

Verifying Fairness in Quantum Machine Learning

Abstract. Due to the beyond-classical capability of quantum computing, quantum machine learning is standing alone or being embedded in classical models for decision making, especially in the field of finance. Fairness and other ethical issues are often one of the main concerns in decision making. In this work, we define a formal framework for the fairness verification and analysis of quantum machine learning decision models, where we adopt one of the most popular notions of fairness in the literature based on the intuition — any two similar individuals must be treated similarly and thus are unbiased. We show that quantum noise can help to improve fairness and develop an algorithm to check whether a (noisy) quantum machine learning model is fair. In particular, this algorithm can find bias kernels of quantum data (encoding individuals) during checking. These bias kernels generate infinitely many bias pairs for investigating the unfairness of the model.

1 Introduction

Fairness in Machine Learning:

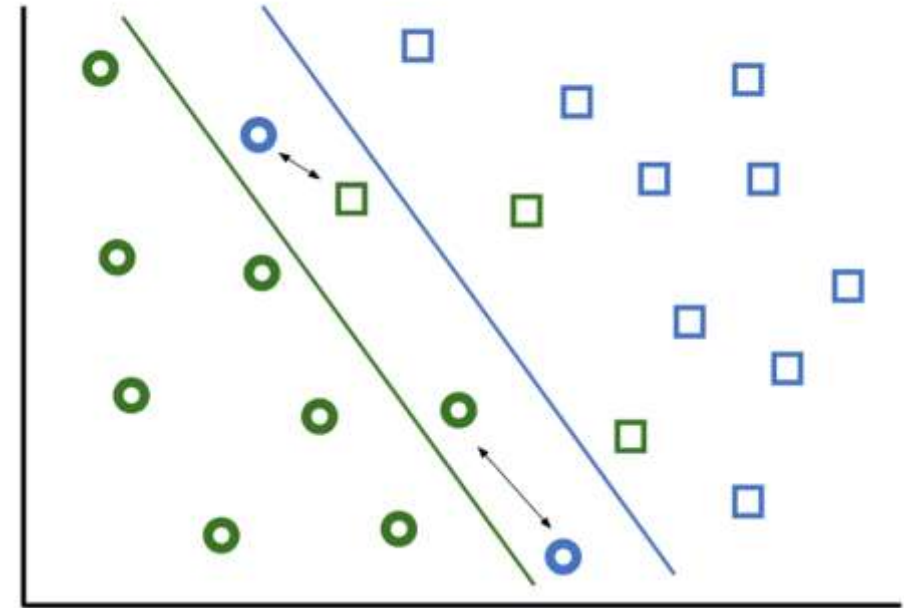
- **Group fairness** aims to achieve through statistical parity the same outcomes across different protected groups (e.g. gender or race)
- **Individual fairness** — treating similar individuals similarly



can be thought of as a kind of **global robustness**

Metric for measuring the similarity

Individual vs Group Fairness



1 Introduction - Challenges

Fairness versus Robustness:

- where a verifier is given a nominal input quantum datum and it checks robustness in a neighborhood of that particular input datum.

Classical versus Quantum Models:

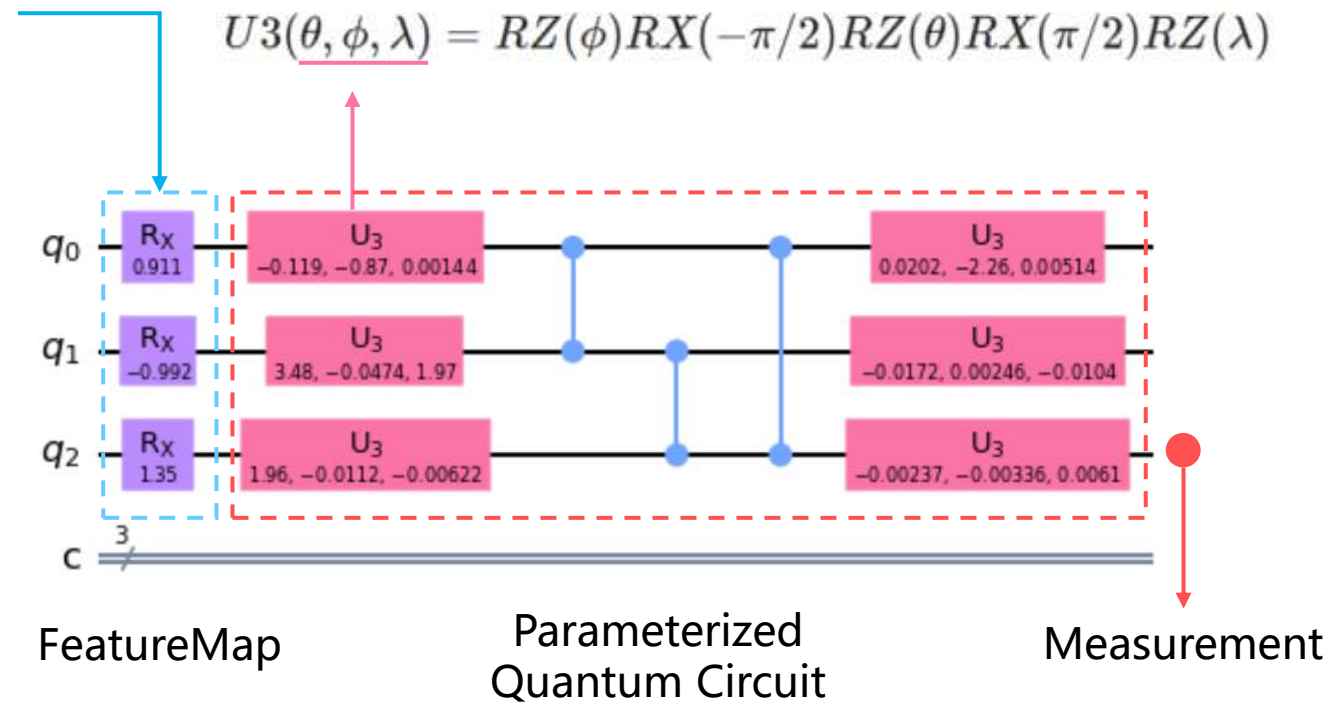
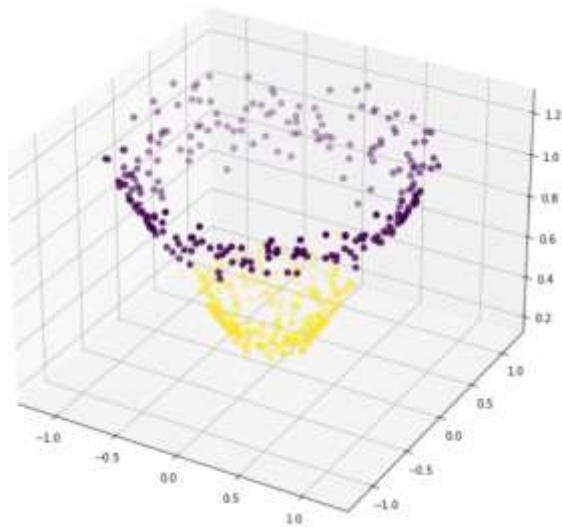
- the corresponding **constraints** in the quantum models are **nonlinear**, and thus searching the data set in a linear domain is ineffective in the quantum case.

Efficiency:

- As the dimension of input data increases **exponentially** with the number of qubits, efficiency is always a key issue in the verification of quantum machine learning models.

2 Quantum Decision Models

[0.289832 -0.315663 0.428539]
 [0.694416 0.758930 1.028683]



2 Quantum Decision Models

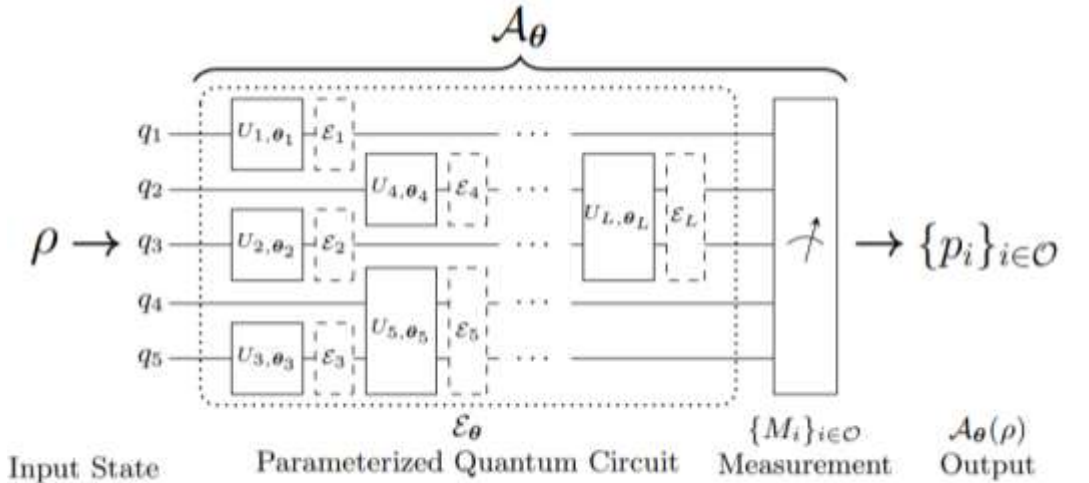
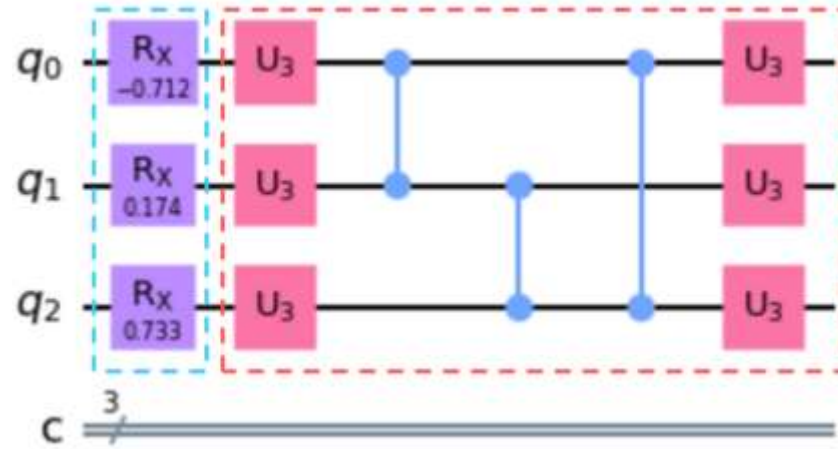


Fig. 1: Noisy Quantum (Machine Learning) Decision Model

Definition 1. A quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ is a randomized mapping:

$$\mathcal{A} : \mathcal{D}(\mathcal{H}) \rightarrow \mathcal{D}(\mathcal{O}) \quad \mathcal{A}(\rho) = \{\text{tr}(M_i \mathcal{E}(\rho) M_i^\dagger)\}_{i \in \mathcal{O}} \quad \forall \rho \in \mathcal{D}(\mathcal{H}),$$

where \mathcal{E} is a super-operator on Hilbert space \mathcal{H} , and $\{M_i\}_{i \in \mathcal{O}}$ is a quantum measurement on \mathcal{H} with \mathcal{O} being the set of measurement outcomes (classical information) we are interested in.

2 Quantum Decision Models

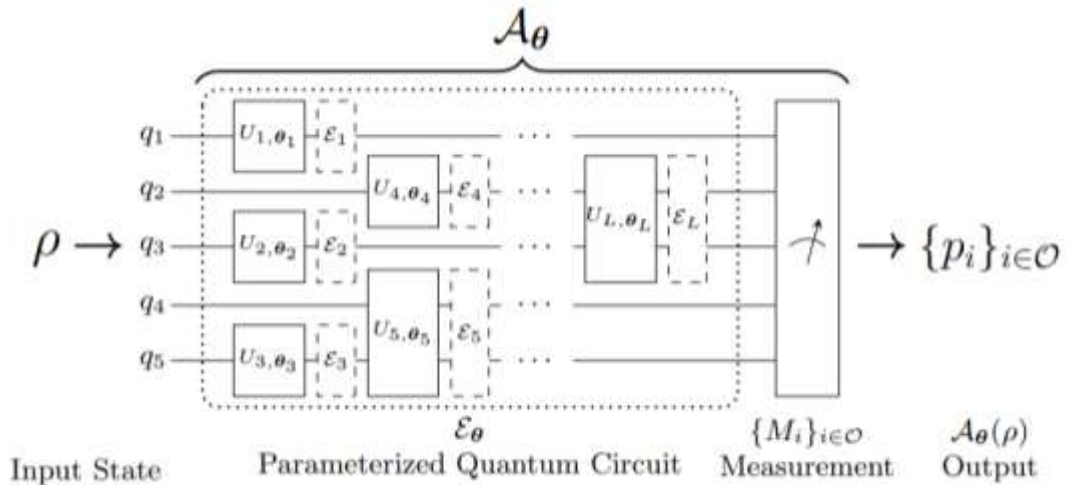
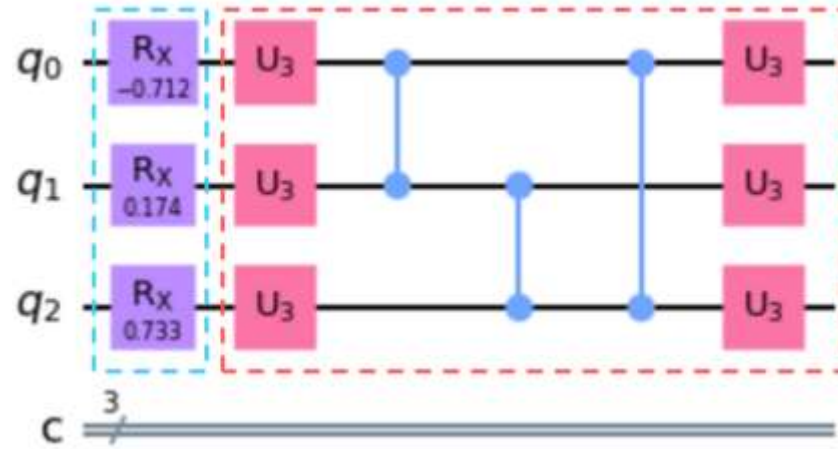


Fig. 1: Noisy Quantum (Machine Learning) Decision Model

Quantum Models: Due to the statistical nature of quantum mechanics, a quantum decision model is inherently a randomized mapping $\mathcal{A} : \mathcal{D}(\mathcal{H}) \rightarrow \mathcal{D}(\mathcal{O})$. Here $\mathcal{D}(\mathcal{H})$ is the set of *quantum states* (data) and to be specific latter. Inspired by the classical models, \mathcal{A} is not predefined but initialized as \mathcal{A}_θ by a parameterized quantum circuit \mathcal{E}_θ (see Fig. 1) with a set of free parameters $\theta = \{\theta_j\}_{j=1}^L$. Following the training strategy of classical machine learning, \mathcal{A}_θ is trained on a set of input quantum states (training dataset) by tuning θ subject to some loss function $\mathcal{L}(\theta)$.

2 Quantum Decision Models

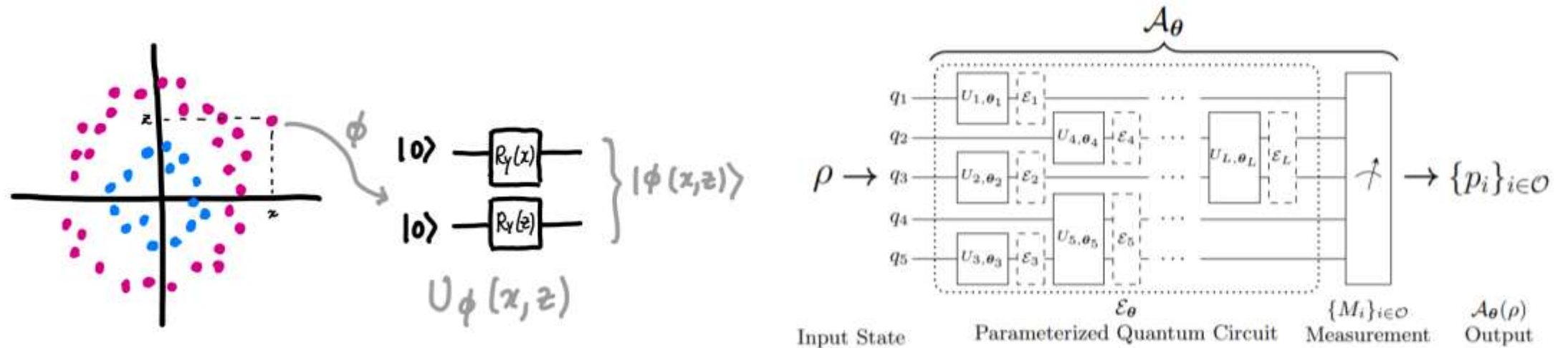


Fig. 1: Noisy Quantum (Machine Learning) Decision Model

Input State ρ : The input state of the model is a quantum *mixed state* ρ , which is mathematically modelled by a positive semi-definite complex matrix, written as $\rho \geq 0$, with unit trace².

a mixed state, meaning that ρ is at $|\psi_k\rangle$ with probability p_k . In particular, if $\rho = \psi$ for some pure state $|\psi\rangle$, then the ensemble is deterministic; that is, it is degenerated to a singleton $\{(1, \psi)\}$. In general, the statistical feature of ρ may result from quantum noise, which is unavoidable in the current NISQ era, from the surrounding environment.

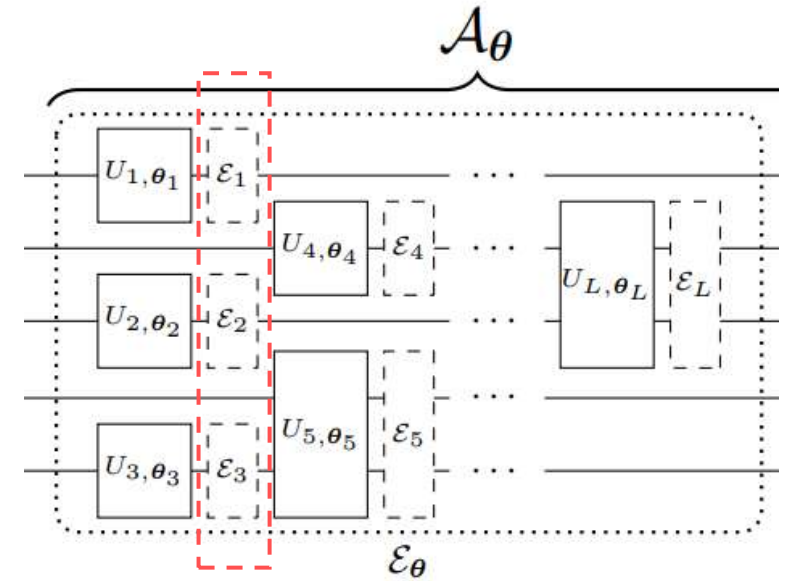
2 Quantum Decision Models

Parameterized Quantum Circuit \mathcal{E}_θ :

\mathcal{E}_θ consists of a sequence of quantum operations: $\mathcal{E}_\theta = \mathcal{E}_{d,\theta_d} \circ \dots \circ \mathcal{E}_{1,\theta_1}$. For each input quantum state ρ , the output of the circuit is $\mathcal{E}_\theta(\rho) = \mathcal{E}_{d,\theta_d}(\dots \mathcal{E}_{2,\theta_2}(\mathcal{E}_{1,\theta_1}(\rho)))$. In the current NISQ era, each component \mathcal{E}_{i,θ_i} is:

- either a parameterized quantum gate \mathcal{U}_{i,θ_i} (the full boxes in Fig 1) with $\mathcal{U}_{i,\theta_i}(\rho) = U_{i,\theta_i}\rho U_{i,\theta_i}^\dagger$, where U_{i,θ_i} is a unitary matrix with parameters θ_i , i.e., $U_{i,\theta_i}^\dagger U_{i,\theta_i} = U_{i,\theta_i} U_{i,\theta_i}^\dagger = I$ (the identity matrix), and U_{i,θ_i}^\dagger is the entry-wise conjugate transpose of U_{i,θ_i} ;
- or a quantum noise \mathcal{E}_i (the dashed boxes in Fig 1). Mathematically, it can be described by a family of Kraus matrices $\{E_{ij}\}$ [28]: $\mathcal{E}_i(\rho) = \sum_j E_{ij}\rho E_{ij}^\dagger$ with $\sum_j E_{ij}^\dagger E_{ij} = I$. Briefly, \mathcal{E}_i is represented as $\mathcal{E}_i = \{E_{ij}\}$.

$$\rightarrow \mathcal{E}(\rho) = (1 - p)\rho + p\frac{I}{N} \quad \forall \rho \in \mathcal{D}(\mathcal{H}),$$



2 Quantum Decision Models

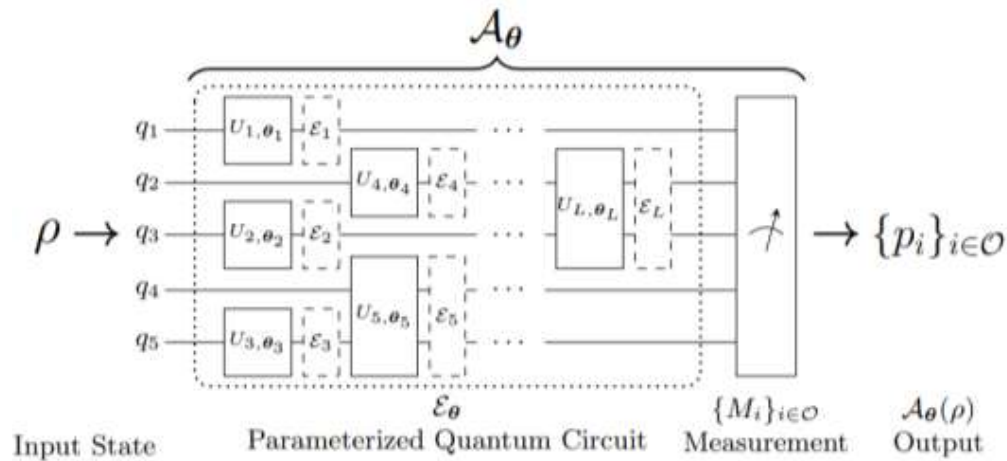


Fig. 1: Noisy Quantum (Machine Learning) Decision Model

modeled by a set $\{M_i\}_{i \in \mathcal{O}}$

$$\sum_{i \in \mathcal{O}} M_i^\dagger M_i = I.$$

$$\underline{p_i = \text{tr}(M_i \mathcal{E}_\theta(\rho) M_i^\dagger)}$$

Like their classical counterparts, quantum decision models are usually classified into two categories: *regression* and *classification* models. Regression models generally predict a value/quantity, whereas classification models predict a label/class. More specifically, a regression model $\mathcal{A}_\mathcal{R}$ uses the output of \mathcal{A} directly as the predicted value of the regression variable $\rho \in \mathcal{D}(\mathcal{H})$. That is $\mathcal{A}_\mathcal{R}(\rho) = \mathcal{A}(\rho)$ for all $\rho \in \mathcal{D}(\mathcal{H})$.

classification model $\mathcal{A}_\mathcal{C}$ further uses the measurement outcome probability distribution $\mathcal{A}(\rho)$ to sign a class label on the input state ρ . The most common way is as follows:

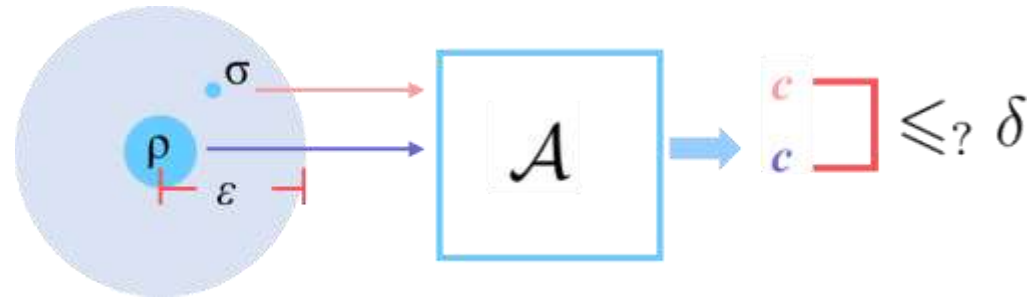
$$\mathcal{A}_\mathcal{C} : \mathcal{D}(\mathcal{H}) \rightarrow \mathcal{O} \quad \boxed{\mathcal{A}_\mathcal{C}(\rho) = \arg \max_i \mathcal{A}(\rho)_i} \quad \forall \rho \in \mathcal{D}(\mathcal{H}), i \in \mathcal{O},$$

where $\mathcal{A}(\rho)_i$ denotes the i -th element of distribution $\mathcal{A}(\rho)$.

3 Defining Fairness

Definition 2 (Bias Pair). Suppose we are given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$, two distance metrics $D(\cdot, \cdot)$ and $d(\cdot, \cdot)$ on $\mathcal{D}(\mathcal{H})$ and $\mathcal{D}(\mathcal{O})$, respectively, and two small enough threshold values $1 \geq \varepsilon, \delta > 0$. Then (ρ, σ) is said to be an (ε, δ) -bias pair if the following is true

$$\underline{[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]}. \quad (1)$$

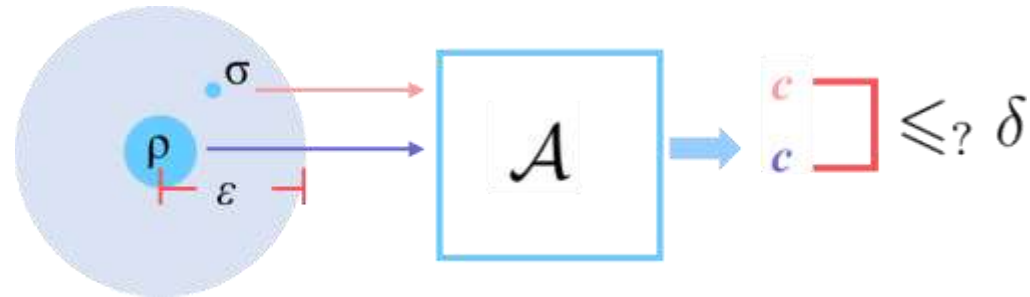


Definition 3 (Fair Model). Let $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ be a decision model. Then \mathcal{A} is (ε, δ) -fair if there is no any (ε, δ) -bias pair.

 global robustness

3 Defining Fairness

$$[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]$$



Input State ρ : The input state of the model is a quantum *mixed state* ρ , which is mathematically modelled by a positive semi-definite complex matrix, written as $\rho \geq 0$, with unit trace².

$$\Rightarrow D(\rho, \sigma) = \frac{1}{2} \text{tr}(|\rho - \sigma|)$$

Measurement $\{M_i\}_{i \in \mathcal{O}}$:



$$p_i = \text{tr}(M_i \mathcal{E}_\theta(\rho) M_i^\dagger)$$

$$\Rightarrow d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) = \frac{1}{2} \sum_i |\text{tr}(M_i^\dagger M_i \mathcal{E}(\rho - \sigma))|$$

Total variation distance:

$$\delta(P, Q) = \frac{1}{2} \|P - Q\|_1 = \frac{1}{2} \sum_{\omega \in \Omega} |P(\omega) - Q(\omega)|$$

4 Characterizing Fairness

4.1 Fairness and Lipschitz constant

Lemma 1. Let $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ be a quantum decision model. Then

$$d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) \leq D(\rho, \sigma). \quad (2)$$

Proof. This mainly results from [28, Theorem 9.1]: for any $\rho, \sigma \in \mathcal{D}(\mathcal{H})$

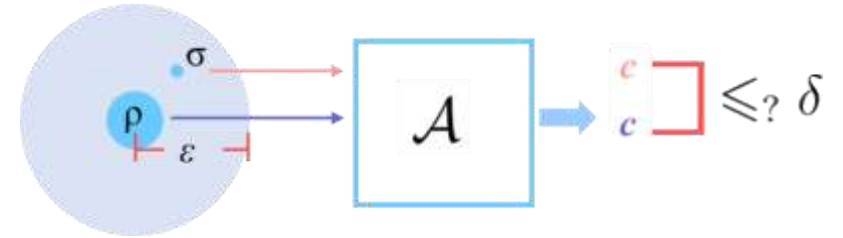
$$\text{tr}(|\rho - \sigma|) = \max_{\{M_i\}_i} \sum_i |\text{tr}(M_i^\dagger M_i(\rho - \sigma))|$$

where the maximization is over all quantum measurements $\{M_i\}_i$. We complete the proof by noting that $\{\sqrt{\mathcal{E}^\dagger(M_i^\dagger M_i)}\}$ is a measurement and

$$\text{tr}(M_i^\dagger M_i \mathcal{E}(\rho - \sigma)) = \text{tr}(\mathcal{E}^\dagger(M_i^\dagger M_i)(\rho - \sigma)),$$

where \mathcal{E}^\dagger is the conjugate map of \mathcal{E} .

$$[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]$$



$$\left\{ \begin{array}{l} D(\rho, \sigma) = \frac{1}{2} \text{tr}(|\rho - \sigma|) \\ d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) = \frac{1}{2} \sum_i |\text{tr}(M_i^\dagger M_i \mathcal{E}(\rho - \sigma))| \end{array} \right.$$

4 Characterizing Fairness

4.1 Fairness and Lipschitz constant

Lemma 1. Let $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ be a quantum decision model. Then

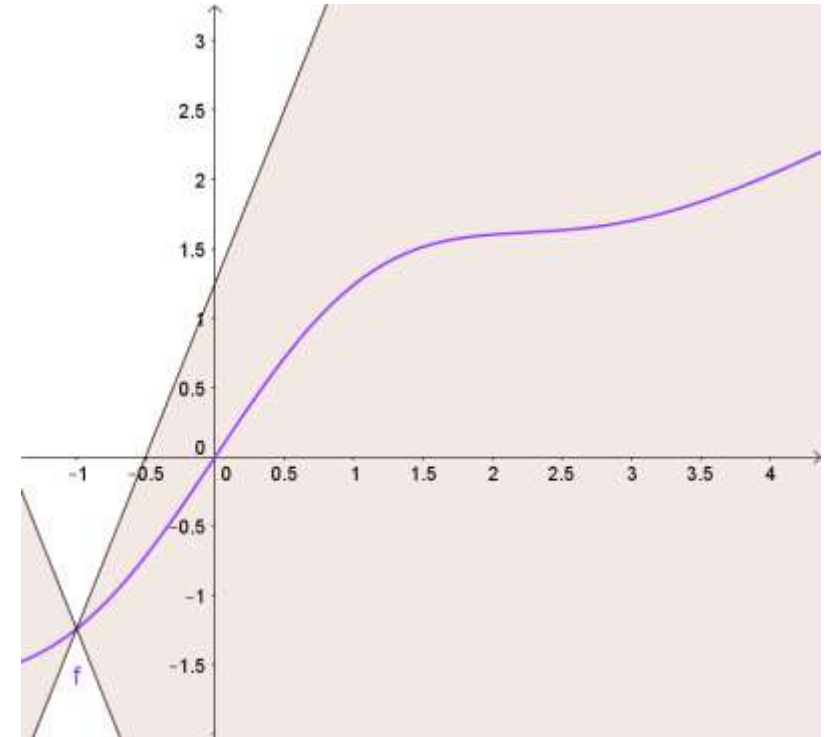
$$d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) \leq D(\rho, \sigma). \quad (2)$$

The above lemma indicates that quantum decision model \mathcal{A} is automatically (ε, δ) fair whenever $\varepsilon = \delta$. Furthermore, we see that \mathcal{A} is unconditionally Lipschitz continuous: there exists a constant $K > 0$ ($K \leq 1$ by Lemma 1) such that for all $\rho, \sigma \in \mathcal{D}(\mathcal{H})$,

$$d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) \leq K D(\rho, \sigma). \quad (3)$$

As usual, K is called a Lipschitz constant of \mathcal{A} . Furthermore, the smallest K , denoted by K^* , is called the (best) Lipschitz constant of \mathcal{A} .

In the context of quantum machine learning, the following theorem shows that K^* actually measures the fairness of decision model \mathcal{A} , i.e., the best (maximum) ratio of δ and ε in a fair model, and the states ψ, ϕ achieving K^* can be used to find bias pairs in fairness verification.



$$|f(x_1) - f(x_2)| \leq K |x_1 - x_2|$$

4 Characterizing Fairness

4.1 Fairness and Lipschitz constant

The above lemma indicates that quantum decision model \mathcal{A} is automatically (ε, δ) fair whenever $\varepsilon = \delta$. Furthermore, we see that \mathcal{A} is unconditionally *Lipschitz continuous*: there exists a constant $K > 0$ ($K \leq 1$ by Lemma 1) such that for all $\rho, \sigma \in \mathcal{D}(\mathcal{H})$,

$$d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) \leq KD(\rho, \sigma). \quad (3)$$

As usual, K is called a Lipschitz constant of \mathcal{A} . Furthermore, the smallest K , denoted by K^* , is called the (best) Lipschitz constant of \mathcal{A} .

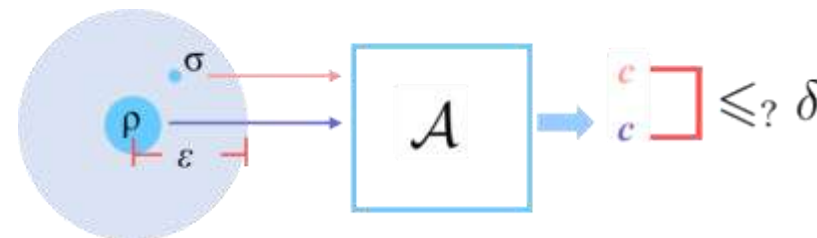
Theorem 1. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ and $1 \geq \varepsilon, \delta > 0$, \mathcal{A} is (ε, δ) -fair if and only if $\delta \geq K^* \varepsilon$.

$$\delta < d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) = K^* D(\rho, \sigma) \leq K^* \varepsilon$$

$$\rightarrow \delta < K^* \varepsilon$$

$$\delta \geq K^* \varepsilon$$

$$[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]$$



4 Characterizing Fairness

4.1 Fairness and Lipschitz constant

Theorem 1. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ and $1 \geq \varepsilon, \delta > 0$, \mathcal{A} is (ε, δ) -fair if and only if $\delta \geq K^* \varepsilon$.

2. If \mathcal{A} is not (ε, δ) -fair, then (ψ, ϕ) achieving K^* is a **bias kernel**, that is, for any quantum state $\sigma \in \mathcal{D}(\mathcal{H})$, (ρ_ψ, ρ_ϕ) is a bias pair where

$$\rho_\psi = \varepsilon\psi + (1 - \varepsilon)\sigma \quad \rho_\phi = \varepsilon\phi + (1 - \varepsilon)\sigma. \quad (4)$$

$$\Rightarrow \delta < K^* \varepsilon$$

By Theorem 3, we have that there exists a pair of mutually orthogonal pure quantum states $|\psi\rangle$ and $|\phi\rangle$ such that

$$d(\mathcal{A}(\psi), \mathcal{A}(\phi)) = K^* D(\psi, \phi).$$

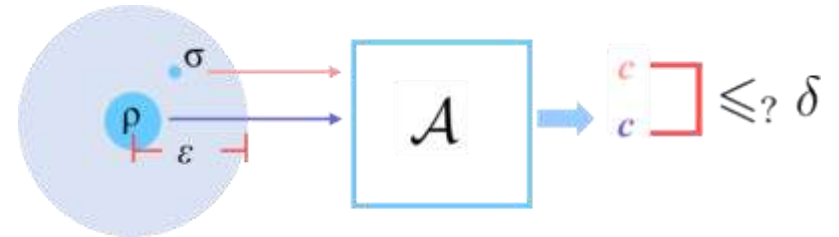
Note that $D(\psi, \phi) = 1$ results from the orthogonality of ψ and ϕ , i.e., $\langle \psi | \phi \rangle = 0$. Given a quantum state σ , let

$$\rho_\psi = \varepsilon\psi + (1 - \varepsilon)\sigma \quad \rho_\phi = \varepsilon\phi + (1 - \varepsilon)\sigma.$$

Then $D(\rho_\psi, \rho_\phi) = \varepsilon$.

$$d(\mathcal{A}(\rho_\psi), \mathcal{A}(\rho_\phi)) = K^* D(\rho_\psi, \rho_\phi) = \varepsilon K^* > \delta$$

$$[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]$$



4 Characterizing Fairness

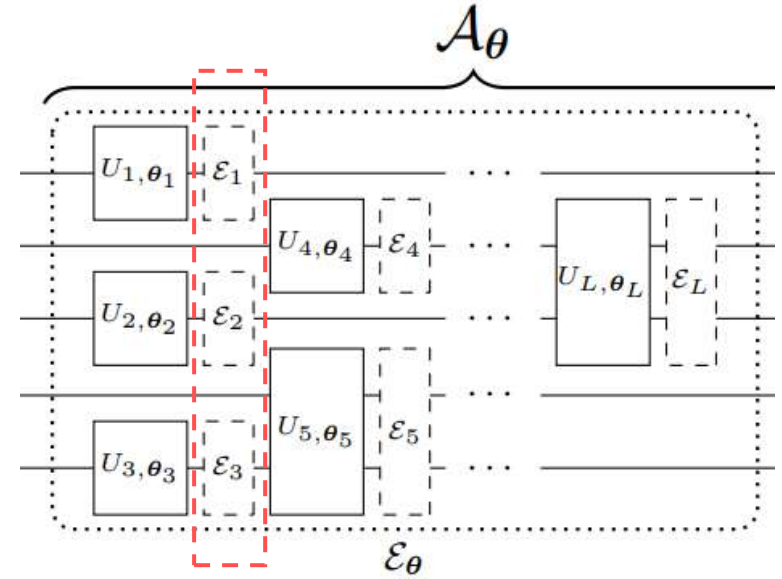
4.2 Fairness and Noises

In this subsection, we turn to consider the relation between fairness and noise. Let us first examine a simple example. Assume a noiseless quantum decision model $\mathcal{A} = (\mathcal{U}, \{\mathcal{M}_i\}_{i \in \mathcal{O}})$ where \mathcal{U} is a unitary operator, i.e., $\mathcal{U} = \{U\}$ for some unitary matrix U . The 1-qubit depolarizing noise in Example 2 can be generalized to a large-size system with the following form:

$$\mathcal{E}(\rho) = (1 - p)\rho + p \frac{I}{N} \quad \forall \rho \in \mathcal{D}(\mathcal{H}), \quad \rightarrow \quad K_{\mathcal{E}}^* = (1 - p)K^*$$

where $0 \leq p \leq 1$ and N is the dimension of the state space \mathcal{H} of the system.

➡ Phase flip, Bit flip, depolarizing, mixed noise



Theorem 2. Let $\mathcal{A} = (\mathcal{U}, \{\mathcal{M}_i\}_{i \in \mathcal{O}})$ be a quantum decision model. Then for any quantum noise represented by a super-operator \mathcal{E} , we have $K_{\mathcal{E}}^* \leq K^*$, where K^* and $K_{\mathcal{E}}^*$ are the Lipschitz constants of \mathcal{A} and $\mathcal{A}_{\mathcal{E}} = (\mathcal{E} \circ \mathcal{U}, \{\mathcal{M}_i\}_{i \in \mathcal{O}})$.

5 Fairness Verification

5.1 Computing Lipschitz Constant

Problem 1 (Fairness Verification Problem). Given a quantum decision model \mathcal{A} and $1 \geq \varepsilon, \delta > 0$, check whether or not \mathcal{A} is (ε, δ) -fair. If not then (at least) one bias pair (ρ, σ) is provided.

Theorem 1. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ and $1 \geq \varepsilon, \delta > 0$, \mathcal{A} is (ε, δ) -fair if and only if $\delta \geq K^* \varepsilon$.



Theorem 3. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$. The Lipschitz constant K^* is:

$$K^* = \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)] \text{ with } M_A = \sum_{i \in A} \mathcal{E}^\dagger(M_i^\dagger M_i),$$

where \mathcal{E}^\dagger is the conjugate map⁵ of \mathcal{E} , and $\lambda_{\max}(M_A)$ and $\lambda_{\min}(M_A)$ are the maximum and minimum eigenvalues of positive semi-definite matrix M_A , respectively.

5 Fairness Verification

5.1 Computing Lipschitz Constant

Theorem 3. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$. The Lipschitz constant K^* is:

$$K^* = \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)] \text{ with } M_A = \sum_{i \in A} \mathcal{E}^\dagger(M_i^\dagger M_i),$$

$$d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) \leq K D(\rho, \sigma).$$

$$\begin{cases} D(\rho, \sigma) = \frac{1}{2} \text{tr}(|\rho - \sigma|) \\ d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) = \frac{1}{2} \sum_i |\text{tr}(M_i^\dagger M_i \mathcal{E}(\rho - \sigma))| \end{cases}$$

Lemma 2. Let $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ be a quantum decision model and K^* the Lipschitz constant of \mathcal{A} . Then

$$2K^* = \max_{\rho, \sigma \in \mathcal{D}(\mathcal{H})} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))| = \max_{\rho \perp \sigma} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))|,$$

where $\mathcal{M}_i = M_i^\dagger M_i$ for all $i \in \mathcal{O}$ and $\rho \perp \sigma$ means that ρ is orthogonal to σ , i.e., $\text{tr}(\rho\sigma) = 0$.

Proof. First of all, by (3),

$$K^* = \max_{\rho, \sigma \in \mathcal{D}(\mathcal{H})} \frac{d(\mathcal{A}(\rho), \mathcal{A}(\sigma))}{D(\rho, \sigma)} = \max_{\rho, \sigma \in \mathcal{D}(\mathcal{H})} \frac{\sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))|}{\text{tr}(|\rho - \sigma|)}.$$

5 Fairness Verification

5.1 Computing Lipschitz Constant

Lemma 2. Let $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ be a quantum decision model and K^* the Lipschitz constant of \mathcal{A} . Then

$$2K^* = \max_{\rho, \sigma \in \mathcal{D}(\mathcal{H})} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))| = \max_{\rho \perp \sigma} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))|, \quad \longrightarrow \quad \{\mathcal{E}^\dagger(\mathcal{M}_i)\}$$

specifically, suppose that Alice is randomly given a state X from two prior known quantum states $\{\rho, \sigma\}$, and she wishes to use a quantum measurement $\{M_i\}_{i \in \mathcal{O}}$ to guess whether X is state ρ or state σ . The best strategy (successful probability) is as follows:

$$\begin{aligned} \Pr(\{M_i\}_{i \in \mathcal{O}} | \rho, \sigma) &= \frac{1}{2} \Pr(i \in A \mid X = \rho) + \frac{1}{2} \Pr(i \notin A \mid X = \sigma) \\ &= \frac{1}{2} + \frac{1}{4} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{M}_i(\rho - \sigma))| \end{aligned}$$

where $\mathcal{M}_i = M_i^\dagger M_i$ for all $i \in \mathcal{O}$, and $A = \{i \in \mathcal{O} \mid \text{tr}(\mathcal{M}_i \rho) \geq \text{tr}(\mathcal{M}_i \sigma)\}$




the distinguishability of a POVM $\{\mathcal{E}^\dagger(\mathcal{M}_i)\}_{i \in \mathcal{O}}$ is $\frac{1}{2}(1 + K^*)$.

5 Fairness Verification

5.1 Computing Lipschitz Constant

Lemma 2. Let $\mathcal{A} = (\mathcal{E}, \{\mathcal{M}_i\}_{i \in \mathcal{O}})$ be a quantum decision model and K^* the Lipschitz constant of \mathcal{A} . Then

$$2K^* = \max_{\rho, \sigma \in \mathcal{D}(\mathcal{H})} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))| = \max_{\rho \perp \sigma} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{E}^\dagger(\mathcal{M}_i)(\rho - \sigma))|,$$


 $2K^*(\{\mathcal{M}_i\}_{i \in \mathcal{O}}) = \max_{|\psi\rangle \perp |\phi\rangle} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{M}_i(\psi - \phi))|, \quad (\mathcal{E} = Id)$

$$\max_{|\psi\rangle \perp |\phi\rangle} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{M}_i(\psi - \phi))| = 2 \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(\mathcal{M}_A) - \lambda_{\min}(\mathcal{M}_A)].$$

First, we claim that the l.h.s is an upper bound of the r.h.s. For any subset $A \subseteq \mathcal{O}$,

$$\begin{aligned} & \max_{|\psi\rangle \perp |\phi\rangle} \sum_{i \in \mathcal{O}} |\text{tr}(\mathcal{M}_i(\psi - \phi))| \\ & \geq \max_{|\psi\rangle \perp |\phi\rangle} [|\text{tr}(\mathcal{M}_A(\psi - \phi))| + |\text{tr}(\mathcal{M}_{\mathcal{O} \setminus A}(\psi - \phi))|] \\ & = 2 \max_{|\psi\rangle \perp |\phi\rangle} |\text{tr}(\mathcal{M}_A(\psi - \phi))| \\ & = 2 \max_{|\psi\rangle \perp |\phi\rangle} \langle \psi | \mathcal{M}_A | \psi \rangle - \langle \phi | \mathcal{M}_A | \phi \rangle \\ & = 2 [\lambda_{\max}(\mathcal{M}_A) - \lambda_{\min}(\mathcal{M}_A)]. \end{aligned}$$

$\max_{|\psi\rangle} \langle \psi | \mathcal{M} | \psi \rangle = \lambda_{\max}(\mathcal{M})$
 $\min_{|\phi\rangle} \langle \phi | \mathcal{M} | \phi \rangle = \lambda_{\min}(\mathcal{M})$



5 Fairness Verification

5.1 Computing Lipschitz Constant

Theorem 3. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$. The Lipschitz constant K^* is:

$$K^* = \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)] \text{ with } M_A = \sum_{i \in A} \mathcal{E}^\dagger(M_i^\dagger M_i),$$

2. Furthermore, let $A^* \subseteq \mathcal{O}$ be an optimal solution of reaching the Lipschitz constant, i.e.,

$$A^* = \arg \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)]$$

and $|\psi\rangle$ and $|\phi\rangle$ be two normalized eigenvectors corresponding to the maximum and minimum eigenvalues of M_{A^*} , respectively. Then we have

$$d(\mathcal{A}(\psi), \mathcal{A}(\phi)) = K^* D(\psi, \phi) = K^*,$$

where $\psi = |\psi\rangle\langle\psi|$ and $\phi = |\phi\rangle\langle\phi|$.

5 Fairness Verification

5.1 Computing Lipschitz Constant

$$K^* = \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)] \text{ with } M_A = \sum_{i \in A} \mathcal{E}^\dagger(M_i^\dagger M_i),$$

Algorithm 1 Lipschitz(\mathcal{A})

Input: A quantum decision model $\mathcal{A} = (\mathcal{E} = \{E_j\}_{j \in \mathcal{J}}, \{M_i\}_{i \in \mathcal{O}})$ on a Hilbert space \mathcal{H} with dimension N .

Output: The Lipschitz constant K^* and (ψ, ϕ) as in Theorem 3.

$ \mathcal{O} $ ●	1: for each $i \in \mathcal{O}$ do N^2 2: $W_i = \mathcal{E}^\dagger(M_i^\dagger M_i) = \sum_{j \in \mathcal{J}} E_j^\dagger M_i^\dagger M_i E_j$ N^3 3: end for N^5
$2^{ \mathcal{O} }$ ●	4: $K^* = 0$, $A^* = \emptyset$ be an empty set and $M_{A^*} = \mathbf{0}$, zero matrix. 5: for each $A \subseteq \mathcal{O}$ do 6: $M_A = \sum_{i \in A} W_i$ and $K_A = \lambda_{\max}(M_A) - \lambda_{\min}(M_A)$ N^2 7: if $K_A > K^*$ then 8: $K^* = K_A$, $A^* = A$ and $M_{A^*} = M_A$ 9: end if 10: end for 11: $ \psi\rangle$ and $ \phi\rangle$ are obtained two normalized eigenvectors corresponding to the maximum and minimum eigenvalues of M_{A^*} , respectively. 12: return K^* and (ψ, ϕ)

$$O(|\mathcal{O}|N^5 + 2^{|\mathcal{O}|}N^2)$$

5 Fairness Verification

5.2 Fair Verification Algorithm

Theorem 1. 1. Given a quantum decision model $\mathcal{A} = (\mathcal{E}, \{M_i\}_{i \in \mathcal{O}})$ and $1 \geq \varepsilon, \delta > 0$, \mathcal{A} is (ε, δ) -fair if and only if $\delta \geq K^* \varepsilon$.
 2. If \mathcal{A} is not (ε, δ) -fair, then (ψ, ϕ) achieving K^* is a bias kernel; that is, for any quantum state $\sigma \in \mathcal{D}(\mathcal{H})$, (ρ_ψ, ρ_ϕ) is a bias pair where

$$\rho_\psi = \varepsilon \psi + (1 - \varepsilon) \sigma \quad \rho_\phi = \varepsilon \phi + (1 - \varepsilon) \sigma. \quad (4)$$

Algorithm 2 FairVeriQ($\mathcal{A}, \varepsilon, \delta$)

Input: A quantum decision model $\mathcal{A} = (\mathcal{E} = \{E_j\}_{j \in \mathcal{J}}, \{M_i\}_{i \in \mathcal{O}})$ on a Hilbert space \mathcal{H} with dimension N , and real numbers $1 \geq \varepsilon, \delta > 0$.

Output: **true** indicates \mathcal{A} is (ε, δ) -fair or **false** with a bias kernel pair (ψ, ϕ) indicates

\mathcal{A} is not (ε, δ) -fair.

1: $(K^*, (\psi, \phi)) = \text{Lipschitz}(\mathcal{A})$

// Call Algorithm 1

2: **if** $\delta \geq K^* \varepsilon$ **then**

3: **return true**

4: **else**

5: **return false** and (ψ, ϕ)

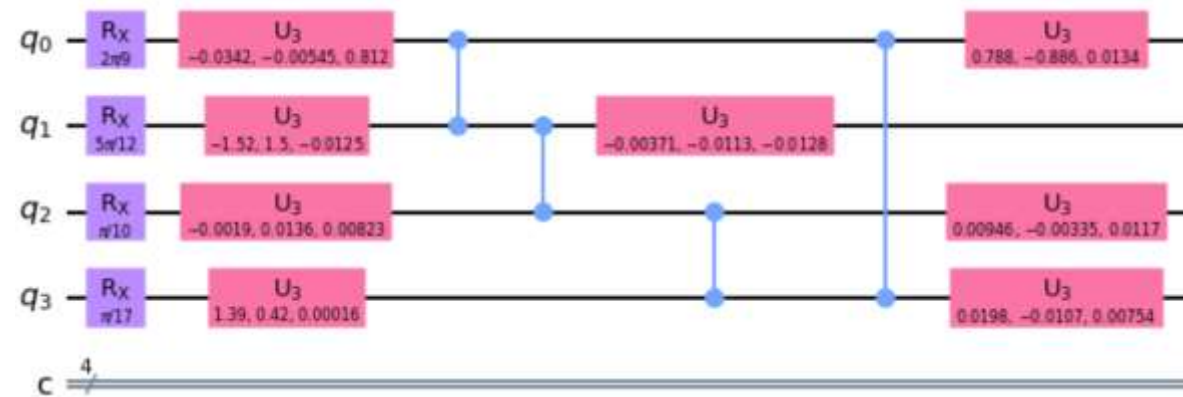
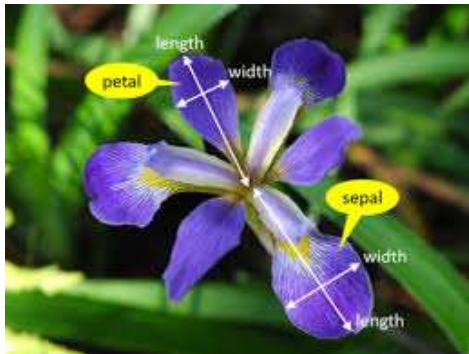
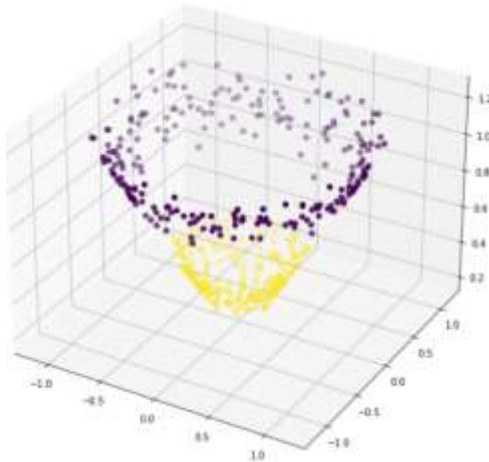
6: **end if**

$$O(|\mathcal{O}|N^5 + 2^{|\mathcal{O}|}N^2)$$

6 Evaluate

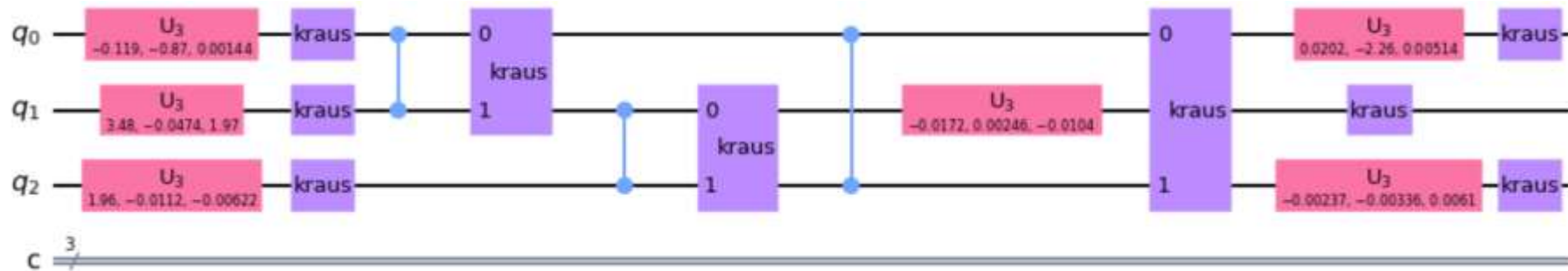
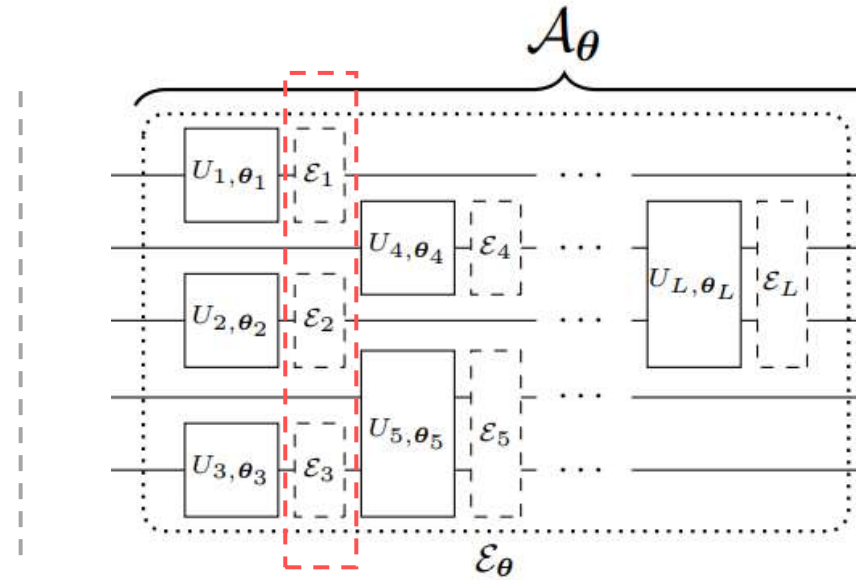
Dataset	Noise		Accuracy		K^*	Time
	type	probability	train	test		
German Credit	None		0.732	0.686	1.0000×10^0	\
	Phase flip	10^{-4}	0.726	0.692	9.9997×10^{-1}	2.36s
		10^{-3}	0.724	0.714	9.9800×10^{-1}	2.02s
		10^{-2}	0.704	0.708	9.6918×10^{-1}	1.94s
	Depolarize	10^{-4}	0.709	0.686	9.9977×10^{-1}	2.77s
		10^{-3}	0.701	0.712	9.9789×10^{-1}	2.93s
		10^{-2}	0.709	0.682	9.7916×10^{-1}	3.44s
	Bit flip	10^{-4}	0.712	0.728	9.9975×10^{-1}	2.27s
		10^{-3}	0.710	0.690	9.9743×10^{-1}	2.47s
		10^{-2}	0.724	0.678	9.7981×10^{-1}	2.05s
	Mixed noise	10^{-4}	0.710	0.704	9.9980×10^{-1}	2.15s
		10^{-3}	0.731	0.682	9.9834×10^{-1}	2.08s
		10^{-2}	0.731	0.692	9.7021×10^{-1}	1.95s

6 Evaluate



6 Evaluate

```
# QuantumError objects
if errorType == 'bit_flip':
    error = pauli_error([('X', p), ('I', 1 - p)])
elif errorType == 'phase_flip':
    error = pauli_error([('Z', p), ('I', 1 - p)])
elif errorType == 'depolarizing':
    error = depolarizing_error(p, num_qubits=1)
```



6 Evaluate

Evaluate <noiseless>:
FINAL ACCURACY: 89.00%

Evaluate <bit_flip>:
FINAL ACCURACY: 89.00%

Evaluate <phase_flip>:
FINAL ACCURACY: 91.00%

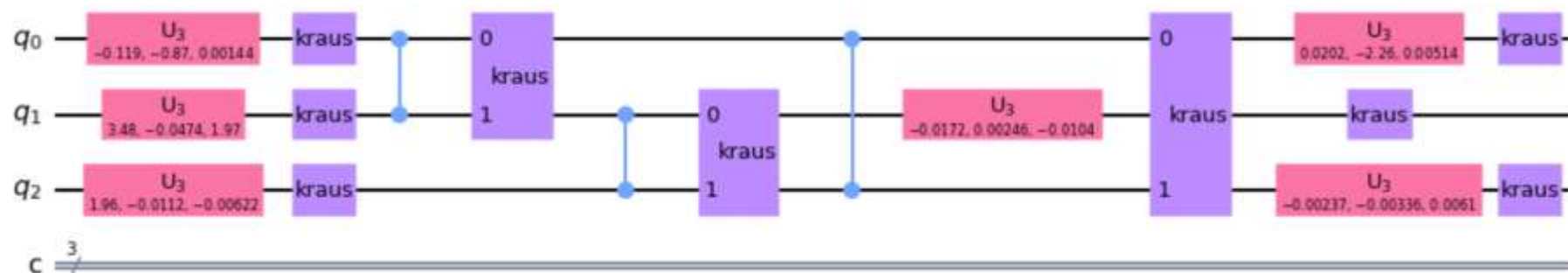
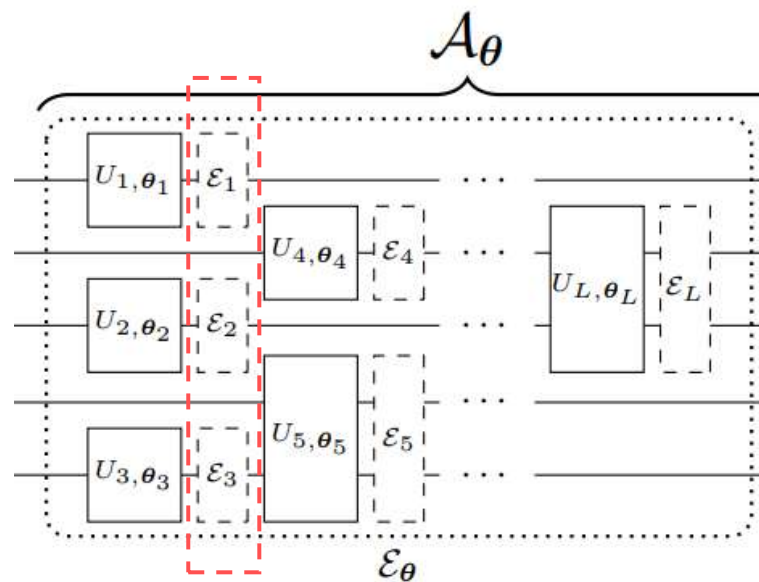
Evaluate <depolarizing>:
FINAL ACCURACY: 89.00%

Evaluate <noiseless>:
FINAL ACCURACY: 100.00%

Evaluate <bit_flip>:
FINAL ACCURACY: 100.00%

Evaluate <phase_flip>:
FINAL ACCURACY: 100.00%

Evaluate <depolarizing>:
FINAL ACCURACY: 100.00%



6 Evaluate

$|O|$

```
# Step 1: Calculate W_i
W = dict()
for i in O:
    W[i] = Dag(E) @ Dag(M[i]) @ M[i] @ E
```

N^5

$2^{|O|}$

```
# Step 2: Calculate K_star
K_star = 0; vectors = [None, None]
M_star = np.zeros(E.shape)
```

```
for S in O:
    if len(S) == 0:
        continue
```

```
# calculate M_S = Σ W_i
M_S = np.zeros(E.shape).astype('complex64')
for i in S:
    M_S += W[i]

# calculate eigenvalues and eigenvectors of M_S
eigenvalues, eigenvectors = np.linalg.eigh(M_S)
min_index = np.where(eigenvalues == eigenvalues.min())
max_index = np.where(eigenvalues == eigenvalues.max())

# calculate K_S
K_S = np.linalg.norm(eigenvalues[max_index][0] - eigenvalues[min_index])
```

N^2

```
if K_S > K_star:
    K_star = K_S
    vectors[0] = eigenvectors.T[max_index][0]
```

$$K^* = \max_{A \subseteq O} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)]$$

Classifier for Iris:

▲ (1.0, array([[5.1224810e-01+0.0000000e+00j, 1.2805447e-01-1.5716428e-01j,
-4.6464182e-02-4.8011953e-01j, -1.7507088e-01-1.1697307e-01j,
1.9302657e-03-2.3970471e-05j, 4.7483321e-04-5.9813249e-04j,
-2.5687154e-04-2.7331561e-03j, -9.9440350e-04-6.6813920e-04j,
3.9289543e-01+1.7302768e-01j, 1.6676332e-01-8.4787413e-02j,
1.3040526e-01-3.9493093e-01j, -8.8753097e-02-1.3898401e-01j,
2.2194656e-03+9.8545512e-04j, 9.4497600e-04-4.7709129e-04j,
4.7669682e-04-1.5082960e-03j, -3.4465280e-04-5.2442454e-04j],
[3.1758568e-01+0.0000000e+00j, 2.9468201e-02-1.8393441e-01j,
-2.2971131e-01+6.4443976e-01j, 2.4498379e-01-2.7509290e-01j,
-5.5171194e-04-8.1406842e-04j, 1.2663535e-04-7.7127159e-04j,
2.1664377e-03-9.2658226e-04j, 4.6017810e-04-7.5275602e-04j,
-9.2287837e-03-1.5119736e-01j, 1.0151771e-01-2.3656576e-03j,
4.2002317e-01-1.1694973e-01j, -1.7974521e-01-7.0664361e-02j,
-1.3843905e-03+2.1386005e-03j, 1.0767560e-03+5.4406049e-04j,
7.4833655e-04-1.9062139e-03j, 4.1396494e-04-2.4905446e-04j]],
dtype=complex64))

6 Evaluate

$|\mathcal{O}|$

```
# Step 1: Calculate W_i
W = dict()
for i in 0:
    W[i] = Dag(E) @ Dag(M[i]) @ M[i] @ E
```

N^5

$2^{|\mathcal{O}|}$

```
# Step 2: Calculate K_star
K_star = 0; vectors = [None, None]
M_star = np.zeros(E.shape)
```

```
for S in 0_:
    if len(S) == 0:
        continue
```

```
# calculate M_S = Σ W_i
M_S = np.zeros(E.shape).astype('complex64')
for i in S:
    M_S += W[i]

# calculate eigenvalues and eigenvectors of M_S
eigenvalues, eigenvectors = np.linalg.eigh(M_S)
min_index = np.where(eigenvalues == eigenvalues.min())
max_index = np.where(eigenvalues == eigenvalues.max())
```

N^2

```
# calculate K_S
K_S = np.linalg.norm(eigenvalues[max_index][0] - eigenvalues[min_index])

if K_S > K_star:
    K_star = K_S
    vectors[0] = eigenvectors.T[max_index][0]
```

$$K^* = \max_{A \subseteq \mathcal{O}} [\lambda_{\max}(M_A) - \lambda_{\min}(M_A)]$$

Classifier for Iris:
 p: 0.0001
 noiseless model: 1.0
 bit_flip model: 0.9974957704544067
 phase_flip model: 0.9983262419700623
 depolarizing model: 0.9986007213592529

p: 0.001
 noiseless model: 1.0
 bit_flip model: 0.975249707698822
 phase_flip model: 0.9834035038948059
 depolarizing model: 0.9860955476760864

p: 0.05
 noiseless model: 1.0
 bit_flip model: 0.29632073640823364
 phase_flip model: 0.4548749029636383
 depolarizing model: 0.4973519444465637

p: 0.1
 noiseless model: 1.0
 bit_flip model: 0.09604907780885696
 phase_flip model: 0.23411400616168976
 depolarizing model: 0.249342143535614



Evaluate <noiseless>:
 FINAL ACCURACY: 100.00%

Evaluate <bit_flip>:
 FINAL ACCURACY: 95.00%

Evaluate <phase_flip>:
 FINAL ACCURACY: 100.00%

Evaluate <depolarizing>:
 FINAL ACCURACY: 100.00%

6 Evaluate

```
def FairVeriQ(A, epsilon, delta):  
    # epsilon <= 1 and delta > 0  
    K_star, kernel = Lipschitz(A)  
  
    if delta >= K_star * epsilon:  
        return True, None  
    else:  
        return False, kernel
```

Iris – $p=0.001$, phase_flip

Lipschitz constant: 0.9834035038948059

A is (1,0.99)-fair: <True>

A is (0.8,0.79)-fair: <True>

A is (0.5,0.3)-fair: <False>

- bias kernel pair is: [[5.12724161e-01+0.0000000e+00j 1.28032640e-01-1.5713547e-01j
-4.64939214e-02-4.8043025e-01j -1.74887925e-01-1.1684963e-01j
1.93114916e-03-2.3846040e-05j 4.74575209e-04-5.9770868e-04j
-2.56348285e-04-2.7266780e-03j -9.90394386e-04-6.6539587e-04j
3.92745733e-01+1.7295834e-01j 1.66417331e-01-8.4614113e-02j
1.30292282e-01-3.9460191e-01j -8.85802880e-02-1.3871221e-01j
2.21203268e-03+9.8205451e-04j 9.40186088e-04-4.7472160e-04j
4.76128538e-04-1.5061955e-03j -3.43756721e-04-5.2313803e-04j]
[1.20339826e-01-0.0000000e+00j 2.04094738e-01-1.3560200e-01j
7.11197257e-02-6.2828250e-02j 4.01143461e-01+5.6327677e-01j
-4.79472615e-03-5.2465363e-03j -1.07054040e-03+1.1039445e-03j
-4.30674199e-03+3.5000923e-03j -1.87959417e-03-1.5740625e-03j
-3.30644876e-01-2.8429070e-01j 4.30287540e-01+2.0151950e-02j
1.10182323e-01-1.9434731e-01j -7.73455948e-02-8.0621324e-02j
8.03802907e-03+3.7426520e-03j 1.15413719e-03-8.2395109e-04j
-2.48990348e-03+1.5234994e-03j -7.15605449e-04+6.0832815e-04j]]

6 Evaluate

$$\rho_\psi = \varepsilon\psi + (1 - \varepsilon)\sigma \quad \rho_\phi = \varepsilon\phi + (1 - \varepsilon)\sigma.$$

```
def generateBiasPair(sigma, kernel, epsilon):
    psi, phi = kernel
    size = len(psi)
    psi = psi.reshape(size, 1) @ Dag(psi.reshape(size, 1))
    phi = phi.reshape(size, 1) @ Dag(phi.reshape(size, 1))

    rou_psi = epsilon * psi + (1 - epsilon) * sigma
    rou_phi = epsilon * phi + (1 - epsilon) * sigma

    return np.array([
        rou_psi, rou_phi
    ])
```

Lipschitz constant: 0.9834035038948059

fairness: False

epsilon=0.9, delta=0.3

D: 0.9000000232731683

d: 0.3912155256508235

$$[D(\rho, \sigma) \leq \varepsilon] \wedge [d(\mathcal{A}(\rho), \mathcal{A}(\sigma)) > \delta]$$

Iris – p=0.001, phase_flip

Lipschitz constant: 0.9834035038948059

A is (1, 0.99)-fair: <True>

A is (0.8, 0.79)-fair: <True>

A is (0.5, 0.3)-fair: <False>

- bias kernel pair is: [[5.12724161e-01+0.0000000e+00j 1.28032640e-01-1.5713547e-01j
-4.64939214e-02-4.8043025e-01j -1.74887925e-01-1.1684963e-01j
1.93114916e-03-2.3846040e-05j 4.74575209e-04-5.9770868e-04j
-2.56348285e-04-2.7266780e-03j -9.90394386e-04-6.6539587e-04j
3.92745733e-01+1.7295834e-01j 1.66417331e-01-8.4614113e-02j
1.30292282e-01-3.9460191e-01j -8.85802880e-02-1.3871221e-01j
2.21203268e-03+9.8205451e-04j 9.40186088e-04-4.7472160e-04j
4.76128538e-04-1.5061955e-03j -3.43756721e-04-5.2313803e-04j]
[1.20339826e-01-0.0000000e+00j 2.04094738e-01-1.3560200e-01j
7.11197257e-02-6.2828250e-02j 4.01143461e-01+5.6327677e-01j
-4.79472615e-03-5.2465363e-03j -1.07054040e-03+1.1039445e-03j
-4.30674199e-03+3.5000923e-03j -1.87959417e-03-1.5740625e-03j
-3.30644876e-01-2.8429070e-01j 4.30287540e-01+2.0151950e-02j
1.10182323e-01-1.9434731e-01j -7.73455948e-02-8.0621324e-02j
8.03802907e-03+3.7426520e-03j 1.15413719e-03-8.2395109e-04j
-2.48990348e-03+1.5234994e-03j -7.15605449e-04+6.0832815e-04j]]

6 Evaluate

