Error Mitigation for Collisionless Quantum Lattice Boltzmann Methods on NISQ Devices

Povel Kann
KTH Royal Institute of Technology
Stockholm, Sweden
pkann@kth.se

***Abstract*—Quantum lattice Boltzmann methods are a promising set of candidates for simulating fluid dynamics on quantum computers, facing two notable barriers. Algorithmic development currently struggles with the non-linearity of the collision operator, while an implementation must tackle the issue of computing on noisy quantum hardware. This study evaluates a remedy for the latter in the form of quantum error mitigation, by using an existing QLBM implementation with a custom interception of the simulation backend to apply zero noise extrapolation. The backend is a noisy quantum simulator, using either a custom noise model or a realistic one based on an IBM device.**

The realistic noise model was found too noisy for the algorithm used; after two time steps, there is no recoverable information. With a uniform depolarizing noise model with variable error rate, however, ZNE proved effective for reducing the noise provided that the noise does not eradicate the data, is large enough that instabilities in the interpolation do not dominate, and is not dominated by shot noise.

***Index Terms*—quantum fluid dynamics, quantum lattice Boltzmann method, quantum error mitigation, zero noise extrapolation**

I. Introduction

As early as 1982, Feynman [3] outlined the difficulties in simulating quantum systems on classical computers, and proposed the idea of a quantum computer. He conjectured that any quantum system may be modelled by a set of simple two-state quantum systems; what we now call qubits. Since then, the idea of a quantum computer has developed from a theoretical proposal to real, usable quantum devices, and a plethora of quantum algorithms. A brief introduction to quantum computing for the uninitiated can be found in appendix B.

While simulation of quantum systems is the most obvious application, it turns out there are several other problems where a quantum algorithm can achieve the coveted exponential speedup over the best known classical algorithms. The most famous example is perhaps Shor's factoring algorithm [16], but a wide range of algorithms promise polynomial or exponential speedups on a sophisticated quantum device. At the time of writing, however, we are in what John Preskill calls the NISQ era (Noisy Intermediate-Scale Quantum) [12] with current quantum devices being too small and too noisy to reach quantum advantage using any discovered algorithm. Still, none can tell when or in what form these more sophisticated quantum computers may appear, and once they do our algorithms and methods must be ready to meet them.

Computational fluid dynamics (CFD) is one of today's heaviest stress tests for classical computers. Despite decades of advances in both hardware and numerical methods, direct numerical simulation of fluid flow often remains an intractable problem. It is therefore worth exploring, as many already have, whether there is potential for quantum advantage even in this field. The first methods suggested were ones aimed at solving general linear systems of equations (LSE). Among the first of these was the Harrow–Hassidim–Lloyd (HHL) algorithm [6], though this is not a viable method for CFD problems on today's devices. We find another route in variational quantum linear solvers (VQLS) [1], which have successfully been used to solve problems of potential flow and Stokes flow [11]. The downside here is that variational quantum algorithms, which solve an optimization problem, in general currently suffer from the so-called barren plateau phenomenon [9] — gradients vanishing for large problem sizes, making optimization tricky.

There is also a different route for CFD, besides dealing with matrix equations, in lattice Boltzmann methods (LBM).

A. The Lattice Boltzmann Method

The 1980s saw the introduction of lattice gas methods for fluid simulation; modelling particles moving and colliding on a discrete lattice, as an alternative to solving Navier-Stokes equations. The particles were later replaced by a density distribution, which is evolved by consecutive streaming and collision operations on the lattice. This latter strategy is known as a lattice Boltzmann method [17], and equates to solving a discrete lattice Boltzmann equation (LBE). The LBE takes the form

$$\frac{\partial f}{\partial t} + \mathbf{u} \cdot \nabla f = \Omega(f). \tag{1}$$

where f is the particle distribution and $\Omega(f)$ the collision operator — the way the state changes due to particle collisions. Most commonly used is the Bhatnagar-Gross-Krook (BGK) collision operator $\Omega(f) = -\frac{1}{\tau}(f - f^{eq})$ which models relaxation toward the equilibrium function f^{eq} in relaxation time τ.

B. The Quantum Transport Method

Because of the linear nature of quantum operators, and therefore also quantum circuits, the non-linearity of the collision operator is a problem. The simplest solution is to skip it, and solve only the transport equation

$$\frac{\partial f}{\partial t} + \mathbf{u} \cdot \nabla f = 0. \tag{2}$$

A quantum transport method (QTM), or collisionless QLBM, has been implemented by Schalkers and Möller [13] [14]. In fact, the former of these papers contains a proof of the impossibility of representing both streaming and collision as unitary operations using either amplitude encoding or computational basis state encoding.

Of particular interest to this study is the implementation of a multi-speed collisionless QLBM (MSQLBM) formulated in the above-mentioned papers. This algorithm, in short, comprises the following parts:

- A qubit register $\left|a_{n_a} \dots a_1 \; g_{n_g} \dots g_1 \; v_{n_v} \dots v_1\right\rangle$ where a_i are ancilla qubits, g_i encode position, and v_i encode velocity. For simplicity, we will assume a two-dimensional simulation on a d by d lattice with four discrete velocities per dimension. We then have $n_a = 6$, $n_g = 2\lceil \log_2 d \rceil$, and $n_v = 4$, totalling $10 + 2\lceil \log_2 d \rceil$ qubits. The general formulas are found in the first paper [13].
- Velocities are encoded using sign-magnitude encoding. This way, a velocity can be reflected in $O(1)$ time by flipping the most significant bit.
- A streaming operation following a cyclic quantum Draper adder approach [2], using quantum Fourier transform (QFT) and phase shifts.
- A specular reflection operation. This is the main contribution of the paper; using ancilla qubits to eliminate the risk of unphysical solutions upon collision with corners.

After simulation, the fluid field is represented by the probability density distribution of the resulting quantum state, which can be approximated by many shots and measurements. For more details, I refer to the source material.

It is worth noting that, since this is a quantum algorithm, measurement collapses the quantum state. This means that a simulation cannot be continued after measurement; each shot runs from the initial time. Simulating n time steps and measuring each of them therefore requires $O(n^2)$ circuit executions. Additionally, while a doubling of the side length of a square lattice only increases the qubit count by two, such an increase still has a significant impact on the simulation time required; the dimension of the encoded state space increases exponentially with the number of qubits, and the number of shots required to achieve some given accuracy in the quantum measurement increases proportionally to the number of lattice points.

C. The `QLBM` Framework

The Python package `QLBM` [4] by Georgescu et al. is a framework for quantum computational fluid dynamics (QCFD) which, as of December 2025, comprises implementations of two algorithms: The collisionless QLBM [13] and the Space-Time QLBM (STQLBM) [15].

D. Simulating a Quantum Backend

`QLBM` is compatible with the *Qiskit Aer* quantum simulator [7], which supports several simulation methods and noise models, including fake backends; noise models based on data from real IBM quantum computers.

E. Quantum Error Mitigation

Until less noisy qubit technologies are developed, this noise must be mitigated in some way. In quantum error correction, the idea is to use error correcting codes to build logical qubits out of several physical ones. Building and running larger circuits on logical qubits is currently a matter of theoretical study, however, and not practical implementation. Instead, we can use strategies for quantum error mitigation; post-processing the measurements from the quantum computer (or simulator) to eliminate noise in the data.

1) Zero Noise Extrapolation: One way of mitigating noise is to compute expectation values at different noise levels, and then extrapolate what the expectation value should be in the noiseless limit; so-called zero noise extrapolation (ZNE) [5]. The amount of noise in the output can be increased by increasing the depth of the circuit, for example by making a mapping $G \mapsto GG^\dagger G$ — so called *unitary folding*, where G is either the whole circuit or each individual gate. Since all quantum gates and circuits are unitary, meaning $G^\dagger G = I$, this would be equivalent were there not non-unitary noise along the way. This way, the overall circuit remains the same, while the folding factor essentially scales the noise. More specifically, the map $G \mapsto G(G^\dagger G)^n$ can be considered to scale the noise by a factor $2n+1$.

With data using different amounts of noise, the expectation value can be interpolated as a function of the noise level, be it linearly, polynomially, or exponentially, after which said interpolation can be used to estimate the noiseless expectation value.

ZNE is implemented by LaRose et al. in the Python package `Mitiq` [10].

II. Method

The Python code with which the simulations were run can be found in appendix A, together with setup instructions.

There are a number of potentially interesting variable parameters. The following were considered in the implementation:

1) `d` — The lattice's side length.
2) `steps` — The number of time steps the simulation runs.
3) `shots` — The number of shots used for the quantum computation.
4) `noise_model` — The noise model used for noisy simulation.
5) `zne_scales` — The factors to which the noise is scaled for ZNE.

The chosen simulation setup comprised a 4 by 4 lattice using a D2Q4 model with periodic boundary conditions. In the initial state, the velocities in the left half of the lattice (the lower half of the span of x) were set to a uniform positive value in the x direction. The result of a simulation was visualized as a fluid density field. Each experiment involves three simulations: An ideal simulation without noise, a noisy simulation using some noise model, and an error mitigated simulation using ZNE. To evaluate the effectiveness of the error mitigation, the root-mean-square error (RMSE) of the ideal and noisy density fields, as well as the ideal and error mitigated density fields, was computed for each time step.

A. *QLBM Implementation*

The simulation was performed using the multi-speed collisionless quantum lattice Boltzmann algorithm (MSQLBM) in the `QLBM` Python package. This included defining the lattice and geometry, the choice of quantum simulation backend, the initial conditions, and the aforementioned parameters. The simulation backend was provided by Qiskit Aer, which allowed for different choices of state representation and different noise models. The simulation method was chosen automatically by `AerSimulator` based on the circuit and noise model. The `QLBM` package does not inherently support extraction of the raw output from the quantum simulator, and is intended to perform all computation and visualization behind the scenes, so to implement ZNE a custom backend wrapper `BackendSpy` was developed. The `QLBM` algorithm constructs and executes circuits directly on the provided backend. The spy intercepts these circuits and generates a batch of unitarily folded circuits using `Mitiq`, which are passed along to the true quantum simulator backend that performs the simulation and returns the measured counts — the number of occurrence of each measured output state, given the circuit and number of shots. This data is stored in `json` files and fed back to `Mitiq` for interpolation, while the results from the unscaled circuit are returned to `QLBM`. A separate function was written to construct `vti` images of the density fields from the raw counts, so that the error mitigated results could be visualized as well.

B. *Noise Models*

Two different noise models were used for experimentation: `FakeFez`, which is a noise model simulating the noise of the IBM Fez quantum computer, and a uniform depolarizing noise model. The former serves as a baseline for current hardware capabilities, while the latter acts as a proxy for logical error rates in future fault-tolerant era devices. The depolarizing noise model contained three forms of noise and depended on one variable p_{err}: The one-qubit gates exhibited depolarizing errors at a rate of $0.1p_{err}$, the two-qubit gates exhibited depolarizing errors at a rate of p_{err}, and the measurement exhibited Pauli-errors at a rate of 0.01. These values were chosen to reflect the status quo for readout errors in quantum devices, with room for smaller than realistic gate error rates, as might be expected in future devices.

C. *Error Mitigation*

Zero noise extrapolation in `Mitiq` was used to mitigate errors, using Richardson extrapolation or linear extrapolation, with unitary folding and scale factors 1, 3, and 5.

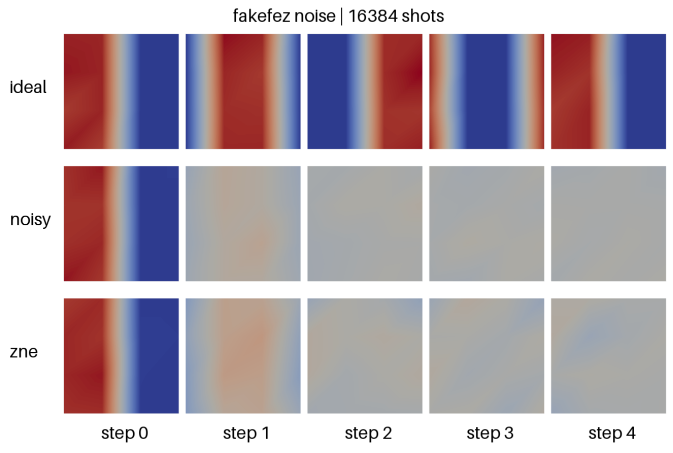

Fig. 1. Results of 2^{14} shots using the `FakeFez` noise model.

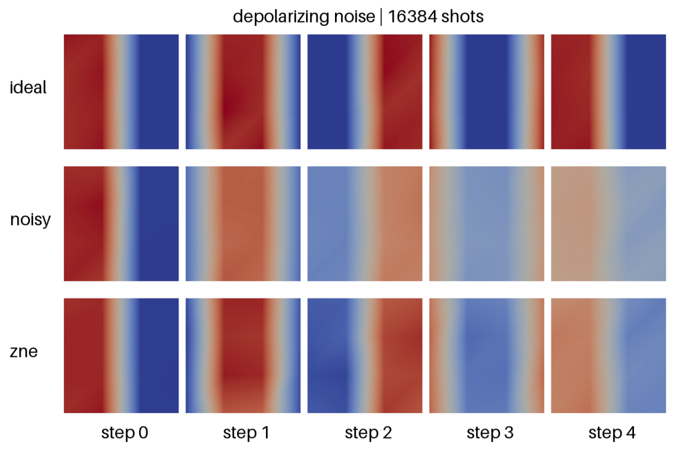

Fig. 2. Results of 2^{14} shots using the depolarizing noise model with $p = 0.001$.

D. *Visualization*

The simulations resulted in spatiotemporal density fields, which were visualized using `PyVista`.

III. Results

The density field visualizations using the `FakeFez` and depolarizing noise models are shown in figures 1 and 2, respectively. Each time step, meaning each square in the figures, was generated from 2^{14} shots.

Figures 3 and 4 show how the root-mean-square error between ideal, noisy, and error mitigated results depends on the number of shots, for both noise models. Figure 5 shows how the RMSE using the depolarizing noise model depends on the model's error rate. All errors are computed with reference to a noiseless simulation with the same number of shots.

IV. Discussion

The first evident result is that the realistic noise model is too noisy for this computation to be of any practical use. The noisy simulation in figure 1 becomes diffuse already in step 1, and is unrecognizable by step 2. From figure 3 we can see

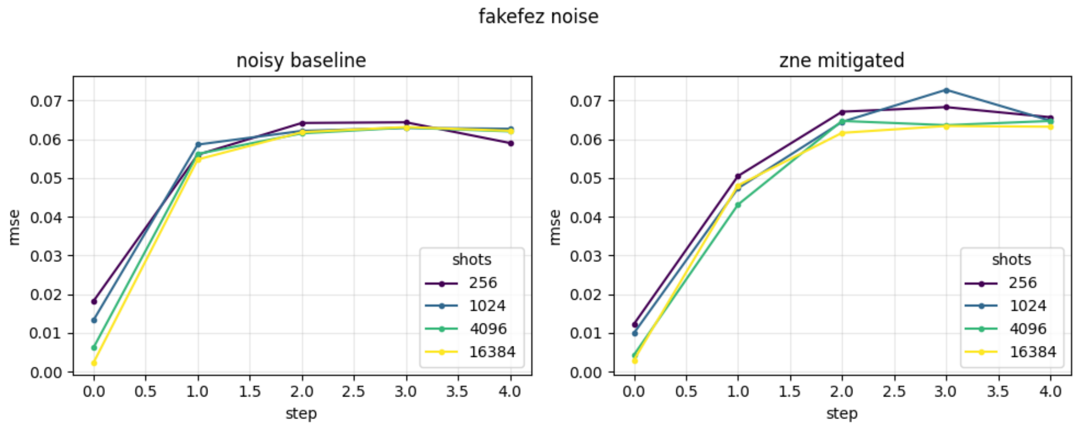

Fig. 3. Density RMSE relative to the noiseless simulation, using the `FakeFez` noise model and Richardson extrapolation for ZNE.

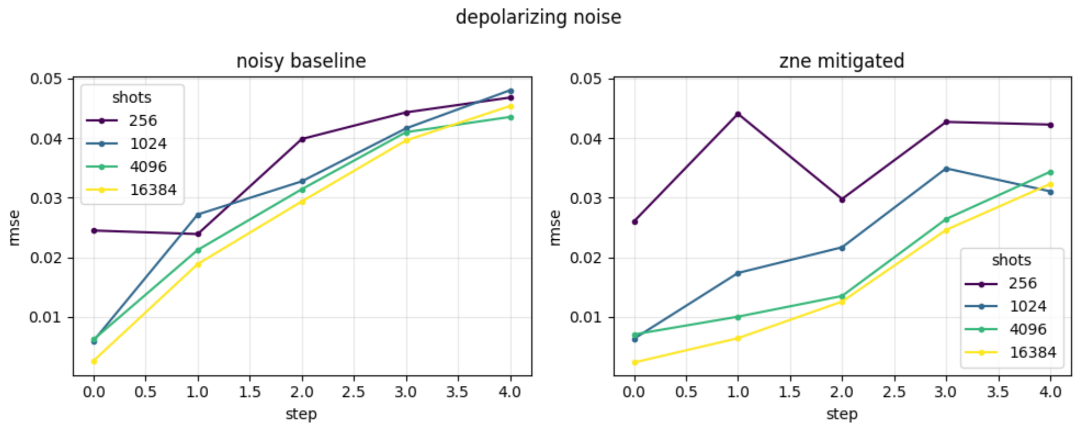

Fig. 4. Density RMSE relative to the noiseless simulation, using the depolarizing noise model with $p = 0.001$ and Richardson extrapolation for ZNE.

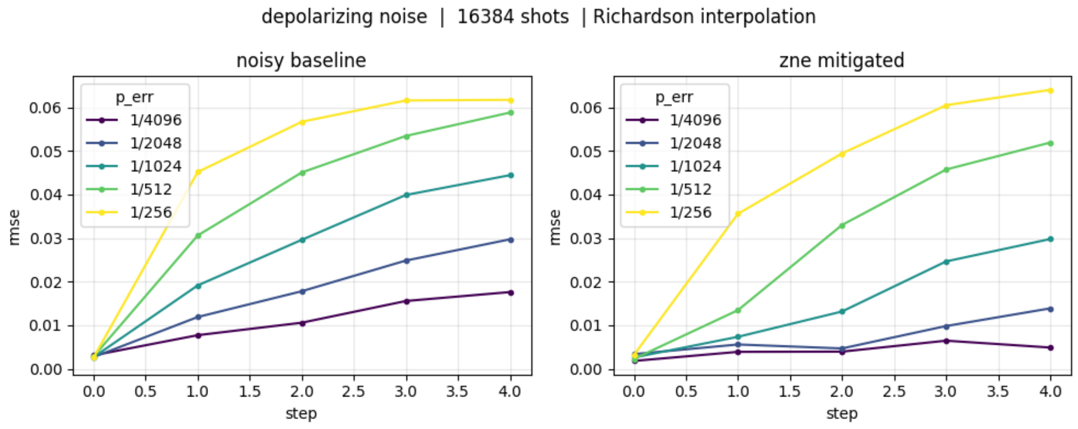

Fig. 5. Density RMSE relative to the noiseless simulation, using the depolarizing noise model with 2^{14} shots and Richardson extrapolation for ZNE.

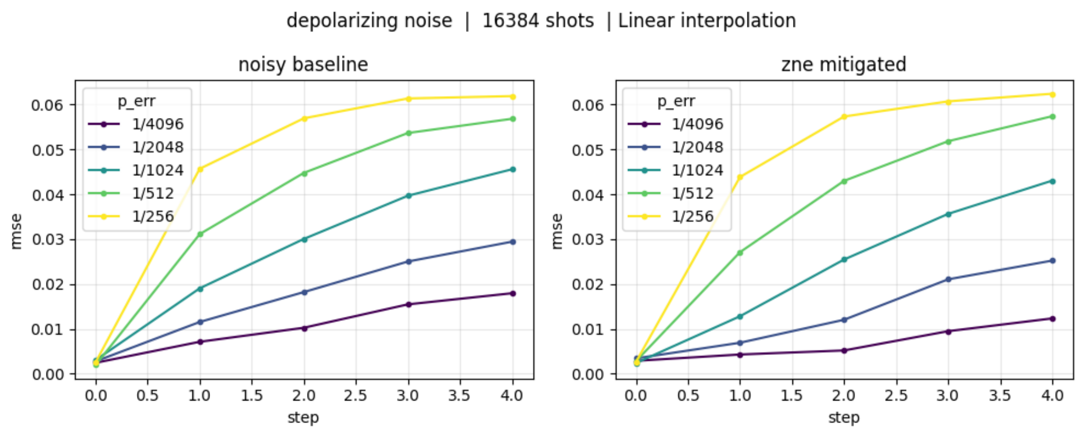

Fig. 6. Density RMSE relative to the noiseless simulation, using the depolarizing noise model with 2^{14} shots and linear extrapolation for ZNE.

that ZNE makes a slight improvement to step 1. From step 2 and onwards, however, only random noise remains, which ZNE can do nothing about. This means that the circuit used is simply too deep for the noisy quantum devices of today. The uniform depolarizing noise model gives much more promising results; with noisy but still identifiable density fields even in step 4. ZNE improves the noisy result and reduces the RMSE by a not insignificant amount.

When the shot count is smaller, a different kind of noise interferes with the results. Since the simulation output is sampled from the true underlying distribution, a sufficient number of shots (samples) are necessary to acquire a good approximation of said distribution. We call this shot noise, which follows a statistical error scaling of $1/\sqrt{N_{shots}}$. In figures 3 and 4, the baseline plots seem to indicate that an increase from 2^{12} to 2^{14} shots has little impact on the simulation itself. However, this halves the statistical uncertainty, and in the error mitigated plots, it appears that this reduction makes the zero noise extrapolation more stable. This aligns with intuition; shot noise introduces random fluctuations in the output, which the extrapolation will likely magnify. For example, the ZNE solutions in step 1 in figures 1 and 2 contain larger such fluctuations than are visible in the noisy or ideal solutions. In short, ZNE is sensitive to shot noise and requires a high shot count to suppress sampling error below the threshold where extrapolation amplifies it.

In figure 5, the error mitigation proves its effectiveness in the middle range of gate noise disturbance. Once the data is so noisy as to make the underlying information irretrievable, the error mitigation merely exaggerates randomly appearing patterns, causing a variance in the error but no consistent improvement. When there is sufficiently little noise, the random variance introduced by error mitigation is more likely to increase the RMSE than improve it. When the noise, on the other hand, is noticeable, but not so heavy as to render the image unrecognizable, ZNE does a good job of reducing the error. Notably, the error mitigated result using a noise rate $p_{err} = 1/2048$ displays improvement — especially in later time steps — over the untreated result with $p_{err} = 1/4096$.

As evidenced by figure 6, linear interpolation performs significantly worse than using Richardson interpolation for zero noise extrapolation with depolarizing noise. In their ZNE paper [5] Giurgica-Tiron et al. derive analytically that unitary folding with a scale factor $\lambda = 1+2n$ exponentially scales the amount of depolarizing noise. It is therefore expected that the linear interpolation would perform poorly. Still, this shows the importance of choosing an appropriate extrapolation model for the type of noise encountered. Exponential interpolation was also attempted, but did not converge.

V. Conclusion

A realistic noise model, based on measurements from an IBM QPU, is too noisy for the collisionless lattice Boltzmann algorithm developed by Schalkers and Möller. A small idealization, however, in the form of a uniform depolarization noise model with a smaller than realistic noise rate for one- and two-qubit gates, makes for successful simulations for a few time steps. Zero noise extrapolation, using Richardson extrapolation with scale factors 1, 3, and 5, proved an effective error mitigation strategy for a range of noise rates, and decreased the root-mean-square error relative to the noiseless case as long as the noisy data still bore some resemblance to it. When the result is too noisy, there is not enough information left for the noise mitigation strategy to procure anything useful, and the error is not reduced. When there is too little noise, on the other hand, the uncertainty introduced by ZNE dominates, and the error may be increased by attempting noise mitigation. There is a significant range in between, however, where ZNE reduces the error substantially; a range in which we may well find the results of many quantum algorithms performed on real QPUs today. If the noise rate of the realistic noise model used in this study was one single order of magnitude smaller, the results of the MSQLBM algorithm would also join that group.

A. Future Work

Qiskit supports GPU-acceleration on Linux machines. This might speed up the simulation significantly, allowing simulations to use:

- more time steps,
- more shots,
- larger lattices, and
- objects for the fluid to interact with, with bounce back boundary conditions.

In their second paper on collisionless QLBM [14], Schalkers and Möller explain how the momentum exchange method can be used to compute forces — e.g. drag and lift coefficients — on objects with bounce back boundary conditions in a QLBM simulation. It would be intriguing to see whether, and under which conditions, noise mitigation strategies improve the estimates of such measurements, which may be of greater interest than the overall appearance of the density field.

There are many more error mitigation strategies worth considering. The `Mitiq` package contains implementations not only of ZNE, but also probabilistic error cancellation (PEC), Clifford data regression (CDR), readout-error mitigation (REM), and more. Some of these are simpler to implement than others, and some more suitable than others for the deep QLBM circuits, but it is entirely possible that one of them can outperform ZNE at this particular task. Parameter choices for ZNE could also be further explored; the scale factors and extrapolation method used. It is natural to choose the smallest odd integers as scale factors, since a full unitary fold adds two to it, but it is not impossible that a different configuration could improve performance. Depending on the circuit and the type of noise encountered, different extrapolation techniques perform drastically differently. It is worth evaluating different options both analytically and experimentally.

The relationship between shot count and ZNE stability also warrants further investigation. While ZNE reduces the bias introduced by gate noise, it also seems to amplify the variance due to shot noise. It would be valuable to have a measure of this variance-shots relationship, to more rigorously determine under what gate error rates and shot counts ZNE is effective.

Finally, more QLBM algorithms should be studied. For example, the ready-implemented Space-Time QLBM [15], which solves the issue of the non-linear collision operator using a clever encoding strategy. This is arguably the most promising direction of development as far as QLBMs are concerned, and has more potential for useful fluid flow simulations than a collisionless QLBM.

References

[1] Carlos Bravo-Prieto et al. “Variational quantum linear solver”. In: *Quantum* 7 (2023), p. 1188.

[2] Thomas G. Draper. *Addition on a Quantum Computer*. 2000. DOI: 10.48550/ARXIV.QUANT-PH/0008033. URL: https://arxiv.org/abs/quant-ph/0008033.

[3] Richard P. Feynman. “Simulating physics with computers”. In: *International Journal of Theoretical Physics* 21.6–7 (June 1982), pp. 467–488. ISSN: 1572-9575. DOI: 10.1007/bf02650179. URL: http://dx.doi.org/10.1007/BF02650179.

[4] Călin A. Georgescu, Merel A. Schalkers, and Matthias Möller. “qlbm – A quantum lattice Boltzmann software framework”. In: *Computer Physics Communications* 315 (2025), p. 109699. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2025.109699. URL: https://www.sciencedirect.com/science/article/pii/S0010465525002012.

[5] Tudor Giurgica-Tiron et al. “Digital zero noise extrapolation for quantum error mitigation”. In: *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, Oct. 2020, pp. 306–316. DOI: 10.1109/qce49297.2020.00045. URL: http://dx.doi.org/10.1109/QCE49297.2020.00045.

[6] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Physical Review Letters* 103.15 (Oct. 2009). ISSN: 1079-7114. DOI: 10.1103/physrevlett.103.150502. URL: http://dx.doi.org/10.1103/PhysRevLett.103.150502.

[7] Ali Javadi-Abhari et al. *Quantum computing with Qiskit*. 2024. DOI: 10.48550/arXiv.2405.08810. arXiv: 2405.08810 `[quant-ph]`.

[8] Eric Johnston, Mercedes Gimeno-Segovia, and Nic Harrigan. *Programming quantum computers*. en. 2019.

[9] Martín Larocca et al. “Barren plateaus in variational quantum computing”. In: *Nature Reviews Physics* 7.4 (Mar. 2025), pp. 174–189. ISSN: 2522-5820. DOI: 10.1038/s42254-025-00813-9. URL: http://dx.doi.org/10.1038/s42254-025-00813-9.

[10] Ryan LaRose et al. “Mitiq: A software package for error mitigation on noisy quantum computers”. In: *Quantum* 6 (Aug. 2022), p. 774. ISSN: 2521-327X. DOI: 10.22331/q-2022-08-11-774. URL: http://dx.doi.org/10.22331/q-2022-08-11-774.

[11] Y.Y. Liu et al. “A variational quantum algorithm-based numerical method for solving potential and Stokes flows”. In: *Ocean Engineering* 292 (2024), p. 116494. ISSN: 0029-8018. DOI: https://doi.org/10.1016/j.oceaneng.2023.116494. URL: https://www.sciencedirect.com/science/article/pii/S0029801823028780.

[12] John Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (Aug. 2018), p. 79. ISSN: 2521-327X. DOI: 10.22331/q-2018-08-06-79. URL: http://dx.doi.org/10.22331/q-2018-08-06-79.

[13] Merel A. Schalkers and Matthias Möller. “Efficient and fail-safe quantum algorithm for the transport equation”. In: *Journal of Computational Physics* 502 (Apr. 2024), p. 112816. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2024.112816. URL: http://dx.doi.org/10.1016/j.jcp.2024.112816.

[14] Merel A. Schalkers and Matthias Möller. “Momentum exchange method for quantum Boltzmann methods”. In: *Computers & Fluids* 285 (2024), p. 106453. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2024.106453. URL: https://www.sciencedirect.com/science/article/pii/S0045793024002846.

[15] Merel A. Schalkers and Matthias Möller. “On the importance of data encoding in quantum Boltzmann methods”. In: *Quantum Information Processing* 23.1 (Jan. 2024). ISSN: 1573-1332. DOI: 10.1007/s11128-023-04216-6. URL: http://dx.doi.org/10.1007/s11128-023-04216-6.

[16] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. ISSN: 1095-7111. DOI: 10.1137/s0097539795293172. URL: http://dx.doi.org/10.1137/S0097539795293172.

[17] Alexander J Wagner. “A practical introduction to the lattice Boltzmann method”. In: *Adt. notes for Statistical Mechanics* 463.2008 (2008), p. 663.

Appendix

A. Source Code

The source code used to generate the simulation data and visualizations in this report is available on GitHub. This code used `QLBM` version 0.0.6, `Mitiq` version 0.48.1, and `Qiskit Aer` version 0.17.2, and may break with future API updates to either of these.

Repository: https://github.com/pouvelenc/DD2444-projekt

Setup: The project requires a specific installation order to manage dependencies between `qlbm`, `Qiskit`, and `Mitiq`. An installation guide is provided in the repository's `README.md` file

B. Quantum Computing Fundamentals

Below is a very brief introduction to quantum computing. For further details, see *Programming Quantum Computers* by Johnston et al. [8].

1) Qubits: A quantum computer is constructed from qubits. A qubit is a two-state quantum system $|\psi\rangle = a|0\rangle + b|1\rangle$, where $a, b \in \mathbb{C}$, $|a|^2 + |b|^2 = 1$, and $|0\rangle$ and $|1\rangle$ are the two basis states of the system. When the qubit $|\psi\rangle$ is measured, the possible measurement outcomes are $|0\rangle$ (with probability $|a|^2$) and $|1\rangle$ (with probability $|b|^2$). This way, a qubit is a vector in 2-dimensional Hilbert space, and a system of n qubits a normalized vector in 2^n-dimensional Hilbert space; we have some probability of measuring each of the 2^n basis states in the system, or *quantum register*.

2) Gates: To control qubits, we use unitary operators called *gates*. Two illuminating examples are the Pauli X gate — which is the quantum equivalent to the classical *NOT* gate — and the Hadamard gate H —- which is useful for putting qubits in an equal superposition of its basis states.

$$X|\psi\rangle = X(a|0\rangle + b|1\rangle) = b|0\rangle + a|1\rangle$$
$$H|\psi\rangle = H(a|0\rangle + b|1\rangle) = \frac{1}{\sqrt{2}}\left((a+b)|0\rangle + (a-b)|1\rangle\right)$$

Gates can also act on multiple qubits. The CX gate (controlled X) has a control qubit and a target qubit, and applies an X gate to the target if the control qubit is measured in state $|1\rangle$. This is clearer if we draw a quantum circuit.

3) Circuits: A quantum circuit is a diagram of a qubit register over time, as gates are applied. One of the simplest interesting circuits — and which happens to nicely demonstrate entanglement — is the one shown in figure 7 which produces a Bell state. In the diagram, we see the two-qubit register in its initial state on the left, and a channel corresponding to each qubit. A Hadamard gate is first applied to the top qubit, after which a CX gate is applied, with the top (control) qubit controlling the bottom (target) qubit. The number of gate

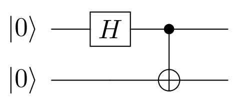

Fig. 7. Circuit which creates a Bell state.

“layers” in a circuit is known as the circuit's *depth*. The Bell circuit has depth 2.

4) Entanglement: The resulting state of this circuit, before measurement, is

$$\frac{|00\rangle + |11\rangle}{\sqrt{2}}.$$

This is interesting because, while measuring one of the qubits to be $|0\rangle$ or $|1\rangle$ is equally likely, once it is measured we know for certain what the measurement outcome of the other qubit will be: We have collapsed the state of the second qubit, despite measuring only the first. This is a frequently used phenomenon in quantum algorithms.

5) Measurement: There are several strategies for encoding the relevant data in a quantum register. The MSQLBM, for example, uses amplitude-based encoding where information is stored in the amplitudes of the basis states. This is a very efficient way of storing information; n qubits means 2^n basis states with amplitudes that are theoretically adjustable with arbitrary precision. Extracting these amplitudes is an issue, however. As mentioned, a measurement collapses the state so that the outcome is one of the basis states. Therefore, one measurement gives very little of the desired information on precise amplitudes. To extract more information, the circuit must be prepared and run again from the beginning, to be measured once more. This is called a *shot*. As the number of shots increases, we expect that the normalized distribution of measurement results will converge toward the true probability distribution, which tells us the squares of the amplitudes: Information retrieved.

Since the measurement outcome of any particular basis state is binary, each shot of an algorithm can be modelled as a Bernoulli trial, with each basis state having some probability p of being the measurement outcome. The variance of such a measurement over N shots is then the binomial variance $p(1-p)/N$, meaning the standard error scales proportionally to $1/\sqrt{N}$

6) Decoherence: A real qubit, whatever technology it is based on, is an open quantum system. It must be, for interaction with gates and other qubits to be possible. This also means that some unwanted interaction between a qubit and its environment is inevitable. The process of information stored in a qubit being lost over time due to unpredictable interactions is called decoherence. This is the source of the noise in NISQ devices. A deeper circuit gives more opportunities for decoherence to occur, which is why gate folding is used in ZNE to increase the noise level.