A Novel Framework in Leveraging Optimization Physics Models using Quantum Computing: Techniques for the 2-D Hubbard Model

Jefferson Lin, Northeast High School

Mentor: Ahmed Ayman, El-Sadat STEM School

Abstract

The new and emerging field of quantum computing harnesses the understanding of the complex dynamics of quantum systems, promising to advance and revolutionize scientific fields that can be applied in our world. However, realizing this potential currently faces its challenges: scaling error-corrected qubits, parameter spaces, and efficiently compiling quantum circuits given hardware constraints. This paper reviews techniques to address these obstacles by integrating automatically differentiable quantum circuits (ADQCs) with tensor networks (TN) to enable reverse-mode automatic differentiation for efficient optimization. We propose applying this ADQC-TN framework to the 2D Hubbard model, which is a foundational model of strongly correlated electron systems exhibiting rich phase diagrams, including unconventional superconductivity. This framework can elucidate detailed mechanisms, such as how dopant atoms influence superconducting electron pairing, by training the model's hopping, interaction, chemical potential, and other parameters on experimental measurements. Robust optimization represents a pivotal bridge between quantum computing and experimental condensed matter physics to advance quantum-based materials modeling and discovery, which has already seen success from the extension of the ground state of quantum lattice models with low fidelities. Successfully trained Hubbard models could facilitate the analysis and understanding of the effects of parameter-tuning and the potential of defects on superconductivity that can aid in other future modeling discoveries through bridging the gap between quantum computing and physical engineering.

I. Introduction

When John Dalton first discovered the atom, a whole quantum world was waiting to be unveiled, a world where the principles of reality break down and what governs is beyond the naked eye. Based on these quantum mechanics, Feynman and other scientists envisioned using quantum computers to simulate quantum systems under the premise that an initial quantum state can be unitarily evolved with a quantum computer that is polynomial in size and

evolution time [1]. Such a potential, where a model Hamiltonian behavior could be simulated, became more than just an idea that physicists can use to broaden their understanding and determine what is intractable with classical, traditional computers, from the smallest distances to cosmological extents [2]. Because classical computers are especially bad at simulating quantum dynamics in predicting how a highly entangled quantum state will change with time [3], this newfound role of quantum computers in astronomically advanced modeling across physics, chemistry, and other domains [4] requires new tuning parameter methodologies to achieve more robust, scalable, and programmable methods in concurrence with technology application [5]. To match the new growing demands of limited hardware, costly simulations, being differentiable, challenging parameter spaces, and more, finding a way to optimize model parameters to match real measurements to have more accurate models and simulations compared to experimental results is becoming a necessity as shown in Fig. 1.

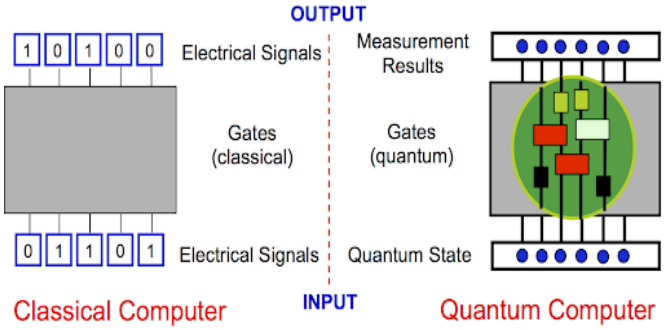

Fig. 1. Illustration of classical computing and quantum computing structure with differing capabilities. Source: [6]

Classical simulations of physical systems typically begin by solving differential equations, for the first order has the approximation $x(t + dt) \approx x(t) + f(x)dt$. However, when we try to simulate the microscopic world in classical computers, the equations of motion are too complex as they are limited by conventional computing and classical physics. So, we must use quantum computing to simulate quantum systems. Through quantum computing (QC), a complex quantum system can be isolated and controlled with sufficient precision to maintain quantum coherence [7]. We can manipulate these quantum states themselves, unlike classical computers, by implementing a universal set of quantum logic gates to approximate any unitary operation on the simulator's qubits. Because Hilbert space grows exponentially with system size $\dim(H_{qubit})^{\otimes N}$, simulating quantum many-body dynamics on a classical computer becomes intractable. Quantum simulators can exploit effects such as entanglement, superposition, and parallelism to encode and manipulate state space and operations compactly, allowing for massively parallel computation on the state vector, circumventing the limitations of classical simulation to develop quantum algorithms that scale sub-exponentially in resources with the system size and desired accuracy. QC offers a computational advantage by meticulously using an exponentially ample Hilbert space for quiet registers (usually through sampling from quantum states created by random entangling circuits) but is limited to specific

tasks and problem 2 types. This advantage can be seen in molecular simulations with one such approach using Lie-Trotter-Suzuki product formulas [8] for Hamiltonian simulations, which reduces scalar error when each decomposition can be implemented into a quantum circuit. Similarly, recent developments in more optimal differentiable quantum generative models DQGMs [9] (on solving Fokker-Planck SDE equations) sampling enable the encoding of classical data into quantum states and the optimization of the quantum circuit parameters. All of this brings an essential discussion of using these quantum systems to sample, solve efficiently, and model problems to reap the far-fetching benefits of QC in cryptography, simulations, optimization, and machine learning.

II. Quantum Principles

A. Schrödinger's Equation

In simulating quantum mechanics, we are first interested in the solution of the time-dependent Schrodinger equation,

$$i\hbar\frac{\partial}{\partial t}|\,\psi(t)\rangle = \hat{H}|\psi(t)\rangle$$

(2.1.1)

where $H = -\frac{\hbar^2}{2m}\nabla^2 + V$ represents the Hamiltonian or the total energy of the wavefunction ψ. This can be further thought of as $\psi(t) = e^{-iHt}\psi(0)$, where the operator e^{-iHt} propagates the initial state $\psi(0)$'s evolution through time. The Schrodinger's equation is a useful representation of quantum dynamics because it fundamentally describes how quantum systems evolve and their Eigen states, respectively. Additionally, it takes a time-independent form $\hat{H}\psi = \hat{E}\psi$, but we will be focusing mainly on the time-dependent form and the applications resulting from it.

B. The Hamiltonian Approximation

The first idea is to approximate $\psi(t+\Delta t)$ as $(I - iH\Delta t)\psi(t)$ like classical conventions, but, however, it is not satisfactory enough. We must turn to using the operator $e^{-iH\Delta t}$ for $\psi(t+\Delta t) = e^{-iH\Delta t}\psi(t)$, under a sufficiently small-time step Δt. For local Hamiltonians like Ising and Hubbard models, we can efficiently simulate by decomposing into smaller subsystems such that their complexity is $O(polyN)$, since applying e^{-iHt} directly is expensive. For Hamiltonians like these as sub-Hamiltonians, it is first formulated by $H = \sum_a^b H_a$. Then, we can apply the Trotter-Suzuki to approximate and decompose the time evolution operator,

$$e^{-iHt} = e^{-it\sum_a^b H_a} = \lim_{n\to\infty}\left(\prod_a^b e^{-iH_a\frac{t}{n}}\right)^n$$

(2.2.2)

This simplifies to,

$$\left(\prod_{a}^{b} e^{-iH_a\frac{t}{n}}\right)^{n} + O\left(\frac{b^2t^2}{n}\right).$$

(2.2.3)

The full Hamiltonian is decomposed into b local terms H_a with e^{-iH_at} representing its approximate evolution under the error $O\left(\frac{b^2t^2}{n}\right)$ with respect to the number of the local Hamiltonian terms. For n Trotter steps, there is a tradeoff to balance between accuracy and efficiency because the error increases for more terms that do not commute $H_{a_1}H_{a_2} \neq H_{a_2}H_{a_1}$ under the time evolution e^{-iH_at}. So, in this first order error approximation, when b increases, the error increases as slowly as n time steps is also increased respectively. This means that at the cost of more matrix multiplication, the accuracy improves. Let's look at the Hubbard model as a simulation example.

C. The Hubbard Model

The Hubbard model describes quantum tunneling as shown in Fig. 2 between neighboring lattice sites and on-site interaction between two fermions of opposite spin [10]. To construct our Hamiltonian, we can imagine a lattice of sites holding up to two electrons that can be spin up or down, hopping from one site to the next, written as

$$\hat{H} = -\sum_{ij}\sum_{\sigma} t_{ij}\left(\hat{a}^{\dagger}_{i\sigma}\hat{a}_{j\sigma} + h.c.\right) + \frac{1}{2}\sum_{ijkl}\sum_{\sigma\acute{\sigma}}\langle ij|v|kl\rangle\hat{a}^{\dagger}_{i\sigma}\hat{a}^{\dagger}_{j\acute{\sigma}}\hat{a}_{l\acute{\sigma}}\hat{a}_{k\sigma}.$$

(2.3.1)

where t_{ij} is the hopping parameter between sites i and j, and v the Coulomb potential. Under the two approximations of restricting hopping to only nearest neighbors and Coulomb interaction to only on-site [11] and adding a chemical potential term μ, we have our second quantized 2-D Hubbard model,

$$\hat{H} = -t\sum_{\langle i,j\rangle,\sigma}\left(\hat{a}^{\dagger}_{i,\sigma}\hat{a}_{j,\sigma} + \hat{a}^{\dagger}_{j,\sigma}\hat{a}_{i,\sigma}\right) + U\sum_{i}\hat{n}_{i,\uparrow}\hat{n}_{i,\downarrow} - \mu\sum_{i,\sigma}\hat{n}_{i,\sigma}.$$

(2.3.2)

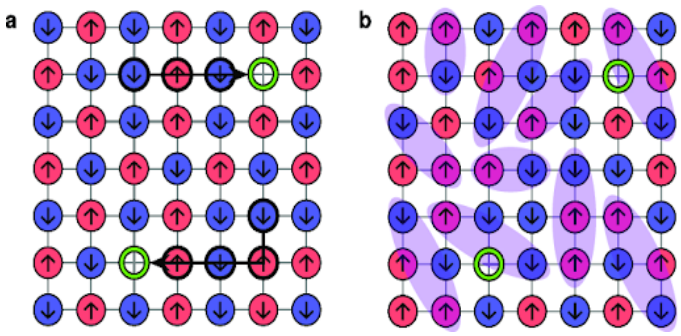

Fig. 2. Illustration of 2-D cartoon quantum tunneling of Hubbard fermions. Source: [12]

The first term represents the kinetic energy of electrons t between neighboring sites $\langle i,j\rangle$ with a $a^{\dagger}_{i,\sigma}$ (creator) creating an election occupation at site i with spin σ and $a_{j,\sigma}$ (annihilator) removing an electron occupation at site j with spin σ respectively. This is followed by the second term U of the potential energy from the on-site Coulomb

repulsion between electrons with opposite spin using the number operator $n_{i,\sigma} = a_{i,\sigma}^{\dagger} a_{i,\sigma}$ for particles with spin σ at site i. The final term μ represents the chemical potential term needed when describing arbitrary electron fillings, not just half-fillings, and affects the total number of particles. When half-filling, the Hubbard model exhibits a Mott metal-insulator transition, taking the system from a metal to an antiferromagnetic insulator. Beyond the Mott transition, the Hubbard model provides insights into diverse broken symmetry phases and phase transitions in strongly correlated systems. The competition parameters $\frac{U}{t}$ between the potential energy term U that favors localization and insulating behavior and the kinetic energy term t that favors delocalization and metallic behavior, alongside the chemical potential μ, give rise to the rich physics of magnetism and superconductivity. This framework of quantum lattices is widely believed to contain the essential ingredients of high-temperature superconductivity because of its delicate competition between the parameters. Doping of charge strips and superconductivity turned by parameter t' has shown the prediction of a Luther-Emery (LE) liquid that demonstrates a close interplay between charge and superconductor correlations [13]. The Hubbard model thus serves as a simplified theoretical foundation for understanding complex interacting quantum materials. However, the model's simplicity is deceptive as it is a mathematically difficult problem to solve due to its many-body nature, with an exact solution only found in the 1D case [14]. A simple example of this model is to consider a system with just two sites,

$$\psi = |n_{1\uparrow} n_{1\downarrow}\rangle |n_{2\uparrow} n_{2\downarrow}\rangle$$

(2.3.3)

having 2^4 possible states depending on the kinetic energy determining how the initial filled each lattice is. This state of an isolated quantum system with n components is represented by a state vector in a Hilbert space of dimension 2^n possible states for n qubits with each qubit able to be in a superposition of 0 and 1. If we want to simulate the dynamics of a quantum system by discretizing time into steps of length Δt at each time step, we need to multiply the state vector by the propagator matrix corresponding to evolving the system for a time Δt. Therefore, the propagator matrix has dimensions $2^n \times 2^n$ with the computation cost scaling as O($2^n \times 2^n$) (see Fig. 3). For most cases, this exponential scaling makes the exact simulation of quantum systems with even $50 - 100$ qubits completely impractical on classical computers due to the exponential growth of the operator that leads to the discussion of using optimization techniques.

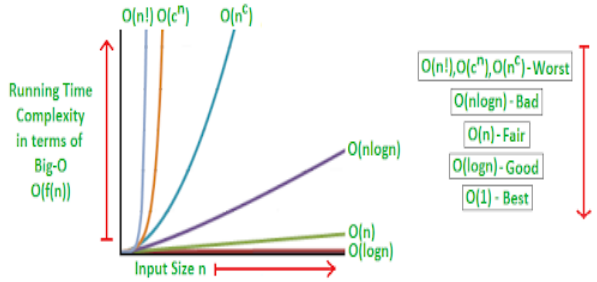

Fig. 3. Demonstration of Big-O complexities. Our goal is to find a low computational cost towards the bottom of the graph. Source: [15]

III. Quantum Computing

A. Jordan Wigner-Mapping

To utilize quantum computing, we must transform the Hamiltonian into a set of operations accessible or understandable to quantum computers. Namely, we can use the simple Jordan Wigner transformation for second quantized Hamiltonians to map occupations to qubit orientations, simulating fermionic systems on qubits and gates by using Pauli-X and -Y to satisfy $a_i^\dagger|0\rangle_i = |1\rangle_i$, $a_i|0\rangle_i = 0$, $a_i^\dagger|1\rangle_i = 0$, and $a_i|1\rangle_i = 0$,

$$a_i^\dagger = \frac{X_i - iY_i}{2}, a_i = \frac{X_i + iY_i}{2}.$$

(3.1.2)

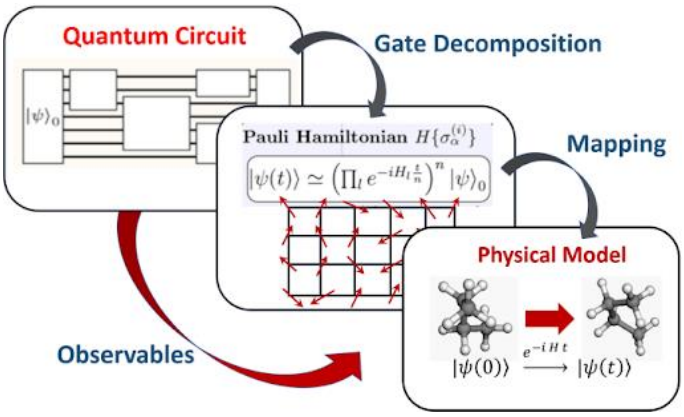

Fig. 4. Illustration of state preparation and mapping of models with the intermediate step of transformation. Source: [16]

For this mapping to capture the antisymmetric characteristics of fermions $a_i^\dagger a_j = -a_j^\dagger a_i$, we must intersperse Pauli-Z to remedy the fact that Pauli Operators do not commute for $XY \neq -YX$, but rather for $XZ = -ZX$ and $YZ = -ZY$:

$$a_1^\dagger = \frac{X_1 - iY_1}{2} \bigotimes 1 \bigotimes 1 \ldots \bigotimes 1$$

$$a_2^\dagger = Z_1 \bigotimes \frac{X_2 - iY_2}{2} \bigotimes 1 \bigotimes 1 \ldots \bigotimes 1$$

$$\vdots$$

$$a_n^\dagger = Z_1 \bigotimes Z_2 \bigotimes \ldots \bigotimes \frac{X_n - Y_n}{2}.$$

(3.1.3)

We get,

$$X_i Y_i = iZ_i$$

$$\therefore n_i = a_i^\dagger a_i = \frac{1 - Z_i}{2}.$$

(3.1.4)

Over the Jordan-Wigner transformation, the Hubbard Hamiltonian in equation (2.3.2) becomes

$$H = -\frac{t}{2}\sum_{\langle i,j\rangle} Z_{j+1:i-1}\left(X_i X_j + Y_i Y_j\right) + \frac{U}{4}\sum_i \left(1 - Z_i^{\uparrow}\right)\left(1 - Z_i^{\downarrow}\right) - \frac{\mu}{2}\sum_{i,\sigma}(1 - Z_i^{\sigma}).$$

(3.1.5)

Other mappings include the Bravyi-Kitaev [17], Parity, and those that can perform more advanced qubit reductions available in quantum computing softwares like Qiskit [18]. From this transformation, we can break down the Hamiltonian into two terms such that $H = H_0 + H_1$, where H_0 is the kinetic term, and H_1 the other two interaction terms. Then, by applying the Trotter-Suzuki decomposition, we get

$$e^{-iHt} = \lim_{n\to\infty}\left(e^{-iH_0\frac{t}{n}}e^{-iH_1\frac{t}{n}}\right)^n.$$

(3.1.6)

This allows us to approximate the time evolution operator by breaking up the evolution into successive n short time steps and repeating while taking larger n improves the accuracy of the approximation. However, we can begin to see the complexities and computational cost arising from higher-order approximations. The natural instinct from this is try to find optimization techniques to circumvent this.

B. Tensor Network Optimization

Now that we can convert our Hamiltonian into quantum computing language, we must face the fundamental problem of optimizing a quantum circuit: efficiently exploiting parameters. We can solve this through differentiable programming given hardware constraints by minimizing the loss function that encodes the problem we want to solve,

$$\alpha_{t+1} = \alpha_t - \eta\nabla_\alpha \mathcal{L}(\alpha, t)$$

(3.2.1)

where α are the parameters, η the learning rate, and $\nabla\mathcal{L}$ the gradient operator on the loss function. This loss function can be largely responsible for the feedback on how well a quantum circuit is performing given a set of parameters, updating it each time for optimization, and is commonly used in machine learning. We can use tensor networks (TN), useful mathematical, graphical representations of multi-dimensional arrays that can store information through tensor nodes in a computation graph. We can significantly enhance optimization by exploiting automatic differentiation in the tensor computation graph. This technique of backpropagation exploits the chain rule of a partial differential to propagate the gradient back from the network output and calculate the

gradient of the weight respectively [19]. With the forward pass being

$$|\psi\rangle = U(\alpha)|\psi_0\rangle$$
$$\langle O\rangle = \langle\psi|O|\psi\rangle$$

(3.2.2)

where $U(\alpha)$ representing a quantum circuit with parameters α acting on initial state $|\psi_0\rangle$ to produce output state $|\psi\rangle$, followed by the expectation value of O, a physical observable that could be represented by a Hermitian operator. We can begin to see the similarities with the original system quantum evolution representation for application,

$$\psi(t) = e^{-iHt}\psi(0).$$

(3.2.3)

From this, the implications of applying backpropagation through the expectation value computation to get gradients with respect to circuit parameters tying to the simulation of quantum systems are obvious. After this forward pass, we can backpropagate (backwards pass) this through the chain rule and see the loss function L depending on the expectation value $\langle O\rangle$,

$$\frac{\partial L}{\partial \alpha} = \frac{\partial L}{\partial \langle O\rangle} \cdot \frac{\partial \langle O\rangle}{\partial \alpha}.$$

(3.2.4)

Representing this in tensor networks through the traversing of data flow as in nodes, we can visually see the chain rule and reverse-mode automatic differentiation [20],

$$\alpha \to W^1 \to W^2 \cdots W^n \to L$$
$$\frac{\partial L}{\partial \alpha} = \frac{\partial L}{\partial W^n}\frac{\partial W}{\partial W^{n-1}}\cdots\frac{\partial W^1}{\partial \alpha}.$$
$$\leftarrow\leftarrow\leftarrow\leftarrow\leftarrow\leftarrow\leftarrow\leftarrow$$

(3.2.5)

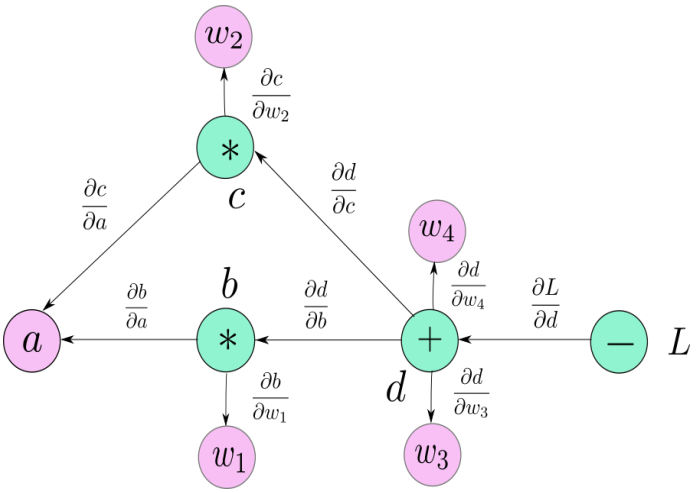

Fig. 5. Illustration of tensor networks through backpropagation to minimize a loss function. The transaction channels can be represented by multiple tensors that can split off respectively depending on intended algorithms that are widely applicable in machine learning. We can also visually see the backpointing technique in storing checkpoints. Our single α tensor network serves as a fundamental representation in equation (3.2.5). Source: [21]

We can also see that the resulting relationship of O relating to physically meaningful observable and arbitrary intermediate tensor W are analogous in the role they play. In order to reduce

memory usage, we can employ the backpointing technique, conceptually like checkpointing (see Fig. 5), where one can simply store the tensor every few steps in the main node. Relying on differentiating through these tensor networks has shown state-of-the-art calculations of specific heat of the Ising model, variational energy, and magnetization of the antiferromagnetic Heisenberg model [20] as an efficient compression of quantum states.

C. Automatically Differentiable Quantum Circuits Tensor Networks (ADQC-TN)

Now that we have discussed optimization, how can we leverage these tensor networks and automatically differentiate them in quantum circuits? We have the tools of TN to simulate quantum models with ample Hilbert space through the backpropagation technique to reduce complexity, but what do we do with them? The simple answer is reframing our thinking to apply to quantum circuits.

Remember our goal is to simulate a model onto quantum circuits,

$$|\psi_{tar}\rangle = U(\alpha)|\psi_{evol}\rangle$$

(3.3.1)

where $|\psi_{tar}\rangle$ is the target state evolved from operation $U(\alpha)$ on $|\psi_{evol}\rangle$ state. We want to minimize the error between the target and evolved state,

$$F = -\frac{1}{N}\ln\ |\langle\psi_{tar}|U(\alpha)|\psi_{evol}\rangle|$$

(3.3.2)

where F is the negative logarithmic fidelity that quantifies the closeness between the $|\psi_{tar}\rangle$ and $|\psi_{evol}\rangle$ states over operation $U(\alpha)$. Therefore, we just need to find a particular set of unitary gates or operations on δ^* parameters that minimizes F, our respective loss function. This can be done by updating the gates towards the opposite direction of their gradients and integrated with TN [22],

$$\delta^\bullet = U\Lambda V^\dagger$$
$$\delta^\bullet \leftarrow \delta_0^\bullet - \eta\frac{\partial F}{\partial \delta^\bullet}$$

(3.3.3)

where $\delta^\cdot$ projects to a set of unitary gate δ^*under $Tr(\delta^\cdot\delta)$ to be maximized from constraint δ. This is then fed into the backpropagation integrated with TN. All we have to do after is apply our desired Hamiltonian into this system and compare it with traditional methods.

IV. Applications

A. Variational Quantum Eigensolver (VQE)

One current, powerful application is the Variational Quantum Eigensolver (VQE) [23], a hybrid quantum-classical algorithm to find a Hamiltonian's minimum eigenvalue and eigenstate. VQE aims to minimize energy by optimizing a set

of parameters α for a chosen ansatz quantum circuit. The circuit has two purposes: prepare the ansatz wave function $|\psi(\alpha)\rangle$ and measure the expectation value $\langle\psi(\alpha)|H|\psi(\alpha)\rangle$ which gives the energy. A classical optimizer uses this measurement to adjust α to lower the energy toward the true ground state. Starting with initial parameters $|\psi(\alpha_0)\rangle = e^{-iH|\psi(\alpha_0)\rangle}|\psi_1\rangle$, the ansatz circuit prepares the trial wavefunction, where $|\psi_1\rangle$ is a simple starting state. In variational quantum algorithms like VQE, the loss function is typically the expectation value and minimizing the loss, we find the ground state energy. Measuring the expectation energy $E(\alpha_0)$ provides an upper bound on the true ground state energy:

$$E(\alpha_0) = \langle\psi(\alpha_0)|H|\psi(\alpha_0)\rangle \geq E_0 \quad (4.1.1)$$

The classical optimizer then generates new parameters α_1, using $E(\alpha_0)$ to guide it closer to the minimum. This optimization loop of state preparation, measurement, and classical processing repeats until converging on an ansatz $|\psi(\alpha^*)\rangle$ that minimizes $E(\alpha) \approx E_0$ .This hybrid quantum algorithm is useful for quantum chemistry and optimization problems that looks analogous to the issues we can solve with ADTNAQCs. VQE implementations on existing quantum hardware are limited by qubit error rates, the number of qubits available, and the allowable gate depth [24], and new techniques implementing better optimization methods and representation of these compact spaces like ADQCs and TNs are necessary for the continued further application of quantum computing despite quantum hardware restrictions as shown in Fig. 6.

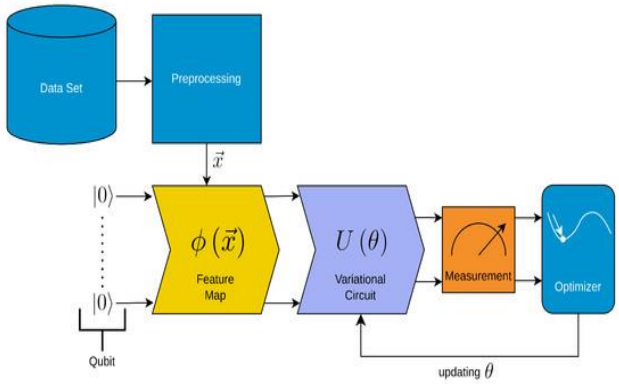

Fig. 6. Illustration of the application of quantum computing for manipulating data sets. We can visually see here the usage of optimization techniques and programming for a desired output, dependent on the algorithm. Source: [25]

B. Modeling

The advent of Automatically Differentiable Quantum Circuits (ADQCs) allows for preparing many-qubit target quantum states by optimizing gates through differentiable programming via backpropagation. ADQCs introduce unconstrained, differentiable latent gates projected to unitary gates satisfying quantum constraints, and therefore optimizing these latent gates layer-by-layer yields efficient state preparation using ADQCs, obtains low fidelities, and can reduce matrix product state (MPS) representations with a compression ratio of $r \sim O(10^{-3})$ [22]. Likewise, tensor networks are a

powerful mathematical tool that can provide a compact representation of high-dimensional quantum states and quantum many-body physics [26, 27] by formulating tensor networks as differentiable computation graphs. A key advantage is directly computing tensor network output gradients, enabling the evaluation of observables. This yields optimal performance for infinite 2D tensor networks and finding the ground states of lattice models [20]. When integrated with ADQCs, TN can demonstrate intelligent quantum circuit construction combining machine learning techniques, a promising approach for training and optimizing quantum systems using modern differentiable programming. Such applications can already be seen in the development of built-in ADQC programs such as Yao [28], which can optimize a variational circuit with 10,000 layers using reverse-mode AD and constructing 20 sites Heisenberg Hamiltonian in approximately 5 seconds. By utilizing this paper's framework and quantum computing softwares, one can model and optimize their desired models.

V. Future Research

There are two main sectors of future research from ADQC-TN optimization on models: other physical models and applicable optimization. Future research into non-local Hamiltonians and other respective techniques and approximations extends to models beyond the 2D Hubbard model. Similarly, higher dimensions and more complex, accurate models are good starting points. Having a framework to see which methods apply to what, so that it can be tailored for real applications, would be branching out. The frame-work proposed for optimizing parameters in this paper relies mainly on quantum computing software. The work in more advanced and enhanced libraries and programing could determine usability, automation, utilization, and integration into real-world systems. New loss functions, flexible architectures, and alternative quiet mappings, in combination with more methodological and pragmatic engineering fronts, will aid in the discovery and application of accurate, performant quantum models. Furthermore, future research can also be seen in real-world applications of the techniques discussed in this paper. The integration of machine learning methods with automatic differentiation and tensor networks using quantum circuits can help in model tuning, complex algorithms, and optimization in the fields of artificial intelligence and quantum simulation. This can already be seen in [29], where hybrid neural network weight tensorization can be represented by a tensor-train data format to compress parameters that developed an energy-efficient machine learning accelerator. More motivation can be seen in [30] for the intellectual combination of tensor networks and neural networks for potential applications in information fusion and more specifically, what neural networks has already seen development like natural language processing and robotics can be improved.

VI. Discussion

Further technical examination of this ADQC-TN approach can reveal several aspects of the modeled Hamiltonian and optimization as a field. Firstly, alternative transformations such as the Bravyi-Kitaev algorithm offer potential for more efficient reductions that could reduce overall circuit depth, and more detailed analysis quantitatively comparing circuit complexity under different fermionic qubit mappings could in advantageous transformations for a given model. Secondly, this tensor network architecture design space is extensive and dependent heavily on depth, entanglement, connectivity, and the desired model. Architectures balancing expressiveness and trainability may be particularly well-suited, but systematic evaluation of tensor network configurations' usage will be essential in creating performant models applicable beyond physical domains. Thirdly, while mathematically convenient, reliance on fidelity as the sole optimization metric is not ideal and is best alongside incorporating alternative domain-specific loss functions for training towards precise experimental objectives under a given objective. Fourthly, while simplified, differentiable programming is still in its infancy and requires more development on its widespread software, meaning modeling difficulty and complexity will depend similarly on the software used.

This paper lays a framework for optimizing the 2-D Hubbard Hamiltonian model implementable in quantum programming languages. For other models alike, it can be done in three simplistic steps: This paper lays out a framework for optimizing the 2-D Hubbard Hamiltonian model implementable in quantum programming languages. For other models alike, it can be done in three simplistic steps:

1. Integrate our desired Hamiltonian model and utilize any unique simplifications and/or approximations in equation (2.3.2).
2. Transform that Hamiltonian to qubit (QC) language via the desired transformation seen in equation (3.1.5).
3. Apply an optimization method to find the desired output and/or parameters of the Hamiltonian model using equation (3.3.3).

VII. Conclusion

Pursuing a simplified framework in leveraging optimization and programming techniques that can be used to transition from theoretical quantum physics to applied quantum computing is essential in further understanding the promising use of new quantum simulation methods and integration of quantum hybrid methods over just classical simulations. This paper has proposed a framework for optimizing parameters of the 2D Hubbard model by integrating automatically differentiable quantum circuits with tensor networks. We reviewed techniques for gradient-based optimization via backpropagation through ADQCs to enable efficient tuning of quantum circuits. The Hubbard model was transformed into qubit operators using the Jordan-Wigner

transformation to implement on a quantum circuit. By minimizing the fidelity error between target and evolved states, the ADQC-TN framework can optimize model parameters to match experimental measurements. The proposed ADQC-TN optimization framework for simulating the 2D Hubbard model offers a path forward despite hardware constraints. As quantum computing matures, techniques like differentiable programming will help implement precise quantum models to revolutionize our understanding of complex quantum materials. New research into complex quantum systems and their applications in quantum computing is necessary for the diversity of software and paradigms. Techniques alike leverage what is already known but make it better and more applicable to reduce problems from new growing demands like computational complexity costly simulations, and is scalable, to name a few. By outlining a methodology of the potential application of automatic differentiation using tensor networks in models like the Hubbard model, this paper summarizes the problems faced and current, novel solutions to it. This can go beyond simulating just local models and be more applicable to other parameters in increasing complexity. As the times of our technology try to catch up, looking from a different perspective and way of thinking will help in the endeavor to make sense of and apply these innovations despite the limitations of our time.

References

[1] D. Wecker, M. B. Hastings, N. Wiebe, B. K. Clark, C. Nayak, and M. Troyer, "Solving strongly correlated electron models on a quantum computer," Phys. Rev. A, vol. 92, no. 6, pp. 062318, Dec. 2015, doi: 10.1103/PhysRevA.92.062318.

[2] C. W. Bauer et al., "Quantum Simulation for High Energy Physics," arXiv, Apr. 07, 2022, doi: 10.48550/arXiv.2204.03381.

[3] J. Preskill, "Quantum Computing in the NISQ era and beyond," Quantum, vol. 2, pp. 79, Aug. 2018, doi: 10.22331/q-2018-08-06-79.

[4] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, and M. Troyer, "Elucidating reaction mechanisms on quantum computers," Proc. Natl. Acad. Sci. U.S.A., vol. 114, no. 29, pp. 7555–7560, Jul. 2017, doi: 10.1073/pnas.1619152114.

[5] J. Fraxanet, T. Salamon, and M. Lewenstein, "The Coming Decades of Quantum Simulation," 2022, doi: 10.48550/ARXIV.2204.08905.

[6] "Quantum Optics and Quantum Many-Body Systems." Quantum Computing. Available: https://qoqms.phys.strath.ac.uk/index.html. Accessed: August 24, 2023.

[7] C. W. Bauer et al., "Quantum Simulation for High Energy Physics," PRX Quantum, vol. 4, no. 2, p. 027001, May 2023, doi: 10.1103/PRXQuantum.4.027001.

[8] Q. Liu, "Comparisons of Conventional Computing and Quantum Computing Approaches," HSET, vol. 38, pp. 502–507, Mar. 2023, doi: 10.54097/hset.v38i.5875.

[9] O. Kyriienko, A. E. Paine, and V. E. Elfving, "Protocols for Trainable and Differentiable Quantum Generative Modelling," arXiv, Feb. 16, 2022. Accessed: Aug. 21, 2023. doi. 10.48550/arXiv.2202.08253.

[10] E. Cocchi et al., "Equation of State of the Two-Dimensional Hubbard Model," Phys. Rev. Lett., vol. 116, no. 17, p. 175301, Apr. 2016, doi: 10.1103/PhysRevLett.116.175301.

[11] V. Celebonovic, "The two-dimensional Hubbard model: a theoretical tool for molecular electronics," J. Phys.: Conf. Ser., vol. 253, p. 012004, Nov. 2010, doi: 10.1088/1742-6596/253/1/012004.

[12] C. Miles et al., "Correlator convolutional neural networks as an interpretable architecture for image-like quantum matter data," Nat Commun, vol. 12, no. 1, p. 3905, Jun. 2021, doi: 10.1038/s41467-021-23952-w.

[13] H.-C. Jiang and T. P. Devereaux, "Superconductivity in the doped Hubbard model and its interplay with next-nearest hopping t’," Science, vol. 365, no. 6460, pp. 1424–1428, Sep. 2019, doi: 10.1126/science.aal5304.

[14] X.-W. Guan, "Algebraic Bethe ansatz for the one-dimensional Hubbard model with open boundaries," J. Phys. A: Math. Gen., vol. 33, no. 30, pp. 5391–5404, Aug. 2000, doi: 10.1088/0305-4470/33/30/309.

[15] "Analysis of Algorithms — Big-O Analysis." GeeksforGeeks. Available: https://www.geeksforgeeks.org/analysis-algorithms-big-o-analysis/article-meta-div. Accessed: August 24, 2023.

[16] J. W. Z. Lau, K. H. Lim, H. Shrotriya, and L. C. Kwek, "NISQ computing: where are we and where do we go?," AAPPS Bull., vol. 32, no. 1, p. 27, Sep. 2022, doi: 10.1007/s43673-022-00058-z.

[17] A. Tranter, P. J. Love, F. Mintert, and P. V. Coveney, "A Comparison of the Bravyi–Kitaev and Jordan–Wigner Transformations for the Quantum Simulation of Quantum Chemistry," J. Chem. Theory Comput., vol. 14, no. 11, pp. 5617–5630, Nov. 2018, doi: 10.1021/acs.jctc.8b00450.

[18] M. Treinish, "Qiskit/qiskit-metapackage: Qiskit 0.44.0." Zenodo, Jul. 27, 2023, doi: 10.5281/ZENODO.2573505.

[19] M. Watabe, K. Shiba, M. Sogabe, K. Sakamoto, and T. Sogabe, "Quantum Circuit Parameters Learning with Gradient Descent Using Backpropagation," 2019, doi: 10.48550/ARXIV.1910.14266.

[20] H.-J. Liao, J.-G. Liu, L. Wang, and T. Xiang, "Differentiable Programming Tensor Networks," Phys. Rev. X, vol. 9, no. 3, p. 031041, Sep. 2019, doi: 10.1103/PhysRevX.9.031041.

[21] A. Kathuria, "PyTorch 101, Part 1: Understanding Graphs, Automatic Differentiation and Autograd," Paperspace Blog. Available: https://blog.paperspace.com/pytorch-101-understanding-graphs-and-automatic-differentiation/. Accessed: August 24, 2023.

[22] P.-F. Zhou, R. Hong, and S.-J. Ran, "Automatically Differentiable Quantum Circuit for Many-qubit State Preparation," 2021, doi: 10.48550/ARXIV.2104.14949.

[23] J. Tilly et al., "The Variational Quantum Eigensolver: a review of methods and best practices," 2021, doi: 10.48550/ARXIV.2111.05176.

[24] J. M. Clary, E. B. Jones, D. Vigil-Fowler, C. Chang, and P. Graf, "Exploring the scaling limitations of the variational quantum eigensolver with the bond dissociation of hydride diatomic molecules," Int J of Quantum Chemistry, vol. 123, no. 11, p. e27097, Jun. 2023, doi: 10.1002/qua.27097.

[25] S. Raubitzek and K. Mallinger, "On the Applicability of Quantum Machine Learning," Entropy, vol. 25, no. 7, p. 992, Jun. 2023, doi: 10.3390/e25070992.

[26] R. Orus, "A Practical Introduction to Tensor Networks: Matrix Product States and Projected Entangled Pair States," 2013, doi: 10.48550/ARXIV.1306.2164.

[27] R. Orus, "Tensor networks for complex quantum systems," 2018, doi: 10.48550/ARXIV.1812.04011.

[28] X.-Z. Luo, J.-G. Liu, P. Zhang, and L. Wang, "Yao.jl: Extensible, Efficient Framework for Quantum Algorithm Design," 2019, doi: 10.48550/ARXIV.1912.10877.

[29] H. Huang, L. Ni, and H. Yu, “LTNN: An energy efficient machine learning accelerator on 3D CMOSRRAM for layer-wise tensorized neural network,” in 2017 30th IEEE International System-on-Chip Conference (SOCC), Munich: IEEE, Sep. 2017, pp. 280–285. doi: 10.1109/SOCC.2017.8226058.

[30] M. Wang, Y. Pan, Z. Xu, X. Yang, G. Li, and A. Cichocki, “Tensor Networks Meet Neural Net-works: A Survey and Future Perspectives,” 2023, doi: 10.48550/ARXIV.2302.09019.