Simulating Physics with Quantum Computers

Team: Quantum is Better!!

Members: SungBin Lee, Gyunghun Kim, Kihyeon Kim, Hosung Lee, Dohwon Lee

Abstract

In this project, we implemented quantum simulator that simulates interacting system that consists of 1D and 2D spin sites. We simulated for all the given models, including XX, XXZ, and disordered XX models, and for all the given initial states, represented by Neel and domain wall states. Also, we simulated the interaction schemes for 1D next neighborhood model and 2D nearest neighborhood model which were given as the advanced task. To validate the result, we also developed classical algorithm to simulate the same system, which are capable up to N=16 sites. Furthermore, we pointed out the inherent limitation of the noisy quantum simulation, and implemented a VFF circuit, which uses diagonalization, and is a potential solution to the problem.

1. Introduction

Many physical simulations could be, in nature, perform better by quantum computing compared to classical ones we are familiar with. This is due to the characteristics of quantum computing which can naturally embrace the fundamental principles of physics like uncertainty principle, Copenhagen interpretation, quantum entanglement, and measurement problem through qubits which, unlike bits in classical computers, represents the superposed states. In Quantum Hackathon 2021 Korea, we utilized such property to simulate the time evolution dynamics of a closed quantum system.

In the Hackathon, we simulate the well-known NN interaction (nearest neighbor interaction) XXZ model with the following Hamiltonian in eq 1. Such closed quantum system is one of the most fundamental systems, with transverse Ising model, in quantum statistical mechanics which exhibits quantum phase transition.

$$H = -\mathcal{J}\sum_{\langle ij\rangle}\left(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y\right) + U\sum_{\langle ij\rangle}\sigma_i^z\sigma_j^z + \sum_i h_j\sigma_j^z \tag{1}$$

It is worth noting that the Hamiltonian in eq 1 does not have a periodic boundary condition unlike most closed quantum system in computational statistical mechanics.

We also expand the scope of the problem by including the NNN interaction (next-nearest neighbor interaction) in the first Hamiltonian.

$$H = -\mathcal{J}\sum_{\langle\langle ij\rangle\rangle}\left(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y\right) + U\sum_{\langle\langle ij\rangle\rangle}\sigma_i^z\sigma_j^z + \sum_i h_j\sigma_j^z \tag{2}$$

To evaluate the time evolution dynamics of a closed quantum system, one needs to apply the imaginary exponential Hamiltonian operator on an initial state.

$$U(t) = e^{-i\hat{H}t} \rightarrow |\psi(t)\rangle = U(t)|\psi(0)\rangle = e^{-i\hat{H}t}|\psi(0)\rangle \tag{3}$$

In classical computers, we indirectly demonstrate such evolution algorithm by computing the eigenstate and eigenvalue of the Hamiltonian.

$$\hat{H}|E_n\rangle = E_n|E_n\rangle \rightarrow e^{-i\hat{H}t}|E_n\rangle = e^{-iE_nt}|E_n\rangle \rightarrow e^{-i\hat{H}t}|\psi(0)\rangle = \sum_n e^{-iE_nt}|E_n\rangle\langle E_n|\psi(0)\rangle \quad (4)$$

However, this procedure requires to calculate the entire eigenvalues and eigenvectors of the Hamiltonian. In quantum statistical physics where the Hamiltonian matrix is exponentially proportional to the size of the system, this approach is very hard to finish within reasonable amount of time and memory.

On the contrary, quantum computing enable us to directly compute the complex exponential Hamiltonian with quantum logic gates on quantum circuits. The number of gates is linear to the system size, thus making it much more efficient in large simulations with dozens of qubits.

In this report, we will first introduce the structure of classical and quantum algorithms utilized to compute the time evolution dynamics of a system. Then we will compare the results between the two methods run on classical computer and qasm_simulator in Qiskit.Aer, respectively. Finally, we will discuss several points on how to improve the quantum results on fake quantum machines in the discussion.

2. Classical Computation

One of the most import motivations of the simulation of physical system in quantum computers is that the classical algorithm has obvious limitations. Both space and time complexity are $O(2^N)$, where N is the number of sites to simulate. Thus, if N exceeds 20, it is almost impossible to simulate with a normal laptop with 4GB of RAM and a single CPU.

However, to validate the results of our quantum circuits representing the Hamiltonians and propagators, reliable classical simulation results are required. We performed simulations with classical algorithm directly with the state vectors and propagators, up to $N = 16$ system which is equivalent to the system with 4 rows and 4 columns of sites.

A state vector to represent the system with N sites requires about $D2^N$ bytes to store all the information, where D is size of a floating-point complex element. For $N = 16$, it is only about several hundreds of kilobytes. However, the Hamiltonian and corresponding propagators have dimensions of 2^{2N}. Moreover, the computation time became too long for our laptops to handle. Several trials showed that our laptops are only capable of simulating $N = 12$ systems.

Thus, we have adopted the Lanczos algorithm to efficiently obtain propagators without storing all of elements of the Hamiltonian explicitly. We referred to the following source.[1] Also, we note that the implemented algorithm of the quspin package is used.[2]

The algorithm uses matrix diagonalization to approximately represent a linear operator and obtain analytic functions of that operator. For diagonalization of a linear operator A, not the explicit form of the matrix, but only the function that returns Ax for any vector x is required. Using this so-called matvec product function, the operator is diagonalized up to the number of eigenvectors which is previously determined and related to precision of the approximated result.

The algorithm finally constructs approximate eigenspace where the operator is diagonalized with these much smaller number of eigenvectors. Computations of operator, for example $\exp(-iHt)$ which used repeatedly to obtain propagators, are then done with this approximately diagonalized operator. Our experiments showed that 50 is enough for the number of eigenvectors to use and we used this number for our classical simulations. The figure below shows that how the approximate results follow the exact one when we increased dimension of the eigenspace.

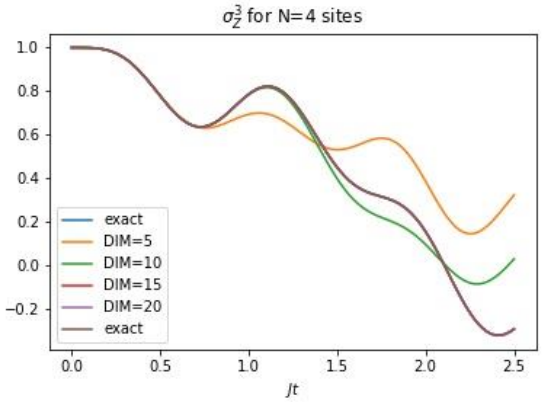

Figure 1 Simulation results of different approximating eigenspace dimensions.

In virtue of this algorithm, we only needed to implement the matvec product function of Hamiltonian. The most important thing was not to use explicit matrix form of the Hamiltonian, because then computational reduction made by the Lanczos algorithms would become no use at all. Thus, we used tensordot function of the numpy package to apply σ_x, σ_y, σ_z to each site, which only took $O(2^N)$ of time complexity. Please refer to the code for further information.

3. Quantum Circuit

In this section, we demonstrate the quantum circuit which implements the unitary time evolution operator for Hamiltonian in eq 1 and 2. Our design is based on quantum logic gates which every quantum computing software like provides.

The entire gate should represent the following unitary operation.

$$U(\delta t) = e^{-i\hat{H}\delta t} = e^{i\mathcal{J}\delta t\left[\sum_j\left(\sigma_j^x\otimes\sigma_{j+1}^x+\sigma_j^y\otimes\sigma_{j+1}^y\right)-\frac{U}{\mathcal{J}}\sum_j\sigma_j^z\otimes\sigma_{j+1}^z-\sum_j\frac{h_j}{\mathcal{J}}\sigma_j^z\right]} \tag{5}$$

Since it is impossible to apply two adjacent two-qubit size quantum gates at the same time, we split the quantum gate into two parts: even and odd ones.

$$U_1(\delta t) = e^{i\mathcal{J}\delta t\left[\sum_{j:odd}\left(\sigma_j^x\otimes\sigma_{j+1}^x+\sigma_j^y\otimes\sigma_{j+1}^y\right)-\frac{U}{\mathcal{J}}\sum_{j:odd}\sigma_j^z\otimes\sigma_{j+1}^z\right]} \tag{6}$$

$$U_2(\delta t) = e^{i\mathcal{J}\delta t\left[\sum_{j:even}\left(\sigma_j^x\otimes\sigma_{j+1}^x+\sigma_j^y\otimes\sigma_{j+1}^y\right)-\frac{U}{\mathcal{J}}\sum_{j:even}\sigma_j^z\otimes\sigma_{j+1}^z\right]} \tag{7}$$

$$U_3(\delta t) = e^{i\mathcal{J}\delta t\left[-\sum_j\frac{h_j}{\mathcal{J}}\sigma_j^z\right]} \tag{8}$$

We can easily copy the single-size qubit gate operator with the Rz rotation-Z gate. However, both two-qubit size gates are more complex. We refer to the paper published in the Physical Review A in 2004 and discover the possible two-qubit size operation based on quantum logic gates: three CNOTs, three Rz gates, and two Ry gates.[3]

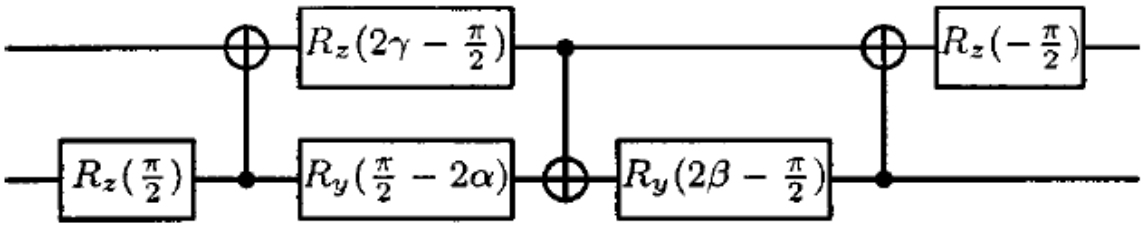

Figure 2. The two-qubit size quantum circuit which implements the nearest neighbor interaction between qubits.

We utilized the symmetric trotter decomposition to minimize the error. According to the paper published in npj Quantum Information in 2019, the basic trotter decomposition has error proportional to the square of δt while the symmetric one is cube.[4]

$$U_1(\delta t)U_2(\delta t)U_3(\delta t) \rightarrow O(\delta t^2) \tag{9}$$

$$U_3\,(\delta t/2)U_2(\delta t/2)U_1(\delta t/2)U_2(\delta t/2)U_1(\delta t/2) \tag{10}$$

The actual circuit utilized for the quantum computing is shown in fig 3.

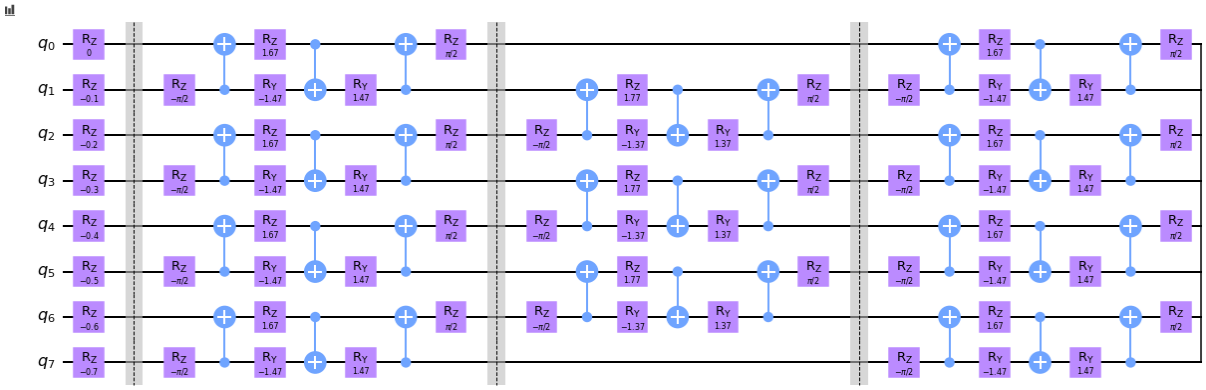

Figure 3. The actual 8-qubit circuit in the quantum algorithm. We first apply the one-qubit size quantum gate and move onto even and odd two-qubit size gates.

4. Local Magnetization

We compared the local magnetization of various states with different initial state, Hamiltonian structure, and computation method. Fig 4 shows that even with the same Hamiltonian, the time evolution dynamics can be different with different initial conditions. It is worth noting that the Neel state reaches the opposite spin more quickly than the domain wall. This is due to the characteristics of nearest neighbor interaction: the opposite spin tends to flip more quickly.

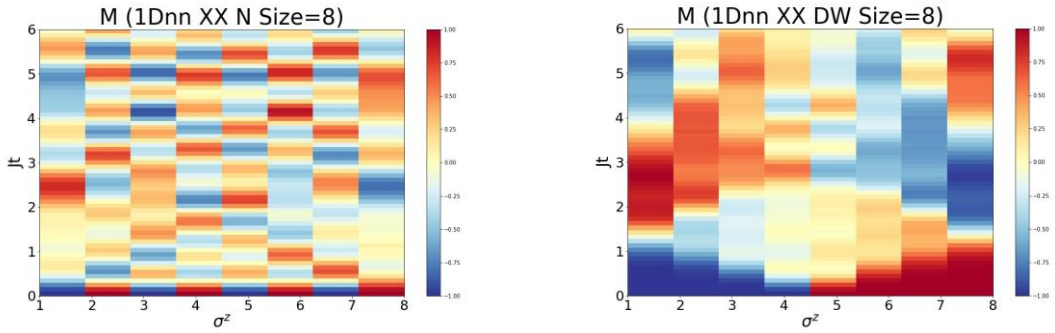

Figure 4. Simulated local magnetization of 1D nearest neighborhood model with XX model for different initial states (Left: Neel state, Right: Domain wall state)

We also compared the similar system with one having an extra next-nearest interaction while the other don't. From fig 5, it is obvious that such extra interaction between qubits maintain the domain wall much more strongly in contrast with system only having just one type of interaction.

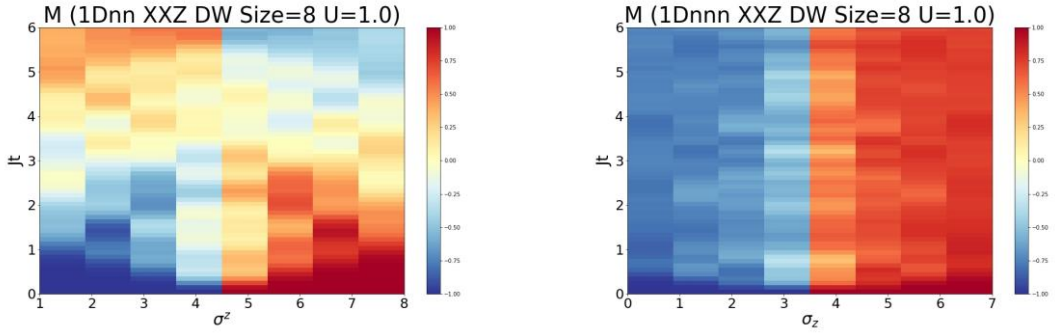

Figure 5. Simulated local magnetization of XXZ model with different interaction schemes (Left: 1D nearest neighborhood, Right: 1D next-nearest neighborhood model)

Finally, we compared the classically computed results displayed on left with the quantum version on right. Although the result estimated by the qasm_simulator seems to be coherent at the beginning, it is obvious that it started to lose fidelity as time evolves.

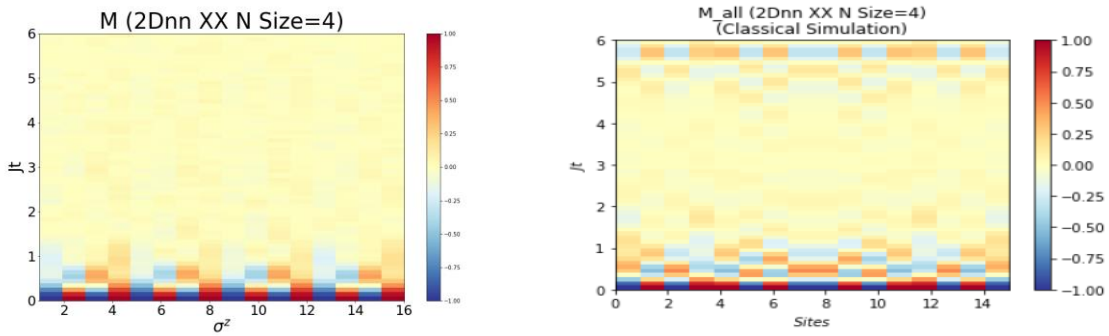

Figure 6. Simulated local magnetization of 2D nearest neighborhood model with XX model using different simulators (Left: Quantum Simulator, Right: Classical Algorithm)

5. N half

In this section, we compare the Nhalf value of various physical systems with different initial conditions, Hamiltonian structure, and computation methods. The following figure represents how Nhalf changes over time.

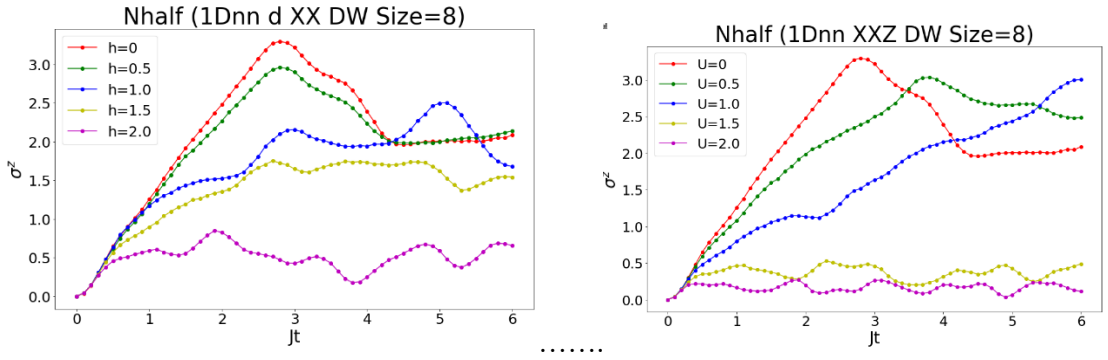

Figure 7. Simulated Nhalf of 1D nearest neighborhood model with different Hamiltonian configurations (Left: XX model, Right: XXZ model)

An extra interaction was added to the Hamiltonian in the right part of the following image. The Nhalf is much lower on the right part in contrast of the left one, indicating that the domain wall remains on the right side.

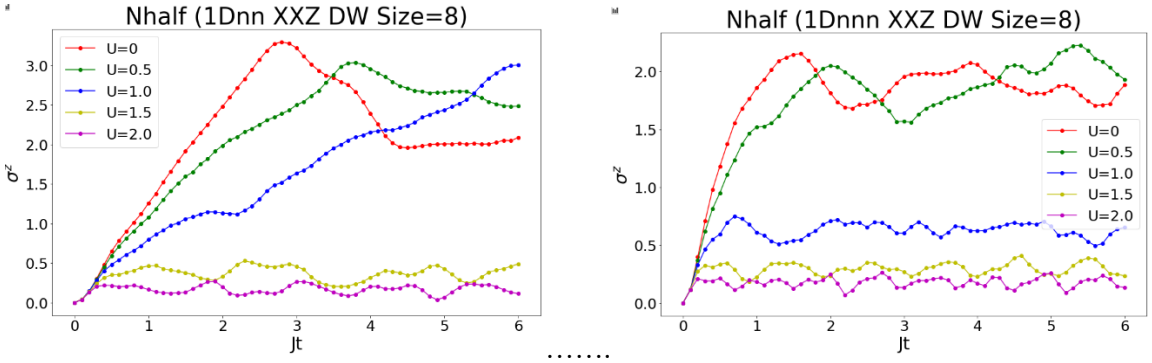

Figure 8. Simulated Nhalf of XXZ model with different interaction schemes (Left: 1D nearest neighborhood, Right: 1D next-nearest neighborhood model)

Then we compared the classically computed results of 2D nearest neighbor interaction XXZ model with domain wall initial condition with quantum computers. Similar with before, the qasm_simulator fails to track the result after certain threshold where the circuit depth is too much for it to handle.

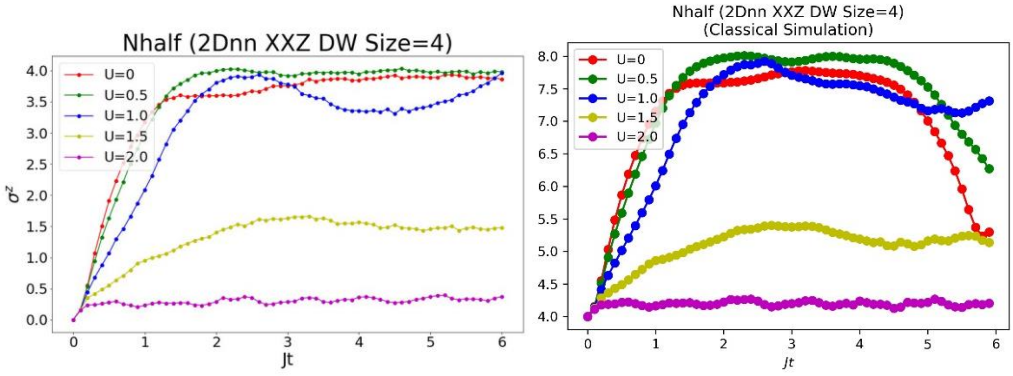

Figure 9. Simulated Nhalf of 2D nearest neighborhood model with XX model using different simulators (Left: Quantum Simulator, Right: Classical Algorithm)

6. Correlation Function

Finally, we computed the correlation function of the closed quantum system in this part of the report. Figure 10 shows the correlation function of the 1D nearest neighbor interacting XX model with different initial conditions. It is obvious that the correlation is much more concrete in the Neel state.

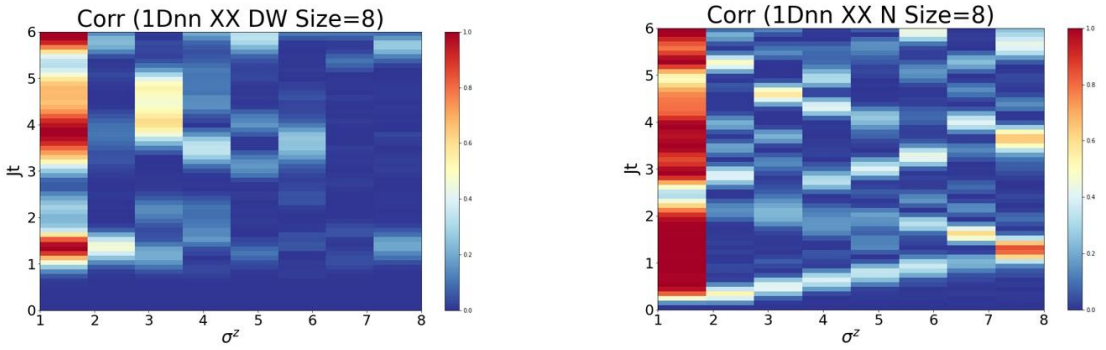

Figure 10. Correlation function with the first site of 1D nearest neighborhood model with different initial states (Left: Domain wall state, Right: Neel state)

The below figure displays the correlation function as a function of time for XX model with different Hamiltonian structure. The right part contains an extra NNN interaction while the left part does't, contributing the difference shown in the figure.

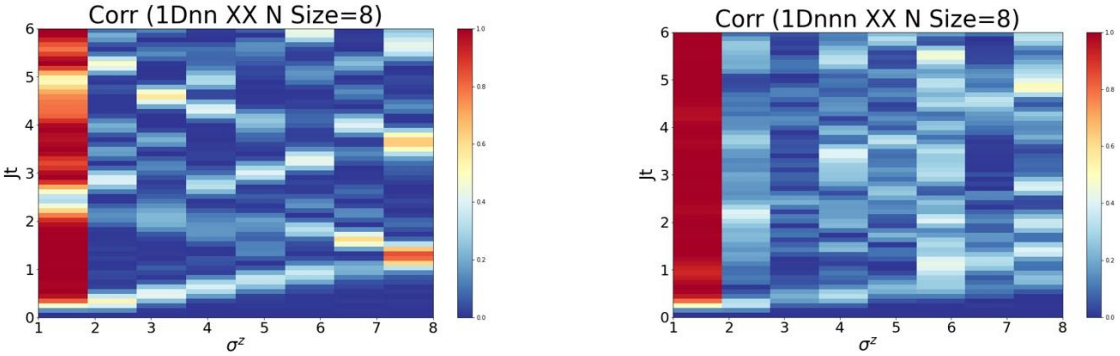

Figure 11. Correlation function with the first site of XX model with different interaction schemes (Left: 1D nearest neighborhood, Right: 1D next-nearest neighborhood model)

The left and right part of fig 12, shows the correlation function estimated by qasm_simulator and classical computers, respectively. It is obvious that qasm_simulator fails to achieve the level of current classical algorithms at extreme situations.

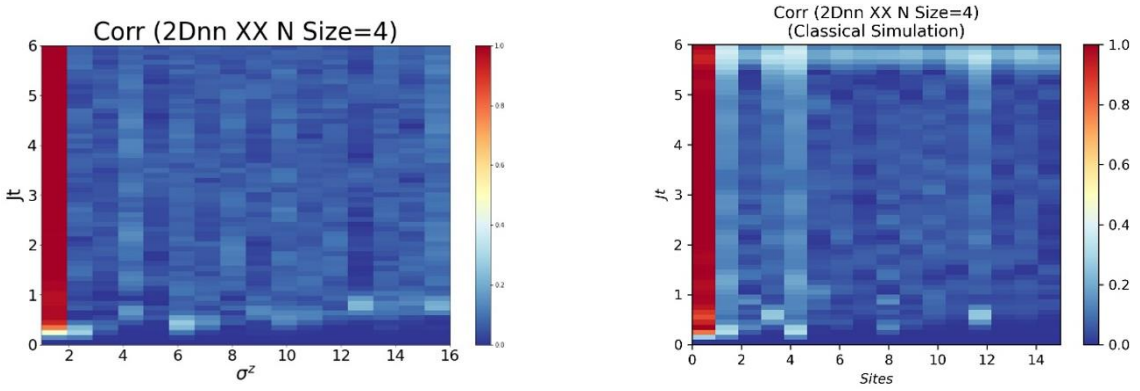

Figure 12. Correlation function with the first site of 2D nearest neighborhood model with XX model using different simulators (Left: Quantum Simulator, Right: Classical Algorithm)

7. Discussion on Limitation of Noisy Quantum Simulation

The quantum computers look promising for the quantum simulation. However, the decoherence and limitation of circuit depth are crucial limitations of those computers in NISQ era. Although there are several methods to correct the errors, they are out of the scope of this project.

Thus, we assumed noisy quantum computers and executed our circuits on those computers. The noise is related to maximum circuit depth that a quantum computer can reliably execute. The depth, in our case, is proportional to steps of propagators. The available maximum depth then restricts the duration of the simulation. Also, if we increase the precision of the simulation, which means more steps for the same duration, then the depth of the total propagator circuit also proportionally increases.

In classical algorithms, these kinds of problems do not exist at all. All the duration time and precision of the simulation can be obtained in cost of computational time. In other words, if we have infinite amount of time, we can simulate the system precisely for infinite time length.

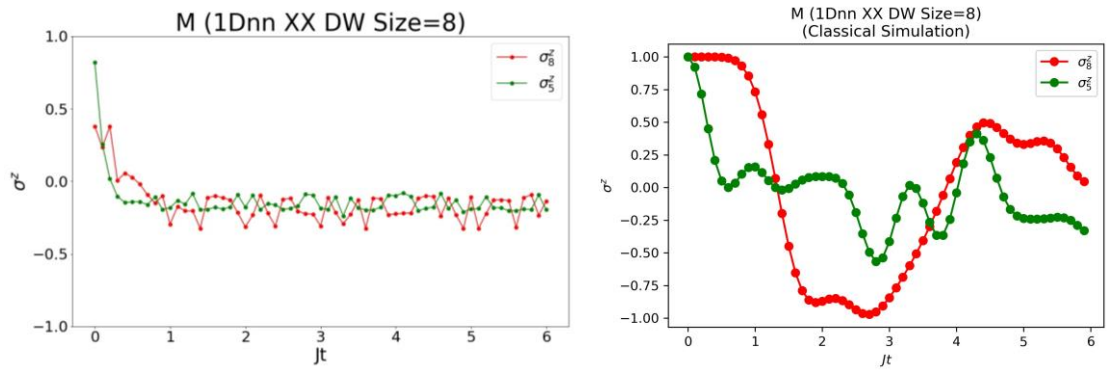

Figure 13. Comparison of simulation on real machine and ideal result

The figure above compares the simulation on noisy real machine (FakePoughKeepsie backend) and the classically simulated ideal result. Unfortunately, the depth limitation by noise on real computers were too tight for our deep propagator circuits. We could observe that the decoherence makes everything to average within ten steps of the state propagation.

A potential solution to this problem is variational fast forwarding,[5] namely the quantum diagonalization. We applied this method to a small system consists of 2 sites and proved that the simulation result became much reliable.

Imagine a propagator $U(dt)$ and assume the diagonalized form of the circuit, $W(\boldsymbol{\theta})A(dt\boldsymbol{\gamma})W^{\dagger}(\boldsymbol{\theta})$. W and A represent quantum circuit parameterized with $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$, respectively. One important requirement is that $A(dt\boldsymbol{\gamma})$ is diagonal and consists of commutative gates. It allows easy calculation of $[A(dt\boldsymbol{\gamma})]^N = A(Ndt\boldsymbol{\gamma})$. The point of this method is to diagonalize the given propagator by variationally learning the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\gamma}$, and to exploit the special commutative property.

There are various ways to construct W and A. We followed the method given in this reference.[6] The figure below shows our ansatz. The circuit before the first barrier is W, and the one between two barriers is A. $\boldsymbol{\theta}$ is a vector of length 18 and $\boldsymbol{\gamma}$ is a vector of length 3. The parameters are realized as rotational gates as shown in the figure.

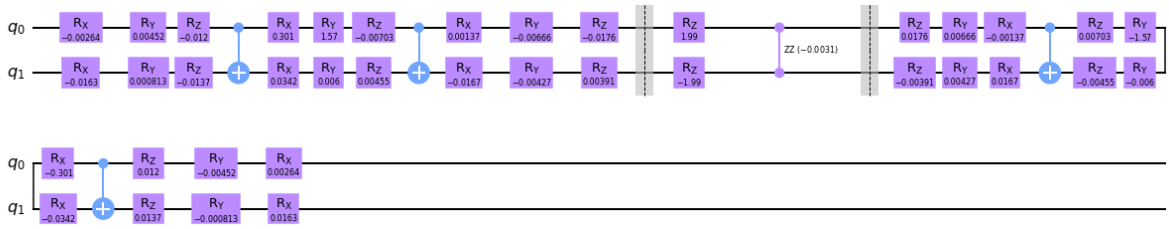

Figure 14. Diagonalized propagator with 21 found parameters

We found the parameters following method proposed in the reference.[6] We adopted the cost function and calculated gradient following that reference. Please refer to the source code for more detailed form of cost functions and learning algorithms. To find all the parameters correctly enough to simulate, 60 iterations of learning were required. Each iteration runs the circuit about 4 times of number of all parameters, which is about a hundred. We diagonalized $U(1)$ for convenience, and applied $W(\boldsymbol{\theta})A(t\boldsymbol{\gamma})W^{\dagger}(\boldsymbol{\theta})$ to obtain the propagator $U(t)$.

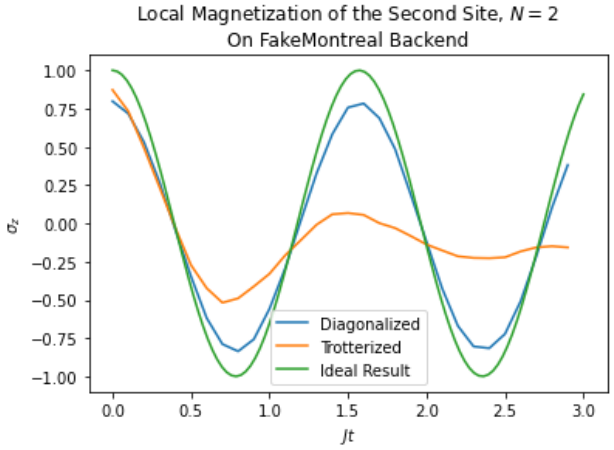

Figure 15. Comparison of circuits on noisy real quantum computer

The cost of constructing ansatz and diagonalizing the circuit by multiple iterations might seem expensive. However, the experimental result shows that it was worth it. The figure above proves that our diagonalized circuit follows the exact result much longer than the trotterized one (the stepwise implementation of propagator).

References

[1] https://www.tensors.net/exact-diagonalization, Retrieved on July 1, 2021.

[2] Phillip Weinberg, Marin Bukov. QuSpin: a Python Package for Dynamics and Exact Diagonalisation of Quantum Many Body Systems part I: spin chains. (2016). arXiv:1610.03042 [physics.comp-ph]

[3] Vatan, F. & Williams, C. Optimal quantum circuits for general two-qubit gates. Phys. Rev. A 69, 032315 (2004).

[4] Smith, A., Kim, M. S., Pollmann, F., & Knolle, J. (2019). Simulating quantum many-body dynamics on a current digital quantum computer. npj Quantum Information, 5(1), 1-13.

[5] Cîrstoiu, C., Holmes, Z., Iosue, J. et al. Variational fast forwarding for quantum simulation beyond the coherence time. npj Quantum Inf 6, 82 (2020). https://doi.org/10.1038/s41534-020-00302-0

[6] Geller, Michael & Holmes, Zoe & Coles, Patrick & Sornborger, Andrew. Experimental Quantum Learning of a Spectral Decomposition. (2021).