



Searching for ground state energy with Heisenberg-limit NISQ algorithm

Team SunnyDelft

as a project for QHack 2023 Open Hackathon

Background - How to find ground state energy?



Problem: Given a Hamiltonian(matrix), How to find its ground state energy(minimum eigenvalue)?

Classical Solution: Let's rely on `numpy.linalg.eig`! But time complexity is $O(n^3)$, so you won't do it easily for large matrix :(.

Quantum Solution: Let's use the powerful *Quantum Phase Estimation*(QPE)! But currently we don't have fault-tolerant quantum computers, and the inevitable noise restricts the power of QPE :(.

NISQ Solution A: Let's try the useful Variational Quantum Eigensolver(VQE)! However, there is no performance guarantee for VQE because of its variational nature, and you don't know when the algorithm will fail :(.

Do we have any solution B?

Hybrid NISQ Algorithm with Heisenberg-limit

Open Access

Heisenberg-Limited Ground-State Energy Estimation for Early Fault-Tolerant Quantum Computers

Lin Lin and Yu Tong
PRX Quantum **3**, 010318 – Published 2 February 2022

Hamiltonian H can be written as a sum of eigenvalues λ_k and projectors Π_k on the eigenstates.

$$H = \sum_{k=0}^K \lambda_k \Pi_k$$

Given a quantum state ρ , We can define its spectrum $p(x)$ and cumulative distribution function (CDF) of the spectrum

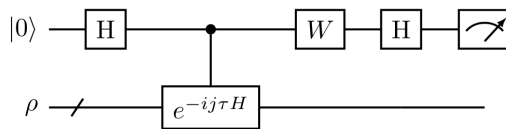
$$p(x) = \sum_{k=0}^K p_k \delta(x - \lambda_k), \text{ CDF: } \mathcal{C}(x) = \sum_{k: \lambda_k < x} p_k \quad p_k = \text{Tr}[\rho \Pi_k]$$

If we can access CDF, we can find the ground state energy: **It's where the CDF jump from zero to a non-zero value!** But Evaluating CDF is not easier than solving the eigenvalue problem.

However, we can access Approximated CDF(ACDF) with the help of quantum computer, and therefore find out the ground state energy using ACDF.

Workflow of the algorithm

- ACDF: $\tilde{C}(x) = \sum_{|j| \leq d} F_j e^{ijx} \text{Tr}[\rho e^{-ij\tau H}]$
- F_j is the Fourier coefficient of Heaviside step function and can be calculated full classically.
 - If we need high precision, the classical routine does take a long time. (e.g. a few hours for $d=20000$)
- $\text{Tr}[\rho e^{-ij\tau H}]$ can be evaluated in quantum computer using one/two ancilla qubits.



Another version using control-free evolution gates is also available

Hadamard Test

- To save complexity of quantum query, we randomly sample j using the distribution J and sum up the sampled terms:

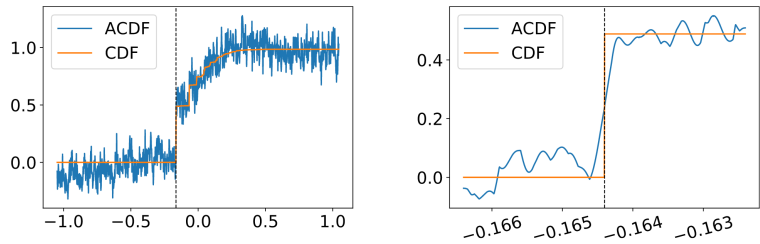
$$P(J = j) \propto |F_j|$$

For more detail, please refer to HA-noiseless-qiskit.ipynb

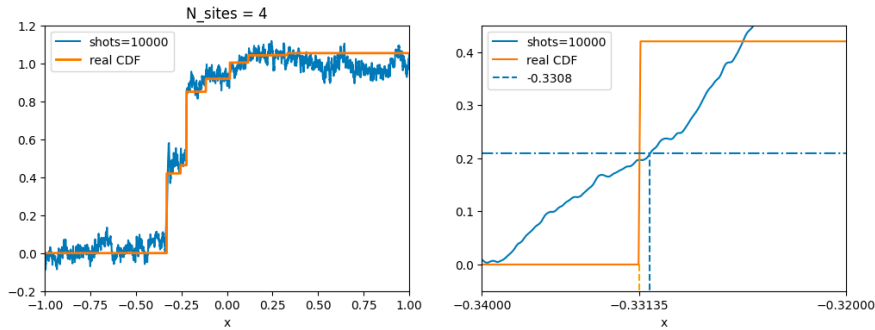
Result Replication – Noiseless simulator

Result in the paper (eight-site, ideal unitary, total_shots=3000):

- Fermi-Hubbard model (half-fill)
- Ground energy is where the ACDF jumped to half value of the CDF.

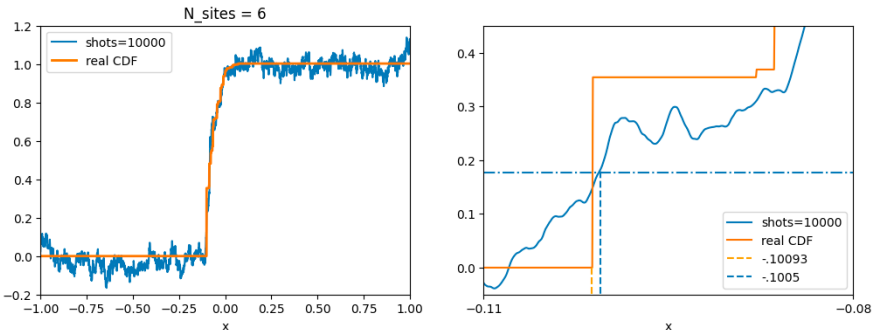


Our result (four-site, trot_step=100, total_shots=10000):



ACDF Value: -0.3308. True: -0.33135. Error: 0.0005 (0.1%)

Our result (six-site, trot_step=100, total_shots=10000):

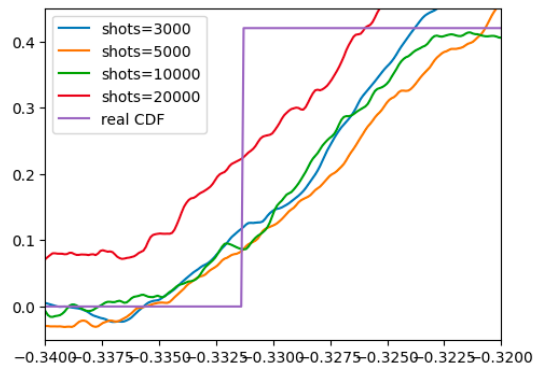
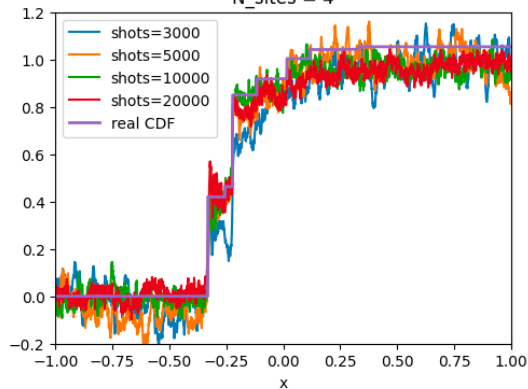


ACDF Value: -0.1005. True: -0.10093. Error: 0.004 (4%)

Effect of statistic noise

Trot_step = 100

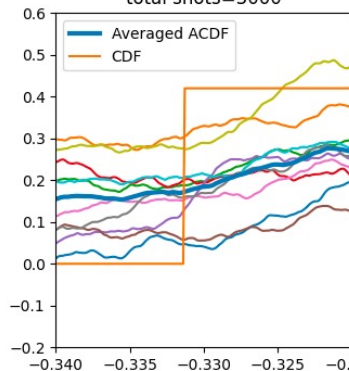
N_sites = 4



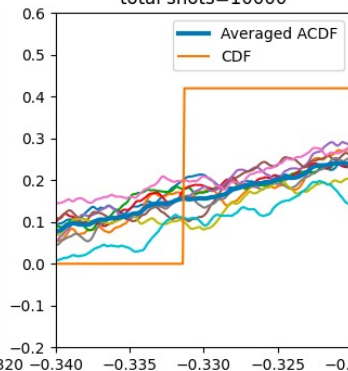
- Statistic noise only affects the variance of the ground state energy. It doesn't affect the mean.
- We can evaluate ACDF for multiple repetitions (with a relatively small total shots) and take average of it.

Trot_step = 10, N_sites=4, 10 repetitions.

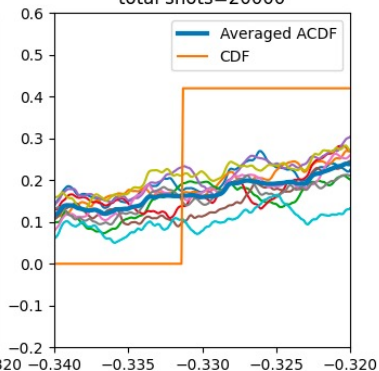
total shots=3000



total shots=10000

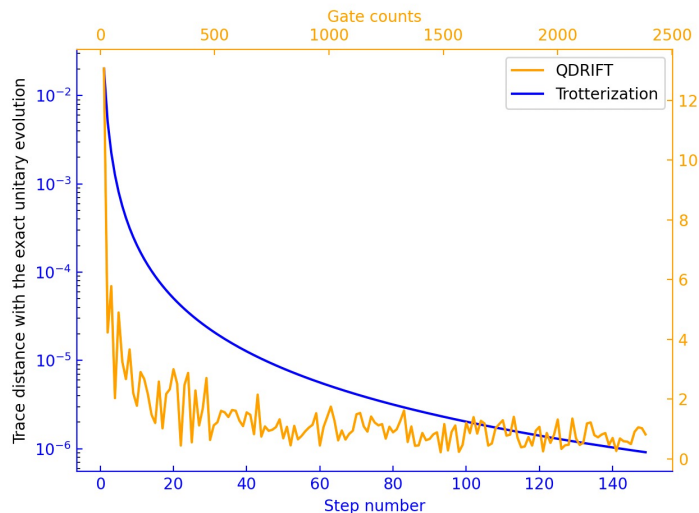


total shots=20000



Trotterization or random compiling?

- We need the evolution gate $e^{-ij\tau H}$ in our circuit.
- We can do trotterization of course, but how about random compiling(QDRIFT)?
- For the Fermi-Hubbard model, there is no need for QDRIFT.

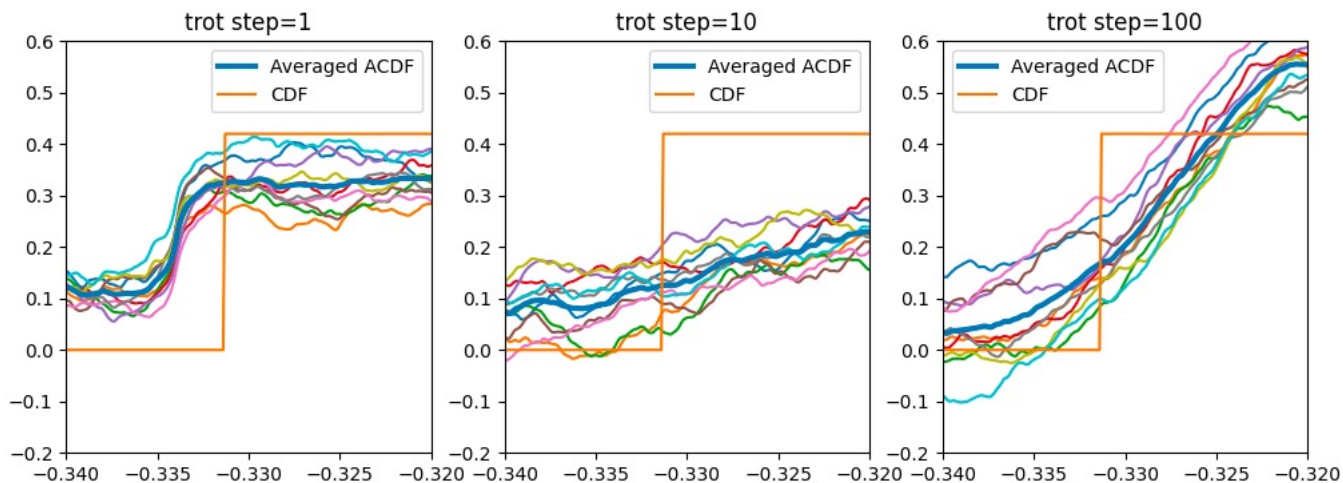


N_sites = 4

Effect of Trotterization step

- Insufficient trotterization step will lead to a flat ACDF, which is not easy to recognize the jump(ground state).

$N_{\text{sites}} = 4$, Repititions = 10



Noisy simulation



We considered noisy simulator with depolarizing noise acted on two-qubit gates.

Two error mitigation techniques are introduced:

- 1. **Error Extrapolation:**
 - CNOT errors are dominant.
 - Expand number of CNOTs in the original circuits from n to $2n + 1$ and execute them. ($n = 0, 1, 2, \dots$)
 - Plot the data and do the curve fitting in order to derive the estimated *zero-error* data.
- 2. **Randomized compiling:**
 - Convert the coherent errors into incoherent errors in order to fit well in depolarizing error models.
 - Add corresponding P, Q, R, S gate around CNOTs.
 - Ideally, all possible P, Q, R, S assignment will make no changes on CNOTs.

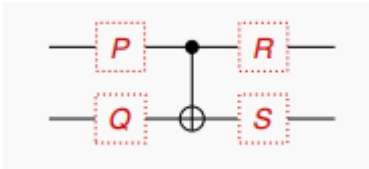
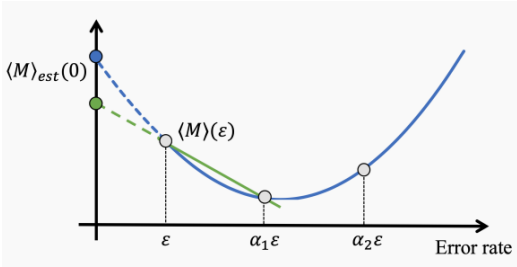
Noise tailoring for scalable quantum computation via randomized compiling

Joel J. Wallman and Joseph Emerson
Phys. Rev. A **94**, 052325 – Published 18 November 2016

Open Access

Practical Quantum Error Mitigation for Near-Future Applications

Suguru Endo, Simon C. Benjamin, and Ying Li
Phys. Rev. X **8**, 031027 – Published 26 July 2018

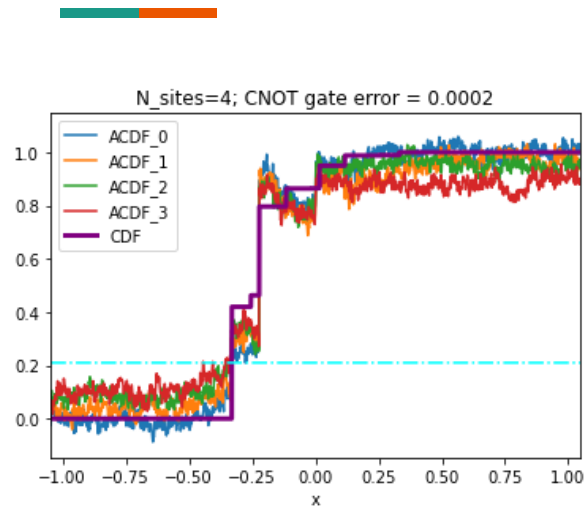


P	Q	R	S	P	Q	R	S	P	Q	R	S	P	Q	R	S
I	I	I	I	Y	I	Y	X	X	I	X	X	Z	I	Z	I
I	X	I	X	Y	X	Y	I	X	X	X	I	Z	X	Z	X
I	Y	Z	Y	Y	Y	X	Z	X	Y	Y	Z	Z	Y	I	Y
I	Z	Z	Z	Y	Z	X	Y	X	Z	Y	Y	Z	Z	I	Z

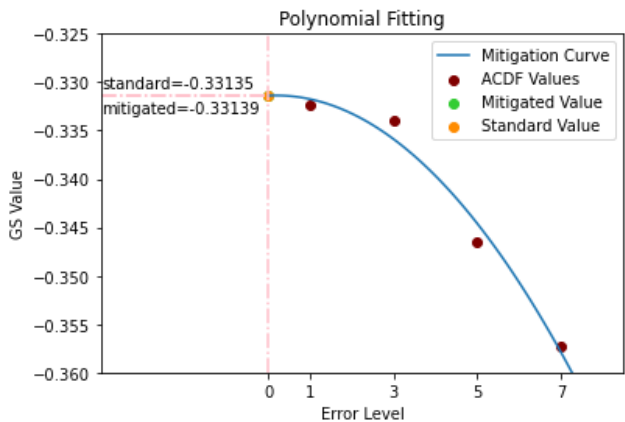
Mitigating Depolarizing Noise on Quantum Computers with Noise-Estimation Circuits

Miroslav Urbaneck, Benjamin Nachman, Vincent R. Pascuzzi, Andre He, Christian W. Bauer, and Wibe A. de Jong
Phys. Rev. Lett. **127**, 270502 – Published 27 December 2021

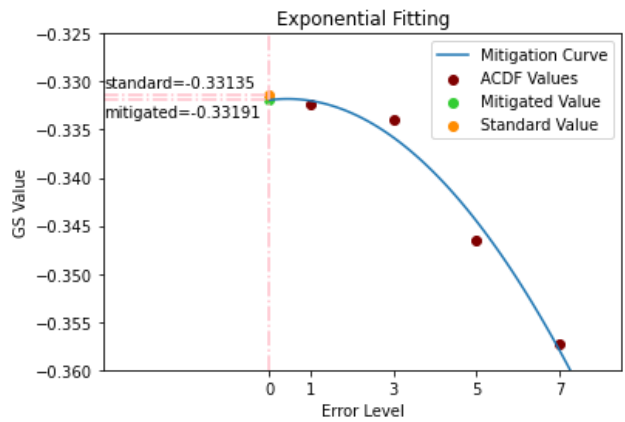
Noisy simulation result (N_sites = 4)



N_sites=4, CNOT gate error = 0.0002,



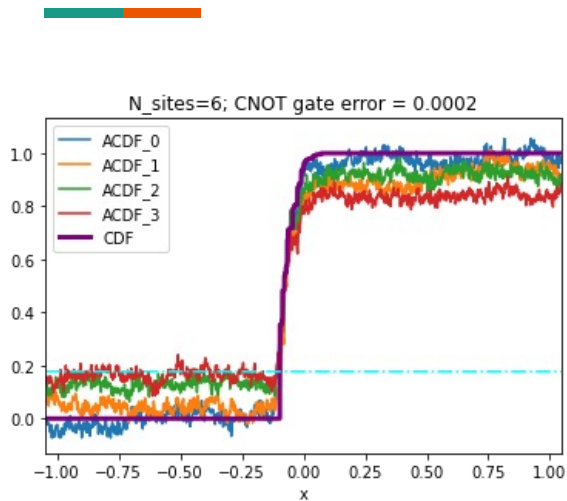
*Polynomial Curve Fitting:
Mitigated Value=-0.33139
Standard Value=-0.33135*



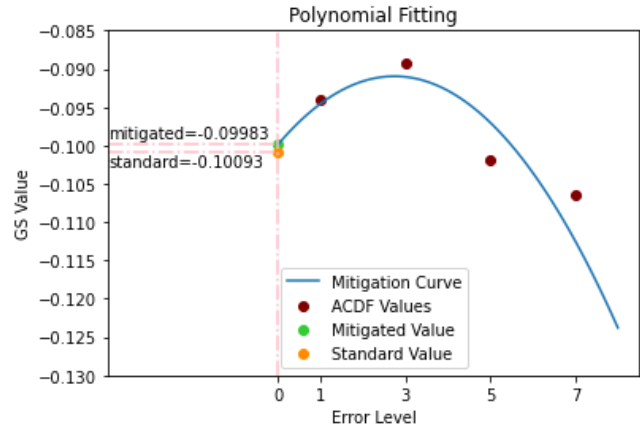
*Exponential Curve Fitting:
Mitigated Value=-0.33191
Standard Value=-0.33135*

Polynomial Extrapolation works better.

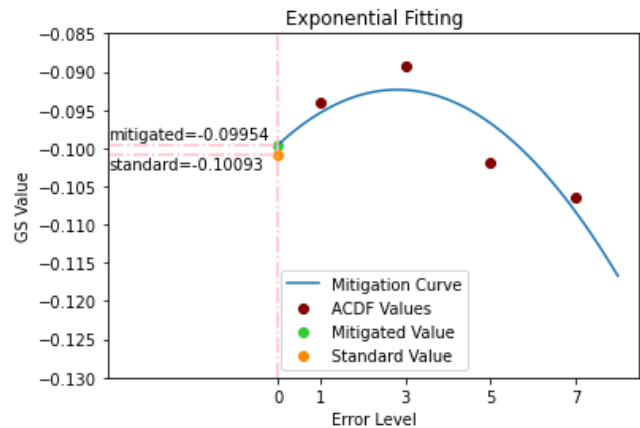
Noisy simulation result (N_sites = 6)



$N_{\text{sites}}=6$, CNOT gate error = 0.0002,



*Polynomial Curve Fitting:
Mitigated Value=-0.09983
Standard Value=-0.10093*

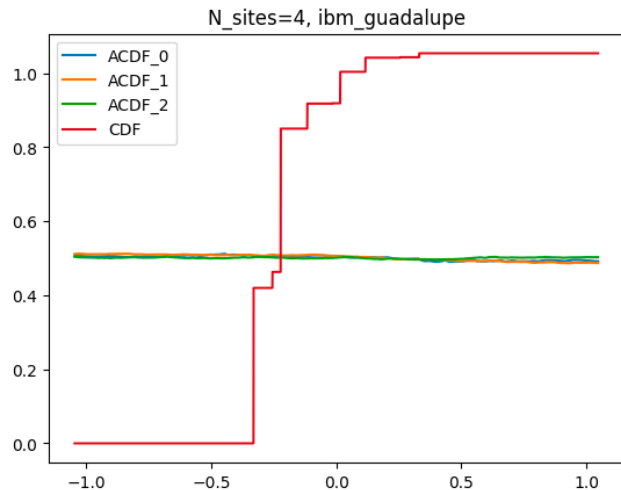


*Exponential Curve Fitting:
Mitigated Value=-0.09954
Standard Value=-0.10093*


Polynomial Extrapolation works better.

A try on *ibm_guadalupe*(16-qubit)

- Thanks to the Power-up, we have the chance to access the *ibm_guadalupe*.
- However, the circuit execution suffered from the depolarizing noise greatly, and extrapolation doesn't help.



Project Summary

- 
- We successfully implemented the Algorithm in [PRXQuantum.3.010318] with Qiskit and AWS-Braket and replicated the result presented in the original paper.
 - Besides, we moved further:
 - We compared QDrift with Trotterization and considered effect of Trotterization for the evolution operator, which is not involved in the original paper.
 - We did simulation in a noisy environment and applied error mitigation on our result.
 - We try to execute our algorithm on the real backend, but the result is not satisfying due to the scale of real noise.
 - Future plan:
 - We will increase noise scale in simulator and try more error mitigation methods, e.g. noise-estimation circuit[PRL 127, 270502].
 - We will optimize our circuit submitted to real backends and aim to produce useful results for small-size Hamiltonians.
 - We will move from Fermi-Hubbard model to molecular Hamiltonians and compare this method with VQE quantitatively.