

Benchmarking the Variational Quantum Eigensolver using different quantum hardware

Amine Bentellis^{†*}, Andrea Matic-Flierl^{*}, Christian B. Mendl^{†‡}, Jeanette Miriam Lorenz^{*}

^{*}Fraunhofer Institute for Cognitive Systems IKS, Munich, Germany

[†]Technical University of Munich, CIT, Department of Computer Science, Garching, Germany

[‡]Technical University of Munich, Institute for Advanced Study, Garching, Germany

{amine.bentellis, andrea.matic-flierl, jeanette.miriam.lorenz}@iks.fraunhofer.de
christian.mendl@tum.de

Abstract—The Variational Quantum Eigensolver (VQE) is a promising quantum algorithm for applications in chemistry within the Noisy Intermediate-Scale Quantum (NISQ) era. The ability for a quantum computer to simulate electronic structures with high accuracy would have a profound impact on material and biochemical science with potential applications e.g., to the development of new drugs. However, considering the variety of quantum hardware architectures, it is still uncertain which hardware concept is most suited to execute the VQE for e.g., the simulation of molecules. Aspects to consider here are the required connectivity of the quantum circuit used, the size and the depth and thus the susceptibility to noise effects. Besides theoretical considerations, empirical studies using available quantum hardware may help to clarify the question of which hardware technology might be better suited for a certain given application and algorithm. Going one step into this direction, within this work, we present results using the VQE for the simulation of the hydrogen molecule, comparing superconducting and ion trap quantum computers. The experiments are carried out with a standardized setup of ansatz and optimizer, selected to reduce the number of required iterations. The findings are analyzed considering different quantum processor types, calibration data as well as the depth and gate counts of the circuits required for the different hardware concepts after transpilation.

Index Terms—Quantum Computing, Variational Quantum Eigensolver, Quantum Hardware Comparison, Ion Trap Quantum Computers, Superconducting Quantum Computers

I. INTRODUCTION

One of the primary applications being explored for quantum computers is quantum chemistry. It has the potential to simulate weakly and strongly correlated molecules and materials [1, 2], which fall under the category of simulation problems [3, 4]. Numerous algorithms have been suggested for this purpose. Some of them rely on fault-tolerant quantum hardware computation, such as the Quantum Phase Estimation (QPE) algorithm, which is currently not feasible. Thus, big parts of the research effort is put into Noisy Intermediate-Scale Quantum (NISQ) compatible algorithms, such as Variational Quantum Algorithms (VQA). The Variational Quantum Eigensolver (VQE) [5] is an exemplary VQA, which estimates the eigenvalues and low-lying eigenstates of a given Hamiltonian.

This research is part of the Munich Quantum Valley, which is supported by the Bavarian state government with funds from the Hightech Agenda Bayern Plus.

Its application is not restricted to ground state energy evaluations but it can also be used to determine optimal molecule geometries [6]. It also finds its uses outside of quantum chemistry, and can be employed in a variety of problems that can be formulated as a Hamiltonian [7].

Despite the vast amount of potential applications of the VQE, it is currently a question for which particular applications the VQE will show benefits in practice. A quantum advantage in general has only been shown in theory so far, such as e.g., in Grover’s and Shor’s algorithms, and in academic examples like in Google’s Sycamore experiment [8]. A practical quantum advantage, i.e., an advantage of quantum computation over classical computation in an industrially relevant problem has however not yet been demonstrated. To eventually reach such a practical quantum advantage, it has to be demonstrated for the complete computation workflow which includes both parts of quantum and classical computation. In particular, NISQ algorithms require the use of classical optimizers, which need to work in tandem with the quantum computation part. Noise presents an additional problem within the NISQ era and may imply that a result of a quantum computation is diluted, if the quantum circuits contained within the calculation are too wide and deep (i.e., require too many qubits and gates). Currently, techniques such as error mitigation and postselection are being utilized to counteract noise. It can also create barren plateaus, which act as obstacles in the loss landscape and hinder the optimization process of variational algorithms [9]. Therefore, it is mandatory within the NISQ era to only consider shallower circuits.

Considering in this context different quantum hardware concepts, these may be better or less well suited to execute a given quantum circuit. However, which kind of quantum algorithm and noise level is acceptable for a given application in dependence on specific quantum hardware concepts is a largely unanswered question. We therefore argue that a systematic application-driven benchmarking procedure is required, that considers all computation steps - may it be classical or quantum computation, and their integration - as well as different quantum hardware technologies and noise levels. The benchmarking procedure is consequently multi-layered, beginning with the definition of the problem, detailing the steps of quantum and classical computation, transpilation to quantum

arXiv:2305.07092v1 [quant-ph] 11 May 2023

hardware and eventually extends to a specific quantum device on which the algorithm is executed. Additionally, the selection of the optimizer, the ansatz design and the problem mapping needs to be considered.

This complete benchmarking procedure being a complicated collection of individual steps, we focus in this study here on the step of selecting the quantum hardware suited for a specific application of the VQE. Therefore, the objective here is to evaluate and assess the solutions of a VQE implementation provided by both superconducting and ion trap hardware, considering both qualitative and quantitative aspects. The findings may help in understanding if a certain hardware concept is more suited than the other for a specific application and may give indications into which directions quantum algorithms, the interplay of quantum and classical computation and quantum hardware need to be developed.

In the particular study within this paper, we concentrate on simulating the H_2 molecule as a basic use case for comparing superconducting and ion trap quantum hardware. Choosing the H_2 molecule provides an example, where the ground state can exactly be calculated theoretically and which has been extensively investigated in the scientific literature. Therefore, despite its relatively simple nature, it provides an adequate foundation for evaluating the quality of the solutions. Our work does not include an analysis of the scaling capabilities of the quantum hardware.

II. HARDWARE

For this study we opted to choose two different quantum processors, one processor of Alpine Quantum Technologies (AQT) [10] and one of IBM Q [11]. The main decisive factor in selecting these two were that they were easily available to us. This however also implies that for the selection of the IBM Q processor, we could not choose the most recent processor with improved error mitigation, as prohibitively long waiting times in the queue made extended tests by us impossible. Equally, in perspective, we will include further quantum hardware of different technologies into our studies, such as e.g. neutral atom systems, which had not been available to us pursuing the work presented in this paper.

A. AQT trapped ion quantum computer

The processor *aqt_marmot* hosted by AQT [10] is based on trapped $^{40}\text{Ca}^+$ ions. The *aqt_marmot* system supports a register size of up to 16 qubits featuring all-to-all connectivity. The native gate set comprises single-qubit gates with arbitrary rotation angles and axis. The entangling operation is a two-qubit gate with arbitrary rotation angle around the x-axis that can be implemented between any qubit pair. The hardware supports all major software development kits, where we use IBM Qiskit [12] to implement the presented measurements. Furthermore, we choose $[rx, rz, rxx]$ as the basis gate set in Qiskit, since it closely resembles the native gate set of the *aqt_marmot* system. The error rates of single-qubit gates (*rx* error) are approximately $3 \cdot 10^{-4}$ on average, whereas the error rate of the two-qubit gate (*rxx* error) is around $1 \cdot 10^{-2}$

on average. The typical gate times are $15 \mu\text{s}$ for single-qubit gates and $200 \mu\text{s}$ for two-qubit gates. T1 and T2 times are respectively 1.14 ± 0.06 seconds and 0.452 ± 0.068 seconds, making the coherence/gate time ratio 10^3 [13].

B. IBM superconducting computer

IBM is a primary supplier of superconducting devices for quantum computing [11], and their machines can be readily operated via their cloud services and Qiskit [12]. All runs are performed on the *ibmq_manila* hardware, with minimal time intervals between them, so that the calibration of the hardware is similar for each run. The specific backend is a Falcon processor Falcon r.11L version 1.1.4. Other available IBM Q processors feature improved coherence properties, but show similar readout (with the exception of *ibm_sherbrooke*), single and double-gate errors (Table I shows the calibration data at the time of the study). Average T1 and T2 across the whole chip are $169 \mu\text{s}$ and $76 \mu\text{s}$.

TABLE I
ibmq_manila CALIBRATION DATA SHOWING 1 AND 2-QUBITS GATE ERROR RATES AS WELL AS THE GATE TIME.

Qubit	ID error	\sqrt{x} (sx) error	Pauli-X error	CNOT error	Gate time (ns)
0	2.057e-4	2.057e-4	2.057e-4	0_1: 5.621e-3	0_1: 277.3
1	2.236e-4	2.236e-4	2.236e-4	1_2: 1.544e-2 1_0: 5.62e-3	1_2: 469.3 1_0: 312.8
2	1.795e-3	1.795e-3	1.795e-3	2_3: 8.233e-3 2_1: 1.544e-2	2_3: 355.6 2_1: 504.8
3	2.025e-4	2.025e-4	2.025e-4	3_4: 5.0636e-3 3_2: 8.233e-3	3_4: 334.2 3_2: 391.1
4	3.341e-4	3.341e-4	3.341e-4	4_3: 5.063e-3	4_3: 298.7

III. EXPERIMENTS AND RESULTS

A. The VQE algorithm

The Variational Quantum Eigensolver (VQE) is a popular quantum chemistry algorithm for near-term quantum computers [5, 14]. The VQE leverages the Rayleigh-Ritz variational principle, whereby a quantum algorithm is trained to find the ground state of a particular molecule. VQE is aimed at finding the energy E_G of a Hamiltonian H ,

$$H |\psi_i\rangle = E_i |\psi_i\rangle, \text{ with } i = G, 1, 2, \dots \quad (1)$$

$$E_G \leq E_1 \leq \dots, \langle \psi_i | \psi_j \rangle = \delta_{ij}$$

Using the Born-Oppenheimer approximation [15], H is represented in second quantization as:

$$H = \sum_{pq} h_{pq} a_p^\dagger a_q + \frac{1}{2} \sum_{pqrs} h_{pqrs} a_p^\dagger a_q^\dagger a_r a_s, \quad (2)$$

with a and a^\dagger are respectively the fermionic annihilation and creation operators. Subsequently, this Hamiltonian is mapped using Jordan-Wigner transformation to the qubit space [16]. After mapping it can be expressed in terms of Pauli strings σ^i as $H = \sum_i c_i \sigma^i$ with the coefficients $c_i \in \mathbb{R}$. The cost function can then be formulated as the expectation value of H over a trial state $|\psi(\theta)\rangle = U(\theta) |\psi_0\rangle$, where $U(\theta)$ is an ansatz and $|\psi_0\rangle$ is an initial state. The objective is to minimize the cost function, which is achieved by adjusting the parameters θ of the ansatz $U(\theta)$.

$$C(\theta) = \langle \psi(\theta) | H | \psi(\theta) \rangle \quad (3)$$

Therefore, the cost function $C(\theta)$ is obtained from a linear combination of expectation values of σ_i . The Rayleigh-Ritz variational principle states that the cost function $C(\theta)$ is both accurate and significant, where $C(\theta) > E_G$ and when $|\psi(\theta)\rangle$ represents the ground state $|\psi_G\rangle$ of H , equality holds true.

B. Ansatz and Optimizer

Two ansatzes were considered: the hardware-efficient ansatz (Figure 1) and the unitary coupled-cluster ansatz (UCC), which is chemically inspired. More specifically for the UCC ansatz, a variant based on single and double electron excitations (UCCSD) [5] was considered. While the UCCSD ansatz is a viable candidate, it suffers from several limitations that render it unsuitable for benchmarking purposes. Specifically, it exhibits poor scalability in terms of gate requirements, leading to increasingly deeper circuits as the molecule size increases. For the H_2 molecule the circuit is already relatively deep (circuit depth of 92 for *ibmq_manila*), and if moving to more complicated molecules like LiH, it becomes impossible to run on real hardware without simplifying the problem. For the purposes of this work, we therefore employed the RY-CNOT ansatz. Figure 1 illustrates that the circuit is composed of only four RY gates, which are parameterized, and an entangling layer that consists of CNOT gates arranged in a circular manner. This all-purpose ansatz not only simplifies the implementation on various quantum hardware with their unique gate sets but also results in a significantly faster optimization process.

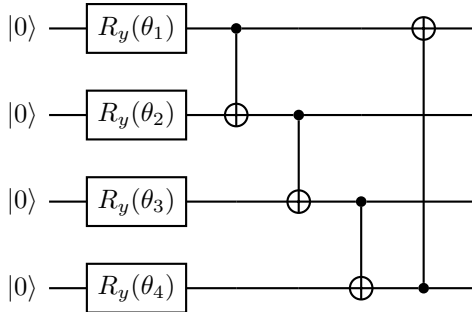


Fig. 1. RY-CNOT ansatz using rotations around the Y axis

After transpilation on *aqt_marmot* and *ibmq_manila* (resp. Figure 3 and 4), both circuits significantly increase in size. For both circuits, Qiskit’s optimization level 3 is used. Optimization level 3 spends the most computational effort to optimize the circuit out of the four available levels. Figure 5 details the difference in terms of gate count and depth. We see circuits of similar depth (16 for *ibmq_manila* and 14 for *aqt_marmot*), however, the two transpiled circuits differ in the number of non-local gates or 2-qubit gates (8 for *ibmq_manila* and 4 for *aqt_marmot*). This discrepancy can be explained by the addition of two swapping operations for the IBM Q hardware since SWAP gates are, when transpiled, transformed into two consecutive and opposite CNOT gates. The use of SWAP gates is necessary depending on the architecture of a superconducting chip as these chips show a limited

connectivity and can only directly entangle qubits with a direct connection on the chip (Figure 2 shows *ibmq_manila*’s topology).



Fig. 2. Chip topology for *ibmq_manila* comprising of five superconducting qubits.

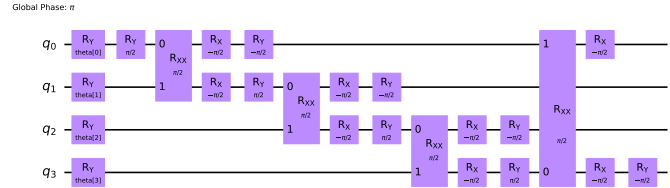


Fig. 3. RY-CNOT circuit after transpilation on *aqt_marmot*.

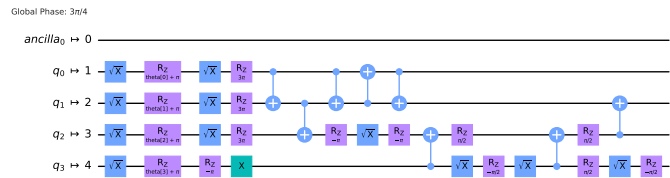


Fig. 4. RY-CNOT circuit after transpilation on *ibmq_manila*.

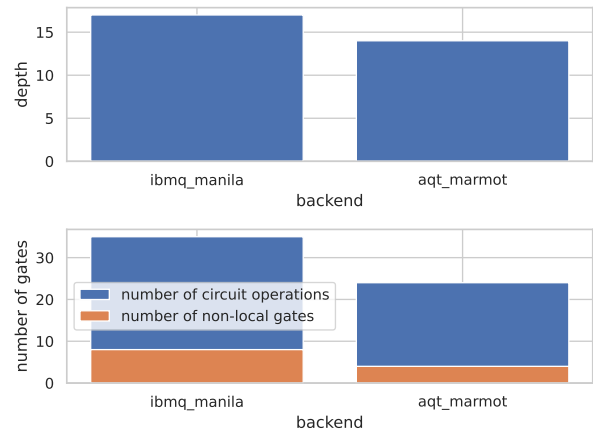


Fig. 5. Depth and gate counts of post transpilation circuits.

The selection of an appropriate optimizer is a more nuanced decision that requires thorough testing. A comparison of several popular optimizers such as Nakanishi-Fuji-Todo (NFT) [17], Nelder-Mead [18] and SPSA [19] for VQE algorithms in this context is presented in the Appendix (Figure 9). All three optimizers evaluated avoid the need for computing the derivative of the circuit, which can be costly for large circuits,

but instead rely on certain strategies to navigate the loss landscape. It is notable to say that gradient-based methods like Adam perform poorly on such tasks [20]. Our results are consistent with [20] and suggest that the NFT optimizer outperforms the other optimizers by a significant margin. Indeed, Figure 9 suggests that NFT achieves convergence in an average of four iterations on this problem.

C. Experiments

For the simulations of the H_2 molecule we use the 4-qubit Hamiltonian:

$$\begin{aligned}
 H = & c_1 I_0 \\
 & + c_2 Z_0 Z_2 \\
 & + c_3 Z_1 Z_3 \\
 & + c_4 (Z_3 + Z_1) \\
 & + c_5 (Z_2 + Z_0) \\
 & + c_6 (Z_2 Z_3 + Z_0 Z_1) \\
 & + c_7 (Z_0 Z_3 + Z_1 Z_2) \\
 & + c_8 (Y_0 Y_1 Y_2 Y_3 + X_0 X_1 Y_2 Y_3 + Y_0 Y_1 X_2 X_3 + X_0 X_1 X_2 X_3)
 \end{aligned}$$

using prefactors c_i at a distance between atoms of 0.735 Å, where the specific values of c_i can be found in the appendix. The Hamiltonian is represented using the minimal STO-3g basis set [21]. The exact ground state energy of the problem, calculated classically, is $E_{\text{FCI}} = -1.136189454088$ Ha. Despite the generic hardware-efficient ansatz utilized, a shot-based simulator (*qasm_simulator*) can achieve relatively accurate energies ($|E_{\text{VQE}} - E_{\text{FCI}}| = 0.00365$ Ha), indicating that the ansatz is capable of finding the solution. The experimental settings chosen for all the runs in this study are:

- Circuit: RY-CNOT (c.f. Figure 1)
- Optimizer: NFT
- Framework: Qiskit
- Number of shots: 200
- No error mitigation
- Number of iterations: 15

Error mitigation has emerged as a one of the key ways to deal with noise in NISQ era devices [14]. However, AQT’s hardware currently does not support the native use of error mitigation techniques. Therefore, results for *aqt_marmot* are provided without error mitigation.

This comparison is based upon nine different parameter seeds (i.e. parameters drawn from a uniform distribution of $[\pi, -\pi]$). We see that the processor *ibmq_manila* takes an average of 7 iterations to find the ground state, whereas the *aqt_marmot* requires approximately 4 iterations (Figure 6). In both cases, the choice of the seed can impact the optimization process, as seen with the two outliers (dotted lines in Figure 6). Furthermore, the effect of noise on the optimization process is demonstrated by the varying sizes of the uncertainty bands. These bands in Figure 6, which exclude the outliers, represent one standard deviation from the mean of the seven runs. Moreover, a mean energy difference of 0.094 between the two processors (or 9.34%) is observed when considering all runs

(Figure 6). The precise reason for this discrepancy in accuracy is not fully understood, as factors such as gate fidelity and connectivity may not fully account for the observed difference in result precision.

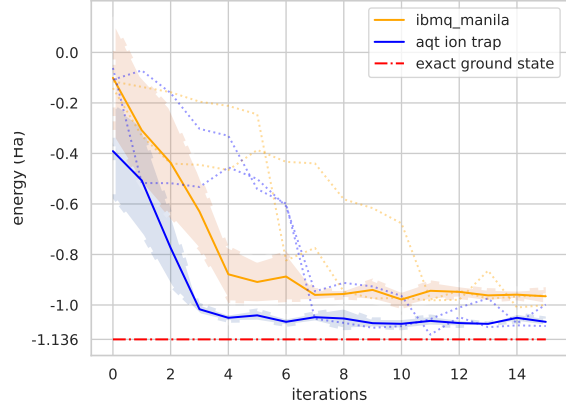


Fig. 6. Comparison of the optimization process between the processors *aqt_marmot* and *ibmq_manila*.

Figure 7 and table II show the energy difference between the last steps of the optimization and the theoretically known ground state energy. This plot is generated from the last four iterations of respective hardware runs. This plot also shows that the unmitigated results are relatively close to the exact ground state.

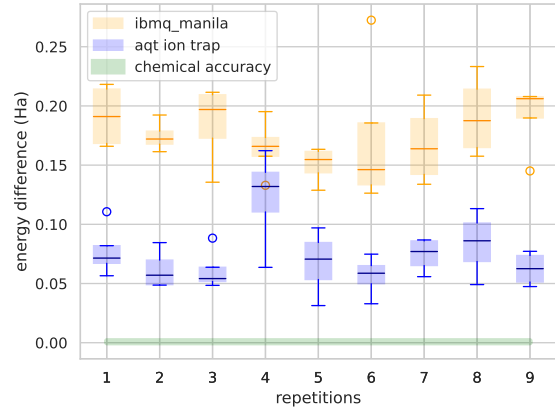


Fig. 7. Energy difference between the theoretically known ground state and the last four iterations of the optimization process for all 9 runs (repetitions).

One significant other parameter to take into consideration is the amount of time required for each hardware run. Over 15 iterations, one run on *aqt_marmot* takes on average 3 hours and 40 minutes, while *ibmq_manila* takes up to 5 minutes to get the results (Table II). We only consider here the time required to evaluate the quantum circuit, and exclude the classical computation parts. It is however unclear how this runtime scales with increasing problem size.

TABLE II
QUANTITATIVE COMPARISON OF THE TWO TESTED QUANTUM PROCESSORS

	Final energy (Ha)	Minimum energy (Ha)	$ E_{\text{VQE}} - E_{\text{FCI}} $ (Ha)	Time taken (quantum) (seconds)
<i>ibmq_manila</i>	-0.975 ± 0.032	-1.006 ± 0.019	0.130 ± 0.019	334 ± 40
<i>aqt_marmot</i>	-1.059 ± 0.028	-1.100 ± 0.010	0.036 ± 0.010	13297 ± 625

The internuclear distance plot shown in Figure 8 determines the molecular geometry, which serves as the foundation for many molecular property simulations. The zoomed part of the plot corresponds to the global minimum located at 0.735 Å where *ibmq_manila* is compared to *aqt_marmot*. We also compare here the use of measurement error mitigation for the *ibmq_manila* hardware (green marker), which reveals an improvement in accuracy.

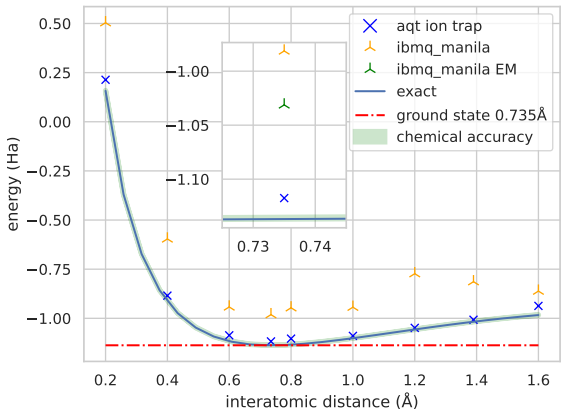


Fig. 8. Ground state energy for different interatomic distances in Ångström. The minimum of the VQE optimization is taken for this plot.

IV. DISCUSSION

Quantum computation in the near future has restrictions regarding the overall number of operations that can be executed. Reducing circuit complexity is one of the motivations for benchmarking and application-based comparison, as we can identify the optimal software/hardware pairing. Comparing different quantum hardware processors, but also quantum algorithms, encompasses multiple interconnected aspects. The accuracy of quantum gate operations, the number of available qubits, the speed of primitive gates, and the duration of coherence times are all crucial fundamental measures for quantum computing hardware at a larger scale. Still, it remains challenging to predict how an algorithm will perform on a particular hardware concept based solely on hardware-close benchmarks.

Here we have proposed the VQE in the context of quantum chemistry as a comparison basis. The ansatz used is problem-agnostic, so we can anticipate comparable performance for other minimization problems involving a sum of expectation values (such as Ising models). Our results demonstrate that all-to-all connectivity as present in ion trap hardware can

significantly decrease the number of non-local gates required. We hypothesize that the optimized energy difference may be due to the impact of SWAP gates (which require 2-qubits gates) on the VQE’s performance. Indeed, the cost of 2-qubit gates is much higher, both in terms of fidelity and execution time.

One other important aspect of benchmarking to consider is the time taken by the algorithm. When using variational algorithms we need to consider both the classical and the quantum execution times. The way the user interfaces with the different quantum hardware platforms can vary. The quantum systems of IBM Q can be accessed via cloud, which is not the case yet for *aqt_marmot*. Running algorithms through cloud services inherently introduces additional time overhead, but also makes the systems widely available. Besides the access time, the time that the quantum algorithm itself requires is more of interest to the study presented here. Here, a notable difference is observed between the two backends. Based on calibration data, it is known that *ibmq_manila* gate times are approximately 10^3 times faster than *aqt_marmot*, which naturally results in a difference in quantum runtime. Another important factor to take into account is the coherence to gate time ratio. This metric not only quantifies the number of operations a quantum system can perform, but also reflects the number of gates that can be execute before the quantum state decays. The *aqt_marmot* backend, with its extended coherence time, can still execute meaningful operations despite having slower gate times.

It is worth noting that this study primarily focused on assessing the quality of the solution and did not delve into evaluating the scalability of the hardware. However, increasing the size of the molecule would directly increase the number of qubits required for the simulation, itself linearly increasing the number of SWAP operations required to connect the first and last qubit in superconducting hardware.

V. CONCLUSION

In conclusion, this study aimed to compare two specific quantum processors in the context of an example application from quantum chemistry using VQE. We have demonstrated that the VQE serves as a suitable benchmark for evaluating application-centered hardware performance. Specifically, we focused on assessing the solution quality achieved by two specific quantum processors, *ibmq_manila* and *aqt_marmot*. Results reported in this paper exclusively apply to these two specific processor types and the specific quantum algorithm considered, and cannot directly be generalized to other quantum processors. Nevertheless, the findings contribute to the understanding of the capabilities and limitations of different

quantum processors for executing variational algorithms. We hope that this study triggers further investigations into how different quantum hardware technology concepts perform for different quantum algorithms and application examples.

VI. ACKNOWLEDGEMENTS

We would like to thank Alexander Erhard and Christian Sommer for their help running the algorithms on the AQT processor *aqt_marmot* and providing the details of the backend. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of neither IBM nor AQT. The results of this work can also not be used as recommendation to use the one or the other quantum hardware technology, as the results are processor specific, calibration specific and depend on the precise quantum algorithm used, as well as depend on the precise classical optimizer.

VII. APPENDIX

A. Optimizers

In the following, we briefly introduce the three optimizers being tested and a plot of the optimization process using the *qasm_simulator*.

- Nelder-Mead is a widely used gradient-free optimizer for classical optimization problems. It evaluates the loss function at the vertices of a simplex, updates the simplex based on the results, and continues iterating until it converges to the minimum of the loss function.
- The NFT algorithm optimizes parameters one at a time by utilizing the sine curve behavior of the loss function for each parameter. It requires two to three evaluations per iteration, based on a hyperparameter.
- Simultaneous Perturbation Stochastic Approximation (SPSA) is a gradient descent approximation. The gradient of a function is approximated by perturbing the parameters in a random way and using the resulting function evaluations to update the parameter estimates.

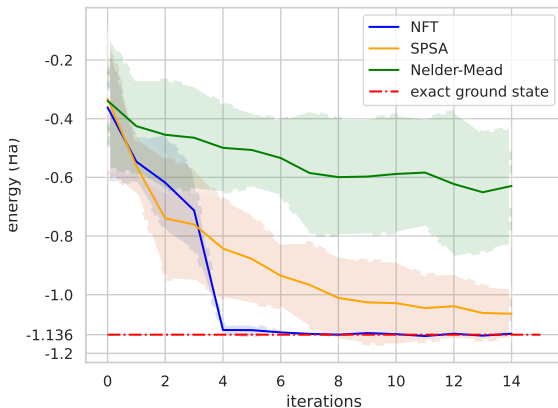


Fig. 9. Popular optimizer comparison, the choice of these 3 optimizers is motivated by the work in [20].

B. Hamiltonian Prefactors

The Hamiltonian is generated using Qiskit's PySCF extension [12].

$$\begin{aligned}
 c_1 &= -0.81054 \\
 c_2 &= 0.16614 \\
 c_3 &= 0.16892 \\
 c_4 &= 0.17218 \\
 c_5 &= -0.22573 \\
 c_6 &= 0.12091 \\
 c_7 &= 0.166145 \\
 c_8 &= 0.04523
 \end{aligned}$$

REFERENCES

- [1] Ryan Babbush, Nathan Wiebe, Jarrod McClean, James McClain, Hartmut Neven, and Garnet Kin-Lic Chan. Low-depth quantum simulation of materials. *Phys. Rev. X*, 8:011044, Mar 2018.
- [2] He Ma, Marco Govoni, and Giulia Galli. Quantum simulations of materials on near-term quantum computers. *npj Computational Materials*, 6(1):85, Jul 2020.
- [3] Sam McArdle, Suguru Endo, Alán Aspuru-Guzik, Simon C. Benjamin, and Xiao Yuan. Quantum computational chemistry. *Rev. Mod. Phys.*, 92:015003, Mar 2020.
- [4] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.
- [5] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, et al. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5(1):4213, Jul 2014.
- [6] Alain Delgado, Juan Miguel Arrazola, Soran Jahangiri, Zeyue Niu, Josh Izaac, et al. Variational quantum algorithm for molecular geometry optimization. *Phys. Rev. A*, 104:052402, Nov 2021.
- [7] Nishikanta Mohanty, Bikash K. Behera, and Christopher Ferrie. Analysis of the vehicle routing problem solved via hybrid quantum algorithms in presence of noisy channels, 2023. arXiv:2205.07630.
- [8] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C. Bardin, et al. Quantum supremacy using a programmable superconducting processor. *Nature*, 574(7779):505–510, Oct 2019.
- [9] Samson Wang, Enrico Fontana, M. Cerezo, Kunal Sharma, Akira Sone, et al. Noise-induced barren plateaus in variational quantum algorithms. *Nature Communications*, 12(1):6961, Nov 2021.
- [10] *Alpine Quantum Technologies, AQT*. <https://www.aqt.eu>, 2023.
- [11] *IBM Quantum*. <https://quantum-computing.ibm.com/>, 2021.
- [12] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023.
- [13] *AQT coherence time*. <https://www.aqt.eu/quantum-memory-lifetime/>, 2023.
- [14] Jules Tilly, Hongxiang Chen, Shuxiang Cao, Dario Picozzi, Kanav Setia, Ying Li, et al. The variational quantum eigensolver: A review of methods and best practices. *Physics Reports*, 986:1–128, 2022.
- [15] M. Born and R. Oppenheimer. Zur Quantentheorie der Molekeln. *Annalen der Physik*, 389(20):457–484, 1927.
- [16] P. Jordan and E. Wigner. Über das Paulische Äquivalenzverbot. *Zeitschrift für Physik*, 47(9):631–651, Sep 1928.
- [17] Ken M. Nakanishi, Keisuke Fujii, and Syngye Todo. Sequential minimal optimization for quantum-classical hybrid algorithms. *Phys. Rev. Res.*, 2:043158, Oct 2020.
- [18] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, 01 1965.
- [19] Spall, J. C. Implementation of the simultaneous perturbation algorithm for stochastic optimization. *IEEE Trans. Aerosp. Electron. Syst.*, 34(3):817–823, 1998.
- [20] Marita Oliv, Andrea Matic, Thomas Messerer, and Jeanette Miriam Lorenz. Evaluating the impact of noise on the performance of the variational quantum eigensolver, 2022. arXiv:2209.12803.
- [21] W. J. Hehre, R. F. Stewart, and J. A. Pople. Self-Consistent Molecular-Orbital Methods. I. Use of Gaussian Expansions of Slater-Type Atomic Orbitals. *The Journal of Chemical Physics*, 51(6):2657–2664, 09 2003.