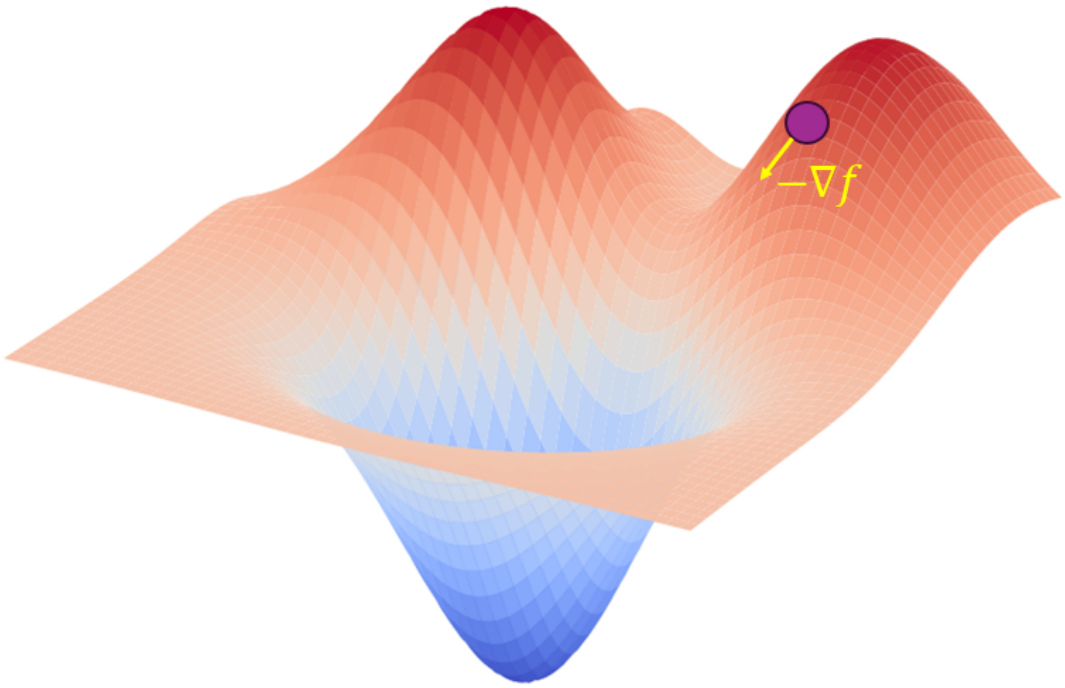


Quantum state tomography with gradient descent

Promoter: Anton Frisk Kockum, Chalmers University of Technology
Co-promoter: Joris Van de Vondel, KU Leuven

Master's thesis in Erasmus Mundus master in nanoscience and nanotechnology

MANUEL SEBASTIAN TORRES HERNANDEZ

DEPARTMENT OF MICROTECHNOLOGY AND NANOSCIENCE - MC2

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

With the support of the Erasmus+ Programme of the European Union

Master's thesis 2024

Quantum state tomography with gradient descent

MANUEL SEBASTIAN TORRES HERNANDEZ

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Microtechnology and Nanoscience - MC2
Applied Quantum Physics
Chalmers University of Technology
Gothenburg, Sweden 2024

Quantum state tomography with gradient descent
Manuel Sebastian Torres Hernandez

Supervisor: Anton Frisk Kockum, Chalmers University of Technology
Examiner: Anton Frisk Kockum, Chalmers University of Technology
Examiner: Joris Van de Vondel, KU Leuven
Examiner: Thilo Bauch, Chalmers University of Technology

Master's Thesis 2024
Department of Microtechnology and Nanoscience - MC2
Applied Quantum Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Gradient-descent visualization constructed in Python, showing some maxima points, and one minima that is the goal.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2024

Quantum state tomography with gradient descent
Manuel Sebastian Torres Hernandez
Department of Microtechnology and Nanoscience - MC2
Chalmers University of Technology

Abstract

Quantum technologies, particularly quantum computing, are advancing by leaps and bounds in recent years. Quantum computing can be implemented using two approaches: one with discrete variables utilizing qubits and the other with continuous variables inspired by the components of the electromagnetic field. A crucial aspect of quantum computing is knowing the system's quantum state. With the quantum state, it is possible to understand and control the system, and know if an algorithm was executed correctly. However, since it is a quantum system, we cannot know the quantum state directly, but we need to infer it from different measurements. Eventually, with the collected data from the measurements, the state of the system can be reconstructed. This reconstruction process is known as quantum state tomography, which is not trivial.

Different approaches have been implemented to solve the reconstruction task. The most standard approach is the maximum-likelihood method. However, it has been shown that there may be other methods that outperform this standard method in different aspects, e.g., a recently proposed machine-learning method with a conditional generative adversarial neural network (CGAN). In this thesis, we propose three methods based on gradient descent to perform quantum state tomography. We benchmark these methods against the standard method of maximum likelihood; for continuous variables, we also benchmark them against the CGAN-based method.

Our results indicate that in certain parameter regimes, some gradient descent-based methods are more efficient and/or reconstruct the state better than the maximum-likelihood method and the CGAN method. The advantages we find are in terms of better overall reconstruction times, using fewer measurement operators, and reconstructing effectively even in the presence of noise in the system.

Keywords: quantum computing, quantum state tomography, maximum likelihood, machine learning, gradient descent, measurement operators, discrete variables, continuous variable.

Acknowledgements

I would like to extend my deepest gratitude to my supervisor, Anton Frisk Kockum, for guiding and motivating me during this experience, which provided me with extensive knowledge in this newly explored area.

I am grateful to my parents who have been my support at all times, regardless of the distance they are always aware of how I am doing and what I am achieving.

Manuel Sebastián Torres Hernández, Gothenburg, July 2024

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AdaGrad	Adaptive gradient method
Adam	Adaptive moment estimation
CGAN	Conditional generative adversarial networks
CV	Continuous variable
DV	Discrete variable
GANs	Generative adversarial networks
GD	Gradient descent
ML	Machine learning
MLE	Maximum likelihood estimation
pdf	Probability density function
POVM	Positive operator-valued measure
QST	Quantum state tomography
lr	Learning rate

Nomenclature

Symbols

N	Number of qubits
$\|\Psi\rangle$	An arbitrary quantum state
C_n	Complex number (probability amplitudes)
α, β	Complex numbers (amplitudes of a general qubit)
θ, ϕ	Angles of the Bloch sphere
$\hat{\rho}$	Density matrix
p_α	probability of measurement outcome α
$\hat{A}, \hat{\mathcal{O}}$	Observable operators
Π_a	Projector operator
$\hat{a}, \hat{a}^\dagger$	Annihilation / creation operator
$\|n\rangle$	Fock states
$\|\alpha\rangle$	Coherent state
$\|\text{cat}\rangle$	Cat state
Q_n^β	Husimi Q function
$\mathbf{d}$	List of the average measure values (data list)
K	Sensing matrix
ρ_f	Flattened density matrix
F	Fidelity
$\mathcal{L}$	Likelihood function
$\mathbb{E}$	Expected value
∇	Gradient
f	Cost / lost function
$\mathcal{W}$	Stiefel manifold matrix
T_G, T_C	Lower triangular complex matrix
C_l	List with the complex values of the state vectors

P_l	Probabilities list
$\mathbf{d}_{\text{noise}}$	Data with noise
η	Step size

Contents

List of Figures

List of Tables

1

Introduction

One of the theories that has most captivated the attention of mankind since its inception at the beginning of the last century is quantum mechanics. Such captivation is not spontaneous; rather, it is due to the fact that since the emergence of this theory our understanding of light and the universe has changed drastically. The subject of light, which was thought to have been explored, underwent a transformation due to Planck's ideas. In 1900, Planck demonstrated that energy is emitted and absorbed in discrete packets, known as quanta [1]. This quanta concept made us realize that we had a limited understanding of light and, consequently, of the universe, setting the stage for a revolution in how we perceive and understand the phenomena of nature. As a result, the theory was applied to describe things such as atoms [2], elementary particles [3], the generation of matter in the universe [4, 5], how the sun works [6], nuclear energy [7], materials [8], and so on.

The quantum theory is built on a few key components: the observables which are represented as operators $\hat{O}$, a probabilistic nature, and the wave functions that are represented as state vectors or state kets $|\Psi\rangle$ in a complex Hilbert space using the Dirac notation [9]. The state kets contain all the information of the physical state [10]. Applying operators to the state vector determines how the system behaves; in particular, observable operators establish the connection between the nano world and the macro world. For this reason, knowing the state vector is crucial in order to understand any quantum system.

This great theory, besides having served to solve fundamental questions about the universe and natural phenomena, has also been of crucial importance for the development of new technologies, thanks to the descriptions of materials as well as how electrons and photons behave and interact. This new understanding of materials and electrons led to the creation of possibly the greatest invention of the last century, the transistor, which is a fundamental part of computers. The transistor became the basic unit of computation with the idea of having two values, 0 or 1, and was called the bit. With computing, many advances have been achieved in humanity, helping in day-to-day life thanks to cell phones and computers. These technologies assist in many aspects of science, such as simulations of chemical reactions and collisions [11], as well as new implementations of artificial intelligence and machine learning [12]. It is almost impossible to live without them these days. Although computation has been a great tool, it has some limitations that make it difficult to perform certain calculations efficiently [13].

Various limitations of computation have led to the consideration of new ways of doing computation, but one in particular led to the approach of computation based on quantum mechanics. That particular limitation appeared in the attempt to simulate quantum systems, where it was realized that computers could not perform such simulations in a practical way, due to the exponential growth of the size of quantum states. Based on that, in 1982, Richard Feynman suggested a type of computer that uses quantum logic to simulate quantum systems [14]. Given its quantum nature, it is logical to think that such simulations can be done easily. However, the idea of a quantum computer remained in the air until the major breakthrough came with Shor in 1994, who came up with a quantum algorithm with a lot of potential. Shor's algorithm, in theory, showed much better efficiency in solving the prime-factorization problem, a crucial pursuit in cryptography. It achieves prime-factorization in polynomial time, while in the case of classical computing, it is accomplished in exponential time [15]. The significant reduction in processing time has triggered the aspiration to continue exploring this new computing approach to identify other systems that can benefit from quantum computing. Its potential applications are being investigated in various fields, including simulation of quantum field theory [16], quantum communication [17], simulations in chemistry [18] and pharmacy [19], finances [20], and more.

For quantum computing, several changes have to be made, the biggest one being its fundamental unit, where the bit in a classical computer does not have the desired quantum nature. Therefore, a quantum bit (qubit) has to be introduced, which is a two-level system that is represented by a wave function, leading to it being represented as a state vector [21]. This type of system does fulfill the required quantum nature, giving it the possibility of being in a superposition between the two energy levels. Although qubits are the most popular way to contain information in the idea of quantum computing, other quantum systems have also been explored to serve this purpose. The other systems are based on terms from quantum optics and are called continuous-variable quantum computation [22].

From the quantum theory, we know that the state vector contains the information of the system, but unfortunately, this representation only works for closed systems. In real terms, it is known that there are many interactions between the quantum system and its surroundings, which is why it is pertinent to define a new more complete component to have the information that the state vector had plus the information due to the interactions. This new object is known as the density matrix ρ. ρ is essentially an operator that characterizes the quantum state of the system. In the world of quantum computing and quantum information, ρ serves us to understand and control the system, design and execute quantum algorithms, implement quantum error correction, explore quantum teleportation [21], and manage a wide array of operations within this context.

Knowing the density matrix is of vital importance for quantum computation, but extracting it from experiments is complicated because of the interaction of quantum systems with measurements. Measuring a quantum state causes the state to

collapse, so we only get a limited amount of information about the system in one measurement. Not having all the information can be harmful to quantum computing because in the case that an algorithm has been applied, we do not know exactly if it has reached the expected state, generating doubts about this quantum technology. To solve this lack of knowledge of the system, we use **quantum state tomography**, which reconstructs the complete description of the quantum state by obtaining the density matrix from the measurement of a set of observables associated with the system. The measurements have to be repeated several times in order to obtain statistics of the observables, which turns the problem into a data-processing task.

Reconstructing the density matrix is not a trivial task. At first sight, it could be considered as a simple system of linear equations, which by inverting the term containing the measurement operators would give the corresponding density matrix, but that is not the case because with that approach we obtain a matrix that is not physical. To achieve a proper reconstruction we have to resort to different mathematical methods or algorithms to obtain a matrix that does represent a quantum system and that is not just a matrix with meaningless numbers. In addition to the physical matrix problem, there is a processing problem due to the exponential growth of the system dimensions. For example, if it is a 1-qubit system it is simply a 2×2 density matrix, with operators having the same dimension. The problem occurs because a quantum computer needs many qubits, so it will no longer be a 2×2 system but a system with matrices $2^N \times 2^N$, where N is the number of qubits. Not only the dimensions of the matrices are growing exponentially, but also the number of measurement operators needed to do the reconstruction have that kind of growth. Hence, each time the system grows the task of reconstructing the density matrix becomes more and more complicated and computationally demanding.

To solve the problems encountered in quantum state tomography, different methods, and algorithms have been proposed over the years, where the maximum-likelihood method has become the standard in the field as proposed in Refs. [23–30]. But just because it is the standard does not mean that it is the definitive method, which is why more methods are being explored to perform the reconstruction of the density matrix in a more optimal and accurate way. A clear example is presented in Ref. [31] which reconstructs the state of a continuous-variable system with fewer iterations and fewer measurement operators using a machine-learning method. Seeing that it is possible to find methods that are better in at least some aspects compared to the standard, creates the opportunity to further explore a better option.

1.1 Thesis aim

The aim of this thesis is to propose new methods to perform quantum state tomography based on the gradient-descent algorithm. To compare the performance of these new models against maximum likelihood and also with the machine learning model of Ref. [31] for the continuous variable case. The goal would be to benchmark these gradient descent methods with respect to the number of iterations, total time, and number of measurement operators needed to reconstruct with an accuracy of

more than 99% the original density matrix of the system.

The idea of working with gradient-descent was inspired by the results in Ref. [32], where a benchmark was made with a gradient-descent model for quantum process tomography (the process of reconstruction of operators based on the measurements). Additionally, we also explore these methods and not other machine-learning methods, because we want to know exactly what the model is doing, to have clearer explanations than a machine-learning model which can be considered as a black box.

2

Quantum information and quantum computing

Since its origins, quantum mechanics has revolutionized the world of science, from giving a better idea of how to properly understand light to understanding more about how the universe is composed. One of the principal components of the quantum theory is the wave function represented by a state vector $|\Psi\rangle$. The state vector can be described with a discrete orthonormal basis in the following form:

$$|\Psi\rangle = \sum_m c_m |m\rangle \tag{2.1}$$

where c_m are complex-valued probability amplitudes, and it has the condition of $\sum_m |c_m|^2 = 1$. In the case of multiple quantum systems, the total state can be expressed from the tensor product $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle \otimes ... \otimes |\psi_m\rangle$. It is important to clarify that the previous representation is not valid in all cases, specifically when there is an interaction between systems that generates entanglement. In these cases, the states cannot be described independently.

This great theory has also been very influential in many technological issues, such as the creation of various components of computers and cell phones, among other things. However, it has been kept in mind that even more technological advances can be made with the quantum theory. Therefore, there has been a lot of research on the subject of quantum computing and quantum information, which is why this chapter is dedicated to explaining those terms of quantum mechanics that are useful for this new technology based on quantum. This chapter will introduce the new basic unit for computation, its states, and the theory of how to reconstruct that state.

2.1 Qubit

In classical computing, the basic unit of information is the bit, which represents binary information by taking the value of 0 or 1. This binary representation is enabled in practice by using transistors, working with the simple concept of allowing or blocking a flow of current. In the case of quantum information and quantum computing, the binary representation (bit) is no longer the basic unit, instead, the new basic unit is the quantum bit, also called qubit. The qubit is a two-level system, in which one of its main characteristics is to be able to be in a state of superposition between states 0 and 1 [21]. The state can be represented using the equation (2.1)

as follows:

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{2.2}$$

where α and β are complex numbers with $|\alpha|^2 + |\beta|^2 = 1$. That constraint is because when a measurement occurs $|\alpha|^2$ gives the probability of getting the result 0 and $|\beta|^2$ the probability of obtaining the result 1 (that generally applies to all c_m and $|m\rangle$) [33].

The state of a qubit can be represented using the Bloch sphere as a useful way of visualization. In this representation, in the poles of the z-axis lie the states $|0\rangle$ and $|1\rangle$, while along the x-axis it has in the poles a superposition of states denoted as $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and $|-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Similarly, along the y-axis, one finds superposition states $|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle)$ and $|-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle)$. These states are illustrated in figure 2.1.

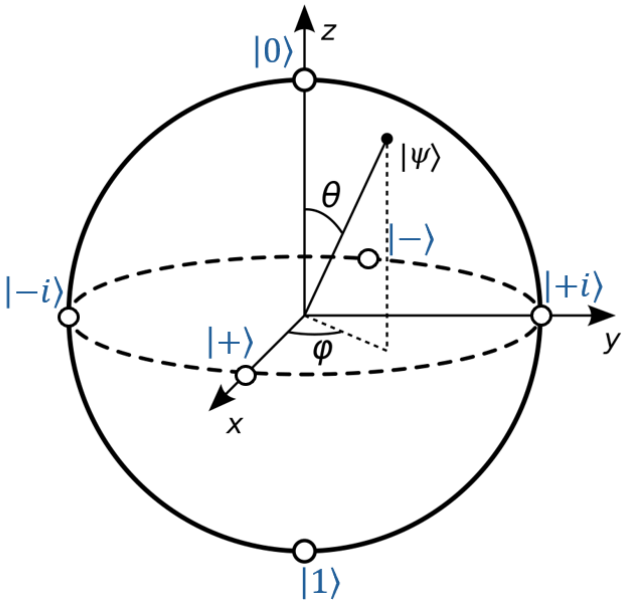


Figure 2.1: Bloch-sphere representation of a qubit state, with the parametrization of $|\Psi\rangle = \cos(\frac{\theta}{2})\,|0\rangle + e^{i\phi}\sin(\frac{\theta}{2})\,|1\rangle$. The image is adapted from [34].

The potential of using qubits as a new technology appears more clear in the case of N qubits for a couple of reasons: First, the presence of entanglement that can occur between multiple qubits, implying a dependence between them; something that does not happen in its classical counterpart in which the bits are independent. Second, the superposition that has already been mentioned is "enhanced" in the sense that with N qubits, the state of the system might be a superposition of 2^N different states. In contrast, only one of the 2^N possible states can be achieved with N classical bits. This implies that at least 2^N classical bits would be required to represent the quantum states.

Quantum computing involving qubits is also called discrete-variable (DV) quantum computation.

2.2 Density matrix

As anyone begins to learn about quantum mechanics, they find that the postulates and equations are written with the state vectors using the Dirac notation, which describes very well the closed systems that are characterized by a single state vector and which are called pure states. However, continuing with this notation would limit us in the analysis of other systems that can be described by a set of several state vectors, e.g. an open quantum system. If it is desired to specify a set of state vectors with their probability value of occurrence (mixed state system), it is necessary to introduce the density matrix. This is essentially an operator that characterizes the quantum state of the system either pure or mixed, where the pure is represented as just the product $\hat{\rho} = |\psi\rangle \langle\psi|$ and the mixed as follows

$$\hat{\rho} = \sum_{\alpha} p_{\alpha} |\psi_{\alpha}\rangle \langle\psi_{\alpha}| \tag{2.3}$$

where p_{α} is the probability of finding the quantum system in the $|\psi_i\rangle$ state, and $\hat{\rho}$ will have dimensions $M \times M$ given by the dimension M of the states involved. Additionally, the maximum rank of this matrix is given by the dimension of its vector space generated by its columns. In this case of density matrices, the rank also means the number of $|\psi_{\alpha}\rangle$ states that constitute ρ.

Fortunately, all the postulates of quantum mechanics can be reassembled with the expression of density matrix [21]. This matrix must satisfy three conditions to be a physical density matrix. First, its trace must be equal to one due to the nature of the probabilities

$$Tr[\hat{\rho}] = \sum_{j} \langle j| \sum_{\alpha} p_{\alpha} |\psi_{\alpha}\rangle \langle\psi_{\alpha}| |j\rangle = \sum_{\alpha} p_{\alpha} = 1 \tag{2.4}$$

second, it must be Hermitian

$$\hat{\rho}^{\dagger} = \sum_{\alpha} p_{\alpha} (|\psi_{\alpha}\rangle \langle\psi_{\alpha}|)^{\dagger} = \sum_{\alpha} p_{\alpha} |\psi_{\alpha}\rangle \langle\psi_{\alpha}| = \hat{\rho} \tag{2.5}$$

and third, it is a positive semi-definite matrix, i.e. all of its eigenvalues must be positive.

To understand the power of the density matrix, it is helpful to determine the elements of the matrix ρ_{ij}. These elements can be represented as:

$$\rho_{ij} = \langle i| \hat{\rho} |j\rangle = \langle i| \sum_{\alpha} p_\alpha |\psi_\alpha\rangle \langle\psi_\alpha| |j\rangle \tag{2.6}$$

$$= \langle i| \sum_{\alpha} \sum_{nm} p_\alpha c_\alpha^n c_\alpha^{m*} |n\rangle \langle m| |j\rangle \tag{2.7}$$

$$= \sum_{\alpha} \sum_{nm} p_\alpha c_\alpha^n c_\alpha^{m*} \langle i|n\rangle \langle m|j\rangle \tag{2.8}$$

$$= \sum_{\alpha} p_\alpha c_\alpha^i c_\alpha^{j*} \tag{2.9}$$

where c_α^n represents the probability amplitudes of the αth state expanded in a basis $|n\rangle$, and it is obtained by applying Eq.(2.1) in Eq.(2.3). The diagonal elements ρ_{ii} are the population terms, that tell the probability of the system being in the ith state [36]:

$$\rho_{ii} = \langle i| \hat{\rho} |i\rangle = \sum_{\alpha} p_\alpha |c_\alpha^i|^2 \tag{2.10}$$

The other elements, the off-diagonal, are the coherence components that has the details of the relative phase of various components within the superposition [36]. In the case of a qubit interacting with an environment, it can tell us the relaxation, excitation, and dephasing rates.

$$\rho_{ij} = \langle i| \hat{\rho} |j\rangle = \sum_{\alpha} p_\alpha |c_\alpha^i c_\alpha^{j*}| e^{i(\phi_\alpha^i - \phi_\alpha^j)} \tag{2.11}$$

This representation of the system becomes pertinent when considering the determination of expected values of observables in quantum systems. The expected value of an observable represents the average measurement outcome, and it is denoted as $\langle \hat{A} \rangle$. This is also called the Born rule, which can be expressed either using the state representation or the density matrix.

$$\langle \hat{A} \rangle = \sum_{\alpha} p_\alpha \langle \psi_\alpha| \hat{A} |\psi_\alpha\rangle \tag{2.12}$$

$$= \sum_{\alpha} \sum_{j} p_\alpha \langle \psi_\alpha| \hat{A} |j\rangle \langle j| |\psi_\alpha\rangle \tag{2.13}$$

$$= \sum_{\alpha} \sum_{j} p_\alpha \langle j|\psi_\alpha\rangle \langle \psi_\alpha| \hat{A} |j\rangle \tag{2.14}$$

$$= \sum_{j} \langle j| \left(\sum_{\alpha} p_\alpha |\psi_\alpha\rangle \langle\psi_\alpha| \right) \hat{A} |j\rangle \tag{2.15}$$

$$= Tr[\hat{\rho}\hat{A}] \tag{2.16}$$

2.3 Measurement

The measurement theory for quantum systems rests on this postulate: It is possible to select any basis and determine the system's state accordingly. After measurement, the system will be found in one of these basis states, even if it was initially

in any state $|\psi\rangle$. The basis state found is random. If the system starts in $|\psi\rangle$, the probability of observing state $|m\rangle$ is given by $|c_m|^2$ (see Eq.(2.1)) [37]. That kind of measurement receives the name of von Neumann measurement. This is the most traditional description of measurement and it can also be seen as a projective measurement. We can take an observable $\hat{A}$, which can be diagonalized

$$\hat{A} = \sum_a \lambda_a \Pi_a \tag{2.17}$$

where Π_a is the projector onto the eigenspace of $\hat{A}$ with eigenvalue λ_a. The projectors are orthonormal fulfilling $\Pi_a \Pi_b = \delta_{a,b} \Pi_a$. In the condition that the eigenvalue set is non-degenerate, the projector operator would be $\Pi_a = |a\rangle \langle a|$ [38]. This rank-1 projector is the von Neumann measurement.

In the case of a pure state, the probability of obtaining the result λ_a is expressed as:

$$p(\lambda_a) = \langle \psi | \Pi_a | \psi \rangle \tag{2.18}$$

After the measurement, the state $|\psi\rangle$ collapses to:

$$\frac{\Pi_a |\psi\rangle}{\sqrt{\langle \psi | \Pi_a | \psi \rangle}} \tag{2.19}$$

More generally, for any state, we can have the probability as $p_a = \mathrm{Tr}[\Pi_a \rho \Pi_a]$, and the state after the measurement is given by:

$$\tilde{\rho}_a = \frac{\Pi_a \rho \Pi_a}{\mathrm{Tr}[\Pi_a \rho \Pi_a]} \tag{2.20}$$

This is the most common way of measuring, at least for discrete variables.

The description provided by von Neumann is generally not sufficient because most observations that we can perform do not align with this nature. For that reason, it is important to define a set of operators $\hat{E}_a = \hat{A}_a^\dagger \hat{A}_a$ which must fulfill the following properties:

1. Hermitian operators $\hat{E}_a^\dagger = \hat{E}_a$
2. Complete $\sum_a \hat{E}_a = \mathbb{1}$
3. Positive operators $\hat{A}_a \geq 0$

This set, known as the positive operator-valued measure (POVM), is apt for determining the probability of a generalized measurement outcome, i.e. with a POVM $\hat{E}_a$ we can have from the postulate the probability of outcome as $p(a) = \langle \psi | \hat{E}_a | \psi \rangle$ [21]. An example illustrating the validity of these POVMs is by considering the case of rank-1 projective measurements. Due to orthonormality and completeness, we have $\hat{E}_a \equiv \Pi_a^\dagger \Pi_a = \Pi_a$ [21], that brings us back to Eq. (2.18).

2.4 Continuous variables

Another way to approach quantum information is through continuous variables (CVs), which has shown potential as another basic unit for encoding information in quantum computing, making it interesting to explore and broadening the research to address more than just qubits. CV is based on concepts of quantum optics and in this paradigm, information is encoded in observables that are characterized by a continuous spectrum. These observables may include the quadratures of the electromagnetic field, which are represented as amplitude $\hat{x} = (\hat{a} + \hat{a}^\dagger)/\sqrt{2}$ and phase $\hat{p} = (\hat{a} - \hat{a}^\dagger)/(i\sqrt{2})$, and they follow the commutation relation $[\hat{x}, \hat{p}] = i$ [40].

The quadratures introduce the non-Hermitian annihilation $\hat{a}$ and creation $\hat{a}^\dagger$ operators, which satisfy $[\hat{a}, \hat{a}^\dagger] = 1$. Moreover, the combination of the creation and annihilation operators gives the number operator $\hat{n} = \hat{a}^\dagger\hat{a}$. This indicates that they will not have only two levels but infinite levels, resulting in a Hilbert space of infinite dimensions even for just one bosonic mode [40].

For the single-mode bosonic field, the states are represented as $|n\rangle$ which are called Fock states or number states, and it appears from the quantization of the electromagnetic field satisfying

$$\hat{n}\,|n\rangle = n\,|n\rangle \tag{2.21}$$

where n represents the number of photons in that mode. These states are orthogonal, and one special case is the $|0\rangle$ that receives the name of the vacuum state.

As it was expressed the operators of annihilation and creation have an important role and it is natural to think that they would have an eigenstate, this one is expressed as $|\alpha\rangle$ and it is called coherent state

$$\hat{a}\,|\alpha\rangle = \alpha\,|\alpha\rangle \tag{2.22}$$

$$\langle\alpha|\,\hat{a}^\dagger = \alpha^*\,\langle\alpha| \tag{2.23}$$

where α is a complex number, because $\hat{a}$ is not hermitian. The coherent state can be expanded in terms of Fock states. Furthermore, $|\alpha\rangle$ is widely appreciated for their notable similarity to classical states, by producing expectation values that closely resemble those of the classical electric field [41], while containing only vacuum noise

$$|\alpha\rangle = e^{-\frac{|\alpha|^2}{2}} \sum_{n=0}^{\infty} \frac{\alpha^n}{\sqrt{n!}}\,|n\rangle \tag{2.24}$$

the coherent state can also be defined as the displacement ($D(\alpha)$) of the vacuum state $|\alpha\rangle = exp(\alpha a^\dagger - \alpha^* a)\,|0\rangle$ [42]. With the coherent state, it is possible to create other states from a combination of two of these, that is the case of the cat state that it is represented as follows

$$|\text{cat}\rangle = \mathcal{N}(|\alpha\rangle + |-\alpha\rangle) \tag{2.25}$$

with $\mathcal{N}$ as a normalization factor. The cat representation has been important for encoding quantum information, and this design allows to correct problems of single photon losses and some phase errors. Also, one of the most important results of this encoding is that it has been able to maintain the coherence longer time than the qubits [43].

In CV it is possible to represent the density matrix in terms of the coherent states via the Glauber-Sudarshan $P(\alpha)$ function

$$\hat{\rho} = \int P(\alpha) \left|\alpha\right\rangle \left\langle\alpha\right| d^2\alpha \tag{2.26}$$

where $P(\alpha)$ is a weight function, that can be considered analogous to the phase-space distributions of statistical mechanics. The values of $P(\alpha)$ are real, due to the hermiticity of the density matrix. Additionally, $P(\alpha)$ can be negative, these cases are categorized as "nonclassical" [41].

For operators that depend on coherent states, we can represent those in a similar way as the density matrix in the P-representation

$$\hat{B} = \int B_p(\alpha, \alpha^*) \left|\alpha\right\rangle \left\langle\alpha\right| d^2\alpha \tag{2.27}$$

To have the average of $\hat{B}$, the same idea as in Eq. (2.16) is used

$$\begin{aligned} \langle\hat{B}\rangle &= Tr(\hat{B}\hat{\rho}) && (2.28)\\ &= \sum_j \left\langle j\right| \int B_p(\alpha, \alpha^*) \left|\alpha\right\rangle \left\langle\alpha\right| \rho \left|n\right\rangle d^2\alpha && (2.29)\\ &= \int B_p(\alpha, \alpha^*) \left\langle\alpha\right| \hat{\rho} \left|\alpha\right\rangle d^2\alpha && (2.30) \end{aligned}$$

In that expression, the term representing the expectation value of the density matrix with respect to the coherent state defines the phase-space probability distribution. Moreover, that term is called the Q function, or Husimi function

$$Q(\alpha) = \frac{1}{\pi} \left\langle\alpha\right| \hat{\rho} \left|\alpha\right\rangle \tag{2.31}$$

In practice for the measurement of optical states, one can apply a displacement with some amplitude β and after some time t measuring the photon number distribution, this procedure at the ends gives a general form of the Q function [44]

$$Q_n^\beta = Tr(\left|n\right\rangle \left\langle n\right| D(-\beta)\hat{\rho}D^\dagger(-\beta)) \tag{2.32}$$

2.5 Quantum state tomography

At this point it is already known how a quantum state is expressed and we know that when measuring a system the information given by the operator $\hat{A}$ acting on

the state is limited, i.e. we do not obtain the expression and the complete information of the density matrix. Not knowing the complete expression of the system is a big problem for the experimental part, since it is not possible to properly track the processes that are applied to the system. Therefore, a method is needed to obtain all the information of the system and that method is known as quantum state tomography.

The purpose of quantum state tomography (QST) is to reconstruct the complete description of a quantum state by obtaining the density matrix ρ. This process is done by measuring some properties of the state, i.e., measurements of a set of observables associated with the system [45], and therefore becomes a data processing task [46]. The data for the process are the outcomes of single shots from positive-operator-valued measures (POVMs); these measurements are repeated many times on an identical quantum state to obtain some statistics. The statistics lead to getting the frequencies d_i that are proportional to the expectation value, and with that, we obtain $\mathbf{d}$ [31]:

$$K\,\rho_f = \mathbf{d} \tag{2.33}$$

$$\rho_f = K^{-1}\mathbf{d} \tag{2.34}$$

where ρ_f is the flattened density matrix and K is the sensing matrix that is determined by the choice of POVMs.

The linear inversion in Eq. (2.34) "looks easy" on paper, but doing the linear inversion does not guarantee a physical matrix. Additionally, due to the complexity of the set of measurement operators and noise in the system or measurements, the reconstruction of the density matrix has to be approached by other methods that could surpass the complexity and facilitate the estimation of this operator. Furthermore, in the case of qubits, the amount of data needed increases exponentially with the number of qubits. This leads to problems in data management and storage, which in turn complicates the process of properly reconstructing the system.

2.5.1 Distance measure

After reconstructing the density matrix from the values of its measurement, it is important to know if this matrix is physical and how well it was reconstructed. In other words, how similar this new matrix is to the original one? To assess this, it is pertinent to employ some distance measures, with fidelity being the most common in QST.

Fidelity serves as a distance measure between two density matrices ρ and σ, indicating the degree of similarity between the two states. A value of 1 means the states are very similar, while 0 means they are quite different. Mathematically it is defined as:

$$F(\rho,\sigma) = Tr\left[\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}}\right] \tag{2.35}$$

The significance and range of the values become clearer when considering the example of a pure state:

$$F(|\psi\rangle\langle\psi|, \rho) = Tr\left[\sqrt{\langle\psi|\rho|\psi\rangle\,|\psi\rangle\langle\psi|}\right] = \sqrt{\langle\psi|\rho|\psi\rangle} \tag{2.36}$$

3

Optimization and machine-learning algorithms

The problem of quantum state tomography has been approached by various algorithms over the years. Among these, the Maximum likelihood estimation (MLE) method, introduced by Hradil in 1996, has emerged as the most popular, offering a flexible approach [48]. Since its introduction, efforts have been made to refine the MLE technique [23–30] or explore alternative approaches such as the integration of Machine Learning (ML), particularly through neural networks [31]. This alternative has demonstrated distinct advantages compared to traditional MLE algorithms.

This chapter aims to provide a brief overview of the theoretical part of these two algorithms (MLE and ML neural networks) in a general context. We will not go into too much detail because we are only looking to compare against these two algorithms; these methods are not new and they were simply implemented to have a benchmark base. Then we will introduce the primary method employed in this thesis, which is gradient descent (GD). A detailed explanation of how this gradient-descent technique operates will be provided.

3.1 Maximum likelihood estimation

The MLE is a technique that is employed to estimate the parameters of a certain probability distribution, using some observed data. In a more general way, this is a technique of data analysis to determine the probability distribution most likely to have produced the sample data.

To formulate the MLE technique, it is important to first define a set of M observations $\mathcal{X} = \{x_1, x_2, ..., x_M\}$ that comes from a probability distribution. It is assumed that the joint probability density function (pdf) follows a known parametric functional form, and will be denoted as $p(\mathcal{X}|\theta)$, with a parameter θ that is unknown and the purpose is to determine its value [49]. Given the parameter θ, the probability of observing data $\mathcal{X}$ is represented by the pdf [50]. This pdf in the context of MLE is known as the likelihood function, and it can be expressed in terms of the M observations as:

$$\mathcal{L}(\theta) = \prod_{k=1}^{N} p(x_k|\theta) \tag{3.1}$$

As the name says, the idea is to find the parameter that maximizes this likelihood function. This expression can become very complex for the calculations. Because if you want to maximize, many product derivatives will appear. For that, we can use the logarithm function, since it is a monotonic and increasing function [49]. Then, it is easier to look for the maximum of this because we would no longer have the multiplication of several components. Instead, we will now have the sum of the different components as a result of the properties of the logarithm.

$$\ell(\theta) = \sum_{k=1}^{N} \ln p(x_k|\theta) \tag{3.2}$$

$$\theta^*_{MLE} = \arg\max_{\theta} \ell(\theta) \tag{3.3}$$

MLE is a powerful method, however, it has its disadvantages, the main one is that this technique does not give you any measure of uncertainty in the estimate.

3.2 Generative adversarial networks

Machine learning is one of the most popular topics nowadays, and it has been implemented in a lot of different areas. That is because the conceptual idea of learning, acquiring new or modified knowledge and skills from experiences, can be applied to a wide range of existing areas, and it is something natural in life. Therefore, extrapolating that fundamental idea, we can take the importance of machine learning, which we can see as a field that investigates algorithms that improve their performance as they are given more data, that is, they learn by experience [51]. This process involves recognizing patterns within data and using them to make decisions or predictions. Generally speaking, there are three types of machine learning: Supervised, where the dataset to be used is labeled, as in the case of image recognition; Unsupervised, where, contrary to the previous case, there are no labels, so it generates knowledge, which is effective in cases of clustering; Reinforcement learning, which learns by rewards and can be seen in cases such as video games.

Just as the idea of learning was borrowed from a natural biological process, machine learning borrows another concept from biology, the connection and function of neurons in the human brain [51]. With this idea in mind, the concept of machine learning with neural networks arises. These networks are connected in different layers, where the output of one layer becomes the input of the next, until the last one, where the output is the final result. The structure of a neuron can be written as follows:

$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right) \tag{3.4}$$

where w_i are the weights of the connections, x_i represents the input value, b is the bias of the neuron, N is the number of connections to that neuron and y is the output of the neuron. The $f(\cdot)$ means the function that the neuron will perform

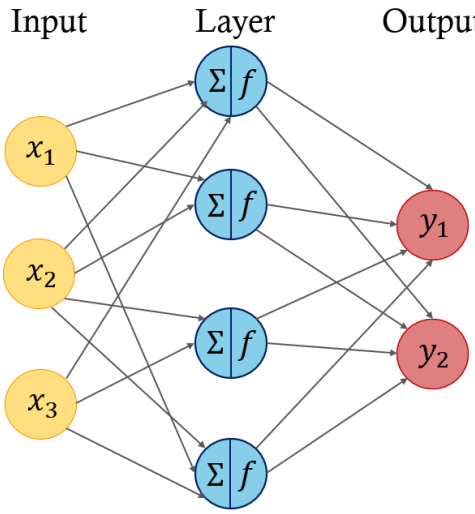


Figure 3.1: A feed-forward neural network with one hidden layer, the symbols in the neurons in the layer represent the sum of the inputs with the weights and bias, and then the application of the function f. Inspired by the illustration from [52].

with the inputs. One common structure of a neuron in a network is shown in Fig. 3.1.

The neural network model has a broad spectrum, but in this thesis only the generative adversarial networks (GANs) [53] will be mentioned. This focus is because that model was the one implemented to benchmark the CV cases in QST, and will be mentioned to provide a context for comparison with the newly proposed models. We will follow the description of GANs given in Ref. [47].

The GAN model is characterized by two separate neural networks, one of which has as input a noise vector z to introduce stochasticity, while the second one has as input the original data distribution $p(x)$. The first network is the generator $G(z;\theta)$, where G is a differentiable function [47], and as the name suggests generates some data from the random input. This generator is trained by the second neural network called discriminator $D(x';\phi)$, which evaluates the outputs of the generator [47]. The outputs of D represent the probability that x' comes from the original distribution $p(x)$, see Fig. 3.2.

Both networks have to be trained, so this training is given alternatively, in which the discriminator has to optimize the probability of assigning the labels correctly, by maximizing the value of expectation with respect to the parameter ϕ [47]

$$\mathbb{E}_{x\sim p(x)}\left[\ln(D(x;\phi))\right] + \mathbb{E}_{z\sim p(z)}\left[\ln(1 - D(G(z;\theta);\phi))\right] \tag{3.5}$$

and the generator has to produce data similar to the data of the original distribution, this is given by varying θ to minimize the expectation [47]

$$\mathbb{E}_{z\sim p(z)}\left[\ln(1 - D(G(z;\theta);\phi))\right] \tag{3.6}$$

Finally, the discriminator evaluates the data given by the generator and with a

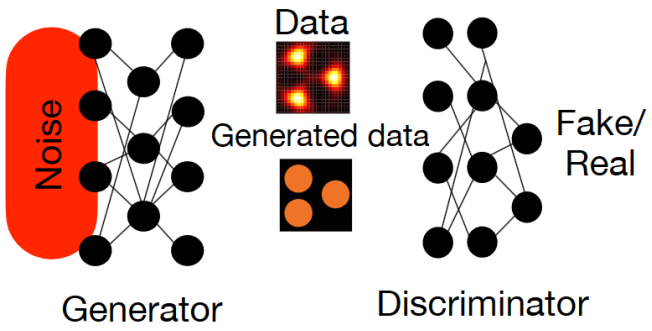


Figure 3.2: Structure for the GAN model. The generator assembles some data from noise, which is then evaluated and trained by the discriminator. Illustration from [47].

boolean output tells us if it is similar or not.

3.3 Gradient descent

The gradient-based optimization method is one of the most widely used today and one of the most reliable methods. This powerful method is based on basic ideas of vector calculus which recovers the concept that the gradient (∇) of a differentiable function f, denoted as the vector of partial derivatives of f, points in the direction of the highest growth of the function f. By knowing in which direction the function has the fastest growth, we can deduce the direction in which it decreases most rapidly by taking the opposite direction. This decreasing direction is denoted as the negative of the gradient and thereby we take steps to update the value of the parameter to be optimized. In recursive form, the model can be expressed as:

$$\theta^{t+1} = \theta^t - \eta \nabla f(\theta^t) \tag{3.7}$$

with θ as the parameter to optimize, $\eta > 0$ as the step size (or learning rate), and f the cost/loss function. This model is known as the vanilla gradient descent algorithm. The choice of the value of the step size has to be in a way to ensure convergence in the minimum point [49], see Fig. 3.3.

This method is divided into three categories. The first one, called batches, evaluates all the data of the system to perform one update, which results in a long processing time. Secondly, the stochastic method evaluates one example at a time and updates the parameter with each evaluation, producing a slow convergence, but it is more capable of avoiding local minima as shown in Refs. [54, 55]. Finally, the mini-batch method updates with batches of data from n examples. This combines the best of batches and stochastic to converge faster and reduce processing time.

3.3.1 Modifications

The vanilla gradient descent is very simple in many aspects. Therefore, it has been explored ways to enhance its performance regarding arriving at the global

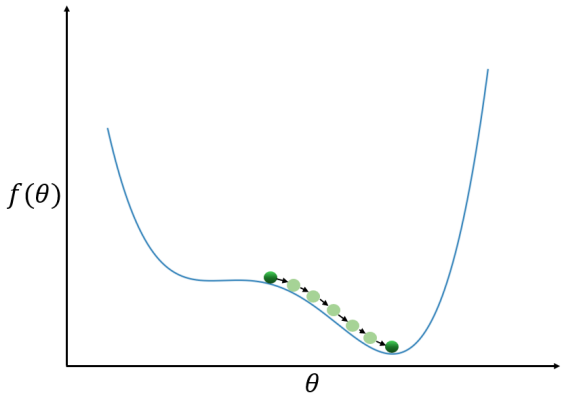


Figure 3.3: Illustration of gradient descent with small learning rate arriving at the global minimum.

minimum and convergence times. Momentum is one of the most popular approaches to accelerate gradient descent, guiding it in the appropriate direction and dampening oscillations around local minima. [47]. This is possible by adding a fraction γ usually with a value of 0.09 and a vector v that modifies the update rule [56]

$$v_t = \gamma v_{t-1} + \eta \nabla f(\theta^t) \tag{3.8}$$

$$\theta^{t+1} = \theta^t - v_t \tag{3.9}$$

these momenta will continue to accumulate and increase depending on whether the direction of the gradient coincides with that of the momentum. Consequently, a faster convergence will occur.

Another modification that can be made to the gradient descent is the adaptive gradient method (AdaGrad), which as its name indicates adapts the learning rate for different parameters considering the information of the previous step [56]. This leads to an η depending on the t step in the algorithm.

From the two modifications mentioned above, it is possible to have a combination of these, which is called adaptive moment estimation (Adam). Aside from using adaptive learning, Adam maintains an exponentially moving average of the previous gradients m_t and of the square gradient v_t [56] that are given by

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{3.10}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{3.11}$$

where β_1, β_2 are coefficients close to 1 and $g_t = \nabla f(\theta^t)$. The vectors m_t and v_t are estimates of the first moment and the second moment, representing the mean and the uncentered variance, respectively. At the initialization of m_t and v_t in the value 0 a bias is generated, and to correct it the following expression is used

$$\tilde{m_t} = \frac{m_t}{1 - \beta_1^t} \tag{3.12}$$

$$\tilde{v_t} = \frac{v_t}{1 - \beta_2^t} \tag{3.13}$$

With all these modifications, the new update expression looks like

$$\theta^{t+1} = \theta^t - \frac{\eta}{\sqrt{\tilde{v}_t} + \epsilon}\tilde{m}_t \tag{3.14}$$

where ϵ is a value of numerical stability, that it is usually around 10^{-8}.

3.3.2 Stiefel manifold

The GD method with Stiefel manifold is a concept borrowed from differential geometry that restricts the gradient so that, after any update, the parameters never leave the manifold. This is possible because of the notion of the manifold, which is a set containing the idea of closeness between elements [32]. The topic of manifolds and the application of differential geometry is quite extensive, therefore we will only mention the necessary parts without going into details. If you want to explore more on this topic, it is recommended to review the supplementary material of Ref. [32] which shows the steps to obtain the final expression, and Ref. [57] for more conceptual details.

The Stiefel manifold is defined as the set of matrices with p orthonormal columns in $\mathbb{R}^n$, this condition can be expressed as: for $K \in R^{n\times p}$ such that $K^T K = \mathbb{1}_p$ [57]. This concept can be extended to a complex space, so now we take a matrix $\mathcal{W} \in \mathbb{C}^{n\times p}$ such that $\mathcal{W}^\dagger\mathcal{W} = \mathbb{1}_p$ [32].

To achieve the gradient restriction several terms must be rewritten to obtain a new gradient expression to be used in GD. For that let us take $G = \nabla_{\mathcal{W}} f(\mathcal{W})$ which after each update has to be normalized by the L2 norm $\tilde{G} = G/||G||_2$. Then we define the stacked matrices $A = [\tilde{G}, \mathcal{W}]$ and $B = [\mathcal{W}, -\tilde{G}]$ [32]. As a result, we have the following expression for the gradient

$$\nabla^*_{\mathcal{W}} f(\mathcal{W}) = A\left(\mathbb{1} + \frac{\eta}{2}B^\dagger A\right)^{-1} B^\dagger\mathcal{W} \tag{3.15}$$

This gradient is applied to the vanilla GD model, giving us

$$\mathcal{W}' = \mathcal{W} - \eta\nabla^*_{\mathcal{W}} f(\mathcal{W}) \tag{3.16}$$

After each update of the values of $\mathcal{W}$, the orthonormality condition is maintained, ensuring that it remains in the manifold.

4

Methods

Reconstructing the density matrix is quite complicated, and cannot be done with a linear inversion because it does not guarantee a physical density matrix (see section 2.5). That is why reconstructing the density matrix is performed with other methods, as mentioned in Chapter 3. However, when those methods are directly applied, they can regenerate a matrix with the desired dimensions, but that matrix lacks the three characteristics that define a physical density matrix.

In order to achieve a correct functioning of the algorithms in the context of QST, constraints from mathematics and physics must be integrated. These constraints serve to guide the algorithms to operate within a defined space where each update effectively reconstructs a physical density matrix. For example, a projection that brings you back into the subspace of physical density matrices, a regularization mechanism that penalizes deviations from the desired subspace, limitations imposed on the search area, and a re-parameterization of the states; see Fig. 4.1.

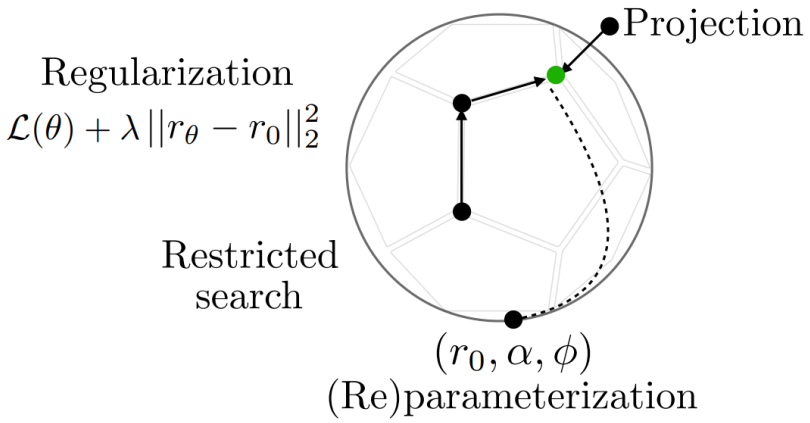


Figure 4.1: Representation of the defined space and some constraints to maintain the points (updates) in that space. The green point is the final point and the black points are trying to arrive at the green point with different constraints. Illustration from [47].

This chapter will focus on using the algorithms discussed in Chapter 3 to perform quantum state tomography. Therefore, it will explain how the algorithms have to be rewritten and which constraints have to be applied in each one in order to reconstruct a physical density matrix. The MLE and GAN algorithms are not new proposals;

they are taken from Refs. [23, 24] for MLE and Refs. [31, 46] for GAN, in order to be compared with the GD models which are proposed in this thesis, which are GD-Cholesky, GD-manifold, and GD-projective. The purpose of this comparison is to benchmark the new methods based on GD against MLE and GAN, taking into account the fidelity values, the algorithm time, the number of iterations, the system to be reconstructed, and the noises.

4.1 Quantum state tomography with maximum likelihood estimation

For the case of maximum likelihood estimation for QST, we will follow the iterative model shown in Ref. [24] and Ref. [23].

Let us first define the set of all POVM elements Π_i that corresponds to the measurement of the system $\hat{\rho}$ [24]. Secondly, the frequency of occurrences for each outcome is indicated by f_i [23]. Finally, P_i represents the conditional probability of measuring outcome i [24]. From the Born rule showed in Eq. (2.16), we can rewrite the probability as $P_i = p(\Pi_i|\hat{\rho}) = \langle i|\hat{\rho}|i\rangle = \mathrm{Tr}[\Pi_i\hat{\rho}]$. This one will be the probability density function of the likelihood function Eq. (3.1)

$$\mathcal{L}(\hat{\rho}) = \prod_i \mathrm{Tr}[\Pi_i\hat{\rho}]^{f_i} \tag{4.1}$$

where in this case we are looking for the $\hat{\rho}$ that maximizes the likelihood (4.1). As mentioned earlier, maximizing the log-likelihood function is preferred as it simplifies calculations and ultimately transforms the task into a convex optimization problem.

$$\ell(\hat{\rho}) = \sum_i f_i \ln(\mathrm{Tr}[\Pi_i\hat{\rho}]) \tag{4.2}$$

The state $\hat{\rho}_{ML}$ that maximizes the log likelihood must satisfy $\hat{\rho}_{ML} = \hat{R}(\hat{\rho}_{ML})\hat{\rho}_{ML}$ where $\hat{R}(\hat{\rho})$ is a positive operator that is represented as [24]

$$\hat{R}(\hat{\rho}) = \frac{1}{M}\sum_i \frac{f_i}{P_i}\Pi_i = \frac{1}{M}\sum_i \frac{f_i}{\mathrm{Tr}[\Pi_i\hat{\rho}]}\Pi_i \tag{4.3}$$

with $M = \sum_j f_j$. The operator $\hat{R}$ and $\hat{\rho}$ are Hermitian, and because of that condition the equation $\hat{\rho}_{ML} = \hat{R}(\hat{\rho}_{ML})\hat{\rho}_{ML}$ leads to the following iterative algorithm

$$\hat{\rho}_{k+1} = \mathcal{N}[\hat{R}(\hat{\rho}_k)\hat{\rho}_k\hat{R}(\hat{\rho}_k)] \tag{4.4}$$

where $\mathcal{N}$ denotes normalization to a unit trace. This representation preserves positivity, ensuring that likelihood increases with each step. Furthermore, the matrix obtained at each stage is a physical density matrix.

This model can run into difficulties in cases where $\mathrm{Tr}[\Pi_i\hat{\rho}] = 0$, giving a divergence in the expression, so it is important to take into account the first ansatz used to start the iterations, i.e., the first ansatz must be constructed with nonzero values

in all inputs. In some cases of DV, the divergence can be present after the updates of the values of ρ, for that it is important to check the average value of the specific measurement operator from the original case and the one that is constructing $\hat{R}$ in Eq.(4.3). If both cases are zero, then there will be a zero contribution in the reconstruction from that specific average of measurement operator. For example, if we are reconstructing the state $|0\rangle$ and, in one part of the algorithm we are close to the reconstruction meaning that we will have a value zero in the position ρ_{11}, then the average of the measurement operator $|1\rangle\langle 1|$ is zero in both cases, telling us that is not necessary to add that information again in $\hat{R}$. Therefore, in the sum of Eq.(4.3), a 0 is assigned multiplying the selected Π_i operator ($0 \cdot |1\rangle\langle 1|$ in the example). With that, we are able to avoid a divergence in $\hat{R}$.

4.2 Quantum state tomography with conditional generative adversarial networks

We have already seen how to adjust the MLE method for QST, now we will look at how to modify the machine learning method to be used in QST. The implementation of the model of a neural network, based on GANs (see section 3.2), will follow the description provided in Refs. [31, 46, 47]. For further details about this model, it is recommended to consult these references.

Starting from the idea shown with the GANs method, it is now necessary to condition the generator so that it does not have random values as input, but now has inputs with respect to an $\mathbf{x}$ variable. This $\mathbf{x}$ variable input would also apply to the discriminator, which will use those $\mathbf{x}$ values to check the similitude of the generator output. This modification is called conditional generative adversarial network (CGAN). Therefore, the generator outputs are now of the form $G(x, z; \theta)$ and the discriminator outputs are $D(x, y; \phi)$ [31]. Despite the modification, it remains with the same concept of optimizing the equations (3.5) and (3.6).

For quantum state tomography, the conditions to be imposed are the average measure values ($\mathbf{d} = \mathrm{Tr}[\Pi_i\hat{\rho}]$) and the measurement operators $\{\Pi_i\}$. The generator takes both inputs and tries to create its own density matrix. For the creation of this matrix, the generator must have two layers with specific tasks. The first one which is called "DensityMatrix" layer is in charge of creating a complex lower diagonal matrix T_G, which must have real values on the diagonal [31]. In this way, by applying the Cholesky decomposition with the matrix T_G a semi-positive hermitian matrix is recovered [58]. Then the matrix is normalized to fulfill all three conditions of a physical density matrix. The normalized Cholesky decomposition is expressed as:

$$\rho_G = \frac{T_G^\dagger T_G}{Tr[T_G^\dagger T_G]} \tag{4.5}$$

where $T_G^\dagger$ is an upper complex diagonal matrix. By using this idea of creating T_G, a constraint of a restrictive search is enforced, see Fig. 4.1.

The second layer, which is called "Expectation", takes the density matrix created by the generated one and together with the measurement operators, calculates the expectations values of these operators ($d' = \mathrm{Tr}[\Pi_i \rho_G]$), which will be important for the discriminator. The discriminator will take the data $\mathbf{d}$ and d', to compare and train the Generator based on the similarity of the two data sets [31]. The process is represented in Fig. 4.2.

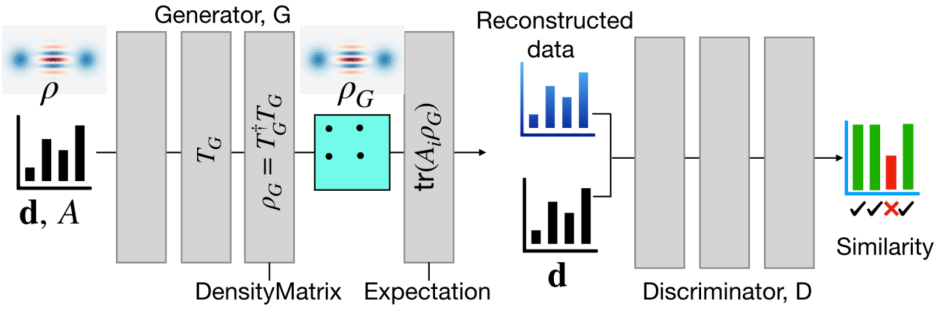


Figure 4.2: CGAN structure for QST. With the data $\mathbf{d}$ and the set $\{\Pi_i\}$ as the inputs of the Generator and Discriminator, and with the inside process of the two neural networks. Illustration from [31].

4.3 Quantum state tomography with gradient descent

For the case of gradient descent, which is the primordial focus of this thesis, the main aspect is to define the methods to maintain the physicality of the states and represent that method in the cost function. The cost function has to take as input the data of the average measured values $\mathbf{d}$, the measurement operators, and an ansatz of the density matrix. This function expresses the difference between the original data $\mathbf{d}$ and the new data d' that is being constructed from the newly reconstructed ρ using Eq. (2.16). It is represented as follows:

$$f(\mathcal{C}) = \sum_i \left[d_i - \mathrm{Tr}[\Pi_i \rho(\mathcal{C})]\right]^2 + \lambda ||\mathcal{C}||_1 \tag{4.6}$$

where $\rho(\mathcal{C})$ serves as a general representation of the density matrix with a parametrization $\mathcal{C}$ acting as a constraint to ensure the system remains as a physical density matrix, and λ is a hyperparameter that accompanies the L1 norm (sum of all the components of $\mathcal{C}$). The term represented by the L1 norm serves as a regularization constraint to penalize the updates in case that the updated term were trying to escape the physical subspace. This is because the L1 norm shrinks certain coefficients and adjusts others to a value of 0, trying to keep the most relevant features [59]. Hence, it promotes sparsity in the coefficients. L1 norm was selected over other norms because it has been shown to be the preferred standard for data-driven constraints as shown in Ref. [60]. This L1 norm contributes in a minor way, which is

why the most important term to keep the matrix in the physical subspace is the parametrization with $\mathcal{C}$.

The parametrization $\rho(\mathcal{C})$ is obtained from the ansatz ρ, this ansatz is generated taking into account the equation (2.3). Therefore, a list with N ket states is generated, in which each state has random values while maintaining the characteristics of being physical states. In addition, a list with probabilities P_i is generated, therefore we have all the elements to represent the system as a density matrix, for instance in the case of 1 qubit with $N = 2$ it would be:

$$\text{ket list} = \begin{bmatrix} |\psi_1\rangle \\ |\psi_2\rangle \end{bmatrix} = \begin{bmatrix} \alpha_1 \\ \beta_1 \\ \alpha_2 \\ \beta_2 \end{bmatrix} \tag{4.7}$$

$$\text{Probabilities list} = \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \tag{4.8}$$

$$\rho = P_1 |\psi_1\rangle \langle\psi_1| + P_2 |\psi_2\rangle \langle\psi_2| \tag{4.9}$$

with these three elements you can obtain $\rho(\mathcal{C})$, which will depend on the method you want to use and will be explained in the following subsections when each constraint method is presented.

Another aspect to consider is that gradient descent will be applied with the mini-batch mode, due to the balance offered by this mode and mainly to avoid getting stuck in local minima (see section 3.3). This extra aspect leaves us with several hyperparameters (λ, η, decay of learning rate, size of mini-batches) in the GD model, to which we have to find the best value to obtain the most optimal version of this algorithm.

Additional aspects will be considered depending on the type of constraint to be applied. In this thesis, we will employ three methods of GD to do QST: Cholesky representation, Stiefel manifold, and projective normalization. Detailed examinations of each follow below.

4.3.1 Cholesky representation

In the Cholesky approach, we use the idea of rewriting and restricting the ansatz to a lower triangular complex matrix $T_{\mathcal{C}}$ (see appendix A), similar to how CGAN generates its density matrix, but the difference is that in this aspect the gradient model focuses on updating the values of the matrix $T_{\mathcal{C}}$ by steps, keeping that the values on the diagonal have to be real. Using Eq.(4.5) and implementing it in the cost function Eq.(4.6), we have the following representation

$$f(T_{\mathcal{C}}) = \sum_i \left[d_i - \mathrm{Tr}\left[\Pi_i \frac{T_{\mathcal{C}}^\dagger T_{\mathcal{C}}}{\mathrm{Tr}[T_{\mathcal{C}}^\dagger T_{\mathcal{C}}]} \right] \right]^2 + \lambda ||T_{\mathcal{C}}||_1 \tag{4.10}$$

looking at it from the vanilla GD model, the updated $T_{\mathcal{C}}$ values would look like

$$T_{\mathcal{C}}^{t+1} = T_{\mathcal{C}}^t - \eta \nabla f(T_{\mathcal{C}}^t) \tag{4.11}$$

After each update of values, it will always maintain the structure of a complex lower triangular matrix and by the Cholesky decomposition, it provides a physical density matrix.
In order to get better results, it is pertinent to use the modification Adam in the case of constraint by Cholesky representation.

4.3.2 Manifold

For the constrain with the Stiefel manifold we have to find the matrix $\mathcal{W}$ such that $\mathcal{W}^\dagger \mathcal{W} = \mathbb{1}_p$ (see subsection 3.3.2). For that, we use the fact that quantum states have a probabilistic nature and that the sum of the probabilities has to be one, then it already gives us an idea of how to rewrite the ψ states and the p_α probabilities. Resulting in a vector with a length of the Hilbert spaces times the number of states in the ansatz, and each value is the complex amplitude of the state accompanied by the square root of the probability of being in that state, e.g. the case of 1 qubit with an ansatz of two states can be appreciated as

$$\mathcal{W}^\dagger \mathcal{W} = \begin{bmatrix} \sqrt{p_1}\alpha_1^* & \sqrt{p_1}\beta_1^* & \sqrt{p_2}\alpha_2^* & \sqrt{p_2}\beta_2^* \end{bmatrix} \begin{bmatrix} \sqrt{p_1}\alpha_1 \\ \sqrt{p_1}\beta_1 \\ \sqrt{p_2}\alpha_2 \\ \sqrt{p_2}\beta_2 \end{bmatrix} \tag{4.12}$$

$$= p_1(|\alpha_1|^2 + |\beta_1|^2) + p_2(|\alpha_2|^2 + |\beta_2|^2) \tag{4.13}$$

$$= p_1 + p_2 = 1 \tag{4.14}$$

With this matrix/vector representation, it is pertinent to express the cost function as follows

$$f(\mathcal{W}) = \sum_i \left[d_i - \sum_\alpha \langle \sqrt{p_\alpha}\psi_\alpha | \, \hat{\Pi}_i \, | \sqrt{p_\alpha}\psi_\alpha \rangle \right]^2 + \lambda ||\mathcal{W}||_1 \tag{4.15}$$

and then apply Eq. (3.16).

4.3.3 Projective normalization

The last form of constraint explored is with the basic idea that in our system, the sum of all probabilities has to be 1. So, we take the two lists created for the ansatz: one with the complex-valued probability amplitudes (c_m) from all the states $|\psi_i\rangle$, and the other with the p_i which are the probabilities of finding the system in

$|\psi_i\rangle$. For the first list (C_l), after using GD with Adam, it is necessary to perform a normalization (a projection) to the new values, i.e., the new values are divided by $\sum_m |c_m|^2$. For example, in the case of 1-qubit pure state Eq. (2.2), after GD the state will be $|\Psi'\rangle = \alpha' |0\rangle + \beta' |1\rangle$, now we have to do the normalization

$$|\tilde{\Psi}\rangle = \frac{\alpha'}{|\alpha'|^2 + |\beta'|^2} |0\rangle + \frac{\beta'}{|\alpha'|^2 + |\beta'|^2} |1\rangle \tag{4.16}$$

In the general case of 1-qubit mixed states, a general ansatz represented by two states can be adopted. This is because, for 1 qubit, the density matrix is 2×2 indicating that the maximum rank is 2. The entire process would then appear as follows:

$$\begin{bmatrix} \alpha_1 \\ \beta_1 \\ \alpha_2 \\ \beta_2 \end{bmatrix} \xrightarrow{\text{Adam GD 1 step}} \begin{bmatrix} \alpha'_1 \\ \beta'_1 \\ \alpha'_2 \\ \beta'_2 \end{bmatrix} \xrightarrow{\text{Projection}} \begin{bmatrix} \frac{\alpha'_1}{|\alpha'_1|^2+|\beta'_1|^2} \\ \frac{\beta'_1}{|\alpha'_1|^2+|\beta'_1|^2} \\ \frac{\alpha'_2}{|\alpha'_2|^2+|\beta'_2|^2} \\ \frac{\beta'_2}{|\alpha'_2|^2+|\beta'_2|^2} \end{bmatrix} = \begin{bmatrix} \tilde{\alpha}_1 \\ \tilde{\beta}_1 \\ \tilde{\alpha}_2 \\ \tilde{\beta}_2 \end{bmatrix} \tag{4.17}$$

On the other hand, for the second list (P_l) which is the same probability list showed in Eq. 4.8, we applied the Adam GD; and after the update of the probabilities values the softmax function will be applied. This function rescales the elements of the list in a way that each element has values in the range [0,1], and that the sum of the elements is one. Since the list is a set of probabilities, then the softmax function suits perfectly well.

$$\text{softmax}(p_i) = \tilde{p}_i = \frac{e^{p_i}}{\sum_i e^{p_i}} \tag{4.18}$$

$$\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} \xrightarrow{\text{Adam GD 1 step}} \begin{bmatrix} P'_1 \\ P'_2 \end{bmatrix} \xrightarrow{\text{softmax}} \begin{bmatrix} \tilde{P}_1 \\ \tilde{P}_2 \end{bmatrix} \tag{4.19}$$

with the two lists, we can now express the cost function, for that one we will use the representation of the expectation value given in Eq. (2.12), which gives us

$$f(C_l, P_l) = \sum_i \left[d_i - \sum_\alpha P_{l\alpha} \langle \psi_\alpha | \hat{\Pi}_i | \psi_\alpha \rangle \right]^2 + \lambda ||C_l||_1 \tag{4.20}$$

and we already know how the Adam GD will update the values of the two lists and how we are constraining those values.

4.4 Input values

To start doing quantum state tomography it is necessary to define the system we intend to treat. Therefore, this work is divided into two parts, the first part consists of a discrete variable system (section 2.1), for 2, 3, 4, 5, and 6 qubits, but focuses

mostly on 4 qubits due to computational limitations. The second part takes a system of a cat state with a value of $\alpha = 2$. Based on these systems defined for discrete variable and continuous variable (section 2.4), it will be performed the benchmarking of the GD methods against the MLE and GAN models.

To use the algorithms we need some initial inputs. First, the measurement operators. Secondly, the **d**-values which are the average measure values of the original density matrix (the one we want to obtain). Finally, an ansatz of a density matrix. The elements and dimensions of the set of operators, of the **d**-values, and of the ansatz for the density matrix will depend on the system defined, in the case of this report those defined in discrete variable and continuous variable.

4.4.1 Input for discrete variables

In the case of discrete variables, the idea is to emulate what would be done in a laboratory, so in this case, they measure the values at the poles of Z and then rotate the qubit to look at the other values. Therefore, it is like taking the following set of measurement operators $\Pi_{1qubit} = \{|0\rangle\langle 0|, |1\rangle\langle 1|, |+\rangle\langle +|, |-\rangle\langle -|, |+i\rangle\langle +i|, |-i\rangle\langle -i|\}$ which are multiplied by a factor of 1/3 to be a POVM. In theory, it is possible to use a smaller set, which would give the advantage of having to calculate fewer averages, but that would put off the idea of trying to emulate what would be done in a laboratory for one or more qubits. For the case of more qubits (N qubits), we apply the tensor product between the POVM components, that procedure gives a total of 6^N measurement operators (1024 for 4 qubits). To give an example of how the measurement operators would look, we take the following case $|0\rangle\langle 0| \otimes |1\rangle\langle 1| = |01\rangle\langle 10|$.

Then a random density matrix is generated with dimensions of $2^N \times 2^N$, and it is taken as the original density matrix i.e. the one that we want to reconstruct. By taking the original matrix and the set of measurement operators we can obtain the **d**-values with the Eq.(2.16). Once we have the measurement operators and the d-values, it is time to generate the ansatz, as indicated in section 4.3. This ansatz has to be adapted to the different forms necessary to use the constraints correctly. In the Cholesky case, the lower triangular matrix $T_{\mathcal{C}}$ is provided (see subsection 4.3.1), for the manifold the vector $\mathcal{W}$ is constructed (see subsection 4.3.2), for the projection the lists P_l and C_l are composed, for MLE is by taking the two lists of the ansatz and then apply the Eq.(2.3) to obtain the matrix representation of ρ_{ansatz}. The ansatz is adapted so that these models start with the same values in order to compare the models in a more consistent way.

To simulate the ρ_{original}, to create the ansatz, and to construct the measurement operators, the QuTiP library for Python was used. QuTiP is an open-source framework for numerical modeling of quantum systems, enabling the development of simulations and calculations on the defined system. It is specifically designed for open quantum systems [61]. This framework generates random density matrices using the Ginibre ensemble and the algorithm of Ref. [62]. The Ginibre ensemble is a set

of random matrices whose entries are independent and identically distributed from the normal distribution. In the case of the random state vectors, they are generated by applying a Haar random unitary to a pure state vector, following the algorithm of Ref. [63]. This is based on the Haar measure, which is a measure that provides a natural probability distribution on the set of unitary matrices U(N) [63].

4.4.2 Input for continuous variables

Now that we know the inputs for the discrete variable case, we will see how some of these inputs change for the continuous variables case. For the case of continuous variables, the measurement operators are given by the Husimi function, computationally the system has to be limited. Therefore, a Hilbert space with dimension 32 is taken (the same size as in the case of 5 qubits in a DV system), with 1024 β displacements in a grid of dimensions 32×32, meaning that we have 1024 measurement operators. Once the measurement operators have been defined, we proceed in the same way as for the discrete variable case, by generating the original density matrix, which in this case is the cat state with $\alpha = 2$. Subsequently, the **d**-values of this cat state are calculated.

Finally, the ansatz is generated and adapted for the different methods, as in the case of discrete variables, with two differences. The first difference is that the size of the kets is 32, and the second is that now, in continuous variables, the CGAN method is introduced. CGAN is introduced in this case because the algorithm is designed and optimized for CV with a fixed Hilbert space. For other cases, qubits or a different Hilbert space, it would be necessary to restructure the code since the layers do not adapt automatically and easily to another system. In the CGAN method, it is not necessary to provide an ansatz since CGAN itself creates the density matrix with its generator layer.

It was decided to use the cat state and those values for the Husimi function to make an easier and more direct comparison of the results of the Ref. [46].

4.5 Benchmark

Now that we have defined how the different methods are going to be implemented for QST, it is important to define with which parameters we can compare the different methods in order to perform the benchmark of the three methods based on GD. Since all methods advance in steps to improve fidelity, the number of iterations that it takes to reach a certain fidelity value can be evaluated. In addition, the time spent in each iteration can also be known, allowing to know the time required to reach the desired fidelity.

Another parameter that can be measured would be the number of measurement operators needed to obtain a fidelity value close to 100% or close to an assigned threshold value, so one can randomly select some measurement operators, starting

from a relatively small number (depending on the total number of operators) and start increasing the number of operators selected.

For the case of DV, we will look at how many ansatz kets have to be generated to reconstruct the original mixed state. We will start with a low rank and increase it until we reach the maximum rank of the system. For example, in the case of 4 qubits, the maximum value would be a rank of 16. In CV, we will look at the range in another way. Because the cat state is a pure state, varying the range in CV will involve adding thermal states to the system. These new states are represented by Fock states, resulting in a mixed state. The mixed state would look like this:

$$\rho = 0.8 \left|cat\right\rangle \left\langle cat\right| + \frac{0.2}{r-1} \sum_{n=0}^{r-2} \left|n\right\rangle \left\langle n\right| \tag{4.21}$$

where r is the rank and it has values greater than 2 because in the case of $r = 1$ it is just the cat state.

So far, it has been considered that all measurements are "perfect". However, in reality, there will always be noise in the expected values. That noise is present due to the measurement signal distribution and the discriminator value taken in the experiments to indicate the regions of the states. Therefore, to emulate the noise that may occur in the measurement of expectation values, uncertainty is added to the **d**-values by incorporating random values given by a Gaussian distribution with a mean of 0 and a standard deviation of σ.

$$\mathbf{d}_{noise} = \mathbf{d} + \mathcal{N}(0, \sigma) \tag{4.22}$$

with this new case of noise addition, we will look at achievable fidelity values and the number of iterations needed to reach the desired fidelity value.

Table 4.1: Values taken for the hyperparameters λ, η, decay of learning rate, size of mini-batches, for the cases of DV and CV.

Methods/Hyperparam	λ	η	decay learning rate	mini-batches size
GD-Cholesky DV	10^{-7}	5×10^{-2}	0.999	0.55× length(**d**)
GD-Manifold DV	0	10^{-2}	0.999	0.55× length(**d**)
GD-Projective DV	0	10^{-1}	0.099	0.55× length(**d**)
GD-Cholesky CV	10^{-9}	10^{-1}	0.999	0.55× length(**d**)
GD-Manifold CV	10^{-4}	5×10^{-2}	0.999	0.55× length(**d**)
GD-Projective CV	10^{-5}	5×10^{-2}	0.009	0.55× length(**d**)

Finally, in order to start benchmarking the different aspects mentioned above, it is necessary to define the hyperparameters (λ, η, decay of learning rate, size of mini-batches) that the GD models need, these are selected after running the models 5 to

10 times for each variation of the hyperparameter value, and the values with which better reconstruction of the density matrix are taken. This analysis and the graphs illustrating the values taken can be found in Appendix B. Table 4.1 shows the values taken.

For all performance metrics to be looked at, a statistic has to be taken to be able to extract a concrete result. This is why the algorithms have to be run several times. A run in the case of DV means to create a random matrix as the original one and to create a random ansatz (see section 4.3), and then with the different methods perform QST. In the case of CV, a run is to make the cat state with $\alpha = 2$ and a random ansatz, to which the methods MLE, GD, and CGAN are applied to reconstruct that cat state.

5
Results

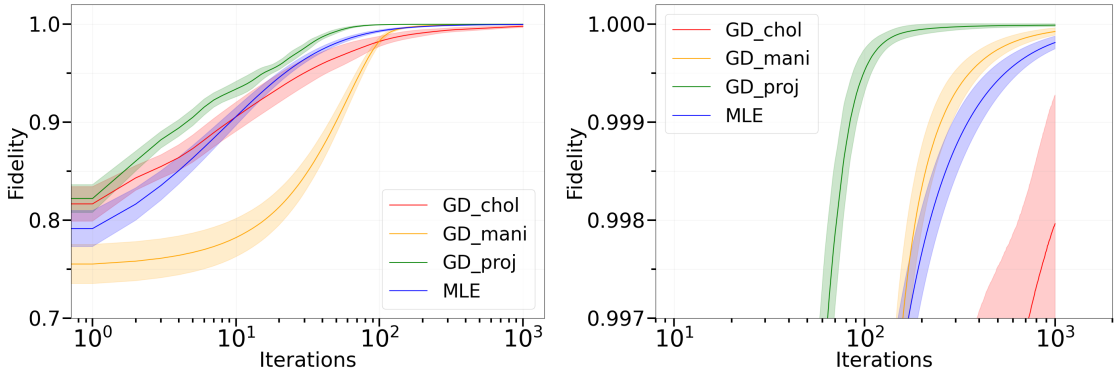
The complexity of quantum state tomography has forced the scientific community over the years to explore different methods of performing a correct reconstruction of the system state, and also to be an optimal method in terms of time, memory management, amount of data, and so on. The fact that an optimal method is desired in several performance metrics, makes this a topic with a lot of room for exploration. Therefore, the search for improvement in any of the aspects that have made the reconstruction of quantum states difficult has been a great motivation to propose and explore the GD algorithm with its different constraints.

In this chapter, we will compare the results obtained for the three cases (Cholesky, manifold, projective) of gradient descent with the MLE of Ref. [24] model for the discrete variable case. And, for the case of continuous variable, we will compare gradient descent against MLE of Ref. [23] and CGAN of Ref. [46] (see subsection 4.4.2). With these comparisons, we seek to benchmark the techniques proposed in this thesis with respect to the other models. The benchmark will be performed taking into account the time and number of iterations required to reach a fixed fidelity value, the number of kets needed to generate the ansatz of the density matrix and to obtain a fidelity close to 100%, the fidelity after adding some thermal state, and lastly, the fidelity that can be obtained in cases with noise.

5.1 Quantum state tomography for discrete variable

The performance of MLE, GD Cholesky, GD manifold, and GD projection methods for the first benchmarking case with 4 qubits is presented in Fig. 5.1. The data shown are from 10 runs for each algorithm, in which it is observed that all methods reach a fidelity close to 100%. More specifically, the final fidelities of the four methods after 10^3 iterations are greater than 99.7%. With the possibility that by performing more iterations a value of fidelity closer to 100% will be reached, however, to reduce time and processing power we will limit the benchmark to 10^3 iterations.

Figure 5.1b shows that the GD-projective method reaches the value of 0.998 with fewer iterations than the other three cases. This projective one is followed by GD-manifold, then very close to manifold is MLE, and finally the case of GD-Cholesky which on average does not reach 0.998 with 10^3 iterations, but reaches a value close to it. In order to simplify the comparison, we will take 10^3 iterations as the value



(a) **(b)**

Figure 5.1: Reconstruction of a random density matrix for a system of 4 qubits, comparing the reconstructed density matrix with the original by the fidelity operation. The fidelity depends on the iterations for the cases of GD-Cholesky (red), GD-manifold (yellow), GD-projective (green), and MLE (blue). The solid lines are the mean fidelity for 10 runs and the shaded areas are the standard deviation of each mean. (a) shows the fidelity in the range between 0.7 - 1.00. (b) fidelity values in the range of 0.997 to 1.00.

at which GD-Cholesky reaches a fidelity of 0.998. Having the number of iterations is not enough to define which of the four methods is better for this case; we also need to look at each method's average time per iteration. These values of time and number of iterations are summarized in table 5.1.

Table 5.1: Values of average time per iteration, number of iterations to reach a fidelity of 0.998, and the total time that the four methods need to reach the 0.998 fidelity. For the case of 4 qubits.

Method	Average time per iteration (ms)	No. of iter for 0.998 fid	Total time (ms)
Chol	2.3912	1000	2391.2
Mani	9.826	185	1817.9
Proj	17.574	71	1247.7
MLE	6.478	220	1425.2

The data shown in Table 5.1 indicates that the fastest method per iteration is GD-Cholesky and the slowest is GD-projective, which is the complete opposite of what was seen with the measure of iterations. Taking into account both measures, it can be seen that for 4 qubits the fastest method to reach the fidelity threshold of 0.998 was GD-projective, it is approximately 12% faster compared to the second fastest method which was the MLE. In Fig. 5.1b it can also be seen that with 10^3 iterations the GD-projective method comes close to convergence to a value closer to 1.0. The other methods will require more iterations and time to achieve values closer to 1.0.

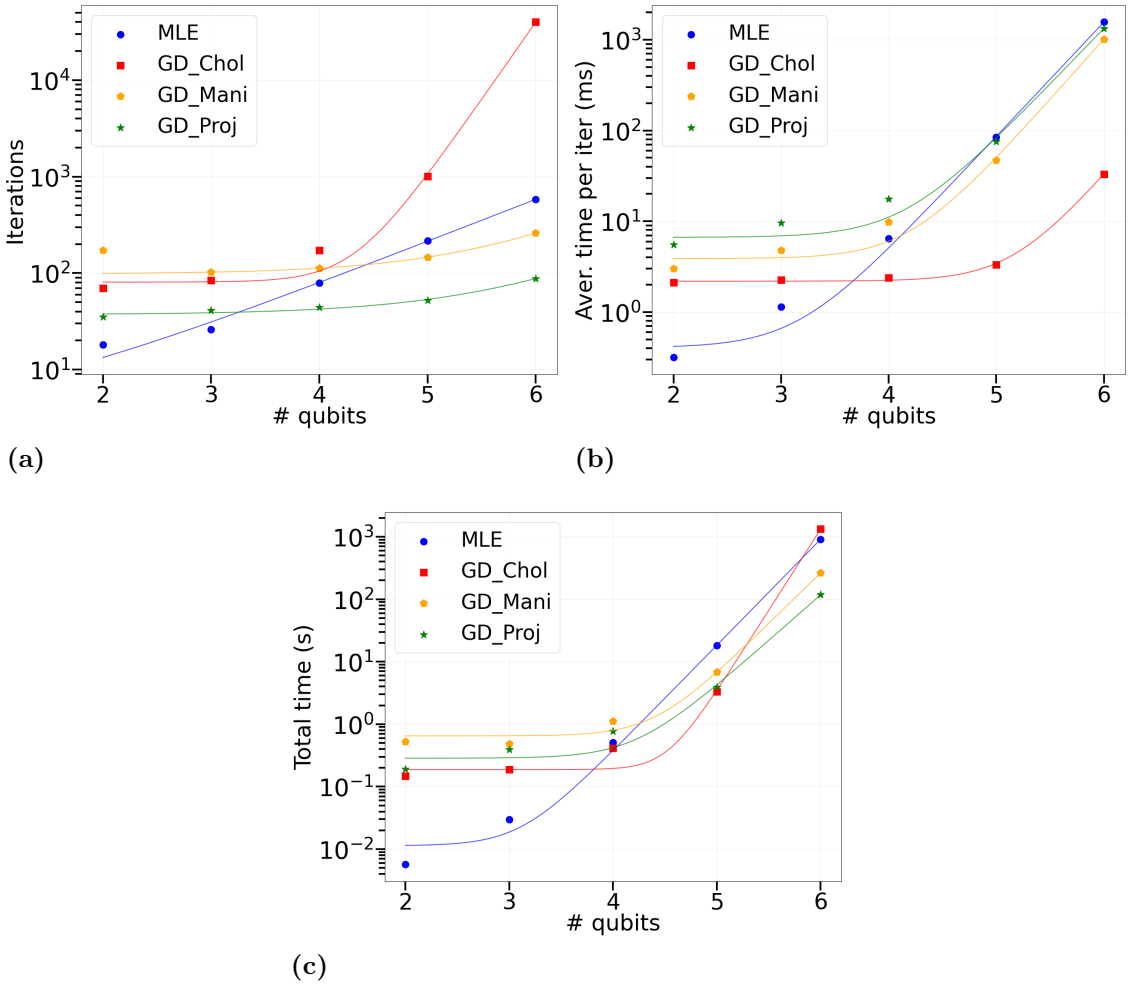


Figure 5.2: Performance in the reconstruction of a random density matrix with a fidelity value of 0.99 for the cases of 2, 3, 4, 5, and 6 qubits, using MLE, GD-Cholesky, GD-manifold, and GD-projective. For each case, we run the methods 10 times. (a) average number of iterations needed to obtain the fidelity 0.99 and (b) the average time per iteration. (c) total time for each of the methods to achieve a fidelity of 0.99.

The analysis of the number of iterations needed to reach a certain fidelity value is extended to the case of 2, 3, 5, and 6 qubits. Fig. 5.2 shows how the number of iterations and the average time per iteration changes with respect to the number of qubits to reach the fidelity value of 0.99. We reduced the fidelity value to 0.99 for simplicity because arriving at 0.998 in 5 and 6 qubits required a lot more iterations and time. In the case of iterations, Fig. 5.2a, it shows how GD-Cholesky is growing extremely fast, while on the other side, GD-projective has not even passed to the next order of magnitude (10^2). The MLE case also starts to have a large growth in the necessary iterations going from magnitude 10^1 to 10^2 and finally GD-manifold stabilizes at an order of magnitude 10^2. The difference between the projective method with respect to the restrictive search methods can be due to the fact that by updating the values it goes out of the physical subspace, and then by projection bringing it back to the

physical subspace it can take "shortcuts" or be projected very close to the desired value. In the other three methods, being restricted to move only in that subspace limits the paths they can take.

In the case of the average time, it is quite the opposite, the GD-projective, GD-manifold, and especially MLE methods have a very large growth when the number of qubits is increased. While the GD-Cholesky method has a much smaller growth, which in a way compensates a bit for the number of iterations that it has to perform to reach the fidelity value of 0.99. These differences can be due to various reasons. In the case of MLE, it has to perform three matrix multiplications to update the density matrix, a division for the R operator that must be checked to avoid divergences (section 4.1), and then the operation of all the average measurement operators, which is a multiplication of two matrices (ρ and Π_i) per measurement operator. For the case of GD-projective and GD-manifold, each average measurement operator must be calculated with equation (2.12). This involves the product of two vectors with a matrix that has to be done m times, m being the rank number, and this is for each average to be calculated. Additionally, in GD-projective, two gradients must be calculated, one for each list. Finally, in the case of GD-Cholesky, only a multiplication of two matrices is needed to obtain ρ and then operate all the average measurement operators, which is a multiplication of ρ and the measurement operator. It is known that multiplying matrices is computationally expensive, so the method that requires fewer multiplications is the one that requires less time.

Table 5.2: Values of the fit for the cases of iterations, average time per iteration, and total time for the reconstruction of the DV system with the function $f(N) = ae^{cN} + b$.

Method	c	a	b	Method	c	a	b
Iter: MLE	1.01	1.35	3.16	Aver: GD-mani	3.07	1.0×10^{-5}	3.87
Iter: GD-Chol	3.70	9.1×10^{-6}	80.6	Aver: GD-proj	2.85	5.0×10^{-5}	6.65
Iter: GD-mani	1.21	0.11	98.1	Total: MLE	3.91	6.0×10^{-8}	1.1×10^{-2}
Iter: GD-proj	1.14	0.05	35.9	Total: GD-Chol	5.99	3.2×10^{-13}	0.19
Aver: MLE	2.91	4.2×10^{-5}	0.40	Total: GD-mani	3.74	4.6×10^{-8}	0.65
Aver: GD-Chol	3.2	1.4×10^{-7}	2.2	Total: GD-proj	3.38	1.8×10^{-7}	0.29

The total time for each method to reach a fidelity value of 0.99 is illustrated in Fig. 5.2c. It shows that the MLE and Cholesky methods start being the fastest, however after 5 qubits MLE starts to be the slowest one, and then at 6 qubits, GD-Cholesky becomes the most time-consuming method to reach the set value. By contrast, GD-manifold and GD-projective start being the slowest methods, but as the system dimensions increase, they begin to show an advantage in the sense that they start to require less time than MLE and GD-Cholesky by almost an order of magnitude.

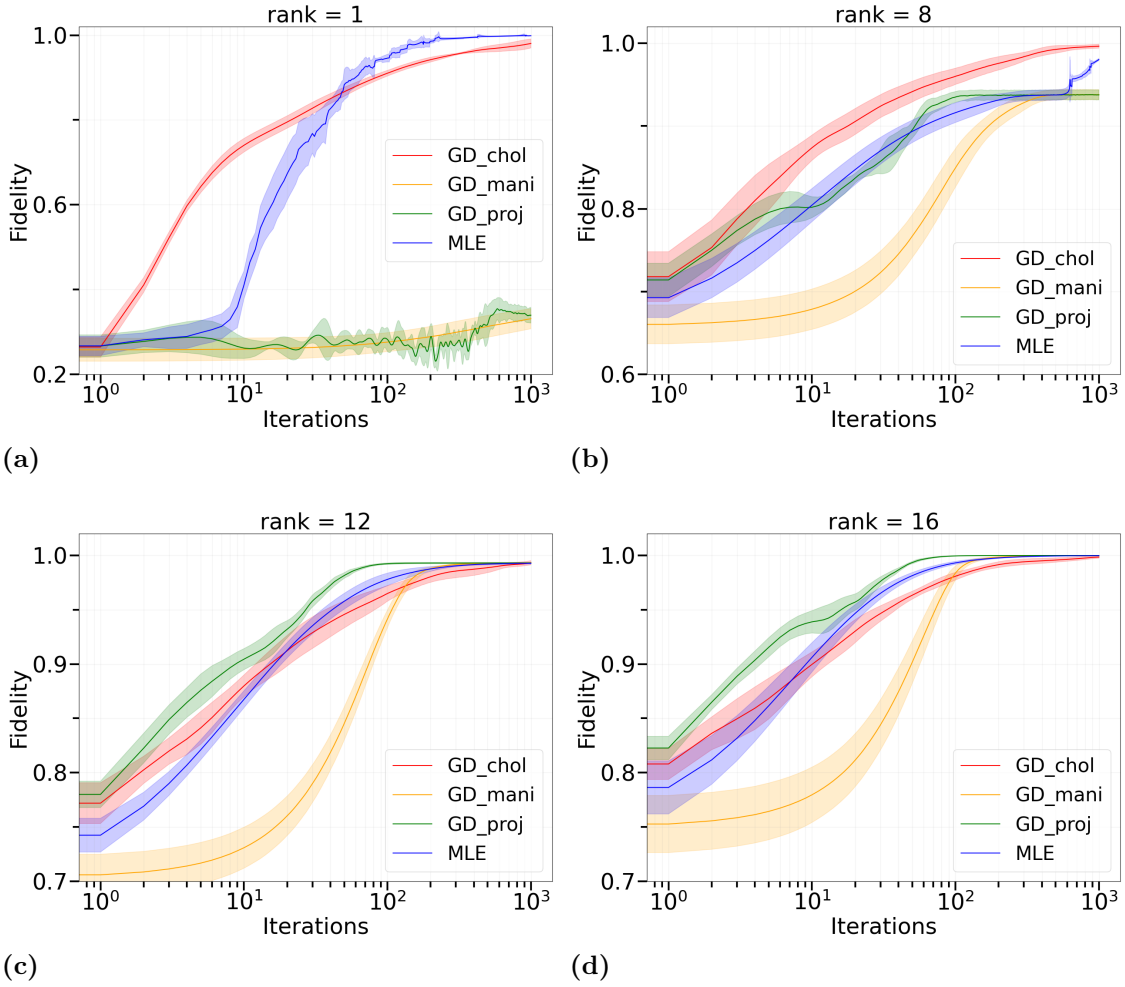


Figure 5.3: Fidelity values for the cases of the ansatz density matrix with different rank values for the reconstruction of a random density matrix of a system of 4 qubits. The 4 methods performed 10 runs for each rank, and the mean fidelity of the runs is represented by the solid lines. (a) Rank 1, (b) rank 8, (c) rank 12, (c) rank 16.

To get an idea of the growth of the time and iterations, we attempted to fit some functions based on the values obtained in Figures (5.2a, 5.2b, 5.2c). It would be expected that since the quantum state grows exponentially, then the growth of the algorithm should also be exponential. Therefore, in all these cases, we fit a function of the form $f(N) = ae^{cN} + b$, obtaining R-squared values greater than 0.99 in all cases. This may not be the best model to define how the time values of the algorithm grow, but it gives an idea of the growth trend of the system and tells us which method tends to have a higher growth rate in case we want to know the behavior for more qubits. For a more accurate model, we should do an analysis of the complexities of the algorithms, see how they interact with the growth of the quantum system, and have more data since 5 points may be too few values for a correct fit. The values of the fit are shown in the table 5.2. By looking at the case of total time we can again see that the method with the lowest growth is the GD-projective case, while GD-Cholesky has the highest growth trend.

5.1.1 Ansatz rank

Turning to the case where the ansatz is constructed by varying the number of kets which is represented by the rank number r, Fig. 5.3 shows that by starting with a very small r value the only methods able to reach a value relatively close to 1.0 are MLE and GD-Cholesky because they are represented in matrix form, which allows them to rank up by updating their values. Whereas GD-manifold and GD-projective models cannot increase the rank and are too limited in the reconstruction. Then, as the value of the rank increases, the methods improve and become more reliable, giving values close to 1.0 for all methods. The best case is when $r = 16$ which is the maximum rank value that can be taken for the 4-qubit system.

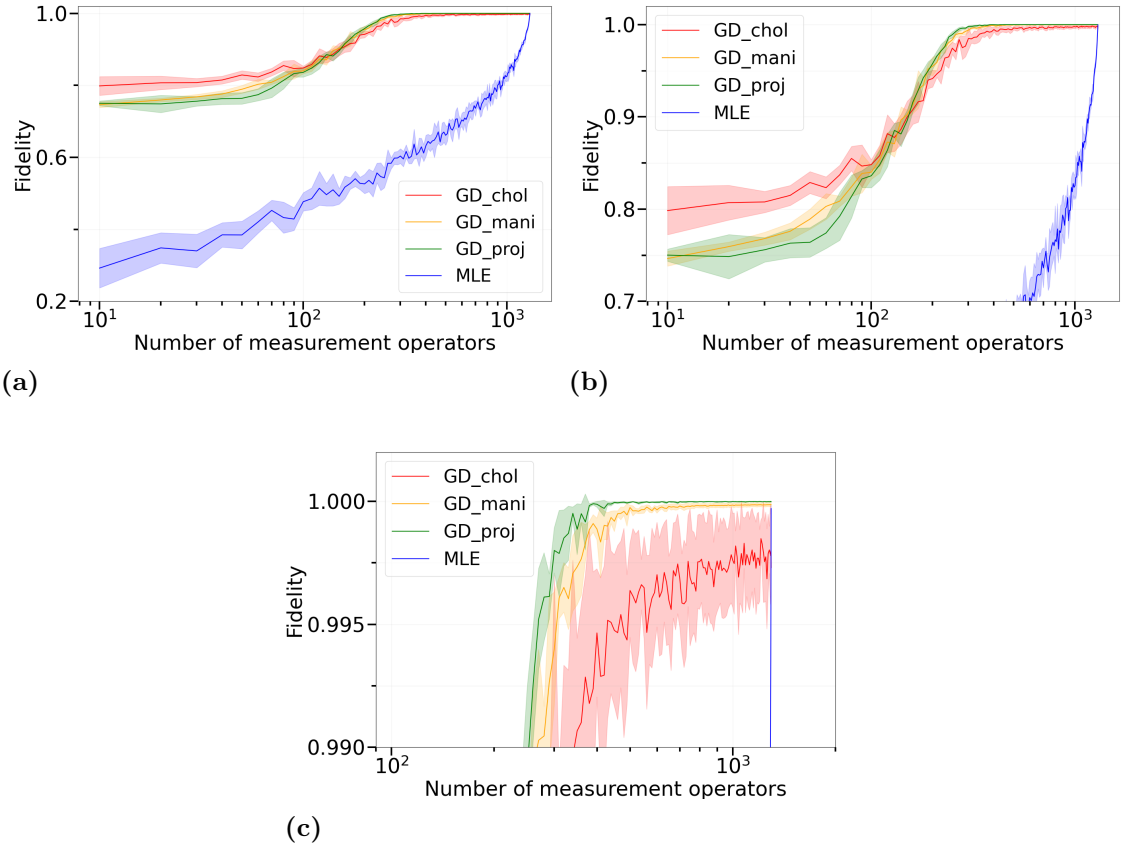


Figure 5.4: Average fidelity for the case of increasing the number of measurement operators. The x-axis increases by 10 randomly selected measurement operators. All the methods were fixed to 10^3 iterations and executed 10 times. (a) Full fidelity plot, (b) zoomed-in for fidelity values ranging from 0.7 to 1.0, (c) zoomed-in for fidelity values ranging from 0.990 to 1.000.

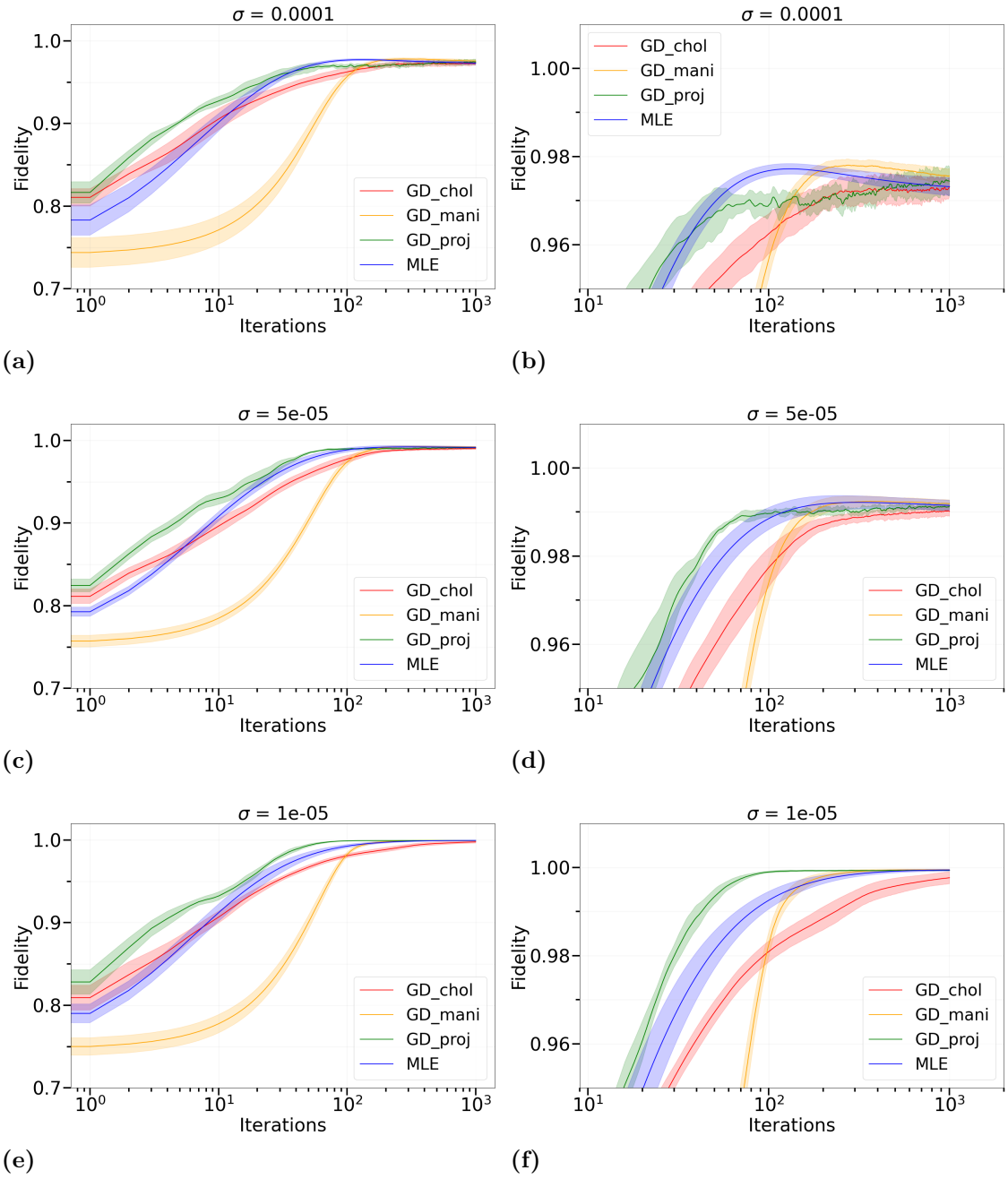


Figure 5.5: The effect of adding Gaussian noise in the average fidelity values for the case of 4 qubits. The noise has 3 different variances. (a) $\sigma = 10^{-4}$, (b) zoom-in of $\sigma = 10^{-4}$, (c) $\sigma = 5 \times 10^{-5}$, (d) zoom-in of $\sigma = 5 \times 10^{-5}$, (e) $\sigma = 10^{-5}$, (f) zoom-in of $\sigma = 10^{-5}$. Each case includes 10 runs per method.

5.1.2 Number of measurement operators and noise

We now look at how the four methods behave when the number of measurement operators is limited, which in other words, is limiting the information available about the original system. Fig. 5.4 shows how the methods improve as more measurement operators are added, but it is evident that the case of MLE is very dependent

on needing all the information from the original system (all the set of measurement operators see subsection 4.4.1) to reach a fidelity value above 0.99. While GD-projective requires 21%, GD-manifold requires 24%, and GD-Cholesky requires 48% of the measurement operator set to reach a fidelity of 0.995 or higher. The big difference between MLE and GD models is that MLE looks for the parameter that maximizes the likelihood function. Not having the complete description of the system, it finds the parameter that maximizes that subsection, which is not necessarily the one that maximizes the likelihood of all the average measurement operator values. This means that it finds a different ρ than the one to be reconstructed.

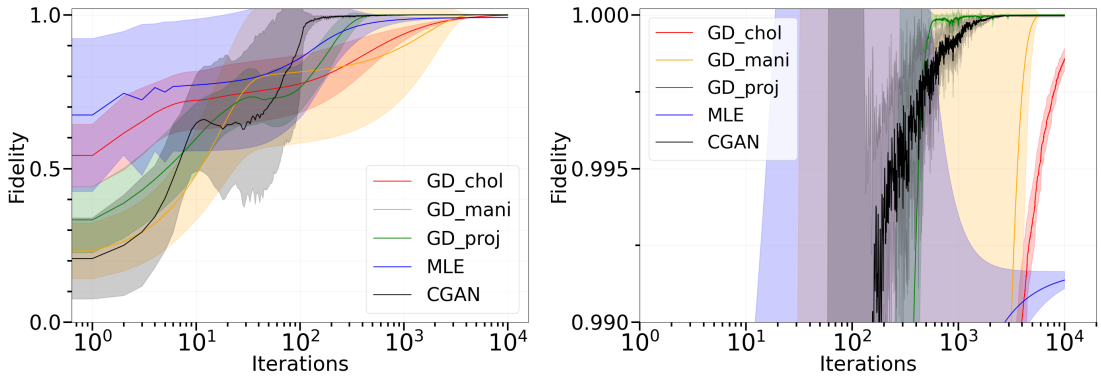
This leads us to wonder whether it is better to have more measurements on fewer measurement operators or to have fewer measurements on each measurement operator but have more of them. Having too few measurements per measurement operator can lead to a large statistical noise σ, making the QST task more difficult. To observe how the methods for QST behave in the presence of noise, we take 3 different values of σ, which were chosen based on the maximum value in the d-values. This means that $\sigma/\max(\mathbf{d}\text{-values})$ in a range between 0.09 to 0.001 with $\max(\mathbf{d}\text{-values})=$ 0.001542, in order to have a noise that does not change abruptly all the data and make them look like a different system. That is why we take $\sigma = 10^{-4}, 5\times10^{-5}, 10^{-5}$.

From Fig. 5.5 we observe that for $\sigma = 10^{-4}$ it is complicated for all the methods to reach a fidelity value higher than 0.98. For the second case with $\sigma = 5 \times 10^{-5}$, they reach a value of 0.99 but do not extend to higher values. In the case of $\sigma = 10^{-5}$ we can see values much closer to 0.999 but not as high as those obtained in the first case without noise represented in Fig. 5.1a. Also comparing with that first case, the behavior of the different methods are very alike, with similar times and a similar number of iterations.

5.2 Quantum state tomography for continuous variables

Moving on to the case of CV, where the big difference that will be presented in this analysis with respect to the case of DV is the inclusion of the CGAN method. Figure 5.6 shows the large contrast of CGAN compared with MLE and GD-based methods. CGAN reaches fidelity values of 0.99 with a small number of iterations in comparison to the other methods. Considering that machine-learning models based on neural networks are a kind of black box, it is not possible to know precisely why there is such an advantage in the case of the number of iterations. The number of iterations for the other methods shows a behavior similar to the DV case, where the "shortcuts" that can be taken due to the projections mean that fewer steps are needed to reconstruct the state.

By contrast, the CGAN method is quite slow, this can be seen in Table 5.3 which illustrates the average time per iteration of the 5 methods being compared. Of these 5 methods, the one that needs less time in total to achieve a value of 0.99 is the



(a) **(b)**

Figure 5.6: Reconstruction of the cat state with $\alpha = 2$, with the distance measurement of the fidelity depending on the iterations for the methods CGAN (black), MLE (blue), GD-Cholesky (red), GD-manifold (orange), and GD-projective (green). Average of 10 runs. (a) Full fidelity range and (b) fidelity values in the range of 0.990 to 1.000.

GD-projective, which is below the time of the others almost by an order of magnitude. Another aspect that can be noticed about GD-projective is that it can be said that it reaches a value closer to 1.0, but it presents a fidelity that exceeds the value of 1 by a factor of magnitude 10^{-7}, which would be incorrect. However, upon examining the density matrices reconstructed by this method, they fulfill the three characteristics of a physical density matrix. Moreover, by comparing the values of the components of the original density matrix with those of the reconstructed one, it is found that the values are the same or very close; the difference for those that are not identical is on the order of magnitude of 10^{-15}. This indicates that a correct reconstruction of the system has been achieved. The value above 1 can be due to different reasons, the first one is that when the original matrix is created in Qutip it can be the case that some eigenvalues are -10^{-15} when they have to be 0 and be positive. The second reason is that in the calculation of the fidelity, which is also done by QuTip, there are errors for those small magnitudes.

Table 5.3: Values of average time per iteration, number of iterations to reach a fidelity of 0.99, and the total time that the four methods need to reach the 0.99 fidelity. For the case of 4 qubits.

Method	Final average fidelity	Average time per iteration (ms)	No. of iter in 0.990 fid	total time (ms)
Chol	0.9985	32.6605	3950	129009.2
Mani	0.9999	19.1945	3154	60539.5
Proj	1.0000	21.1086	380	8021.2
MLE	0.9913	25.9182	2738	70964.0
CGAN	0.9999	210.5401	185	38949.9

By looking at the last values that the methods can obtain, it can be seen in Fig. 5.6b

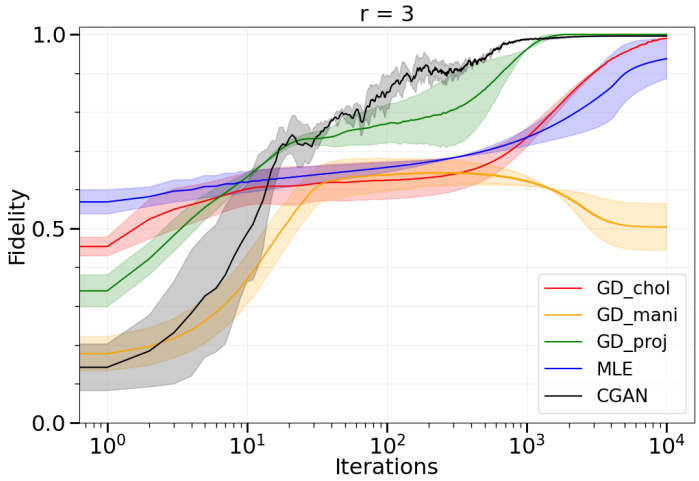


Figure 5.7: Reconstruction of a mixed state of continuous variables following Eq. (4.21) with $r = 3$ for the original density matrix and for the ansatz.

that the MLE method does not reach values above 0.995, while the other methods do. By observing the trend of MLE it can be seen that it has no more growth to reach higher values even if the number of iterations is increased.

5.2.1 Mixed state continuous variables

Turning to the case where it is now a mixed state with a rank value of 3, Eq. (4.21), Fig. 5.7 illustrates how the only cases to reach values close to 1 are the GD-projective, the CGAN method, and the GD-Cholesky method. This is followed by MLE which would need more iterations for a better fidelity value. Finally, the case of GD-manifold has to be further explored because it is not doing a proper reconstruction. This may happen because it is based on a vanilla GD model and the only way to escape from local minima is through the mini-batches, so probably to improve the result the size of the mini-batches could be tuned for mixed CV cases.

5.2.2 Number of measurement operators for continuous variable

In Fig. 5.8 we consider the case of varying the number of measurement operators, so it is observed that the GD-Cholesky, CGAN, and GD-projective methods are the ones that need less measurement, but the only relatively reliable one would be the GD-Cholesky because the other two have many fluctuations in the fidelity values, therefore they depend a lot on the selected measurement operators. For MLE and GD-manifold it is more noticeable that they need the whole set of measurement operators to perform a proper reconstruction. This was expected for the case of MLE and it was also observed in DV. An abrupt change in comparison with DV is the necessary number of measurement operators for GD-manifold; in this model,

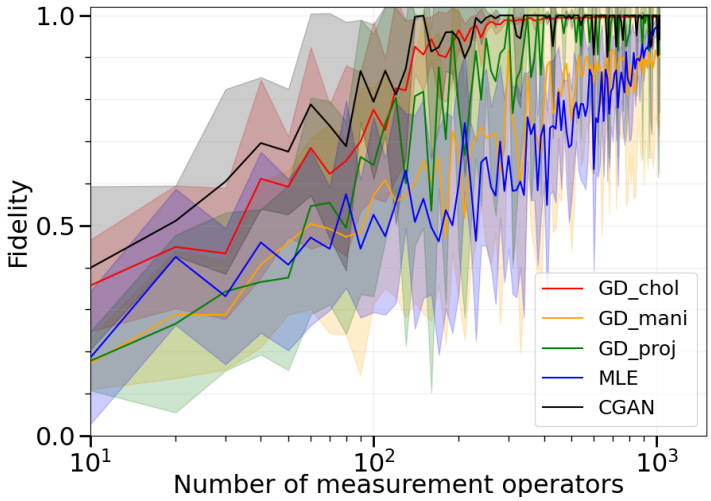
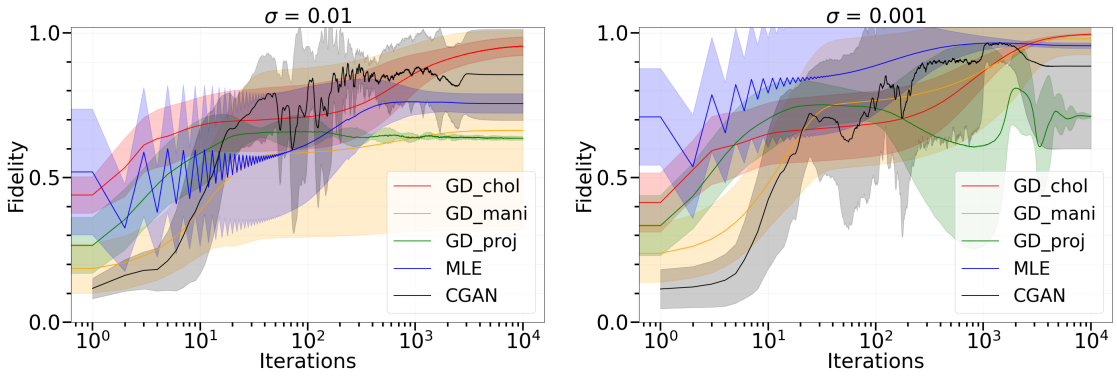


Figure 5.8: Average fidelity in relation to the increasing number of measurement operators. The x-axis represents increments of 10 randomly selected measurement operators. CGAN was performed with 10^3 iterations and the other methods with 10^4 iterations.

it is given that all measurement operators are needed, but also with the standard deviation of the data, it can be seen that it is probably possible to make a correct reconstruction of the original state with fewer number of measurement operators. Therefore, we would need more statistics and possibly another calibration of the hyperparameters to obtain a better idea of the behavior of GD-manifold for CV.



(a) **(b)**

Figure 5.9: Fidelity reconstruction for the cat state after the addition of noise by a Gaussian function. (a) For a variance of 0.01, and (b) a variance of 0.001.

5.2.3 Noise for continuous variables

For the final case of adding Gaussian noise, the values are taken in a similar way as with DV, where the maximum value of the **d**-values is 0.47364. By taking σ with

values of 0.01 and 0.001, it would give that the ratio of σ and the maximum value is in the range of 0.002 to 0.02. Looking at Fig. 5.9 the method that best fits these noises is the GD-Cholesky method achieving values above 0.95 for $\sigma = 0.01$ and values very close to 1 ($f > 0.99$) for $\sigma = 0.001$. The other methods fail completely for the first σ value. For a smaller noise, which is the case of the second σ, GD-manifold comes quite close to reconstruction obtaining fidelity values greater than 0.97. MLE also comes close but does not exceed fidelity values of 0.95. Finally, CGAN and GD-projective are not very stable, so they do not have a tendency only to grow. Also, they cannot reconstruct anything close to the original density matrix.

6

Conclusion and outlook

In this thesis, we proposed three different methods for quantum state tomography based on gradient descent (Cholesky, manifold, and projective). These methods achieve fidelity values of at least 0.99 in most cases, meaning that they are able to perform a good reconstruction of quantum systems. Looking at the case of a system defined by qubits and comparing it with the MLE method that is the standard in the field, we start to notice an advantage of the GD-projective and GD-manifold methods at 5 qubits and above with respect to MLE, at least in the case of total time required and number of iterations to reach a fidelity value of 0.99, illustrated in Fig. 5.2c.

In the case of 4 qubits, there is an advantage with GD-projective when it comes to the total time to reach a value even closer to 1 ($f > 0.998$). Therefore, from 4 qubits onwards, the GD-projective method could be taken as the best method to do QST in terms of time, followed by the GD-manifold method which also manages to outperform the MLE method as the system becomes larger.

The big advantage of these GD methods is when we look at the amount of measurement operators needed to achieve a reconstruction of the original system, as shown in Fig. 5.4, which leaves the question of whether it is better to have more statistics on a small set of measurement operators, or less statistics on each operator but taking a set with more measurement operators. From the experimental perspective, taking fewer measurement operators would save time. Other advantages may exist from this perspective, but that depends on each experiment. As I did not have direct contact with experimental procedures, I cannot conclude more from the experimental point of view. From the computational side, it would be better to have fewer measurement operators due to the amount of memory and processing time issues.

Regarding the behavior of the methods with respect to the noise generated by a Gaussian function for the case of 4 qubits, it has been found that in cases where σ is large (10^{-4}), none of the methods can achieve a proper reconstruction. Therefore, σ needs to have a value of 5×10^{-5} or smaller to be able to do a proper QST. It is not possible to say that one of the methods is better than the others, they are all very even and manage to have a good reconstruction of the system as long as the noise is not too large. The last aspect of the DV case is that it is necessary to create an ansatz with the maximum value of rank to achieve a correct reconstruction of the original system of qubits for a random density matrix as the original. In the case that, in an experiment, we know that the original system is not full-rank and

we have knowledge of the approximate rank, we can adapt the ansatz to have that rank value, allowing us some flexibility in designing the ansatz.

On the CV side, in which the GD methods are now compared against the CGAN and MLE methods, it is found that GD-projective needs less time to achieve fidelity values of at least 0.99 compared to the other four methods, at least for the case of the cat state with the full set of measurement operators. However, if we take into account all the other cases that were explored such as the rank, and the number of measurement operators, the most reliable method is GD-Cholesky because in all these cases it achieves an adequate reconstruction. The other methods fail in some of these cases. The most important feature of GD-Cholesky is that it stands out in the cases of the number of measurement operators and noise. By requiring few measurement operators, as is the case with CGAN, it can give certain advantages in the experimental aspect with respect to the time to make the measurements, as discussed for DV. Although the case of CGAN also requires a few measurements, it is not as consistent as GD-Cholesky. Finally, by being exposed to noises the only method that can correctly reconstruct the density matrix under these conditions is GD-Cholesky. Even though GD-Cholesky always requires more time, it is known that at least it will perform a correct reconstruction.

Comparing specifically the GD methods with the CGAN case, we can see other advantages from the technical side. The concept, implementation, and coding of the GD methods are much simpler than the CGAN case, which requires more knowledge to understand it, to be able to create a code of that complexity and to interpret it because the methods based on machine-learning are like black boxes. Besides that CGAN has the great disadvantage of being very rigid in the sense that it only accepts as input a system with a Hilbert space of a fixed number (see subsection 4.4.2), in order to use it in a system with other dimensions it would be necessary to change several terms of the layers which makes it a not very versatile method. On the contrary, the GD methods are very adaptable and adjust to any system dimension without the need to change the code.

6.1 Further research

The analysis of these methods could be extended to the case of more qubits, but it would require more computational power since a 40 GB GPU could only process up to the case of 6 qubits. Another improvement that would be required is to increase the statistics of each case, i.e. run more times the code to have more value for the average and the standard deviation of this. The results obtained in this thesis present in many cases a large standard deviation, so ideally it would be better to have more statistics. But even so, with the statistics taken in this thesis, we can already see patterns of how the proposed QST methods behave. In Ref. [31] they have 100 runs, so it would be good to run the GD methods 100 times.

Concerning the case of DV, it would be convenient to compare GD models against superfast MLE [25], compressed sensing (CS) [64], and projected gradient descent [65].

Ideally adapting the codes to run in Python and that they start with the same ansatz for all models, to make the comparison fair.

For the scope of CV, it would be ideal to see other states that are not only the cat state, so there are still several tests to be able to reach general conclusions on continuous variables.

It would be desirable that the three proposed methods could be implemented with real values given from the experiments. Nevertheless, in that case, it would be necessary to change the way in which we look if the reconstruction was done properly because we do not know the original state. One way that could be used is to look at the loss function and see if it becomes 0 or very close to 0.

From this work, we cannot say that GD is the definitive method. There could be an algorithm with advantages over GD in some parameters, or perhaps a different form of constraint could be found that is more efficient for GD. It is also possible that there is another version of the GD algorithm, such as AdaMax, Adadelta, or some other that converts the algorithm faster to the desired minimum.

Bibliography

[1] Helge Kragh. "Max Planck: the reluctant revolutionary". In: *Physics World* 13.12 (2000), p. 31.

[2] Hermann Haken and Hans Christoph Wolf. *The physics of atoms and quanta: introduction to experiments and theory*. Springer Science & Business Media, 2006.

[3] Mark Thomson. *Modern particle physics*. Cambridge University Press, 2013.

[4] Georges Aad et al. "Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC". In: *Physics Letters B* 716.1 (2012), pp. 1–29.

[5] Serguei Chatrchyan et al. "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC". In: *Physics Letters B* 716.1 (2012), pp. 30–61.

[6] Mitsuru Kikuchi. *Fusion physics*. International Atomic Energy, 2002.

[7] Raymond Murray and Keith E Holbert. *Nuclear energy: An introduction to the concepts, systems, and applications of nuclear processes*. Elsevier, 2014.

[8] Efthimios Kaxiras and John D. Joannopoulos. *Quantum Theory of Materials*. Cambridge University Press, 2019.

[9] David J Griffiths and Darrell F Schroeter. *Introduction to quantum mechanics*. Cambridge university press, 2018.

[10] Jun John Sakurai. *Modern quantum mechanics; rev. ed.* Addison-Wesley, 1994.

[11] Markus Bursch et al. "Best-practice DFT protocols for basic molecular computational chemistry". In: *Angewandte Chemie International Edition* 61.42 (2022), e202205735.

[12] Sumit Das et al. "Applications of artificial intelligence in machine learning: review and prospect". In: *International Journal of Computer Applications* 115.9 (2015).

[13] Gregg Jaeger. "Classical and quantum computing". In: *Quantum Information: An Overview* (2007), pp. 203–217.

[14] Richard P. Feynman. "Simulating Physics with Computers". In: *International Journal of Theoretical Physics* 21.6 (June 1, 1982), pp. 467–488. ISSN: 1572-9575. DOI: 10.1007/BF02650179.

[15] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[16] John Preskill. "Simulating quantum field theory with a quantum computer". In: *arXiv preprint arXiv:1811.10085* (2018).

[17] H Jeff Kimble. "The quantum internet". In: *Nature* 453.7198 (2008), pp. 1023–1030.

[18] Yudong Cao et al. "Quantum chemistry in the age of quantum computing". In: *Chemical reviews* 119.19 (2019), pp. 10856–10915.

[19] Sarah J Trenfield et al. "Advancing pharmacy and healthcare with virtual digital technologies". In: *Advanced Drug Delivery Reviews* 182 (2022), p. 114098.

[20] Daniel J Egger et al. "Quantum computing for finance: State-of-the-art and future prospects". In: *IEEE Transactions on Quantum Engineering* 1 (2020), pp. 1–24.

[21] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

[22] Samuel L Braunstein and Peter Van Loock. "Quantum information with continuous variables". In: *Reviews of modern physics* 77.2 (2005), p. 513.

[23] Alexander I Lvovsky. "Iterative maximum-likelihood reconstruction in quantum homodyne tomography". In: *Journal of Optics B: Quantum and Semiclassical Optics* 6.6 (2004), S556.

[24] Tillmann Baumgratz et al. "A scalable maximum likelihood method for quantum state tomography". In: *New Journal of Physics* 15.12 (2013), p. 125004.

[25] Jiangwei Shang, Zhengyun Zhang, and Hui Khoon Ng. "Superfast maximum-likelihood reconstruction for quantum tomography". In: *Physical Review A* 95.6 (2017), p. 062336.

[26] Zdeněk Hradil et al. "3 maximum-likelihood methods in quantum mechanics". In: *Quantum state estimation* (2004), pp. 59–112.

[27] Robin Blume-Kohout. "Optimal, reliable estimation of quantum states". In: *New Journal of Physics* 12.4 (2010), p. 043034.

[28] Christopher Granade, Joshua Combes, and DG Cory. "Practical bayesian tomography". In: *new Journal of Physics* 18.3 (2016), p. 033024.

[29] Christopher Ferrie and Robin Blume-Kohout. "Maximum likelihood quantum state tomography is inadmissible". In: *arXiv preprint arXiv:1808.01072* (2018).

[30] Andreas Lindholm et al. *Machine learning: a first course for engineers and scientists*. Cambridge University Press, 2022.

[31] Shahnawaz Ahmed et al. "Quantum state tomography with conditional generative adversarial networks". In: *Physical review letters* 127.14 (2021), p. 140502.

[32] Shahnawaz Ahmed, Fernando Quijandría, and Anton Frisk Kockum. "Gradient-descent quantum process tomography by learning Kraus operators". In: *Physical Review Letters* 130.15 (2023), p. 150402.

[33] Giuliano Benenti, Giulio Casati, and Giuliano Strini. *Principles of quantum computation and information-volume I: Basic concepts*. World scientific, 2004.

[34] Smite Meister. *File:bloch sphere.svg*. `https://commons.wikimedia.org/wiki/File:Bloch_sphere.svg` [Accessed: May. 03, 2024]. Jan. 2009.

[35] Paul Roman. *Advanced quantum theory: an outline of the fundamental ideas*. Addison-Wesley publishing company, 1965.

[36] Kurt Jacobs and Daniel A Steck. "A straightforward introduction to continuous quantum measurement". In: *Contemporary Physics* 47.5 (2006), pp. 279–303.

[37] Kurt Jacobs. *Quantum measurement theory and its applications*. Cambridge University Press, 2014.

[38] Howard M Wiseman and Gerard J Milburn. *Quantum measurement and control*. Cambridge university press, 2009.

[39] Heinz-Peter Breuer and Francesco Petruccione. *The theory of open quantum systems*. OUP Oxford, 2002.

[40] Anton Frisk Kockum et al. "Lecture notes on quantum computing". In: *arXiv preprint arXiv:2311.08445* (2023).

[41] Christopher C Gerry and Peter L Knight. *Introductory quantum optics*. Cambridge university press, 2023.

[42] Roy J Glauber. "Coherent and incoherent states of the radiation field". In: *Physical Review* 131.6 (1963), p. 2766.

[43] Nissim Ofek et al. "Extending the lifetime of a quantum bit with error correction in superconducting circuits". In: *Nature* 536.7617 (2016), pp. 441–445.

[44] Gerhard Kirchmair et al. "Observation of quantum state collapse and revival due to the single-photon Kerr effect". In: *Nature* 495.7440 (2013), pp. 205–209.

[45] G Mauro D'Ariano, Matteo GA Paris, and Massimiliano F Sacchi. "Quantum tomography". In: *Advances in imaging and electron physics* 128 (2003), pp. 206–309.

[46] Shahnawaz Ahmed et al. "Classification and reconstruction of optical quantum states with deep neural networks". In: *Physical Review Research* 3.3 (2021), p. 033278.

[47] Shahnawaz Ahmed. *Machine learning for quantum information and computing*. PhD thesis (Department of Microtechnology and Nanoscience, Applied Quantum Physics Laboratory, Chalmers University of Technology,) 2023.

[48] Zdenek Hradil. "Quantum-state estimation". In: *Physical Review A* 55.3 (1997), R1561.

[49] Sergios Theodoridis. *Machine learning: a Bayesian and optimization perspective*. Academic press, 2015.

[50] In Jae Myung. "Tutorial on maximum likelihood estimation". In: *Journal of mathematical Psychology* 47.1 (2003), pp. 90–100.

[51] Jugal Kalita. *Machine learning: Theory and practice*. Chapman and Hall/CRC, 2022.

[52] Cburnett. *File:Colored neural network es.svg*. `https://commons.wikimedia.org/wiki/File:Colored_neural_network_es.svg` [Accessed: Jun. 20, 2024]. May 2019.

[53] Ian Goodfellow et al. "Generative adversarial nets Advances in neural information processing systems". In: *arXiv preprint arXiv:1406.2661* (2014).

[54] Dimitri P Bertsekas and John N Tsitsiklis. "Gradient convergence in gradient methods with errors". In: *SIAM Journal on Optimization* 10.3 (2000), pp. 627–642.

[55] Bobby Kleinberg, Yuanzhi Li, and Yang Yuan. "An alternative view: When does SGD escape local minima?" In: *International conference on machine learning*. PMLR. 2018, pp. 2698–2707.

[56] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *arXiv preprint arXiv:1609.04747* (2016).

[57] Nicolas Boumal. *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[58] Alston S Householder. *The theory of matrices in numerical analysis*. Courier Corporation, 2013.

[59] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 58.1 (1996), pp. 267–288.

[60] Fadil Santosa and William W Symes. "Linear inversion of band-limited reflection seismograms". In: *SIAM journal on scientific and statistical computing* 7.4 (1986), pp. 1307–1330.

[61] J Robert Johansson, Paul D Nation, and Franco Nori. "QuTiP: An open-source Python framework for the dynamics of open quantum systems". In: *Computer Physics Communications* 183.8 (2012), pp. 1760–1772.

[62] Wojciech Bruzda et al. "Random quantum operations". In: *Physics Letters A* 373.3 (2009), pp. 320–324.

[63] Francesco Mezzadri. "How to generate random matrices from the classical compact groups". In: *arXiv preprint math-ph/0609050* (2006).

[64] David Gross et al. "Quantum state tomography via compressed sensing". In: *Physical review letters* 105.15 (2010), p. 150401.

[65] Eliot Bolduc et al. "Projected gradient descent algorithms for quantum state tomography". In: *npj Quantum Information* 3.1 (2017), p. 44.

A

Cholesky decomposition

In this appendix, we will explain how to go from the matrix expression of ρ to the matrix $T_{\mathcal{C}}$. Usually, in the literature, it is easy to find the procedure to do it in the case of the classical Cholesky decomposition $TT^\dagger$, but in the alternative representation $T^\dagger T$, which is also valid, it is not very easy to find it. For that reason I will show the procedure to do it in the representation used in the GD algorithm.

We start with ρ represented by a matrix upper triangular and one lower triangular, in which the upper one is the conjugate transpose of the lower one, and the lower one has real values in the diagonal and complex values in the other entries. We will represent those with general values in the entries and with matrices 3x3.

$$\rho = T^\dagger T = \begin{bmatrix} T_{11} & T_{21}^* & T_{31}^* \\ 0 & T_{22} & T_{32}^* \\ 0 & 0 & T_{33} \end{bmatrix} \begin{bmatrix} T_{11} & 0 & 0 \\ T_{21} & T_{22} & 0 \\ T_{31} & T_{32} & T_{33} \end{bmatrix} \tag{A.1}$$

$$= \begin{bmatrix} T_{11}^2 + |T_{21}|^2 + |T_{31}|^2 & T_{21}^* T_{22} + T_{31}^* T_{32} & T_{31}^* T_{33} \\ T_{22} T_{21} + T_{32}^* T_{31} & T_{22}^2 + |T_{32}|^2 & T_{32}^* T_{33} \\ T_{33} T_{31} & T_{33} T_{32} & T_{33}^2 \end{bmatrix} \tag{A.2}$$

Now we represent the values of the density matrix with general values in the entries.

$$\begin{bmatrix} \rho_{11} & \rho_{21}^* & \rho_{31}^* \\ \rho_{21} & \rho_{22} & \rho_{32}^* \\ \rho_{31} & \rho_{32} & \rho_{33} \end{bmatrix} = \begin{bmatrix} T_{11}^2 + |T_{21}|^2 + |T_{31}|^2 & T_{21}^* T_{22} + T_{31}^* T_{32} & T_{31}^* T_{33} \\ T_{22} T_{21} + T_{32}^* T_{31} & T_{22}^2 + |T_{32}|^2 & T_{32}^* T_{33} \\ T_{33} T_{31} & T_{33} T_{32} & T_{33}^2 \end{bmatrix} \tag{A.3}$$

Comparing we can obtain the value of $T_{33} = \sqrt{\rho_{33}}$, and with that one we can start expressing the other ones.

$$T = \begin{bmatrix} T_{11} & 0 & 0 \\ T_{21} & T_{22} & 0 \\ T_{31} & T_{32} & T_{33} \end{bmatrix} = \begin{bmatrix} \sqrt{\rho_{11} - |T_{21}|^2 - |T_{31}|^2} & 0 & 0 \\ (\rho_{21} - T_{32}^* T_{31})/T_{22} & \sqrt{\rho_{22} - |T_{32}|^2} & 0 \\ \rho_{31}/T_{33} & \rho_{32}/T_{33} & \sqrt{\rho_{33}} \end{bmatrix} \tag{A.4}$$

The values in the entries can be expressed in the following equations

$$T_{i,i} = \sqrt{\rho_{ii} - \sum_{k=j+1}^{N} T_{k,i}^{*} T_{k,i}} \tag{A.5}$$

$$T_{i,j} = \frac{1}{T_{i,i}} \left(\rho_{i,j} - \sum_{k=i+1}^{N} T_{k,i}^{*} T_{k,j} \right) \quad \text{for i>j} \tag{A.6}$$

where N is the number of columns or rows, and it is crucial to start the algorithm with the last component, that is the one that we have the full information at the beginning.

B

Hyperparameters

The GD models have four hyperparameters which were necessary to adjust to obtain values that made the three GD models more optimal. This appendix shows the data that were taken in order to define the best values to use for the correct reconstruction of the density matrices. The process of taking the data involved performing 10 runs of each method and each hyperparameter value, and then comparing which value was the most optimal. This procedure was done for the DV case and CV case.

B.1 Hyper-parameters DV

As there are three GD methods and four hyperparameters (λ, Decay rate, Batches size, η), we will go through each hyperparameter using GD-Cholesky first, followed by GD-manifold, and finally GD-projective. For that, we take the case of 4 qubits.

B.1.1 λ hyperparameter

The selected values of λ are: 10^{-7} for GD-Cholesky from Fig. B.1, 0 for GD-manifold from Fig. B.2, and 0 for GD-projective from Fig. B.3.

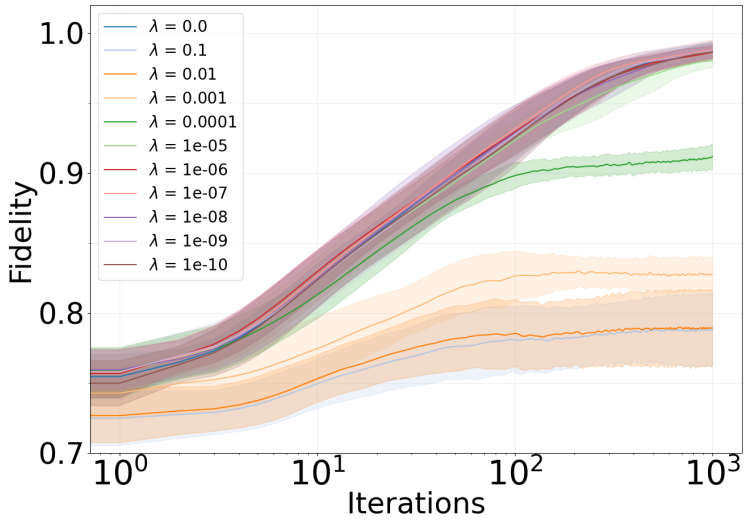


Figure B.1: Random ρ for 4 qubits, starting with 16 states. Lambda hyperparameter for GD-Cholesky. Taking $\lambda = 10^{-7}$.

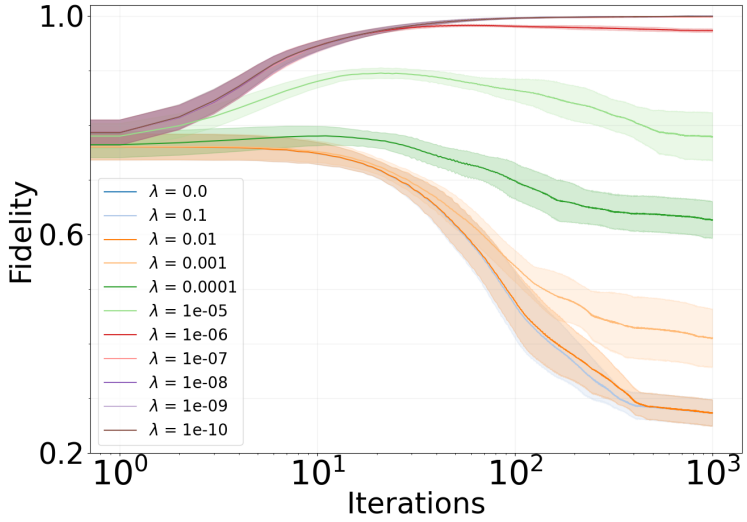


Figure B.2: Random ρ for 4 qubits, starting with 16 states. Lambda hyperparameter for GD-manifold. Taking $\lambda = 0$.

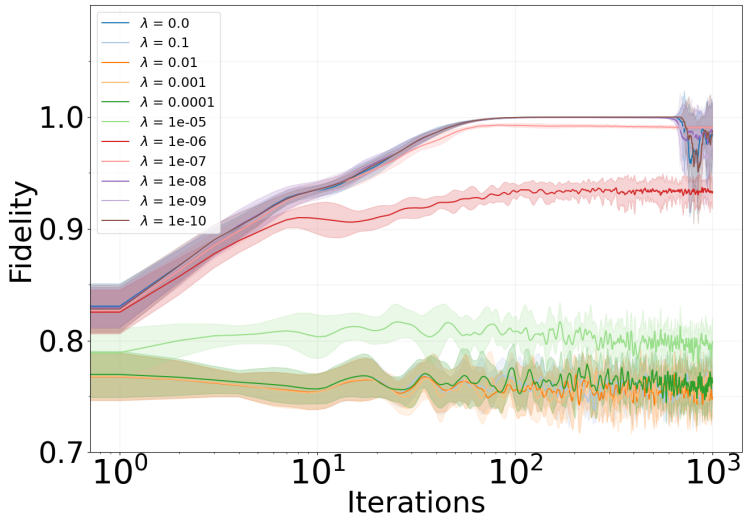


Figure B.3: Random ρ for 4 qubits, starting with 16 states. Lambda hyperparameter for GD-projective. Taking $\lambda = 0$

B.1.2 Decay

The selected value for the decay is 0.999 for the three methods GD-Cholesky Fig. B.4, GD-manifold Fig. B.5, and GD-projective Fig. B.6.

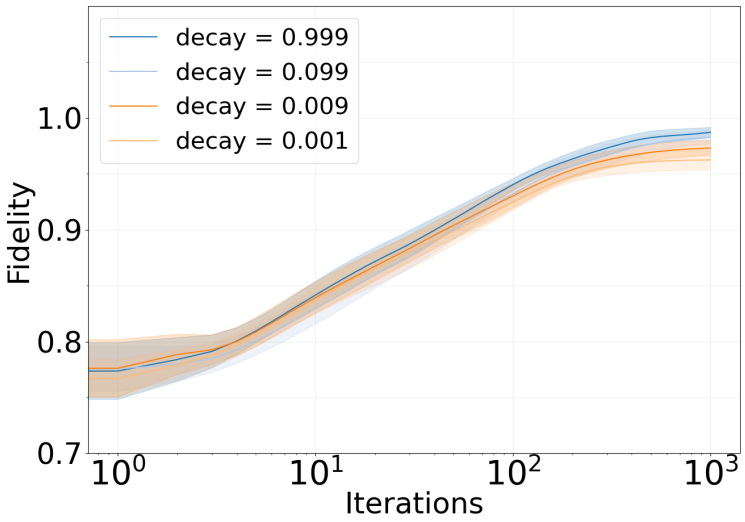


Figure B.4: Random ρ for 4 qubits, starting with 16 states. Decay hyperparameter for GD-Cholesky. Taking decay = 0.999.

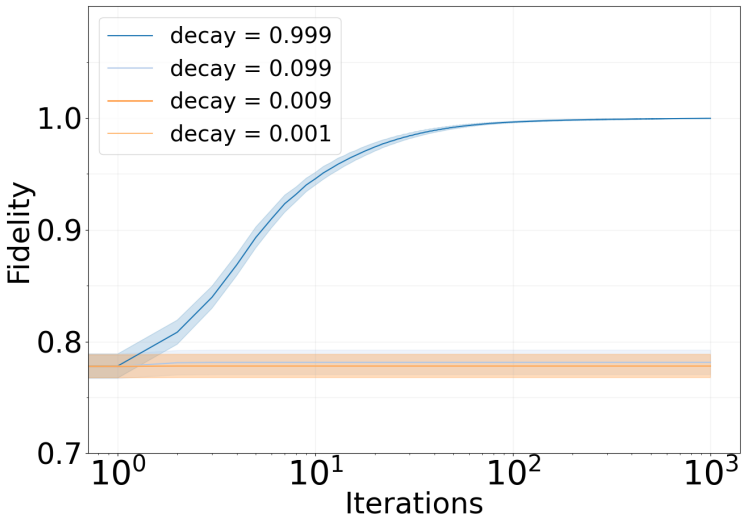


Figure B.5: Random ρ for 4 qubits, starting with 16 states. Decay hyperparameter for GD-manifold. Taking decay = 0.999.

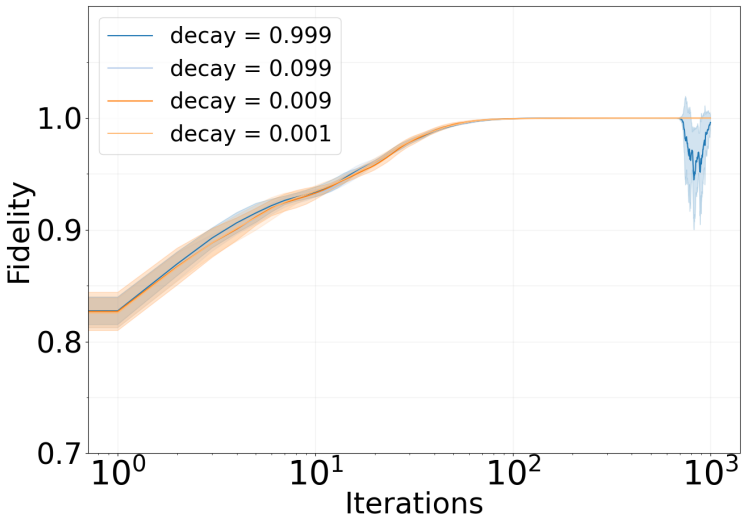


Figure B.6: Random ρ for 4 qubits, starting with 16 states. Decay hyperparameter for GD-projective. Taking decay = 0.099.

B.1.3 Batches

The selected value for the mini-batch size is 55% of the total set of measurement operators for the three methods GD-Cholesky Fig. B.7, GD-manifold Fig. B.8, and GD-projective Fig. B.9.

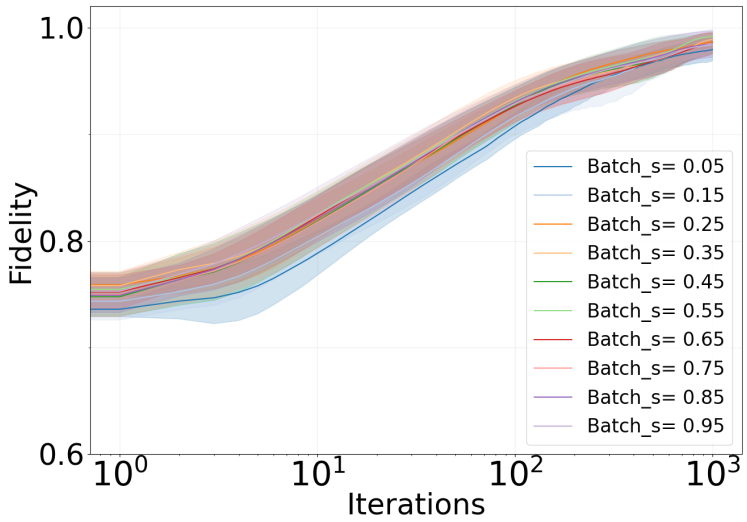


Figure B.7: Random ρ for 4 qubits, starting with 16 states. Batches hyperparameter for GD-Cholesky. Taking Batches 55%.

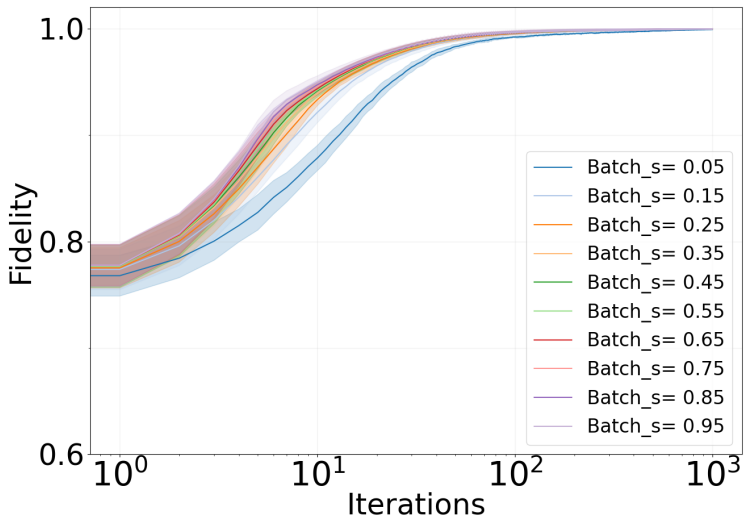


Figure B.8: Random ρ for 4 qubits, starting with 16 states. Batches hyperparameter for GD-manifold. Taking Batches 55%.

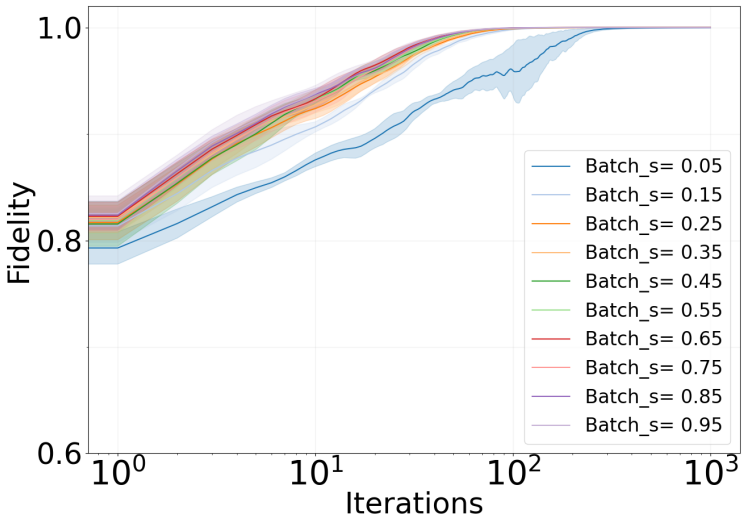


Figure B.9: Random ρ for 4 qubits, starting with 16 states. Batches hyperparameter for GD-projective. Taking Batches 55%.

B.1.4 Initial learning rate (η)

The selected values of learning rate are: 0.05 for GD-Cholesky Fig. B.10, 0.01 for GD-manifold Fig. B.11, and 0.1 for GD-projective Fig. B.12.

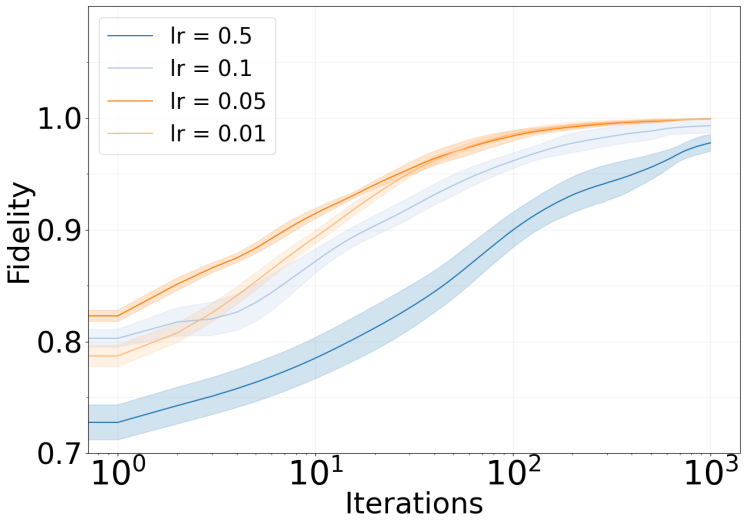


Figure B.10: Random ρ for 4 qubits, starting with 16 states. Initial learning rate hyperparameter for GD-Cholesky. Taking $lr = 0.05$.

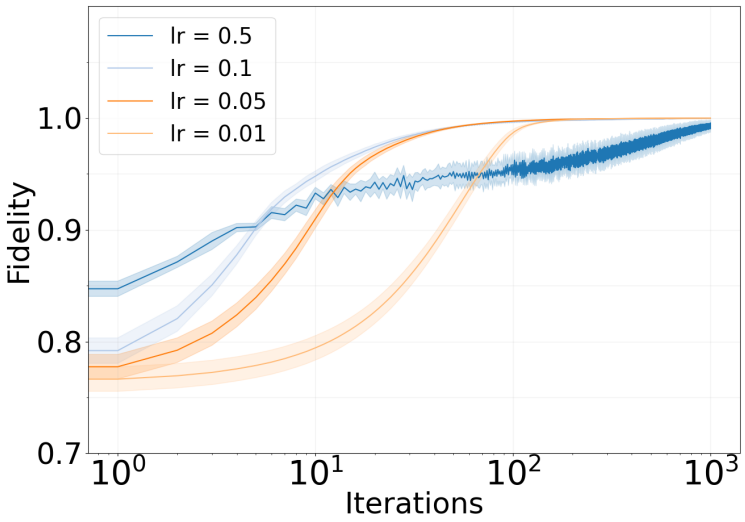


Figure B.11: Random ρ for 4 qubits, starting with 16 states. Initial learning rate hyperparameter for GD-manifold. Taking $lr = 0.01$.

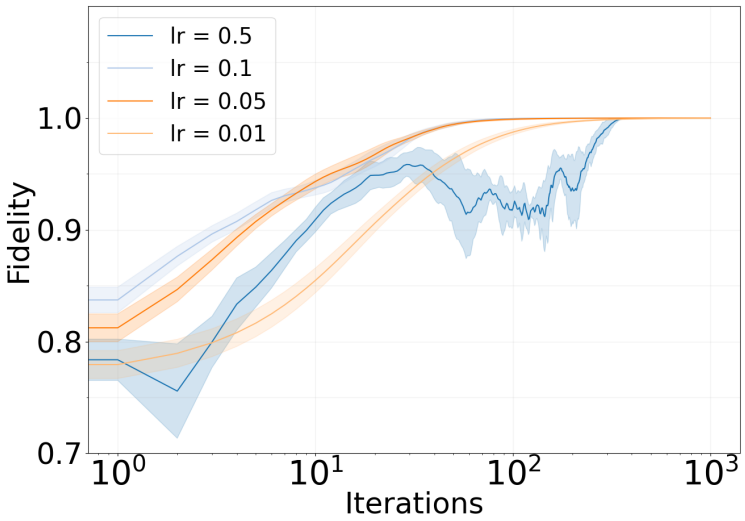


Figure B.12: Random ρ for 4 qubits, starting with 16 states. Initial learning rate hyperparameter for GD-projective. Taking $lr = 0.1$.

B.2 Hyper-parameters CV

For CV we repeat the same procedure that was done with DV, with the difference that here we adjust the hyperparameter values with a cat state $\alpha = 2$.

B.2.1 λ hyperparameter

The selected values of λ are: 10^{-9} for GD-Cholesky Fig. B.13, 0.001 for GD-manifold Fig. B.14, and 10^{-4} for GD-projective Fig. B.15.

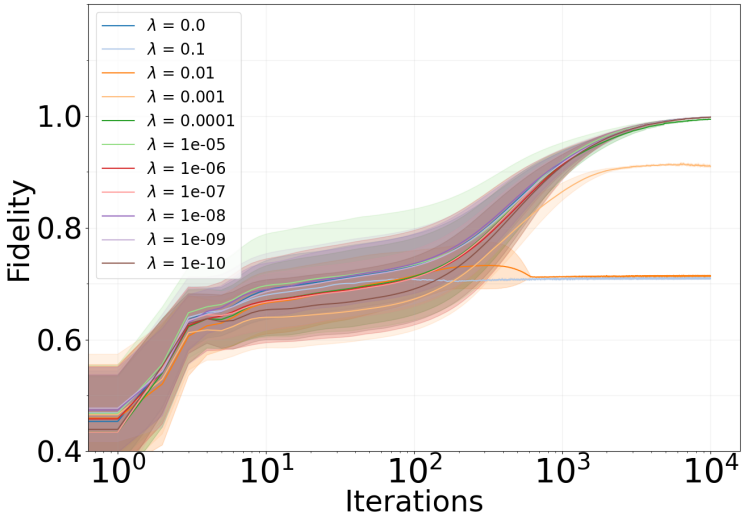


Figure B.13: Cat state $\alpha = 2$, starting with 1 state. Lambda hyperparameter for GD-Cholesky. Taking $\lambda = 10^{-9}$.

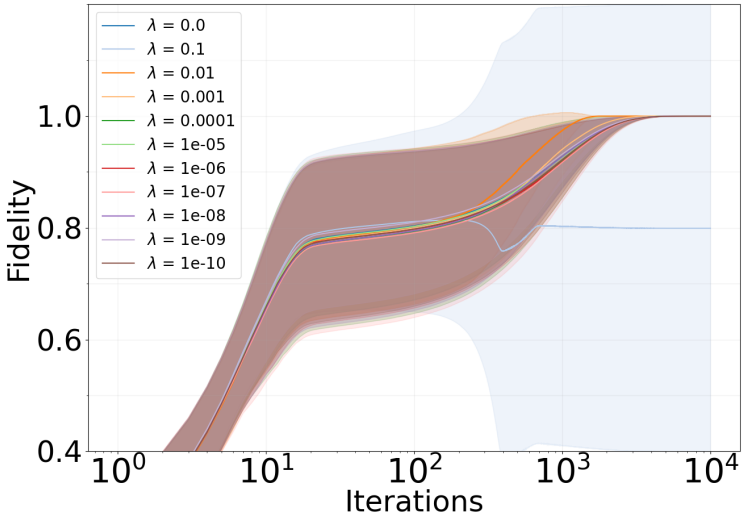


Figure B.14: Cat state $\alpha = 2$, starting with 1 state. Lambda hyperparameter for GD-manifold. Taking $\lambda = 0.001$.

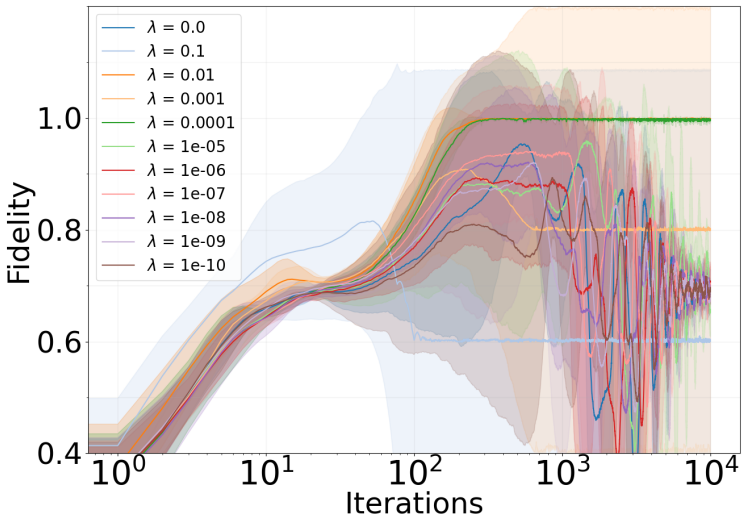


Figure B.15: Cat state $\alpha = 2$, starting with 1 state. Lambda hyperparameter for GD-projective. Taking $\lambda = 10^{-4}$.

B.2.2 Decay

The selected values of decay are: 0.999 for GD-Cholesky Fig. B.16, 0.999 for GD-manifold Fig. B.17, and 0.009 for GD-projective Fig. B.18.

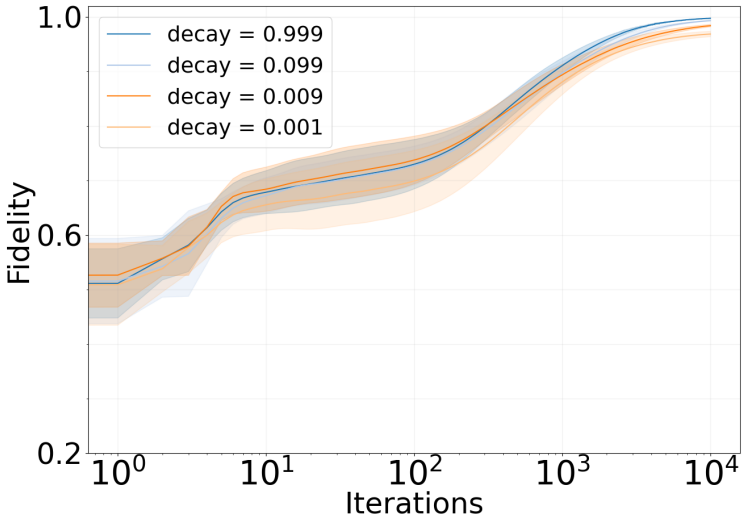


Figure B.16: Cat state $\alpha = 2$, starting with 1 state. Decay hyperparameter for GD-Cholesky. Taking decay = 0.999.

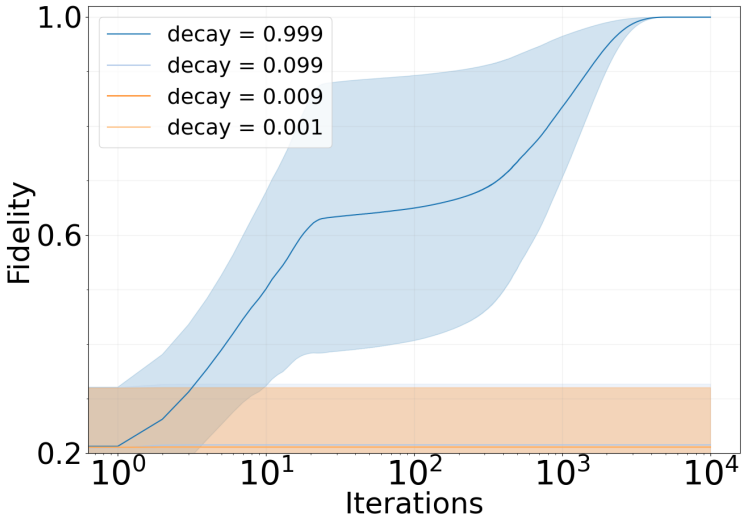


Figure B.17: Cat state $\alpha = 2$, starting with 1 state. Decay hyperparameter for GD-manifold. Taking decay = 0.999.

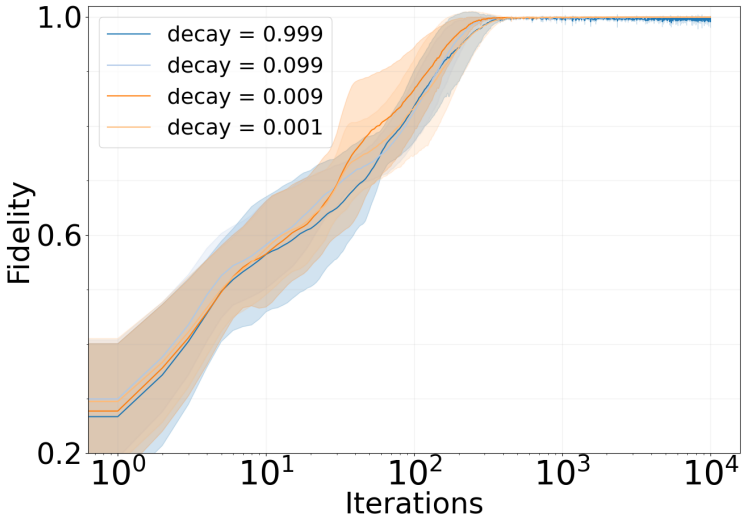


Figure B.18: Cat state $\alpha = 2$, starting with 1 state. Decay hyperparameter for GD-projective. Taking decay = 0.009.

B.2.3 Batches

The selected value for the mini-batch size is 55% of the total set of measurement operators for the three methods GD-Cholesky Fig. B.19, GD-manifold Fig. B.20, and GD-projective Fig. B.21.

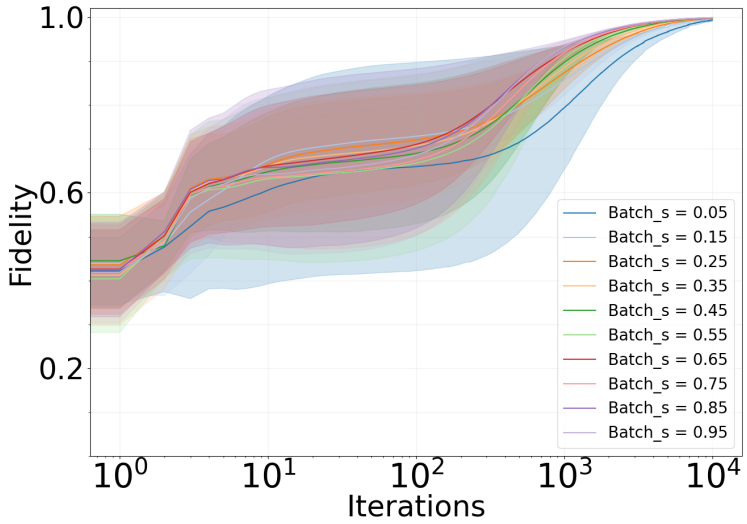


Figure B.19: Cat state $\alpha = 2$, starting with 1 state. Batches hyperparameter for GD-Cholesky. Taking batches = 55%.

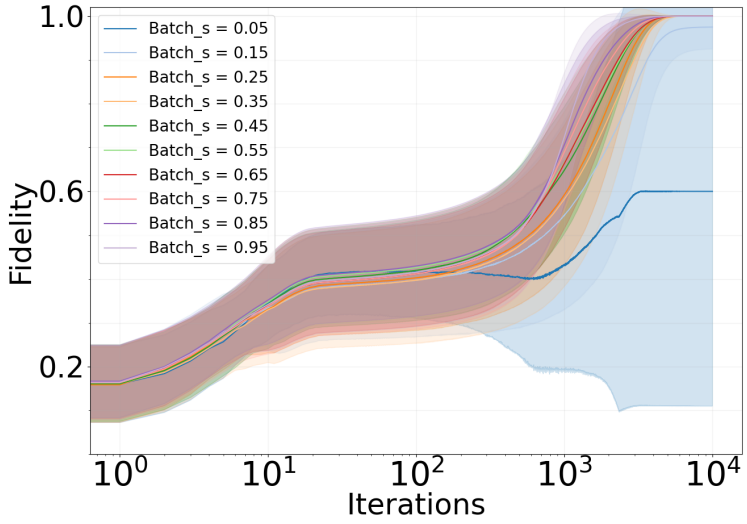


Figure B.20: Cat state $\alpha = 2$, starting with 1 state. Batches hyperparameter for GD-manifold. Taking batches = 55%.

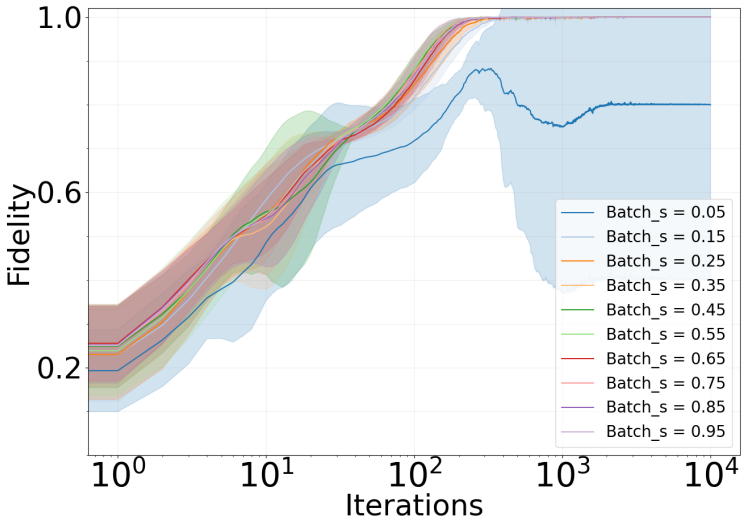


Figure B.21: Cat state $\alpha = 2$, starting with 1 state. Batches hyperparameter for GD-projective. Taking batches = 55%.

B.2.4 Learning rate

Selected values of learning rate: 0.1 for GD-Cholesky Fig. B.22, 0.05 for GD-manifold Fig. B.23, and 0.05 for GD-projective Fig. B.24.

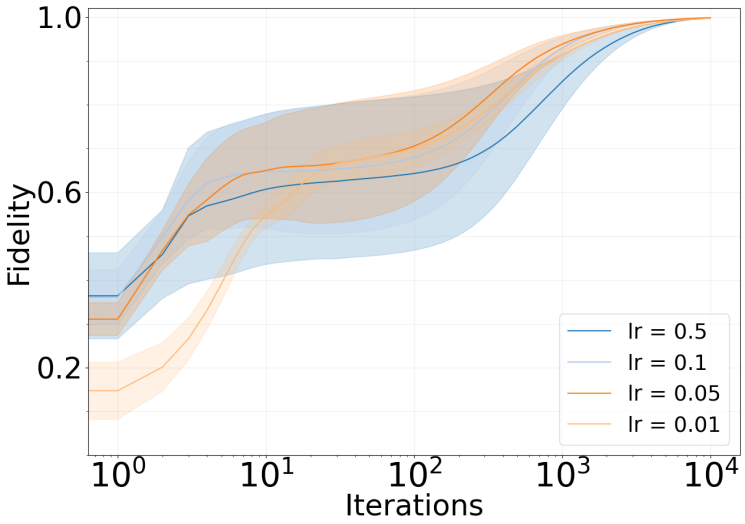


Figure B.22: Cat state $\alpha = 2$, starting with 1 state. Learning rate hyperparameter for GD-Cholesky. Taking $lr = 0.1$.

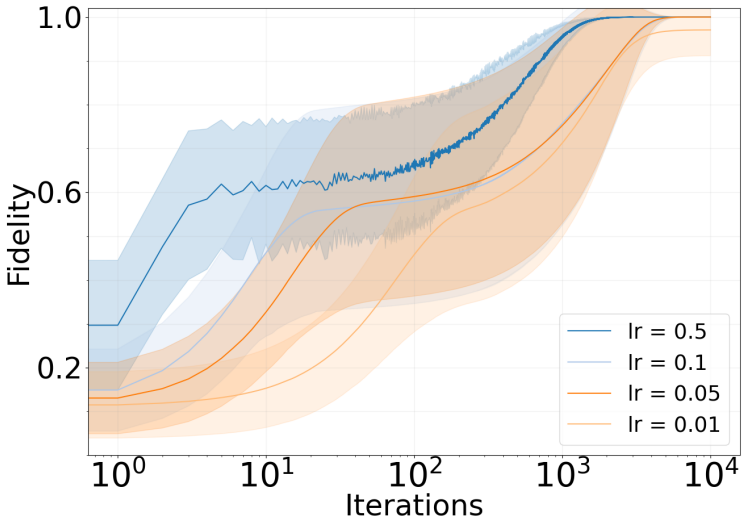


Figure B.23: Cat state $\alpha = 2$, starting with 1 state. Learning rate hyperparameter for GD-manifold. Taking $lr = 0.05$.

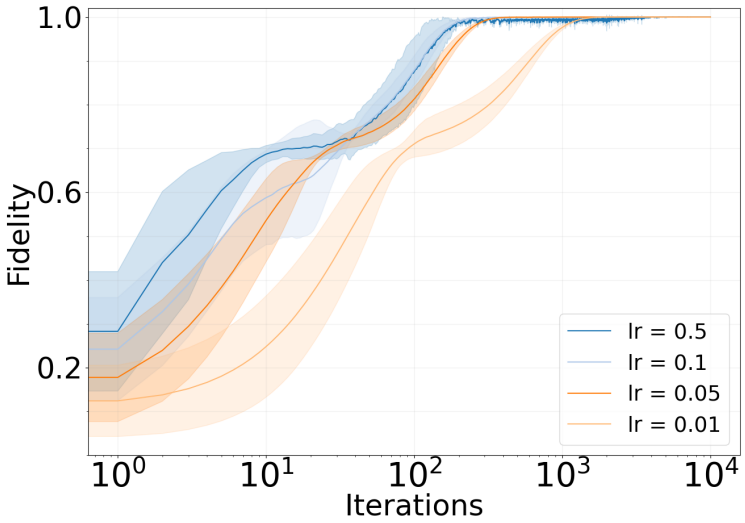


Figure B.24: Cat state $\alpha = 2$, starting with 1 state. Learning rate hyperparameter for GD-projective. Taking $lr = 0.05$.

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY