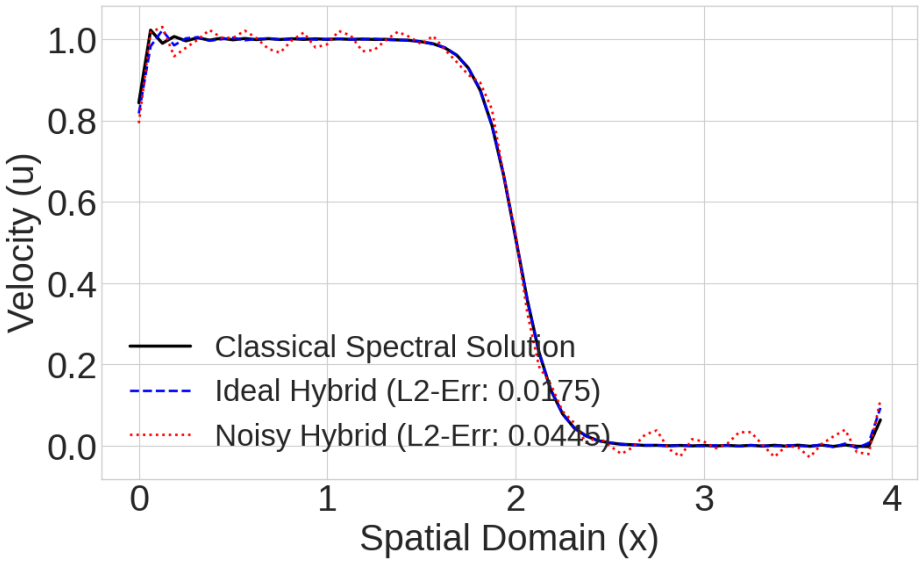

Figure 4: Comparison of final velocity profiles for the classical spectral solver (ground truth), the ideal hybrid solver, and a noisy hybrid solver.

Table 1: Benchmark Summary (N=64 grid points, 6 qubits)

Solver	**Wall-Clock Time (s)**	**L_2-Error vs. Classical**
Classical Spectral	0.0284	N/A
Ideal Hybrid (Noiseless)	1.2725	0.017485
Noisy Hybrid (0.1% error)	6.7625	0.044466

6 Resource & Noise Analysis

The resource requirements and noise tolerance are critical for assessing feasibility.

Table 2: Resource Estimates (per time step, N=64)

Resource	**Count (Abstract)**	**Hardware Scaling**
Qubits	6	$n = \log_2(N)$
Gate Depth	2	Polynomial in n
Total Gate Count	2	Polynomial in n

- **Noise Analysis**: A depolarizing channel with a 0.1% error probability more than doubled the L_2-error, demonstrating high sensitivity.
- **Mitigation Strategy**: For a hardware run, techniques like **Zero-Noise Extrapolation (ZNE)** would be essential.

7 Scalability Analysis

The key to this algorithm's potential is avoiding the exponential scaling that plagues classical CFD and naive quantum state preparation.

- **Classical Scaling**: Classical CFD solvers scale polynomially with the number of grid points, N. For example, a spectral solver using FFTs scales as $O(N \log N)$.
- **Quantum Scaling**: The quantum algorithm uses $n = \log_2(N)$ qubits. The MPS-based state preparation circuit (Figure 2) has a depth that scales polynomially with n. The evolution step, being a single diagonal unitary, also has a polynomial depth upon decomposition for structured Hamiltonians. Therefore, the overall quantum circuit depth scales as $\text{poly}(n) = \text{poly}(\log N)$.

This **polylogarithmic scaling** with the grid size N is the source of the potential exponential speedup over classical methods. The 'qml.StatePrep' function used in the prototype code is a black box that would scale exponentially for an arbitrary state; however, it serves as a stand-in for the efficient MPS preparation circuit which is the true theoretical foundation of this approach.

8 Algorithm Comparison: QTN & HSE Trade-offs

The hybrid approach presents a trade-off between classical and quantum computation.

- **HSE (Quantum)**:
  - **Pros**: Offers a potential exponential speedup in simulation time due to polylogarithmic scaling.
  - **Cons**: Suffers from overheads in state readout (tomography) and is highly susceptible to hardware noise.
- **QTN (Classical)**:
  - **Pros**: The MPS compression provides the key to efficient, polynomial-depth state preparation on the quantum computer.
  - **Cons**: Adds classical computation time to each step for SVD-based compression.

Overall Trade-off: The algorithm's potential for quantum advantage relies on the assumption that the state of the fluid system can be efficiently represented by a low-bond-dimension MPS. If this holds, the polynomial quantum circuit offers a significant scaling advantage over classical methods for very large N.

References

[1] Xiang-Bao Meng and Jin-Lei Yang. "Quantum simulation of the viscous Burgers equation". In: *Physical Review Research* 5.3 (2023), p. 033182. DOI: 10.1103/PhysRevResearch.5.033182.

[2] Raghavendra Dheeraj Peddinti et al. "Quantum-inspired framework for computational fluid dynamics". In: *Communications Physics* 7.1 (Apr. 2024), p. 135. ISSN: 2399-3650. DOI: 10.1038/s42005-024-01623-8. URL: https://doi.org/10.1038/s42005-024-01623-8.