Quantum Approximate Optimization with n-spin 1D Ising Model

Marti Vives

Junior Paper
Department of Physics
Princeton University
Fall 2018

Advised by Professor Hakan Türeci

This paper represents my work in accordance with University regulations.

/s/ Marti Vives

Abstract

We seek to approximate the minimum of cost functions given by the n-spin 1D Ising Model using QAOA (Quantum Approximate Optimization Algorithm) on actual quantum hardware through IBM Q's Cloud Service. The motivation behind this project lies at the applicability of the Ising problem; many classical NP-Hard problems map to finding the ground state of an appropriate Ising model and these are promising candidates to be addressed by true quantum computers. We detail how the QAOA algorithm can be implemented using quantum gates, and simulate QAOA written in Quantum Assembly Language (*qasm*) on a classical computer. Finally we run QAOA on an actual quantum computer accessed via the IBM Q network and compare performance with *qasm* simulations.

Contents

1 Introduction

Quantum Computers can potentially surpass the current capabilities of classical computers and provide feasible solutions to previously intractable and NP-hard problems. The potential speed-up is achieved by the availability of a Hilbert space that grows exponentially with the number of quantum bits (qubits) due to the ability to access quantum mechanical superpositions of the classical bits. The catch is that internal states are susceptible to errors and decoherence, and we are currently unable to efficiently implement quantum error correction to yield fault-tolerant systems.

Current quantum systems referred to as Noisy Intermediate Scale Quantum (NISQ) are non-fault tolerant, with commercially available devices ranging from 5 to 79 qubits with varying orders of connectivity [1]. Companies like IBM and Rigetti provide Cloud Based services on these devices, allowing for independent research to be carried out. In this project we will use IBM Q Cloud Service, which gives users access to up to 15-qubit devices.

1.1 Quantum Heuristics

We will review recent progress in algorithms for Near Term Quantum Hardware, focusing on quantum heuristics. The limited number of reliable quantum operations in NISQ devices due to errors and decoherence make hybrid quantum-classical algorithms the best use of current quantum hardware. The most relevant such algorithms are the Quantum Optimization Approximation Algorithm (QAOA) and the Variational Quantum Eigensolver (VQE), both of which comprise of a quantum routine that prepares a given state according to variational parameters and a classical routine that takes in many measurements and carries out a classical optimization. These algorithms have a wide range of applications, for instance, the VQE is mainly used in quantum chemistry problems, such as finding the binding energy of a specific compound. The algorithms then translates the given interaction Hamiltonian to a fermionic problem that can then be solved in quantum hardware.

We will focus on the QAOA, which has many applications in combinatorial optimization, for instance in solving the Max-Cut problem [2] or the Traveling Salesman Problem (TSP) [3]. To understand why the algorithm works it is useful to get an intuition from the Quantum Adiabatic Algorithm (QAA). We will discuss the QAA in Section 2.2 and follow with the QAOA in section 2.3, comparing the two in section 2.4. The implementation will follow in section 3 first studying simulated results and then focusing on results from IBM Q's quantum devices.

2 Theoretical background

2.1 1D Ising Model

The 1D Ising model is a mathematical model of ferromagnetism, representing n atomic spins that can pairwise interact in one dimension. The energy configuration for $\hat{\sigma}^z$ spins in an external magnetic field in the x direction is given by the following Hamiltonian:

$$\hat{H} = \sum_{j=1}^{n} J_j \hat{\sigma}_j^z \hat{\sigma}_{j+1}^z + \sum_{j=1}^{n} h_j \hat{\sigma}_j^x \tag{1}$$

Where $J_j < 0 \ \forall \ j$ implies the ferromagnetic interaction, and $h_j < 0$ implies spin j tends to line up in the negative x direction. Throughout the work we will focus on optimizing a minimum for the 1D Ising model in the absence of a magnetic field, defining the cost Hamiltonian:

$$\hat{H}_c = \sum_{j=1}^{n} J_j \hat{\sigma}_j^z \hat{\sigma}_{j+1}^z \tag{2}$$

Keeping in mind general optimization applications, we will generally be interested in the case where J_j are uncorrelated and may have different signs. This problem falls in the NP-Complete group, and quantum approximations tools will prove useful in attempting to solve it. One such tool is the quantum adiabatic algorithm.

2.2 Quantum Adiabatic Algorithm

2.2.1 Theory

The foundation of the QAA lies in the Adiabatic Theorem, which states: *A physical system remains in its instantaneous eigenstate if a given perturbation is acting on it slowly enough and if there is a gap between the eigenvalue and the rest of the Hamiltonian's spectrum* [4].

Consider the problem of obtaining the ground state of a complicated hamiltonian $\hat{H}_f$ in an N dimensional Hilbert space, and prepare a system in the ground state of a simple reference hamiltonian $\hat{H}_{ref}$. Now define a smooth one-parameter time evolution $\hat{H}(s)$ for $0 \leq s \leq 1$, with T controlling the evolution rate of $\hat{H}(t)$:

$$\hat{H}(t) = \lambda(t/T)\hat{H}_f + (1 - \lambda(t/T))\hat{H}_{ref} \tag{3}$$

Define the instantaneous eigenstates and eigenenergies of the evolving hamiltonian by $\hat{H}(s)\,|n, s\rangle = E_n(s)\,|n, s\rangle$, where $E_0(s) \leq E_1(s) \leq ... \leq E_{N-1}(s)$. Then from the adiabatic theorem, if $E_1(s) - E_0(s) > 0\ \forall s \in [0, 1]$ and T is large enough then we expect to stay in the ground state of the evolving hamiltonian and end up at the $\hat{H}_f$ ground state [5]. That is, since the ground state of $\hat{H}_f$ is by definition $|n = 0, s = 1\rangle$ and the system state at time T is $\psi(T)$, then defining F as the fidelity to reach the ground state of $\hat{H}_f$:

$$F = |\,\langle n = 0; s = 1|\psi(T)\rangle\,| \tag{4}$$

we have:

$$\lim_{T\to\infty} F = 1 \tag{5}$$

Moreover, Farhi et al. (2000) [5] proved the fidelity can be made arbitrarily close to one by choosing:

$$T \gg \frac{\mathcal{E}}{g_{min}^2} \tag{6}$$

where

$$g_{min} = \min_{s\in[0,1]} (E_1(s) - E_0(s)) \tag{7}$$

is the minimum spectral gap between the instantaneous ground state and first excited state, while

$$\mathcal{E} = \max_{s\in[0,1]} |\langle n=1;s| \, d\hat{H}/ds \, |n=1;s)\rangle| \tag{8}$$

is a measure of the how rapidly the spectral manifold evolves with s.

2.2.2 Classical simulation

To better understand the time evolution and get a sense of how the QAA works we will carry out a simulation of QAA applied to the quantum system on a classical computer (which could also run on a quantum machine using actual gate operations). We will run the QAA for the simple one spin hamiltonian:

$$\hat{H}(t) = \lambda(t/T)h\hat{\sigma}^z + (1-\lambda(t/T))h\hat{\sigma}^x \tag{9}$$

In the following, we set $h = 1$, which then sets the scale of time and energy in the evolution. The time evolution of a state $|\psi(t)\rangle$ under $\hat{H}(t)$ for a time step dt is defined as follows:

$$|\psi(t+dt)\rangle = \exp\Big\{-i\hat{H}(t)dt\Big\} \, |\psi(t)\rangle \tag{10}$$

and

$$\begin{aligned} \exp\Big\{-i\hat{H}(t)dt\Big\} = \; & \exp\{-i\lambda(t/T)\hat{\sigma}^z dt\} \exp\{-i(1-\lambda(t/T))\hat{\sigma}^x dt\} \\ & \exp\Big\{\frac{1}{2}[-i\lambda(t/T)\hat{\sigma}^z, -i(1-\lambda(t/T))\hat{\sigma}^x](dt)^2\Big\} \end{aligned} \tag{11}$$

Notice for small dt that the last exponential term can be ignored then:

$$|\psi(t+dt)\rangle \approx \exp\{-i\lambda(t/T)\hat{\sigma}^z dt\} \exp\{-i(1-\lambda(t/T))\hat{\sigma}^x dt\} |\psi(t)\rangle \tag{12}$$

The above defines a time evolution step that we can use to calculate $|\psi(T)\rangle$. Let us first choose a time step of $dt = 0.1$ and the simplest time evolution for $\lambda : [0, T] \to [0, 1]$ which is the linear case: $\lambda(t/T) = t$, then since we know the ground states of both $\hat{\sigma}^z$ and $\hat{\sigma}^x$ we can measure the fidelity in terms of T. Additionally, by fixing T we can also compare the fidelity in terms of the functional dependence of λ:

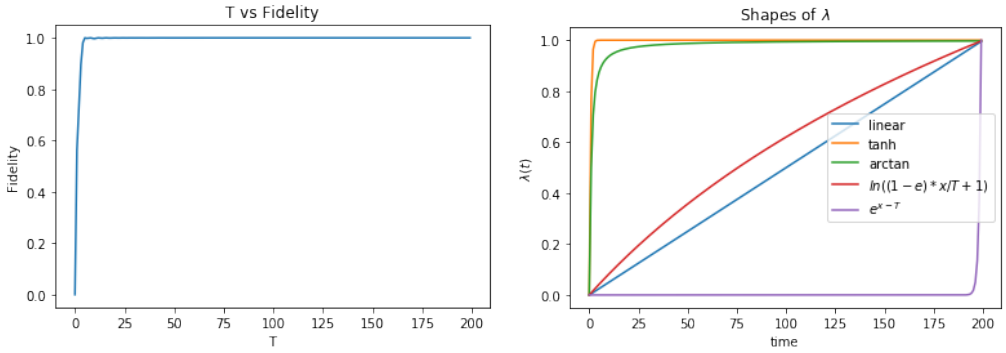


Figure 1: (Left) Fidelity of the evolving Hamiltonian with linear λ as a function of T. (Right) Different functional dependence of λ.

From Figure 1 on the left we observe the fidelity converges very fast to 1, suggesting that choosing a value of $dt = 0.1$ is sufficient to ignore the order $O(dt^2)$ in the Trotter decomposition for the time evolution of the system, and the approximation in Equation (12) holds. In the right panel, we study how the fidelity depends on the functional form of λ in Figure 1, and observe that $\lambda(x) = \ln\{(e-1)x/T+1\}$, for $x \in [0, T]$ yields a similar fidelity to the linear case (see Table 1). It is worth mentioning that profiles that involve rapid changes in λ have lower fidelity for the same T. This makes intuitive sense since these values will have a larger $\mathcal{E}$ and therefore as shown in Equation (6) this will force a longer evolution time T to reach a higher fidelity. Interestingly one could perform a variational optimization to obtain the optimal function for λ that would maximize fidelity for a fixed T; this motivates the shift towards the QAOA, which does include the latter.

Profile of $\lambda(x)$	**Fidelity**
linear	0.9999948
tanh	0.6794001
arctan	0.8666626
$\ln\{(e-1)x/T+1\}$	0.9999966
e^{x-T}	0.5229548

Table 1: Fidelity of reaching the $\hat{\sigma}^z$ ground state following time evolution of the Hamiltonian in equation (9) up to $T = 100$.

2.3 Quantum Approximate Optimization Algorithm

The QAOA finds approximate solutions to combinatorial optimization problems specified by n bits and m clauses. Consider the problem of maximazing the following objective function:

$$C(z) = \sum_{\alpha=1}^{m} C_\alpha(z) \tag{13}$$

where $z = z_1 z_2 \cdots z_n$ is the n bit string and $C_\alpha \in \{0, 1\}$ is the α constraint clause which depending on the string is either fulfilled ($C_\alpha(z) = 1$) or not ($C_\alpha(z) = 0$). Classically the simplest approach would be to query all strings, leading to $O(2^n)$ queries.

Instead, various algorithms have been developed to solve this optimization problem on a quantum computer. Working on a 2^n dimensional Hilbert space with computational basis $|z\rangle$, the objective function becomes an operator $\hat{H}_c$ and the expectation value for a given state $|\psi\rangle$ is given by $\langle\psi|\hat{H}_c|\psi\rangle$. To find the state corresponding to the largest eigenvalue we will define an initial state $|s\rangle$ in the uniform superposition of the computational basis:

$$|s\rangle = \frac{1}{\sqrt{2^n}} \sum_z |z\rangle \tag{14}$$

Then we will apply two types of unitary operators that will rotate $|s\rangle$ and try to bring it as close as possible to the lowest eigenvalue eigenstate of $\hat{H}_c$. Define

$U(\hat{H}_c, \gamma)$ for $\gamma \in [0, 2\pi]$ as

$$U(\hat{H}_c, \gamma) = \exp\Big\{-i\gamma\hat{H}_c\Big\} = \exp\Bigg\{-i\gamma\sum_{\alpha=1}^{m}\hat{H}_{c,\alpha}\Bigg\} = \prod_{\alpha=1}^{m}\exp\Big\{-i\gamma\hat{H}_{c,\alpha}\Big\} \tag{15}$$

The last step follows from the fact that since the computational basis is defined as the one that diagonalizes $\hat{H}_c$ then all the terms in the product commute in the Trotter decomposition [6]. Also define the operator $\hat{B}$ and the unitary rotation $U(\hat{B}, \beta)$ as

$$\hat{B} = \sum_{j=1}^{n}\hat{\sigma}_j^x \tag{16}$$

$$U(\hat{B}, \beta) = \exp\Big\{-i\beta\hat{B}\Big\} = \prod_{j=1}^{n}\exp\big\{-i\beta\hat{\sigma}_j^x\big\} \tag{17}$$

where the last step follows from the fact that the decomposition of $\hat{B}$ is exact since spins for different qubits commute and $\hat{\sigma}^x$ commutes with itself, so $[\hat{\sigma}_i^x, \hat{\sigma}_j^x] = 0$. Also note the resemblance of the exponential term with the rotation unitary about the x axis $R_x(2\beta)$, makes us take $2\beta \in [0, 2\pi]$, so $U(\hat{B}, \beta)$ is just applying an $R_x(2\beta)$ gate to every qubit.

Applying $U(\hat{B}, \beta)U(\hat{H}_c, \gamma)$ once might not be enough to get close to the solution, so repeat the process p times, $p \geq 1$, defining $2p$ angles $\boldsymbol{\gamma} = \gamma_1 \cdots \gamma_p$ and $\boldsymbol{\beta} = \beta_1 \cdots \beta_p$ defining the evolution on the initial state $|s\rangle$ to be:

$$|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle = U(\hat{B}, \beta_p)U(\hat{H}_c, \gamma_p)\cdots U(\hat{B}, \beta_1)U(\hat{H}_c, \gamma_1)\,|s\rangle \tag{18}$$

A p-level QAOA algorithm proceeds as follows (see Figure 2):

1. (QPU) Apply the initial state $|s\rangle$ to the n qubits
2. (CPU) Choose $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$

3. (QPU) Apply the $U(\hat{H}_c, \boldsymbol{\gamma})$ and $U(\hat{B}, \boldsymbol{\beta})$ and measure $|\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ in the computational basis

4. (CPU) Calculate the expectation $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}| \hat{H}_c |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$

5. (CPU) Repeat steps 1-4 many times and perform a classical optimization to find $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ that maximize the expectation

6. (CPU) Repeat step 5 with the current maximizing $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ to find new optimal values in the classical optimization

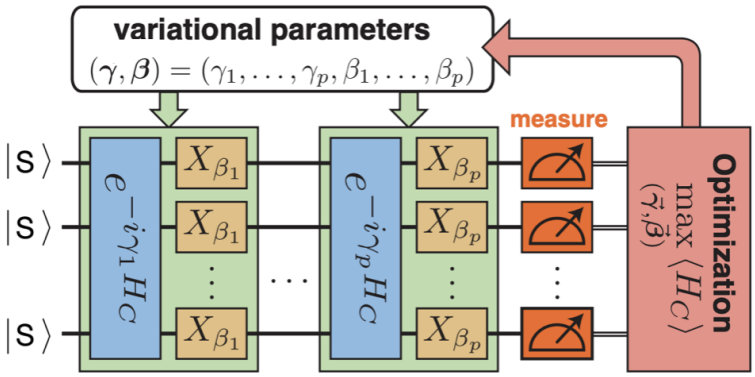


Figure 2: Diagram showcasing key steps in the QAOA algorithm, starting in the $|s\rangle$ state, choosing some initial angles $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, performing the rotation unitaries $U(\hat{H}_c, \boldsymbol{\gamma})$ and $U(\hat{B}, \boldsymbol{\beta})$, measuring the system and performing a classical optimization. Schematic from Zhou et al.(2018) [7].

2.4 The relation between QAOA and QAA

It is not immediately clear why the QAOA provides a good approximate solution to the maximization problem. Farhi et al. [6] cleverly showed that QAOA and QAA become the same for $p \to \infty$. For that consider the following hamiltonian:

$$\hat{H}(t) = (1 - t/T)\hat{B} + (t/T)\hat{H}_c \tag{19}$$

The initial state $|s\rangle$ is no other than the highest energy eigenstate of $\hat{B}$ so taking a sufficiently large T to ensure an adiabatic evolution in the QAA will yield

the largest energy eigenstate of $\hat{H}_c$. Now in the QAOA this will be equivalent to applying the $U(\hat{H}_c, \boldsymbol{\gamma})$ and $U(\hat{B}, \boldsymbol{\beta})$ unitaries with small angles to ensure a good Trotterized approximation (analogous to a small time step int the QAA) and a large p to ensure we can get as close to $\max_{\boldsymbol{\gamma},\boldsymbol{\beta}} \langle \boldsymbol{\gamma}, \boldsymbol{\beta}| \hat{H}_c |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$ as we want [6].

So why are we using QAOA if QAA ensures an optimal solution? The main reason is that QAOA requires a smaller coherent time than QAA, and some problems require a very large coherent time which makes them intractable in QAA. Moreover QAOA provides a fast approximate solution and even though it is guaranteed to provide the optimal value for $p \to \infty$ not much is known about its performance for $1 < p < \infty$ [7]. This provides an additional layer of motivation to the project.

3 QAOA on classical and quantum hardware

3.1 Problem Statement

We wish to solve for the ground state of the simple 1D Ising model, with the following cost Hamiltonian for n spins

$$\hat{H}_c = \sum_{j=1}^{n} J_j \hat{\sigma}_j^z \hat{\sigma}_{j+1}^z, \tag{20}$$

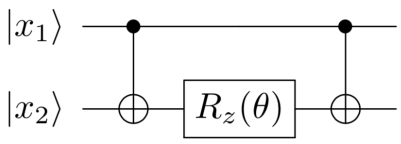
for a known but generally uncorrelated set of J_j, by implementing the Quantum Approximate Optimization Algorithm.

3.2 From operators to a quantum circuit

In order to implement QAOA we will use IBM Q's Cloud Services using qiskit's environment, which allows us to create quantum circuits and run them virtually on classical computers, and also - more importantly - directly on quantum hardware [8]. The first step is to write the QAOA evolution in terms of known quantum gates to implement the circuit.

First things first, the basis for which $\hat{H}_c$ is diagonalised is just the $\hat{\sigma}^z$ basis,

therefore the initial state $|s\rangle = H^{\otimes n}|0\rangle$. Next, for a chosen $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$, we can construct $U(\hat{H}_c, \gamma_i)$ using the following circuit identity for the operator $\exp\{-i\frac{\theta}{2}\hat{\sigma}_j^z \otimes \hat{\sigma_k^z}\}$ (see proof in Appendix A1):



which expresses the operator as a CNOT gate followed by an $R_z(\theta)$ rotation and another CNOT gate. A CNOT gate is a controlled NOT gate, the control in this case being $|x_1\rangle$ and an $R_z(\theta)$ gate is a rotation of θ about the z axis (refer to Appendix A1).

We need to apply this for every pair i, j of digits. Next, the $U(\hat{H}_c, \beta_i)$ unitary is represented by an $R_x(2\beta_i)$ rotation on qubit i. Following these operations, we measure the system and calculate the expectation values to determine $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}| \hat{H}_c |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$. Finally, we perform a classical optimization to obtain the angles $\boldsymbol{\gamma}_{opt}$ and $\boldsymbol{\beta}_{opt}$ that provide a minimum for $\langle \boldsymbol{\gamma}, \boldsymbol{\beta}| \hat{H}_c |\boldsymbol{\gamma}, \boldsymbol{\beta}\rangle$.

3.3 QAOA simulations for 1D Ising model

In this section we implement the QAOA for determining the minimum of $\hat{H}_c$ given by the Ising model. For concreteness, we consider the $n = 3$ case and implement a $p = 1$ depth QAOA. We run the circuit (see Figure 3) on the *qasm* (Quantum Assembly language) "ideal" simulator in qiskit, which provides ideal results without simulating errors in the quantum machine. A major advantage of implementing QAOA in *qasm* is that the circuit implementation can straightforwardly be run on quantum hardware; this is done in Section 3.4.

3.3.1 Case Study $J_j = -1 \ \forall \ j$

We start by performing a sanity check and simulating the QAOA for a $J_j = -1 \ \forall \ j$ 1D Ising model on a classical computer, for which we know the minimum of $\hat{H}_c$ is -3 corresponding to the ferromagnetic state.

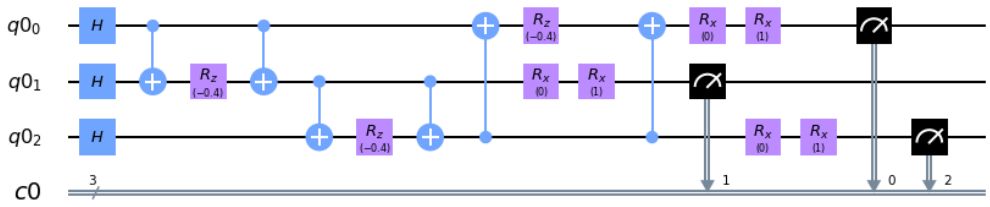


Figure 3: Circuit design of the $n = 3$, $p = 1$, $J_j = -1 \; \forall \; j$, 1D Ising in IBM's qiskit.

We wish to understand the distribution of energies with respect to the variational parameters. For QAOA with $p = 1$, the variational space reduces to two parameters, $\beta_1 \equiv \beta$, $\gamma_1 \equiv \gamma$. One may consider two approaches to choose the angles for a given number of circuit runs m:

1. Pick a uniformly random angle $\gamma, 2\beta \in [0, 2\pi]$ for each sample
2. Create a grid $\gamma \times 2\beta \in [0, 2\pi] \times [0, 2\pi]$

The first approach is the only one feasible for larger p and n but the second allows a systematic study of features for the low p and n, as is the present case. For large enough runs m, both methods should of course yield the same results. For the classical optimization routine we use the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Hessian update strategy using *scipy* packages.

Running QAOA for $J_j = -1 \; \forall \; j$ using the above scheme, for general non-optimal angles, a histogram of measured counts is shown in Figure 4 (a). Following the classical optimization, optimal values of $\beta = 2.327$ and $\gamma = 5.527$ are obtained, which yield the histogram count in Figure 4 (b). The results indicate a state close to the equal superposition of all spin zero and all spin one; in particular, the optimal circuit realizes $|\psi\rangle = \sqrt{0.498}\,|000\rangle + \sqrt{0.502}\,|111\rangle$.

The expectation value is classically calculated from the output of the quantum circuit. When executing a circuit either in the *qasm* simulator or the real devices the result is a counts array providing the fraction of the total counts each state

was measured. That allows us to calculate the probability of being in each state:

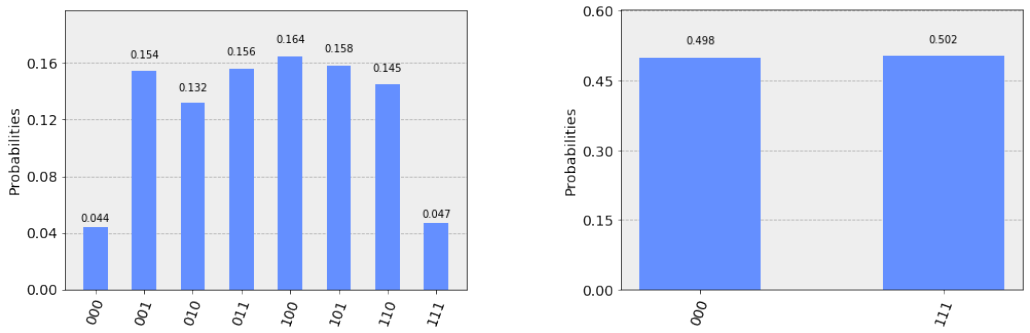
$$P_n = \frac{\text{counts}(|n\rangle)}{\sum_n \text{counts}(|n\rangle)} \tag{21}$$

which allows the construction of the final state as $|\psi\rangle = \sum_n c_n |n\rangle$ where $c_n = \sqrt{P_n}$. Note that measuring counts determines probabilities P_n but misses relative phase information. However, for the problem at hand since the cost Hamiltonian $\hat{H}_c$ is diagonal in the computational basis, the P_n are sufficient to determine the expectation value $\langle\psi| \hat{H}_c |\psi\rangle$:

$$\langle\psi| \hat{H}_c |\psi\rangle = \sum_n |c_n|^2 \langle n| \hat{H}_c |n\rangle = P_n \langle n| \hat{H}_c |n\rangle \tag{22}$$

which simply requires knowledge of $\langle n| \hat{H}_c |n\rangle$ in the computational basis.

With $m = 2500$ runs, the first and second approach give very similar β but different γ, and both the correct expectation (see Table 2). As we will discuss later, for the particular problem at hand, γ values modulo integer multiples of $\pi/2$ yield equivalent QAOA operations; the observed difference here is approximately $3\pi/2$, rendering the two values equivalent.



(a) (b)

Figure 4: (a) Histogram count plot for the circuit in Figure 3 with non-optimal angles $\beta = 0.5$, $\gamma = 0.2$. (b) Histogram count plot for the circuit in Figure 3 with optimal angles $\beta = 2.327$, $\gamma = 5.527$, which gives a close approximation to the equal superposition state, $|\psi\rangle = \sqrt{0.498}\,|000\rangle + \sqrt{0.502}\,|111\rangle$, which is exactly what we expect for the minimum energy eigenstate.

Parameters	Approach 1	Approach 2
γ	5.527	0.754
β	2.327	2.388
$\langle\gamma,\beta\vert\,\hat{H}_c\,\vert\gamma,\beta\rangle$	-3.000	-3.000

Table 2: Results of QAOA for the $\hat{H}_c$ with $J_j = -1\ \forall\ j$ with approach 1 and approach 2.

To understand the manifold of $\hat{H}_c$ values in the variational parameter space, including the location of the minima and periodicities with respect to the variational angles, we explore a grid of β-γ and construct a density map of the QAOA results (see Figure 5). The results make intuitive sense since we expect certain periodicities for the very simple $J_j = -1\ \forall\ j$ case. Namely, we already expect to have periodicities in the β axis due to the $U(\hat{H}_c, \beta)$ unitary. Notice how $R_x(2\beta)$ completes a full rotation [1] at $2\beta = \pi$, thus implying a periodicity of $\pi/2$ in β. This is confirmed by looking at Figure 5. From now onward we will therefore restrict the β values in the optimization to $\beta \in [0, \pi/2]$.

The γ periodicity comes from the symmetry in the chosen $J_j = -1\ \forall\ j$, and can be understood by computing the matrix corresponding to the $U(\hat{H}_c, \gamma)$ using a *Mathematica* script, yielding the $\pi/2$ result for the $J_j = -1\ \forall\ j$ case (see Appendix A2). For instance, for $J = [1, -0.25, 0.25]$ we see a much different profile in for the γ axis, as shown in Figure 6.

[1] $R_x(\theta)$ is defined as a θ rotation about the x axis, analogously as $R_z(\theta)$, $R_x(\theta) = \exp\{-i\frac{\theta}{2}\hat{\sigma}^x\}$.

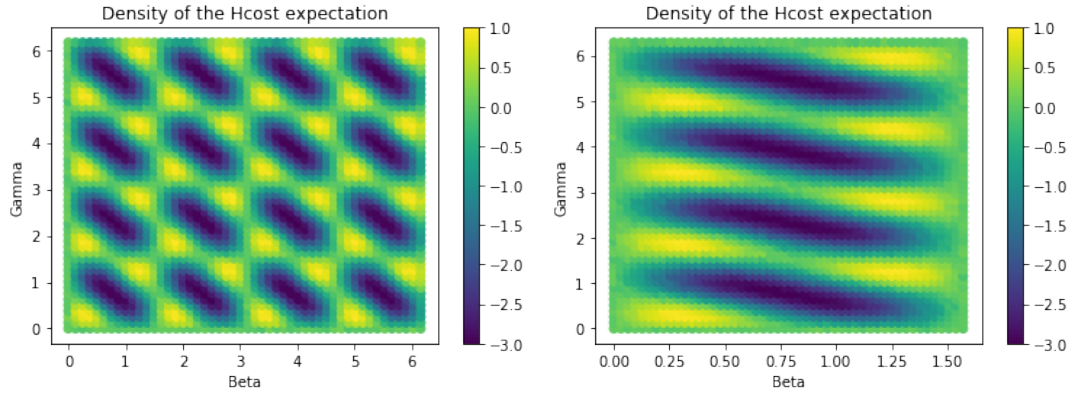


Figure 5: (Left) Density of the $\hat{H}_c$ expectation with respect to β and γ values in a grid. (Right) Same plot with restricted β domain due to the periodicity in $U(\hat{H}_c, \beta)$. In this case due to symmetry in J = [-1,-1,-1] the γ axis is also $\pi/2$ symmetric.

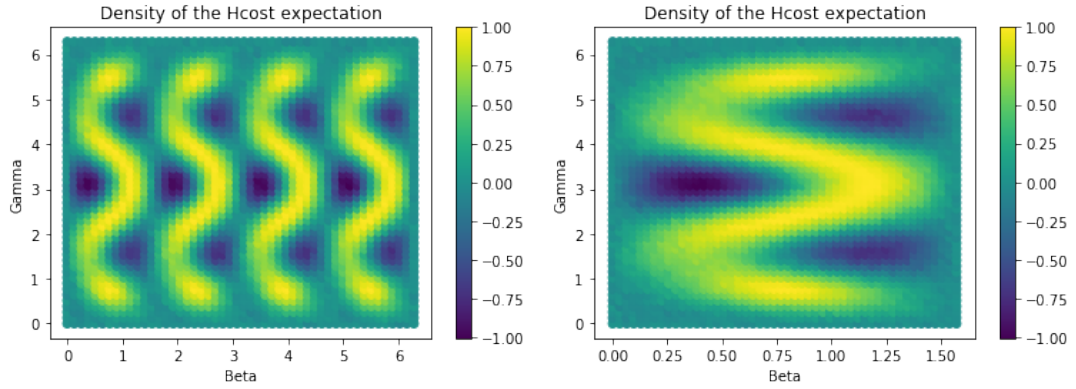


Figure 6: Density plots for J = [1,-0.25,0.25] (Left) Density of the $\hat{H}_c$ expectation with respect to β and γ values in a grid. (Right) Same plot with restricted β domain due to the periodicity in $U(\hat{H}_c, \beta)$.

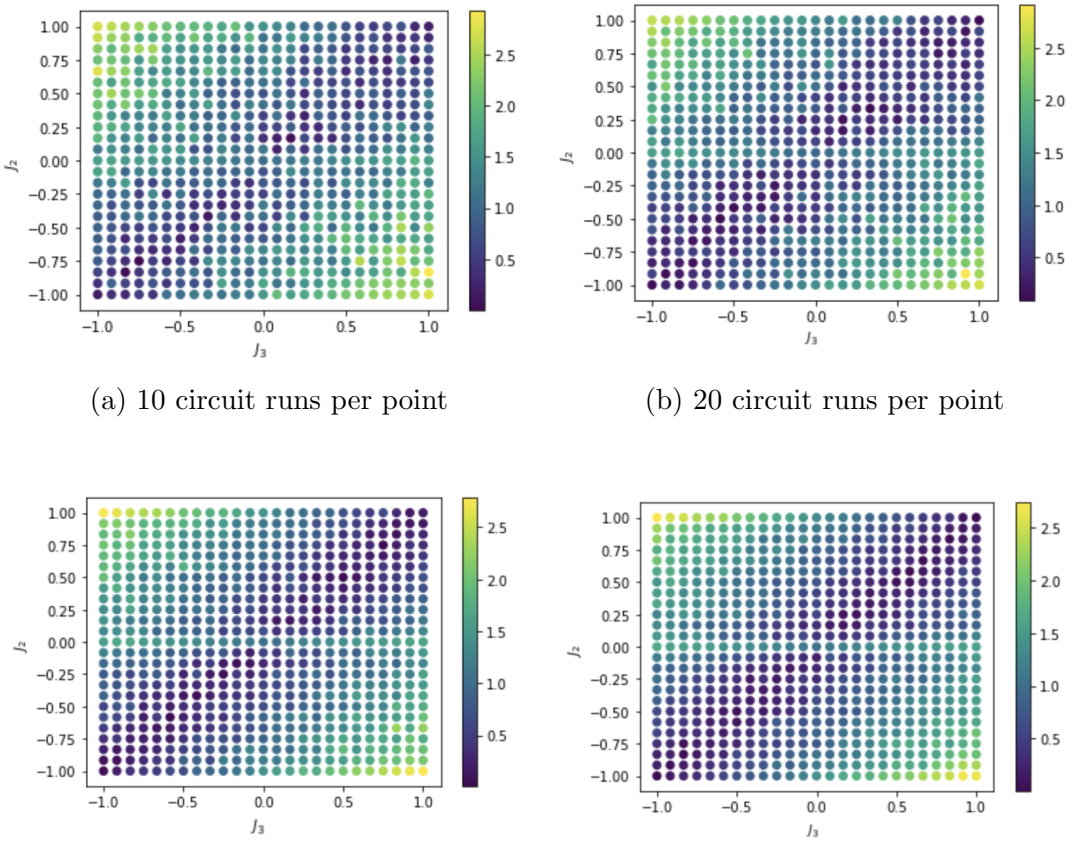
3.3.2 General J case

Before moving on to the implementation on quantum hardware, we need to understand how the "ideal" simulation works as we vary J configurations. Consider fixing $J_1 = 1$ and taking $J_2, J_3 \in [-1, 1]$, now varying J_2, J_3 we can build a landscape of the difference between the minimum costs obtained by direct diagonalization of $\hat{H}_c$, and the QAOA result; this value should ideally be close to zero.

As shown in Figure 7, the performance is not uniform, with the best performing values lying with correlated J_2 and J_3, along the diagonal. This indicates that the performance of QAOA and similar methods does vary with the complexity of the model being minimized.

It is also worth considering how this landscape improves with increasing number of circuit runs m; as shown in Figure 7 (a)-(d), increasing the number of samples makes this correlation smoother and broadens the dark blue region where the QAOA result agrees well with the expected true minimum.

Having understood the QAOA implementation on the *qasm* simulator, we are well-placed to execute QAOA circuits on actual quantum hardware. This is discussed in the next section.



(a) 10 circuit runs per point

(b) 20 circuit runs per point

(c) 50 circuit runs per point

(d) 100 circuit runs per point

Figure 7: Difference in performance between the QAOA output and classical calculation of the minimum eigenvalue (absolute value of difference between QAOA value and true minimum shown in color) for varying J_2 and J_3. A sample of 625 points as allowed by CPU resources with varying number of circuit runs m per point.

3.4 QAOA on IBM Q's Yorktown device

We run QAOA for the $n = 3$ Ising model on IBM's 5 qubit Yorktown device due to the direct 3 qubit connectivity, which matches the connectivity we need for the $n = 3$ Ising model, see Figure 8.

We run the optimization procedure by first sampling 20 values to obtain reasonable starting points for the gradient descent. The chosen optimization routine is the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Hessian update. Each QAOA

circuit execution takes more than 2 minutes, with delays incurred by the necessity of connection to IBM Q's Cloud Service, as well as wait times until jobs are released. Running a full optimization can therefore be prohibitively slow; we thus restrict ourselves to $m = 20$ circuit runs instead of $O(10^3)$ used for the *qasm* simulator, allowing a full optimization to be run in about 3 hours. Choosing optimal initial points for gradient descent is essential for a good optimization, so in addition to the QPU errors we expect worse performance due to small sampling. Actually the results start to be statistically significant when the samples are in the order of thousands [9] so we shouldn't expect to get very significant results. Ideally we would compute a landscape like Figure 7 with enough samples but that is not feasible time-wise so instead we will focus on specific examples and using a small number of circuit executions.

3.4.1 Case study: $J_j = -1 \ \forall \ j$

First things first, we can start seeing the errors in the real device by looking at the $J_j = -1 \ \forall \ j$ case for the optimal angles found in the previous section for the *qasm* simulator. Executing the circuit with those optimal angles $\beta = 2.327$, $\gamma = 5.527$ yields a noisy but still correct solution of the final states of QAOA as seen in the right of Figure 8.

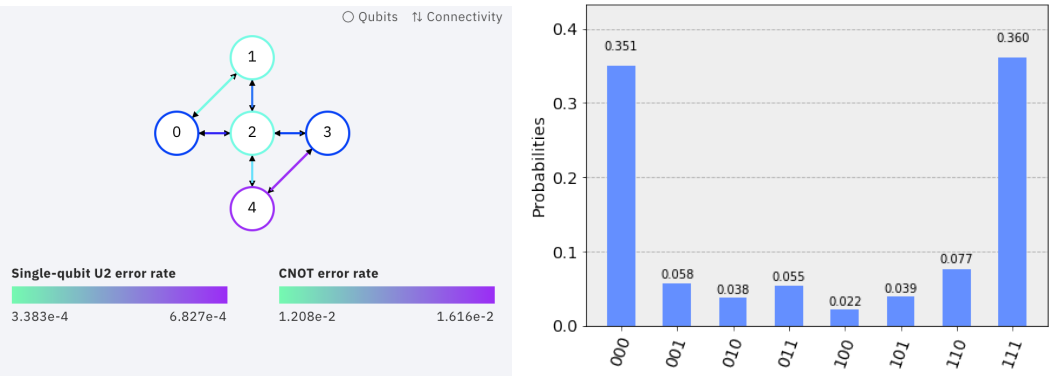


Figure 8: (Left) IBM Q's Yorktown 5 qubit device, with gate errors. (Right) Count histogram for the 1D simple Ising Model with $J_j = -1 \ \forall \ j$. The ideal result would be 0.5 probability on the 000 and 111 state.

After 132 steps of gradient descent optimization, a minimum is found with a

value of -2.252, in comparison to the correct value of -3, see Table 3.

	True minimum	**Simulation (qasm)**	**Yorktown**
Min. $\langle \hat{H}_c \rangle$	-3.000	-2.968	-2.252
β	-	0.684	0.644
γ	-	4.097	4.281

Table 3: Ising $J_j = -1 \ \forall \ j$ results (note that the simulation value was obtained from 20 samples to match the samples ran in the QPU).

3.4.2 General J case

Also interesting is to check how non trivial J's perform in Yorktown. Setting $J_1 = 1$ we study results for J_2 and J_3 that perform well in the *qasm* simulator ($J_2 \approx J_3$) as well as those that do not, according to Figure 7. The results shown in Figure 9 (for explicit results refer to Table 4 in Appendix B) and are taken with 20 circuit runs, which is not very accurate as the initial guesses for the optimizer are very important.

We find that the Yorktown results (green) consistently perform worse than the *qasm* simulations (orange), as expected: the difference can be attributed to errors in gate operations on the actual quantum hardware. Additionally, both differ from the true minimum due to a limited number of circuit runs used.

To visualize the data we can also plot the difference in results between Yorktown and both the true minimum eigenvalues and the *qasm* simulation. As shown in Figure 10 (a) the results from Yorktown showcase similar performance landscape as in Figure 7, where correlated J_2 and J_3 values perform better than uncorrelated ones with respect to the true minimum eigenvalues. Comparing the Yorktown output to the *qasm* simulation also yields the trend of similarity along the diagonal and greater dissimilarity elsewhere as shown in Figure 10 (b).

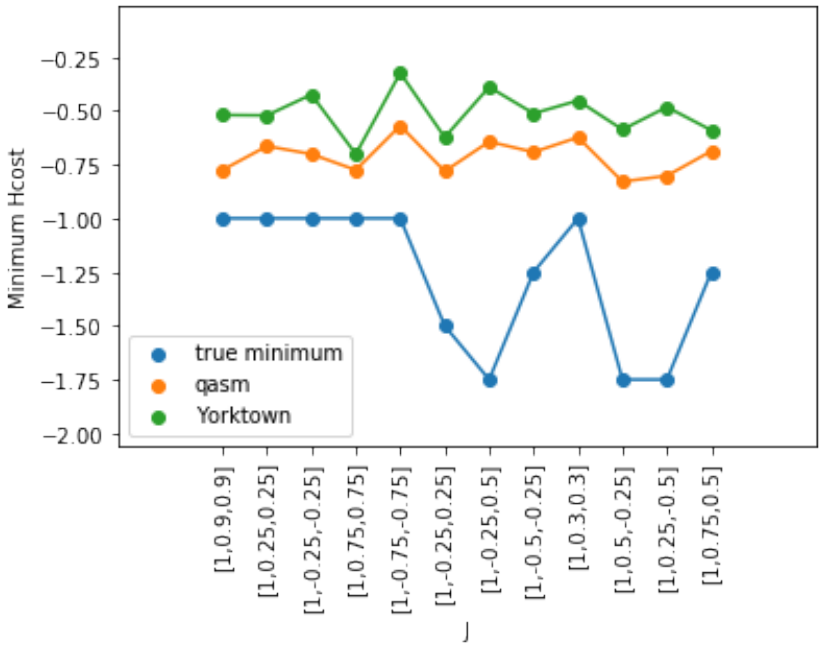
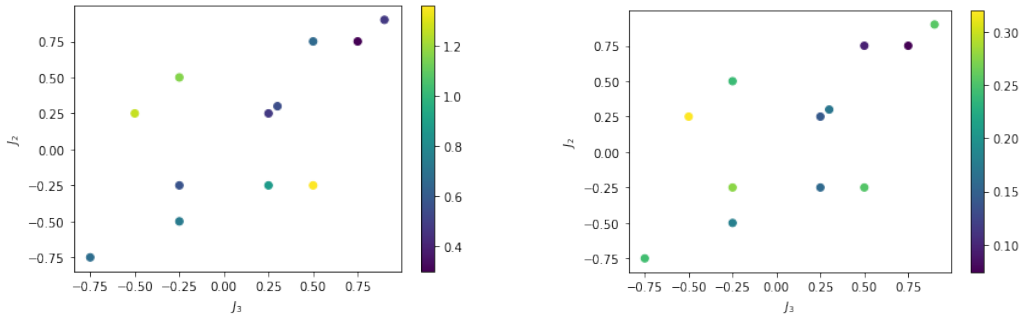


Figure 9: QAOA results for different J's using 20 circuit runs for both the *qasm* simulator and Yorktown.



(a) (b)

Figure 10: (a) Difference in performance between the QAOA Yorktown output and the true minimum eigenvalue (shown in color-scale) for $J_1 = 1$ and varying J_2,J_1. (b) Difference in performance between the QAOA Yorktown output and the *qasm* simulator output. Both use 20 circuit runs per point to find the initial guesses for the optimizer.

As explained before, in addition to expected errors from the quantum device such as gate errors and decoherence these results are also error prone due to the

small number of circuit runs (due to time constraints on using IBM Q).

4 Future work

While we have successfully implemented QAOA using both the *qasm* simulator and on quantum hardware accessed via the IBM Q network, a number of interesting questions have been raised along the way that can be addressed by future extensions of this work.

Unfortunately, constraints of access to (quantum) computational resources have limited our ability to study p-depth QAOA on the n-spin 1D Ising chain for $n > 3$ and $p > 1$, even thought the current code base has been written to be able to handle arbitary n, p (subject to actual quantum hardware connectivities, of course). As such, with more time one could study the performance of QAOA with increasing n and for more complex spin configurations.

Perhaps most interesting is the behaviour observed in Sec. 3.3.2, where the $p = 1$ QAOA exhibits varying levels of performance depending on the J values defining the specific 1-D Ising model. Naturally one may ask whether increasing the depth of QAOA to $p > 1$ improves performance. We have begun addressing this implementation of QAOA, with some preliminary results highlighted in the following section, as well as a discussion of the scope and nature of the problem, which is an active area of research.

4.1 Dependence on depth p

The p dependence is an intricate issue due to the increase in variational parameters, having to work on a $2p$-dimensional space growing exponentially in n.

By continuing with the random angles approach used with $p = 1$ we consider the $J_j = -1 \; \forall \; j$ case and explore p dependence. As shown in Figure 11 a) there is no such dependence so we hypothesize $J_j = -1 \; \forall \; j$ is too trivial due to the almost perfect performance in the simulator.

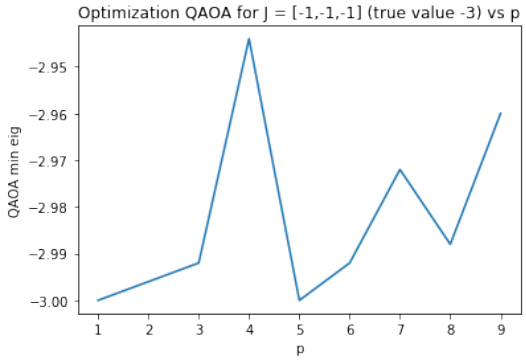
Instead to study we consider a good but not perfect example like $J = [1, -0.25, 0.25]$

which performs well but not perfectly for $p = 1$ according to Figure 7. However, computing the same plots with varying sample size doesn't improve the performance with increasing p (up to $p = 9$ due to CPU resources), see Figure 11.

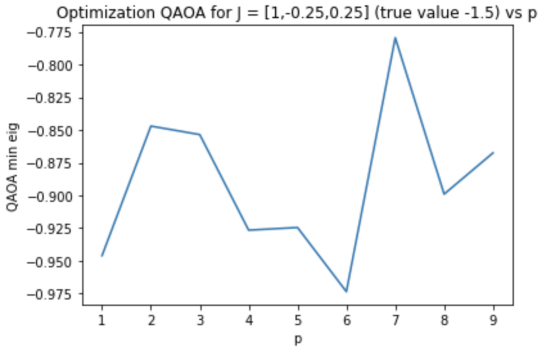
We conclude that random sampling for $p > 1$ is not an optimal approach to find the angles, and that more statistically significant ways to generate angles is needed. Indeed it is still an open question how to efficiently optimize QAOA circuits [2], and novel methods have been proposed recently. Zhou et al. (2018) showed that β_i and γ_i vary slowly following a smooth curve and proposed choosing $p+1$ level angles based on the optimal p level angles. They also introduced another heuristic method using discrete Sine/Cosine Transform using the optimized p level frequency to obtain the $p + 1$ level angles [7]. Pagano et al. (2019) expanded on the study of the β_i and γ_i curves by looking at minima profiles and also using p level angles as a prior for $p + 1$ level angles [10].

5 Conclusions

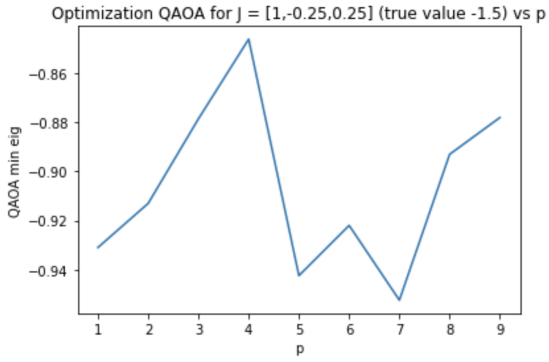
Overall this report provides the theoretical background needed to understand why and how the QAOA works, showcasing the inner workings as applied to the 1D Ising Model with $n = 3$. Moreover we also show in detail how to implement the QAOA using quantum assembly language, in a way that can be directly run on a quantum computer, and how to analyze the actual physical output to construct the cost function. We implement an optimization procedure and explore the performance landscape including patterns and periodicities in the variational space. This is first used on an ideal quantum simulator which allows us to understand underlying properties of the solution space and then we implement it on IBM Q's Yorktown devices. We observe correlated J values perform better both on the simulations and the real devices and that the quantum hardware results consistently perform worse than the simulations due to errors in gate operations, decoherence and undersampling. Our QAOA algorithm is written for general n and depth p allowing for the implementation of larger circuits and connectivities in the future.



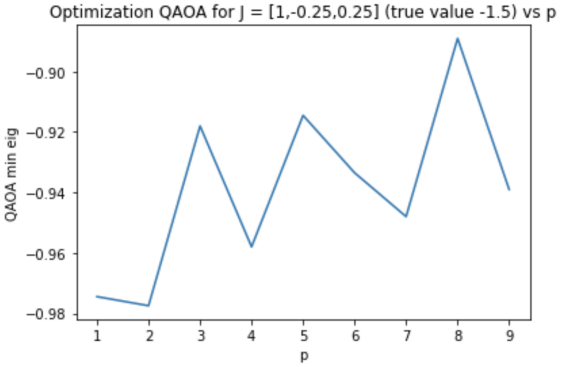
(a) $J_j = -1 \ \forall \ j$ with 250 samples



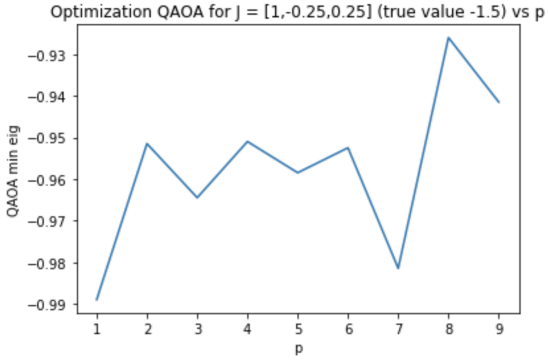
(b) $J = [1, -0.25, 0.25]$ with 250 samples



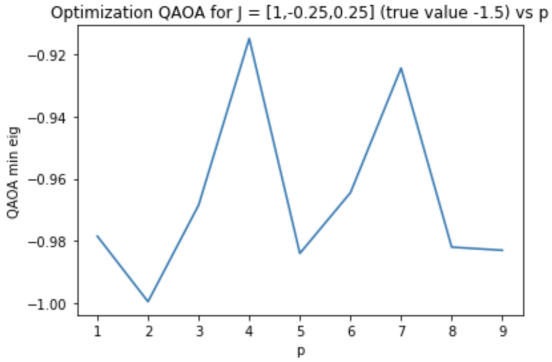
(c) $J = [1, -0.25, 0.25]$ with 500 samples



(d) $J = [1, -0.25, 0.25]$ with 1000 samples



(e) $J = [1, -0.25, 0.25]$ with 2000 samples



(f) $J = [1, -0.25, 0.25]$ with 4000 samples

Figure 11: Simulated QAOA performance of $J = [1, -0.25, 0.25]$ with p. Note how $p \in [1, 9]$ doesn't seem to affect performance and how increasing the samples reduces the bounds but still does not reach the true minimum of -1.5.

6 Code

Throughout the report we use Jypyter Notebooks to simulate and run the QAA and QAOA; for the periodicity analysis we use a Mathematica script. All can be accessed in this *Git*[2] repository.

[2] *https://github.com/vives1/JP_QAOA.git*

Acknowledgements

I would like to thank Professor Türeci and Saeed Khan for advising and guiding me throughout the project. I would also like to thank Abraham Asfaw, the Global Lead of Quantum Education at IBM Q, for providing access to IBM Q's network.

References

[1] AD Corcoles, A Kandala, A Javadi-Abhari, DT McClure, AW Cross, K Temme, PD Nation, M Steffen, and JM Gambetta. Challenges and opportunities of near-term quantum computing systems. *arXiv preprint arXiv:1910.02894*, 2019.

[2] Gavin E Crooks. Performance of the quantum approximate optimization algorithm on the maximum cut problem. *arXiv preprint arXiv:1811.08419*, 2018.

[3] Matthew Radzihovsky, Joey Murphy, and Mason Swofford. A qaoa solution to the traveling salesman problem using pyquil. 2019.

[4] Max Born and Vladimir Fock. Beweis des adiabatensatzes. *Zeitschrift für Physik*, 51(3-4):165–180, 1928.

[5] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

[6] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[7] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D Lukin. Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices. *arXiv preprint arXiv:1812.01041*, 2018.

[8] Héctor Abraham, Ismail Yunus Akhalwaya, Gadi Aleksandrowicz, Thomas Alexander, Gadi Alexandrowics, Eli Arbel, Abraham Asfaw, Carlos Azaustre, AzizNgoueya, Panagiotis Barkoutsos, George Barron, Luciano Bello, Yael Ben-Haim, Daniel Bevenius, Lev S. Bishop, Samuel Bosch, David Bucher, CZ, Fran Cabrera, Padraic Calpin, Lauren Capelluto, Jorge Carballo, Ginés Carrascal, Adrian Chen, Chun-Fu Chen, Richard Chen, Jerry M. Chow, Christian Claus, Christian Clauss, Abigail J. Cross, Andrew W. Cross, Juan Cruz-Benito, Chris Culver, Antonio D. Córcoles-Gonzales, Sean Dague, Matthieu

Dartiailh, DavideFrr, Abdón Rodríguez Davila, Delton Ding, Eric Drechsler, Drew, Eugene Dumitrescu, Karel Dumon, Ivan Duran, Pieter Eendebak, Daniel Egger, Mark Everitt, Paco Martín Fernández, Albert Frisch, Andreas Fuhrer, MELVIN GEORGE, IAN GOULD, Julien Gacon, Gadi, Borja Godoy Gago, Jay M. Gambetta, Luis Garcia, Shelly Garion, Gawel-Kus, Juan Gomez-Mosquera, Salvador de la Puente González, Donny Greenberg, Wen Guan, John A. Gunnels, Isabel Haide, Ikko Hamamura, Vojtech Havlicek, Joe Hellmers, Łukasz Herok, Stefan Hillmich, Hiroshi Horii, Connor Howington, Shaohan Hu, Wei Hu, Haruki Imai, Takashi Imamichi, Raban Iten, Toshinari Itoko, Ali Javadi-Abhari, Jessica, Kiran Johns, Naoki Kanazawa, Anton Karazeev, Paul Kassebaum, Knabberjoe, Arseny Kovyrshin, Vivek Krishnan, Kevin Krsulich, Gawel Kus, Ryan LaRose, Raphaël Lambert, Joe Latone, Scott Lawrence, Dennis Liu, Peng Liu, Panagiotis Barkoutsos ZRL Mac, Yunho Maeng, Aleksei Malyshev, Jakub Marecek, Manoel Marques, Dolph Mathews, Atsushi Matsuo, Douglas T. McClure, Cameron McGarry, David McKay, Srujan Meesala, Antonio Mezzacapo, Rohit Midha, Zlatko Minev, Michael Duane Mooring, Renier Morales, Niall Moran, Prakash Murali, Jan Müggenburg, David Nadlinger, Giacomo Nannicini, Paul Nation, Yehuda Naveh, Nick-Singstock, Pradeep Niroula, Hassi Norlen, Lee James O'Riordan, Oluwatobi Ogunbayo, Pauline Ollitrault, Steven Oud, Dan Padilha, Hanhee Paik, Simone Perriello, Anna Phan, Marco Pistoia, Alejandro Pozas-iKerstjens, Viktor Prutyanov, Daniel Puzzuoli, Jesús Pérez, Quintiii, Rudy Raymond, Rafael Martín-Cuevas Redondo, Max Reuter, Diego M. Rodríguez, Mingi Ryu, Tharrmashastha SAPV, Sam-Ferracin, Martin Sandberg, Ninad Sathaye, Bruno Schmitt, Chris Schnabel, Travis L. Scholten, Eddie Schoute, Ismael Faro Sertage, Nathan Shammah, Yunong Shi, Adenilton Silva, Yukio Siraichi, Iskandar Sitdikov, Seyon Sivarajah, John A. Smolin, Mathias Soeken, SooluThomas, Dominik Steenken, Matt Stypulkoski, Hitomi Takahashi, Charles Taylor, Pete Taylour, Soolu Thomas, Mathieu Tillet, Maddy Tod, Enrique de la Torre, Kenso Trabing, Matthew Treinish, TrishaPe, Wes Turner, Yotam Vaknin, Carmen Recio Valcarce, Francois Varchon, Desiree Vogt-Lee, Christophe Vuillot, James Weaver, Rafal Wieczorek, Jonathan A. Wildstrom, Robert Wille, Erick Winston, Jack J.

Woehr, Stefan Woerner, Ryan Woo, Christopher J. Wood, Ryan Wood, Stephen Wood, James Wootton, Daniyar Yeralin, Jessie Yu, Christopher Zachow, Laura Zdanski, Zoufalc, anedumla, azulehner, bcamorrison, brandhsn, dennis-liu 1, dime10, drholmie, elfrocampeador, faisaldebouni, fanizzamarco, gruu, kanejess, klinvill, kurarrr, lerongil, ma5x, merav aharoni, mrossinek, ordmoj, strickroman, tigerjack, toural, yang.luh, and yotamvakninibm. Qiskit: An open-source framework for quantum computing, 2019.

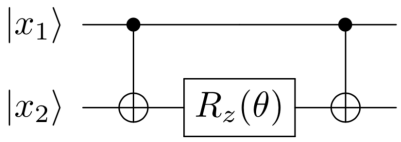
[9] Gian Giacomo Guerreschi and AY Matsuura. Qaoa for max-cut requires hundreds of qubits for quantum speed-up. *Scientific reports*, 9(1):6903, 2019.

[10] G Pagano, A Bapat, P Becker, KS Collins, A De, PW Hess, HB Kaplan, A Kyprianidis, WL Tan, C Baldwin, et al. Quantum approximate optimization with a trapped-ion quantum simulator. *arXiv preprint arXiv:1906.02700*, 2019.

A Supplementary material

A.1 Circuit Identity

We wish to show that $\exp\{-i\frac{\theta}{2}\hat{\sigma}_j^z \otimes \hat{\sigma_k^z}\}$ is equivalent to the following circuit:



First note that the circuit is represented in matrix notation by the following gates: $CNOT * [I \otimes R_z(\theta)] * CNOT$ and

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{23}$$

$$R_z(\theta) = \exp\left\{-i\frac{\theta}{2}\hat{\sigma}^z\right\} = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 \\ 0 & e^{i\frac{\theta}{2}} \end{pmatrix} \tag{24}$$

Then the circuit is equivalent to:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{i\frac{\theta}{2}} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{2}} \end{pmatrix}$$

Conversely, since

$$\hat{\sigma}_j^z \otimes \hat{\sigma_k^z} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

Then it follows that

$$\exp\left\{-i\frac{\theta}{2}\hat{\sigma}_j^z \otimes \hat{\sigma_k^z}\right\} = \begin{pmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 & 0 \\ 0 & e^{i\frac{\theta}{2}} & 0 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} & 0 \\ 0 & 0 & 0 & e^{-i\frac{\theta}{2}} \end{pmatrix}$$

A.2 Periodicity Analysis

The Mathematica code can be accessed in this *Git* repository.

A.2.1 $J_j = -1 \ \forall \ j$ case

The Ising gate M matrix is as follows:

$$\begin{pmatrix} e^{3i\gamma}\cos^3(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & -ie^{3i\gamma}\sin^3(\beta) \\ ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & e^{-i\gamma}\cos^3(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -ie^{-i\gamma}\sin^3(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) \\ ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & e^{-i\gamma}\cos^3(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & -ie^{-i\gamma}\sin^3(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) \\ -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & e^{-i\gamma}\cos^3(\beta) & -ie^{-i\gamma}\sin^3(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) \\ ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & -ie^{-i\gamma}\sin^3(\beta) & e^{-i\gamma}\cos^3(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) \\ -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -ie^{-i\gamma}\sin^3(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & e^{-i\gamma}\cos^3(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) \\ -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & -ie^{-i\gamma}\sin^3(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) & -e^{-i\gamma}\sin^2(\beta)\cos(\beta) & e^{-i\gamma}\cos^3(\beta) & ie^{-i\gamma}\sin(\beta)\cos^2(\beta) \\ -ie^{3i\gamma}\sin^3(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & -e^{3i\gamma}\sin^2(\beta)\cos(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & ie^{3i\gamma}\sin(\beta)\cos^2(\beta) & e^{3i\gamma}\cos^3(\beta) \end{pmatrix} \tag{25}$$

Adding the Hadamards gates yields a much larger matrix M_f that we will not print. The trace however is calculated to be:

$$\begin{aligned} Tr(M_f) &= -\frac{3ie^{-i\gamma}\sin^3(\beta)}{\sqrt{2}} - \frac{ie^{3i\gamma}\sin^3(\beta)}{\sqrt{2}} - \frac{3ie^{-i\gamma}\sin(\beta)\cos^2(\beta)}{\sqrt{2}} + \frac{3ie^{3i\gamma}\sin(\beta)\cos^2(\beta)}{\sqrt{2}} \\ &= 3\sqrt{2}e^{i(\gamma+\pi/2)}\{\cos^2(\beta)\{\sin(\beta)\sin(2\gamma) + \cos(2\gamma)\} - 1\} \end{aligned} \tag{26}$$

Now since $\gamma \in [0, 2\pi]$, then the $\sin(\beta)\sin(2\gamma)+\cos(2\gamma)$ term will be periodic in π and since the expectation is calculated from the modulus of the complex number (which is $\pi/2$ periodic), then the periodicity in the γ axis will end up being $\pi/2$.

B Yorktown data

The table below has the numerical values for a subset of tested J's in IBM Q's Yorktown device:

J	**True minimum**	**Simulation (qasm)**	**Yorktown**
[-1,-1,-1]	-3.000	-2.968	-2.252
[1,0.9,0.9]	-1.000	-0.775	-0.519
[1,0.25,0.25]	-1.000	-0.664	-0.521
[1,-0.25,-0.25]	-1.000	-0.701	-0.424
[1,0.75,0.75]	-1.000	-0.775	-0.701
[1,-0.75,-0.75]	-1.000	-0.569	-0.323
[1,-0.25,0.25]	-1.500	-0.781	-0.623
[1,-0.25,0.50]	-1.750	-0.643	-0.389
[1,-0.50,-0.25]	-1.250	-0.692	-0.512
[1,0.3,0.3]	-1.000	-0.623	-0.352
[1,0.5,-0.25]	-1.750	-0.829	-0.587
[1,0.25,-0.50]	-1.750	-0.802	-0.482
[1,0.75,0.50]	-1.250	-0.687	-0.590

Table 4: QAOA results for different J's using 20 circuit runs for both the *qasm* simulator and Yorktown.