IBMq Trotter adaptive VQA

Jason Saroni, I-Chi Chen

March 2022

1 Methodology

1.1 AVQDS

We achieve a best performance of around 93.42±0.25 percent fidelity with the jakarta backend and will describe the methodology below. Variational quantum algorithms (VQAs) are emerging as leading strategies for obtaining quantum advantage on Noisy Intermediate Scale Quantum devices (NISQ). We implement a variational quantum algorithm that out-competes naive Trotter simulations in gate depth and ultimately state tomography fidelity. An Adaptive Variational Quantum Dynamics Simulation (AVQDS) (Yao et al., 2021; Yuan, Endo, Zhao, Li, & Benjamin, 2019) automatically generates a variational ansatz and adaptively expands it along the time evolution path. The ansatz closely matches that from exact diagonalization time evolution and the circuits require less number of gates than Trotter simulations up to the final time given. The optimal variational ansatz is obtained by minimizing the McLachlan distance L defined below

$$\frac{1}{\sqrt{2}}\left(|0\rangle \otimes |\psi_0\rangle + e^{i\theta}\,|1\rangle \otimes R_1^\dagger R_2^\dagger ... U_k^\dagger R_k^\dagger ... R_N^\dagger R_N ... R_q U_q ... R_1\,|\psi_0\rangle\right)$$

$$\begin{aligned} L^2 &\equiv \left\| \sum_\mu \frac{\partial \rho[\vec{\theta}]}{\partial \theta_\mu}\dot{\theta_\mu} - \mathcal{L}[\rho] \right\|_F^2 \\ &= \sum_{\mu\nu} M_{\mu\nu}\dot{\theta}_\mu\dot{\theta}_\nu - 2\sum_\mu V_\mu \dot{\theta}_\mu + \mathrm{Tr}[\mathcal{L}(\rho)^2] \end{aligned} \tag{1}$$

The matrix M is real symmetric and defined as

$$M_{\mu\nu} \equiv \mathrm{Tr}\left[\frac{\partial \rho[\vec{\theta}]}{\partial \theta_\mu}\frac{\partial \rho[\vec{\theta}]}{\partial \theta_\nu}\right]$$

The vector V is given by

$$V_\mu \equiv \mathrm{Tr}\left[\mathrm{Re}\left(\frac{\partial \rho[\vec{\theta}]}{\partial \theta_\mu}\mathcal{L}[\rho]\right)\right] \tag{2}$$

where $||\rho||_F \equiv \sqrt{\mathrm{Tr}[\rho^\dagger \rho]}$ is the Frobenius norm. $\rho = |\Psi\rangle\langle\Psi|$ is the density matrix for the pure state. θ_μ are the variational parameters and $\dot{\theta}_\mu$ are the time derivatives. The derivations of these expressions are shown in more detail in the following reference (Yuan et al., 2019). The variational ansatz has the following pseudo-Trotter form.

$$\Psi[\vec{\theta}] = \prod_{\mu=0}^{N_\theta - 1} e^{-i\theta_\mu \hat{A}_\mu} |\Psi_0\rangle \tag{3}$$

Where $\hat{A}_\mu$ are operators from the terms that construct the hamiltonian in this case XX, YY, and ZZ, and correspond to the relevant variational parameters θ_μ obtained through minimizing the McLachlan distance discussed below through a schematic diagram. The initial state $|\Psi_0\rangle = |110\rangle$ used in IBM corresponds to the initial state $|\Psi_0\rangle = |011\rangle$ in AVQDS as a different convention of enumerating the states is used. The following schematic diagram shows how the AVQDS algorithm works.

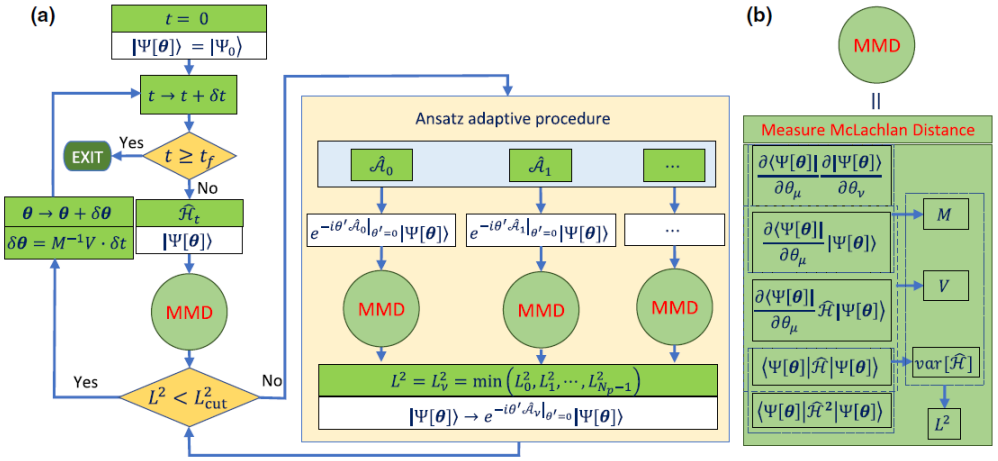

Figure 1: (a) Shows the iterative process of evolving the ansatz along the time evolution path and (b) shows the M matrix and V vector that should be calculated to minimize the McLachlan distance.

From Figure 1.(Yao et al., 2021), The optimal variational parameters are selected using using scipy.optimize.minimize in the ansatz.py module such that the McLachlan distance is minimized. The variational parameters are then evolved according to the equation $\delta\vec{\theta} = M^{-1}V\delta t$ and the optimization process

repeated as the classical component of the variational algorithm. The equations are derived in (Yuan et al., 2019; Yao et al., 2021).

1.2 Non Echoed R_{ZX} gate

Implementing the two qubits rotation gates requires two cnot gate on the quantum device. An alternative approach is to leverage knowledge of the basic set of pulses used to generate two-qubit gates at the hardware level. On IBM QPUs, well-calibrated CNOT gates are built by adding single qubit gates before and after the $R_{ZX}(\pi/2)$ gate (Alexander et al., 2020), which is defined via $R_{ZX}(\theta) = e^{-i\theta Z_c X_t/2}$, where c and t denote the control and target qubits, respectively. Ref. (Earnest, Tornow, & Egger, 2021) demonstrates how to adjust the rotating angle of $R_{ZX}(\pi/2)$ to arbitrary angle. They also show the qiskit code to do the adjustment (just like the Open Science Prize: Supplementary Material shown). The code not only allow user to use $R_{ZX}(\theta)$ gate but also optimize the pulse sequence so that the duration of whole circuits is not too long.

1.3 Quantum Error Mitigation

1.3.1 Readout Error Mitigation

To mitigate the effect of readout errors on QPU results, one can use readout error mitigation methods(Bravyi, Sheldon, Kandala, Mckay, & Gambetta, 2021) as described in Ref. and built into Qiskit Ignis. Suppose that C_{ideal} is a vector containing the list of measurement counts for each computational basis state in the absence of readout error, and that C_{noisy} is the same quantity with readout error. The relationship between C_{ideal} and C_{noisy} can be characterized by a readout error matrix M defined as

$$MC_{\text{ideal}} = C_{\text{noisy}}. \tag{4}$$

To obtain the ideal result from the noisy result, one can invert the readout error matrix:

$$C_{\text{ideal}} = M^{-1}C_{\text{noisy}}. \tag{5}$$

Qiskit Ignis supports several methods to obtain the readout error matrix M. One is to approximate M as the tensor product of readout error matrices for each qubit as follows:

$$M = \begin{bmatrix} 1-\epsilon_1 & \eta_1 \\ \epsilon_1 & 1-\eta_1 \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} 1-\epsilon_n & \eta_n \\ \epsilon_n & 1-\eta_n \end{bmatrix}, \tag{6}$$

where ϵ_j and η_j are the readout error rates for $0 \to 1$ and $1 \to 0$ respectively. This method is attractive because the ϵ_j and η_j can be estimated by executing two circuits to prepare the states $|0\ldots0\rangle$ and $|1\ldots1\rangle$ and then measuring all qubits in the CB. Another method that we call "complete readout error mitigation" prepares and measure all 2^L CB states from $|0\ldots0\rangle$ to $|1\ldots1\rangle$. This allows the elementwise extraction of M for all computational basis states, but is more costly to perform as it involves measuring exponentially many CB states.

1.3.2 Dynamical Decoupling

In a quantum computer, physical two-qubit gates have different execution times. When we stack one- and two-qubit gates into a AVQDS circuit, there are some idle qubits suffering from thermal relaxation and white noise dephasing. To reduce the decoherence, one can apply appropriate pulse sequences to stabilize the idle qubits during this waiting period. Here, we utilize the pulse sequence $\tau_{\rm iq}/4 - X_\pi - \tau_{\rm iq}/2 - X_{-\pi} - \tau_{\rm iq}/4$ with $\pm\pi$ pulse $X_{\pm\pi} = R_X(\pm\pi)$ and delay time $\tau_{\rm iq} = (T_{\rm idle} - 2t_{x,\pi})$. Here, $T_{\rm idle}$ is the idle time of the qubit and $t_{x,\pi}$ is the duration of the $X_{\pm\pi}$ pulse (Viola & Lloyd, 1998; Pokharel, Anand, Fortman, & Lidar, 2018).

2 The files folder

2.1 Requirement

To run the AVQDS code, one should first use pip to install the necessary modules. These are QuTip and h5py in addition to numpy.

2.2 Files in AVQDS heis

2.2.1 Run.py

To find the parameters and operators for $U_{AVQDS}(\pi)$, one should execute the **run.py** in AVQDS heis folder. At the end of a run, the program will print out the final variational parameters and operators closest to the time $t = \pi$. For example, it prints out that the variational parameters at 3.14149 are 0.0407, 9.4937, 0.2617, 0.2617, 6.5688 and the labels for the operators corresponding to each variational parameter are XX(0,1), ZZ(1,2), XX(1,2), YY(1,2), ZZ(0,1) where the (n, m) means the connection between two qubits (n, m). From the file params trace.dat and ans. ansatz[1] list, we will explain how to reconstruct the operators. AVQDS has an adaptive time step and so it does not precisely hit the irrational π value but time 3.14149 is closest to π and so we use the variational parameters and operators output at this time. The operators from

ansatz.h5 label [i, a, i+1, a] where a can be either 1, 2, or 3, means that the operator acts on sites i and $i+1$ and it is XX if a=1, YY if a=2, and ZZ if a=3. From above the variational parameter 0.0407 corresponds to [0, 1, 1, 1], 9.4937 to [1, 3, 2, 3], 0.2617 to [1, 1, 2, 1], 0.2617 to [1, 2, 2, 2], and 6.5688 to [0, 3, 1, 3]. This constructs the optimal variational ansatz in Pseudo-Trotter form. The resulting implemented circuit is below.

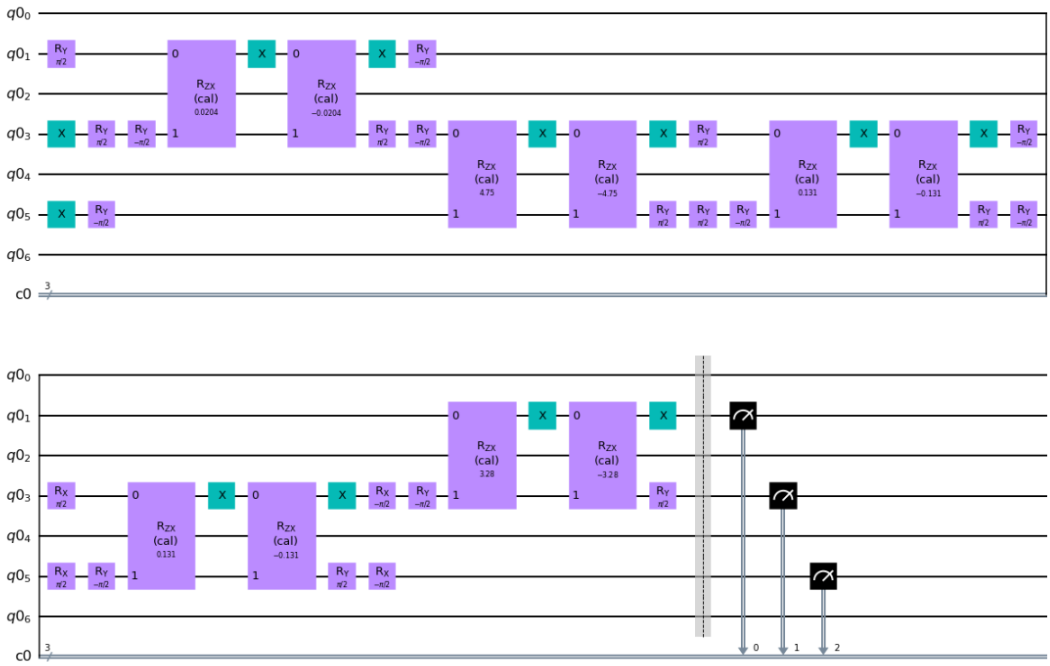

Figure 2: Circuit given by AVQDS

The gate depth of the AVQDS circuit is much shallower than that from Trotter making it more ideal for NISQ devices and giving better state tomography fidelity compared to the best Trotter result we get.

$$\Psi[\vec{\theta}] = \prod_{\mu=0}^{N_\theta - 1} e^{-i\theta_\mu \hat{A}_\mu} |\Psi_0\rangle \tag{7}$$

After executing the code of **run.py**, you will get two files, **params trace.dat** and **ansatz.h5**. Meanwhile, you can get the U_{AVQDS}'s state tomography result from the simulator. The **params trace.dat** stores all the variational parameters at each time step while **ansatz.h5** stores the operators at the final time from *AVQDS*. The variational parameters and corresponding operators are simply obtained from the final printed output from run.py as explained above.

2.2.2 Other python files

ansatz.py is used to find the variational wavefunction using a pool of operators that construct the Hamiltonian and corresponding variational parameters that minimize the McLachlan distance. **avaridyn.py** is used to stores records of desired quantities. **model.py** is used to define the Heisenberg model Hamiltonian with open boundary conditions. **plot.py** is used to plot quantities of interest like the probability of state 110 (Loschmidt echo).

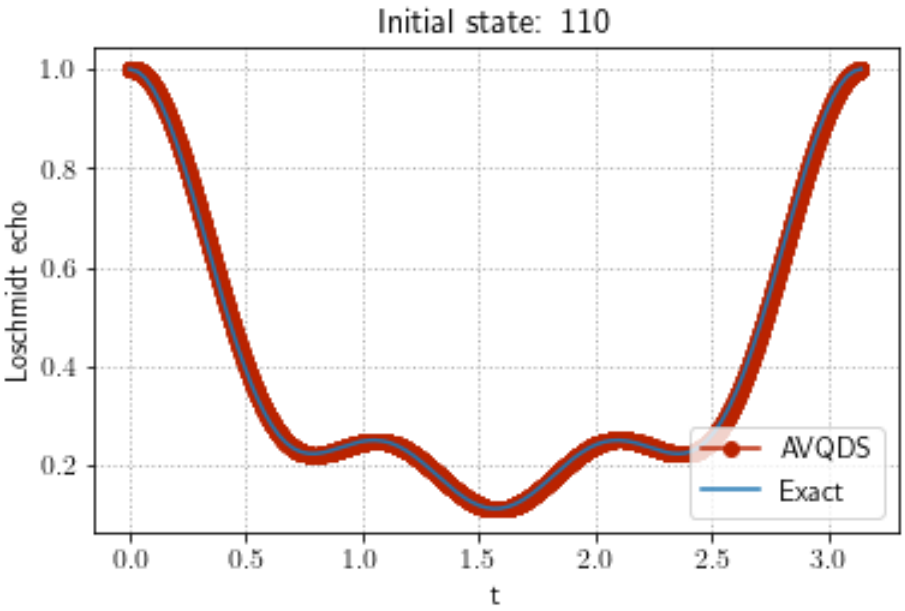

Figure 3: Shows a plot of the probability of state 110 (defined as the Loschmidt echo) from the exact wave function and from AVQDS.

Figure 3. shows very good agreement of the AVQDS and Exact wavefunctions demonstrating the efficiency of the adaptive variational hybrid algorithm in capturing the dynamics. AVQDS uses an adaptive time step and so we have used interp1d from scipy.interpolate to evaluate the results from each approach at the same time points from interpolation.

2.2.3 Jupyter Notebook

For the jupyter notebooks, there are three Jupyter notebooks. one can choose our notebook file named Mini Heisenberg model variational gate 93 U.ipynb to execute the $U_{AVQDS}(\pi)$ variational circuits. If the queue of quantum device is not too long, one may also ignore the job retrieve part. In case the judge don't think the $U_{AVQDS}(\pi)$ doesn't satisfy the rule, there is another file called Mini Heisenberg model variational gate U4 70.ipynb which repeat $U_{AVQDS}(\pi/4)$ 4

times to achieve 70 percents fidelity. The beginning of the notebooks is about how to run AVQDS simulation to create the parameters and corresponding operators. The final part of notebooks is about how to run the trained parameterized circuits on the quantum real device.

2.3 The Results Folder

In the results folder, there are 4 files. **params trace pi.dat** and **ansatz pi.h5** denote the parameters and operator sequence of $U_{AVQDS}(\pi)$. **params trace pi4.dat** and **ansatz pi4.h5** denote the parameters and operator sequence of $U_{AVQDS}(\pi/4)$.

3 Result

As the three jupyter notebooks show, the $U_{\text{AVQDS}}(\pi)$ unitary has the best performance around 93.42 ± 0.25 percent fidelity. The 4 time repeated $U_{\text{AVQDS}}(\pi/4)$ also has 70.44 ± 0.84 percent fidelity. The 3-Cnot Heisenberg operator with 6 Trotter steps just has 57.05 ± 0.8 percents fidelity. The result shows that the AVQDS unitary successfully minimize the Trotter errors so the AVQDS's result outperform the 3-CNOT Heisenberg operator's result.

References

Alexander, T., Kanazawa, N., Egger, D. J., Capelluto, L., Wood, C. J., Javadi-Abhari, A., & McKay, D. C. (2020, aug). Qiskit pulse: programming quantum computers through the cloud with pulses. *Quantum Science and Technology*, *5*(4), 044006. Retrieved from https://doi.org/10.1088/2058-9565/aba404 doi: 10.1088/2058-9565/aba404

Bravyi, S., Sheldon, S., Kandala, A., Mckay, D. C., & Gambetta, J. M. (2021, Apr). Mitigating measurement errors in multiqubit experiments. *Phys. Rev. A*, *103*, 042605.

Earnest, N., Tornow, C., & Egger, D. J. (2021, Oct). Pulse-efficient circuit transpilation for quantum applications on cross-resonance-based hardware. *Phys. Rev. Research*, *3*, 043088. Retrieved from https://link.aps.org/doi/10.1103/PhysRevResearch.3.043088 doi: 10.1103/PhysRevResearch.3.043088

Pokharel, B., Anand, N., Fortman, B., & Lidar, D. A. (2018, Nov). Demonstration of fidelity improvement using dynamical decoupling with superconducting qubits. *Phys. Rev. Lett.*, *121*, 220502.

Viola, L., & Lloyd, S. (1998, Oct). Dynamical suppression of decoherence in two-state quantum systems. *Phys. Rev. A*, *58*, 2733–2744.

Yao, Y.-X., Gomes, N., Zhang, F., Wang, C.-Z., Ho, K.-M., Iadecola, T., & Orth, P. P. (2021, Jul). Adaptive variational quantum dynamics simulations. *PRX Quantum*, *2*, 030307.

Yuan, X., Endo, S., Zhao, Q., Li, Y., & Benjamin, S. C. (2019, October). Theory of variational quantum simulation. *Quantum*, *3*, 191. Retrieved

from https://doi.org/10.22331/q-2019-10-07-191 doi: 10.22331/q-2019-10-07-191