Quantum Machine Intelligence (2022) 4:28
https://doi.org/10.1007/s42484-022-00083-z

RESEARCH ARTICLE

A quantum generative adversarial network for distributions

Amine Assouel[1] · **Antoine Jacquier**[2,3] · **Alexei Kondratyev**[2,4]

Received: 3 December 2021 / Accepted: 3 August 2022 / Published online: 26 September 2022

Abstract
Recent advances in Quantum Computing have shown that, despite the absence of a fault-tolerant quantum computer so far, quantum techniques are providing exponential advantage over their classical counterparts. We develop a fully connected Quantum Generative Adversarial network and show how it can be applied in Mathematical Finance, with a particular focus on volatility modelling.

Keywords Quantum computing · GAN · Quantum phase estimation · SVI · Volatility

1 Introduction

Machine Learning has become ubiquitous, with applications in nearly every aspect of society today, in particular for image and speech recognition, traffic prediction, product recommendation, medical diagnosis, stock market trading and fraud detection. One specific Machine Learning tool, deep neural networks, has seen tremendous developments over the past few years. Despite clear advances, these networks however often suffer from the lack of training data: in Finance, time series of a stock price only occur once, physical experiments are sometimes expensive to run many times. To palliate this, attention has turned to methods aimed at reproducing existing data with a high degree of accuracy. Among these, Generative Adversarial Networks (GAN) are a class of unsupervised Machine Learning devices whereby two neural networks, a generator and a discriminator, contest against each other in a minimax game in order to generate information similar to a given dataset (Goodfellow et al. 2014). They have been successfully applied in many fields over the past few years, in particular for image generation (Yu et al. 2018; Schawinski et al. 2017), medicine (Anand and Huang 2018; Zhavoronkov 2019), and in Quantitative Finance (Ruf and Wang 2021). They however often suffer from instability issues, vanishing gradient and potential mode collapse (Saxena and Cao 2021). Even Wasserstein GANs, assuming the Wasserstein distance from optimal transport instead of the classical Jensen–Shannon Divergence, are still subject to slow convergence issues and potential instability (Gulrajani et al. 2017).

In order to improve the accuracy of this method, Lloyd and Weedbrook (2018) and Dallaire-Demers and Killoran (Dallaire-Demers and Killoran 2018) simultaneously introduced a quantum component to GANs, where the data consists of quantum states or classical data while the two players are equipped with quantum information processors. Preliminary works have demonstrated the quality of this approach, in particular for high-dimensional data, thus leveraging on the exponential advantage of quantum computing (Huang et al. 2021). An experimental proof-of-principle demonstration of QuGAN in a superconducting quantum circuit was shown in Hu et al. (2019), while in Stein et al. (2020) the authors made use of quantum fidelity measurements to propose a loss function acting on quantum states. Further recent advances, providing more insights on how quantum entanglement can play a decisive role, have been put forward in Niu et al. (2022).

✉ Antoine Jacquier
a.jacquier@imperial.ac.uk

Amine Assouel
amine.assouel@ens-paris-saclay.fr

Alexei Kondratyev
a.kondratyev@imperial.ac.uk

1 ENS Paris-Saclay, Gif-sur-Yvette, France

2 Department of Mathematics, Imperial College London, London, UK

3 Alan Turing Institute, London, UK

4 Abu Dhabi Investment Authority (ADIA), Abu Dhabi, United Arab Emirates

While actual Quantum computers are not available yet, Noisy intermediate-scale quantum (NISQ) algorithms are already here and allow us to perform quantum-like operations (Bharti et al. 2021). The importance of such computations appear can be seen through the lens of data. Indeed, over the past five years, Quantitative Finance has put a large emphasis on data-based models (with the use of deep learning and reinforcement learning), with the obvious increasing need for large amount of data for training purposes. Generative models (Kondratyev and Schwarz 2019) have thus found themselves key to help generate (any amount of) realistic data that can then be used for training, and any computational speedup (due to the extremely large size of these datasets), is urgently welcome; in particular that of quantum computing. In fact, quoting from (Herman et al. 2022), 'Numerous financial use cases require the ability to assess a wide range of potential outcomes. To do this, banks employ algorithms and models that calculate statistical probabilities. Such techniques are fairly effective, but not infallible. In a world where huge amounts of data are generated daily, computers that can compute probabilities accurately are becoming a predominant need. For this reason, several banks are turning to quantum computing given its promise to analyse vast amounts of data and compute results faster and more accurately than what any classical computer has ever been able to do'.

We focus here on building a fully connected Quantum Generative Adversarial network (QuGAN) [1], namely an entire quantum counterpart to a classical GAN. A quantum version of GAN was first introduced in Dallaire-Demers and Killoran (2018) and Lloyd and Weedbrook (2018), showing that it may exhibit an exponential advantage over classical adversarial networks. We should also like to mention some closely related works, in particular Situ et al. (2020), making clever use of Matrix Product State (MPS) quantum circuits, Nakaji and Yamamoto (2021) for classification and Zoufal et al. (2019), where the generated distributions are brilliantly used to bypass the need to load classical data in quantum computers (here for option pricing purposes), a standard bottleneck in quantum algorithms. However, all these advances use a quantum generator and a classical discriminator, slightly different from our approach here, which builds a fully quantum GAN.

The paper is structured as follows: In Section 2, we recall the basics of a classical neural network and show how to build a fully quantum version of it. This is incorporated in the full architecture of a Quantum Generative Adversarial Network in Section 3. Since classical GANs are becoming an important focus in Quantitative Finance (Koshiyama et al. 2021; Buehler et al. 2019; Ni et al. 2020; Wiese et al. 2020), we provide an example of application for QuGAN for volatility modelling in Section 4, hoping to bridge the gap between the Quantum Computing and the Quantitative Finance communities. For completeness, we gather some essential background on Quantum Computing in Appendix A.

[1] The terminology 'QuGAN' should not be confused with 'QGAN', used to denote quantised versions of GAN, as in (Wang et al. 2019), nor with 'Quant GAN', which refers (Wiese et al. 2020) to the use of GAN in Quantitative Finance; neither Quant GAN nor QGAN are related whatsoever to Quantum Computing.

2 A quantum version of a non-linear quantum neuron

The quantum phase estimation procedure lies at the very core of building a quantum counterpart for a neural network. In this part, we will mainly focus on how to build a single quantum neuron. As the fundamental building block of artificial neural networks, a neuron classically maps a normalised input $\mathbf{x} = (x_0, \ldots, x_{n-1})^\top \in [0,1]^n$ to an output $g(\mathbf{x}^\top \mathbf{w})$, where $\mathbf{w} = (w_0, \ldots, w_{n-1})^\top \in [-1,1]^n$ is the weight vector, for some activation function g. The non-linear quantum neuron requires the following steps:

- Encode classical data into quantum states (Section 2.2);
- Perform the (quantum version of the) inner product $\mathbf{x}^\top \mathbf{w}$ (Section 2.3);
- Applying the (quantum version of the) non-linear activation function (Section 2.4).

Before diving into the quantum version of neural networks, we recall the basics of classical (feedforward) neural networks, which we aim at mimicking.

2.1 Classical neural network architecture

Artificial neural networks (ANNs) are a subset of machine learning and lie at the heart of Deep Learning algorithms. Their name and structure are inspired by the human brain (Marblestone et al. 2016), mimicking the way that biological neurons signal to one another. They consist of several layers, with an input layer, one or more hidden layers, and an output layer, each one of them containing several nodes. An example of ANN is depicted in Fig. 1.

For a given an input vector $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$, the connectivity between $\mathbf{x}$ and the jth neuron $h_j^{(1)}$ of the first hidden layer (Fig. 1) is done via $h_j^{(1)} = \sigma_{1,j}(b_{1,j} + \sum_{i=1}^n x_i w_{i,j})$, where $\sigma_{1,j}$ is called the activation function. By denoting $H_k \in \mathbb{R}^{s_k}$ the vector of the kth hidden layer, where $s_k \in \mathbb{N}^*$ and $H_k = (h_1^{(k)}, \ldots, h_{s_k}^{(k)})$ the connectivity model generalises itself to the whole network:

$$h_j^{(k+1)} = \sigma_{k+1,j}\left(b_{k+1,j} + \sum_{i=1}^{s_k} h_i^{(k)} w_{i,k+1,j}\right), \tag{2.1}$$

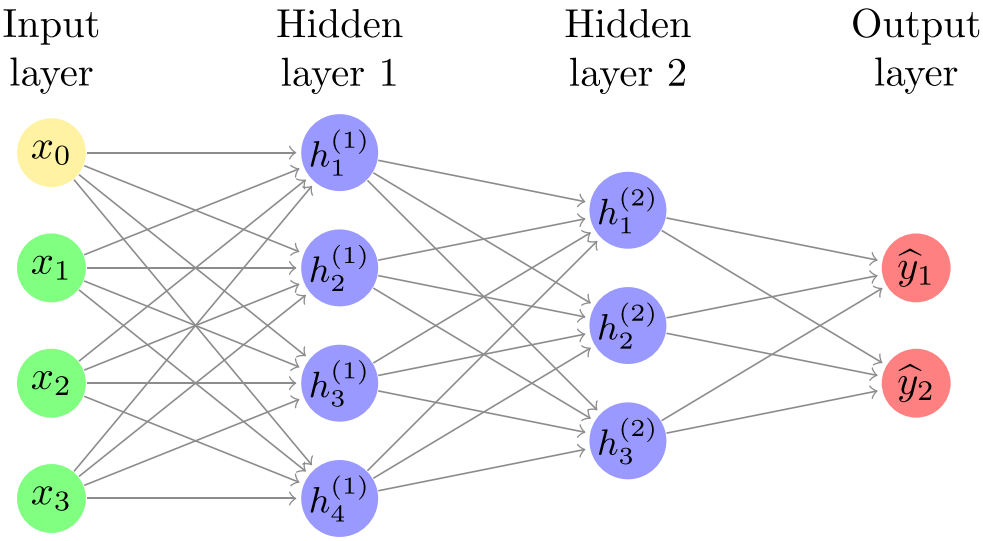


Fig. 1 ANN with one input layer, 2 hidden layers and one output layer

where $j \in \{1, \ldots, s_{k+1}\}$. Therefore for l hidden layers the entire network is parameterised by $\Omega = (\sigma_{k,r_k}, b_{k,r_k}, w_{v_k,k,r_k})_{k,r_k,v_k}$ where first $1 \leq k \leq l$, then $1 \leq r_k \leq s_k$ and $1 \leq v_k \leq s_{k-1}$. For a given training data set of size N, $(X_i, Y_i)_{i=1,\ldots,N}$, the goal of a neural network is to build a mapping between $(X_i)_{i=1,\ldots,N}$ and $(Y_i)_{i=1,\ldots,N}$. The idea for the neural network structure comes from the Kolmogorov-Arnold representation Theorem (Arnold 1957; Kolmogorov 1956):

Theorem 2.1 *Let $f : [0,1]^d \to \mathbb{R}$ be a continuous function. There exist sequences $(\Phi_i)_{i=1,\ldots,2d}$ and $(\Psi_{i,j})_{i=1,\ldots,2d;i=1,\ldots,d}$ of continuous functions from $\mathbb{R}$ to $\mathbb{R}$ such that for all $(x_1, \ldots, x_d) \in [0,1]^d$,*

$$f(x_1, \ldots, x_d) = \sum_{i=1}^{2d} \Phi_i \left(\sum_{j=1}^{d} \Psi_{i,j}(x_j) \right). \tag{2.2}$$

The representation of f resembles a two-hidden layer ANN, where $\Phi_i, \Psi_{i,j}$ are the activation functions.

2.2 Quantum encoding

Since a quantum computer only takes qubits as inputs, we first need to encode the classical data into a quantum state. For $x_j \in [0,1]$ and $p \in \mathbb{N}$, denote by $\frac{x_{j,1}}{2} + \frac{x_{j,2}}{2^2} + \ldots + \frac{x_{j,p}}{2^p}$ the p-binary approximation of x_j, where each $x_{j,k}$ belongs to $\{0,1\}$, for $k \in \{1,2,\ldots,p\}$. The quantum code for the classical value x_j is then defined via this approximation as

$$|x_j\rangle := |x_{j,1}\rangle \otimes |x_{j,2}\rangle \otimes \ldots \otimes |x_{j,p}\rangle = |x_{j,1}x_{j,2}\ldots x_{j,p}\rangle,$$

and therefore the encoding for the vector $\mathbf{x}$ is

$$|\mathbf{x}\rangle := |x_{0,1}x_{0,2}\ldots x_{0,p}\rangle \otimes \ldots \otimes |x_{n-1,1}\ldots x_{n-1,p}\rangle. \tag{2.3}$$

2.3 Quantum inner product

We now show how to build the quantum version of the inner product performing the operation

$$|0\rangle^{\otimes m}|\mathbf{x}\rangle \to |\widetilde{\mathbf{x}}^\top \mathbf{w}\rangle|\mathbf{x}\rangle.$$

Denote the two-qubit controlled Z-Rotation gate by

$${}_c\mathrm{R}_z(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \mathrm{e}^{2\mathrm{i}\pi\alpha} \end{pmatrix},$$

where α is the phase shift with period π. For $x \in \{0,1\}$ and $|+\rangle := \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, note that, for $k \in \mathbb{N}$,

$${}_c\mathrm{R}_z\left(\frac{1}{2^k}\right)(|+\rangle|x\rangle) = \frac{1}{\sqrt{2}}\left(|0\rangle|x\rangle + \exp\left\{\frac{2\mathrm{i}\pi x}{2^k}\right\}|1\rangle|x\rangle\right)$$

Indeed, either $x = 0$ and then $|x\rangle = |0\rangle$ so that

$${}_c\mathrm{R}_z\left(\frac{1}{2^k}\right)(|+\rangle|x\rangle) = \frac{1}{\sqrt{2}}\left(|0\rangle|0\rangle + |1\rangle|0\rangle\right),$$

or $x = 1$ and hence

$${}_c\mathrm{R}_z\left(\frac{1}{2^k}\right)(|+\rangle|x\rangle) = \frac{1}{\sqrt{2}}\left(|0\rangle|1\rangle + \exp\left\{\frac{2\mathrm{i}\pi}{2^k}\right\}|1\rangle|1\rangle\right).$$

The gate ${}_c\mathrm{R}_z(\alpha)$ applies to two qubits where the first one constitutes what is called an ancilla qubit since it controls the computation. From there one should define the ancilla register that is composed of all the qubits that are used as controlled qubits.

2.3.1 The case where with m ancilla qubits and $\mathbf{x}^\top\mathbf{w} \in \{0, \ldots, 2^m - 1\}$

The first part of the circuit consists of applying Hadamard gates on the ancilla register $|0\rangle^{\otimes m}$, which produces

$$\mathrm{H}^{\otimes m}|0\rangle^{\otimes m}|\mathbf{x}\rangle = \left(\frac{1}{\sqrt{2^m}}\sum_{j=0}^{2^m-1}|j\rangle\right)\otimes|\mathbf{x}\rangle. \tag{2.4}$$

The goal here is then to encode as a phase the result of the inner product $\mathbf{x}^\top\mathbf{w}$. With the binary approximation 2.3 for $|\mathbf{x}\rangle$ and m ancilla qubits, define for $l \in \{1, \ldots, m\}$, $j \in \{0, \ldots, n-1\}$ and $k \in \{1, \ldots, p\}$, ${}_{\mathrm{c}}\mathrm{R}_z^{l,j,k}(\alpha)$, the ${}_{\mathrm{c}}\mathrm{R}_z(\alpha)$ matrix applied to the qubit $|x_{j,k}\rangle$ with the lth qubit of the ancilla register as control. Finally, introduce the unitary operator

$$\mathrm{U}_{\mathbf{w},m} := \prod_{l=0}^{m-1}\left\{\prod_{j=0}^{n-1}\prod_{k=1}^{p}{}_{\mathrm{c}}\mathrm{R}_z^{m-l,j,k}\left(\frac{w_j}{2^{m+k}}\right)\right\}^{m-l}. \tag{2.5}$$

Proposition 2.2 *The following identity holds for all* $n, p, m \in \mathbb{N}$*:*

$$\mathrm{U}_{\mathbf{w},m}\mathrm{H}^{\otimes m}|0\rangle^{\otimes m}|\mathbf{x}\rangle = \left(\frac{1}{\sqrt{2^m}}\sum_{j=0}^{2^m-1}\exp\left\{2\mathrm{i}\pi j\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^m}\right\}|j\rangle\right)\otimes|\mathbf{x}\rangle, \tag{2.6}$$

where

$$\widetilde{\mathbf{x}}^\top\mathbf{w} := \sum_{j=0}^{n-1}w_j\sum_{k=1}^{p}\frac{x_{j,k}}{2^k}$$

is the p*-binary approximation of* $\mathbf{x}^\top\mathbf{w}$.

Proof We prove the proposition for $n = p = m = 2$ for simplicity and the general case is analogous. Therefore we consider $\mathrm{U}_{\mathbf{w},2} := \left\{\prod_{j=0}^{1}\prod_{k=1}^{2}{}_{\mathrm{c}}\mathrm{R}_z^{2,j,k}\left(\frac{w_j}{2^{2+k}}\right)\right\}^2 \prod_{j=0}^{1}\prod_{k=1}^{2}{}_{\mathrm{c}}\mathrm{R}_z^{1,j,k}\left(\frac{w_j}{2^{2+k}}\right)$. First, we have

$$\begin{aligned}&\prod_{j=0}^{1}\prod_{k=1}^{2}{}_{\mathrm{c}}\mathrm{R}_z^{1,j,k}\left(\frac{w_j}{2^{2+k}}\right)\otimes\left(\frac{1}{\sqrt{2^2}}\sum_{j=0}^{2^2-1}|j\rangle\right)\otimes|\mathbf{x}\rangle\\ &= \frac{1}{\sqrt{2^2}}(|0\rangle+|1\rangle)\left(|0\rangle+\exp\left\{2\mathrm{i}\pi\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^2}\right\}|1\rangle\right)\otimes|\mathbf{x}\rangle,\end{aligned}$$

result to which we apply $\left\{\prod_{j=0}^{1}\prod_{k=1}^{2}{}_{\mathrm{c}}\mathrm{R}_z^{2,j,k}\left(\frac{w_j}{2^{2+k}}\right)\right\}^2$ which yields

$$\frac{1}{\sqrt{2^2}}\left(|0\rangle+\exp\left\{2\mathrm{i}\pi 2\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^2}\right\}|1\rangle\right)\otimes\left(|0\rangle+\exp\left\{2\mathrm{i}\pi\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^2}\right\}|1\rangle\right)\otimes|\mathbf{x}\rangle,$$

achieving the proof of 2.6. □

From the definition of the Quantum Fourier transform in A.3, if $\widetilde{\mathbf{x}}^\top\mathbf{w} = k \in \{0, \ldots, 2^m - 1\}$, the resulting state is

$$\mathrm{U}_{\mathbf{w},m}\left((\mathrm{H}^{\otimes m}|00\rangle)\otimes|\mathbf{x}\rangle\right) = \left({}_{\mathrm{q}}\mathcal{F}|k\rangle\right)\otimes|\mathbf{x}\rangle = \left({}_{\mathrm{q}}\mathcal{F}|\widetilde{\mathbf{x}}^\top\mathbf{w}\rangle\right)\otimes|\mathbf{x}\rangle.$$

Thus only applying the Quantum Inverse Fourier Transform would be enough to retrieve $|\widetilde{\mathbf{x}}^\top\mathbf{w}\rangle$. The pseudo-code is detailed in Algorithm 1 and the quantum circuit in the case $n = p = m = 2$ is depicted in Fig. 2 (and detailed in Example 2.3).

Example 2.3 To understand the computations performed by the quantum gates, consider the case where $n = p = 2$. Therefore we only need 2×2 qubits to represent each element of the dataset which constitute the main register. Introduce an ancilla register composed of $m = 2$ qubits each initialised at $|0\rangle$, and suppose that the input state on the main register is $|\mathbf{x}\rangle$. The goal here is then to encode as a phase the result of the inner product $\mathbf{x}^\top\mathbf{w}$ where $\mathbf{w} = (w_0, w_1)^\top$.

Input: $\mathbf{w} \in [-1, 1]^n, \mathbf{x} \in [0, 1]^n$, $\varepsilon > 0$ the probability to mismeasure $\mathbf{x}^\top\mathbf{w}$, $\mathrm{U}_{\mathbf{w},m}$ unitary matrix, m the number of ancilla qubits, p the precision used for the binary fraction of each component of $\mathbf{x}$, together with the constraint $\mathbf{x}^\top\mathbf{w} \in \{0, \ldots, 2^m - 1\}$.

Procedure:

1. $|0\rangle^{\otimes m}|\mathbf{x}\rangle$ ▷ Initial state with $|0\rangle^{\otimes m}$ as ancilla register and $|\mathbf{x}\rangle$ as data made of $n \times p$ qubits
2. $|0\rangle^{\otimes m}|\mathbf{x}\rangle \mapsto \mathrm{H}^{\otimes m}|0\rangle^{\otimes m}|\mathbf{x}\rangle = \frac{1}{\sqrt{2^m}}\sum_{j=0}^{2^m-1}|j\rangle|\mathbf{x}\rangle$ ▷ Apply Hadamard gates to the m ancillas register
3. $\longrightarrow \frac{1}{\sqrt{2^m}}\sum_{j=0}^{2^m-1}\exp\left\{2\mathrm{i}\pi j\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^m}\right\}|j\rangle|\mathbf{x}\rangle$ ▷ Apply $\mathrm{U}_{\mathbf{w},m}$
4. $\longrightarrow |\widetilde{\mathbf{x}}^\top\mathbf{w}\rangle|\mathbf{x}\rangle$ ▷ Apply the inverse QFT where $|\widetilde{\mathbf{x}}^\top\mathbf{w}\rangle$ is a p-qubit approximation of $\widetilde{\mathbf{x}}^\top\mathbf{w}$
5. $\longrightarrow |\widetilde{\mathbf{x}}^\top\mathbf{w}\rangle$ ▷ Measure $\widetilde{\mathbf{x}}^\top\mathbf{w}$ with a probability at least $1 - \varepsilon$

return Output: $\widetilde{\mathbf{x}}^\top\mathbf{w}$

Algorithm 1 Quantum Inner Product (QIP) $(\mathbf{w}, \mathbf{x}, \mathbf{U}_{\mathbf{w},m}, m, p, \varepsilon)$

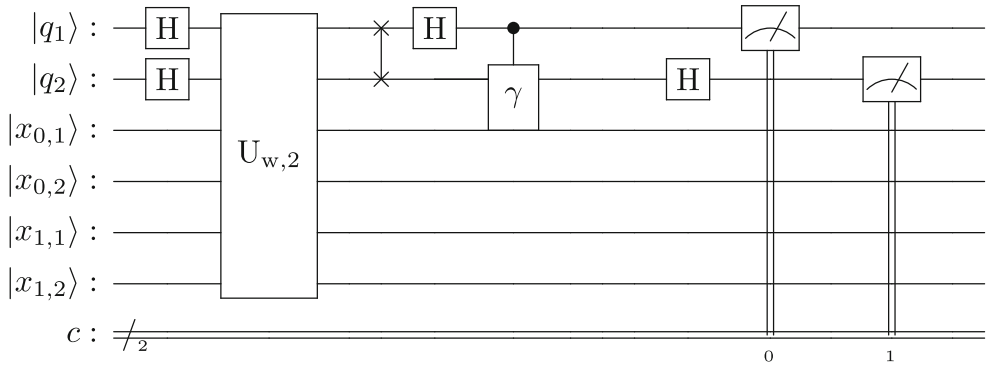


Fig. 2 QIP circuit for $m = 2$ ancilla qubits. The c line represents the classical register from which we retrieve the outcomes of the measurements. The controlled gate γ performs as $C(\gamma) : |q_1\rangle|q_2\rangle \mapsto 11_{|q_1\rangle=|1\rangle}(|q_1\rangle)|1\rangle \otimes e^{-i\frac{\pi}{4}}|q_2\rangle + 11_{|q_1\rangle=|0\rangle}(|q_1\rangle)|0\rangle \otimes |q_2\rangle$

So in this example the entire wave function combining both the main register's qubits and the ancilla register's qubits is encoded in six qubits. By denoting ${}_{c}R_z^{1,j,k}(\alpha)$ the ${}_{c}R_z(\alpha)$ matrix applied to the first qubit of the ancilla register and the qubit $|x^i_{j,k}\rangle$, and ${}_{c}R_z^{2,j,k}(\alpha)$ the ${}_{c}R_z(\alpha)$ matrix applied to the second qubit of the ancilla register and the qubit $|x_{j,k}\rangle$. Using the gates in 2.5, namely

$$U_{\mathbf{w},1} = \prod_{j=0}^{1}\prod_{k=1}^{2} {}_{c}R_z^{1,j,k}\left(\frac{w_j}{2^{1+k}}\right) \quad \text{and}$$

$$U_{\mathbf{w},2} = \left\{\prod_{j=0}^{1}\prod_{k=1}^{2} {}_{c}R_z^{2,j,k}\left(\frac{w_j}{2^{2+k}}\right)\right\}^2 \prod_{j=0}^{1}\prod_{k=1}^{2} {}_{c}R_z^{1,j,k}\left(\frac{w_j}{2^{2+k}}\right).$$

Remark 2.4 There is an interesting and potentially very useful difference here between the quantum and the classical versions of a feedforward neural network; in the former, the input $\mathbf{x}$ is not lost after running the circuit, while this information is lost in the classical setting. This in particular implies that it can be used again for free in the quantum setting.

2.3.2 The case $\mathbf{x}^\top\mathbf{w} \notin \{0, \ldots, 2^m - 1\}$

What happens if $\widetilde{\mathbf{x}}^\top\mathbf{w}$ is not a integer and $\widetilde{\mathbf{x}}^\top\mathbf{w} \geq 0$? Again, the short answer is that we are able to obtain a good approximation of $\widetilde{\mathbf{x}}^\top\mathbf{w}$, which is already an approximation of the true value of the inner product $\mathbf{x}^\top\mathbf{w}$. Indeed, with the gates constructed above, QIP performs exactly like QPE. Just a quick comparison between what is obtained at stage 3 of the QPE Algorithm (Algorithm 2) and the output obtained at the third stage of the QIP 2.6 would be enough to state that the QIP is just an application of the QPE procedure. Thus $\left\{\prod_{j=0}^{n-1}\prod_{k=1}^{p} {}_{c}R_z^{1,j,k}\left(\frac{w_j}{2^{m+k}}\right)\right\}$ is a unitary matrix such that $|1\rangle \otimes |\mathbf{x}\rangle$ is an eigenvector of eigenvalue $\exp\left\{2i\pi\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^m}\right\}$.

Let $\phi := \frac{1}{2^m}\widetilde{\mathbf{x}}^\top\mathbf{w}$; the QPE procedure (Appendix A) can only estimate $\phi \in [0, 1)$. Firstly $\phi \leq 0$ can happen and secondly $|\phi| \geq 1$ can also happen. Therefore such circumstances have to be addressed. One first step would be to have $\mathbf{w} \in [-1, 1]^n$, so that $|\widetilde{\mathbf{x}}^\top\mathbf{w}| \leq n$. Then one should have m (the number of ancillas) large enough so that

$$\left|\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2^m}\right| \leq 1, \tag{2.7}$$

which produces $m \geq \log_2(n)$. Having these constrains respected, one obtains $|\phi| \leq 1$, which is not enough since we should have $\phi \in [0, 1)$ instead. The main idea behind solving that is based on computing $\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}$ instead of $\widetilde{\mathbf{x}}^\top\mathbf{w}$ which means dividing by 2 all the parameters of the ${}_{c}R_z^{m,j,k}$ gates. Indeed with 2.7, we have $-2^m \leq \widetilde{\mathbf{x}}^\top\mathbf{w} \leq 2^m$, and thus $-2^{m-1} \leq \frac{1}{2}\widetilde{\mathbf{x}}^\top\mathbf{w} \leq 2^{m-1}$.

- In the case where $\widetilde{\mathbf{x}}^\top\mathbf{w} \geq 0$ we have $\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2} \in [0, 2^{m-1}]$ and then by defining $\widetilde{\phi}^+ := \frac{1}{2^m}\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}$ we then obtain $\widetilde{\phi}^+ \in [0, \frac{1}{2}]$, therefore the QPE can produce an approximation of $\widetilde{\phi}^+$ as put forward in Algorithm 2 which then can be multiplied by 2^{m+1} to retrieve $\widetilde{\mathbf{x}}^\top\mathbf{w}$.
- In the case where $\widetilde{\mathbf{x}}^\top\mathbf{w} \leq 0$, then $\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2} \in [-2^{m-1}, 0]$. As above, $|1\rangle \otimes |\mathbf{x}\rangle$ is an eigenvector of $\left\{\prod_{j=0}^{n-1}\prod_{k=1}^{p} {}_{c}R_z^{1,j,k}\left(\frac{\frac{w_j}{2}}{2^{m+k}}\right)\right\}$ with corresponding eigenvalue $\exp\left\{2i\pi\frac{\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}}{2^m}\right\} = \exp\left\{2i\pi\left[1 + \frac{\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}}{2^m}\right]\right\}$. Defining $\widetilde{\phi}^- := \frac{1}{2^m}\left(2^m + \frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}\right) = 1 + \frac{1}{2^m}\frac{\widetilde{\mathbf{x}}^\top\mathbf{w}}{2}$ we then obtain $\widetilde{\phi}^- \in [\frac{1}{2}, 1]$ which a QPE procedure can estimate and from which we can retrieve $\widetilde{\mathbf{x}}^\top\mathbf{w}$

For values of ϕ measured in $[0, \frac{1}{2}) \cup (\frac{1}{2}, 1)$ we are sure about the associated value of the inner product. This means that for a fixed $\mathbf{x}$, the map

$$f : \left[0, \frac{1}{2}\right) \cup \left(\frac{1}{2}, 1\right) \ni \phi \mapsto \widetilde{\mathbf{x}}^\top\mathbf{w} \in [-n, n]$$

is injective. A measurement output equal to *half* could mean either that $\widetilde{\mathbf{x}}^\top\mathbf{w} = 2^m$ or $\widetilde{\mathbf{x}}^\top\mathbf{w} = -2^m$, which could be prevented for $\mathbf{w} \in [-1, 1]^n$ and m large enough such that $n < 2^m$. Under these circumstances, f can be extended to an injective function on $[0, 1)$, with 1 being excluded since the QPE can only estimate values in $[0, 1)$.

2.4 Quantum activation function

We consider an activation function $\sigma : \mathbb{R} \to \mathbb{R}$. A classical example is the sigmoid $\sigma(x) := \left(1 + \mathrm{e}^{-x}\right)^{-1}$. The goal here is to build a circuit performing the transformation $|x\rangle \mapsto |\sigma(x)\rangle$ where $|x\rangle$ and $|\sigma(x)\rangle$ are the quantum encoded versions of their classical counterparts as in Section 2.2. Again, we shall appeal to the Quantum Phase Estimation algorithm. For a q-qubit state $|x\rangle = |x_1 \dots x_q\rangle \in \mathbb{C}^{2^q}$, we wish to build a matrix $\mathrm{U} \in \mathcal{M}_{2^q}(\mathbb{C})$ such that

$$\mathrm{U}|x\rangle = \mathrm{e}^{2\mathrm{i}\pi\sigma(x)}|x\rangle.$$

Considering

$$\mathrm{U} := \mathrm{Diag}\left(\mathrm{e}^{2\mathrm{i}\pi\sigma(0)}, \mathrm{e}^{2\mathrm{i}\pi\sigma(1)}, \mathrm{e}^{2\mathrm{i}\pi\sigma(2)}, \dots, \mathrm{e}^{2\mathrm{i}\pi\sigma(2^q-1)}\right),$$

then, for m ancilla qubits, the Quantum Phase estimation yields

$$\mathrm{QPE} : |0\rangle^{\otimes m} \otimes |x\rangle \mapsto |\widetilde{\sigma(x)}\rangle \otimes |x\rangle,$$

where again $\widetilde{\sigma(x)}$ is the m-bit binary fraction approximation for $\sigma(x)$ as detailed in Algorithm 2. In Fig. 3, we can see that the information flows from $|\mathbf{x}\rangle = |x_{0,1}x_{1,1}x_{2,1}x_{3,1}\rangle$ to the register attached to $|q_2\rangle$ to obtain the inner product and from the register $|q_2\rangle$ to $|q_1\rangle$ for the activation of the inner product. This explains why only measuring the register $|q_1\rangle$ is enough to retrieve $\sigma(\mathbf{x}w^\top\mathbf{w})$.

3 Quantum GAN architecture

A Generative Adversarial Network (GAN) is a network composed of two neural networks. In a classical setting, two agents, the generator and the discriminator, compete against each other in a zero-sum game (Kakutani 1941), playing in turns to improve their own strategy; the generator tries to fool the discriminator while the latter aims at correctly distinguishing real data (from a training database) from generated ones. As put forward in Goodfellow et al. (2014), the generative model can be thought of as an analogue to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminator plays the role of the police, trying to detect the counterfeit currency. Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles. Under reasonable assumptions (the strategy spaces of the agents are compact and convex) the game has a unique (Nash) equilibrium point, where the generator is able to reproduce exactly the target data distribution. Therefore, in a classical setting, the generator $\mathbf{G}$, parameterised by a vector of parameters $\boldsymbol{\theta}_G$, produces a random variable $X_{\boldsymbol{\theta}_G}$, which we can write as the map

$$\mathbf{G} : \boldsymbol{\theta}_G \to X_{\boldsymbol{\theta}_G}.$$

The goal of the discriminator $\mathbf{D}$, parameterised by $\boldsymbol{\theta}_D$, is to distinguish samples $\mathbf{x}_{\boldsymbol{\theta}_G}$ of $X_{\boldsymbol{\theta}_G}$ from $\mathbf{x}_{Real} \in \mathcal{D}$, where $\mathbf{x}_{Real}$ has been sampled from the underlying distribution $\mathbb{P}_{\mathcal{D}}$ of the database $\mathcal{D}$. The map $\mathbf{D}$ thus reads

$$\mathbf{D} : \mathbf{x}_{\boldsymbol{\theta}_G}, \boldsymbol{\theta}_D \mapsto \mathbb{P}_{\boldsymbol{\theta}_D}\left(\mathbf{x}_{\boldsymbol{\theta}_G} \text{ sampled from } \mathbb{P}_{\mathcal{D}}\right).$$

We aim here at mimicking this classical GAN architecture into quantum version. Not surprisingly, we first build a quantum discriminator, followed by a quantum generator, and we finally develop the quantum equivalent of the zero-sum game, defining an objective loss function acting on quantum states.

3.1 Quantum discriminator

In the case of a fully connected quantum GAN — which we study here — where both the discriminator and generator are quantum circuits, one of the main differences between a classical GAN and a QuGAN lays in the input of the discriminator. Indeed, as said above, in a classical

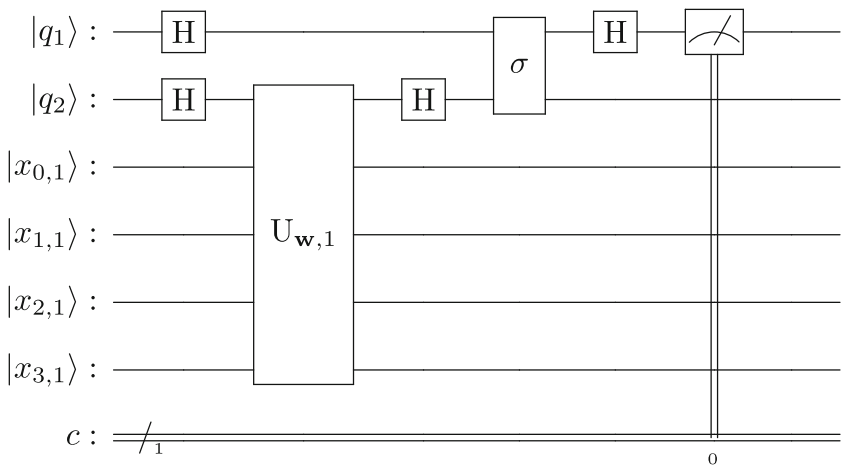


Fig. 3 Quantum single neuron for $|\mathbf{x}\rangle \in \mathbb{C}^{2^4}$, one ancilla qubit $|q_2\rangle$ for the QIP implemented via the controlled gate $\mathrm{U}_{\mathbf{w},1}$ for $w \in [-1, 1]^4$, and one ancilla qubit $|q_1\rangle$ for the activation function σ

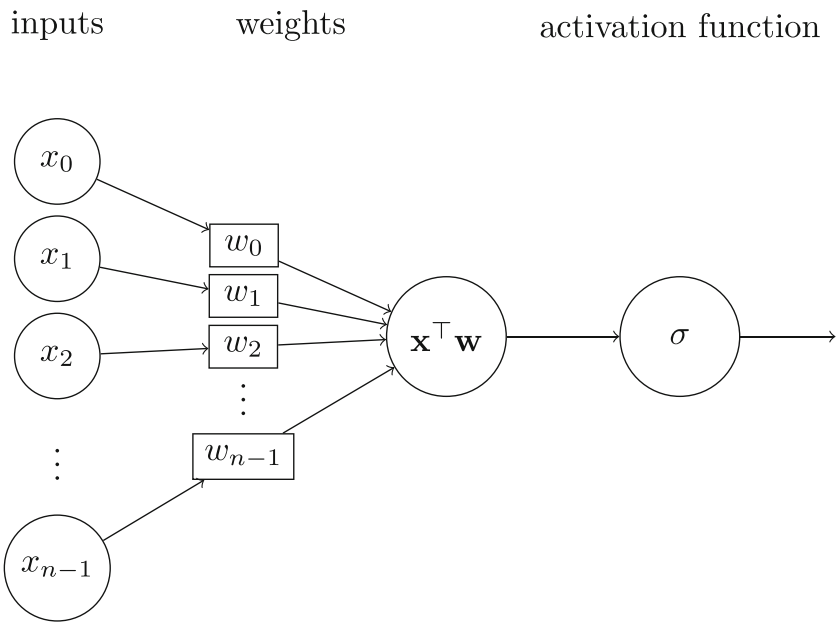


Fig. 4 Classical perceptron mapping $\mathbf{x} \in \mathbb{R}^n$ to $\sigma\left(\mathbf{x}^\top \mathbf{w}\right) \in \mathbb{R}$

discriminator the input is a sample $\mathbf{x}_{\boldsymbol{\theta}_G}$ generated by the generator $\mathbf{G}$, whereas in a quantum discriminator the input is a wave function

$$|v_{\boldsymbol{\theta}_G}\rangle = \sum_{j=0}^{2^n-1} v_{j,\boldsymbol{\theta}_G} |j\rangle \tag{3.1}$$

generated by a quantum generator. In such a setting, the goal is to create a wave function of the form 3.1 which is a physical way of encoding a given discrete distribution, namely

$$\mathbb{P}\left(|v_{\boldsymbol{\theta}_G}\rangle = |j\rangle\right) = |v_{j,\boldsymbol{\theta}_G}|^2 = p_j, \qquad \text{fo each } j = 0, \ldots, 2^n - 1, \tag{3.2}$$

where $(p_j)_{j=0,\ldots,2^n-1} \in [0,1]^{2^n}$ with $\sum_{j=0}^{2^n-1} p_j = 1$. We choose here a simple architecture for the discriminator, as a quantum version of a perceptron with a sigmoid activation function (Fig. 4).

This approach of building the circuit is new since in the papers that use quantum discriminators, the circuits that are used are what is called ansatz circuits (Braccia et al. 2021), in other words generic circuits built with layers of rotation gates and controlled rotation gates (see 3.6 and 3.7 below for the definition of these gates). Such ansatz circuits are therefore parameterised circuits as put forward in Chakrabarti et al. (2019), where generally an interpretation on the circuit's architecture performing as a classifying neural network cannot be made. As pointed out in Braccia et al. (2021), the architectures of both the generator and the discriminator are the same, which on the one hand solves the issue of having to monitor whether there is a imbalance in terms of expressivity between the generator and the discriminator; however, on the other hand,3 it prevents us from being able to give a straightforward interpretation for the given architectures.

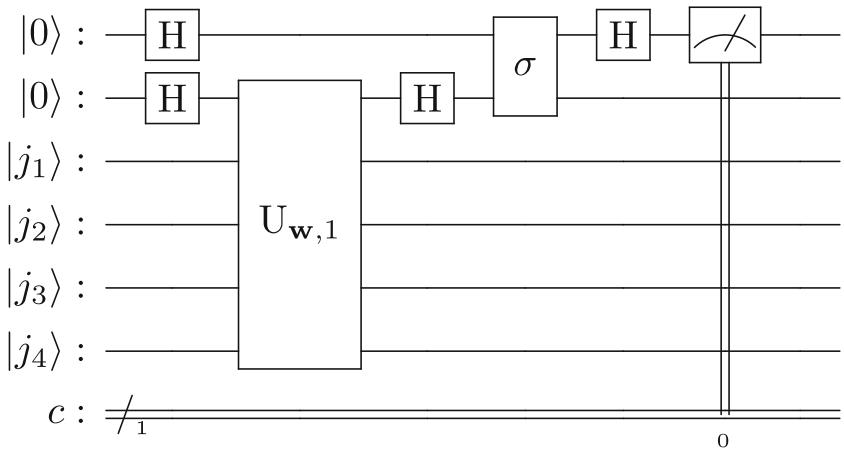


Fig. 5 Quantum perceptron with $\mathbf{w} \in \mathbb{R}^4$ and one ancilla qubit for the inner product ($m_2 = 1$) and one ancilla qubit for the activation ($m_1 = 1$). Here we only measure the result produced by the activation function

The main task here is then to translate these classical computations to a quantum input for the discriminator. This challenge has been taken up in both Sections 2.3 and 2.4 where we have built from scratch a quantum perceptron which performs exactly like a classical perceptron. There is however one main difference in terms of interpretation: let the wave function 3.1 be the input for the discriminator with $N = 2^n$ and, for $j = \overline{j_1 \cdots j_n}$ (defined in A.4), define $\phi_j := (j_1, \ldots, j_n)$. Denote $\mathfrak{D}(\mathbf{w}) \in \mathcal{M}_{2^{n+m_1+m_2}}(\mathbb{C})$ the transformation performed by the entire quantum circuit depicted in Fig. 5, where $\mathfrak{D}(\mathbf{w})$ is unitary and $\mathbf{w} \in \mathbb{R}^n$, namely for $m_1 + m_2$ ancilla qubits,

$$\mathfrak{D}(\mathbf{w})|0\rangle^{\otimes m_1+m_2}|j\rangle = |\sigma\left(\phi_j^\top \mathbf{w}\right)\rangle|\phi_j^\top \boldsymbol{w})\rangle|j\rangle,$$

where $|\sigma\left(\phi_j^\top \mathbf{w}\right)\rangle \in \mathbb{C}^{2^{m_1}}$ and $|\phi_j^\top \mathbf{w}\rangle \in \mathbb{C}^{2^{m_2}}$ and where we only measure $|\sigma\left(\phi_j^\top \mathbf{w}\right)\rangle$. Thus, for the input 3.1, the discriminator outputs the wave function (with $m_1 + m_2$ ancilla qubits)

$$\mathfrak{D}(\mathbf{w})|0\rangle^{\otimes m_1+m_2}|v_{\boldsymbol{\theta}_G}\rangle = \sum_{j=0}^{2^n-1} v_{j,\boldsymbol{\theta}_G}|\sigma\left(\phi_j^\top \mathbf{w}\right)\rangle|\phi_j^\top \mathbf{w})\rangle|j\rangle. \tag{3.3}$$

Therefore, in a QuGAN setting the goal for the discriminator is to distinguish the target wave function $|\psi_{\text{target}}\rangle$ from the generated one $|v_{\boldsymbol{\theta}_G}\rangle$. In Zoufal et al. (2019) where — for a distribution with 2^3 possible outcomes — the authors use a classical discriminator composed of a 512-node input layer, a 256-node hidden layer, and a single-node output layer; in contrast, our quantum discriminator has only $n = 3$. Therefore while achieving comparable results, our approach avoids an over-parameterisation of the discriminator. While this over-parameterisation may be useful (for example to reduce the error of the estimation made by sampling from the generator, as in Zoufal et al. (2019)), it is not always desirable as interpretability of the network may suffer (Molnar 2020). A precise characterisation of the optimal network (number of gates for example) is still an open question, as in classical machine learning, which we shall investigate in the future.

Example 3.1 As an example, consider $m_2 = 1$ ancilla qubit for the inner product, $m_1 = 1$ ancilla qubit for the activation, $|\psi_{\text{target}}\rangle = \psi_0|0\rangle + \psi_1|1\rangle$ and $|v_{\boldsymbol{\theta}_G}\rangle = v_{0,\boldsymbol{\theta}_G}|0\rangle + v_{1,\boldsymbol{\theta}_G}|1\rangle$. As we only measure the outcome produced by the activation function, the only possible outcomes are $|0\rangle$ and $|1\rangle$. Therefore, measuring the output of the discriminator only consists of a projection on either $|0\rangle$ or $|1\rangle$. Define these projectors

$$\Pi_0 := |0\rangle\langle 0| \otimes \mathrm{I_d}^{\otimes m_2+n} \in \mathcal{M}_{2^{m_1+n+m_2}}(\mathbb{C}) \qquad \text{and} \qquad \Pi_1 := |1\rangle\langle 1| \otimes \mathrm{I_d}^{\otimes m_2+n} \in \mathcal{M}_{2^{m_1+n+m_2}}(\mathbb{C}),$$

where $m_2 = 1$ and $n = 1$ since in our toy example the wave functions encoding the distributions are 1-qubit distributions. Interpreting measuring $|0\rangle$ as labelling the input distribution *Fake* and measuring $|1\rangle$ as labelling it *Real*, the optimal discriminator with parameter $\mathbf{w}^*$ would perform as

$$\begin{aligned}
\mathbb{P}\left(\mathfrak{D}(\mathbf{w}^*)|0\rangle^{\otimes m_1+m_2}|v_{\boldsymbol{\theta}_G}\rangle = |0\rangle \otimes \sum_{j=0}^{2^n-1} v_{j,\boldsymbol{\theta}_G}|\phi_j^\top \mathbf{w}^*\rangle|j\rangle\right) &= \left\|\Pi_0 \mathfrak{D}(\mathbf{w}^*)|0\rangle^{\otimes m_1+m_2}|v_{\boldsymbol{\theta}_G}\rangle\right\|^2 = 1, \\
\mathbb{P}\left(\mathfrak{D}(\mathbf{w}^*)|0\rangle^{\otimes m_1+m_2}|\psi_{\text{target}}\rangle = |1\rangle \otimes \sum_{j=0}^{2^n-1} \psi_j|\phi_j^\top \mathbf{w}^*\rangle|j\rangle\right) &= \left\|\Pi_1 \mathfrak{D}(\mathbf{w}^*)|0\rangle^{\otimes m_1+m_2}|\psi_{\text{target}}\rangle\right\|^2 = 1,
\end{aligned} \tag{3.4}$$

where still in our toy example we have $n = 1$, $m_1 = 1$ and $m_2 = 1$. Here n could be any positive integer. We illustrate the circuit in Fig. 5.

3.1.1 Bloch sphere representation

The Bloch sphere (Nielsen and Chuang 2000) is important in Quantum Computing, providing a geometrical representation of pure states. In our case, it yields a geometric visualisation of the way an optimal quantum discriminator works as it separates the two complementary regions

$$\begin{aligned}
\mathcal{R}_F &:= \left\{ \sum_{i=0}^{2^{m-1}-1} \alpha_i|i\rangle \text{ such that } \sum_{i=0}^{2^{m-1}-1} |\alpha_i|^2 = 1 \right\}, \\
\mathcal{R}_T &:= \left\{ \sum_{i=2^{m-1}}^{2^m-1} \alpha_i|i\rangle \text{ such that } \sum_{i=2^{m-1}}^{2^m-1} |\alpha_i|^2 = 1 \right\},
\end{aligned} \tag{3.5}$$

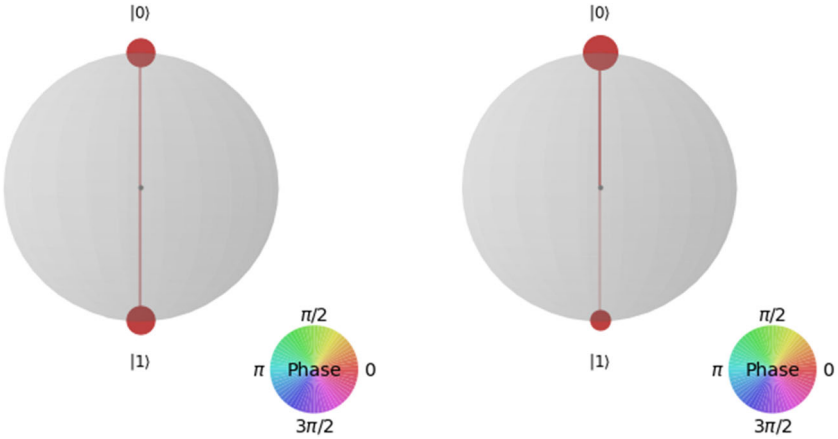


Fig. 6 Bloch spheres representations for $|\psi_{\text{target}}\rangle$ (left) and $|v_{\theta_G}\rangle$ (right) where there is no phase shift between $|0\rangle$ and $|1\rangle$ and where the sizes of the lobes are proportional to the probability of measuring the associated states

where $m := m_1 + m_2 + n$ is the total number of qubits for the inputs of the discriminator. The optimal discriminator $\mathfrak{D}(\mathbf{w}^*)$ would perform as

$$\mathfrak{D}(\mathbf{w}^*)|Fake\rangle \in \mathcal{R}_F \quad \text{and} \quad \mathfrak{D}(\mathbf{w}^*)|Real\rangle \in \mathcal{R}_T, \quad \text{almost surely,}$$

where $|Fake\rangle := |0\rangle|0\rangle|v_{\theta_G}\rangle$ and $|Real\rangle := |0\rangle|0\rangle|\psi_{\text{target}}\rangle$. Now, the challenge lays in finding such an optimal discriminator; however, one should note that the nature of the state $|Fake\rangle$ plays a major role in finding such a discriminator. Therefore, in the following part we focus on the generator responsible for generating $|Fake\rangle$.

Example 3.2 Consider Example 3.1 with $(\psi_0, \psi_1) = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ and $(v_{0,\theta_G}, v_{1,\theta_G}) = (\frac{\sqrt{3}}{2}, \frac{1}{2})$. The states $|\psi_{\text{target}}\rangle$ and $|v_{\theta_G}\rangle$ are shown in Fig. 6. The wave function produced by the discriminator is composed of three qubits ($m_1 = 1$, $m_2 = 1$ and $n = 1$ qubit for the input wave function 3.3); therefore, one optimal transformation for the discriminator having $|\psi_{\text{target}}\rangle$ as an input is one such that the first qubit never collapses onto the state $|0\rangle$ (Fig. 7).

3.2 Quantum generator

The quantum generator is a quantum circuit producing a wave function that encodes a discrete distribution. Such a circuit takes as an input the ground state $|0\rangle^{\otimes n-m_1-m_2}$ and outputs a wave function $|v_{\theta_G}\rangle$ parameterised by $\boldsymbol{\theta}_G$, the set of parameters for the discriminator. We recall here a few quantum gates that will be key to constructing a quantum generator. Recall that a quantum gate can be viewed as a unitary matrix; of particular interest will be gates acting on two (or more) qubits, as its allows quantum entanglement, thus fully leveraging the power of quantum computing. The NOT gate X acts on one qubit and is represented as

$$\mathrm{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

so that $\mathrm{X}|0\rangle = |1\rangle$ and $\mathrm{X}|1\rangle = |0\rangle$. The $\mathrm{R_Y}$ is a one-qubit gate represented by the matrix

$$\mathrm{R_Y}(\theta) := \begin{pmatrix} \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right) \end{pmatrix}, \tag{3.6}$$

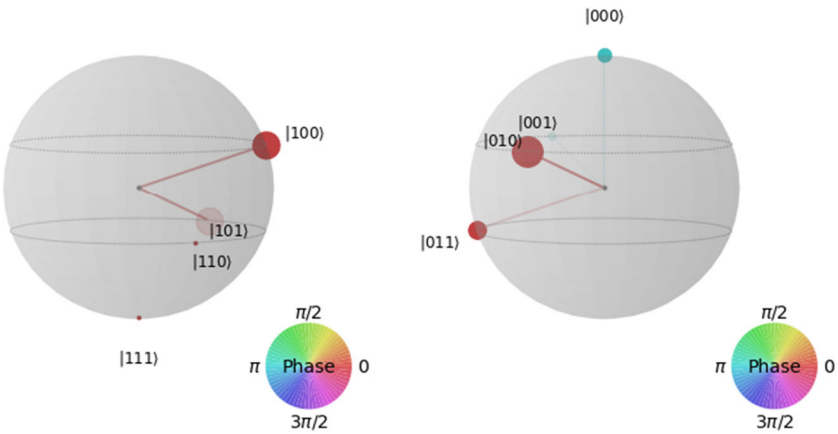


Fig. 7 Left: $\mathfrak{D}(w_1^*)|0\rangle|0\rangle|\psi_{\text{target}}\rangle$. Total system post-one optimal discriminator transformation. The first qubit never collapses onto $|0\rangle$ and therefore such a discriminator is optimal at labelling $|\psi_{\text{target}}\rangle$ as *Real*. Right: $\mathfrak{D}(w_2^*)|0\rangle|0\rangle|v_{\theta_G}\rangle$. Total system post-one optimal discriminator transformation. The first qubit never collapses onto $|1\rangle$ and therefore such a discriminator is optimal at labelling $|v_{\theta_G}\rangle$ *Fake*

thus performing as

$$\mathrm{R_Y}(\theta)|0\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)|1\rangle \qquad \text{and} \qquad \mathrm{R_Y}(\theta)|1\rangle = \cos\left(\frac{\theta}{2}\right)|1\rangle - \sin\left(\frac{\theta}{2}\right)|0\rangle.$$

The ${}_{\mathrm{c}}\mathrm{R_Y}$ Gate is the controlled version of the $\mathrm{R_Y}$ gate, acting on two qubits, one control qubit and one transformed qubit, producing quantum entanglement. The $\mathrm{R_Y}$ transformation applies on the second qubit only when provided the control qubit is in $|1\rangle$, otherwise leaves the second qubit unaltered. Its matrix representation is

$${}_{\mathrm{c}}\mathrm{R_Y}(\theta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\left(\frac{\theta}{2}\right) & -\sin\left(\frac{\theta}{2}\right) \\ 0 & 0 & \sin\left(\frac{\theta}{2}\right) & \cos\left(\frac{\theta}{2}\right). \end{pmatrix} \tag{3.7}$$

Given n qubits let $\mathbf{X} := (X_1 \dots X_n)$ be a random vector taking values in $\mathcal{X}_n := \{0,1\}^n$. Set

$$p_{\mathbf{x}} := \mathbb{P}[\mathbf{X} = \mathbf{x}], \quad \text{for } \mathbf{x} \in \mathcal{X}_n.$$

When building the generator we are looking for a quantum circuit that implements the transformation

$$|0\rangle^{\otimes n} \mapsto \sum_{\mathbf{x}\in\{0,1\}^n} \sqrt{p_{\mathbf{x}}}\mathrm{e}^{\mathrm{i}\theta_{\mathbf{x}}}|\mathbf{x}\rangle.$$

We could follow a classical algorithm. For $1 \le k \le n$, let $\mathbf{x}_{:k} := (x_1, \dots, x_k)$ and, given $\mathbf{x} \in \mathcal{X}_n$,

$$q_{\mathbf{x}_{:k}} := \begin{cases} \mathbb{P}[X_1 = 0], & \text{if } k = 1, \\ \mathbb{P}[X_k = 0 | \mathbf{X}_{:k-1} = \mathbf{x}_{:k-1}], & \text{if } 2 \le k \le n. \end{cases} \tag{3.8}$$

We then proceed by induction: start with a random draw of $\mathbf{X}_1$ as a Bernoulli sample with failure probability $q_{\mathbf{x}_1}$. Assuming that $\mathbf{X}_{:k-1}$ has been sampled as $\mathbf{x}_{:k-1}$ for some $1 \le k \le n$, sample $\mathbf{X}_k$ from a Bernoulli distribution with failure probability $q_{\mathbf{x}_{:k-1}}$. The quantum circuit will equivalently consist of n stages, where at each stage $1 \le k \le n$ we only work with the first k qubits, and at the end of each stage there is the correct distribution for the first k qubits in the sense that, upon measuring, their distribution coincides with that of $\mathbf{X}_{:k}$.

The first step is simple: a single Y-rotation of the first qubit with angle $\theta \in [0, \pi]$ satisfying $\cos(\frac{\theta}{2}) = \sqrt{q_{\mathbf{x}_1}}$. In other words, with $\mathrm{U}_1 := \mathrm{R_Y}(\theta)$, we map $|0\rangle$ to $\mathrm{U}_1|0\rangle = \sqrt{q_{\mathbf{x}_1}}|0\rangle + \sqrt{1-q_{\mathbf{x}_1}}|1\rangle$. Clearly, when measuring the first qubit, we obtain the correct law. Now, inductively, for $2 \le k \le n$, suppose the first $k-1$ qubits fixed, namely in the state

$$\sum_{\mathbf{x}_{:k-1}\in\mathcal{X}_{k-1}} \sqrt{p_{\mathbf{x}_{:k-1}}}|\mathbf{x}_{:k-1}\rangle|0\rangle^{\otimes n-k+1},$$

For each $\mathbf{x}_{:k-1} \in \mathcal{X}_{k-1}$, let $\theta_{\mathbf{x}_{:k-1}} \in [0;\pi]$ satisfy $\cos\left(\frac{1}{2}\theta_{\mathbf{x}_{:k-1}}\right) = \sqrt{q_{\mathbf{x}_{:k-1}}}$ and consider the gate $\mathrm{C}_{\mathbf{x}_{:k-1}}$ acting on the first k qubits which is a $\mathrm{R_Y}(\theta_x)$ on the last qubit k, controlled on whether the first $k-1$ qubits are equal to $\mathbf{x}_{:k-1}$. We then have

$$C_{\mathbf{x}_{:k-1}}|\mathbf{y}\rangle|0\rangle = \begin{cases} \sqrt{q_{\mathbf{x}_{:k-1}}}|\mathbf{x}_{:k-1}\rangle|0\rangle + \sqrt{1-q_{\mathbf{x}_{:k-1}}}|\mathbf{x}_{:k-1}\rangle|1\rangle, & \text{if } \mathbf{y} = \mathbf{x}_{:k-1}, \\ |\mathbf{y}\rangle|0\rangle, \quad \text{for } \mathbf{y} \ne \mathbf{x}_{:k-1}. \end{cases} \tag{3.9}$$

Therefore, defining $\mathrm{U}_k := \prod_{\mathbf{x}_{:k-1}\in\mathcal{X}_{k-1}} \mathrm{C}_{\mathbf{x}_{:k-1}}$, and noting that the order of multiplication does not affect the computations below, it follows that

$$\mathrm{U}_k \sum_{\mathbf{x}_{:k-1}\in\mathcal{X}_{k-1}} \sqrt{p_{\mathbf{x}_{:k-1}}}|\mathbf{x}_{:k-1}\rangle|0\rangle^{\otimes n-k+1} = \sum_{\mathbf{x}_{:k-1}\in\mathcal{X}_{k-1}} \left\{\sqrt{p_{\mathbf{x}_{:k-1}}q_{\mathbf{x}_{:k-1}}}|\mathbf{x}_{:k-1}\rangle + \sqrt{p_{\mathbf{x}_{:k-1}}\left(1-q_{\mathbf{x}_{:k-1}}\right)}|1\rangle\right\}|0\rangle$$
$$\sum_{\mathbf{x}_{:k}\in\mathcal{X}_k} \sqrt{p_{\mathbf{x}_{:k}}}|\mathbf{x}_{:k}\rangle|0\rangle^{\otimes n-k},$$

where the last equality follows from properties of conditional expectations since

$$p_{\mathbf{x}_{:k-1}}q_{\mathbf{x}_{:k-1}} = p_{\mathbf{x}_{:k-1}.0} \qquad \text{and} \qquad p_{\mathbf{x}_{:k-1}}\left(1-q_{\mathbf{x}_{:k-1}}\right) = p_{\mathbf{x}_{:k-1}.1},$$

for $\mathbf{x}_{:k-1} \in \mathcal{X}_{k-1}$, $\mathbf{x}_{:k-1}.0 \in \mathcal{X}_k$ and $\mathbf{x}_{:k-1}.1 \in \mathcal{X}_k$ (see after A.4 for the binary representation of decimals). This concludes the inductive step. The generator has therefore been built accordingly to a 'classical' algorithm, however only up until $\mathcal{X}_2$ (see Fig. 8 for the architecture for qubits q_3 and q_2) to avoid to have a network that is too deep and therefore untrainable in a differentiable manner because of the barren plateau phenomenon (McClean et al. 2018). Indeed, in order to build U_k from simple controlled

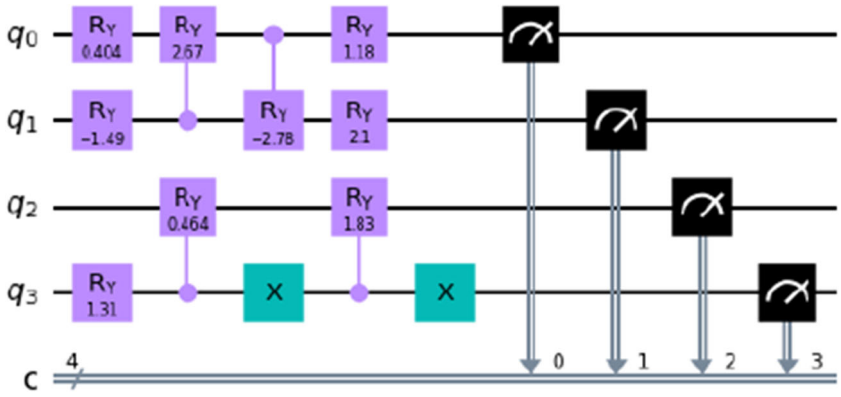


Fig. 8 Entangled generator composed of R_Y, ${}_cR_Y$ and X gates, with parameters values for $\{\theta_1, \ldots, \theta_9\}$ indicated alongside the gates

gates (with only one control qubit) the number of gates is of order $\mathcal{O}(2^{k-1})$, making the generator deeper. Thus the number of gates we would have to use would be of order $\mathcal{O}(2^n)$, making the generator very expressive yet very hard to train.

Example 3.3 With $n = 4$, the architecture for our generator is depicted in Fig. 8 and the full QuGAN (generator and discriminator) algorithm in Fig. 9.

3.3 Quantum adversarial game

In GANs the goal of the discriminator (D) is to discriminate real (R) data from the fake ones generated by the generator (G), while the goal of the latter is to fool the discriminator by generating fake data. Here both real and generated data are modeled as quantum states, respectively described by their wave functions $|\psi_{\text{target}}\rangle$ and $|v_{\boldsymbol{\theta}_G}\rangle$. Define the objective function

$$\mathcal{S}(\boldsymbol{\theta}_G, \mathbf{w}_D) := \mathbb{P}\Big(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle \in \mathcal{R}_T\Big) - \mathbb{P}\Big(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v_{\boldsymbol{\theta}_G}\rangle \in \mathcal{R}_T\Big),$$

where the region $\mathcal{R}$ is defined in 3.5. Here $\mathbb{P}(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle \in \mathcal{R}_T)$ is the probability of labelling the real data $|0\rangle|0\rangle|\psi_{\text{target}}\rangle$ as real via the discriminator and $\mathbb{P}(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v_{\boldsymbol{\theta}_G}\rangle \in \mathcal{R}_T)$ is the probability of having the generator fool the discriminator. As stated in 3.4 for two ancilla qubits ($m_1 + m_2 = 2$, i.e. one qubit for inner product and one qubit for activation) we have

$$\mathbb{P}\Big(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle \in \mathcal{R}_T\Big) = \big\|\Pi_1\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle\big\|^2.$$

By defining the projection of the output of the discriminator onto $\mathcal{R}_T$,

$$|\psi_{\text{out,target},\mathbf{w}_D}\rangle := \Pi_1\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle,$$

we can also write

$$\mathbb{P}\Big(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi_{\text{target}}\rangle \in \mathcal{R}_T\Big) = \text{Tr}(\rho_{\text{out,target},\mathbf{w}_D}),$$

where $\rho_{\text{out,target},\mathbf{w}_D} := |\psi_{\text{out,target},\mathbf{w}_D}\rangle\langle\psi_{\text{out,target},\mathbf{w}_D}|$ is the density operator associated to $\psi_{\text{out,target},\mathbf{w}_D}$. The same goes for the probability of fooling the discriminator, namely

$$\mathbb{P}\Big(\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v_{\boldsymbol{\theta}_G}\rangle \in \mathcal{R}_T\Big) = \big\|\Pi_1\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v_{\boldsymbol{\theta}_G}\rangle\big\|^2 = \text{Tr}(\rho_{\text{out},\boldsymbol{\theta}_G,\mathbf{w}_D}),$$

where $|\psi_{\text{out},\boldsymbol{\theta}_G,\mathbf{w}_D}\rangle := \Pi_1\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v_{\boldsymbol{\theta}_G}\rangle$ and $\rho_{\text{out},\boldsymbol{\theta}_G,\mathbf{w}_D} := |\psi_{\text{out},\boldsymbol{\theta}_G,\mathbf{w}_D}\rangle\langle\psi_{\text{out},\boldsymbol{\theta}_G,\mathbf{w}_D}|$. The min-max game played by the Generative Adversarial network is therefore defined as the optimisation problem

$$\min_{\boldsymbol{\theta}_G}\max_{\mathbf{w}_D} \mathcal{S}(\boldsymbol{\theta}_G, \mathbf{w}_D). \tag{3.10}$$

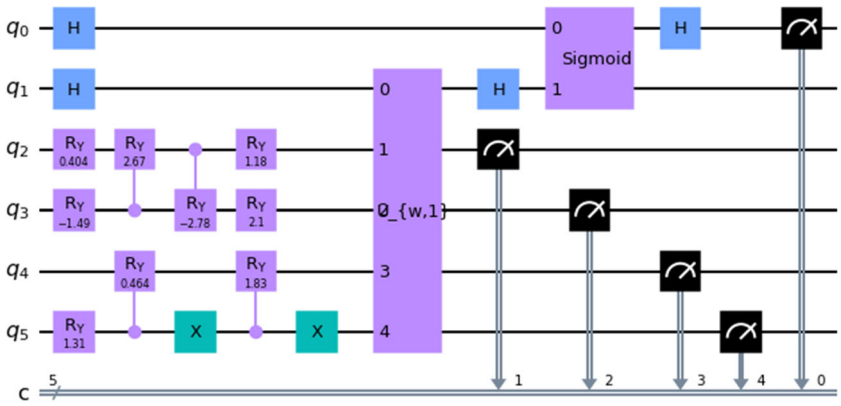


Fig. 9 The entire associated entangled QuGAN

Moreover, since $\mathcal{S}$ is differentiable and given the architecture of our circuits, according to the shift rule formula (Schuld et al. 2019), the partial derivatives of $\mathcal{S}$ admit the closed-form representations

$$\begin{aligned}\nabla_{\boldsymbol{\theta}_G}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) &= \frac{1}{2}\left\{\mathcal{S}\left(\boldsymbol{\theta}_G+\frac{\pi}{2},\mathbf{w}_D\right)-\mathcal{S}\left(\boldsymbol{\theta}_G-\frac{\pi}{2},\mathbf{w}_D\right)\right\},\\ \nabla_{\mathbf{w}_D}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) &= \frac{1}{2}\left\{\mathcal{S}\left(\boldsymbol{\theta}_G,\mathbf{w}_D+\frac{\pi}{2}\right)-\mathcal{S}\left(\boldsymbol{\theta}_G,\mathbf{w}_D-\frac{\pi}{2}\right)\right\}, \quad (3.11)\end{aligned}$$

so that training will be based on stochastic gradient ascent and descent. The reason for a stochastic algorithm lies in the nature of $\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D)$, seen as the difference between two probabilities to estimate. A natural estimator for l measurements/observations is

$$\widehat{\mathcal{S}}(\boldsymbol{\theta}_G,\mathbf{w}_D)_l := \frac{1}{l}\sum_{k=1}^{l} 11_{\left\{\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|\psi^k_{\text{target}}\rangle\in\mathcal{R}_T\right\}} - 11_{\left\{\mathfrak{D}(\mathbf{w}_D)|0\rangle|0\rangle|v^k_{\boldsymbol{\theta}_G}\rangle\in\mathcal{R}_T\right\}},$$

where $|v^k_{\boldsymbol{\theta}_G}\rangle$ is the kth wave function produced by the generator and $|\psi^k_{\text{target}}\rangle$ is the kth copy for the target distribution.

Given the nature of the problem, two strategies arise: for fixed parameters $\boldsymbol{\theta}_G$, when training the discriminator, we first minimise the labelling error, ie.

$$\max_{\mathbf{w}_D}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D),$$

which we achieve by stochastic gradient ascent with a learning rate $\eta_D = 0.9$. Moreover, we chose to initialise the weights following a Uniform distribution as $\mathbf{w}_D \sim \mathcal{U}([-1,1])$. Then, when training the generator the goal is to fool the discriminator, so that, for fixed $\mathbf{w}_D$, the target is

$$\min_{\boldsymbol{\theta}_G}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D),$$

which is achieved by stochastic gradient descent with a learning rate $\eta_G = 0.05$. Similarly to the discriminator, we initialise the weights as $\boldsymbol{\theta}_G \sim \mathcal{U}([0,2\pi])$. Our experiments seem to indicate that other initialisation assumptions overall yield analogous results. This choice of learning rates may look arbitrary at first sight. Unfortunately, there is yet no rigorous approach to finding optimal learning rates, even in the classical machine learning / stochastic gradient literature. One could also use tools from annealing, i.e. start with large values of learning rates and slowly decrease them, to go from exploration to exploitation, but we leave this to future investigations.

Remark 3.4 In the classical GAN setting, this optimisation problem may fail to converge (Goodfellow 2014). Over the past few years, progress has been made to improve the convergence quality of the algorithm and to improve its stability, using different loss functions or adding regularising terms. We refer the interested reader to the corresponding papers (Arjovsky et al. 2017; Denton et al. 2015; Deshpande et al. 2018; Gulrajani et al. 2017; Miyato et al. 2018; Radford et al. 2016; Salimans et al. 2016), and leave it to future research to integrate these improvements into a quantum setting.

Proposition 3.5 *The solution* $(\boldsymbol{\theta}^*_G,\mathbf{w}^*_D)$ *to the* $\min-\max$ *problem 3.10 is such that the wave function* $|v_{\boldsymbol{\theta}^*_G}\rangle$ *satisfies* $|\langle\psi_{\text{target}}||v_{\boldsymbol{\theta}^*_G}\rangle|^2 = 1$, *namely, for each* $i\in\{0,\dots,2^n-1\}$, $\mathbb{P}(|\psi_{\text{target}}\rangle)=|i\rangle) = \mathbb{P}(|v_{\boldsymbol{\theta}^*_G}\rangle=|i\rangle)$.

Proof Define the density matrices $\rho_{\text{target}} := |\psi_{\text{target}}\rangle\langle\psi_{\text{target}}|$ and $\rho_{\boldsymbol{\theta}_G} := |v_{\boldsymbol{\theta}_G}\rangle\langle v_{\boldsymbol{\theta}_G}|$ as well as the operator $P^R_{\mathbf{w}_D} := \mathfrak{D}(\mathbf{w}_D)^\dagger\Pi_1^\dagger\Pi_1\mathfrak{D}(\mathbf{w}_D)$. Then

$$\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) = \text{Tr}\left(P^R_{\mathbf{w}_D}\{\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\}\right)$$

Since $\Pi_1+\Pi_0 = \text{I}_\text{d}$ and $\mathfrak{D}(\mathbf{w}_D)$ is unitary, setting $P^F_{\mathbf{w}_D} := \mathfrak{D}(\mathbf{w}_D)^\dagger\Pi_0^\dagger\Pi_0\mathfrak{D}(\mathbf{w}_D)$, it is straightforward to rewrite $\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D)$ as

$$\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) = \text{Tr}\left(P^R_{\mathbf{w}_D}\rho_{\text{target}}\right)+\text{Tr}(P^F_{\mathbf{w}_D}\rho_{\boldsymbol{\theta}_G}\Big)-1,$$

since $\text{Tr}(\rho_{\boldsymbol{\theta}_G}) = 1$ according to the Born Rule (Theorem A.1) and $P^R_{\mathbf{w}_D}+P^F_{\mathbf{w}_D}=\text{I}_\text{d}$. Again, we also have

$$\begin{aligned}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) =& -1+\frac{1}{2}\text{Tr}\left(\left(P^R_{\mathbf{w}_D}+P^F_{\mathbf{w}_D}\right)\left(\rho_{\text{target}}+\rho_{\boldsymbol{\theta}_G}\right)\right)\\ &+\frac{1}{2}\text{Tr}\left(\left(P^R_{\mathbf{w}_D}-P^F_{\mathbf{w}_D}\right)\left(\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\right)\right),\end{aligned}$$

and finally

$$\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D) = \frac{1}{2}\text{Tr}\left(\left(P^R_{\mathbf{w}_D}-P^F_{\mathbf{w}_D}\right)\left(\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\right)\right).$$

Recall that for two Hermitian matrices A, B, the inequality $\text{Tr}(AB)\le\|A\|_p\|B\|_q$ holds for $p,q\ge1$ with $\frac{1}{p}+\frac{1}{q}=1$, where $\|\cdot\|_p$ denotes the p-norm. Since $P^R_{\mathbf{w}_D}$ and $P^F_{\mathbf{w}_D}$ are Hermitian, we obtain (with $p=\infty$ and $q=1$)

$$\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D)\le\frac{1}{2}\left\|P^R_{\mathbf{w}_D}-P^F_{\mathbf{w}_D}\right\|_\infty\left\|\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\right\|_1,$$

where $\left\|P^R_{\mathbf{w}_D}-P^F_{\mathbf{w}_D}\right\|_\infty\le1$. Thus the optimal $\mathbf{w}^*_D$ satisfies

$$\max_{\mathbf{w}_D}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D)=\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}^*_D)=\frac{1}{2}\left\|\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\right\|_1.$$

Again, since $\|\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\|_1\ge0$ the optimal $\boldsymbol{\theta}^*_G$ gives

$$\min_{\boldsymbol{\theta}_G}\max_{\mathbf{w}_D}\mathcal{S}(\boldsymbol{\theta}_G,\mathbf{w}_D)=\mathcal{S}(\boldsymbol{\theta}^*_G,\mathbf{w}^*_D)=0,$$

which is equivalent to $\|\rho_{\text{target}}-\rho_{\boldsymbol{\theta}_G}\|_1=0$, itself also equivalent to $\mathbb{P}(|v_{\boldsymbol{\theta}^*_G}\rangle=|i\rangle)=\mathbb{P}(|\psi_{\text{target}}\rangle=|i\rangle)=p_i$, for all $i\in\{0,\dots,2^n-1\}$. □

Remark 3.6 Our strategy to reach and approximate a solution to the $\min-\max$ problem will be as follows: we train the discriminator by stochastic gradient ascent n_D times and then train the generator n_G times by stochastic gradient descent and repeat this $\mathfrak{e}$ times.

4 Financial application: SVI goes quantum

We provide here a simple example of generating data in a financial context with the aim to increase interdisciplinarity between quantitative finance and quantum computing.

4.1 Financial background and motivation

Some of the most standard and liquid traded financial derivatives are so-called European Call and Put options. A Call (resp. Put) gives its holder the right, but not the obligation, to buy (resp. sell) an asset at a specified price (the strike price K) at a given future time (the maturity T). Mathematically, the setup is that of a filtered probability space $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t\geq 0}, \mathbb{P})$ where $(\mathcal{F}_t)_{t\geq 0}$ represents the flow of information; on this space, an asset $S = (S_t)_{t\geq 0}$ is traded and assumed to be adapted (namely S_t is $\mathcal{F}_t$-measurable for each $t \geq 0$). We further assume that there exists a probability $\mathbb{Q}$, equivalent to $\mathbb{P}$ such that S is a $\mathbb{Q}$-martingale. This martingale assumption is key as the Fundamental Theorem of Asset Pricing (Delbaen and Schachermayer 1994) in particular implies that this is equivalent to Call and Put prices being respectively equal, at inception of the contract, to

$$\mathrm{C}(K,T) = \mathbb{E}[\max(S_T - K, 0)|\mathcal{F}_0] \qquad \text{and} \qquad \mathrm{P}(K,T) = \mathbb{E}[\max(K - S_T, 0)|\mathcal{F}_0],$$

where the expectation $\mathbb{E}$ is taken under the risk-neutral probability $\mathbb{Q}$. Under sufficient smoothness property of the law of S_T, differentiating twice the Call price yields that the probability density function of the log stock price $\log(S_T)$ is given by

$$p_T(k) = \left(\frac{\partial^2 \mathrm{C}(K,T)}{\partial K^2}\right)_{K=S_0 e^k}, \tag{4.1}$$

implying that the real distribution of the (log) stock price can in principle be recovered from options data. However, prices are not quoted smoothly in (K, T) and interpolation and extrapolation are needed. Doing so at the level or prices turns out to be rather cumbersome and market practice usually does it at the level of the so-called implied volatility. The basic fundamental model of a continuous-time financial martingale is given by the Black-Scholes model (Black and Scholes 1973), under which

$$\frac{\mathrm{d}S_t}{S_t} = \sigma \mathrm{d}W_t, \qquad S_0 > 0,$$

where $\sigma > 0$ is the (constant) instantaneous volatility and W a standard Brownian motion adapted to the filtration $(\mathcal{F}_t)_{t\geq 0}$. In this model, Call prices admit the closed-form formula

$$\mathrm{C}_{\mathrm{BS}}(K,T,\sigma) := \mathbb{E}[\max(S_T - K, 0)|\mathcal{F}_0] = S_0 \mathrm{BS}\left(\log\left(\frac{K}{S_0}\right), \sigma^2 T\right),$$

where

$$\mathrm{BS}(k,v) := \begin{cases} \mathcal{N}(d_+(k,v)) - e^k \mathcal{N}(d_-(k,v)), & \text{if } v > 0, \\ (1 - e^k)_+, & \text{if } v = 0, \end{cases}$$

with $d_\pm(k,v) := -\frac{k}{\sqrt{v}} \pm \frac{\sqrt{v}}{2}$, where $\mathcal{N}$ denotes the cumulative distribution function of the Gaussian distribution. With a slight abuse of notation, we shall from now on write $C_{\mathrm{BS}}(K,T,\sigma) = C_{\mathrm{BS}}(k,T,\sigma)$, where $k := \log(\frac{K}{S_0})$ represents the logmoneyness.

Definition 4.1 Given a strike $K \geq 0$, a maturity $T \geq 0$ and a Call price $C(K,T)$ (either quoted on the market orcomputed from a model), the implied volatility $\sigma_{\mathrm{imp}}(k,T)$ is defined as the unique non-negative solution to the equation

$$\mathrm{C}_{\mathrm{BS}}(k, T, \sigma_{\mathrm{imp}}(k,T)) = \mathrm{C}(K,T). \tag{4.2}$$

Note that this equation may not always admit a solution. However, under no-arbitrage assumptions (equivalently under bound constraints for $C(K,T)$), it does so. We refer the interested reader to the volatility bible (Gatheral 2006) for full explanations of these subtle details. It turns out that the implied volatility is a much nicer object to work with (both practically and academically); plugging this definition into (4.1) yields that the map $k \mapsto \sigma_{\mathrm{imp}}(k,T)$ fully characterises the distribution of $\log(S_T)$ as

$$p_T(k) = \left(\frac{\partial^2 \mathrm{C}_{\mathrm{BS}}(k,T,\sigma_{\mathrm{imp}}(k,T))}{\partial K^2}\right)_{K=S_0 e^k}. \tag{4.3}$$

While a smooth input $\sigma_{\mathrm{imp}}(\cdot,T))$ is still needed, it is however easier than for option prices. A market standard is the Stochastic Volatility Inspired (SVI) parameterisation proposed by Gatheral (2004) (and improved in Gatheral and Jacquier (2013) and Guo et al. (2016)), where the total implied variance $w_{\mathrm{SVI}}(k,T) := \sigma^2_{\mathrm{imp}}(k,T)T$ is assumed to satisfy

$$w_{\mathrm{SVI}}(k,T) = a + b\left(k - m + \rho\sqrt{(k-m)^2 + \xi^2}\right), \quad \text{for any } k \in \mathbb{R}, \tag{4.4}$$

with the parameters $\rho \in [-1,1]$, $a, b, \xi \geq 0$ and $m \in \mathbb{R}$. The probability density function (4.1) of the log stock price then admits the closed-form expression (Gatheral 2004)

$$p_T(k) = \frac{g_{\mathrm{SVI}}(k,T)}{\sqrt{2\pi\, w_{\mathrm{SVI}}(k,T)}} \exp\left\{-\frac{d_-(k, w_{\mathrm{SVI}}(k,T))^2}{2}\right\}, \tag{4.5}$$

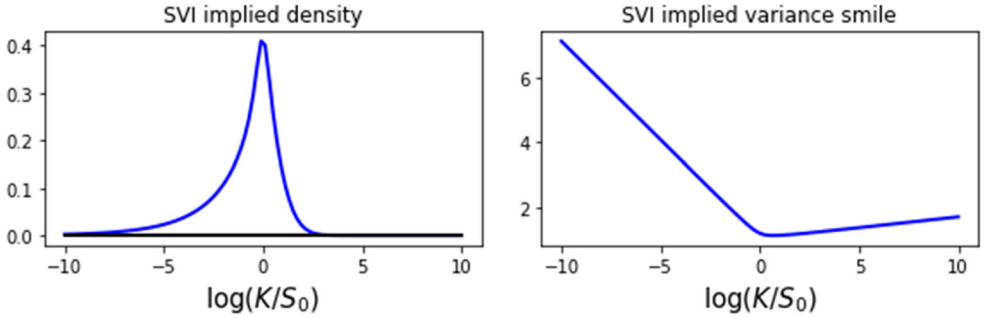


Fig. 10 Density of $\log(S_T)$ computed from 4.5 and the corresponding SVI total variance 4.4. The parameters are given in (4.6)

where

$$g_{\mathrm{SVI}}(k,T) := \left(1 - \frac{k w'_{\mathrm{SVI}}(k,T)}{2 w_{\mathrm{SVI}}(k,T)}\right)^2 - \frac{w'_{\mathrm{SVI}}(k,T)^2}{4}\left(\frac{1}{4} + \frac{1}{w_{\mathrm{SVI}}(k,T)}\right) + \frac{w''_{\mathrm{SVI}}(k,T)}{2},$$

where all the derivatives are taken with respect to k. In Fig. 10, we plot the typical shape of the implied volatility smile, together with the corresponding density for the following parameters:

$$a = 0.030358, \quad b = 0.0503815, \quad \rho = -0.1, \quad m = 0.3, \quad \xi = 0.048922, \quad T = 1. \tag{4.6}$$

4.2 Numerics

The goal of this numerical part is to be able to generate discrete versions of the SVI probability distribution given in (4.5). Our target distribution shall be the one plotted in Fig. 10, corresponding to the parameters (4.6). Since the Quantum GAN (likewise for the classical GAN) algorithm starts from a discrete distribution, we first need to discretise the SVI one. For convenience, we normalise the distribution on the closed interval $[-1, 1]$ and discretise with the uniform grid.

$$\left\{\left\lfloor (2^n - 1)\left(\frac{k+1}{2}\right)\right\rfloor\right\}_{k=0,\ldots,2^n-1},$$

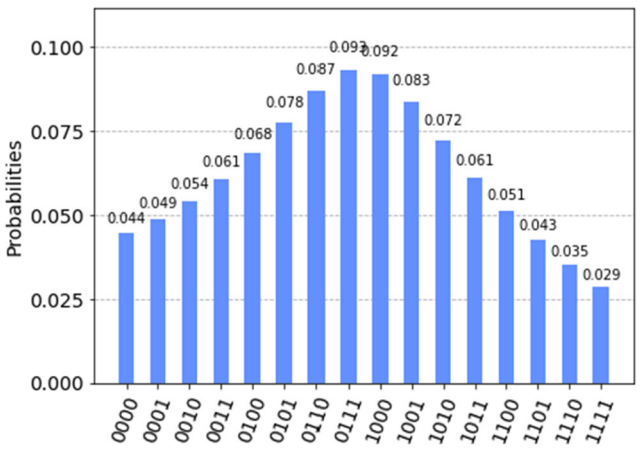


Fig. 11 Discretised version for the distribution of $\log(S_T)$ on $[-1, 1]$ with 2^4 points

which we then convert into binary form. This uniform discretisation does not take into account the SVI probability masses at each point, and a clear refinement would be to use a one-dimensional quantisation of the SVI distribution. Indeed, the latter (see (Pagès et al. 2004) for full details about the methodology) minimises the distance (with respect to some chosen norm) between the initial distribution and its discretised version. We leave this precise study and its error analysis to further research, in the fear that it would clutter the present description of the algorithm. The discretised distribution, with n qubits, together with the binary mapping, is plotted in Fig. 11 and gives rise to the wave function

$$|\psi_{\mathrm{target}}\rangle = \sum_{i=0}^{2^n-1} \sqrt{p_i}|i\rangle,$$

where, for each $i \in \{0, \ldots, 2^n - 1\}$,

$$p_i = \mathbb{P}\left(\log(S_T) \in \left[-1 + \frac{2i}{2^n}, -1 + \frac{2(i+1)}{2^n}\right)\right).$$

We need metrics to monitor the training of our QuGAN algorithm, for example the Fidelity function (Nielsen and Chuang 2000, Chapter 9.2.2)

$$\mathcal{F} : |v_1\rangle, |v_2\rangle \in \mathbb{C}^{2^n} \times \mathbb{C}^{2^n} \mapsto |\langle v_1||v_2\rangle|,$$

so that for the wave function (3.1) $|v_{\theta_G}\rangle = \sum_{i=0}^{2^n-1} v_{i,\theta_G}|i\rangle$, the goal is to obtain $\mathcal{F}\left(|v_{i,\theta_G}\rangle, |\psi_{\mathrm{target}}\rangle\right) = 1$, which gives $\mathbb{P}(|v_{\theta_G}\rangle = |i\rangle) = \left|v_{i,\theta_G}\right|^2 = p_i$, for all $i \in \{0, \ldots, 2^n - 1\}$. The Kullback-Leibler Divergence is also a useful monitoring metric, defined as

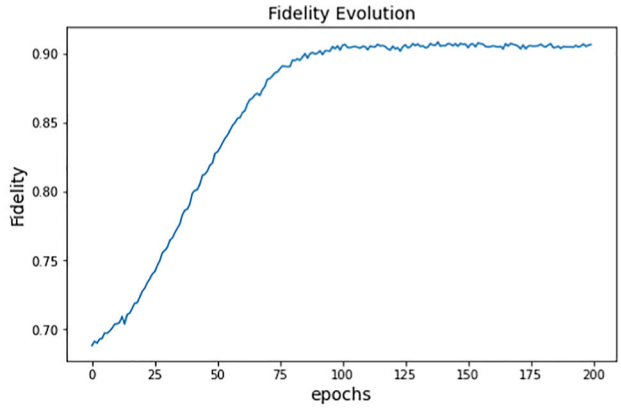
$$\mathrm{KL}(|\psi_{\mathrm{target}}\rangle, |v_{\theta_G}\rangle) := \sum_{i=0}^{2^n-1} p_i \log\left(\frac{p_i}{\left|v_{i,\theta_G}\right|^2}\right).$$

4.2.1 Training and generated distributions

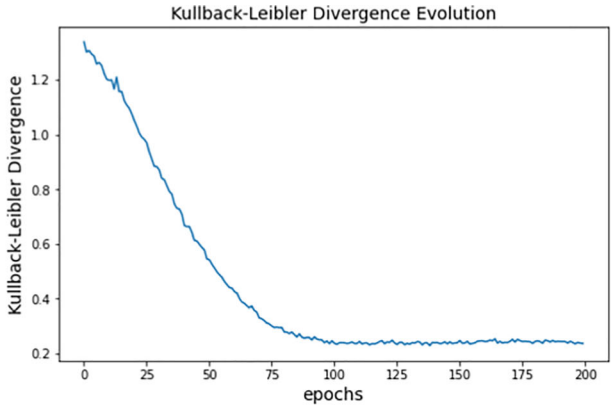
In the training of the QuGAN algorithm, in each epoch $\mathfrak{e}$, we train the discriminator $n_D = 9$ times and the generator $n_G = 1$. The results, in Figure 4.2.1, are quite interesting as the QuGAN manages to overall learn the SVI distribution. Aside from the limited number of qubits, the limitations however could be explained via the expressivity of our network which is only parameterised via $(\theta_i)_{i\in\{1,\dots,9\}}$ and $(w_i)_{i\in\{1,\dots,4\}}$ which is clearly not enough. This lack of expressivity is a choice, and more parameters deepen the network, but can create a barren plateau phenomenon (McClean et al. 2018), where the gradient vanishes in $\mathcal{O}(2^{-d})$ where d is the depth of the network. This would in turn require an exponentially larger number of shots to obtain a good enough estimation of (3.11), thereby creating a trade-off between expressivity and trainability in a differentiable manner.

4.2.2 Results: further improvements

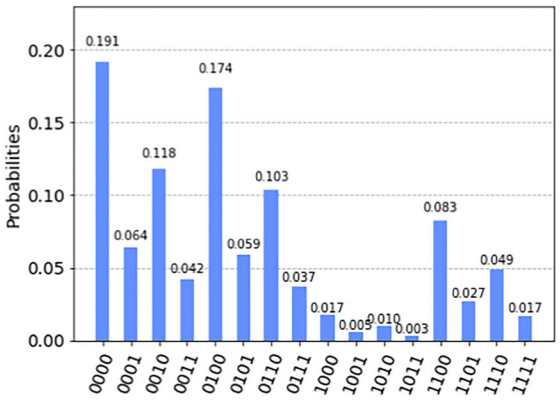
By looking at the obtained results, we are able to observe a convergence for the training routine that we have followed. However, the aforementioned convergence doe not occur at the neighborhood of 0 for the Kullback-Leibler Divergence proxy metric, this could be explained by the shape of the target distribution. Indeed, given any target distribution, the generator's architecture will allow for reproducing exactly the target distribution only for a unique set of variable θ. At this point, when combining this unicity in terms of optimal solution with the shape of the target distribution that induces a certain geometry for the score function that we are trying to optimise, there is a risk of converging at sub-optimal points, i.e. saddle points in our case. Therefore, an entire study on such geometry induced by the shape of the target along with the development of a strategy preventing us from falling into such saddle points will constitute potential future candidate for further research (Fig. 12).



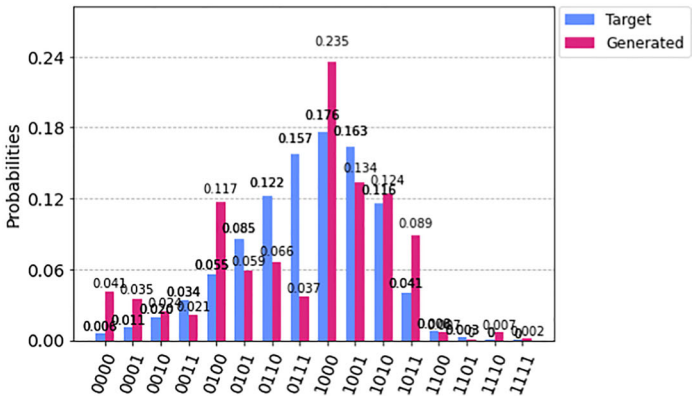
(A) Fidelity during QuGAN training.



(B) KL Divergence while training the QuGAN.



(A) Generated distribution at $\mathfrak{e} = 0$, with uniform random initialisation of $(\theta)_{i\in\{1,\dots,9\}}$.



(B) Comparison between the target and the generated distributions at the end of the training.

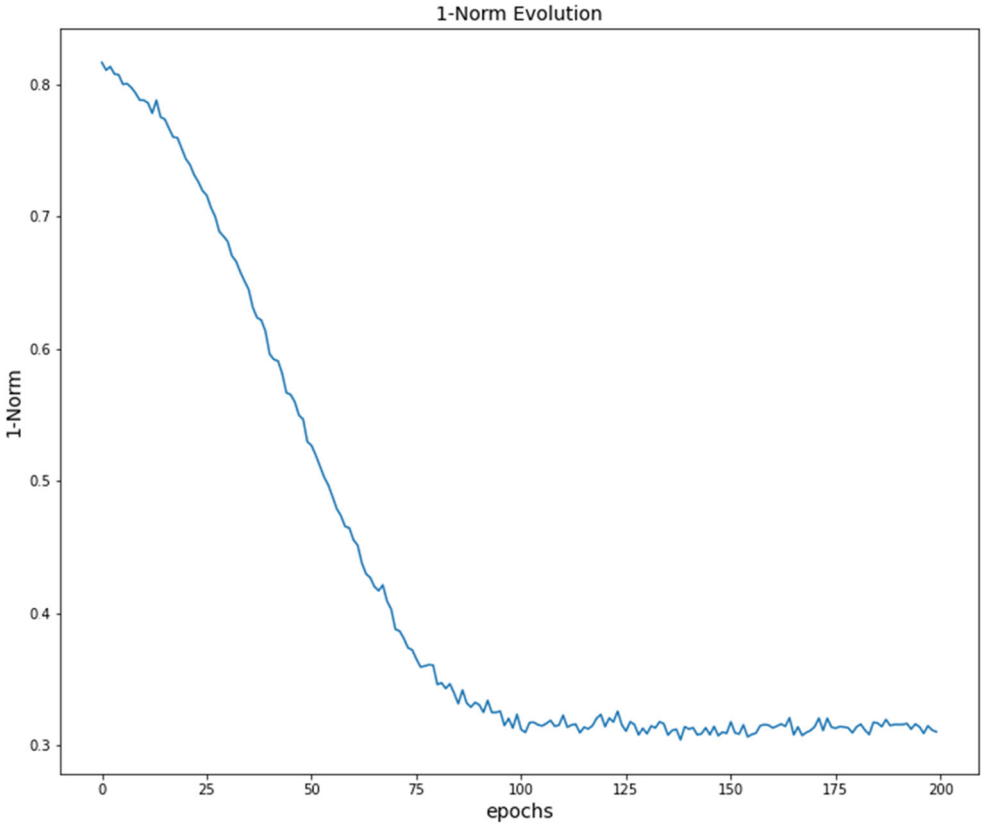


Fig. 12 Evolution of $\| |v_{\theta_G}\rangle\langle v_{\theta_G}| - |\psi_{\text{target}}\rangle\langle\psi_{\text{target}}| \|_1$ during QuGAN training

All the numerics in the paper were performed using the `IBM-Qiskit` library in `Python`.

Appendix A. Review of Quantum Computing techniques and algorithms

In Quantum mechanics the state of a physical system is represented by a ket vector $|v\rangle$ of a Hilbert space $\mathcal{H}$, often $\mathcal{H} = \mathbb{C}^{2^n}$. Therefore, for a basis $(|0\rangle, \ldots, |2^n - 1\rangle)$ of $\mathcal{H}$, we obtain the wave function $|v\rangle = \sum_{j=0}^{2^n-1} v_j |j\rangle$. The Hilbert space is endowed with the inner product $\langle v|w\rangle$ between two states $|v\rangle$ and $|w\rangle$, where $\langle v| := |v\rangle^\dagger$ is the conjugate transpose. Recall that a pure quantum state is described by a single ket vector, whereas a mixed quantum state cannot. The following are standard in Quantum Computing, and we recall them simply to make the paper self-contained. Full details about these concepts can be found in the excellent monograph (Nielsen and Chuang 2000).

Theorem A.1 (Born's rule) *If* $|v\rangle \in \mathbb{C}^{2^n}$ *be a pure state, then* $\|v\| = 1$.

Given a pure state $|v\rangle = \sum_{j=0}^{2^n-1} v_j |j\rangle$, the probability of measuring $|v\rangle$ collapsing onto the state $|j\rangle$ for $j \in \{0, \ldots, 2^n - 1\}$ is defined via

$$\mathbb{P}(|v\rangle = |j\rangle) = |\langle j|v\rangle|^2 = \text{Tr}\left(|j\rangle\langle j||v\rangle\langle v|\right) = |v_j|^2, \quad \text{(A.1)}$$

where Tr is the Trace operator. Moreover, for a given state $|v\rangle$, its density matrix is defined as $\rho_v := |v\rangle\langle v|$.

A.1. Quantum Fourier transform

In the classical setting, the discrete Fourier transform maps a vector $(x_0, \ldots, x_{2^n-1}) \in \mathbb{C}^{2^n}$ to

$$y_k = \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} \exp\left\{\frac{2i\pi jk}{2^n}\right\} x_j, \qquad \text{for } k = 0, \ldots, 2^n - 1. \quad \text{(A.2)}$$

Similarly, the quantum Fourier transform is the linear operator

$$|j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \exp\left\{\frac{2i\pi jk}{2^n}\right\} |k\rangle, \quad \text{(A.3)}$$

and the operator

$${}_{\text{q}}\mathcal{F} := \frac{1}{\sqrt{2^n}} \sum_{j,k=0}^{2^n-1} \exp\left\{\frac{2i\pi jk}{2^n}\right\} |k\rangle\langle j|$$

represents the Fourier transform matrix which is unitary as ${}_{\text{q}}\mathcal{F} * {}_{\text{q}}\mathcal{F}^\dagger = \text{I}_{\text{d}}$. In an n-qubit system ($\mathcal{H} = \mathbb{C}^{2^n}$) with basis $(|0\rangle, \ldots, |2^n - 1\rangle)$; for a given state $|j\rangle$, we use the binary representation

$$j := \overline{j_1 \cdots j_n}, \quad \text{(A.4)}$$

with $(j_1, \ldots, j_n) \in \{0, 1\}^n$ so that $|j\rangle = |j_1 \cdots j_n\rangle = |j_1\rangle \otimes \ldots \otimes |j_n\rangle$. Likewise, the notation $0.j_1 j_2 \ldots j_n$ represents the binary fraction $\sum_{i=1}^{n} 2^{-i} j_i$. Elementary algebra then yields

$$
{}_{\mathrm{q}}\mathcal{F}|j\rangle = \frac{1}{2^{\frac{n}{2}}} \left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.j_n}|1\rangle\right) \otimes \left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.j_{n-1}j_n}|1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.j_1 \ldots j_n}|1\rangle\right). \tag{A.5}
$$

A.2. Quantum phase estimation (QPE)

The goal of QPE is to estimate the unknown phase $\phi \in [0, 1)$ for a given unitary operator U with an eigenvector $|u\rangle$ and eigenvalue $\mathrm{e}^{2\mathrm{i}\pi\phi}$. Consider a register of size m, so that $\mathcal{H} = \mathbb{C}^{2^m}$ and define $b^* := \sup_{j \le 2^m \phi} \{j = 2^m 0.j_1 \cdots j_m\}$. Thus with $b^* = \overline{b_1 \cdots b_m}$, we obtain that $2^{-m} b^* = 0.b_1 \cdots b_m$ is the best m-bit approximation of ϕ from below. The quantum phase estimation procedure uses two registers. The first register contains the m qubits initially in the state $|0\rangle$. Selecting m relies on the number of digits of accuracy for the estimate for ϕ, and the probability for which we wish to obtain a successful phase estimation procedure. Up to a SWAP transformation, the quantum phase circuit gives the output

$$
|\psi_{\mathrm{out}}\rangle = \frac{\left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.\phi_m}|1\rangle\right) \otimes \left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.\phi_{m-1}\phi_m}|1\rangle\right) \otimes \cdots \otimes \left(|0\rangle + \mathrm{e}^{2\mathrm{i}\pi 0.\phi_1 \ldots \phi_m}|1\rangle\right)}{2^{\frac{m}{2}}},
$$

which is exactly equal to the Quantum Fourier Transform for the state $|2^m\phi\rangle = |\phi_1\phi_2 \ldots \phi_m\rangle$ as in A.5, and therefore $|\psi_{state}\rangle = {}_{\mathrm{q}}\mathcal{F}|2^m\phi\rangle$. Since the Quantum Fourier Transform is a unitary transformation, we can inverse the process to retrieve $|2^m\phi\rangle$. Algorithm 2 below provides pseudo-code for the Quantum Phase Estimation procedure and we refer the interested reader to Nielsen and Chuang (2000, Chapter 5.2) for detailed explanations.

Input: Unitary matrix U with $\mathrm{U}|u\rangle = \mathrm{e}^{2\mathrm{i}\pi\phi}|u\rangle$; $m = n + \lceil \log(2 + \frac{1}{2\varepsilon}) \rceil$ ancilla qubits initialised at $|0\rangle$.

Procedure:

1. $|0\rangle^{\otimes m}|u\rangle$ ▷ Initial state with $|0\rangle^{\otimes m}$ being the ancilla register and $|u\rangle$ the eigenstate register

2. $\longrightarrow \frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} |j\rangle|u\rangle$ ▷ Hadamard gates applied to the ancilla register

3. $\longrightarrow \frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} |j\rangle \mathrm{U}^j|u\rangle = \frac{1}{\sqrt{2^m}} \sum_{j=0}^{2^m-1} |j\rangle \mathrm{e}^{2\mathrm{i}\pi j\phi}|u\rangle$ ▷ Controlled U^j gates applied to the eigenstate register

4. $\rightarrow |\widetilde{\phi}\rangle|u\rangle$ ▷ Apply the inverse QFT where $\widetilde{\phi}$ is a m-qubit approximation of ϕ with an accuracy of 2^{-n}

5. $\rightarrow \widetilde{\phi}$ ▷ Measure $\widetilde{\phi}$ with a probability at least $1 - \varepsilon$

return Output: $\widetilde{\phi}$

Algorithm 2 Quantum phase estimation $(\mathbf{U}|u\rangle, m, \varepsilon)$

Acknowledgements The authors would like to thank Konstantinos Kardaras and Alexandros Pavlis for insightful discussion on quantum algorithms and spins.

Funding AJ received financial support from the EPSRC EP/T032146/1 grant.

Declarations

Conflict of interest The authors declare no competing interests.

References

Anand N, Huang P (2018) Generative modeling for protein structures ICLR 2018 Workshop

Arjovsky M, Chintala S, Bottou L (2017) Wasserstein generative adversarial networks, in International conference on machine learning. PMLR 214–223

Arnold V (1957) On functions of three variables. In: Proceedings of the USSR academy of sciences, vol 114

Bharti K, Cervera-Lierta A, Kyaw TH, Haug T, Alperin-Lea S, Anand A, Degroote M, Heimonen H, Kottmann JS, Menke T, Mok W.-K., Sim S, Kwek L-C, Aspuru-Guzik A (2021) Noisy intermediate-scale quantum (NISQ) algorithms

Black F, Scholes M (1973) The pricing of options and corporate liabilities. J Polit Econ 81:637–654

Braccia P, Caruso F, Banchi L (2021) How to enhance quantum generative adversarial learning of noisy information. New J Phys 23:053024

Buehler H, Gonon L, Teichmann J, Wood B (2019) Deep hedging. Quant Finance 19:1271–1291

Chakrabarti S, Huang Y, Li T, Feizi S, Wu X (2019) Quantum Wasserstein generative adversarial networks

Dallaire-Demers P.-L., Killoran N (2018) Quantum generative adversarial networks. Phys Rev A 98

Delbaen F, Schachermayer W (1994) A general version of the fundamental theorem of asset pricing. Math Ann 300:463–520

Denton E, Chintala S, Szlam A, Fergus R (2015) Deep generative image models using a Laplacian pyramid of adversarial networks in neurIPS

Deshpande I, Zhang Z, Schwing AG (2018) Generative modeling using the sliced Wasserstein distance. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3483–3491

Gatheral J (2004) A parsimonious arbitrage-free implied volatility parameterization with application to the valuation of volatility derivatives

Gatheral J (2006) The volatility surface. A Practitioner's Guide, Wiley Finance

Gatheral J, Jacquier A (2013) Arbitrage-free, SVI volatility surfaces. Quant Finance 14:59–71

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. Advances in neural information processing systems 27

Goodfellow IJ (2014) On distinguishability criteria for estimating generative models arXiv:1412.6515

Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC (2017) Improved training of Wasserstein GANs. In: NeurIPS, pp 5767–5777

Guo G, Jacquier A, Martini C (2016) Generalised arbitrage-free SVI volatility surfaces. SIAM Journal on Financial Mathematics 7:619–641

Herman D, Googin C, Liu X, Galda A, Safro I, Sun Y, Pistoia M, Alexeev Y (2022) A survey of quantum computing for finance, arXiv:2201.02773

Hu L, Wu S-H, Cai W, Ma Y, Mu X, Xu Y, Wang H, Song Y, Deng D-L, Zou C-L, et al. (2019) Quantum generative adversarial learning in a superconducting quantum circuit. Sci Adv 5:27–61

Huang H-L, Du Y, Gong M, Zhao Y, Wu Y, Wang C, Li S, Liang F, Lin J, Xu Y, Yang R, Liu T, Hsieh M.-H., Deng H, Rong H, Peng C-Z, Lu C-Y, Chen Y-A, Tao D, Zhu X, Pan J-W (2021) Experimental quantum generative adversarial networks for image generation

Kakutani S (1941) A generalization of Brouwer's fixed point theorem. Duke Math J 8:457–459

Kolmogorov A (1956) On the representation of continuous functions of several variables by superpositions of continuous functions of a smaller number of variables. In: Proceedings of the USSR Academy of Sciences, vol 108

Kondratyev A, Schwarz C (2019) The market generator Available at SSRN 3384948

Koshiyama A, Firoozye N, Treleaven P (2021) Generative adversarial networks for financial trading strategies fine-tuning and combination. Quant Finance 21:797–813

Lloyd S, Weedbrook C (2018) Quantum generative adversarial learning. Phys Rev Lett 121

Marblestone AH, Wayne G, Kording KP (2016) Toward an integration of deep learning and neuroscience. Front Comput Neurosci 10:94

McClean JR, Boixo S, Smelyanskiy VN, Babbush R, Neven H (2018) Barren plateaus in quantum neural network training landscapes. Nat Commun 9

Miyato T, Kataoka T, Koyama M, Yoshida Y (2018) Spectral normalization for GANs arXiv:1802.05957

Molnar C (2020) Interpretable machine learning Lulu. com

Nakaji K, Yamamoto N (2021) Quantum semi-supervised generative adversarial network for enhanced data classification. Sci Rep 11:1–10

Ni H, Szpruch L, Wiese M, Liao S, Xiao B (2020) Conditional Sig-Wasserstein GANs for time series generation arXiv:2006.05421

Nielsen MA, Chuang IL (2000) Quantum computation and quantum information. Cambridge University Press, Cambridge

Niu MY, Zlokapa A, Broughton M, Boixo S, Mohseni M, Smelyanskyi V, Neven H (2022) Entangling quantum generative adversarial networks. Phys Rev Lett 128:220505

Pagès G., Pham H, Printems J (2004) Optimal quantization methods and applications to numerical problems. In: Finance, in Handbook of Computational and Numerical Methods in Finance. Springer

Radford A, Metz L, Chintala S (2016) Unsupervised representation learning with deep convolutional generative adversarial networks

Ruf J, Wang W (2021) Neural networks for option pricing and hedging: a literature review. Journal of Computational Finance 24

Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved techniques for training GANs in neurIPS

Saxena D, Cao J (2021) Generative adversarial networks (gans) challenges, solutions, and future directions. ACM Computing Surveys (CSUR) 54:1–42

Schawinski K, Zhang C, Zhang H, Fowler L, Santhanam GK (2017) Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. Monthly Notices of the Royal Astronomical Society: Letters 467

Schuld M, Bergholm V, Gogolin C, Izaac J, Killoran N (2019) Evaluating analytic gradients on quantum hardware. Phys Rev A 99

Situ H, He Z, Wang Y, Li L, Zheng S (2020) Quantum generative adversarial network for generating discrete distribution. Inf Sci 538:193–208

Stein SA, Baheri B, Tischio RM, Mao Y, Guan Q, Li A, Fang B, Xu S (2020) QuGAN: A generative adversarial network through quantum states

Wang P, Wang D, Ji Y, Xie X, Song H, Liu X, Lyu Y, Xie Y (2019) QGAN: Quantized generative adversarial networks

Wiese M, Knobloch R, Korn R, Kretschmer P (2020) Quant gan: deep generation of financial time series. Quantitative Finance 20:1419–1440

Yu J, Lin Z, Yang J, Shen X, Lu X, Huang T (2018) Generative image inpainting with contextual attention. In: Proceedings of the IEEE conference on computer vision and pattern recognition

Zhavoronkov A (2019) Deep learning enables rapid identification of potent DDR1 kinase inhibitors. Nat Biotechnol 37:1038–1040

Zoufal C, Lucchi A, Woerner S (2019) Quantum generative adversarial networks for learning and loading random distributions. Npj Quantum Inf 5:1–9