Measuring Ground State Energy of Simple Molecules using VQE

Project Report – Group 28A

Sourav Datta, Karleen Tur, Binabrata Das, Karthik Rajagopalan

1 Overview

Variation Quantum Eigensolver (VQE) is an effective and noise resilient algorithm used in Quantum Chemistry to find molecular structures that, in theory, can run partly on NISQ-era quantum computers and partly on classical systems. In this project we use VQE to find ground state energy of simple molecules like H2, LiH and H2O. We primarily use quantum simulators running on classical computers to determine the output of the VQE algorithm and compare those with classical ground state solvers. Also, we use two different types of simulators, namely, IBM's **Qiskit** [2] and **Classiq** [3] simulators – the first for **H2** and the second for **LiH** and **H2O** molecules.

2 Methodology

The general structure of the algorithm used in qiskit and classiq is as follows.

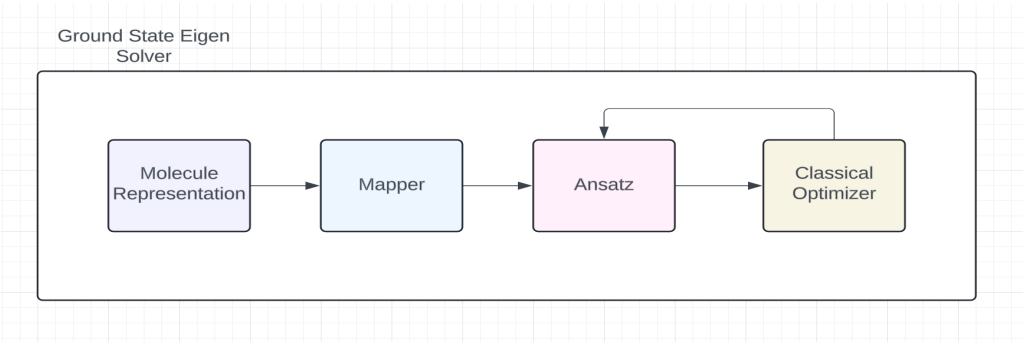

Figure 1 - General structure of the algorithm flow

1. Molecular representations – we use spatial coordinates representation (e.g., **H 0 0 0; H 0 0 0.735** for H2 molecule at equilibrium bond length (0.735 Angstrom) in a singlet state with no charge.)
2. Mapper – **Jordan Wigner** and **BravyiKitaev** for mapping second quantized form to pauli-strings.
3. Ansatz – Unitary Couple-Cluster Singles and Doubles (UCCSD).
4. Classical Optimizer – SLSQP and COBYLA.

3 Results

Below are results obtained for the three molecules with accuracies.

3.1 Accuracy Comparison with Exact Eigensolver

We measured the accuracy of the VQE solution by comparing the results with a purely classical exact solver like **NumPyMinimumEigensolver** and found the below results.

Molecule	Electronic Energy (Numpy)	Total Energy (Numpy)	Electronic Energy (VQE)	Total Energy (VQE)	Classical Optimizer	VQE Simulator
H2	-1.857275030202	-1.137306035753	-1.8593368706597844	-1.1393678762108048	SLSQP	Qiskit

H2	-1.857275030202	-1.137306035753	-1.5807675641584518	-0.8607985697094721	COBYLA	Classiq – HEA
LiH	-8.877080999128	-7.882386993639	-8.113872579878574	-7.119178574389853	COBYLA	Classiq
LiH	-8.877080999128	-7.882386993639	-8.876853391779	-7.88215938629	Parity, SLSQP (max iterations = 2500)	Qiskit v0.28.0
H2O	-84.206350593117	-75.012437432494	-80.48048482084272	-71.28657166021934	COBYLA	Classiq
H2O	-84.206350593117	-75.012437432494	-84.206171751538	-75.012258590914	Parity, SLSQP (max iterations = 2500)	Qiskit v0.28.0

Figure 2 - Comparison of classical and VQE eigen solvers

3.2 Execution Time Comparison

Molecule	Classical Eigensolver (s)	Qiskit Simulator (s)	Classiq Simulator (s)
H2	1.05	1.16	0.181
LiH	4.75	70*	38.0
H2O	8.92	545*	364.4

Figure 3 - Comparison of execution time

[* - indicates the algorithm never finished on a local qiskit 1.0 simulator even after running for more than 2 hours. The results were obtained from an earlier version of qiskit (0.28.0) which completed locally with **Parity** mapping]

4 Heuristic Considerations

Below are some of considerations and results in order to optimize the VQE performance.

4.1 Hardware-Efficient Ansatz

Hardware efficient ansat [4] is a fixed structure ansatz that is used to fit a specific quantum hardware where the VQE algorithm will be running. It is sometimes suggested over UCCSD. The input parameters to the Ansatz are a qubit mapping based on which the ansatz then creates entanglements between the supplied number of qubits. Using classiq's simulator, below is an example of 4 qubit map specific for H2 molecule.

```
hwea_params = HEAParameters(
    num_qubits=4,
    connectivity_map=[(0, 1), (1, 2), (2, 3)],
    reps=3,
    one_qubit_gates=["x", "ry"],
    two_qubit_gates=["cx"],
)
```

Figure 4 - Hardware efficient Ansatz for H2 (Classiq)

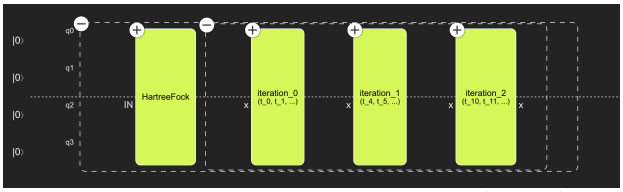

Figure 5 - Circuit diagram for the HE Ansatz (Classiq)

A downside of HEA is that its expressibility is often limited as it spans a smaller Hilbert space and hence cannot guarantee an accurate ground state can be produced always.

4.2 Adaptive Structure Ansatz [5]

Adaptive ansatz (e.g., Fermionic-ADAPT-VQE) are created by iteratively adding gates to the original ansatz such that each step adds to the overall energy or any other metric. Although we haven't investigated or used these types of ansatz in the project, they may become efficient alternatives in the future.

4.3 Future areas of investigation for Ansatz selection

Based on current research [6], **k-UpCCGSD** ansatz appears to be numerically more accurate and linearly scalable than our current choice UCCSD – we would like to extend the project to investigate more on how to use the ansatz variants on complex molecules like LiH and H2O.

4.4 Classical Optimizer Selection

Although we only ran on a classical simulator, selection classical optimizer (Gradient vs Gradient-free) produced a significant execution time difference. Below is the table for H2 molecule.

Molecule	SLSQP (s)	COBYLA (s)	Nelder-Mead (s)
H2	0.924	2.55	12.1

Figure 6 - Execution time comparison of Optimizers

4.5 Mapper Selection

We experimented with different selections of fermionic second quantization encoding and below is the result for H2 molecule (with optimization fixed to SLSQP)

Molecule	JordanWigner (s)	BravyiKitaevMapper (s)	BravyiKitaevSuperFastMapper (s)
H2	1.16	1.07	1.05

Another important point to note is that Parity mapping improved the execution time significantly for qiskit simulations of LiH and H2O. Jordan Wigner or Bravyi Kitaev mappings took many minutes to finish or did not finish at all.

5 Challenges

One major challenge was the execution time of qiskit programs for bigger molecules like LiH and H2O did not finish running on local machines. This happened once we switched to version 1.0 of Qiskit. To get around that, we employed Classiq's quantum algorithm modeler and ran the simulations on a fast and powerful simulator running in the cloud (provided by classiq). Another approach we took was to go back to Python 3.9 and install an older version of qiskit (0.28.0) which is known to complete. Then we ported the code to a t2.large Linux machine on AWS and ran the code. This printed the results correctly.

References

1. All code in the form of notebooks can be found here: https://github.com/souravdatta/qcml_project
2. Qiskit https://qiskit.github.io/ecosystem/
3. Classiq https://platform.classiq.io/
4. Hardware Efficient Ansatz https://arxiv.org/abs/1704.05018
5. Quantum Chemistry https://arxiv.org/abs/1808.10402
6. Lee, J., Huggins, W.J., Head-Gordon, M., Whaley, K.B.. Generalized unitary coupled cluster wave functions for quantum computation. Journal of Chemical Theory and Computation 2018;15(1):311–324. doi:10.1021/acs.jctc.8b01004.
7. Qiskit Nature https://qiskit-community.github.io/qiskit-nature/