A QVE implementation on a 2-qubit problem

Robin Giroud

May 18, 2021

Contents

1 Introduction 1

2 Theory 1
- 2.1 The Quantum Variational Eigensolver 1
- 2.2 Computation of $\langle H_i \rangle$ 2

3 Implementation of the QVE 2
- 3.1 The problem 2
- 3.2 Computation of $\langle H_i \rangle$ 3
- 3.3 The ansätze 3
  - 3.3.1 α-ansatz 3
  - 3.3.2 general ansatz 4
- 3.4 Terminaison condition 5

4 Results 5
- 4.1 Simulator without noise 5
- 4.2 Simulator with noise and errors . . . 7
- 4.3 Measurement error mitigation 7

5 Conclusion 8

6 Appendix 8

1 Introduction

One big topic of the quantum information world is about quantum algorithms. Among them, a class of algorithm mixes a quantum part with a classical part. In this work, one of those algorithms, called the Quantum Variational Eigensolver (short QVE) is being studied and implemented. It will raise a problem related to another big branch in the quantum information theory, which is the initialization of quantum states in an hardware. Finally, the results will bring a small discussion on the errors that occur with the real quantum computer, and a way to solve one of these kind of errors will be implemented. This will allow to have a glimpse to the world of quantum software engineering.

2 Theory

2.1 The Quantum Variational Eigensolver

The Quantum Variational Eigensolver (short QVE) is an algorithm that belongs to the class of hybrid quantum-classical algorithm.

Let H be a Hamiltonian that can be decomposed in a finite sum of tensor product of Pauli operators:

$$H = \sum_{i,\alpha} h^i_\alpha \sigma^i_\alpha + \sum_{i,j,\alpha,\beta} h^{ij}_{\alpha\beta} \sigma^i_\alpha \otimes \sigma^j_\beta + ...$$

where α, β, ... $\in$ {x,y,z} and i,j,... denotes the subspace which is acted on.

The QVE applies classical optimization methods to minimize the expectation value $\langle \psi(\{\alpha\}) | H | \psi(\{\alpha\}) \rangle$ of an ansatz state $|\psi(\{\alpha\})\rangle$, where $\{\alpha\}$ is the set of parameters of the ansatz.

By linearity, the expectation value is equal to

$$\langle \psi(\{\alpha\}) | H | \psi(\{\alpha\}) \rangle = \sum_{i,\alpha} h^i_\alpha \langle \psi(\{\alpha\}) | \sigma^i_\alpha | \psi(\{\alpha\}) \rangle$$
$$+ \sum_{i,j,\alpha,\beta} h^{ij}_{\alpha\beta} \langle \psi(\{\alpha\}) | \sigma^i_\alpha \otimes \sigma^j_\beta | \psi(\{\alpha\}) \rangle + ...$$

The variational principle of quantum mechanics ensures that $\langle \psi | H | \psi \rangle \geq E_0$, where E_0 is the energy of the ground state of the system, and furthermore it is exactly equal to $E_0 \Leftrightarrow |\psi\rangle$ is the ground state. By minimizing the expectation value, the QVE finds the set of parameters such that ansatz state is as close as possible to the ground state, and it gives an upper bound on the value of the ground state energy.

The different steps of the QVE are displayed on Figure 1. The quantum part of the algorithm prepares the initial $|0\rangle \otimes \ldots \otimes |0\rangle$ of the quantum circuit into the ansatz state $|\psi(\{\alpha\})\rangle$ for a given set of parameters (part A in the scheme). It then computes the expectation value for each term in the decomposition of H (part B in the scheme). The classical part takes all those computations and get the final expectation value. Finally, it computes a new set of parameters with classical optimization algorithm (such as gradient descent, which is the method choosed for this work) and the whole process is repeated with this new set of parameters. This goes on until some precision ϵ is met.

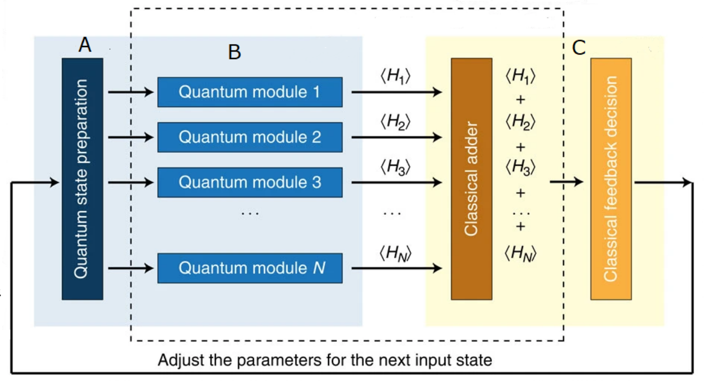
A
B
C
Quantum state preparation
Quantum module 1
Quantum module 2
Quantum module 3
Quantum module N
⟨H1⟩
⟨H2⟩
⟨H3⟩
⟨HN⟩
Classical adder
⟨H1⟩ + ⟨H2⟩ + ⟨H3⟩ + ... + ⟨HN⟩
Classical feedback decision
Adjust the parameters for the next input state

Figure 1: [1] *Scheme of the VQE.*

It is to notice that each of these 'Quantum modules' in Figure 1 need to be run a certain number of times N, as the expectation value $\langle H_i \rangle$ is computed based on the measure at the end of the circuit (see Section 2.2). The theoretical upper bound on N to reach a certain accuracy ϵ is given by

$$N \leq \left(\frac{\sum_{i,j,\ldots,\alpha,\beta,\ldots} |h^{ij\ldots}_{\alpha\beta\ldots}|}{\epsilon} \right)^2 [2] \qquad (1)$$

2.2 Computation of $\langle H_i \rangle$

The computation of $\langle H_i \rangle$ for a given input state, where H_i is a tensor product of Pauli operators, is done in the following way. The state is rotated with the rotation to go from the computational basis into the H_i eigenbasis. Then a measurement of the computational basis occurs, in order to obtain a set of probabilities, which is finally mixed together (addition and substraction) according to a theoretical computation of $\langle H_i \rangle$ done before.

This set of probabilities obviously depends on the number of shots N done by the quantum computer and need to be large enough to obtain precise expectation values.

A practical example of such a computation is showed in the Section 3.2.

3 Implementation of the QVE

In this section, a specific problem is laid down so that the QVE can be implemented and tested.

The algorithm has been implemented with Qiskit (an open-source framework for quantum computing). The Qiskit implementation has been run on a (perfect*) quantum computer simulator and later on a noisy quantum computer simulator, giving the experimental results. Unfortunately, it has not been tested fully with the real computer, as it requires too many computations. The theoretical results are known, as the problem dealt with in this report is fully known and small enough to be computed directly with Matlab. This will let a comparison emerge between the theoretical and experimental results, in order to check if the set of parameters given by the QVE indeed matches the theoretical results obtained with Matlab, and if the convergence of N follows (1). A discussion on the errors and how to eventually get rid of it will also emerge.

*Note that 'perfect' means that the simulator does not try to reproduce the noise of the real machine.

3.1 The problem

Let's consider 2 qubits and the Hamiltonian H:

$$H = \frac{1}{2}\left(\sigma_x^1 + \sigma_x^2\right) + \sigma_z^1 \otimes \sigma_z^2$$

This Hamiltonian is equivalent to

$$H = \begin{pmatrix} 1 & 0.5 & 0.5 & 0 \\ 0.5 & -1 & 0 & 0.5 \\ 0.5 & 0 & -1 & 0.5 \\ 0 & 0.5 & 0.5 & 1 \end{pmatrix}$$

and its lowest eigenvalue is $\lambda_{low} = -\sqrt{2}$ with corresponding normalized eigenstate $|\psi_{low}\rangle = |-0.27, 0.65, 0.65, -0.27\rangle$.

3.2 Computation of $\langle H_i \rangle$

In this section, a practical example of what does each 'Quantum module' of Figure 1 is showed.

Let's consider the first term of H, i.e. $H_1 = \sigma_x^1 = \sigma_x \otimes \mathbb{I} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$ and a general input state $|\psi\rangle = |a, b, c, d\rangle$ on 2 qubits.

The goal is to obtain $\langle\psi| H_1 |\psi\rangle$ with the quantum computer.

First, it is needed to know what it is theoretically equal to: $\langle\psi| H_1 |\psi\rangle = 2(ac + bd)$.

Then, to obtain this with the quantum computer, it is needed to rotate the state $|\psi\rangle$ in the eigenbasis of H_1. In this case, this is done by adding an Hadamard gate $\frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ to the first qubit. The final state is then $|\psi_{final}\rangle = \frac{1}{\sqrt{2}}|a + c, b + d, a - c, b - d\rangle$.

By measuring this $|\psi_{final}\rangle$ in the computational basis, one obtain the state $|1000\rangle$ with probability $p_1 = \frac{1}{2}(a + c)^2$, and so on for the 3 other computational states and probabilities p_i, $i \in$ 2,3,4.

By doing again the experiment N times and letting N grow, one can get a more precise probabilities histogram, such as the one presented as example in Figure 2.

Knowing those p_i's, it is easy to check that $\langle\psi| H_1 |\psi\rangle = 2(ac + bd) = p_1 - p_2 + p_3 - p_4$.

In general, the eigenbasis rotation must be done as follows: add an Hadamard gate on the qubits acted on by σ_x, add an Hadamard gate followed by S-gate (phase gate) on the qubits acted on by σ_y and do nothing on the qubits acted on by σ_z (as the eigenbasis is already the computational basis).

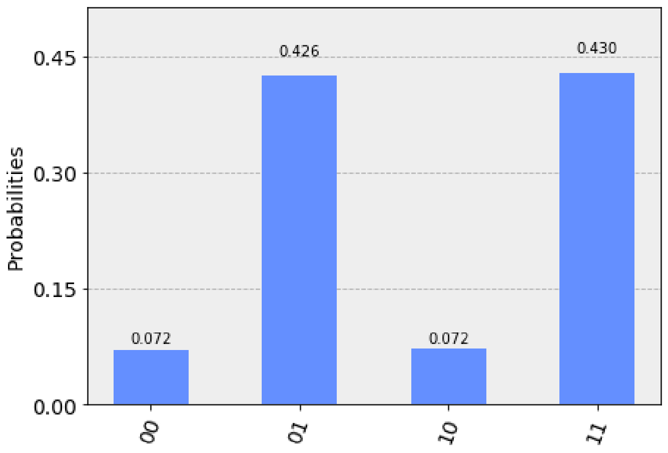

Figure 2: *Example of probability histogram.*

3.3 The ansätze

One must now choose an ansatz state in order to run the implementation of the QVE. In this work, two different ansätze have been tried. The first one is easier, and depends only on one variable α, while the second is the most general for two qubits.

It is to note that the QVE needs to rotate the usual $|0...0\rangle$ quantum circuit starting state into the desired ansatz state. This is related to a big actual open problem, known as the initialization of quantum circuit, and it has been solved in this work for the two different ansätze.

3.3.1 α-ansatz

The α-ansatz is the following:

$$|\psi_\alpha(\alpha)\rangle = \frac{1}{norm(\alpha)}|1 - \alpha, 1, 1, 1 - \alpha\rangle$$

.

where $norm(\alpha) = \sqrt{2(1 + (1 - \alpha)^2)}$. Notice that $|\psi_\alpha(\alpha)\rangle = |\psi_{low}\rangle \Leftrightarrow \alpha = \sqrt{2}$.

The initialization circuit for this ansatz is shown in Figure 3, where

$$\theta(\alpha) = 2\cos^{-1}\left(\sqrt{2}\left(\frac{1 - \alpha}{norm(\alpha)}\right)\right) \quad (2)$$

In order to run the QVE with this ansatz, it remains to choose and implement the classical optimization algorithm. As mentioned in the Section

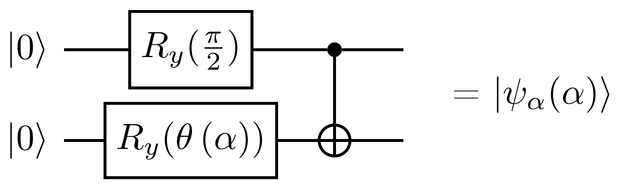

Figure 3: *Circuit used to initialize the α-ansatz, where $\theta(\alpha)$ is given by (2).*

2.1, a gradient descent is used in this work. The initial choice of α is totally arbitrary, and after an iteration of the QVE, the new α_{new} is given by

$$\begin{aligned}\alpha_{new} &= \alpha - d\alpha\frac{\partial}{\partial\alpha}\left(\langle\psi_\alpha(\alpha)|\, H\, |\psi_\alpha(\alpha)\rangle\right)\\ &= \alpha - d\alpha\frac{2\alpha^2-4}{(\alpha^2-2\alpha+2)^2}\end{aligned}$$

where $d\alpha$ is an arbitrary 'step' parameter.

3.3.2 general ansatz

The general ansatz is the following:

$$|\psi_{gen}(a,b,c,d)\rangle = \frac{1}{norm(a,b,c,d)}\,|a,b,c,d\rangle$$

.

where $norm(a,b,c,d) = \sqrt{a^2+b^2+c^2+d^2} \equiv norm$.

The initialization circuit for this general ansatz is more complicated. The goal is to obtain a circuit corresponding to a unitary operation on two qubits whose first matrix column must be the vector

$$\frac{1}{norm}\begin{pmatrix}a\\b\\c\\d\end{pmatrix}$$

so that when applying this circuit to the initial $|0\rangle\otimes|0\rangle$ state of a quantum circuit, one exactly obtain $|\psi_{gen}(a,b,c,d)\rangle$.

One can check that the unitary U given by

$$U = 2\,|v\rangle\langle v| - \mathbb{I}$$

where

$$|v\rangle = \frac{|\psi_{gen}(a,b,c,d)\rangle + |0\rangle}{\sqrt{2(1+\frac{a}{norm})}}$$

does the job.

At this point, using the general method given in the book of Nielsen and Chuang (4.5.1 page 189 of [3]), one can obtain a decomposition of U as a product of 2-levels unitaries: $U = FEDCBA$. The Matlab code to obtain these unitaries is given in the appendix. Notice that as $F, E, ...$ are 2-level matrices, one can get 2x2 matrices (call them $F', E', ...$) that only contain the information on the 2 states where it is not trivially acted on. As they are 2x2 matrices, one can easily obtain a decomposition into three rotation gates (Theorem 4.1 page 175 of [3]) for each of them. A controlled version of them is then exactly the same as controlling the three rotation gates one after the other. The Matlab code for this 1-qubit gate decomposition is also provided in the appendix.

Figure 4 displays the final circuit implementing the unitary U, using those 2x2 $F', E', ...$ matrices.

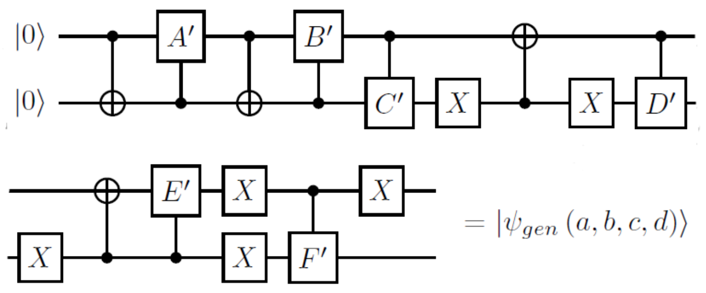

Figure 4: *Circuit used to initialize the general ansatz.*

In order to run the QVE with this ansatz, it remains to implement the corresponding gradient descent. The initial choice of the parameters $\{a,b,c,d\}$ is totally arbitrary, and after an iteration of the QVE, the new parameters are given by:

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}_{new} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} - \mathrm{dx}\overrightarrow{\nabla}\left(\langle\psi_{gen}| H |\psi_{gen}\rangle\right) = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} - \frac{\mathrm{dx}}{norm^2}\begin{pmatrix} norm(2a+b+c) - 2af(a,b,c,d) \\ norm(a-2b+d) - 2bf(a,b,c,d) \\ norm(a-2c+d) - 2cf(a,b,c,d) \\ norm(b+c+2d) - 2df(a,b,c,d) \end{pmatrix}$$

where

$$f(a,b,c,d) = a^2 + ab + ac - b^2 + bd - c^2 + cd + d^2$$

and dx is an arbitrary 'step' parameter.

3.4 Terminaison condition

In general, the lowest eigenvalue and its groundstate are not known. It is then impossible to stop the algorithm when some precision ϵ is met w.r.t. to those. The two choices left are then: stopping the QVE when either the difference between the expectation values or the set of parameters given in two iterations in a row is smaller than an arbitrary ϵ.

In this work, the latter as been chosen, with $\epsilon = 1e^{-4}$.

4 Results

Note that the main results shown in this part are the ones obtained while running the QVE with the α-ansatz, as there is only one parameter and it is thus easier to plot graphs. Nevertheless, all results shown here are also working for the general ansatz.

4.1 Simulator without noise

The first objective is to check that the QVE works and that the convergence of N follows equation (1).

In order to observe this, the α-ansatz is used, with the arbitrary initial α set to 3, and the QVE has been run with increasing N. To have more precise results, for a given N, the QVE has been run 100 times, so that for each step in the QVE (i.e. for each α), 100 values of the expectation value is computed and one can get rid of the statistical fluctuations. Figure 5, Figure 6, Figure 7 and Figure 8 display examples of the results with increasing N.

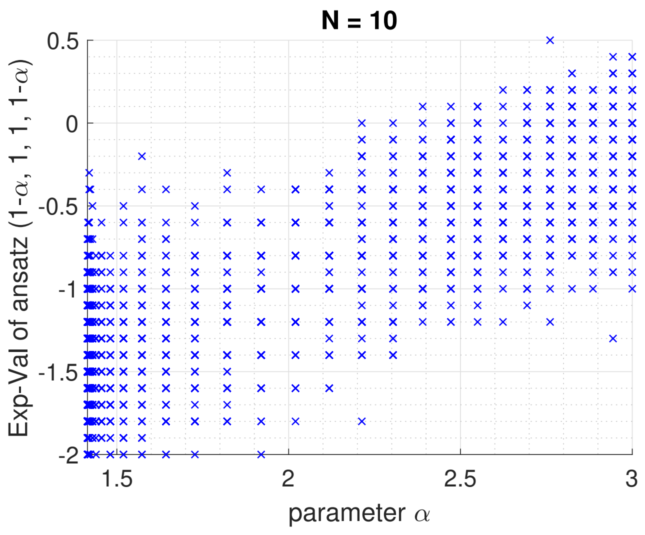

Figure 5: *100 QVE results for N = 10 with initial* $\alpha = 3$.

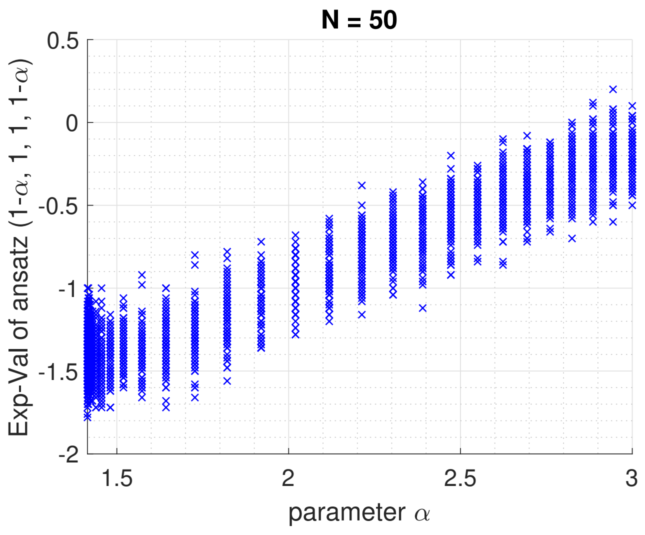

Figure 6: *100 QVE results for N = 50 with initial* $\alpha = 3$.

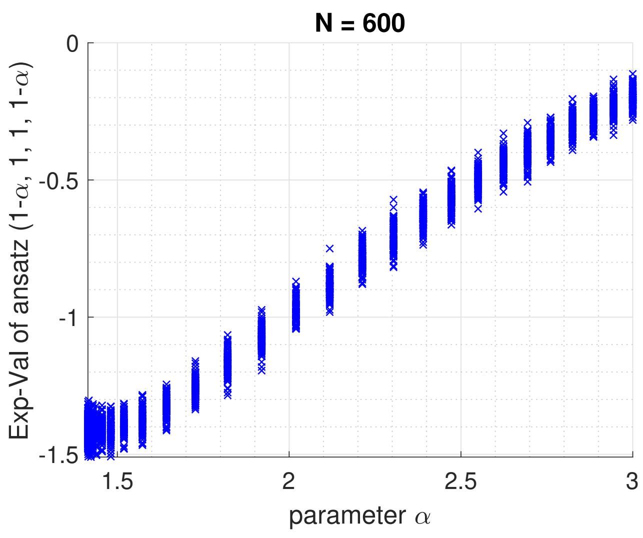

Figure 7: *100 QVE results for $N = 600$ with initial $\alpha = 3$.*

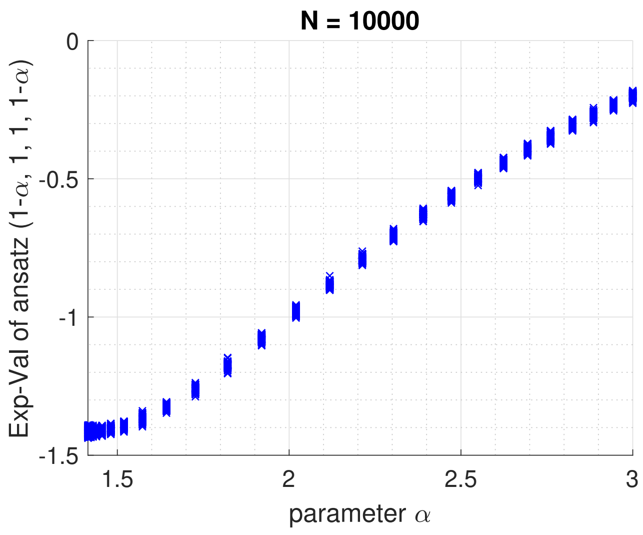

Figure 8: *100 QVE results for $N = 10000$ with initial $\alpha = 3$.*

The first thing to observe is that, as expected, the α parameter starts at value 3 and goes the way down to the value $\sqrt{2}$, while the corresponding expectation value get closer to $\lambda_{low} = -\sqrt{2}$. Based on the informations from Section 3.3.1, this shows that the algorithm works as intended.

Figure 9 and Figure 10 shows the mean and standard deviation of the normal distribution obtained through the 100 experiments for different N.

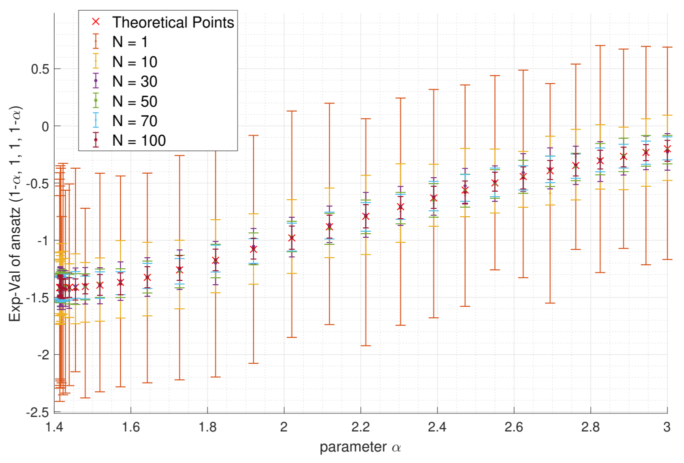

Figure 9: *Mean and std of a sample of 100 runs of the QVE for different N's.*

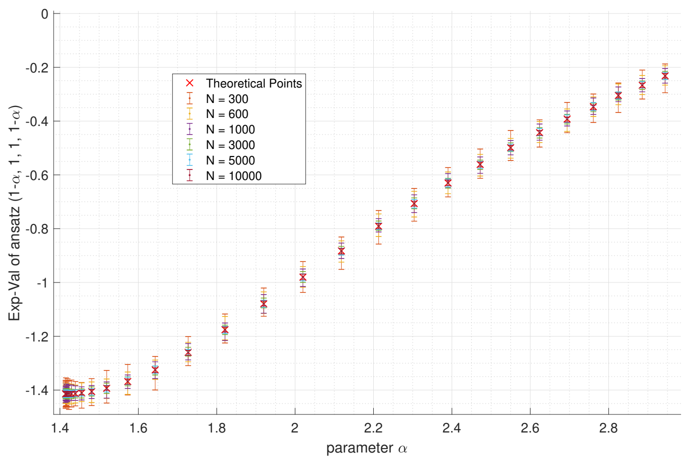

Figure 10: *Mean and std of a sample of 100 runs of the QVE for different N's.*

One can indeed observe that as N increases, the expectation value computed by the QVE gets more precise. Figure 11 shows that this gain in precision follows the bound given by equation (1).

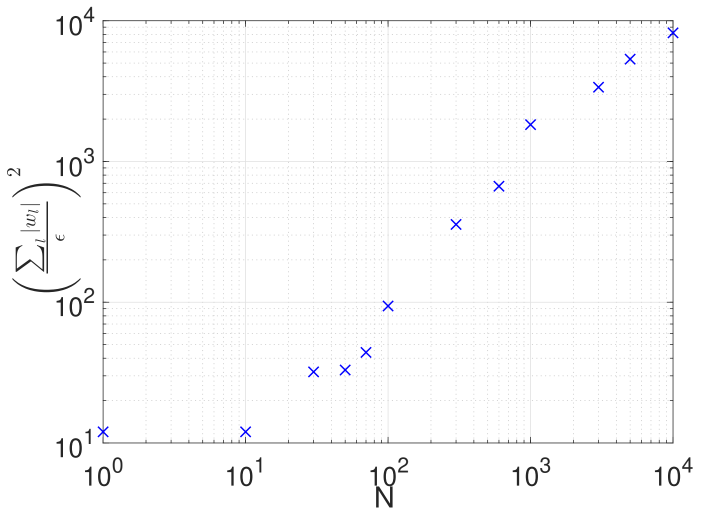

Figure 11: *Upper bound on N given by equation (1).*

4.2 Simulator with noise and errors

In this part, the simulator is not perfect anymore and try to simulate the noise of the real quantum computer. Figure 12 shows the comparison between the two kind of simulators.

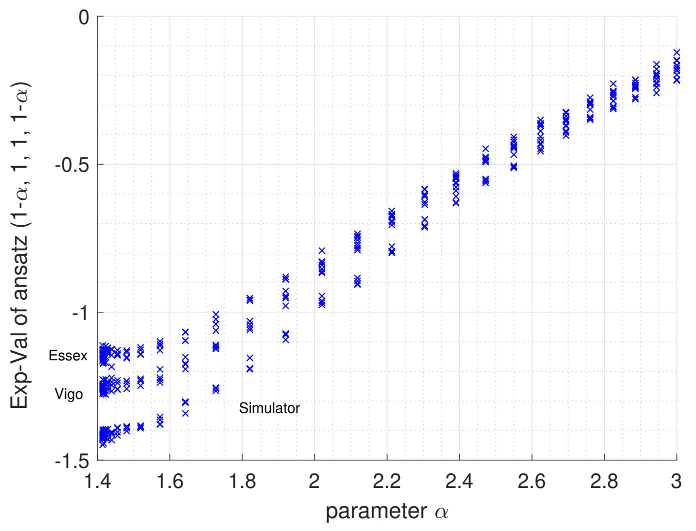

Figure 12: *Results obtained with the QVE while using the perfect simulator and two different noisy simulator. (Essex and Vigo are two free-to-use quantum computers from IBM lab)*

It is noticeable that while having noise, the result isn't correct and seems to have a bias. This is due to two kind of errors that occurs while using quantum computer.

The first one is the measurement error, which exists because the hardware has a probability to measure a '0' while the actual state is '1', and the other way around.

The second one is the gate error, which is a probability that a gate in the quantum circuit doesn't do exactly what it should (f.e. a X gate that fails to switch the states '0' and '1') and this leads to a wrong answer at the end of the QVE. This kind of error especially occurs on 2-qubits gates, like control-gates.

In this implementation of the QVE, as there is not a lot of 2-qubits gates (4 CNOT in the general ansatz, 1 CNOT in the α-ansatz), this bias observed is due to the measurement error. It can be solved by using an error mitigation trick.

4.3 Measurement error mitigation

In order to get almost fully rid of the measurement error, one can suppose the following: the measurement process, when it occurs, does first a perfect measurement to get $|\psi_{perfect}\rangle$. Afterwards, it outputs the state $|\psi_{noisy}\rangle = M|\psi_{perfect}\rangle$, where M is a matrix of elements $(m)_{ij}$, corresponding to the probability that the i'th component of $|\psi_{perfect}\rangle$ is wrongly measured and gives some noise in the j'th component of $|\psi_{noisy}\rangle$.

Each column of this M matrix must then be a discrete probability distribution (i.e. $\sum_i m_{ij} = 1 \;\; \forall j$) and $M = \mathbb{I}$ in a system without noise.

For a given quantum computer or a given quantum noisy simulator, one can easily obtain this matrix M, for instance by running a circuit containing an equal superposition of all states and observing the differences between the measurement and what is known and expected.

Having this M, one can inverse it and find the circuit implementing M^{-1} (for instance with the method of the N.&C. [3]).

Adding this circuit at the end of the implementation of the QVE, one gets almost fully rid of the measurement error, although everything is done on a noisy hardware.

In this work, this mitigation process has been implemented with a package in the AER part of Qiskit.

5 Conclusion

Through this work, it has been made possible to have a quick touch to the quantum software engineering world, and in particular to observe that the Quantum Variational Eigensolver is a cool working hybrid quantum-classical algorithm that let's you find the closest approximation to the ground energy of a system. Through this work, it has been fully implemented for a two qubits system, including the initialization of the quantum state, which is still today an open problem (for bigger systems) in the quantum information world. Finally, a discussion on the errors made by the QVE (and more generally by a quantum computer) has been briefly discussed.

6 Appendix

The following Matlab files are included in the report.

'decompNC.m' which is the code running the decomposition provided in the NC book. (4.5.1 page 189 of [3])

'decompUnitaries2x2.m' which is the code providing the decomposition of a 1-qubit gate into 3 rotations gates, so that it can be implemented in a quantum circuit. (Th. 4.1 page 175 of [3])

'FindU.m' which is the code to run in order to find the decomposition from the Section **3.3.2**. The parameters a,b,c,d are variables in this program.

References

[1] Peruzzo A., ... (2013) A variational eigenvalue solver on a quantum processor. `https://arxiv.org/abs/1304.3061`

[2] Higgot O., Wang D., Brierley S. (2019) Variational Quantum Computation of Excited States. `https://arxiv.org/abs/1805.08138`

[3] Nielsen M., Chuang I. (2000) Quantum Computation and Quantum Information