# QOSF Mentorship Screening Task

Chirag Wadhwa

# Contents

# Chapter 1

# Introduction

This documents describes my attempt at the $2^{nd}$ Screening Task of the $4^{th}$ Cohort of the QOSF Mentorship Program

## 1.1 The Task

- Prepare 4 random 4-qubit quantum states of your choice
- Create and train a variational circuit that transforms input states into predefined output states. Namely
  - If random state 1 is provided, it returns state $|0011\rangle$
  - If random state 2 is provided, it returns state $|0101\rangle$
  - If random state 3 is provided, it returns state $|1010\rangle$
  - If random state 4 is provided, it returns state $|1100\rangle$
- What would happen if you provided a different state?

# Chapter 2

# Implementation

## 2.1 Creating initial states

I chose to randomize the starting states for each run of the algorithm. However, over various attempts, I realised that there are certain sets of random input states for which the variational algorithm will never successfully return the desired output states. Thus, I decided to create random input states that satisfy the following two conditions:

- **Each pair of input states must be orthogonal.** If this is not true for the input states, the output states will also not be pairwise orthogonal. We would thus never obtain the desired states.
- **The input states must preserve the Hamming distance relations between the desired output states.** This statement is somewhat informal and I don't yet have a concrete explanation for the reason behind this. However, my intuition led me to believe that the chosen input states must have a relationship between each other that reflects the relationship between the output states. Through trial and error, this was the one that worked best for me.

Keeping the above in mind, I followed these steps to generate the initial states:

- A 4-bit binary string called the *seed* is chosen randomly. The *seed* cannot be '0000' or '1111' to ensure the circuit isn't trivial
- A set of 4 strings, *str* are used to construct states.
  For $i \in [0, 3]$, *str[i]* is defined as the bitwise *xor* of *seed* and the $i^{th}$ desired output state
- For each of the 4-bit strings, a single 4-qubit state is constructed as follows:
  - For every '1' in the string, apply an X gate to the corresponding qubit

- Apply an H gate to each of the 4 qubits

For example, a seed of '1010' leads to the following steps:

- *str* = {'1001','1111','0000','0110'}
- *states* = $\{|-++-\rangle, |----\rangle, |++++\rangle, |+--+\rangle\}$, where $|+\rangle$ and $|-\rangle$ are the eigenstates of the Pauli-X matrix

## 2.2 Creating and training variational circuit

After experimenting with various variational forms, I obtained the best results with the following:
2 layers of parameterised Ry gates on each qubit and Controlled-Z gates on each pair of qubits [2]

The objective function simply returns the ratio of the measured results that gave the correct output. Rather than using other modules to give the fidelity between two states, I chose to use measurement results for my objective function to ensure this code would also run on a real device, not just a simulator. I used the SciPy optimizer to train the circuit, using the 'COBYLA' [1, 3] optimization method.

# Chapter 3

# Results

After optimization, we get a trained circuit that gives the desired output states with a very high probability (usually above 99%). Thus this circuit can be used reliably for this task.

## 3.1 What would happen if the input state was different?

Let us first see what the circuit actually does. For this consider the following unitary: $U_{seed} |x\rangle = |x \oplus seed\rangle$ , where $x, seed \in \{0,1\}^4$. Note that $U_{seed}$ is its own inverse.

Now we can see that for a given output state $|y_i\rangle$ and random bitstring *seed*, the initial state is constructed as:
$|x_i\rangle = H^{\otimes 4} U_{seed} |y_i\rangle$

Let the ideal circuit be represented by the unitary $V$. Then,
$V |x_i\rangle = |y_i\rangle$,
$V H^{\otimes 4} U_{seed} |y_i\rangle = |y_i\rangle$,
$V = (H^{\otimes 4} U_{seed})^\dagger$,
$V = U_{seed} H^{\otimes 4}$

Now that we know the action of the ideal circuit, we can understand its actions on all possible states. Since the trained circuit is a close approximation of the ideal circuit, we can also predict its actions on all input states.

# Bibliography


[1] Abraham Asfaw et al. *Learn Quantum Computation Using Qiskit*. 2020. URL: `http://community.qiskit.org/textbook`.

[2] Marcello Benedetti et al. "A generative modeling approach for benchmarking and training shallow quantum circuits". In: *npj Quantum Information* 5.1 (May 2019), p. 45. URL: `https://doi.org/10.1038/s41534-019-0157-8`.

[3] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: `10.1038/s41592-019-0686-2`.