Quantum function fitting and classification beyond the single-qubit model

Author: Berta Casas Font
Supervised by: Alba Cervera-Lierta
Barcelona Supercomputing Center
July 11, 2022

Quantum function fitting and classification beyond the single-qubit model

Berta Casas Font

Supervised by: Alba Cervera-Lierta[1]

Barcelona Supercomputing Center, Barcelona[1]

11 July 2022

Quantum Neural Networks (QNNs) have emerged as one promising Quantum Machine Learning (QML) technique. While the models for single and multi-qubit QNNs have been extensively studied, it remains unknown if using higher-dimensional systems provide any advantage. In this work, we investigate the theoretical foundation of the qubit model and we compare it with the qutrit prototype. First, we show that a single qubit can reproduce a Fourier series, while a qutrit can fit a more complicated type of function, with additional degrees of freedom that the model can adjust. Second, we explore the benefits of the third extra level of the qutrit for the classification task. In addition, we examine the two-qubit classifier and see that using a local cost function on the training improves the results, according to recent studies. Beyond the theoretical discussion, we provide numerical benchmarks of the models studied.

Keywords: Quantum neural network, Qutrits, Quantum machine learning.

Acknowledgements

I would like to thank Dr. Alba Cervera-Lierta for her guidance and support during the project. Also, I would like to acknowledge the Quantic BSC group for the suggestions and discussions provided during these months.

Contents

Berta Casas Font: berta.casas@bsc.es

1 Introduction

Quantum mechanics is a field of physics that describes the microscopic world we live in. The basis of everything is quantum-mechanical. Even though we have a solid mathematical theory that describes quantum mechanics and it has many applications we use on a daily basis, it still challenges our intuition. In 1980 it was first proposed to study quantum systems with a quantum Turing machine, by Paul Benioff [Ben80]. At first, it was a theoretical idea that suggested that there could exist machines that operate under the laws of quantum mechanics, performing universal computation and extending the limits of classical computation. This idea gained importance when Feynman suggested that these computers could be useful to simulate nature [Fey82]. Quantum systems grow exponentially with the number of particles. Then, it is difficult to simulate big systems on a conventional computer since it consumes an exponential amount of computational resources, such as memory and time. The idea of a quantum computer enriched our understanding of quantum mechanics: from studying quantum mechanical systems in nature to using nature to design computers.

Quantum computers are more than a theoretical computational model or a machine to simulate quantum mechanical systems. This was evidenced when Shor proposed in 1994 a quantum algorithm that enables efficiently factorizing numbers [Sho94]. Most of the current cryptographic classical protocols, such as RSA [RSA78], rely on the fact that factorization is a hard problem on conventional computers. The quantum algorithm version provides an exponential speed-up compared with the most efficient classical algorithms, being able to break present cryptography. Shor's algorithm showed that this quantum model of computation offered more possibilities beyond nature simulation. This algorithm is an example of the computational extension that implies a quantum mechanical model of computation. It belongs to the BQP complexity class, which includes the type of algorithms that can be efficiently solved on a quantum computer. This class goes beyond classes like P, where are included those algorithms efficiently solvable by classical computers. Since

then, the field of quantum computation has quickly grown, and now we are witnessing its biggest expansion with the construction of the first quantum computers.

By analogy to the classical bit, in quantum computation is used the qubit, which is the minimal unit of quantum information. The classical bit can be in a well-defined state, zero or one. The qubit can be in one of these states, or it can be in a superposition of both. The general state of this system is expressed as:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{1}$$

where $\alpha, \beta \in \mathbb{C}$ are the amplitudes, $|\alpha|^2+|\beta|^2 = 1$ and $|0\rangle$ and $|1\rangle$ are orthonormal states. If the qubit is in this superposition state and we measure it, it will collapse in the state $|0\rangle$ with probability $|\alpha|^2$ and with probability $|\beta|^2$ into the state $|1\rangle$. Although the qubit is widely used, there are higher-dimensional systems that can be used for quantum computation, the qudits. These have dimension d and their general state is given by $|\psi\rangle = \sum_{i=0}^{d-1} c_i\,|i\rangle$ where $\sum_{i=0}^{d-1} |c_i|^2 = 1$ and $\{|0\rangle\,, |1\rangle\,, ..., |d-1\rangle\}$ are the computational basis states of a qudit, which are orthonormal, $\langle i|j\rangle = \delta_{ij}$. The next high-dimensional state after qubits is the qutrit, so they are becoming the first qudits to be studied in detail. There is evidence that in some cases higher-dimensional systems such as qutrits can improve performance in some algorithms [Gus22]. In the recent years, qutrits are becoming another experimental quantum computational tool [BRS+21, YSK+20, RMP+21, BRS+21, CLKAGG22].

The state-of-the-art of quantum computing technology is in the so-called Noisy Intermediate-Scale Quantum (NISQ) era [Pre18, KBKH+22], which refers to the fact that we only have access to noisy devices with a limited number of qubits (currently 50-100 qubits) which are not error corrected. For applying quantum error correction, we need thousands of physical qubits for implementing the codes, which is beyond state-of-the-art. Several quantum algorithms, such as Shor's factorization algorithm, require an error-correcting scheme to work. Because of the lack of fault-tolerant quantum computers, lots of efforts have been made on developing algorithms that can be supported by current noisy quantum computers.

In this framework emerge the Variational Quantum Algorithms (VQA) [CAB+21]. They pertain to a family of quantum-classical hybrid algorithms. The quantum part comprises a parameterized quantum circuit (PQC), i.e., a quantum circuit that depends on a series of parameters that can be fine-tuned and is used to compute an expected value. The classical part is an optimization subroutine, used to find a better approach to the parameters of the PCQ and update them variationally. VQA relies on the variational principle, which states that, given a Hamiltonian H and an arbitrary quantum state $|\psi\rangle$, the expected value of H in this state is an upper bound to the ground state energy [HNPR86]. In VQA algorithms, we produce variational states, which give us an upper bound to the solution of our problem. The first VQA algorithms proposed were the Variational Quantum Eigensolver (VQE) [Per14], which finds an approximation to the ground state energy of a given Hamiltonian, and the Quantum Approximate Optimization Algorithm (QAOA) [FGG14], proposed to solve combinatorial optimization problems. The variational core of the algorithms allows them to have a noise-resistance nature, because of the adaptability of the parameters. These can adopt a value that cancels partially the noise present in the quantum device, which makes the circuit more resilient to noise [FFR+21]. Hence, these algorithms are appropriate for current NISQ devices.

Besides chemistry and optimization applications, represented by the VQE and QAOA algorithms, the VQA have also applications in Quantum Machine Learning (QML). QML is a research area that integrates Machine learning (ML) and quantum computation. Although the quantum advantage of these algorithms has not been proved yet, the wide range of possible applications and its hybridization with classical computational techniques have

aroused great interest in the scientific community. QML includes classical algorithms that process quantum data and quantum algorithms that process quantum or classical data [BWP+17, CHI+18]. In the QML algorithms that process classical data in a quantum computer, exist several strategies used to encode the data such as quantum kernel methods [SK19, HCT+19], or data re-uploading [PSCLGFL20].

Some QML algorithms, such as the ones that use data re-uploading, are inspired by the structure of classical Neural Networks (NN). NN are widely used in ML and pertain to the class of supervised learning models [JGR20]. In supervised models, we have two different sets of data: the training and the testing set. The training consists of a set of data points, and each one has assigned a label of the category it pertains to. When training the model with this data, we will correct it when it makes a bad prediction and, by optimizing a cost function that codifies the solution of our problem, we will go in the direction of correct classification. Then, the testing set, which has to be different from the training, will be passed to the model to see how it performs with unseen data. In this way, we can verify if the model has learned the distribution. The quantum analog of NN are called Quantum Neural Networks (QNN), and VQA are one possible implementation of these QNN.

In this work, we approach function fitting and classification problems using a QNN trained as a VQA. We study these two problems as they are related because classification consists in delimiting regions, which can be understood as drawing one or more functions in order to separate regions. Our goal is to explore new directions in this field by studying these problems with high-dimensional systems, in particular with qutrits. We will study the type of functions we can generate with the qubit and the qutrit model and compare their performance depending on the number of parameters and gates used in the model. We will see that qutrits perform better than qubits under some figures of merit. All benchmarks will be simulated classically since we aim to study the theoretical performance but, since VQA algorithms are supported in NISQ devices, the algorithms could be tested in real quantum computers.

The work is structured as follows: In the first section, we will introduce theoretical concepts that later will be used in this work. In the second section, we will approach function fitting. Depending on the choice of the PQC and the system used to encode the data, we will generate different quantum models. We will study the functions generated with a general quantum model. Later, we will discuss specific examples such as the single-qubit, with which it has been proven universality [PSCLGFL20, YYLW22]. We also explore the qutrit model, which, as far as we know, has never been used in this context. The third section consists of the study of the classification problem. We will study the performance of different classifiers, providing an intuition of what effects may be affecting the model. Finally, in the conclusion and discussion section, we summarize the results obtained in the work and we comment on further directions.

2 Theoretical background

In this section we will investigate the qutrit, the 3-level system that we use in this work. Also, we explore some of the most relevant concepts about VQA and QML.

2.1 Qutrits

Most of the quantum algorithms and communication protocols are designed for qubits. However, great efforts have been made in the last years in studying higher-dimensional level systems, such as the qutrit. The qutrit has been explored in the context of quantum

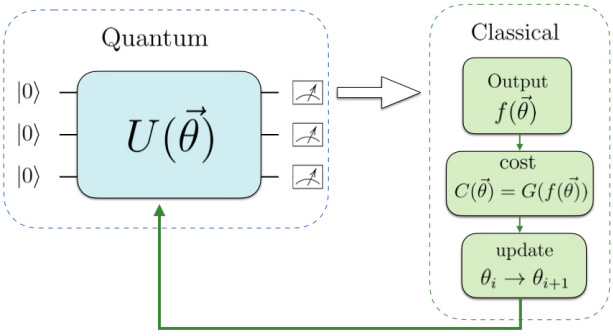


Figure 1: Schematic representation of a VQA. They are divided into two parts: The quantum part, which consists in the generation of a variational state with a PQC which it is later measured. As a result, we obtain a probability distribution, with which we build an expected value. This is used to construct a cost function. At each iteration, a classical subroutine optimizes the cost function and the parameters are introduced in the quantum circuit. The procedure is repeated until we converge on the solution to our problem.

cryptographic protocols [GJV+06], and multipartite entanglement [CM00]. Moreover, it has been demonstrated that, in some cases, for simulating dimensional systems greater than two, qutrits are more tolerant to gate noise than qubits [Gus22].

One of the potential advantages of using a high-dimensional system such as the qutrit is that we have an extra level, where we can encode information in. Because we are in the NISQ era, it is crucial to use the minimum amount of resources, i.e., gates and circuit depth, to perform calculations. The existence of this extra level could reduce the resource requirements to achieve a similar exploration of the Hilbert space. Moreover, the 3-dimensional Hilbert space has more structure than the 2-dimensional, in the sense that we have a wider variety of quantum gates, richer entanglement, and more parameters to describe a general unitary transformation. This will be useful when using a PQC.

Let us explore some general properties of these systems such as the general state of a qutrit:

$$|\psi\rangle = \alpha\,|0\rangle + e^{i\theta}\beta\,|1\rangle + e^{i\phi}\gamma\,|2\rangle\,, \tag{2}$$

where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$ with $\alpha, \beta, \gamma \in \mathbb{C}$ and $\theta, \phi \in \mathbb{R}$. $\{|0\rangle, |1\rangle, |2\rangle\}$ form an orthonormal basis. The unitary operators of a qutrit pertain to the special unitary group of dimension 3, SU(3). The 8 generators of this space are the Gell-Mann matrices λ_i. With the exponentiation of these matrices, we can generate one-qutrit rotations on a two-level subspace. For example, $R_y^{(02)}$ performs a R_y rotation on the subspace of the first and third level, leaving the second untouched. Since we can act with these one-qutrit gates, with a qutrit we can also simulate a qubit if we only operate on two of the levels. Then, all the results that we prove for qubits, can also be applied with qutrits with the correct implementation. We give more detail on Gell-Mann matrices and one-qutrit rotation gates in Appendix A.

Since we have eight generators, we also would need the same number of parameters to describe a general unitary transformation, while for qubits we only need 3 of them. One possible decomposition of a unitary transformation for a single qutrit is given by [DW11]:

$$M = R_y^{(01)}(\beta)R_y^{(02)}(\gamma)R_y^{(01)}(\delta)R_z^{(01)}(\theta)R_z^{(02)}(\rho)R_y^{(01)}(\beta')R_y^{(02)}(\gamma')R_y^{(01)}(\delta'), \tag{3}$$

2.2 Variational Quantum Algorithms

Here we examine VQA deeply and we present their main applications. The general structure of these algorithms is shown in Fig. 1. First, we will give details of the general procedure, and later we will focus on the structure of the PQC.

The generic process on a VQA is the following: first, we prepare the initial state, which is usually the zero state, because it is easy to prepare for a quantum computer: $|0\rangle^{\otimes n}$, where n is the number of qubits or qutrits used. Then, we apply the PQC, which can be expressed as a unitary that depends on some parameters: $U(\vec{\theta}) = U(\theta_1, \theta_2, ...\theta_k)$, where k is the total number of parameters used. After this, we obtain a given state $|\psi(\vec{\theta})\rangle = U(\vec{\theta})\,|0\rangle$ and we measure it to obtain a probability distribution. We post-process it, obtaining another value, for instance, an expected value or fidelity, and we use it to build a cost function. This must encode the solution of the problem we aim to solve. Then, the cost function is optimized with some classical subroutine, such as gradient descent method, direct search, or other techniques such as genetic algorithms. We find a better approach to the parameters $\vec{\theta}$ and these are updated again in the PQC, repeating the procedure until we converge on a solution. Finally, we have a set of parameters $\vec{\theta}_{\text{opt}}$ that generate the unitary circuit that prepares the ground state of our problem cost-function.

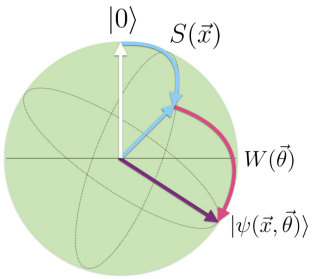
Let's take a closer look at the structure of the PQC. The strategy we use is to encode the parameters in the angles of the rotations of qubit or qutrit gates. The design of the circuit with the parameterized gates is called the ansatz. For designing the PQC ansatz, we will use only one and two-qubit or qutrit gates, since any unitary of n-qubits or qutrits has an approximate version, with arbitrarily small error, with only one and two-qubit or qutrit gates. This is ensured by the Solovay–Kitaev theorem [DN05], which states that this decomposition can be approximated by a circuit with polynomial depth with the number of qubits or qutrits. This is a theorem of existence and, then, it does not provide the exact decomposition of the unitary, but it provides a mathematical statement that justifies the use of only one and two-qubit/qutrit gates. For qubits we will use the CNOT gate and the rotation gates, $R_i(x) = \exp(ix\sigma_i/2)$, with $i \in \{x, y, z\}$. For qutrits, we can use single-qutrit rotation gates such as $R_y^{(02)}$, and one possible generalization of the CNOT gate, the ternary controlled-X gate [DW11]. More details about these gates can be found in Appendix A.

Ideally, we want to have the most general unitary transformation in order to explore the majority of the Hilbert space. The reality is that when the number of qubits or qutrits grows, the Hilbert space also grows exponentially. Thus, it is difficult to examine the space. This is related to the expressibility of the PQC, which is the ability of this circuit to generate diverse variational states. In VQA we also have to deal with the Barren-Plateaus (BP) effect [MBS+18], which consists in a vanishing gradient effect on the cost function. If the gradient becomes flat, the classical optimization methods are not capable to find a solution or get trapped in local minima, an effect caused precisely for the exponential size of the Hilbert space. In this work, we are not going to focus on this effect. However, we will explore the recent evidence stating that with a local cost function we can avoid partially the BP effect [UB21].

2.3 Quantum Machine Learning

We can classify classic and quantum ML algorithms with two criteria: depending on the algorithm nature, quantum or classic, and depending on the data processed, which can also be quantum or classic. In this work, we focus on quantum algorithms that process classical data.

(a) General procedure on a QML algorithm.



(b) Example of the QML procedure for a single qubit.

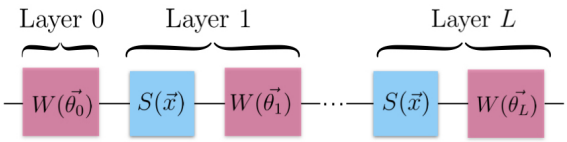
Figure 2: In the figure on the left, we represent the basic procedure of a QML algorithm: the encoding, the data processing, and the measurement. In figure (b), we show a diagrammatic example of this procedure for a qubit. The action of the encoding gate $S(\vec{x})$ is represented by the blue line, and the trainable gate $W(\vec{\theta})$ action by the magenta line.

The general procedure of a QML algorithm is shown in Fig. 2a. First, we have to encode the data into the circuit. We have different strategies for this. One approach is to encode it in the amplitude of the state, i.e., $|\psi\rangle = \sum_{i=1}^{M} x_i |i\rangle$, with M the total number of points. A more direct option is to encode it in the parameters of a unitary rotation. For instance, to encode a given data point in one qubit we can use a rotational gate and introduce it like $|\psi\rangle = \cos\frac{x}{2}|0\rangle + \sin\frac{x}{2}|1\rangle$. Then, we encode the data in the circuit via the parameters of the gates. To encode more than one data point, we can introduce them into one qubit using more than one gate, e.g., for encoding $\vec{x} = (x_1, x_2)$ we can do it like $R_z(x_1)R_z(x_2)$. For the data processing, we can follow a similar strategy, already explored in VQA algorithms: to use parameterized gates. Therefore, in a QML PQC of this kind, we have two types of gates: the encoding gates $S(x)$, which are the one that introduces the data, and the trainable gates $W(\vec{\theta})$, which contain the parameters that are optimized and that process the data. Finally, we measure the output of the circuit and we post-process it by introducing it to the optimization subroutine, which is the same as in the VQA algorithm. A specific example of the encoding and the processing of the data is given in 2b, where we apply the procedure explained to a single qubit. We use the encoding gate to introduce the data into the PQC and a trainable gate to process it.

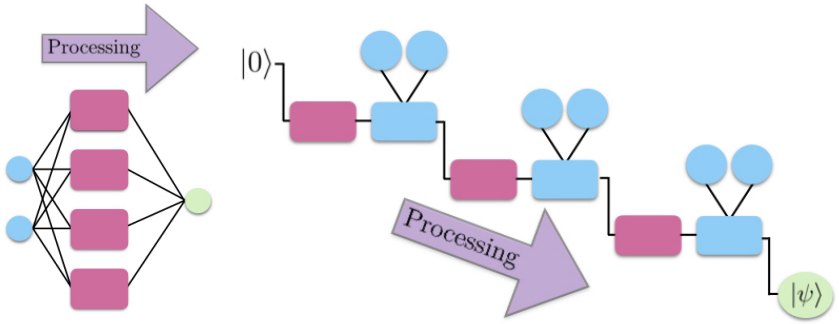
Another strategy to encode the data in the circuit is the so-called data re-uploading [PSCLGFL20]. This technique consists in repeatedly introducing the data into the system via an encoding gate followed by a trainable gate, which process it, repeating this formula several times. A representation of this procedure for one qutrit or qubit is shown in Fig. 3a. The combination of an encoding and a trainable gate forms a layer, except the layer 0, which consists only of a trainable gate. We will use various layers, uploading the data and processing it with the tunable parameters $\vec{\theta_i}$ on the trainable gates. These parameters will be different from layer to layer. Conversely, the data point will be the same at every layer.

As illustrated in Fig. 3b and 3c, the parameters of the gates are usually compared to the weights and the biases of a NN because they play a similar role: they have to be trained to solve a specific problem or task. Also, a layer of the data re-uploading model acts similar to a single neuron in NN, because in both cases every unit is repeatedly fed with the data. Once uploaded the data with the encoding gate, the trainable gate only offers a unitary transformation. We have to repeatedly re-upload the data in the circuit, alternating it with a processing gate. It was proven that, with this architecture, it was achieved universality [PSCLGFL20]. It is not enough to upload the data once and then use only trainable gates,

since all the parametric unitary operations can be grouped forming another single unitary. Hence, we need to repeat the uploading of the data at each step. This is what introduces the non-linearities needed to solve function fitting and classification problems.



(a) Schematic representation of data re-uploading.



(b) Neural network (c) Quantum neural networl using data re-uploading.

Figure 3: In the figure above, we have pictured the data re-uploading ansatz for a qubit or a qutrit. The gates $W^{(i)} = W(\vec{\theta}_i)$, for $i \in \{0, .., L\}$, are the trainable gates and $S(x)$ are the encoding gates. We have an initial layer composed of a parameterized gate $W^{(0)}$. In addition to this layer, we have L layers composed of one encoding gate with the same data point everywhere and another parameterized gate, with different parameters in each layer. The two figures below show a comparison of a one-layer NN with a one-qubit quantum model based on data re-uploading. We can see that at each layer of the NN the data is passed as an input to each neuron of the layer. On the other hand with the quantum circuit, we introduce the data at each step. Every pair of boxes on (c) represents a layer and we compare it with a neuron of a NN.

3 Function fitting

In this section, we aim to solve function fitting with the data re-uploading procedure. We present the general form of the model for a qubit or a qutrit. The results can also be generalized to qudits. For the concrete case of one qubit, there is evidence that claims that we can reproduce any arbitrary function [SSM21], [YYLW22]. We compare both qubit and qutrit models and we observe that they do not generate the same type of functions, at least with the ansatzs used.

The goal of function fitting is the following: given an arbitrary function $g(\vec{x})$, we train our circuit ansatz to produce an output that, with the correct post-processing, approximates well the target function, i.e., $f_{\boldsymbol{\theta}}(\mathbf{x})$ (from now one we replace $\vec{x} = \boldsymbol{x}$ and $\vec{\theta} = \boldsymbol{\theta}$). The quantum model used in this section is given by

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \langle 0| U^{\dagger}(\boldsymbol{x}, \boldsymbol{\theta}) O U(\boldsymbol{x}, \boldsymbol{\theta}) |0\rangle , \tag{4}$$

where $U(\boldsymbol{x}, \boldsymbol{\theta})$ represents the PQC (we refer the reader to Fig. 3a for an example with one qubit), O a given observable and $\mathbf{x}$ is the specific point for which we are computing the function. Following Ref. [SSM21], we see that the quantum model can naturally represent

a function of the type

$$f_{\boldsymbol{\theta}}(x) = \sum_{\omega \in \Omega} c_\omega(\boldsymbol{\theta}) e^{i\omega \cdot x}, \tag{5}$$

which is a Fourier series only if $\omega \in \mathbb{K}$. In this case, the function is a truncated Fourier series. We provide the definition of a Fourier series and a Laurent Polynomial in Appendix B.

We will study the case $x \in \mathbb{R}$. For uploading the data in the general unitary $U(x, \boldsymbol{\theta})$, we use an encoding gate of the form $S(x) = e^{-ixH}$, where H is an arbitrary encoding Hamiltonian. Let us consider the following general ansatz:

$$U(x, \boldsymbol{\theta}) = W^{(L)} S(x) W^{(L-1)} ... W^{(1)} S(x) W^{(0)}, \tag{6}$$

where $W^{(i)} = W(\boldsymbol{\theta_i})$, for $i \in \{0, ..., L\}$, are the gates that contain the trainable parameters, and $S(x)$ the encoding gates. A circuit example that implements this unitary transformation for one qubit or qutrit is the one in Fig. 3*a*. The Hamiltonian of the encoding blocks $S(x) = e^{-ixH}$ is arbitrary and, in principle, does not have to be diagonal. However, if singular value decomposition is performed $H = V \Sigma V^\dagger$, being V a unitary matrix and Σ a matrix with diagonal entries only, we can write the encoding block as

$$S(x) = e^{-ixV\Sigma V^\dagger} = \sum_{m=0}^{\infty} \frac{1}{m!} (-ixV\Sigma V^\dagger)^m = 1 + \sum_{m=1}^{\infty} \frac{1}{m!} V(-ix\Sigma)^m V^\dagger = V e^{-ix\Sigma} V^\dagger. \tag{7}$$

For every $m \geq 1$ we have the terms $(-Vix\Sigma V^\dagger)...(-Vix\Sigma V^\dagger)$ in the sum and $V^\dagger V = \mathbb{I}$. Thus, we can express the encoding block as a diagonal matrix with a change of basis. With the final term in Eq. (7), we can 'absorb' V and $V^\dagger$ with the tunable gates in Eq. (6), such that we define $W'^{(i)} = V^\dagger W^{(i)} V$, except for $W'^{(0)} = V^\dagger W^{(0)}$ and $W'^{(L)} = W^{(L)} V$, without loss of generality, because W are tunable parametric unitary gates and V, $V^\dagger$ are also unitary. For this reason, we can consider that the Hamiltonian of the encoding block is diagonal $H = \text{diag}(\lambda_1, ..., \lambda_N)$.

The quantum model from Eq. (4) can be expressed as $\langle \psi(x, \boldsymbol{\theta}) | \, O \, | \psi(x, \boldsymbol{\theta}) \rangle$, where the state is given by $|\psi(x, \boldsymbol{\theta})\rangle = W^{(L)} e^{-ixH} W^{(L-1)} ... W^{(1)} e^{-ixH} W^{(0)} |0\rangle$. The full derivation of the final form of the quantum model in Eq. (4) can be found in Appendix C. Here we present the result:

$$f_{\boldsymbol{\theta}}(x) = \sum_{\mathbf{k},\mathbf{j} \in [N]^L} a_{\mathbf{k},\mathbf{j}} e^{-ix(\Lambda_{\mathbf{k}} - \Lambda_{\mathbf{j}})}. \tag{8}$$

$\mathbf{j}$ and $\mathbf{k}$ are multi-indexes, defined as $\vec{j} \equiv \{j_1, ..., j_L\} \in [N]^L$, being L the number of layers and $N = d^n$, with d the dimension of the qubit or the qutrit, and n how many of them we use. Each $\mathbf{j}$ is a combination of L numbers that can go from 0 to $N - 1$. Then, we have N^L possible values. For example, for a single qutrit with 2 layers, we have $N = 3^1$ and $\mathbf{j} = \{j_1, j_2\}$, where we have 3 possibles values of each j_i, because each one runs from 0 to $N - 1 = 2$. Also, we have defined $\Lambda_{\mathbf{j}} = \lambda_{j_1} + ... + \lambda_{j_L}$, which is a multi-index sum of L eigenvalues of H. This means that $\Lambda_{\mathbf{j}}$ has also N^L possible values. Following the previous example, $\Lambda_{\mathbf{j}}$ consists of the sum of $L = 2$ terms, and each j_i can have 3 possible eigenvalues: $\Lambda_{\mathbf{j}} = \{\lambda_0 + \lambda_0, \lambda_0 + \lambda_1, ..., \lambda_2 + \lambda_2\}$.

Finally, the terms that pertain to the trainable gates and the observable are grouped like:

$$a_{\mathbf{k},\mathbf{j}} = \sum_{i,i'} (W^*)^{(0)}_{i,k_1} (W^*)^{(1)}_{k_1,k_2} ... (W^*)^{(L)}_{k_L,i} O_{i,i'} W^{(L)}_{i',j_L} ... W^{(1)}_{j_2,j_1} W^{(0)}_{j_1,1}, \tag{9}$$

where we sum over i, i' and the free indices are $\mathbf{k} = (k_1, ..., k_L)$ and $\mathbf{j} = (j_1, ..., j_L)$. Comparing Eq. (33) with Eq. (5) we can see, by analogy, that the 'frequency spectrum' Ω', which,

in reality, it will only be a frequency spectrum if the model generate integer frequencies, is given by

$$\begin{aligned}\Omega' =&\{\Lambda_{\mathbf{k}} - \Lambda_{\mathbf{j}} \mid \mathbf{k},\mathbf{j} \in [N]^L\} = \\ &\{\lambda_{k_1} + ... + \lambda_{k_L} - (\lambda_{j_1} + ... + \lambda_{j_L}) \mid j_i, k_i \in \{1, ..d\} \text{ for } i \in \{1, ..., L\}\},\end{aligned} \tag{10}$$

where each λ_{j_i} is a possible eigenvalue of the encoding Hamiltonian. The 'spectrum', then, is given by all the N^L sums of possible combinations of the eigenvalues λ_i minus another possible sum of other N^L eigenvalues. Therefore, the 'frequency spectrum' is fully characterized by the eigenvalues of the encoding Hamiltonian. Regarding the coefficients c_ω, they are given by

$$c_\omega = \sum_{\substack{\mathbf{k},\,\mathbf{j}\in[d]^L \\ \Lambda_{\boldsymbol{k}}-\Lambda_{\mathbf{j}}=\omega}} a_{\mathbf{k},\,\mathbf{j}}, \tag{11}$$

Taking everything into consideration, we can conclude that the general quantum model can generate a function like the one in Eq. (5), which is a truncated Fourier series if the frequencies generated are integers. We remark that this will not be the case for all models and, therefore, we may obtain a function like this but which is not a Fourier series. In all cases, the 'frequencies' ω are determined by the eigenvalues of the encoding Hamiltonian, and the coefficients are specified by the form of the trainable gates and the observable. Hence, the ansatz used is very relevant, because it is going to shape the accessible ω and the coefficients of the model. Another important remark is that in the derivation, there is no specification of what system are we working on. Therefore, it works for one or more qubits, qutrits or qudits. This is a significant theoretical result that helps us to understand the relevance of the ingredients we use to build a quantum model.

If we want to shape our accessible 'frequencies', i.e., the values of ω in Eq. (33), we can add a tunable parameter in our encoding gate, such that $S(\alpha, x) = e^{ix\alpha H/2}$, where α is different in each layer. Now, the accessible ω are modified in the following way

$$\Omega'_{\boldsymbol{\alpha}} = \{\alpha_1(\lambda_{k_1} - \lambda_{j_1}) + ... + \alpha_L(\lambda_{k_L} - \lambda_{j_L})\}. \tag{12}$$

Hence, we have access to more 'frequencies' but we lose the possible structure of a Fourier series.

3.1 One qubit function fitting

Here we show that, with one qubit, we always obtain an integer frequency spectrum $\Omega \in \mathbb{K}$, if we use $H = \exp(i\sigma_i/2)$ on the encoding gate, being σ_i any of the Pauli matrices. The eigenvalues of these Hamiltonians are $\frac{1}{2}$ and $-\frac{1}{2}$. Looking at Eq. (10), we can see that the spectrum will be composed by the difference between L terms and other L terms, with possible values $\frac{1}{2}$ and $-\frac{1}{2}$. This gives a set of integer numbers. For instance, let us explore a simple case, such as a single qubit with $L = 2$. The accessible frequencies will be $\Omega_{\text{qub}} = \{-2, -1, 0, +1, +2\}$. In general, we can say that with the one-qubit model we obtain Fourier series or, equivalently, a Laurent polynomial of degree at least 2

The trainable gates and the observable determine the coefficient of the Fourier series, and the eigenvalues of the encoding Hamiltonian shape the frequency spectrum. An illustration of this is shown in Fig. 4. If we force the coefficients to be real,i.e., we use trainable gates with only real entries, the quantum model will only be able to generate Fourier series with real-valued coefficients. Having this in mind, we are going to choose trainable gates

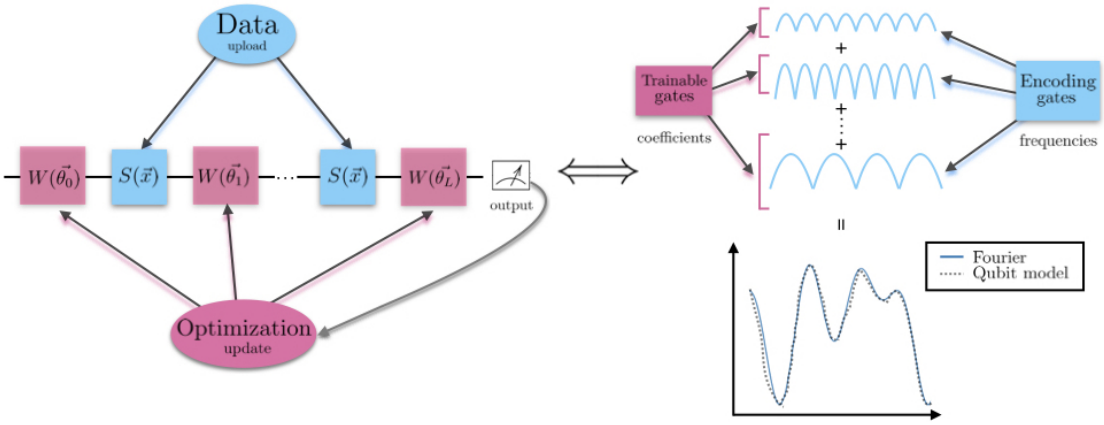


Figure 4: Schematic diagram of a one-qubit QNN that learns Fourier series. The circuit architecture is composed by the encoding blocks in pink and the trainable blocks in blue. The first ones shape the accessible frequencies of the model and the second type determine the values of the coefficient. (Figure inspired from [YYLW22]).

that have complex-valued entries, in order to have the most general possible model. Let us consider the following ansatz:

$$U^{WZW}_{\boldsymbol{\theta},\boldsymbol{\phi},x} = W(\theta_L, \phi_L)R_z(x)W(\theta_{L-1}, \phi_{L-1})...R_z(x)W(\theta_0, \phi_0), \tag{13}$$

with $W(\theta, \phi) = R_Y(\theta)R_z(\phi)$ and $S(x) = R_z(x)$, since it contain the terms $e^{ix/2}$ and $e^{-ix/2}$, similar to exponentials of the Fourier series. The observable we chose is the Pauli Z-gate, $M = \sigma_z$. The trainable blocks have the form:

$$W(\theta, \phi) = \begin{bmatrix} \cos(\theta/2)e^{-i\phi/2} & -\sin(\theta/2)e^{i\phi/2} \\ \sin(\theta/2)e^{-i\phi/2} & \cos(\theta/2)e^{i\phi/2} \end{bmatrix}. \tag{14}$$

With this choice, the model is able to reproduce functions with real and complex coefficients. Therefore, it is important to have the $R_Z(\phi)$ gate, because it is the one that introduces complex phases.

Now, we are going to show that any the output of the quantum model can be expressed as a Laurent polynomial with complex coefficients. The following lemma was inspired by Ref. [YYLW22].

Lemma 3.1. *There exists* $\boldsymbol{\theta} = (\theta_0, \theta_1, ...\theta_L) \in \mathbb{R}^{L+1}$ *and* $\boldsymbol{\phi} = (\phi_0, \phi_1, ...\phi_L) \in \mathbb{R}^{L+1}$ *such that*

$$U^{WZW}_{\boldsymbol{\theta}\boldsymbol{\phi},L} = \begin{bmatrix} P & -Q \\ Q^* & P^* \end{bmatrix}, \tag{15}$$

where P, Q *have the structure of a Laurent polynomial* $\in \mathbb{C}[e^{ix/2}, e^{-ix/2}]$ *until degree* $L-1$.

Lemma 3.2. *The output of the quantum model* T *is a Laurent polynomial of degree, at most,* L, *given by*

$$T = \langle 0 | U^{WZW\dagger}_{\boldsymbol{\theta}\boldsymbol{\phi},L} \sigma_z U^{WZW}_{\boldsymbol{\theta}\boldsymbol{\phi},L} | 0 \rangle . \tag{16}$$

This lemma 3.2 states that with this ansatz, we can reproduce any Laurent polynomial if we use enough layers. Since any square-integrable function can be approximated by these functions, we can say that the quantum model can, consequently, approximate any square-integrable function. The complete proof of these lemmas is in Appendix D.

Here we explore the single model with the real and complex coefficient ansatz and the qutrit model. We show the results in Fig. 5. We have used the Nelder-Mead method as a classical optimizer subroutine and 100 training and testing points. First, we present the cost function that we use to encode the solution to our problem, which will be the same for all the models used for function fitting:

$$C(\boldsymbol{\theta}) = \sum_{i=1}^{M} (g(x) - f_{\boldsymbol{\theta}}(x))^2, \tag{17}$$

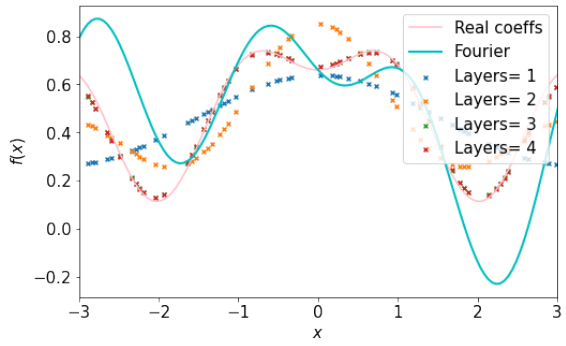
where M is the total number of training points and $g(x)$ is the function we try to reproduce. By minimizing the cost function, the model will be as similar as possible to the function for all data points, x. We use as target function the Fourier series $g(x) = \sum_{n=-3}^{3} c_n e^{ixn}$ with $c_0 = 0.24$ and $c_1 = c_2^* = -c_3^* = 0.09 + 0.09i$. First, we study the performance of an ansatz that contains only real-valued entries in the trainable gates, $W(\theta) = R_Y(\theta)$. The encoding gate we have chosen is $R_Z(x)$. Next, we study an ansatz with $W(\theta, \phi) = R_Y(\theta)R_Z(\phi)$ and the same encoding gate. Secondly, we can appreciate in Fig. 5a that the model can not approximate the full function, since the ansatz only has real coefficients and the function is complex, but it can clearly approximate the real part of the function, also plotted in the figure. On the other hand, in Fig. 5b we can see that the model achieves to fit the Fourier series of degree 3 with only three layers. Lemma 3.2 states that with this ansatz we can reproduce any Fourier series of degree, at most, L. In the case studied, this is accomplished. Moreover, the model is not only able to fit testing points in the same domain that in the testing, but it is also able to fit the function in any domain (see Fig. 5c). This evidence supports the idea that this quantum model can fit naturally a Fourier series. Finally, we can see that, in both cases, the use of more layers is directly related to the number of harmonics combined to create the function. With only one layer, we can generate only one harmonic, i.e., the combination of a trigonometric function with only one frequency. With two layers, we achieve to generate the combination of two of them and so on.

3.2 The single-qutrit model

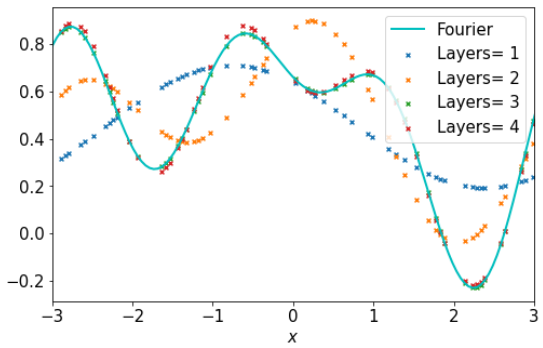
In this subsection, we use a qutrit quantum model and we study the type of functions generated. We explore if, in this case, using a larger system to store quantum information, the qutrit can improve the performance of the method. Lets take the ansatz

$$\begin{aligned} W(\theta_1, \theta_2, \phi_1, \phi_2) &= R_y^{(01)}(\theta_1) R_z^{(02)}(\phi_1) R_y^{(02)}(\theta_2) R_z^{(01)}(\phi_1) \\ S_3(x) &= R_z^{(01)}(x) R_z^{(02)}(-x) = \mathrm{diag}(1, e^{ix/2}, e^{-ix/2}). \end{aligned} \tag{18}$$

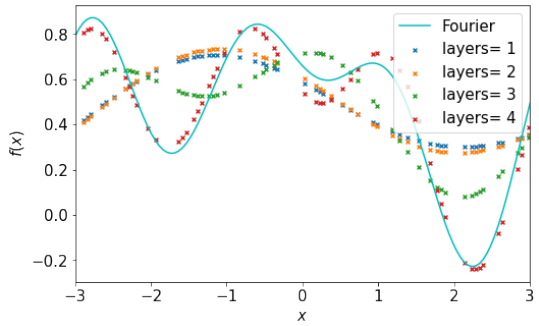
where we have used the single-qutrit rotation gate (explained in Appendix A. To construct the quantum model in Eq. (4), the observable used is $O = U_z$, defined as $U_z = (-\lambda_3 + \sqrt{3}\lambda_8)/2 = \mathrm{diag}(0, 1, -1)$. This encoding is inspired by the qubit ansatz used in the previous section. There is no formal proof of this ansatz being better than another one. Nevertheless, we argue that this could be a convenient one, following the ansatz used on the one-qubit section. First of all, we have seen that the trainable gates for one qubit that are sufficient to reproduce any Fourier series are $R_Y(\theta)R_Z(\phi)$. Therefore, we have decided to use a similar encoding with qutrits. With higher dimensional systems, we can not rotate



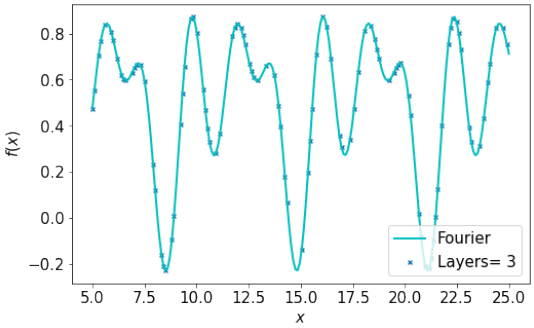
(a) One-qubit quantum model using $R_y(\theta)$ as trainable gate.



(b) One-qubit quantum model using $R_y(\theta)R_Z(\phi)$ as trainable gates.



(c) One-qutrit quantum model using $R_y^{(01)}(\theta_1)R_z^{(02)}(\phi_1)R_y^{(02)}(\theta_2)R_z^{(01)}(\phi_1)$ as trainable gates and $S(x) = R_z^{(01)}(x)R_z^{(02)}(-x)$ as encoding gates.

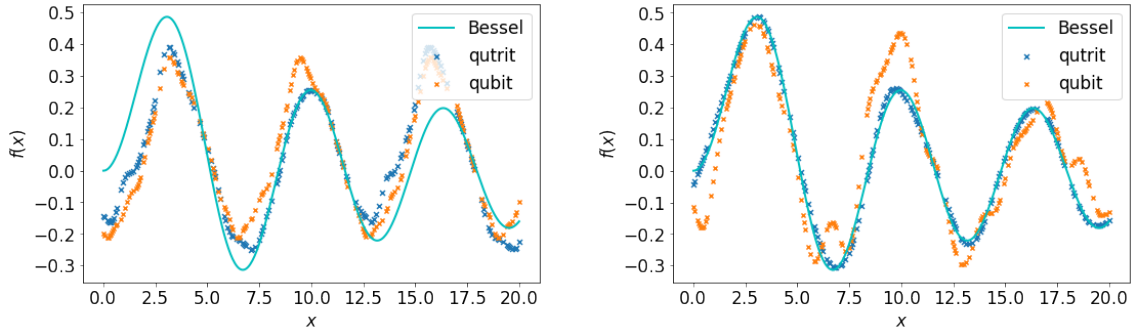


(d) Test set for a not trained domain with the ansatz used in Fig. (b).

Figure 5: Comparison of the quantum models for function fitting. The Fourier series represented in a solid blue line is $g(x) = \sum_{n=-3}^{3} c_n e^{ixn}$, i.e., of degree 3 and with $c_0 = 0.24$ and $c_1 = c_2^* = -c_3^* = 0.09+0.09i$. In the upper-left figure, we plot the results obtained with a model with real coefficients for the various layers. The solid pink line in the first figure is the same Fourier series but with real coefficients: $\sum_{n=-3}^{3} |c_n| e^{ixn}$. In the upper-right figure, we show a model with complex tunable parameters and the performance of the model. The lower-left figure is a model with qutrits. Finally, the lower-right figure is the model with complex tunable gates extended to a domain different from the training.

along one axis. Actually, rotations are performed while maintaining a plane constant. This makes the illusion of rotating along one single axis in 3-dimensiona. Now we are in a bigger subspace, and we are going to rotate according to R_Y and R_Z qubits gates in 2-dimensional subspaces of the qutrit. According to what we have seen in previous sections, we are interested in the eigenvalues of the Hamiltonian that generates this encoding gate $S_3(x)$. The corresponding Hamiltonian is $H = \mathrm{diag}(0, +1/2, -1/2)$. The only difference regarding the qubit encoding gate used in this work, which has eigenvalues $+1/2$ and $-1/2$, is that now we have an extra eigenvalue: 0. If we go back to Eq. (10), we can see that the 'frequencies' that the model has access to are given by the difference of all N^L possible eigenvalues. Let us study the possible values for qubits and qutrits, for example for $L = 2$ with the ansatz discussed for them:

$$\begin{aligned} \Omega_{\text{qub}} &= \{ -2, -1, 0, +1, +2\} \\ \Omega'_{\text{qut}} &= \{ -2, -3/2, -1, -1/2, 0, +1/2, +1, +3/2, +2\}, \end{aligned} \tag{19}$$



(a) Accuracy obtained with the qutrit (5 layers, 24 parameters): 69.22%, and with the qubit(10 layers, 24 parameters): 58.21%.

(b) Accuracy obtained with the qutrit (5 layers, 34 parameters): 98.21%, and with the qubit (10 layers, 32 parameters): 77.24%.

Figure 6: Comparison of the quantum models using different ansatz for fitting the spherical Bessel function of second order. In the left, we have the ansatzs for the qutrit and the qubit without parameters in the encoding gates. In the right, we include the parameters α in the encoding gates. The qutrit encoding gate is $R_Z^{(01)}(\omega_1 x)R_Z^{(02)}(-\omega_2 x)$ and for the qutrit we use $S(x) = R_Z(wx)$. The trainable gate for both qutrit models is $R_Y^{(01)}(\theta_1)R_Z^{(02)}(\phi_1)R_Y^{(02)}(\theta_2)R_Z^{(02)}(\phi_2)$, while for the qubit we have chosen $R_Y(\theta)R_Z(\phi)$. We have used 200 samples for the training set and 200 for the test. The optimization subroutine used is the Nelder-Mead method.

where we have combined the differences which contribute in each layer. With qubits, we have 2 possibilities for each eigenvalue while with qutrits we have 3 different eigenvalues to combine. Due to the extra eigenvalue, we have more accessible 'frequencies'. To be precise, with qubits we can achieve $2L + 1$ different ω, while with qutrits $4L + 1$. With enough layers, with qutrits, we have access to double the frequencies.

As we can appreciate in Fig. 5c, it seems that the qutrit does not approximate the Laurent polynomial as good as the qubit, at least with this specific ansatz. Therefore, this is evidence that with this qutrit model, we do not generate directly Laurent polynomials, in contrast with what happens with the qubit ansatz discussed before. This is because, with this ansatz, we also have fractional 'frequencies' (see Eq. (19)) that are not allowed by a Fourier series. Consequently, the model struggles to cancel these terms and it becomes hard to fit a Fourier series. However, these extra terms will contribute to a better fitting of another type of function, as we will explore.

3.3 Beyond Fourier Series

As we have seen, a single qubit can approximate any Fourier series, i.e., it can approximate any continuous square-integrable function. However, the model has some difficulties when approximating functions that have very different structures compared with Fourier. In these cases, the model struggles to find the optimal parameters to reproduce properly the function with few layers. One example of a family of functions where this occurs is the Bessel functions. They are not periodic functions and usually, they are said to seem like oscillating functions that decay proportionally to $x^{-1/2}$. We test to approximate a Bessel function of degree 2, i.e., $g(x) = \left(\frac{3}{x^2} - 1\right)\frac{\sin x}{x} - \frac{3\cos x}{x}$. We have used the Nelder-Mead method as an optimization subroutine, using 200 points in the training and in the testing set.

As we can appreciate in Fig. 6a, the qubit model discussed before does not approximate

well the Bessel function of second order. We only achieve a 58.21% of accuracy. We would need more layers, and, hence, more parameters to approximate correctly this function. On the other hand, the qutrit model with the ansatz on Eq. (18), achieves a better fitting, even though it is still struggling in some parts. We achieve an accuracy of $69,22\%$, with the same number of parameters as in the qubit model. The reason why we achieve to fit better with the same number of parameters is that, as discussed before, with the qutrit model we also have access to fractional 'frequencies' that seem to be more appropriated to fit these type of functions. Note that with qutrits we have used 5 layers, instead of 10 used with qubits, in order to have the same number of parameters. Hence, even with less layers, the semi-integer ω provides more flexibility for the function fitting problem.

Another option to access more frequencies consists in changing the encoding strategy and introducing tunable parameters. We show in Fig 6b the results obtained for the qubit and qutrit model using this encoding. On one hand, for the single-qubit, the possible exponents that we will have now are given by Eq. (12). On the other hand, in the case of the single-qutrit, the 'frequencies' are slightly different, because we have used two encoding gates. We parameterize the encoding gates like: $R_z^{(01)}(\alpha_1^{(i)}x)R_z^{(02)}(-\alpha_2^{(i)}x)$, where i is the layer where the encoding gate is applied and α_1, α_2 are the tunable parameters. In this case, the accessible frequencies are given by $\Omega'_\alpha = \{(\lambda_{k_1}^{(\alpha^{(1)})} - \lambda_{j_1}^{(\alpha^{(1)})}) + ... + (\lambda_{k_L}^{(\alpha^{(L)})} - \lambda_{j_L}^{(\alpha^{(L)})})\}$, where each 3 eigenvalues on a layer i are given by $\lambda_{j_i}^{(\alpha^{(i)})} \in \{\frac{\alpha_1+\alpha_2}{2}, \frac{\alpha_1}{2}, \frac{\alpha_2}{2}\}$. We have even more freedom than with qubits, and this can be reflected in the high accuracy obtained (almost 99%) when fitting the second order Bessel function.

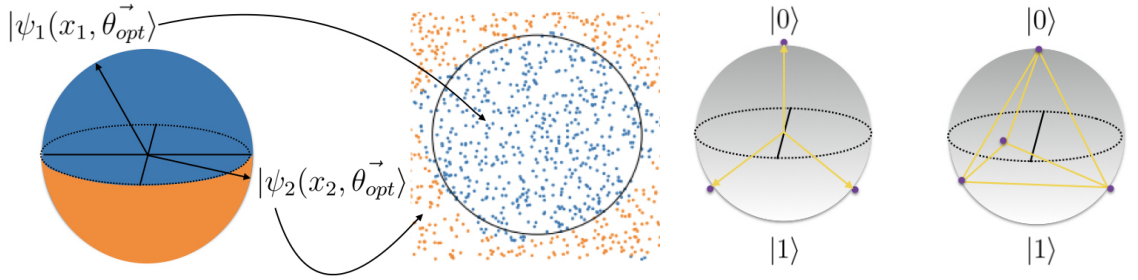
To sum up, we can appreciate that the performance of qutrits is better in the two studied cases for a Bessel function. Even when the number of layer used is less, with the same number of parameters the qutrit fits better the function. Then, there exists a trade-off: if the function has Fourier structure, qubits will be enough and will consume fewer resources. On the other hand, if the fitting we want to perform has a different structure from a Fourier series, then qutrits could be a better option, even using more resources. These results are relevant, because in NISQ devices, we want to reduce the number of resources, in order to avoid noise, decoherence and other source of errors. Using fewer layers means using fewer gates, hence, less computational time and less source of errors.

4 Classification

In this section, we address the problem of classification. The goal is, having a M-dimensional data set $\vec{x} = (x_1, ..., x_M)$, classify it into k different categories. We follow the same strategy as in the previous section: use a PQC and re-upload the data into it, assisted with an optimization classical subroutine. We aim to classify unseen data according to the criteria imposed once the optimal parameters are obtained after the training.

The criteria decided for classification is the following: each data point $\boldsymbol{x_i}$ pertains to a given category y_i, which is represented by a quantum state $|y_i\rangle$. The PQC generates an output quantum state $|\psi(\boldsymbol{\theta}, \boldsymbol{x_i})\rangle$. Our cost function will compute the distance between this state and $|y_i\rangle$. The criteria for assigning a certain category k to $\boldsymbol{x_i}$ is to chose among all the fidelities $F_k(\boldsymbol{x_i}, \boldsymbol{\theta}) = |\langle\psi(\boldsymbol{\theta}, \boldsymbol{x_i})|\, |y_k\rangle|^2$, the greatest one, say F'_k. Then, assign the category k' from y'_k to the data point x_i. The cost function that codifies the classification problem with this criteria is the following

$$f(\boldsymbol{\theta}) = \sum_{i=1}^{M} \left(1 - |\langle y_k\, |\psi(\boldsymbol{\theta}, x_i)\rangle|^2\right), \tag{20}$$



(a) Classification criteria for a qubit.

(b) Maximally orthogonal states of a qubit.

Figure 7: In the left figure we show a schematic representation of the classification into two categories $\{|y_k\rangle\} = \{|0\rangle, |1\rangle\}$ using a one-qubit classifier. The states located on the surface of the northern hemisphere are classified as points inside the circle. Otherwise, the states on the surface of the southern hemisphere, are classified as pertaining outside the circle. In the right figure we show the maximally orthogonal set of 3 and 4 states on the Bloch sphere. The purple points indicate the exact point on the surface of the Bloch sphere where the states are situated.

where M is the total number of training points, $|y_k\rangle$ is the known reference state assigned to each training point, $\boldsymbol{\theta}$ the set of parameters and $|\psi(\boldsymbol{\theta}, \vec{x_i})\rangle$ the state after the parameterized circuit. Our goal is to maximize the fidelity between the reference state and the output state, or, in other words, minimize the infidelity, i.e., one minus the fidelity.

For the specific case of a qubit, we can divide the Bloch sphere according to the categories $|y_k\rangle$ we have. A schematic representation is shown in Fig. 7a. As an example, imagine we have two categories to classify, for instance, the points inside or outside a circle. We may assign to each category the reference states $|0\rangle$ and $|1\rangle$, respectively. Then, if the final state $|\psi(\boldsymbol{\theta}, \mathbf{x})\rangle$ is closer to the zero-state, i.e., is it located in the northern hemisphere of the Bloch sphere, we will assign the data point $\mathbf{x}$ to the first category. The analogous reasoning applies if the state is closer to $|1\rangle$. Then, the data point will be assigned to the second category. Because operations in quantum mechanics are unitary and, therefore, reversible, we can not map more than one state into a single reference state. Instead, what happens is that we assign a defined region of the Bloch sphere to a category. If we had more than one category to classify, we would have to divide differently the regions on the Bloch sphere. An example for 3 and 4 categories division is given in Fig. 7b. The optimal strategy is to find a set of maximally orthogonal states. Then, we can assign to each category one of these reference states. The output of the quantum circuit which is closer to a specific target state will be associated with that category. Hence, the Bloch sphere will be divided according to the distance to the reference states. Even though this is a good strategy, it works better to use orthogonal states, as we will see.

We also study a 2-qubit classifier. Regarding the measurement strategy of this model, we have two options: In the first place, we can measure both two qubits and compute the cost function considering reference states of the Hilbert space of a two-qubit system. This cost function is called the global cost function. With this strategy, we would have 4 orthogonal states to use in the classification, instead of only 2 with a single qubit. The other strategy consists in focusing on the information contained in only one of the qubits. Then, we can perform the classification according to one qubit, i.e., like the method discussed for the single-qubit classifier. By building the cost function in this way, we are introducing local information about the 2-qubit state. This is why it is called local cost function. This local function is computed via measuring the two-qubit state, $|\psi(\boldsymbol{\theta}, \mathbf{x})\rangle$ and, then, computing the density matrix of one of the qubits, for instance, the second qubit. Therefore, we trace

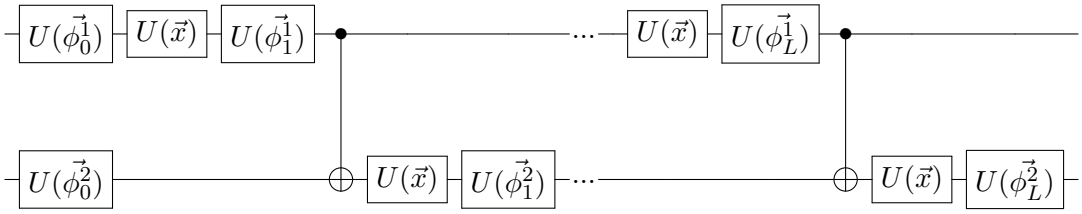


Figure 8: Representation of the circuit ansatz of the 2-qubit classifier.

out the information of the first system: $\rho_2 = \mathrm{Tr}_1 |\psi(\boldsymbol{\theta}, x)\rangle \langle\psi(\boldsymbol{\theta}, x)|$. With this reduced density matrix, we can compute the fidelity between the reference states $|y_k\rangle$ and this, in general, mixed state $F(\psi_s, \rho_2) = \langle y_k| \rho_2 |y_k\rangle$. Finally, for the qutrit, we take advantage of the 3 orthonormal states and we use the three of them as reference states.

4.1 Ansatzs for the classifiers

Now that we have all the components, we can explore the single-qubit, 2-qubit, and qutrit classifier. The ansatzs used with qutrits and qubits are very similar to the ones used in the previous section. The ansatz used for one qubit will consist of the encoding gate given by

$$S(\vec{x}) = U(\vec{x}) = R_z(x_1)R_y(x_2)R_z(x_3), \tag{21}$$

where U is a general rotation of a qubit. If the data is $2-$dimensional, we simply set x_3 to zero. For the trainable gates we also use $W(\vec{\theta}) = U(\vec{\theta})$.

For the qutrit, we use the following blocks:

$$W_{\mathrm{qut}}(\vec{\theta}) = R_y^{(01)}(\theta_1)R_y^{(02)}(\theta_2)R_z^{(01)}(\theta_3), \qquad S_{\mathrm{qut}}(\vec{x}) = R_y^{(01)}(x_1)R_y^{(02)}(x_2). \tag{22}$$

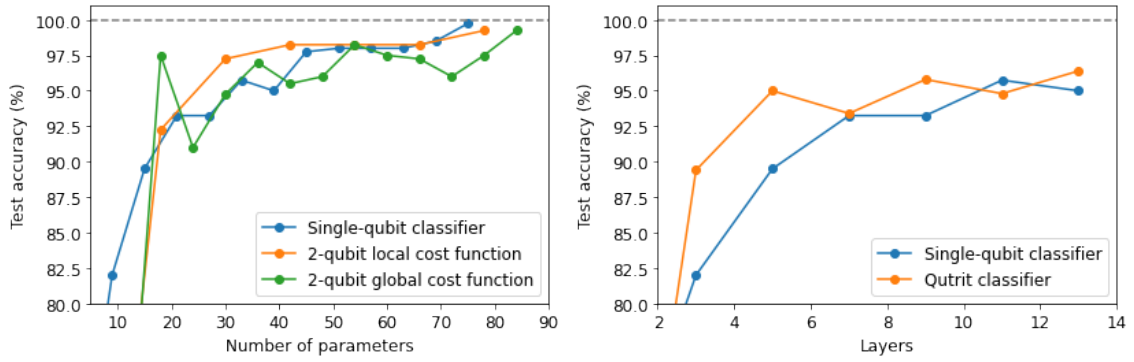
The encoding and training gates are inspired by the general unitary of a qutrit, from Eq. (3). However, we do not use the 8 parameters, we use only 3. In this case, we do not use rotations on the z axis for the encoding gates. In principle, this does not affect the performance of this problem. As we argued in Eq. (7), the only thing which contributes to the frequency spectrum that the model has access to is the eigenvalues of the Hamiltonian of the encoding gate. These values are the same for all single-qutrit gates we use. By using this ansatz, we have the same number of trainable and encoding gates as in the single-qubit model. Consequently, the two models use the same number of parameters per layer. This allows us to compare the methods properly.

Finally, for the 2-qubit ansatz, we have to introduce entanglement, in order to have more general unitary transformations. In Fig. 8 it is shown how we decided to introduce entanglement via CNOT gates in the middle of each layer.

4.2 Benchmarks

In this subsection, we simulate the models with the ansatzs discussed previously. We have used the 'L-BFGS-B' method as a classical subroutine to optimize the cost function. First, we present the results of the classification of a circle using the qubit classifier and the two-qubit with a local and global cost function. Secondly, we compare the qubit and qutrit classifier for the classification of two circles, with three different categories.

The results obtained for the classification of a circle are shown in In Fig. 9a. We have included the testing accuracy obtained with a different number of layers for the three models: one-qubit classifier and two-qubit classifier with local and global cost functions.



(a) Circle classification

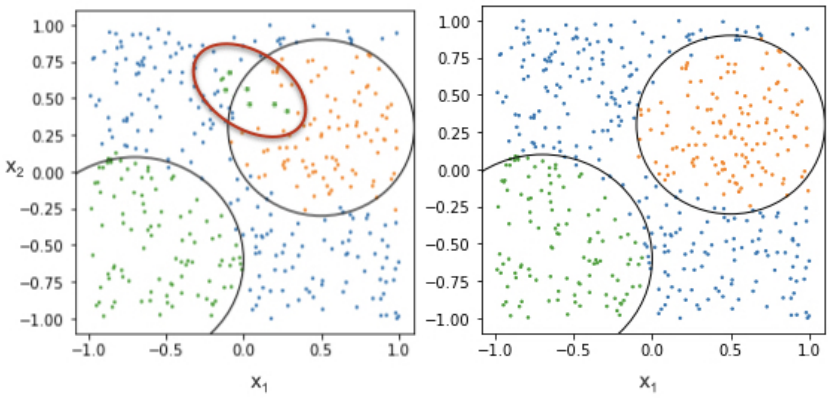
(b) Two circle classification.

Figure 9: On the left, we plot the accuracy in front of the number of parameters used for the circle classification problem with the single-qubit and the two-qubit with global and local cost function models. On the right figure, we compare the single-qubit classifier and the qutrit classifier. Both models have the same number of parameters per layer. The classical optimization method used is L-BFGS-B, with 100 training and testing data points.

Since we want to compare the efficiency of the model, we have represented the accuracy in comparison to the total number of parameters used, because both two-qubit models use double parameters and gates per layer compared with the single-qubit classifier. On one hand, the model with a global cost function performs better for fewer numbers of parameters. We can appreciate a big gap around 18 parameters. This may be caused because in this regime, it gains more importance the fact of having 4 possible orthogonal states for classification. On the other hand, for a greater number of parameters, the method with a local cost function performs better. This could be because this method presents two advantages: in the first place, there is evidence that using a local cost function avoids partially the Barren-Plateaus (BP) effect [UB21], as we have argued before. This effect may be manifesting for the two-qubit model with a large number of parameters and we may be avoiding it partially with the local cost function. Secondly, gathering the information of only one of the qubits will give us as a result, in general, a mixed state. Mixed states live inside the Bloch sphere, then we can map points not only on the surface but also in the interior. Hence, we have bigger regions on the Hilbert space which can be assigned to a certain category. Regarding the entanglement introduced in the two-qubit models, there is still not enough evidence to claim that it provides an advantage. However, we have seen that local cost functions do provide a certain improvement and we need entanglement to build such types of functions.

In Fig. 9b we compare the single-qubit and the qutrit classifiers for a classification problem with 3 categories. For the qubit model, we have used the three maximally orthogonal states illustrated in Fig. 7b. On the contrary, for the qutrit model, we have used the three orthogonal states of the computational basis: $\{|0\rangle, |1\rangle, |2\rangle\}$. As we discussed in the previous section, we have used the same number of parameters and gates per layer.

We can see that the qutrit model obtains better test accuracy results in almost all regimes of layers. With the qutrit classifier, we are only using 3 parameters among the 8 possible parameters that have a general unitary transformation. This could mean that the extra orthogonal state of the qutrit contributes to providing an advantage with respect to the qubit model. Similar results are observed in the fitting function section: the qutrit is able to reproduce function with more structure. The same applies to classification because fitting more complicated functions is equivalent to delimiting regions with more



(a) Qubit model (b) Qutrit model

Figure 10: In the figure on the left we have the results for the qubit model, where we have used 13 layers and we obtained an accuracy of 95.00%. The figure on the left, shows the results for the qutrit model with 11 layers. The accuracy obtained was 94.80%.

structure. Despite the ansatzs used in both problems are not the same and, hence, we can not establish a direct relation between both models, the general properties that we have discussed still apply for this section. Moreover, with the qutrit classifier, we can choose orthogonal reference states, while with the qubit we have to work with maximally orthogonal states. This undoubtedly provides an advantage to the qutrit model, since the space assigned to each category is larger in the qutrit Hilbert space.

Another interesting thing that we observe is that, with a similar accuracy obtained in both qubit and qutrit models, the qubit model does not capture the frontiers of the classification as well as the qutrit model. In Fig 10a it can be appreciated that there is a region, marked with a red circumference, where points are not well classified. In principle, it seems that the green points are assigned to the lower-left circle. However, there are some points (x_1, x_2) that do not pertain to this circle but are classified as if they were. This may be caused because of the division of the Bloch sphere, where two or more regions assigned to a certain category may overlap. As we can see in Fig. 10b, with qutrits we do not have this issue: the model captures well the structure of the problem. These results on the qubit model may be because regions are better divided in this case or as a consequence of having more coefficients in the exponentials, and thus more freedom to create delimiting regions. It is reasonable that the model confuses a point on the border of the circle with a point outside a circle. However, assigning categories to points that are far away from that category may be a more significant error.

Code and data availability

All the code used in this work can be found on the GitHub repository https://github.com/bertacasas/TFM_Berta_Casas.

5 Conclusions

In this work, we have studied a model of QNN based on the re-uploading of the data in the quantum circuit. We have introduced the use of a higher-dimensional system, the qutrit,

into these quantum models. We have explored the advantages and the drawbacks when we use them. In particular, we have studied two problems: function fitting and classification.

For the function fitting problem, we have seen, on the one hand, that a qubit is able to reproduce any Fourier series if we use enough layers. On the other hand, the qutrit model with the specific ansatz used creates a function with fractional exponents. Because of the semi-integer exponents, the function generated is not a Fourier series anymore. Despite this, the extra accessible exponents enable to fit other functions with fewer layers. Since we can simulate a qubit with a qutrit, choosing the appropriate ansatz that only operates on a 2-level subspace of a qutrit, we will recover the results obtained for a qubit and we will be able to generate Fourier series.

In the classification section, we also appreciate a better performance of the qutrit model in comparison to the qubit model. With qutrits, we not only obtain better accuracy, but we also appreciate better learnability even when the accuracy is the same in both qubit and qutrit models. Moreover, we study the 2-qubit classifier, using both local and global cost functions. We observe that the 2-qubit classifier with a local cost function obtains a better accuracy, even compared with the single-qubit model.

We have seen that the use of the qutrit provides an advantage in simple QNN models. This may indicate that, for more sophisticated models, it can also be beneficial to use high-dimensional systems. Moreover, it is still not characterized if qutrit entanglement would play a significant role when used in these models. This question remains unanswered and more work has to be done in this direction. As we have argued, we believe that qutrits may be useful for noisy and fault-tolerant current devices. Hence, another possible test for our hypothesis is to try these models in a real quantum computer. If this was done, we would have to think of a quantum tomography protocol to fully determine the state of the qutrit. Other things that have not been considered are the training errors, caused by the minimization subroutine. In the future, we should characterize this for knowing exactly which errors come from the classical subroutine and which come from having noisy quantum devices. In addition, we have seen the importance of the cost function for obtaining good performance of the model. Then, more work can be done in characterizing the properties of cost functions with distributions coming from a quantum computer.

Bibliography

[Ben80] Paul Benioff. The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *Journal of Statistical Physics*, 22(5):563–591, May 1980.

[BRS+21] M. S. Blok, V. V. Ramasesh, T. Schuster, K. O'Brien, J. M. Kreikebaum, D. Dahlen, A. Morvan, B. Yoshida, N. Y. Yao, and I. Siddiqi. Quantum information scrambling on a superconducting qutrit processor. *Phys. Rev. X*, 11:021010, Apr 2021.

[BWP+17] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, sep 2017.

[CAB+21] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. *Nature Reviews Physics*, 3(9):625–644, aug 2021.

[CHI+18] Carlo Ciliberto, Mark Herbster, Alessandro Davide Ialongo, Massimiliano Pontil, Andrea Rocchetto, Simone Severini, and Leonard Woss-

nig. Quantum machine learning: a classical perspective. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2209):20170551, jan 2018.

[CLKAGG22] Alba Cervera-Lierta, Mario Krenn, Alán Aspuru-Guzik, and Alexey Galda. Experimental high-dimensional greenberger-horne-zeilinger entanglement with superconducting transmon qutrits. *Phys. Rev. Applied*, 17:024062, Feb 2022.

[CM00] Carlton M. Caves and Gerard J. Milburn. Qutrit entanglement. *Optics Communications*, 179(1-6):439–446, may 2000.

[DN05] Christopher M. Dawson and Michael A. Nielsen. The solovay-kitaev algorithm, 2005.

[DW11] Yao-Min Di and Hai-Rui Wei. Elementary gates for ternary quantum logic circuit, 2011.

[Fey82] Richard P Feynman. Simulating physics with computers. *International journal of theoretical physics*, 21(6/7):467–488, 1982.

[FFR+21] Enrico Fontana, Nathan Fitzpatrick, David Muñ oz Ramo, Ross Duncan, and Ivan Rungger. Evaluating the noise resilience of variational quantum algorithms. *Physical Review A*, 104(2), aug 2021.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[GJV+06] Simon Gröblacher, Thomas Jennewein, Alipasha Vaziri, Gregor Weihs, and Anton Zeilinger. Experimental quantum cryptography with qutrits. *New Journal of Physics*, 8(5):75, 2006.

[Gus22] Erik Gustafson. Noise improvements in quantum simulations of sqed using qutrits, 2022.

[HCT+19] Vojtěch Havlíček, Antonio D. Córcoles, Kristan Temme, Aram W. Harrow, Abhinav Kandala, Jerry M. Chow, and Jay M. Gambetta. Supervised learning with quantum-enhanced feature spaces. *Nature*, 567(7747):209–212, mar 2019.

[HNPR86] L. Herrera, Luis Nunez, Alberto Patiño, and Héctor Rago. A variational principle and the classical and quantum mechanics of the damped harmonic oscillator. *American Journal of Physics*, 54:273–277, 03 1986.

[JGR20] Tammy Jiang, Jaimie L Gradus, and Anthony J Rosellini. Supervised machine learning: a brief primer. *Behavior Therapy*, 51(5):675–687, 2020.

[KBKH+22] Alba Cervera-Lierta Kishor Bharti, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek, and Alá n Aspuru-Guzik. Noisy intermediate-scale quantum algorithms. *Reviews of Modern Physics*, 94(1), feb 2022.

[MBS+18] Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9(1), nov 2018.

[Per14] McClean J.-Shadbolt P. *et al.* Peruzzo, A. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5, 2014.

[Pre18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, August 2018.

[PSCLGFL20] Adrián Pérez-Salinas, Alba Cervera-Lierta, Elies Gil-Fuster, and José I. Latorre. Data re-uploading for a universal quantum classifier. *Quantum*, 4:226, February 2020.

[RMP+21] Martin Ringbauer, Michael Meth, Lukas Postler, Roman Stricker, Rainer Blatt, Philipp Schindler, and Thomas Monz. A universal qudit quantum processor with trapped ions, 2021.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[Sho94] P.W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

[SK19] Maria Schuld and Nathan Killoran. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.*, 122:040504, Feb 2019.

[SSM21] Maria Schuld, Ryan Sweke, and Johannes Jakob Meyer. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A*, 103:032430, Mar 2021.

[UB21] A V Uvarov and J D Biamonte. On barren plateaus and cost function locality in variational quantum algorithms. *Journal of Physics A: Mathematical and Theoretical*, 54(24):245301, may 2021.

[YSK+20] M. A. Yurtalan, J. Shi, M. Kononenko, A. Lupascu, and S. Ashhab. Implementation of a walsh-hadamard gate in a superconducting qutrit. *Phys. Rev. Lett.*, 125:180504, Oct 2020.

[YYLW22] Zhan Yu, Hongshun Yao, Mujin Li, and Xin Wang. Power and limitations of single-qubit native quantum neural networks, 2022.

A Quantum gates

In this appendix, we show the quantum gates that we will be using in this work, both for qubits and qutrits. First, for qubits, we introduce the Pauli matrices, which are a set of 2-dimensional Hermitian, unitary and traceless matrices. They are given by

$$\begin{aligned} \sigma_x &= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \\ \sigma_y &= \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \\ \sigma_z &= \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \end{aligned} \tag{23}$$

The Pauli matrices are the generators of SU(2). By exponentiation, they generate the rotations of a qubit:

$$\begin{aligned} R_x =& e^{i\theta\sigma_x/2} = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \\ R_y =& e^{i\theta\sigma_y/2} = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \\ R_z =& e^{i\theta\sigma_z/2} = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}. \end{aligned} \tag{24}$$

We also introduce the CNOT gate, a 2-qubit gate which is given by

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{25}$$

This gate acts on the qubits: the control qubit and the target qubit. If the control qubit is in the 0 state, then the target qubit remains unchanged, while if the control qubit is in the state 1, it is applied a X gate to the target qubit. A X gate is basically a swap gate. Its summarized by $X|0\rangle = |1\rangle$ and $X|1\rangle = |0\rangle$. Then, another expression of this gate can be:

$$CNOT = |0\rangle\langle 0| \otimes \mathbb{I} + |0\rangle\langle 0| \otimes X. \tag{26}$$

The CNOT gate is widely used to create entanglement. The graphic representation of this gate is shown in Fig. 11.

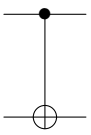


Figure 11: Graphic representation of the symbol of a CNOT used in a quantum circuit.

Now we introduce the quantum gates for qutrits. First, we present the eight Gell-Mann matrices, which are the generators of SU(3), the special unitary group of dimension 3. They

are given by

$$\lambda_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \lambda_2 = \begin{bmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \lambda_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$
$$\lambda_4 = \begin{bmatrix} 0 & & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad \lambda_5 = \begin{bmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{bmatrix} \qquad \lambda_6 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{27}$$
$$\lambda_7 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{bmatrix} \qquad \lambda_8 = \frac{1}{\sqrt{3}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{bmatrix}.$$

Any 1-qutrit operation can be generated by $U = \exp(i \sum_j a_j \lambda_j)$, where a_j are arbitrary real parameter. By combining Kronecker products of the generators we can obtain any $n-$qutrit gate. By exponentiation, they generate the possible rotations of two levels on a qutrit:

$$R_x^{(01)} = e^{i\theta\lambda_1/2} = \begin{bmatrix} \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} & 0 \\ -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad R_y^{(01)} = e^{i\theta\lambda_2/2} = \begin{bmatrix} \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} & 0 \\ \sin\frac{\theta}{2} & \cos\frac{\theta}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$R_z^{(01)} = e^{i\theta\lambda_3/2} = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 \\ 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad R_x^{(02)} = e^{i\theta\lambda_4/2} = \begin{bmatrix} \cos\frac{\theta}{2} & 0 & -i\sin\frac{\theta}{2} \\ 0 & 1 & 0 \\ -i\sin\frac{\theta}{2} & 0 & \cos\frac{\theta}{2} \end{bmatrix}$$
$$R_y^{(02)} = e^{i\theta\lambda_5/2} = \begin{bmatrix} \cos\frac{\theta}{2} & 0 & -\sin\frac{\theta}{2} \\ 0 & 1 & 0 \\ \sin\frac{\theta}{2} & 0 & \cos\frac{\theta}{2} \end{bmatrix} \qquad R_z^{(02)} = e^{i\theta(\lambda_3+\sqrt{3}\lambda_8)/4} = \begin{bmatrix} e^{-i\frac{\theta}{2}} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix}$$
$$R_x^{(12)} = e^{i\theta\lambda_6/2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\frac{\theta}{2} & -i\sin\frac{\theta}{2} \\ 0 & -i\sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix} \qquad R_y^{(12)} = e^{i\theta\lambda_7/2} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\frac{\theta}{2} & -\sin\frac{\theta}{2} \\ 0 & \sin\frac{\theta}{2} & \cos\frac{\theta}{2} \end{bmatrix}$$
$$R_z^{(02)} = e^{i\theta(-\lambda_3+\sqrt{3}\lambda_8)/4} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & e^{-i\frac{\theta}{2}} & 0 \\ 0 & 0 & e^{i\frac{\theta}{2}} \end{bmatrix} \tag{28}$$

The superindices (ij) with $i > j$ indicate that the rotation is made in the i and j energy levels subspace. Essentially, what this means is that we are performing one-qubit rotations but only on the subspace of the i and j levels (subspace that can be thought of as a qubit), remaining the other level untouched.

There is a generalization of the CNOT gate for the qutrit, the ternary controlled-X gate. First, we introduce the X-qutrit gates. They are different from the qubit X-gate since with three levels we have to specify what is a swap. The three quantum X gates are

given by

$$X^{(01)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
$$X^{(02)} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{29}$$
$$X^{(12)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

These gates act on 2 subspaces of a qutrit and perform a swap between two of the levels. We define the Ternary Controlled-X gate (TCX) as follows:

$$TCX = |0\rangle\langle 0| \otimes X^{(01)} + |1\rangle\langle 1| \otimes X^{(02)} + |2\rangle\langle 2| \otimes X^{(12)}. \tag{30}$$

Hence, if the control qubit is in the zero state, we apply $X^{(01)}$ to the target qutrit, if it is in the state 1 we apply $X^{(12)}$ and if it is in the state 2 we apply $X^{(02)}$.

B Fourier series and Laurent polynomial definition

Here we provide the definition of both the Fourier series and the Laurent polynomials. First, let's start with the Fourier series:

Definition 1. *A truncated Fourier series is an expansion of a periodic real-valued function, that can be expressed as a sum of sines and cosines. In the exponential form, it is given by:*

$$g(x) = \sum_{l=-M}^{M} c_l e^{i\frac{\pi}{T}l}, \tag{31}$$

where T is half of a period of the function $g(x)$ and $l\frac{\pi}{T}$ are the multiples of the fundamental frequency $\frac{\pi}{T}$. $\{l\frac{\pi}{T}\}_l$ form the frequency spectrum. The coefficients $c_l \in \mathbb{C}$, fulfill $c_l = c^*_{-l}$. A Fourier series with enough large degree can approximate any continuous and square-integrable function.

A truncated Fourier series can also be expressed as a Laurent Polynomial:

Definition 2. *Laurent Polynomial can be defined from a partial Fourier sum by substitution $z = e^{i\frac{\pi}{T}x}$:*

$$P(z) = \sum_{l\in\Omega} c_l z^l. \tag{32}$$

We define the degree of the Laurent Polynomial as the maximal absolute value of all z exponents. The parity of the polynomial is said to be 0 if we only have even coefficients, and to be 1 if all the coefficients correspond to odd l.

C Effect of data encoding

In this section we derive how we can express a given quantum model, based on re-uploading of the data, like

$$f_{\boldsymbol{\theta}}(x) = \sum_{\mathbf{k},\mathbf{j}\,\in[N]^L} a_{\mathbf{k},\mathbf{j}} e^{-ix(\Lambda_{\mathbf{k}}-\Lambda_{\mathbf{j}})}, \tag{33}$$

which is a truncated Fourier series if all the frequencies in the spectrum are integers and it is fulfilled that $c_n = c^*_{-n}$. First of all, we recall that the general quantum model that we are considering is given by

$$f_{\boldsymbol{\theta}}(x) = \langle 0| U^\dagger(x,\boldsymbol{\theta}) O U(x,\boldsymbol{\theta}) |0\rangle = \langle \psi(x,\boldsymbol{\theta})| O |\psi(x,\boldsymbol{\theta})\rangle \tag{34}$$

where O is some arbitrary observable, and $U(x,\boldsymbol{\theta})$ an unitary operation given by

$$U(x,\boldsymbol{\theta}) = W^{(L)} S(x) W^{(L-1)} ... W^{(1)} S(x) W^{(0)}, \tag{35}$$

where L is the number of layers, $W^{(i)}$ and $S(x) = e^{ixH}$ are the trainable gate at the layer i and the encoding gate, respectively. As we argued, we are free to consider a diagonal Hamiltonian and this is why we are going to assume: $H = \text{diag}(\lambda_1, ..., \lambda_d)$. First of all, let us consider the state after the parameterized quantum circuit

$$U(x)|0\rangle = |\psi(x,\boldsymbol{\theta})\rangle = W^{(L)} e^{-ixH} W^{(L-1)} ... W^{(1)} e^{-ixH} W^{(0)} |0\rangle, \tag{36}$$

We consider the i component of this state vector

$$\begin{aligned}\left[\, |\psi(x,\boldsymbol{\theta})\rangle \,\right]_i = \sum_{j_1,...,j_L=1}^{N} W^{(L)}_{i,j_L} e^{-ix\lambda_{j_L}} W^{(L-1)}_{j_L,j_{L-1}} ... W^{(1)}_{j_2,j_1} e^{-ix\lambda_{j_1}} W^{(0)}_{j_1,j_0} \underbrace{\delta_{j_0,1}}_{|0\rangle} = \\ \sum_{j_1,...,j_L=1}^{N} e^{-ix(\lambda_{j_1}+...+\lambda_{j_L})} W^{(L)}_{i,j_L} W^{(L-1)}_{j_L,j_{L-1}} ... W^{(1)}_{j_2,j_1} W^{(0)}_{j_1,1},\end{aligned} \tag{37}$$

where we have contracted the last trainable gate $W^{(0)}$ with the state $|0\rangle$, which corresponds to the index 1. The indices $j_1, ..., j_L$ run from 1 the dimension of the encoding Hamiltonian, $N = d^n$. Now we will introduce the multi-index notation $\mathbf{j} = \{j_1, ..., j_L\} \in [N]^L$. This means that each $\mathbf{j}$ is a possible combination of L numbers and each one can run from 1 to N. We can also define $\Lambda_j = \lambda_{j_1} + ... + \lambda_{j_L}$. This is a sum that has $|\mathbf{j}| = d^L$ possible values, given by all the possible combinations of λ_{j_i}. Then, we can re-express the i-vector state:

$$\left[\, |\psi(x,\boldsymbol{\theta})\rangle \,\right]_i = \sum_{\vec{j}\in[d]^L} e^{-ix\Lambda_j} W^{(L)}_{i,j_L} |j_L\rangle \langle j_L| W^{(L-1)}_{j_L,j_{L-1}} ... W^{(1)}_{j_2,j_1} |j_1\rangle \langle j_1| W^{(0}_{j_1,1}. \tag{38}$$

Finally, we can express the quantum model in Eq. (34):

$$\begin{aligned} f_{\boldsymbol{\theta}}(x) = \sum_{i,j} \left[\, \langle \psi(x,\boldsymbol{\theta})| \,\right]_i M_{i,j} \left[\, |\psi(x,\boldsymbol{\theta})\rangle \,\right]_j = \\ \sum_{k,j\in[d]^L} e^{-ix(\Lambda_k - \Lambda_j)} a_{k,j}, \end{aligned} \tag{39}$$

where we have grouped

$$a_{k,j} = \sum_{i,i'} (W^*)^{(0)}_{i,k_1} (W^*)^{(1)}_{k_1,k_2} ... (W^*)^{(L)}_{k_L,i} M_{i,i'} W^{(L)}_{i',j_L} ... W^{(1)}_{j_2,j_1} W^{(0)}_{j_1,1}. \tag{40}$$

Then, comparing Eq. (39) with the general structure of a Fourier series, we can see that the frequency spectrum is given by

$$\Omega = \{\Lambda_{\mathbf{k}} - \Lambda_{\mathbf{j}} \mid \mathbf{k}, \mathbf{j} \in [d]^L\}, \tag{41}$$

Only if the frequencies obtained are integers. Now, if we group all the terms that have the same frequency, i.e., $\omega = \Lambda_k - \Lambda_j$ we can re-express the quantum model

$$f_{\boldsymbol{\theta}}(x) = \sum_{\omega \in \Omega} c_\omega e^{-ix\omega}, \tag{42}$$

where we grouped the coefficients

$$c_\omega = \sum_{\substack{\mathbf{k},\, \mathbf{j} \in [d]^L \\ \Lambda_{\boldsymbol{k}} - \Lambda_{\mathbf{j}} = \omega}} a_{\mathbf{k},\, \mathbf{j}}. \tag{43}$$

This is simply a way of expressing that we have added up all the coefficients which have the same 'frequency' ω.

D Proof of Lemma 3.1 and 3.2

Lemma D.1. *There exists* $\boldsymbol{\theta} = (\theta_0, \theta_1, ...\theta_L) \in \mathbb{R}^{L+1}$ *such that*

$$U^{WZW}_{\boldsymbol{\theta\phi},L} = \begin{bmatrix} P & -Q \\ Q^* & P^* \end{bmatrix}, \tag{44}$$

where P, Q *have the structure of a Laurent polynomial* $\in \mathbb{R}[e^{ix/2}, e^{-ix/2}]$ *until degree* $L-1$.

Proof: First, let us show that P, Q have the structure of a Laurent Polynomial until degree $L-1$, meaning that they fulfill the property of Laurent Polynomials $c_l = c^*_{-l}$, with exception of the case $l = L$. Let's propose P, Q for L layers and then study the case for $L+1$ layers, without assuming anything in the form of the polynomials. We obtain the following result

$$\begin{aligned}
&\begin{bmatrix} P & -Q \\ Q^* & P^* \end{bmatrix} R_z(x) R_y(\theta_L) \\
&= \begin{bmatrix} P & -Q \\ Q^* & P^* \end{bmatrix} \begin{bmatrix} \cos\frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} & -\sin\frac{\theta_{L+1}}{2} e^{-i\frac{L+1}{2}} \\ \sin\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} & \cos\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} \end{bmatrix} \\
&= \begin{bmatrix} \cos\frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} P - \sin\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} Q^{-in} & -\sin\frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} P - \cos\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} Q \\ \cos\frac{\theta_{L+1}}{2} e^{-i\frac{\theta_{L+1}}{2}} e^{-i\frac{x}{2}} Q^* + \sin\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} P^* & \cos\frac{\theta_{L+1}}{2} e^{i\frac{jL+1}{2}} e^{i\frac{x}{2}} P^* - \sin\frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} Q^* \end{bmatrix}
\end{aligned} \tag{45}$$

Hence,

$$\begin{aligned}
P_{L+1} &= \frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} P - \sin\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} Q \\
Q_{L+1} &= \sin\frac{\theta_{L+1}}{2} e^{-i\frac{x}{2}} P + \cos\frac{\theta_{L+1}}{2} e^{i\frac{x}{2}} Q.
\end{aligned} \tag{46}$$

Assuming that $P=\sum_{l=-L}^{L} p_l e^{ixl}$ and $Q=\sum_{l=L}^{L} q_l e^{ixl}$, we can write

$$
\begin{aligned}
P_{L+1} &= \sum_{l=-\frac{L+1}{2}}^{\frac{L+1}{2}} \bar{p}_l e^{ixl}, \text{ with } \begin{cases} \bar{p}_{-\frac{L+1}{2}} = \cos\frac{\theta_{L+1}}{2} p_{-L/2} \\ \bar{p}_l = \cos\frac{\theta_{L+1}}{2} p_{\frac{2l+1}{2}} - \sin\frac{\theta_{L+1}}{2} q_{\frac{2l-1}{2}} \\ \bar{p}_{\frac{L+1}{2}} = -\sin\frac{\theta_{L+1}}{2} q_{L/2} \end{cases} \\
Q_{L+1} &= \sum_{l=-\frac{L+1}{2}}^{\frac{L+1}{2}} \bar{q}_l e^{ixl}, \text{ with } \begin{cases} \bar{q}_{-\frac{L+1}{2}} = \sin\frac{\theta_{L+1}}{2} p_{-L/2} \\ \bar{q}_l = \sin\frac{\theta_{L+1}}{2} p_{\frac{2l+1}{2}} + \cos\frac{\theta_{L+1}}{2} q_{\frac{2l-1}{2}} \\ \bar{q}_{\frac{L+1}{2}} = -\cos\frac{\theta_{L+1}}{2} q_{L/2}. \end{cases}
\end{aligned}
\tag{47}
$$

Let us check if $\bar{p}_l = \bar{p^*_{-l}}$ and $\bar{q}_l = \bar{q^*_{-l}}$ for $-L \leq l \leq L$. We obtain the two equations

$$
\begin{aligned}
\cos\frac{\theta_L}{2} p_{\frac{2l+1}{2}} - \sin\frac{\theta_L}{2} q_{\frac{2l+1}{2}} &= \cos\frac{\theta_L}{2} p_{-\frac{2l-1}{2}} - \sin\frac{\theta_L}{2} q_{\frac{2l+1}{2}} \\
\sin\frac{\theta_L}{2} p_{\frac{2l+1}{2}} - \cos\frac{\theta_L}{2} q_{\frac{2l-1}{2}} &= \sin\frac{\theta_L}{2} p_{-\frac{2l+1}{2}} - \cos\frac{\theta_L}{2} q_{-\frac{2l+1}{2}}.
\end{aligned}
\tag{48}
$$

With these two equations, we arrive at

$$
\cot(\frac{\theta_L}{2} + \frac{\pi}{4}) \frac{p_{\frac{2l+1}{2}} - p_{\frac{2l-1}{2}}}{q_{\frac{2l-1}{2}} - q_{-\frac{2l-1}{2}}} = 0 \tag{49}
$$

This equation is fulfilled if $\theta_L = \frac{1}{2}(4\pi n + \pi)$ with $n \in \mathbb{Z}$ or $p_{\frac{2l+1}{2}} = p_{\frac{2l-1}{2}}$. Then, there is always a solution that fulfils $p_l = p^*_{-l}$. For $l = \pm L/2$ in Eq. (48), we arrive at $\tan\frac{\theta_L}{2} = -\frac{p_{-\frac{L+1}{2}}}{q_{\frac{L-1}{2}}}$ and $\cot\frac{\theta_L}{2} = \frac{p_{-\frac{L-1}{2}}}{q_{\frac{L-1}{2}}}$, which is an incompatible system. Then, we have demonstrated that P_L , Q_L are not Laurent polynomials but they fulfill the conditions of a polynomial of this type until the last degree. We have demonstrated the real-coefficient case. The case with complex coefficients is analogous and it can be demonstrated following the same steps.

Lemma D.2. *The output of the quantum model T is a Laurent polynomial of degree, at most, L, given by*

$$
T = \langle 0 | U^{WZW\dagger}_{\boldsymbol{\theta\phi},L} \sigma_z U^{WZW}_{\boldsymbol{\theta\phi},L} | 0 \rangle . \tag{50}
$$

Proof: We demonstrate that T is, indeed, a Laurent polynomial. With the Polynomials defined in Eq. (47), we can write

$$
\begin{aligned}
D = |P_L|^2 - |Q_L|^2 &= \sum_{n,m=-L/2}^{L/2} (\bar{p_n}\bar{p_m}^* - \bar{q_n}\bar{q_m}^*) e^{ix(n-m)} = \\
&\sum_{\rho=-L}^{L} \sum_{n=-L/2}^{L/2} (\bar{p_n}\bar{p_{n-\rho}}^* - \bar{q_n}\bar{q_{n-\rho}}^*) e^{ix\rho} \equiv \sum_{\rho=_L}^{L} d_\rho e^{ix\rho},
\end{aligned}
\tag{51}
$$

where we defined $\rho = n - m$. Then, we have that $d_\rho = \sum_{n=-L/2}^{L/2} (\bar{p_n}\bar{p_{n-\rho}}^* - \bar{q_n}\bar{q_{n-\rho}})$. Let us see if $d_p = d^*_{-p}$ for all cases:

1. Case $n, n-\rho, n+\rho \neq \pm L/2$. Solving the system for this case, we obtain two possible solutions

$$\begin{aligned} \bar{p_n}(\bar{p}^*_{n-\rho} - \bar{p}_{n+\rho}) &= 0 \\ \bar{q_n}(\bar{q}^*_{n-\rho} - \bar{q}_{n+\rho}) &= 0. \end{aligned} \tag{52}$$

   If $\bar{p}_n = 0$, then, $\tan\frac{\theta_n}{2} = -\frac{p_{(2n+1)/2}}{q_{(2n-1)/2}}$. Also, we can have $\bar{p}^*_{n-\rho} = \bar{p}_{n+\rho}$, where there are values of θ that allow this to happen. Hence, seems that in general there is always possible to obtain $d_\rho = d^*_{-\rho}$. There are multiple solutions that allow it.

2. Case $n = L/2$ or $n \pm \rho \neq \pm L/2$.Then we have

$$\begin{aligned} \bar{p_n} = \bar{p_{L/2}} &= -\sin\frac{\theta_L}{2} q_{\frac{L-1}{2}} \\ \bar{q_n} = \bar{q_{L/2}} &= -\cos\frac{\theta_L}{2} q_{\frac{L-1}{2}}. \end{aligned} \tag{53}$$

   As before, we have the same possible solutions. $\bar{p}_{L/2} = 0$ or $q^*_{n-\rho} = q_{n+\rho}$.

3. Case $n - \rho = L/2$ and $n, \rho \neq \pm L/2$. We can find a solution by imposing $p_n = 0$ and $\bar{q}_n(\cos\frac{\theta_L}{2} q_{\frac{L-1}{2}} - \sin\frac{\theta_{n+\rho}}{2} p_{\frac{L-1}{2}} + \cos\frac{\theta_{n+\rho}}{2} q_{\frac{L-1}{2}} = 0$. This can be fulfilled if the second term of the multiplication is zero, which is it possible.

4. In the last case $\rho = 0$, all equations are fulfilled if p^*_n and $q^*_n = q_n$

Then, we can say that it is satisfied for all cases since $n - l = n + l = L/2$ only occurs if $\rho = 0$, which is the fourth case. Hence, T is a Laurent polynomial, since all the relations are fulfilled.