CENTRO BRASILEIRO DE PESQUISAS FÍSICAS

ASSIGNMENTS FROM COARSE-GRAINED SYSTEMS USING MACHINE LEARNING

Author:

Nina O'NEILL

Rio de Janeiro

July 2, 2021

CENTRO BRASILEIRO DE PESQUISAS FÍSICAS

Masters Dissertation

ASSIGNMENTS FROM COARSE-GRAINED SYSTEMS USING MACHINE LEARNING

Author:
Nina O'NEILL

Supervisor:
Dr. Fernando DE MELO

Quantum Information Group - CBPF
Theoretical Physics department

Rio de Janeiro

July 2, 2021

In a minute or two the Caterpillar took the hookah out of its mouth and yawned once or twice, and shook itself. Then it got down off the mushroom, and crawled away in the grass, merely remarking as it went, "One side will make you grow taller, and the other side will make you grow shorter." "One side of what? The other side of what?" thought Alice to herself. "Of the mushroom," said the Caterpillar, just as if she had asked it aloud; and in another moment it was out of sight.

Lewis Carroll, *Alice in Wonderland*

Aknowledgements

First, I thank the heroine or hero who invented black tea. Throughout my academic career, I was told I'd eventually acquire a taste for coffee. Yet, that never happened, and this dissertation might have not existed if not for tea.

I also thank my supervisor, Dr. Fernando de Melo, who may not be a teacher, but has the best qualities of one. I could not forget Dr. Raúl Oscar Vallejos and Dr. Alexandre Baron Tacla. Their input and critique was of fundamental importance to this work. I am grateful to CAPES for funding this project. I also thank my high school physics teacher Sérgio Lins Gouveia, who inspired me to be a physicist in the first place.

I am grateful to my parents, for supporting me in the waiting game that is the academic career. For always encouraging me, I thank my friends Alice and Bia, my boyfriend Daniel and my brother Lucas.

Finally, I'd like to thank my cat, for always cheering me up, but alas, he cannot read.

Resumo

O objetivo deste projeto foi utilizar técnicas computacionais para aproximar o melhor estado microscópico a ser designado para um sistema dada uma descrição macroscópica deste, a partir de resultados de pesquisa recentes sobre o assunto de mapeamento quântico da escala macro para a micro. Três métodos são apresentados neste trabalho.

Primeiro, um método de amostragem por rejeição, em que os microestados quânticos são amostrados uniformemente, um mapa de *coarse-graining* é aplicado neles e, então, os estados são submetidos a condições de valor limiar para construir um conjunto que aproxima o ensemble de todos os microestados que satisfazem as condições macroscópicas.

Em segundo lugar, um método de amostragem e partição, onde a imagem do mapa de *coarse-graining* é dividido em células. Essa divisão, feita com o algoritmo de aprendizado de máquina não supervisionado *k-means++*, é utilizada então para classificar os microestados com base na posição dos estados efetivos correspondentes no espaço determinado pela imagem do mapa. A abordagem cria células menores em lugares onde a concentração de dados é alta, consequentemente permitindo que os assignments associados a macroestados mais prováveis sejam calculados com precisão maior.

Finalmente, o leitor ou leitora é apresentado(a) a perspectivas para um método inspirado em *variational autoencoders* para aprender como amostrar diretamente de microestados que satisfaçam as condições macroscópicas.

Palavras-chave: *Assignments*, *coarse-graining*, física estatística, *machine learning*, *k-means*, *variational autoencoders*.

Abstract

The goal of this project was to build upon existing research on the subject of macro-to-micro quantum mapping by using computational tools to approximate the best microscopic assignment for a system given a macroscopic description of it. Three methods are presented in this work.

First, a rejection sampling method, in which quantum microstates are sampled uniformly, acted upon with a coarse-graining map and submitted to threshold conditions in order to construct a set which approximates the ensemble of all microstates that satisfy the macroscopic conditions.

Second, a partition sampling method, in which the image space of the coarse-graining map is divided into cells. This division, performed with the unsupervised learning algorithm k-means++, is then used to label the microstates based on the position of their coarse-grained counterparts in the image space. The approach creates smaller cells in places where the image space is densely populated, thereby allowing the assignments associated with more probable macrostates to be calculated more precisely.

Lastly, the reader is presented with perspectives for a variational autoencoder inspired method for learning how to sample microstates that satisfy the macroscopic constraints directly.

Keywords: Assignments, coarse-graining, statistical physics, machine learning, k-means, variational autoencoders.

Contents

Chapter 1

Introduction

In his book Reliable Knowledge, John Ziman defines the goal of science as "a consensus of rational opinion over the widest possible field" [1]. Throughout History, scientists have worked together to tackle this monumental task. For many physicists, the ideal way to achieve this goal is by finding a theory that is universal in the sense that its tenets can be applied to describe natural phenomena from the scale of atoms and subatomic particles to that of celestial bodies and the universe itself.

One of the main difficulties of this pursuit is the need to conciliate the laws of physics at extraordinarily disparate scales. The properties of electrons in potential wells seem completely unrelated to those of, for example, my cat curled up in his favorite box. Although it is technically true that my cat's properties could be described by the Schrödinger equation of his nuclei and electrons and their Coulomb interactions, the amount of variables in this equation make it practically unsolvable. If the goal of science is to achieve consensus between experts of a field, then the theory which describes the interconnections between phenomena at different levels of description must be amenable to be publicly assessed by the scientific community.

Statistical mechanics was one of the first fields of physics to bridge the macroscopic and microscopic descriptions of the world. When it first originated, it sought out to derive the thermal properties of matter from the laws that govern the behavior of its constituent particles. This was achieved through the use of thermodynamic ensembles. Three important ensembles of this sort, introduced by Gibbs in 1902 [2] are the microcanonical, canonical and grand canonical ensembles. The microcanonical ensemble describes a system in which the total energy and number of particles do not change, in other words, a mechanically and adiabatically isolated system. In the canonical ensemble, the number of particles is still unchanging, but it is the temperature, rather than the energy, that is fixed. It models a system in weak thermal contact with a heat bath, for example. Finally, the grand canonical ensemble refers to a system with a specified temperature and chemical potential, but whose energy and number of particles may vary [3].

In all of these scenarios, there are either no exchanges between the system and the environment, or the distinction between the two is clear enough that it is possible to describe their states before the interaction occurs. In such situations, the degrees of freedom of the environment, which are physically inaccessible, are discarded via the partial trace [4]. However, there are many physical situations in which the split between our system of interest and the surrounding environment is not so clear-cut. For example, recently, open quantum systems with non-Markovian evolutions have been gaining the interest of a diverse set of research communities [5, 6, 7, 8, 9, 10].

One proposed method to deal with setups of this sort involves the use of coarse-graining maps. A coarse-graining map is a generalization of the partial trace in the sense that it is a mathematical tool that can be used to discard the experimentally unobtainable degrees of freedom of a physical system. However, unlike the partial trace, which can only be applied to situations with a clear separation between system and environment, coarse-graining maps can be used to model many different kinds of phenomena. For instance, they have been used to describe the performance of a blurred and saturated detector attempting to measure the state of atoms trapped in an optical lattice [11, 12, 13]. In chapter 2, we will review quantum channel formalism which supports coarse-graining maps in order to explain them in further detail.

Using coarse-graining maps, it is possible to define a statistical ensemble which allows us to assign to systems with strongly interacting parts a microscopic state which abides by all the macroscopical constraints [13]. This can be used in order to assign a microscopic state to a system, given only information on its macroscopic state. This is interesting because it allows for a more complete understanding of the interelations of the descriptions of physical phenomena at different scales.

As is the case with thermodynamical ensembles, defining the set of relevant microstates and calculating the ensemble average is not always trivial. Therefore, the goal of this project was to use modern computational techniques, such as unsupervised machine learning, which have recently generated some interesting applications in physics [14, 15], to create algorithms that could approximate a system's average microscopic state given a certain coarse-graining. In order to quantify our algorithms' efficiency, we applied them to simple situations for which the average global state can be calculated analytically. This will allow us to estimate the numerical error involved in applying them to other coarse-graining operators.

Our three main approaches were: brute force rejection sampling, a refinement of the rejection sampling algorithm using unsupervised machine learning techniques and a variational autoencoder inspired algorithm used to sample microstates non-uniformly. The algorithms and preliminary results will be discussed in chapters 4, 5 and 6, respectively.

Chapter 2

Coarse-graining maps

2.1 Quantum channels

The quantum channel framework is the most general way to describe physically realizable changes in the states of quantum systems. A quantum channel is a map which, in the Schrödinger picture, describes the time evolution of a quantum state. Let us denote this map $\Lambda : \mathcal{L}(\mathcal{H}_A) \rightarrow \mathcal{L}(\mathcal{H}_B)$. In this notation, $\mathcal{L}(\mathcal{H}_A)$ refers to the space which contains all of the linear operators acting on a Hilbert space of dimension A. Mathematically speaking, a quantum channel must satisfy the following properties:

- Linearity
- Trace-preservation
- Complete positivity.

A map that satisfies the above properties is usually called a CPTP (Completely Positive Trace-Preserving) map. Each of these properties is necessary in order for the evolution described by this map to be physically meaningful. In the following subsections, we will define each of these requirements and explain their physical interpretation.

2.1.1 Linearity

Let us take the map mentioned before, $\Lambda : \mathcal{L}(\mathcal{H}_A) \rightarrow \mathcal{L}(\mathcal{H}_B)$. In order for this map to be linear, it must satisfy the following properties [16]:

- $\Lambda(\rho_1 + \rho_2) = \Lambda(\rho_1) + \Lambda(\rho_2)$, for any matrices $\rho_1, \rho_2 \in \mathcal{L}(\mathcal{H}_A)$
- $\Lambda(\alpha\rho) = \alpha\Lambda(\rho)$, for any scalar α.

Maps that describe time evolutions of quantum states must be linear, since a non-linear time evolution would result in a violation of the no-signaling condition. An in-depth discussion on this can be found in [17].

2.1.2 Trace preservation

A trace-preserving map is one that, when applied to an operator, will not change that operator's trace. It is easy to see why this is a requirement for a map Λ to describe a valid physical change in a system. In order for a density operator ρ to represent a valid quantum state, it must satisfy $\mathrm{Tr}[\rho] = 1$. This is known as the normalization condition, and it is necessary in order for the density matrix to have a coherent probabilistic interpretation. The state resulting from the application of this map, $\Lambda(\rho)$, must also have this property, i.e.,

$$\mathrm{Tr}[\Lambda(\rho)] = \mathrm{Tr}[\rho]. \tag{2.1.1}$$

Hence, Λ must preserve the trace.

2.1.3 Complete positivity

Consider an arbitrary density matrix ρ in a vector space $\mathcal{L}(\mathcal{H}_A)$. Density matrices must be positive in order to be physically meaningful. Therefore, if the map $\Lambda : \mathcal{L}(\mathcal{H}_A) \to \mathcal{L}(\mathcal{H}_B)$ is to represent a valid time evolution, then Λ must necessarily satisfy:

$$\Lambda(\rho) \geq 0. \tag{2.1.2}$$

Maps that satisfy this condition for any operator are called positive maps. In the context of quantum mechanics, however, this condition is not all that is necessary for $\Lambda(\rho)$ to represent a valid quantum state in all possible cases. Consider, for example, the case in which the system ρ is actually a bipartite system composed of two subsystems that have become entangled. Suppose that some physical change described by the map Λ is applied to one of the subsystems, while the other one remains unchanged. The time evolution of the entire system is described by the map $\tilde{\Lambda} = \mathbb{1}_k \otimes \Lambda$, where $\mathbb{1}_k$ is the identity matrix associated to the dimention k of the unchanged subsystem. In order for $\tilde{\Lambda}$ to yield a valid quantum state, the following must be true [17]:

$$\tilde{\Lambda}(\rho) = \mathbb{1}_k \otimes \Lambda(\rho) \geq 0. \tag{2.1.3}$$

In order to cover all possible cases of this sort, this equation must be true for all ρ, and for any value of k up to the dimension of the space containing ρ. This condition is called complete positivity, and is a stronger requirement than positivity.

Quantum channels are not only able to describe the dynamics of open systems, but can also be applied to situations in which we are only looking at a single part of a complex system, or account for the effects of imperfect tools on the information that we can obtain from the system through measurement. In the next section, we will see how this is done in more detail.

2.2 Coarse-graining channels

The idea behind coarse-grained modeling it is to simulate the behavior of complex systems using a simplified, effective representation.

This work uses the definition of coarse-graining maps provided in [11]. Given a quantum channel $\Lambda : \mathcal{L}(\mathcal{H}_D) \to \mathcal{L}(\mathcal{H}_d)$, where $\mathcal{H}_D$ and $\mathcal{H}_d$ are Hilbert spaces of dimentions D and d respectively, we say that Λ is a coarse-graining map if and only if $D > d$. In other words, a coarse-graining map is a quantum channel that reduces the dimensionality of the description of a quantum state. This mathematical property makes it the ideal tool for creating effective descriptions of systems by eliminating degrees of freedom that are experimentally unobtainable or irrelevant to the phenomena being studied.

In this project, I used two particular coarse-graining maps corresponding to different physical scenarios. These will be discussed in the following subsections.

2.2.1 The partial trace

The partial trace is a coarse-graining map that has a crucial role in the theory of open quantum systems [18]. The primary goal of open quantum systems theory is to describe the local dynamics of a system coupled to an environment. The environment is usually a large and complicated system, which makes solving this problem by explicitly determining all the degrees of freedom of the environment impratical or even unfeasible. Hence, the evolution of the system is described by an effective dynamics. Given two linear operators L_A and L_B

$$\begin{aligned} \mathrm{Tr}_B : \mathcal{L}(\mathcal{H}_A \otimes \mathcal{H}_B) &\to \mathcal{L}(\mathcal{H}_A) \\ : L_A \otimes L_B &\to \mathrm{Tr}(B) \cdot L_A. \end{aligned} \tag{2.2.1}$$

A familiar physical scenario to which the open quantum systems theory can be applied is that of a bipartite system composed of a subsystem of interest interacting with a thermal bath [4]. The Hilbert spaces associated each of these parts are denoted $\mathcal{H}_A$ and $\mathcal{H}_B$, respectively. The total system is described by a density matrix $\rho_{AB} \in \mathcal{L}(\mathcal{H}_A \otimes \mathcal{H}_B)$, and their joint unitary evolution is

given by $U(t) = e^{-iHt}$, where H is the total Hamiltonian. If the initial joint state is $\rho_{AB}(0)$, then, by Schrödinger's equation:

$$\rho_{AB}(t) = U(t)\rho_{AB}(0)U^{\dagger}(t). \tag{2.2.2}$$

The spectral decomposition of the initial state of the bath, $\rho_B(0)$, can be written as:

$$\rho_B(0) = \sum_i \lambda_i \left|i\right\rangle\!\left\langle i\right|, \tag{2.2.3}$$

where $\{\lambda_i\}$ are the eigenvalues and $\{|i\rangle\}$ the eigenvectors which form an orthonormal basis for $\rho_B(0)$. The state of the system at time t, $\rho_A(t)$, is given by:

$$\begin{aligned} \rho_A(t) &= \mathrm{Tr}_B[\rho_{AB}(t)] = \mathrm{Tr}_B[U(t)\rho_{AB}(0)U^{\dagger}(t)] \\ &= \sum_j \left\langle j\right| U(t)\rho_{AB}(0)U^{\dagger}(t) \left|j\right\rangle. \end{aligned} \tag{2.2.4}$$

In order to effectively remove the degrees of freedom of the bath, we assume the initial state is totally decoupled, i.e.,

$$\rho_{AB}(0) = \rho_A(0) \otimes \rho_B(0). \tag{2.2.5}$$

This is a reasonable assumption to make, considering that, if the systems have only been in contact for a short amount of time, strong correlations will not have formed yet. This considered, the effective dynamics for A are given by:

$$\begin{aligned} \rho_A(t) &= \sum_j \left\langle j\right| \left[U(t)\rho_A(0) \otimes \sum_i \lambda_i \left|i\right\rangle\!\left\langle i\right| U^{\dagger}(t)\right] \left|j\right\rangle \\ &= \sum_{i,j} \sqrt{\lambda_i} \left\langle j\right| U(t) \left|i\right\rangle \rho_A(0) \sqrt{\lambda_i} \left\langle i\right| U^{\dagger}(t) \left|j\right\rangle \\ &= \sum_{i,j} K_{ij}(t)\rho_A(0)K_{ij}^{\dagger}(t), \end{aligned} \tag{2.2.6}$$

where $K_{ij}(t) = \sqrt{\lambda_i} \left\langle j\right| U(t) \left|i\right\rangle$ are called the Kraus operators.

As we can see, the effectiveness of this approach relies on the assumption that the correlations between the subsystem of interest and the thermal bath are, initially, weak enough that the global system can be described as a product state. This description is not appropriate for all physical scenarios. In situations in which the split between the system and envinronment is not so clear-

cut, it would not be possible to describe the global state in this way. An example of this type of circumstance will be given in the following subsection.

2.2.2 The blurry detector coarse-graining map

In [11, 12], a coarse-graining approach was proposed to model isolated quantum systems measured by a detector with an imperfect resolution. One of the situations that can be described by this approach is that of cold atoms trapped in an optical lattice [19, 20, 21]. Consider an experiment involving two such atoms. It is possible to model each atom as a qubit, where the state $|0\rangle$ refers to atoms in the spin-up state, and $|1\rangle$ to atoms in the spin-down state. The global state of the system can be described by a density matrix in $\mathcal{L}(\mathcal{H}_4)$.

The spin state of the atoms can be detected using a fluorescence imaging technique [22]. An experimenter shines a laser on the system so that, for each atom, if it is in state $|1\rangle$, it scatters light, and if it is in state $|0\rangle$, it does not. Suppose the experimental conditions are such that the detector cannot resolve each individual atom. Furthermore, the amount of light coming from a single atom is already enough to saturate the detector. In this case, one would not be able to discriminate between states $|01\rangle$, $|10\rangle$ and $|11\rangle$. In other words, it is not possible to fully measure the microscopic state of the system with this detector.

It is possible to represent the partial information that we obtained via experiment using coarse-grained modeling. The pair of atoms is represented as a single 'effective' atom which is in state $|1\rangle$ if light was scattered, and in state $|0\rangle$ if not.

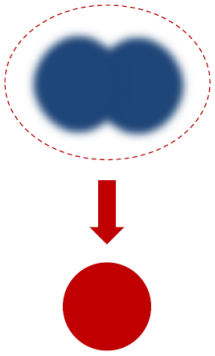


Figure 2.2.1: Pictoral representation of the coarse-grained model. The drawing on the top represents the blurred signal obtained by the detector, while the red sphere represents the approximation of the signal as that of a single effective atom.

One possible coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_4) \to \mathcal{L}(\mathcal{H}_2)$ that describes this situation is: [11, 12]

$$
\begin{aligned}
\Lambda_{CG}(|00\rangle\langle 00|) &= |0\rangle\langle 0| \qquad & \Lambda_{CG}(|01\rangle\langle 00|) &= \frac{|1\rangle\langle 0|}{\sqrt{3}} \\
\Lambda_{CG}(|00\rangle\langle 01|) &= \frac{|0\rangle\langle 1|}{\sqrt{3}} \qquad & \Lambda_{CG}(|01\rangle\langle 01|) &= |1\rangle\langle 1| \\
\Lambda_{CG}(|00\rangle\langle 10|) &= \frac{|0\rangle\langle 1|}{\sqrt{3}} \qquad & \Lambda_{CG}(|01\rangle\langle 10|) &= 0 \\
\Lambda_{CG}(|00\rangle\langle 11|) &= \frac{|0\rangle\langle 1|}{\sqrt{3}} \qquad & \Lambda_{CG}(|01\rangle\langle 11|) &= 0 \\
\Lambda_{CG}(|10\rangle\langle 00|) &= \frac{|1\rangle\langle 0|}{\sqrt{3}} \qquad & \Lambda_{CG}(|10\rangle\langle 01|) &= 0 \\
\Lambda_{CG}(|11\rangle\langle 00|) &= \frac{|1\rangle\langle 0|}{\sqrt{3}} \qquad & \Lambda_{CG}(|11\rangle\langle 01|) &= 0 \\
\Lambda_{CG}(|10\rangle\langle 10|) &= |1\rangle\langle 1| \qquad & \Lambda_{CG}(|11\rangle\langle 10|) &= 0 \\
\Lambda_{CG}(|11\rangle\langle 11|) &= |1\rangle\langle 1| \qquad & \Lambda_{CG}(|10\rangle\langle 11|) &= 0.
\end{aligned}
\tag{2.2.7}
$$

There are a few interesting observations to make about this map and how it reflects the physical situation.

First of all, note that the coherences between distinguishable states are reduced by a $1/\sqrt{3}$ factor. This is necessary so that the blurry detector coarse-graining map satisfies the conditions of complete positivity, linearity and trace preservation that were discussed previously. This signals that the probability of observing coherence in this circumstance is reduced, but does not vanish [12].

Furthermore, recall that, since the detector is saturated, if any of the atoms is in an excited state, what we will observe is an 'effective' atom in an excited state. This is why it is impossible to discriminate between states $|01\rangle$, $|10\rangle$ and $|11\rangle$. There are two immediate consequences to this fact. The first is that there cannot be any coherece between these states, hence why these terms vanish in the coarse-grained description. The second is that states $|10\rangle\langle 10|$, $|11\rangle\langle 11|$ and $|01\rangle\langle 01|$ are all mapped to the same state, $|1\rangle\langle 1|$. A consequence of this is that, if one were to uniformly sample a finite number of microstates $\psi_i \in \mathcal{L}(\mathcal{H}_4)$, where $\psi_i = |\psi_i\rangle\langle\psi_i|$, and apply to them the blurry detector coarse-graining map, the most densely populated region of the image space would be the neighborhood of $|1\rangle\langle 1|$.

In contrast, note that the effective state $\rho = |0\rangle\langle 0|$ can only be observed if both atoms are in the pure state $|0\rangle$. In other words, there is only one possible microstate that leads to the macrostate $|0\rangle\langle 0|$. In the case that was previously described, where one sampled a finite number of microstates $\psi_i \in \mathcal{L}(\mathcal{H}_4)$ and applied to them the blurry detector coarse-graining map, the probability of obtaining $\Lambda(\psi_i) = |0\rangle\langle 0|$ would be null.

It is possible to visualize this property of the map by representing the coarse-grained states

using their Bloch vector coordinates, as I will show in the following subsection.

2.2.3 Visualizing properties of the map using the Bloch sphere

Using the computational basis, it is possible to parameterize pure single cubit states in the following manner:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle\,, \tag{2.2.8}$$

where $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$. The parameters θ and ϕ can be re-interpreted in spherical coordinates as the polar and azimuthal angles of a point on a sphere of radius one. Since mixed states are convex combinations of pure states, all 1 qubit states can be represented as points of a three-dimensional ball with radius one [23]. This ball is called the Bloch sphere. An illustration of the Bloch sphere is shown in figure 2.2.2.

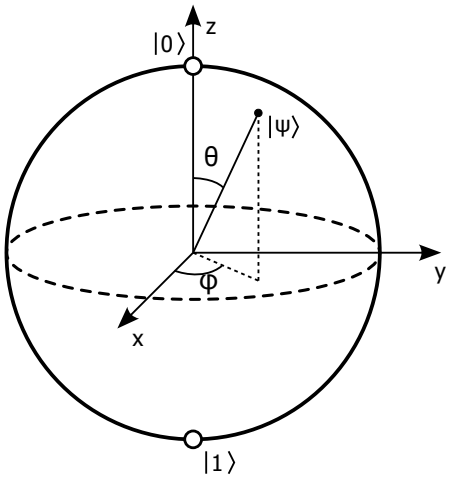


Figure 2.2.2: Illustration of the Bloch sphere. The states $|0\rangle$ and $|1\rangle$ lie, respectively, on the north and south poles of the ball. Image by Smite-Meister, CC BY-SA 3.0 <https://creativecommons.org/licenses/by-sa/3.0>, via Wikimedia Commons.

The vector composed of the cartesian coordinates of a one qubit state represented in this manner is called its Bloch vector, and its components are defined as:

$$\begin{aligned} r_x &= \mathrm{Tr}(\rho\sigma_x) \\ r_y &= \mathrm{Tr}(\rho\sigma_y) \\ r_z &= \mathrm{Tr}(\rho\sigma_z), \end{aligned} \tag{2.2.9}$$

where σ_x, σ_y and σ_z are the Pauli matrices:

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$
$$\sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \tag{2.2.10}$$
$$\sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Using the formulas in 2.2.9, one can verify that the state $|0\rangle\langle 0|$ is represented as the north pole of the Bloch sphere, while $|1\rangle\langle 1|$ is the south pole.

Figure 2.2.3 shows a plot which allows one to visualize the effect of applying the blurry detector coarse-graining map on a large number of uniformly sampled pure states. Using the QuTip toolbox [24], I generated a set of 1000 Haar distributed pure quantum states $\psi_i \in \mathcal{L}(\mathcal{H}_4)$ and applied the coarse-graining map to each one. The graph shows that, because of the properties of the map, random states $|\psi_i\rangle \in \mathcal{H}_4$ are more likely to be mapped onto the southern hemisphere of the Bloch sphere than to the northern hemisphere. Figure 2.2.4 shows a histogram of the expected values of the Pauli matrix σ_z for the states $\Lambda(\psi_i)$ generated, further illustrating this point. This assymetry is a consequence of the physical limitations of the detector which the coarse-graining map describes.

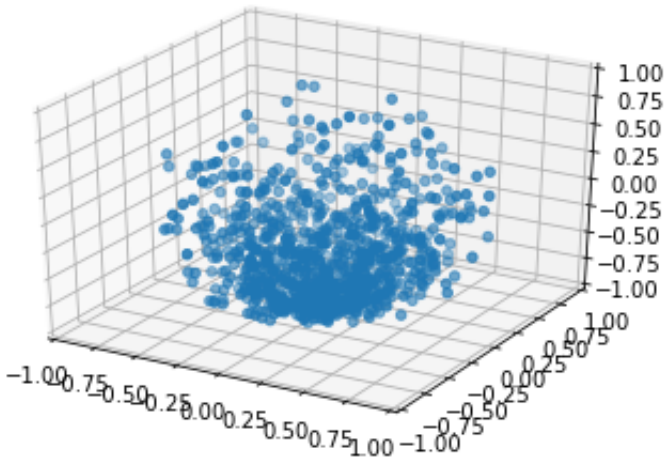


Figure 2.2.3: Plot showing the coordinates of the Bloch vectors of uniformly sampled pure states $\psi_i \in \mathcal{H}_4$ to which the blurry detector coarse-graining map was applied. This plot allows us to visualize the effect the map has on the pure states. It is interesting to observe that there is a tendency for the effective states to concentrate in the southern hemisphere of the Bloch sphere. The incapacity of the detector of distinguishing both atoms leads to many different microstates being mapped onto $|1\rangle\langle 1|$, while the effective state $|0\rangle\langle 0|$ only occurs for $|\psi_i\rangle = |00\rangle$. This originates the assymetry observed in this plot.

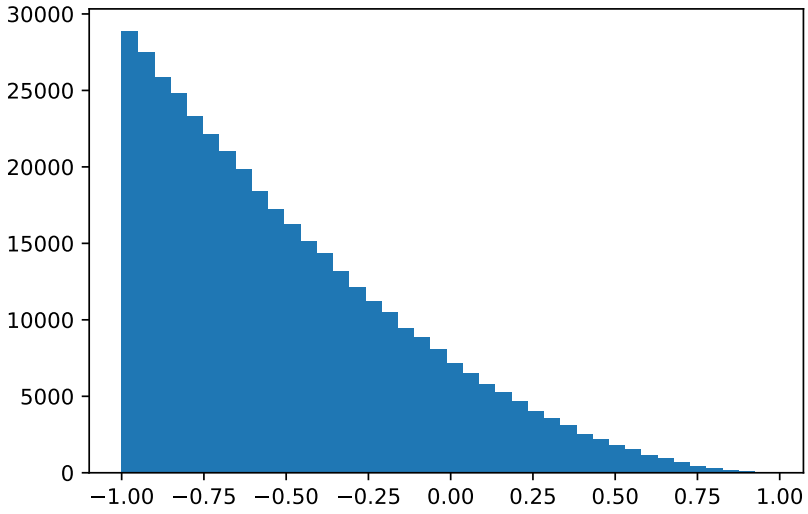


Figure 2.2.4: Histogram of the expectation values of σ_z for a set of 400000 states $\Lambda(\psi_i)$ where ψ_i were uniformly sampled pure states $\in \mathcal{H}_4$. This histogram shows that if one were to sample a finite number of microstates $\psi_i \in \mathcal{H}_4$ and apply to them the blurry detector coarse-graining map, the most densely populated region would be in the southern hemisphere of the Bloch sphere, while the northern hemisphere is sparsely populated in comparison.

Chapter 3

Generalized statistical ensembles

Statistical ensembles were introduced in [2] to describe thermodynamic equilibrium and explain how the laws of thermodynamics could be derived from classical mechanics. One can think of an ensemble as an extremely large — sometimes infinite — amount of copies of a system, each of which corresponds to a microstate that satisfies the macroscopical constraints. This concept is based on the fact that, in practice, a physical law is established by repeating the same experiment a large number of times. Experiments performed under the same conditions differ in the microscopic scale. It is only the probability distribution of the microscopic variables that remains the same for all samples, given specific macroscopic conditions [25].

The three main statistical ensembles that are commonly studied in graduate courses are the microcanonical, canonical and grand canonical ensembles. The microcanonical ensemble describes system of fixed energy. The canonical ensemble, which models a system in weak thermal contact with a heat bath, characterizes a system of fixed temperature. Finally, the grand canonical ensemble refers to a system with a specified temperature and chemical potential [3].

In a recent work [13], coarse-graining maps were used to define a statistical ensemble that applies to situations that are more general than those covered by the thermodynamic ensembles mentioned previously. Consider a system which can be described by an experimentally obtainable density matrix $\rho \in \mathcal{L}(\mathcal{H}_d)$, where $\mathcal{H}_d$ is a Hilbert space of dimention d. We assume that some of the information about the system is lost in the process of measurement, so that ρ is an effective description of an underlying microstate, to which we assign a pure quantum state $\psi = |\psi\rangle\langle\psi| \in \mathcal{L}(\mathcal{H}_D)$. This process is described by a coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \to \mathcal{L}(\mathcal{H}_d)$. In order to satisfy the empirical constraints, we must have:

$$\rho = \Lambda(\psi). \tag{3.0.1}$$

Generally, there will be multiple pure states that satisfy this condition. This leads us to define a

set of all the pure quantum states which abide by the constraints:

$$\Omega_\Lambda(\rho) = \{\psi \in \mathcal{L}(\mathcal{H}_D) | \Lambda(\psi) = \rho\}. \tag{3.0.2}$$

This set can be thought of as a statistical ensemble, and it is possible to think of the process of preparing a quantum state with the macroscopic properties described by ρ as sampling a microstate ψ from it. Since there is no information which would lead one to believe that any particular microstate is more probable than the other, the principle of indifference is invoked, and all of the microstates are assumed to be equally likely. Under the equal probabilities assumption, the most likely microstate to obtain in a random preparation is the average over all microstates in the set. Hence, the best assignment for the microstate is defined in [13] as:

$$\mathcal{A}_\Lambda(\rho) \equiv \overline{\Omega_\Lambda(\rho)}^{\psi} = \int_{\Omega_\Lambda} \mathrm{d}\mu_\psi \mathrm{Pr}_\Lambda(\psi|\rho)\psi. \tag{3.0.3}$$

In this expression, $\mathrm{d}\mu_\psi$ is the uniform Haar measure over pure states, and $\mathrm{Pr}_\Lambda(\psi|\rho)$ is the probability density of the microscopic state ψ given the macroscopic constraints imposed by ρ and the coarse-graining map Λ. Note that, in general, this assignment map is neither completely positive nor linear. This is not a problem, since the assignment map does not represent any kind of physical evolution of the system. It merely expresses the best guess for the underlying microstate of a system given the available information. Furthermore, since, operationally, the assignment is the result of averaging over a set of valid quantum states, it will always yield a valid quantum state.

In the following sections, we will show some examples of the application of this method, using the coarse-graining operators mentioned the previous chapter.

3.1 Open quantum systems

Consider the scenario described in subsection 2.2.1, in which we had a bipartite system composed of a subsystem of intererest interacting with a thermal bath. Let us assume that the state of the subsystem is completely known and can be described as $\rho_A \in \mathcal{L}(\mathcal{H}_A)$. The density matrix ρ_A can be written as:

$$\rho_A = \sum_i P_i \left|\psi_i\right\rangle\!\left\langle\psi_i\right|, \tag{3.1.1}$$

where $\sum_i P_i = 1$, $P_i \geq 0$ and the set of $\{|\psi_i\rangle\langle\psi_i|\}$ defines an orthonormal basis in $\mathcal{L}(\mathcal{H}_A)$. The coarse-graining operator which maps the description of the global system to that of the subsystem ρ_A is $\Lambda = \mathrm{Tr}_B$. In this case, the set of states that satisfy the macroscopic constraints, $\Omega_\Lambda(\rho_A)$, is defined as:

$$\Omega_\Lambda(\rho_A) = \{\psi \in \mathcal{L}(\mathcal{H}_{d_A} \otimes \mathcal{H}_{d_B}) | \operatorname{Tr}_B[\psi] = \rho_A\}. \tag{3.1.2}$$

The states in this set are the purifications of ρ. Using equation 3.1.1, they can be written in the following form [23]:

$$\left|\psi^U\right\rangle = \sum_i \sqrt{P_i} \left|\psi_i\right\rangle_A \otimes U \left|i\right\rangle_B. \tag{3.1.3}$$

In this expression, U is an arbitrary unitary operator acting on $\mathcal{H}_B$, and the set $\{\left|i\right\rangle_B\}$ form an orthonormal basis in $\mathcal{H}_B$. We can show that equation 3.1.3 satisfies $\operatorname{Tr}_B[\psi] = \rho_A$ as follows. Calculating the density matrix of $\left|\psi^U\right\rangle\!\left\langle\psi^U\right|$ we find that:

$$\left|\psi^U\right\rangle\!\left\langle\psi^U\right| = \sum_{i,j} \sqrt{P_i P_j} \left|\psi_i\right\rangle\!\left\langle\psi_j\right| \otimes U \left|i\right\rangle\!\left\langle j\right| U^\dagger. \tag{3.1.4}$$

Applying the partial trace, the result is:

$$\operatorname{Tr}_B \left|\psi^U\right\rangle\!\left\langle\psi^U\right| = \sum_{i,j} \sqrt{P_i P_j} \left|\psi_i\right\rangle\!\left\langle\psi_j\right| \operatorname{Tr} U \left|i\right\rangle\!\left\langle j\right| U^\dagger.$$

Using the properties of the trace operator and the orthonormality of $\{\left|i\right\rangle_B\}$, we find that:

$$\operatorname{Tr} U \left|i\right\rangle\!\left\langle j\right| U^\dagger = \operatorname{Tr} \left\langle j\right| U^\dagger U \left|i\right\rangle = \operatorname{Tr} \left\langle j|i\right\rangle = \delta_{ji}.$$

Therefore,

$$\operatorname{Tr}_B \left|\psi^U\right\rangle\!\left\langle\psi^U\right| = \sum_{i,j} \sqrt{P_i P_j} \left|\psi_i\right\rangle\!\left\langle\psi_j\right| \delta_{ij} = \sum_i P_i \left|\psi_i\right\rangle\!\left\langle\psi_i\right| = \rho_A. \tag{3.1.5}$$

The details of the evalution of the assignment map are included in [13], however, I included here an alternate derivation of the result which can be interesting due to its simplicity. First, we define a state $\left|\psi^{\mathbb{1}}\right\rangle$:

$$\left|\psi^{\mathbb{1}}\right\rangle = \sum_i \sqrt{P_i} \left|\psi_i\right\rangle_A \otimes \left|i\right\rangle_B, \tag{3.1.6}$$

Using the definition of the assignment in 3.0.3, we can write:

$$\begin{aligned}\mathcal{A}_\Lambda &= \int_U \mathrm{d}U(\mathbb{1}_A \otimes U \left|\psi^{\mathbb{1}}\right\rangle\!\left\langle\psi^{\mathbb{1}}\right| \mathbb{1}_A \otimes U^\dagger) \\ &= \int_U \mathrm{d}U(\mathbb{1}_A \otimes U) \left(\sum_{i,j} \sqrt{P_i P_j}\, |\psi_i\rangle\langle\psi_j| \otimes |i\rangle\langle j|\right) (\mathbb{1}_A \otimes U^\dagger) \\ &= \sum_{i,j} \sqrt{P_i P_j}\, |\psi_i\rangle\langle\psi_j| \otimes \int_U \mathrm{d}U U\, |i\rangle\langle j|\, U^\dagger.\end{aligned} \tag{3.1.7}$$

In this expression, $\mathbb{1}_A$ denotes the identity matrix in the corresponding Hilbert space $\mathcal{H}_A$. Since there are no other constraints on the global system, we invoke the principle of indifference and integrate over the set of unitary operators in $\mathcal{H}_B$ uniformly using the Haar Measure, which gives us the result [16]:

$$\int_U \mathrm{d}U U\, |i\rangle\langle j|\, U^\dagger = \frac{\mathrm{Tr}(|i\rangle\langle j|)\mathbb{1}}{B} = \frac{\delta_{ij}\mathbb{1}_{\mathbb{B}}}{B}. \tag{3.1.8}$$

With this, we find that:

$$\begin{aligned}\mathcal{A}_\Lambda(\rho_A) &= \sum_{i,j} \sqrt{P_i P_j}\, |\psi_i\rangle\langle\psi_j|\, \delta_{ij} \otimes \frac{\mathbb{1}_B}{B} \\ &= \sum_i P_i\, |\psi_i\rangle\langle\psi_i| \otimes \frac{\mathbb{1}_B}{B} \\ &= \rho_A \otimes \frac{\mathbb{1}_B}{B}.\end{aligned} \tag{3.1.9}$$

Therefore, given $\rho_A \in \mathcal{L}(\mathcal{H}_A)$, the best description for the state in $\mathcal{L}(\mathcal{H}_A \otimes \mathcal{H}_B)$ is $\mathcal{A}_\Lambda(\rho_A) = \rho_A \otimes \frac{\mathbb{1}_B}{B}$.

Note that the state $\mathcal{A}_\Lambda(\rho)$ is the same that we would obtain by maximizing entropy of the system. Let us consider the interpretation of entropy as the quantity which defines our uncertainty about the set of all conceivable microscopic experiments for a given macrostate [25]. It is possible to see that $\rho_A \otimes \frac{\mathbb{1}_B}{B}$ maximizes this uncertainty, since it is a description that conveys only the information given by our constraints — that the state of the subsystem A is ρ_A. The description for B is an equiprobable mixture of states, which corresponds to a maximum level of ignorance about the environment.

3.2 A blurred and saturated detector

Consider an experiment like the one described in 2.2.2. Given an experimentally obtained density matrix $\rho \in \mathcal{L}(\mathcal{H}_2)$ which contains partial information on the system, our objective is to find its best microscopic description. To calculate the best assingment using equation 3.0.3, we must first define the set Ω_Λ of all possible microstates ψ_i that obey the constraint $\Lambda(\psi_i) = \rho$. To do this, first we write $|\psi\rangle$ in the computational basis:

$$|\psi\rangle = \sum_{i,j=0}^{i,j=1} c_{ij} |ij\rangle . \tag{3.2.1}$$

In this expression, $c_{ij} \in \mathbb{C}$ and $\sum_{ij} |c_{ij}|^2 = 1$. Using this notation, the $\Lambda(\psi_i)$ can be written:

$$\Lambda(\psi) = \begin{pmatrix} |c_{00}|^2 & \frac{1}{\sqrt{3}} c_{00}[c_{01}^* + c_{10}^* + c_{11}^*] \\ \frac{1}{\sqrt{3}} c_{00}^*[c_{01}+, c_{10} + c_{11}] & |c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2 \end{pmatrix}. \tag{3.2.2}$$

From this, we find that the microstates that obey the constraints given by the available information must satisfy:

$$\rho_{00} = |c_{00}|^2 \tag{3.2.3}$$

$$\rho_{01} = \frac{1}{\sqrt{3}} c_{00}[c_{01}^* + c_{10}^* + c_{11}^*] \tag{3.2.4}$$

$$\rho_{10} = \frac{1}{\sqrt{3}} c_{00}^*[c_{01} + c_{10} + c_{11}] \tag{3.2.5}$$

$$\rho_{11} = |c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2. \tag{3.2.6}$$

Integrating over all states in Ω_Λ is equivalent to integrating over all pure states in $\mathcal{H}_4$ that obey equations 3.2.3-3.2.6. The probability density $\Pr_\Lambda$ to be integrated is proportional to a product of delta functions:

$$\begin{aligned} \Pr_\Lambda \propto \delta(|c_{00}|^2 - \rho_{00}) \delta \left(\frac{1}{\sqrt{3}} c_{00}[c_{01}^* + c_{10}^* + c_{11}^*] - \rho_{01} \right) \times \\ \delta \left(\frac{1}{\sqrt{3}} c_{00}^*[c_{01}+, c_{10} + c_{11}] - \rho_{10} \right) \times \\ \delta(|c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2 - \rho_{00}). \end{aligned} \tag{3.2.7}$$

This integral can be exactly calculated. Details on how the calculations are performed can be seen in [13]. After imposing the restrictions via the delta functions, we arrive at the following result:

$$\mathcal{A}_{\Lambda}(\rho) = \begin{pmatrix} \rho_{00} & \frac{\rho_{01}}{\sqrt{3}} & \frac{\rho_{01}}{\sqrt{3}} & \frac{\rho_{01}}{\sqrt{3}} \\ \frac{\rho_{01}}{\sqrt{3}}^* & \frac{\rho_{11}}{3} & \frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6} & \frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6} \\ \frac{\rho_{01}}{\sqrt{3}}^* & (\frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6})^* & \frac{\rho_{11}}{3} & \frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6} \\ \frac{\rho_{01}}{\sqrt{3}}^* & (\frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6})^* & (\frac{|\rho_{01}|^2}{2\rho_{00}} - \frac{\rho_{11}}{6})^* & \frac{\rho_{11}}{3} \end{pmatrix}. \tag{3.2.8}$$

In contrast to the average assignment related to the partial trace case, the assignment for the blurred and saturated detector has a nonlinear dependence on the coefficients of ρ. Note, however, that the nonlinear terms are related to the coherences within the subspace spanned by $\{|01\rangle, |10\rangle, |11\rangle\}$. As mentioned previously, the detector cannot resolve these coherences, hence, all nonlinear terms vanish when the coarse-graining map Λ is applied. This nonlinearity has interesting applications when considering the effective dynamics of the system. However, this discussion is beyond the scope of this work. The reader is encouraged to consult [13] for further details.

In the next chapter, we will see how these results were used to test the quality of algorithms that approximate the average assignment in scenarios such as these.

Chapter 4

Rejection sampling

4.1 The rejection sampling algorithm

The goal of this work was to use computational methods in order to create datasets which could approximate the set defined in equation 3.0.2. The first approach was that of brute force rejection sampling. As discussed in chapter 3, we assume that we have access to a description $\rho \in \mathcal{L}(\mathcal{H}_d)$ of our system that contains only part of the information that describes its entire structure, which belongs to a larger space $\mathcal{L}(\mathcal{H}_D)$. Furthermore, the loss of information can be represented by a coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \to \mathcal{L}(\mathcal{H}_d)$.

The idea of this method is that of trial and error. One uniformly samples random quantum states $\psi_i = |\psi_i\rangle\langle\psi_i|$ such that $\psi_i \in \mathcal{L}(\mathcal{H}_D)$, then checks if they satisfy the constraints of equation 3.0.1. Randomly sampling a random quantum state that satisfies this equation exactly is, however, impossible, so it was necessary to impose a threshold condition. The criteria for accepting or rejecting quantum states was based on the trace distance between $\Lambda(\psi_i)$ and ρ, which is defined [23]:

$$T(\Lambda(\psi_i), \rho) = \frac{1}{2} \operatorname{Tr} |\Lambda(\psi_i) - \rho|, \tag{4.1.1}$$

where, for any operator A, $|A| = \sqrt{A^\dagger A}$.

Physically, the trace distance represents a measure of the distinguishability between two states. Hence, we require that the random states generated satisfy $T(\Lambda(\psi_i), \rho) \leq \epsilon$ where ϵ is some fixed threshold parameter, in order to be included in our dataset. This process is repeated for a number of iterations, until the dataset reaches a certain size D. The resulting dataset, denoted $\widetilde{\Omega}_\Lambda(\rho)$ is a numerical approximation to the set $\Omega_\Lambda(\rho)$ defined in equation 3.0.2. Mathematically, it can be written:

$$\widetilde{\Omega}_\Lambda(\rho) = \{\psi_i | T(\Lambda(\psi_i), \rho) \leq \epsilon\}, \tag{4.1.2}$$

where , $0 \leq i \leq D-1$. The approximation to the assignment $\mathcal{A}_\Lambda(\rho)$ defined in equation 3.0.3 is the average of our dataset, given by:

$$\widetilde{\mathcal{A}}_\Lambda(\rho) = \overline{\widetilde{\Omega}_\Lambda}(\rho) = \frac{1}{D}\sum_{i=0}^{D-1} \psi_i. \tag{4.1.3}$$

In summary, these are the steps for calculating the average microstate using rejection sampling:

1. Generate a random quantum state $\psi = |\psi\rangle\langle\psi|$, with $\psi \in \mathcal{L}(\mathcal{H}_D)$.
2. Apply the coarse-graining map Λ to ψ_i.
3. Calculate the trace distance between $\Lambda(\psi_i)$ and ρ. This quantity is denoted $T(\Lambda(\psi_i), \rho_c)$.
4. Check if $T(\Lambda(\psi_i), \rho) \leq \epsilon$, where ϵ is an arbitrary threshold parameter. If the condition holds, add ψ_i to the set of states that satisfy it.
5. Repeat steps 1-4 until the dataset reaches a size D. The resulting dataset is denoted $\widetilde{\Omega}_\Lambda(\rho)$ and is defined in equation 4.1.2.
6. Calculate the average of $\widetilde{\Omega}_\Lambda(\rho)$, as shown in equation 4.1.3. This is the approximate assignment$\widetilde{\mathcal{A}}_\Lambda(\rho)$.

This program was implemented in Python using the QuTiP library [24], which allows one to generate random quantum states and perform operations on them. A flowchart illustrating this algorithm is shown in Figure 4.1.1.

Two main variables influence how good of an approximation the assignment is to the desired average microstate. The first one is the threshold parameter ϵ. As we can see from equation 4.1.1, as $\epsilon \to 0$, $\Lambda(\psi_r) \to \rho$. In other words, the randomly generated states in the dataset satisfy the desired conditions more exactly. The second variable is the size D of the dataset. The law of large numbers tells us that the averages calculated from an experiment involving a certain number of trials become closer to the theoretical average as the number of trials increases. The process of finding the average microstate by calculating the average of a set of randomly generated quantum states can be thought of as an experiment of this sort, in which the number of trials is equal to the size of the dataset. Therefore, we expect the approximated average to be closer to the theoretical ensemble average as D increases.

Evidently, the size of the dataset is directly related to the number iterations of the algorithm. Generating larger datasets reduces statistical error, but increases the time taken by the program to calculate the approximate assignment. Therefore, it was necessary to consider these two factors in

order to set a minimum value of D at which to stop the loop. This process shall be discussed in the following subsection.

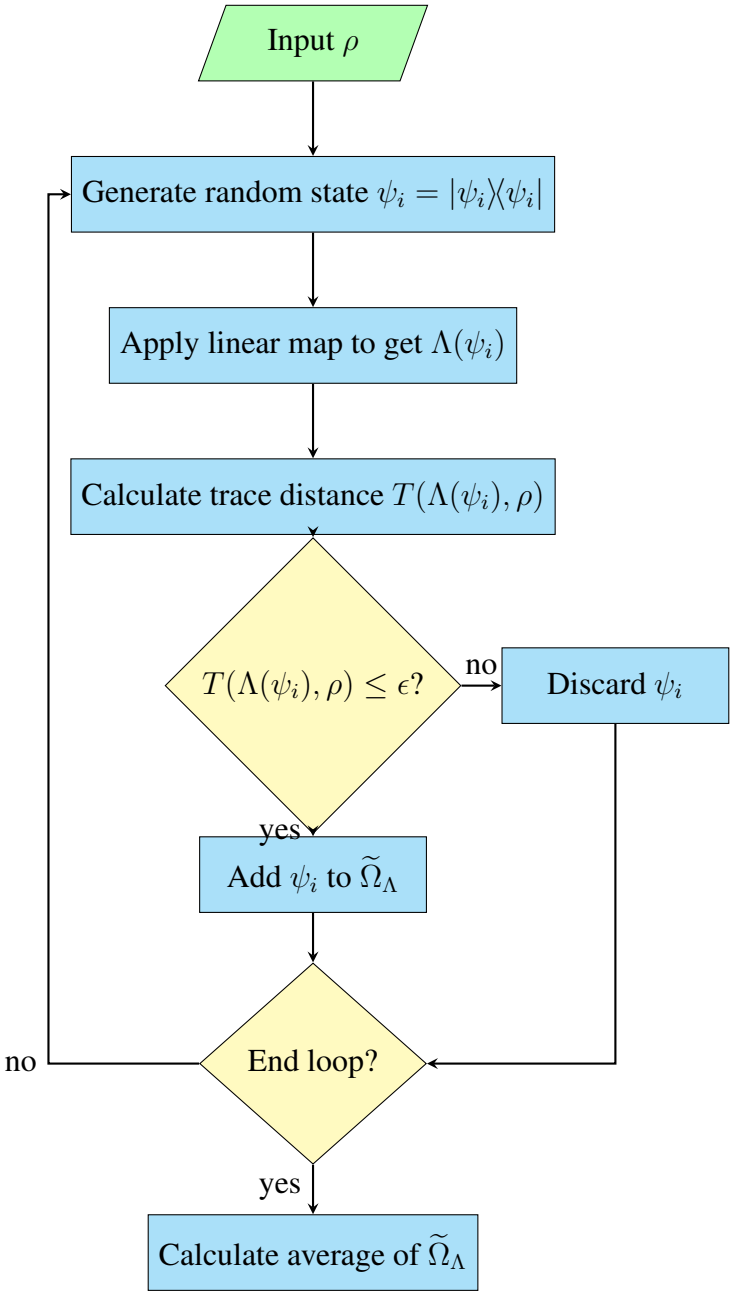


Figure 4.1.1: The rejection sampling loop

4.2 Convergence criteria

Let $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}(\rho)$ be the approximate assignment given by the rejection sampling algorithm for an instance in which the size of the dataset was D_n. Furthermore, let the set of sets be ordered, so that $D_1 < D_2 < \ldots < D_n$. My first objective was to determine the qualitative behavior of $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}(\rho)$ as the datasize size increased, in order to determine a good cutoff point for the rejection sampling loop.

Since the only relevant variables were the sizes of the dataset, there were no particular restrictions on the choice of Λ, ρ or ϵ to run this test with. It was only necessary that all of these inputs remained constant. I decided to use the blurry detector coarse-graining map, for which the analytical solution for $\mathcal{A}_{\Lambda}(\rho)$ is known, and calculate the quality of the approximation as D_n varied. For this, I used two different metrics. The first one was the trace distance $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_n}, \mathcal{A}_{\Lambda})$, as defined in equation 4.1.1. Here, I omitted the dependence of the variables on ρ in order to simplify the notation.

One possible limitation of this metric is that a relatively constant trace distance $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_n}, \mathcal{A}_{\Lambda})$ does not necessarily imply that the algorithm's output has converged to a fixed value. It is possible that, instead, the output $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ is fluctuating around states within the same trace distance of $\mathcal{A}_{\Lambda}$. Because of this, I considered a second metric, the absolute value of the maximum difference between the coefficients of the matrix representation of $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ and $\mathcal{A}_{\Lambda}$. One first takes the difference matrix, $\widetilde{\mathcal{A}}_{\Lambda}^{D_n} - \mathcal{A}_{\Lambda}$ [26]:

$$\begin{bmatrix} \left\langle i^{(1)}\right|\widetilde{\mathcal{A}}_{\Lambda}^{D_n}\left|j^{(1)}\right\rangle - \left\langle i^{(1)}\right|\mathcal{A}_{\Lambda}\left|j^{(1)}\right\rangle & \cdots & \left\langle i^{(1)}\right|\widetilde{\mathcal{A}}_{\Lambda}^{D_n}\left|j^{(N)}\right\rangle - \left\langle i^{(1)}\right|\mathcal{A}_{\Lambda}\left|j^{(N)}\right\rangle \\ \vdots & \ddots & \vdots \\ \left\langle i^{(N)}\right|\widetilde{\mathcal{A}}_{\Lambda}^{D_n}\left|j^{(1)}\right\rangle - \left\langle i^{(N)}\right|\mathcal{A}_{\Lambda}\left|j^{(1)}\right\rangle & \cdots & \left\langle i^{(N)}\right|\widetilde{\mathcal{A}}_{\Lambda}^{D_n}\left|j^{(N)}\right\rangle - \left\langle i^{(N)}\right|\mathcal{A}_{\Lambda}\left|j^{(N)}\right\rangle \end{bmatrix}, \tag{4.2.1}$$

where we have taken an orthonormal basis $\langle i|j\rangle = \delta_{ij}$, and both matrices have dimentions $N \times N$. After that, one calculates the absolute values of the coefficients of this matrix and takes the maximum value.

In order to gain an understanding of the influence of statistical error on my results, I plotted these metrics for the states $\rho = |1\rangle\langle 1|$ and $\rho = |0\rangle\langle 0|$ using the blurry detector coarse-graining. The reason for this choice is that these two cases represent, respectively, an example of a macrostate for which it is easy to generate a large number of matrices in the dataset D, and one for which it is very hard to do so.

Setting the input parameters of the program to $\rho = |1\rangle\langle 1|$ and $\epsilon = 0.2$, I had the program calculate $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for $D_n = \{100, 200, \ldots, 3000\}$. The graph of how both the trace distance and the maximum coefficient metric varied with the dataset size can be seen in figure 4.2.1. The mean of

the output values for $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_n}, \mathcal{A}_{\Lambda})$ was approximately 0.30, and the range was of approximately 0.04. The minimum value occured at $D_n = 100$, and the maximum occured at $D_n = 300$. For the maximum coefficient metric, the mean was approximately 0.14, and the range, approximately 0.02. The minimum value occured at $D_n = 900$ and the maximum, at $D_n = 200$. In conclusion, for both metrics, the most significant statistical variations occured within values of D up to 900.

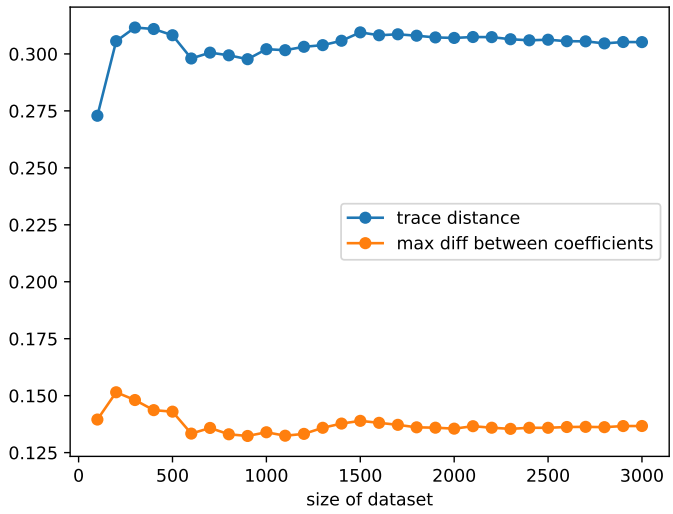


Figure 4.2.1: Graph showing the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for different dataset sizes D_n, and the analytically derived assignment $\mathcal{A}_{\Lambda}$ for the blurry detector coarse-graining map, with $\rho = |1\rangle\langle 1|$ and $\epsilon = 0.2$. The trace distance metric is defined in equation 4.1.1, and the maximum difference between coefficients is largest of the absolute values of the coefficients in the matrix shown in equation 4.2.1. For both metrics, the most significant statistical fluctuations occur within values of D up to 900, and, after that, they plateau.

As mentioned previously, in general, we do not know how to calculate the analytical value of $\mathcal{A}_{\Lambda}$ for an arbitrary coarse-graining operator. Therefore, it was of utmost importance to establish a way of studying the behavior of $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for different dataset sizes that was independent of the analytical prediction for $\mathcal{A}_{\Lambda}$. In order to do this, I decided to use the metrics introduced before to analyze the variation between $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for consecutive values of D_n. For this I used the previously generated data, consisting of a set of states $\left\{\widetilde{\mathcal{A}}_{\Lambda}^{D_n}\right\}$ with $D_n = \{100, 200, \ldots, 3000\}$. The metrics which were evaluted were the trace distance between consecutive outputs, $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_{n-1}}, \widetilde{\mathcal{A}}_{\Lambda}^{D_n})$, and the maximum of the absolute values of the coefficients of the difference matrix $\widetilde{\mathcal{A}}_{\Lambda}^{D_{n-1}} - \widetilde{\mathcal{A}}_{\Lambda}^{D_n}$. The results can be seen in figure 4.2.2.

As can be seen in Figure 4.2.2, both metrics decrease rapidly for datasets with up to 1000 elements, and after that, the output variations being to plateau. One can argue that the statistical fluctuations observed when analyzing the difference between the output and the analytical result are related to the output variations that occur within $200 \leq D_n \leq 1000$. Once the variations in the output $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ decrease, the distance between the numerical approximation and the analytical result

begins to converge to a constant value. For $1500 \leq D_n \leq 3000$, the mean value of $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_{n-1}}, \widetilde{\mathcal{A}}_{\Lambda}^{D_n})$ was approximately 0.003, and the mean value for the maximum coefficient metric was approximately 0.002. These values are quite small compared to the error between the analytical result and the program's output, therefore $D = 1500$ was set as the ideal minimum value at which to stop the rejection sampling loop.

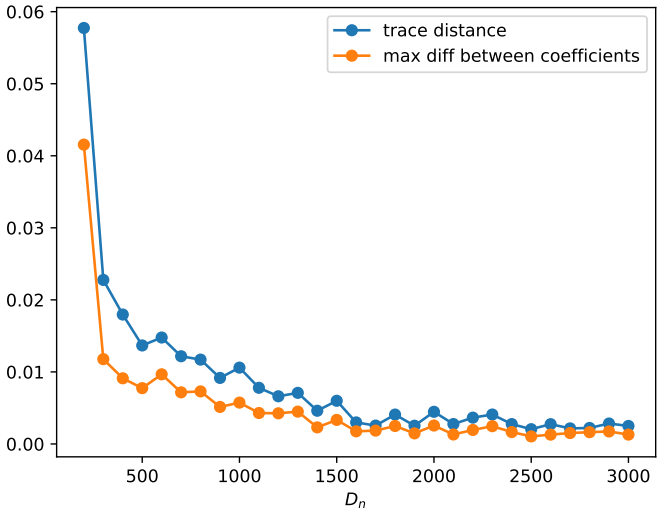
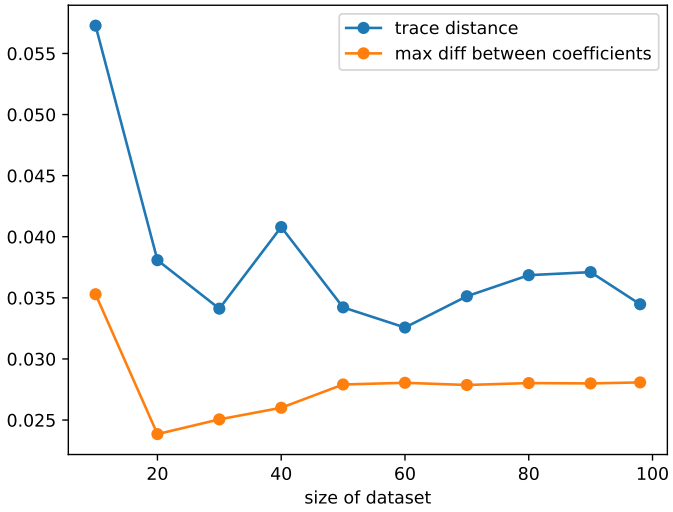


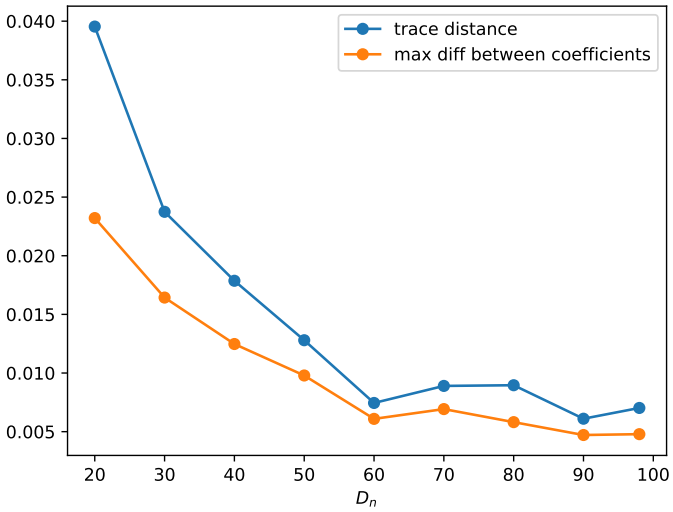
Figure 4.2.2: Graph showing the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for consecutive values of D. The coarse-graining map used was the blurry detector one, with $\rho = |1\rangle\langle 1|$ and $\epsilon = 0.2$. The trace distance metric is defined in equation 4.1.1, and the maximum difference between coefficients is largest of the absolute values of the coefficients in the matrix shown in equation 4.2.1. Both metrics decrease rapidly for datasets with up to 1000 elements, and after that, plateau and remain within values inferior to 0.005.

For certain choices of coarse-graining maps and macrostates ρ, however, achieving this ideal minimum value can be computationally slow. For example, in the case of the blurry detector coarse-graining map with $\rho = |0\rangle\langle 0|$ and $\epsilon = 0.05$, running the loop for 13000000 iterations yielded a dataset with only 98 states. When evaluating the behavior of the trace distance $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_{n-1}}, \widetilde{\mathcal{A}}_{\Lambda}^{D_n})$ and maximum coefficient metrics for this case, it was observed that the mean value of $T(\widetilde{\mathcal{A}}_{\Lambda}^{D_{n-1}}, \widetilde{\mathcal{A}}_{\Lambda}^{D_n})$ was approximately 0.008, and the mean value for the maximum coefficient metric was approximately 0.006. Despite the statistical variations being higher than observed for the case of $\rho = |1\rangle\langle 1|$, $\epsilon = 0.2$ and $1500 \leq D_n \leq 3000$, the differences between the outputs and the analytical results were generally much lower. These results can be seen in figure 4.2.3.

This suggests that statistical fluctuations do not influence the algorithm's quality in approximating $\mathcal{A}_{\Lambda}$ as much as other factors such as the characteristics of the map, the choice of macrostate and the ϵ parameter. In the following section, we will discuss the influence of ϵ in further detail.



(a) Graph showing the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for different dataset sizes D_n, and the analytically derived assignment $\mathcal{A}_{\Lambda}$.



(b) Graph showing the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for consecutive values of D.

Figure 4.2.3: Graph (a) shows the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for different dataset sizes D_n, and the analytically derived assignment $\mathcal{A}_{\Lambda}$ and graph (b) the difference between the outputs $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ for consecutive values of D, for the case of the blurry detector coarse-graining map with $\rho = |0\rangle\langle 0|$ and $\epsilon = 0.05$. The trace distance metric is defined in equation 4.1.1, and the maximum difference between coefficients is largest of the absolute values of the coefficients in the matrix shown in equation 4.2.1. Due to the characteristics of this map, obtaining large datasets for this choice of macrostate is difficult, hence the datasets were much smaller. The variations between consecutive outputs were observed to have a similar behavior to that of the case of $\rho = |1\rangle\langle 1|$ and $\epsilon = 0.2$, however, the differences between $\widetilde{\mathcal{A}}_{\Lambda}^{D_n}$ and the analytical result were much smaller.

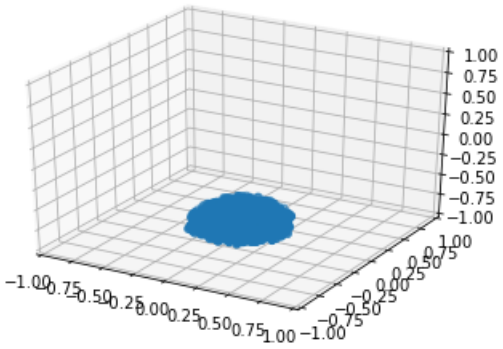
4.3 The ϵ parameter

As mentioned previously, the constant ϵ is a threshold parameter for accepting or rejecting random quantum states into the algorithm's dataset. The dataset $\widetilde{\Omega}_\Lambda(\rho)$ generated by the rejection sampling algorithm is meant to approximate the ensemble of microstates that satisfy the conditions given by $\Lambda(\psi_i) = \rho$. However, the probability of randomly sampling a state that satisfies this equation exactly is zero. Hence the need for a threshold parameter.

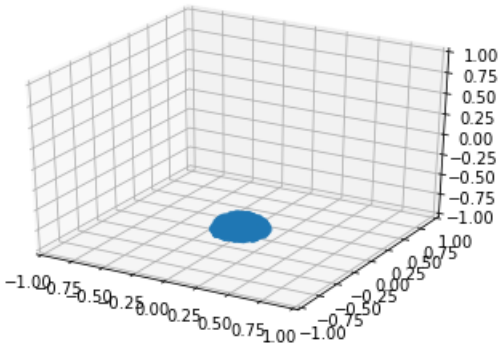
In order to better visualize the effect of reducing ϵ on dataset, I created a plot using the same method that was explained in subsection 2.2.2. I ran the rejection sampling algorithm for blurry detector coarse-graining map with $\rho = |1\rangle\langle 1|$ and three different threshold parameters: $\epsilon = 0.2$, $\epsilon = 0.1$ and $\epsilon = 0.05$. I took the Bloch vectors of $\Lambda(\psi_i)$ for the states which passed the threshold conditions for each case and plotted the result in a 3D graph, which can be seen in figure 4.3.1.

From these images, it can be observed that decreasing ϵ has the effect of shrinking the region the microstates in the dataset are mapped to around the vicinity of ρ. Consequently, one can intuit that for $\epsilon \to 0$ all states will be mapped exactly unto ρ. In other words, reducing the ϵ parameter will result in the microstates included in $\widetilde{\Omega}_\Lambda(\rho)$ satisfying the macroscopic constraints more exactly, however, the number of iterations necessary to attain a satisfactory number of microstates will increase.

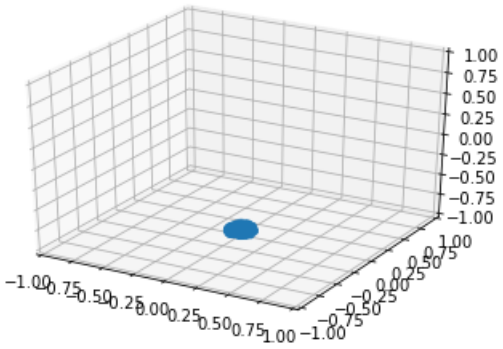
In order to better understand the influence of ϵ on the numerical approximation of the assignment $\widetilde{\mathcal{A}}_\Lambda(\rho)$, the rejection sampling loop was applied to a set of different values of $\epsilon \in \{0.5, 0.4, 0.3, 0.2, 0.1, 0.05\}$ for each of the studied macrostates ρ. Two different coarse-graining maps were used, the partial trace and the blurry detector coarse-graining map. Since, for both of these maps, the analytical solution for the assignment is known, I used the trace distance $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ as a measure of the program's error and plotted it against ϵ to observe how the algorithm's performanced varied as the threshold conditions became stricter. The following section shows the results of this study.



(a) $\epsilon = 0.2$



(b) $\epsilon = 0.1$



(c) $\epsilon = 0.05$

Figure 4.3.1: Visualization of $\Lambda(\psi_i)$ for datasets with different values of ϵ. The algorithm was run with the blurry detector coarse-graining map and $\rho = |1\rangle\langle 1|$. From this figure, it is possible to see that as we reduce the threshold parameter, the states $\Lambda(\psi_i)$ become more localized around the macrostate ρ. Also note that the shape of the volume to which selected microstates are mapped to on the Bloch sphere is the intersection between a sphere of radius ϵ centered on the Bloch coordinates of the macrostate and the Bloch sphere itself.

4.4 Results

4.4.1 The partial trace

In studying the algorithm's performance for the partial trace coarse-graining map, I considered four different macroscopic states: $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.9 \end{bmatrix}$, $\rho_3 = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.7 \end{bmatrix}$ and $\rho_4 = \frac{\mathbb{1}_2}{2}$. When plotting the error in the approximation, measured by $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus the parameter ϵ, it was observed that $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ fluctuated around 0.019 ± 0.006 for the maximally mixed state, ρ_4. For all of the other states $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ was observed to decrease as ϵ decreased. At $\epsilon = 0.5$, ρ_1 had the highest value for $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$, followed by ρ_2, ρ_3 and finally ρ_4. At $\epsilon = 0.05$, the algorithm gave an approximation with an error measure $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda) \leq 0.025$ for all of the states. This plot can be seen in figure 4.4.1.

An interesting aspect of this result is that the states ρ_1 to ρ_4 have decreasing values of purity, as defined by [23]:

$$P(\rho) = \mathrm{Tr}\big(\rho^2\big). \tag{4.4.1}$$

The purity of a state is always less than or equal to 1, with the equality occuring if and only if ρ is pure. For the macrostates used in this test, $P(\rho_1) = 1$, $P(\rho_2) = 0.82$, $P(\rho_3) = 0.58$ and $P(\rho_4) = 0.5$. This result suggests that the threshold conditions one must impose to achieve an approximation such that $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda) \leq 0.025$ are stricter for pure states than they are for mixed states. Furthermore, the lower the purity of the state, the higher ϵ can be.

Note that applying the partial trace to a two qubit state will only yield a pure state if there is no entanglement, i.e., the system is in a product state. To see this, recall that any pure state in a bipartite system can be written in terms of its Schmidt decomposition [23]:

$$|\psi_{AB}\rangle = \sum_i \lambda_i \, |i_A\rangle \, |i_B\rangle \,, \tag{4.4.2}$$

where the Schmidt coefficients λ_i are all non-negative numbers with $\sum_i \lambda_i = 1$, and the orthonormal bases $|i_A\rangle$ and $|i_B\rangle$ are called the Schmidt bases. Applying the partial trace to this state, we see that:

$$\mathrm{Tr}_B \left[|\psi_{AB}\rangle\langle\psi_{AB}|\right] = \sum_i \lambda_i^2 \, |i_A\rangle\langle i_A| \,. \tag{4.4.3}$$

If ρ_A is pure, $\lambda_i = 1$ for some value of i, and 0 for all the others. Hence, $|\psi_{AB}\rangle = |i_A\rangle \, |i_B\rangle$ is a product state. Previous results in the field [27] have shown that that entangled states are more probable to be obtained through random sampling in $\mathcal{L}(\mathcal{H}_4)$ in comparison to non-entangled ones.

This means that it is more difficult to sample states that satisfy the macroscopic conditions $\mathrm{Tr}_B = \rho$ when ρ is pure than when it is mixed. Therefore, the errors involved in calculating the approximated assignment for pure states are larger and the threshold conditions must be stricter in order to obtain better results.

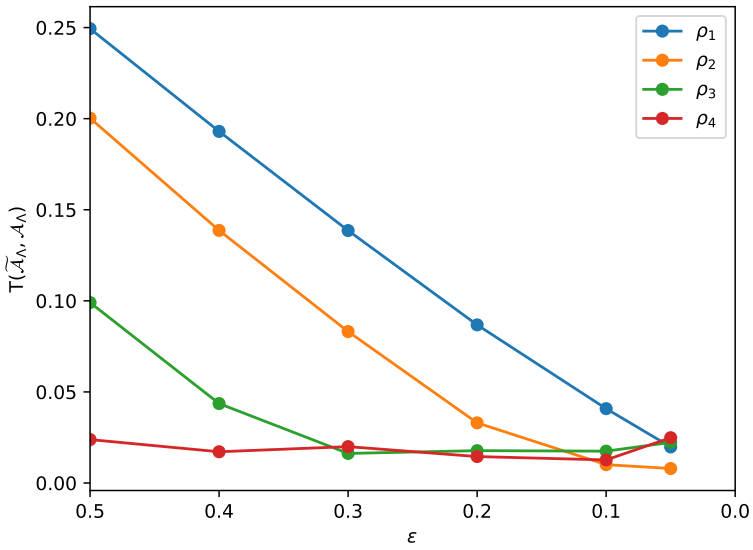


Figure 4.4.1: Rejection sampling error by ϵ for four different macroscopic states: $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.9 \end{bmatrix}$, $\rho_3 = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.7 \end{bmatrix}$ and $\rho_4 = \frac{\mathbb{1}_2}{2}$. Macrostate ρ_1 is a pure state, ρ_4 is maximally mixed state, and the other two have intermediate values of purity. Note that the axis is oriented in the direction of decreasing ϵ. Observe that, for higher of ϵ, the algorithm performs worse on states with higher purity. As ϵ decreases, however, the quality of the approximation becomes similar for all states.

4.4.2 Blurry detector coarse-graining map

In studying the algorithm's performance for the blurry detector coarse-graining map, I considered four different macroscopic states: $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$. The states ρ_1 and $\rho_2$were chosen because, given the assymetry of the image space of the coarse-graining map discussed in subsection 2.2.2, I was interested in observing the difference in how the algorithm performed for each of them. State $\rho_3 = \frac{\mathbb{1}_2}{2}$ was chosen because I wanted to see how the algorithm would behave when ρ was a mixed state. Finally, state $\rho_4$was chosen to observe the algorithm's performance on a macrostate which exhibitted coherences between the states $|0\rangle$ and $|1\rangle$.

A graph of the error measure $T(\widetilde{\mathcal{A}}_\Lambda(\rho), \mathcal{A}_\Lambda)$ versus ϵ for each of the states can be seen in Figure 4.4.2. Two things can be observed in this graph. The first is that, as expected, the error measure decreases with ϵ for all of the ρ analyzed. The second is that the algorithm performs better for some

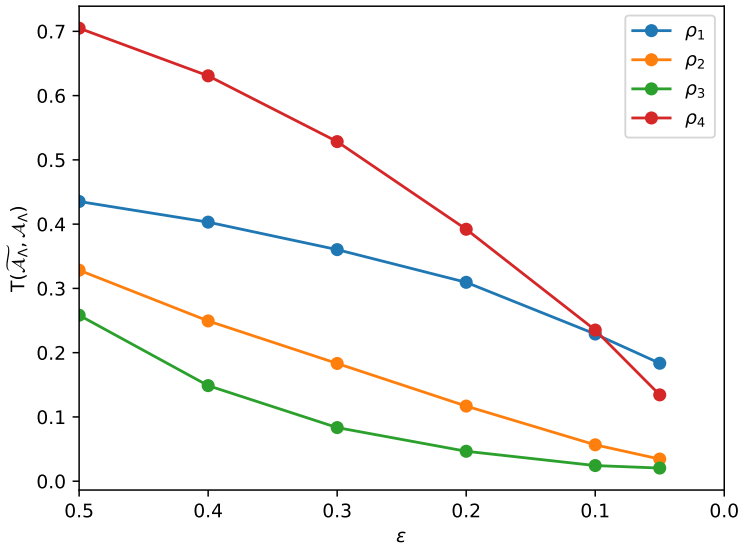


Figure 4.4.2: Rejection sampling error by ϵ for four different macroscopic states: $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$. Note that the axis is oriented in the direction of decreasing ϵ. As expected, the algorithm's performance improves as ϵ decreases. However, the quality of the approximation given by the algorithm varies depending on the input macrostate.

macrostates ρ than for others.

The macrostate that exhibited the lowest values of $T(\widetilde{\mathcal{A}}_\Lambda(\rho), \mathcal{A}_\Lambda)$ for all ϵ was $\rho_3 = \frac{\mathbb{1}_2}{2}$. In order to better understand this, let us consider the region of space to which the Bloch vectors of $\Lambda(\psi_i)$ for $\psi_i \in \widetilde{\Omega}_\Lambda(\rho_3)$ are mapped to. Using equation 2.2.9, it is possible to verify that the Bloch coordinates of ρ_3 are $[0, 0, 0]$, i.e., exactly the center of the Bloch sphere. The region described by $T(\Lambda(\psi_i), \rho_3) \leq \epsilon$ is approximately a sphere of radius ϵ. It is possible to argue that the spherical symmetry of this region helps the algorithm's performance, as errors in opposing directions will tend to cancel each other out when the average over the whole set is performed. This was not the case with any of the other macrostates analyzed, since they were all pure states. As we know, pure 1 qubit states are situated on the surface of the Bloch sphere. This means that the states $\Lambda(\psi_i)$ for $\psi_i \in \widetilde{\Omega}_\Lambda(\rho_{i\neq 3})$ will not be distributed in a radially symmetric way around ρ, because if they were, the set would include vectors that are outside the Bloch sphere, and therefore, not valid quantum states.

Another interesting observation was that the algorithm performed better for $\rho_2 = |0\rangle\langle 0|$ than for $\rho_1 = |1\rangle\langle 1|$, for all values of ϵ. As mentioned previously, the only microscopic state that satisfies the macroscopic conditions described by ρ_2 is the state $|00\rangle\langle 00|$. In contrast, there are an infinite number of microstates that satisfy the macroscopic conditions described by ρ_1. A consequence of this is that there is a smaller variation between the matrices in the dataset $\widetilde{\Omega}_\Lambda(\rho_2)$ as opposed to those in $\widetilde{\Omega}_\Lambda(\rho_1)$. This leads to a better performance for ρ_2, despite the computational difficulties in

sampling states that satisfy the microscopic conditions through this method.

The algorithm performed the worst on ρ_4 for ϵ up to 0.1. It is possible that for larger values of ϵ, the sets $\widetilde{\Omega}_\Lambda(\rho_4)$ contained more noise in comparison to the other states. The Bloch coordinates of ρ_4 are $[0.8, 0, -0.6]$, which means it is in the southern hemisphere. As discussed in chapter 2, when randomly sampling states ψ_i and applying to them the blurry detector map, the resulting dataset will contain states $\Lambda(\psi_i)$ that are mostly distributed about the south pole. Let us denote the set $\Lambda \circ \widetilde{\Omega}_\Lambda(\rho_4)$, where symbol $\circ$ denotes the element-wise application of Λ on $\widetilde{\Omega}_\Lambda$. For larger values of ϵ, it is possible that $\Lambda \circ \widetilde{\Omega}_\Lambda(\rho_4)$ contained more states in the neighborhood of more probable macrostates, such as ρ_1, than states in the vicinity of ρ_4. Hence, stricter threshold conditions were needed to eliminate the noise.

Evidently, this is not an extensive study of how the algorithm performs for all ρ. The way the program is constructed makes it inconvenient to run for many different macrostates, because each time the value of ρ is changed, the loop has to be run again. This difficulty lead to the development of a method that circumvents this problem, which will be discussed in the following chapter.

Chapter 5

Partition sampling

5.1 Motivation for this approach

One clear drawback to the rejection sampling algorithm shown in the last section is that it only saves the random microstates $\psi_i = |\psi_i\rangle\langle\psi_i|$ such that $T(\Lambda(\psi_i), \rho) \leq \epsilon$. This means that for each different value of the macrostate ρ, we must run the algorithm again from the beginning. It would be more efficient to be able to create one large dataset of uniformly generated quantum states, save it and reuse it for each input ρ. In order to try to solve this problem, I developed an algorithm which I will refer to in this work as *partition sampling*.

The main objective of this method is to create a labeled dataset of random microstates ψ_i such that, given any macrostate ρ, an approximate average microstate can be calculated. Let us assume, as before, that we are using a coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \to \mathcal{L}(\mathcal{H}_d)$. Any quantum state can be parametrized in such a way as to be interpreted as a point in some vector space. The core idea of this method is to label each ψ_i based on the region of space in which $\Lambda(\psi_i)$ lies. This is done through the following steps:

1. Generate a dataset D with uniformly distributed pure quantum states $\psi_i \in \mathcal{L}(\mathcal{H}_D)$.
2. Apply the coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \to \mathcal{L}(\mathcal{H}_d)$ to all ψ_i. The resulting set will be denoted $\Lambda \circ D$.
3. Divide the points in $\Lambda \circ D$ into k partitions. Each partition is denoted K_n, with $n = \{0, 1, ..., k\}$.
4. Label each element of D in the following manner. Given a state $\Lambda(\psi_i) \in \Lambda \circ D$, if $\Lambda(\psi_i)$ belongs to a certain partition K_n, then the label of ψ_i is n. Save the labels in a dataset L.

This process only has to be performed once. Once it is done and all of the datasets are saved, the approximate assignment for any given ρ can be calculated as follows:

1. For a given macrostate ρ, determine which partition ρ belongs to.

2. Let us say that ρ belongs to the partition K_n. The set $\widetilde{\Omega}_\Lambda(\rho)$ which approximates the one given by equation 3.0.2 will be composed of the elements of D that possess the label n.

3. The approximate assignment $\widetilde{\mathcal{A}}_\Lambda(\rho)$ is given by the average over $\widetilde{\Omega}_\Lambda(\rho)$.

Note that, for this method, the number k of partitions we choose to divide $\mathcal{H}_d$ into has a role equivalent to that of the ϵ parameter in the rejection sampling method. This is because using more partitions means dividing the space into smaller regions. A consequence of smaller partitions is that, on average, there will be less states in each partition. On the other hand, the states in that partition will satisfy the macroscopic constraints more exactly.

A question one may ask themselves when attempting to implement partition sampling is: what is the best way to partition the data points? It was this question that piqued my interest in an unsupervised machine learning method called k-means clustering. The next section will discuss the k-means++ algorithm and how it was used in this work.

5.2 Partitioning with k-means++

K-means clustering is an algorithm that originates from signal processing [28]. Its objective is to partition n observations into k clusters. The criteria for including an observation into a cluster is that the observation must be closer to the center of its respective cluster than to any of the other cluster centers. In mathematical terms, this is equivalent to partitioning the data space into Voronoi cells [29].

Given a positive integer k and a set of n data points in $\mathbb{R}^d$, the objective of the algorithm is to choose k points such that the distance between each data point and its closest center is minimal. Denoting the set of centers as C and the set of data points as X, the cost function ϕ that this algorithm attempts to minimize is:

$$\phi = \sum_{x \in X} \min_{c \in C} |x - c|^2. \tag{5.2.1}$$

This quantity ϕ is commonly called the inertia in the context of machine learning, and shall be reffered to as such in this text. In this work, I used the algorithm k-means++ [30], an improvement upon the traditional k-means algorithm [28]. The steps of the algorithm are as follows:

1. Randomly choose k initial centers $C = \{c_1, c_2, \cdots c_k\}$ from X.

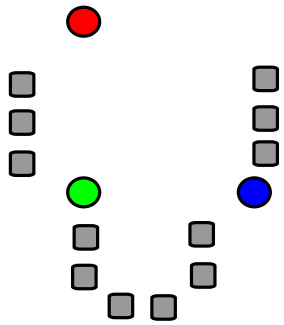
2. For each $i \in \{1, \cdots k\}$, set the cluster K_i to be the set of points in X that are closer to c_i than they are to all c_j, for all $j \neq i$.

3. For each $i \in \{1, \cdots k\}$, update c_i to be the center of mass of all points in K_i: $c_i = \frac{1}{|K_i|} \sum_{x \in K_i} x$. Note that c_i does not necessarily need to correspond to a data point in X.

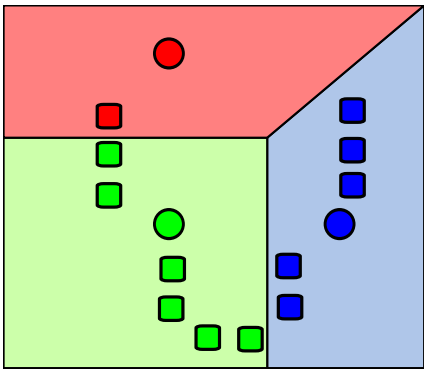
4. Repeat steps 2 and 3 until C no longer changes.

Figure 5.2.1 illustrates the steps of the traditional k-means algorithm. More details on k-means++ can be found in [30]. The algorithm was implemented in Python using the scikit-learn library [31].

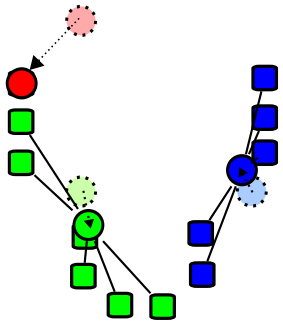
Let us consider the minimization objective of k-means and how it will affect the distribution of the partitions for the cases in which it will be applied. From what we have seen in the previous chapters, we know that, in general, the states in $\Lambda \circ D$ will not be uniformly distributed in $\mathcal{L}(\mathcal{H}_d)$. This means that, if we represent this set as a series of points in $\mathbb{R}^d$, there will be regions of the space with higher density than others. Given the learning objective of k-means++, the algorithm will tend to create smaller partitions where the data concentration is higher, and larger partitions where the data is sparse. This is physically interesting because, recall, the regions of space that are more densely populated correspond macrostates which are more likely to be observed for any given experiment. Hence, partitioning using k-means++ allows for a greater precision in calculating $\widetilde{\mathcal{A}}_\Lambda(\rho)$ for the most commonly observed macrostates. This is the physical motivation behind this choice of algorithm.



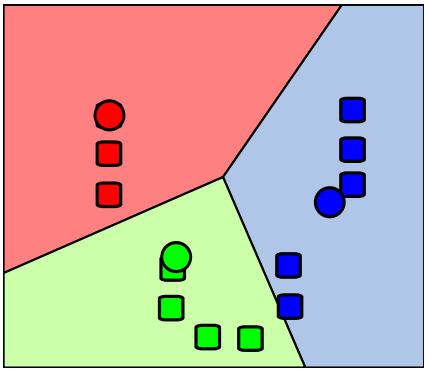
(a) Step 1 of the k-means clustering method. Squares represent data points, and circles represent the cluster centers. At first, the initial cluster centers are chosen at random. Note that in the case illustrated, the cluster centers do not correspond to data points. This is one of the differences between k-means and k-means++.



(b) Step 2 of the k-means clustering method. All points in the dataset are labeled according to which cluster center they are closest to. The colored partitions represent the clusters, and datapoints are colored in accordance to which cluster they were placed in.



(c) Step 3 of the k-means clustering method. The cluster centers are redefined as the centers of mass of all points in their respective clusters. Note that since there was initially only one data point in the red cluster, the cluster center is placed exactly on the same position as this single data point.



(d) After step 3, the points in the dataset are relabeled according to which cluster center they are closest to, resulting in a new distribution of the data along the partitions.

Figure 5.2.1: Figures ilustrating the steps of the k-means algorithm. Steps 2-3 are repeated until, when recalculating the cluster centers, the algorithm finds that they no longer change coordinates. All images shown here were created by Weston.pace and distributed under a CC BY-SA 3.0 license.

5.3 The k parameter and partition size

I applied this method to the previous cases, described by the partial trace and the blurry detector coarse-graining maps. The partition sampling algorithm requires the use of two datasets. The dataset D contained the microstates, a set of uniformly sampled pure quantum states $\psi_i \in \mathcal{L}(\mathcal{H}_4)$. The second set, $\Lambda \circ D$, contained the corresponding coarse-grained states $\Lambda(\psi_i) \in \mathcal{L}(\mathcal{H}_2)$. Since the k-means algorithm can only be applied to data consisting of arrays of real numbers, the macrostates were not stored in their density matrix representation, but in their Bloch vector representation.

Analogously to what was done in chapter 4, I applied partition sampling using a variety of k values and analyzed the behavior of the trace distance $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ as k varied. As before, $\widetilde{\mathcal{A}}_\Lambda$ denotes the numerical approximation of the assignment given by our algorithm, and $\mathcal{A}_\Lambda$ the analytical result. The number k of clusters the dataspace is divided into in the partition sampling method has a comparable role to that of the ϵ parameter in the rejection sampling method. In the rejection sampling algorithm, reducing ϵ reduces the region of the Bloch sphere occupied by the states $\Lambda(\psi_i)$ which satisfy the condition $T(\Lambda(\psi_i), \rho) \leq \epsilon$. Similarly, in the partition sampling algorithm, increasing the k parameter reduces the size of each partition. Figure 5.3.1 shows the k-means partitioning of the same dataset for different values of k. It is possible to see that as the number of partitions increases, the size of each individual one decreases.

In order to better compare the performance of partition to rejection sampling, I wanted to choose values for k which would induce k-means++ to partition the Bloch sphere into cells of comparable volumes to those occupied by the Bloch vector representations of the coarse-grained states $\Lambda(\psi_i)$ for different ϵ (recall figure 4.3.1, which shows a visualization of this). In order to do this, I used two approximations. First, I considered that it was possible to approximate the volume occupied by all $\Lambda(\psi_i)$ which satisfy $T(\Lambda(\psi_i), \rho) \leq \epsilon$ to that of a sphere of radius ϵ centered on the Bloch vector coordinates of ρ. Using a second assumption, that k-means++ would divide the Bloch sphere into partitions of approximately equal sizes, one could argue that the value of k which would induce the algorithm to create partitions of compatible sizes to those of $\{\Lambda(\psi_i) | T(\Lambda(\psi_i), \rho) \leq \epsilon\}$ would be:

$$k = \frac{V_{\text{Bloch sphere}}}{V_{\text{sphere of radius}\epsilon}} = \frac{1}{\epsilon^3}. \tag{5.3.1}$$

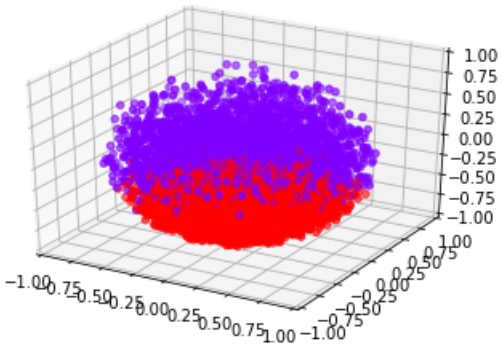
It must be noted, however, that none of these approximations are applicable in general. As discussed in the previous chapter, approximating the volume of $\{\Lambda(\psi_i) | T(\Lambda(\psi_i), \rho) \leq \epsilon\}$ to a sphere is not accurate for all possible choices of ρ and ϵ. For instance, any pure 1 qubit macrostate lies on the surface of the Bloch sphere, and therefore, part of the volume of any sphere centered on ρ will contain points that are not valid quantum states. Also, in general, it is not true that the partitions created by k-means++ are of similar sizes. In fact, in cases in which the distribution of data points is not uniform in the Bloch sphere, k-means++ will tend to create larger partitions in regions where

data points are sparse and smaller ones in regions where they are densely concentrated. Nevertheless, these considerations were helpful as a guide for the choice of k values. The values for which the algorithm was run were $k \in 8, 16, 37, 125, 1000$.

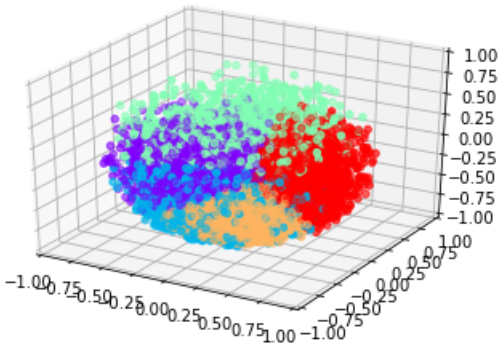
Once the algorithm had been run, I was able to construct a more accurate metric with which to compare both methods. Given a macrostate ρ belonging to a partition K_i, I calculated the maximum trace distance between all random states $\Lambda(\psi_i) \in K_i$, i.e.,

$$d_{max} = \max\{T(\Lambda(\psi_i), \rho) | \Lambda(\psi_i) \in K_n\}. \tag{5.3.2}$$

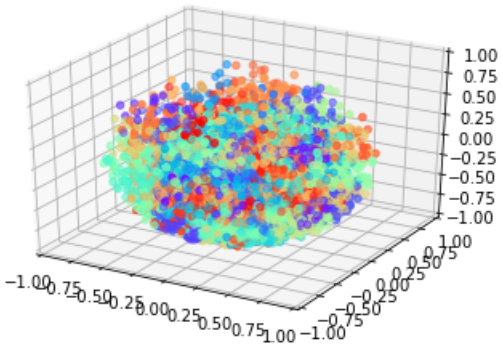
In the rejection sampling method, the set of states that approximated the set defined in equation 3.0.2 was $\widetilde{\Omega}_\Lambda(\rho) = \{\psi_i | T(\Lambda(\psi_i), \rho) \leq \epsilon\}$. From this expression, one can see that ϵ functions as an upper bound to the distance between the coarse-grained states $\{\Lambda(\psi_i) | \psi_i \in \widetilde{\Omega}_\Lambda\}$ and ρ. Furthermore, recall that the states ψ_i are uniformly sampled before the threshold conditions are applied. Taking this into account, one can see that perfoming the same procedure used to calculate d_{max} for the rejection sampling case must yield $d_{max} \approx \epsilon$. Given this, it is possible to directly compare the quality of the partition and rejection sampling algorithm by plotting the variation of the trace distance $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ with d_{max} for both of the methods.



(a) $k = 2$



(b) $k = 5$



(c) $k = 125$

Figure 5.3.1: Visualization of k-means partitioning of the feature space for different values of k. The training data shown in the graph was generated by applying the blurry detector coarse-graining map to 10000 randomly generated quantum states $\psi_i \in \mathcal{L}(\mathcal{H}_4)$ and taking the Bloch vector representations of the resulting effective states.

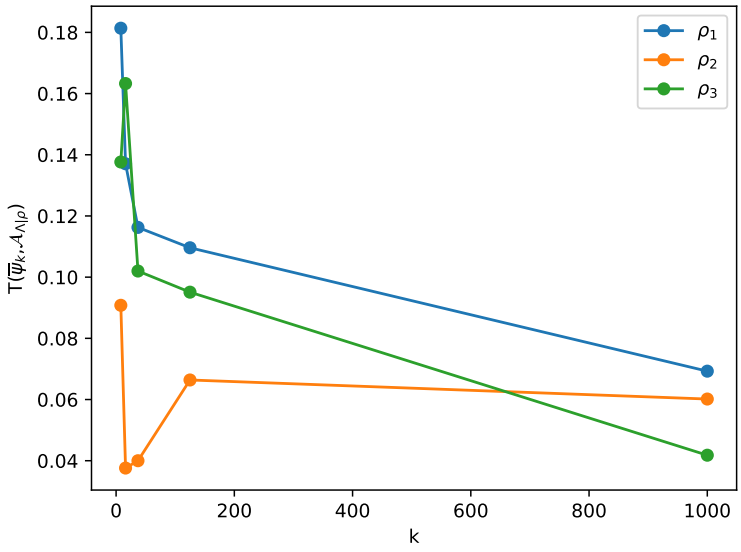
5.4 Results

5.4.1 Partial trace

Using the partial trace map, I ran the algorithm on the previously used states, $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = 0.1\,|0\rangle\langle 0| + 0.9\,|1\rangle\langle 1|$, $\rho_3 = 0.3\,|0\rangle\langle 0| + 0.7\,|1\rangle\langle 1|$ and $\rho_4 = \frac{\mathbb{1}_2}{2}$. The test was performed using a dataset with 4×10^5 random quantum states and a single initialization of k-means++, using a personal notebook, making the precision that it was possible to obtain with this method limited. Despite this, some interesting properties of the method were able to be observed from these preliminary results.

When investigating the partitions each of these states had been assigned to for each value of k, I found that states ρ_1, ρ_2 and ρ_3 had all been assigned to the same partition for all k values. Recall that, for a given macrostate ρ, the assignment $\widetilde{\mathcal{A}}_\Lambda$ is calculated by taking the average of all states ψ_i such that $\Lambda(\psi_i)$ falls into the partition $K_n \ni \rho$. This means that all of these three states were assigned the same microstate $\widetilde{\mathcal{A}}_\Lambda$ for all values of k. This exemplifies one of the drawbacks of the partition sampling model in relation to the rejection sampling one. Contrary to what we saw with the rejection sampling algorithm, the partitions K_i do not have any dependence on ρ. Instead, they are related to the geometrical distribution of the data points, which is only a property of the map Λ. Hence, this method can end up assigning the same microstate $\widetilde{\mathcal{A}}_\Lambda(\rho)$ for values of ρ whose Bloch vectors are situated in close proximity. As one can imagine, the quality, measured by the trace distance $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ was not the same for ρ_1, ρ_2 and ρ_3. A graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k for the three states can be seen in figure 5.4.1a.

The graph shows that, for $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = 0.1\,|0\rangle\langle 0| + 0.9\,|1\rangle\langle 1|$ and $\rho_3 = 0.3\,|0\rangle\langle 0| + 0.7\,|1\rangle\langle 1|$, $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ fluctuated as k increased, while for $\rho_1 = |1\rangle\langle 1|$, it decreased monotonically. The performance of the algorithm was observed to be directly related to the difference between the macrostate and the center of the cluster it was assigned to. Table 5.4.1b shows the trace distance between each macrostate and ρ_{c_i}, the quantum state whose Bloch vector coordinates would lie on center c_i of the partition $K_i \ni \rho$. Comparing the table with the graph in figure 5.4.1, one can see that the algorithm performed better for the macrostates ρ for which $T(\rho, \rho_{c_i})$ was lowest.



(a) Graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k.

$T(\rho, \rho_{c_i})$	$k = 8$	$k = 16$	$k = 37$	$k = 125$	$k = 1000$
ρ_1	0.18160353	0.13702275	0.11610937	0.10946386	0.062845
ρ_2	0.09115251	0.03739661	0.03941795	0.06602427	0.04881202
ρ_3	0.13771948	0.16323312	0.10172241	0.09385382	0.03361624

(b) Table showing the trace distance $T(\rho, \rho_{c_i})$ between each macrostate ρ and ρ_{c_i}.

Figure 5.4.1: Results of applying partition sampling to the case of the partial trace, on three different macrostates $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = 0.1\,|0\rangle\langle 0| + 0.9\,|1\rangle\langle 1|$ and $\rho_3 = 0.3\,|0\rangle\langle 0| + 0.7\,|1\rangle\langle 1|$. and The three macrostates were allocated in the same partition for all k, but the quality of the approximation $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ is different for each macrostate at each point in the graph shown in (a). The best approximations correspond to points at which $T(\rho, \rho_{c_i})$ was minimum, where ρ_{c_i} is defined as the quantum state whose Bloch vector coordinates lie on center c_i. This can be seen by comparing the graph (a) to the table shown in (b).

The largest fluctuations in the quality of the approximation as k increased were observed for $\rho_4 = \frac{\mathbb{1}_2}{2}$, whose graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k can be seen in figure 5.4.2. Furthermore, when plotting the graph d_{max} versus k for $\rho_4 = \frac{\mathbb{1}_2}{2}$, I observed a different behavior to that seen in the case of the other three macrostates. While for $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = 0.1\,|0\rangle\langle 0| + 0.9\,|1\rangle\langle 1|$ and $\rho_3 = 0.3\,|0\rangle\langle 0| + 0.7\,|1\rangle\langle 1|$, d_{max} decreased monotonically with k, this was not the case for ρ_4. These graphs can be seen in figure 5.4.3.

To better understand this behavior, I observed the partitions in which ρ_4 was placed into for each k value, plotted in figure 5.4.4. Analyzing this figure, one can see that, although the partitions decrease in size, for $k = 16$, ρ_4 fell near the center of the partition, while for $k = 37$, it fell on the edge. As a consequence, d_{max}, the maximum trace distance between ρ and all of the states in $\Lambda \circ \widetilde{\Omega}_\Lambda(\rho)$, is larger for the case of $k = 37$ than for $k = 16$. Another factor to take into account it that the partition in which ρ_3 falls in at $k = 16$ exhibits greater radial symmetry around ρ_4 than

the one it falls into at $k = 37$. It is likely that, even though both partitions include a great number of states that are relatively distant from ρ_3, when calculating the average of the set $\widetilde{\Omega}_\Lambda(\rho)$, radial symmetry leads to these errors canceling each other out. In contrast, the partition ρ_3 falls in at $k = 37$ is shifted towards the negative x-axis in relation to ρ_3. This could shift the average away from the theoretically expected value of the assignment.

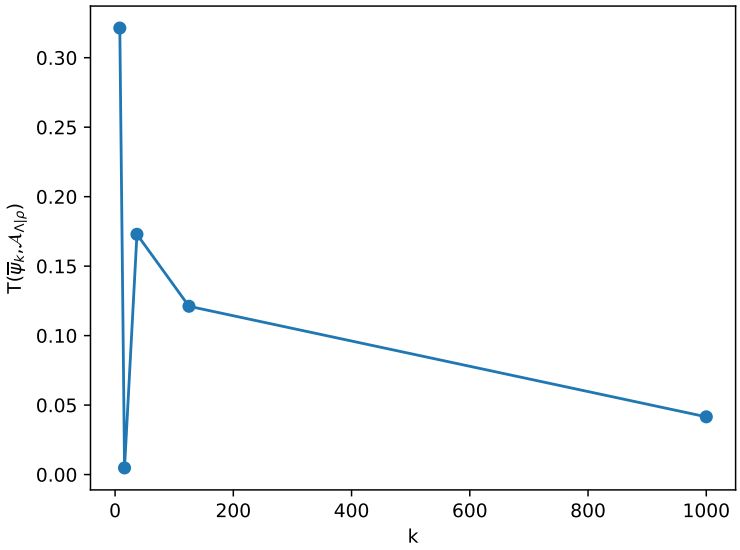
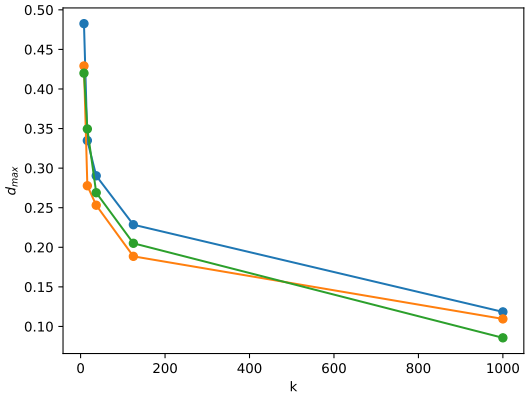
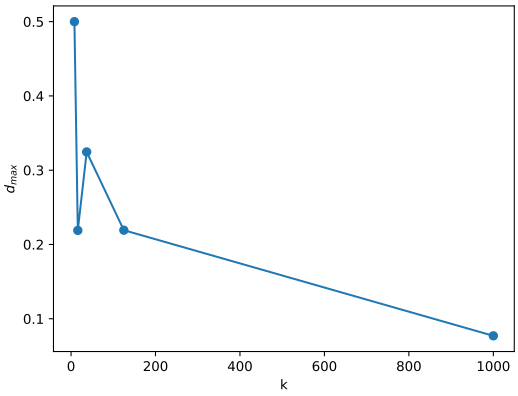


Figure 5.4.2: Graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k for $\rho_4 = \frac{\mathbb{1}_2}{2}$. We see that the error drops sharply from $k = 8$ to $k = 16$, rises again at $k = 37$. With this, it is possible to see that there are factors other than the number of partitions that affect the quality of the algorithm.

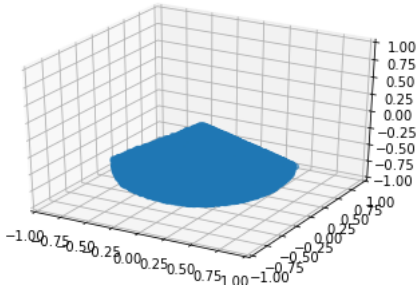


(a) $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.9 \end{bmatrix}$ and $\rho_3 = \begin{bmatrix} 0.3 & 0 \\ 0 & 0.7 \end{bmatrix}$

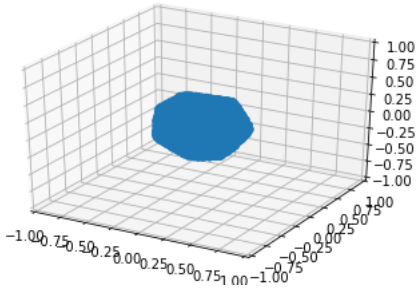


(b) $\rho_4 = \frac{\mathbb{1}_2}{2}$

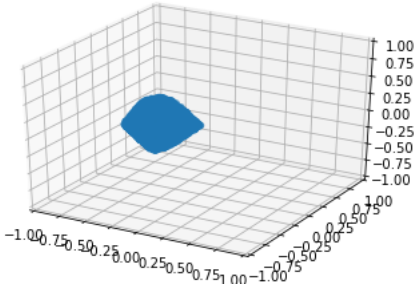
Figure 5.4.3: Graphs of d_{max} versus k for ρ_1, ρ_2 and ρ_3 and for ρ_4. It is possible to see that, unlike in the case of the first three macrostates, the value of d_{max} did not decrease monotonically with k. This suggests that even though the sizes of the partitions decrease on average as k increases, it is still possible for d_{max} to increase if the macrostate ρ is placed in a suboptimal manner.



(a) $k = 8$



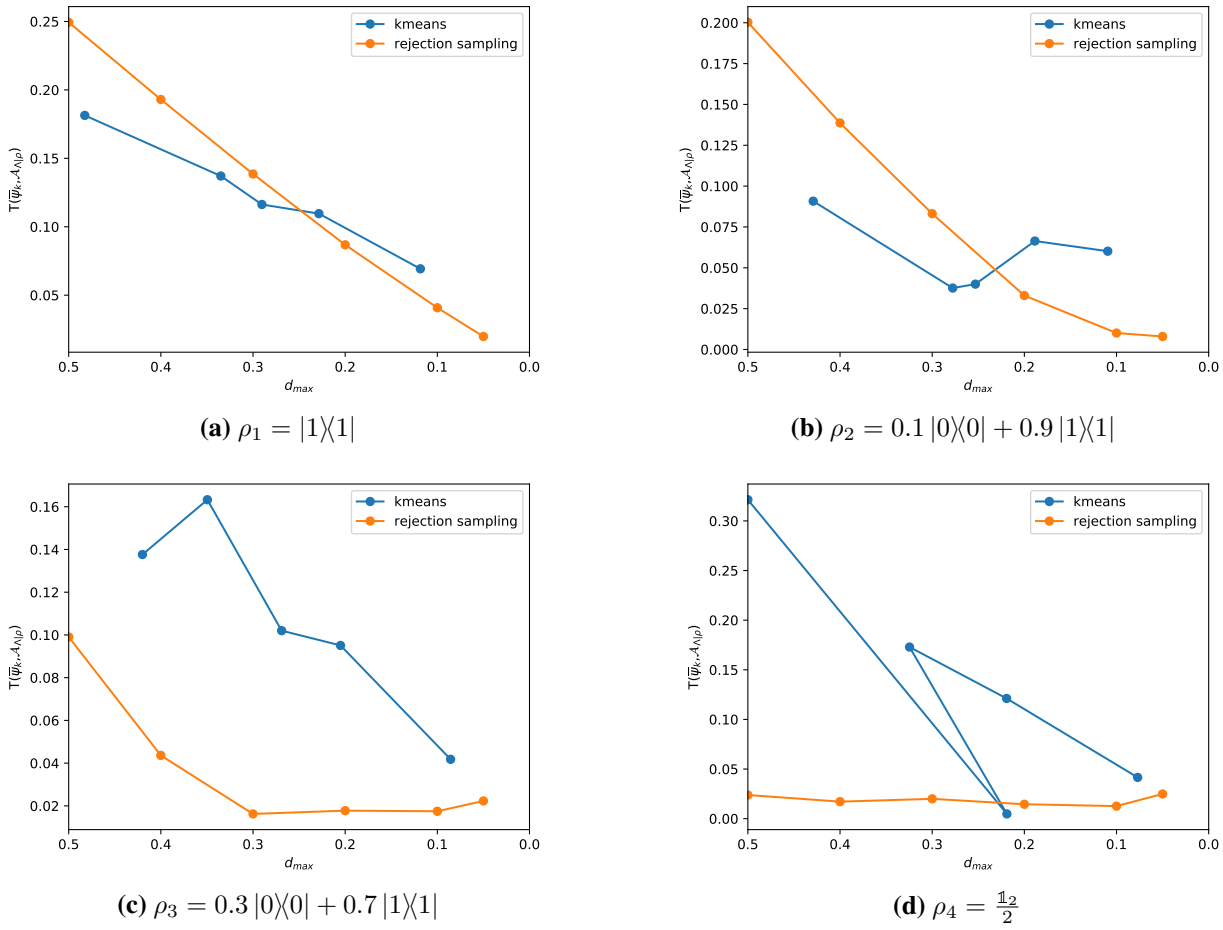
(b) $k = 16$



(c) $k = 37$

Figure 5.4.4: Partitions to which $\rho_4 = \frac{\mathbb{1}_2}{2}$ was assigned to for different k values. As k increases, the partitions become more smaller. However, some choices of partition may not be optimal and can shift $\widetilde{\mathcal{A}}_\Lambda$ away from the analytically calculated value.

Plotting $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus d_{max} for the partition and rejection sampling cases, it was found that, in general, for the partial trace coarse-graining map, the rejection sampling method gave better approximations to the analytically derived assignment $\mathcal{A}_\Lambda$. The results can be seen in figure 5.4.5. It is possible that the errors exhibited by the partition sampling method in this case could be mitigated by choosing larger values of ρ and therefore, reducing the sizes of the partitions. It would be interesting to investigate this possibility by running the algorithm on a computer cluster.



(a) $\rho_1 = |1\rangle\langle 1|$

(b) $\rho_2 = 0.1\,|0\rangle\langle 0| + 0.9\,|1\rangle\langle 1|$

(c) $\rho_3 = 0.3\,|0\rangle\langle 0| + 0.7\,|1\rangle\langle 1|$

(d) $\rho_4 = \frac{\mathbb{1}_2}{2}$

Figure 5.4.5: Graphs of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus d_{max} for the case of the partial trace coarse-graining map, given different macrostates. It was found that, although partition sampling is computationally faster, the approximations given by the rejection sampling method are generally more accurate.

5.4.2 Blurry detector coarse-graining map

For the blurry detector coarse-graining map, I ran the algorithm on the states, $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$ and plotted the graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k, which is shown in figure 5.4.6. As before, the test was performed using a dataset with 4×10^5 random quantum

states and a single initialization of k-means++. For this test, it was observed that $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ decreased monotonically as k increased.

The values of d_{max} also decreased monotonically with k, as can be seen in figure 5.4.7. In addition, it was found that, the sizes of d_{max} were generally lower for some states than for others. Note, however, that the states with lowest values for d_{max} and the states with the lowest values of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ do not overlap. This suggests that d_{max} was not the most important factor influencing the performance of the algorithm.

Unlike in the case of the previous section, the different macrostates generally fell into different partitions. Figure 5.4.8 shows the partitions for the different macrostates in the case in which $k = 1000$. It must also be noted that the only macrostate which is located on the center of its partition is $\rho_3 = \frac{\mathbb{1}_2}{2}$. This is also the macrostate for which the algorithm performed better globally. Once again, it is possible to argue that radial symmetry around ρ_3 benefitted the algorithm when calculating the average over the states $\{\psi_i | \Lambda(\psi_i) \in K_i\}$. Being pure states, all of the other states lied on the edges of their respective partitions. As was discussed previously, when the average of $\Lambda(\psi_i) \in K_n$ is shifted away from ρ, the error in the approximation increases. Hence, the algorithm does not perform as well for states on the edges of their partitions.

It is also interesting to note that, unlike what was seen with the rejection sampling method, partition sampling performed similarly for $\rho_1 = |1\rangle\langle 1|$ and $\rho_2 = |0\rangle\langle 0|$. Although both these states were placed on the edges of their partitions, it is possible to see that there is some radial symmetry about them if one takes into account only the x-y plane. For $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$, on which the algorithm performed worse globally, there is no symmetry of this sort.

The graphs comparing the rejection and partition sampling algorithms for the case of the blurry detector coarse-graining map can be seen in figure 5.4.9. One observes that the partition sampling algorithm performs better than the rejection sampling algorithm for some states, and worse for others.

For $\rho_1 = |1\rangle\langle 1|$, the k-means partition sampling algorithm performed better overall. It was possible to generate a partition K_n cointaining states ψ_i such that $T(\Lambda(\psi_i), \rho) \leq 0.05$, which lead to a value of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ smaller than obtained by the rejection sampling algorithm. Regarding $\rho_2 = |0\rangle\langle 0|$, both methods performed similarly in terms of quality. This is a great result considering that ρ_2 was the state for which the rejection sampling algorithm took the most time to give results. However, the best approximation was given by the rejection sampling method. The case of $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$ is similar. The algorithm performed better for equivalent values of d_{max}, but best approximation overall was given by the rejection sampling algorithm. It would be interesting to

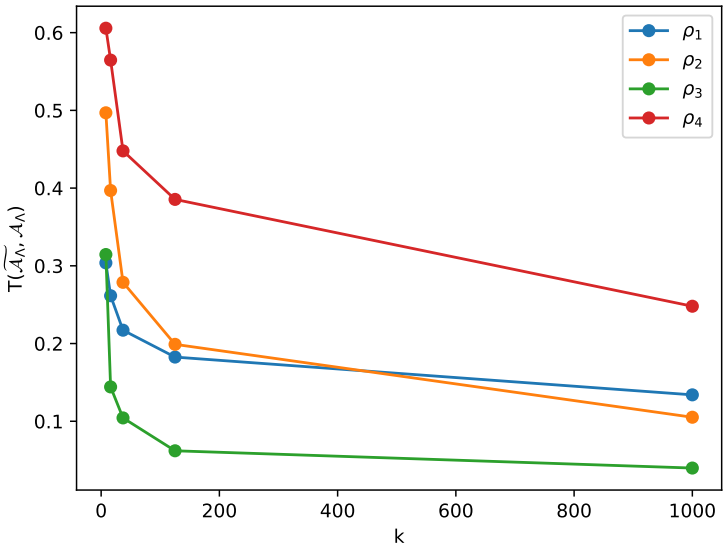


Figure 5.4.6: Graph of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus k for the case of the blurry detector coarse-graining map, given the macrostates $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$.

see if lower values of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ than what was observed for the rejection sampling method could be achieved by implementing partition sampling with larger values of k. For both ρ_2 and ρ_4, the states in the partitions for $k = 1000$ were such that $T(\Lambda(\psi_i), \rho) \geq 0.05$, and therefore, the rejection sampling algorithm performed better. Finally, in the case of $\rho_3 = \frac{\mathbb{1}_2}{2}$, k-means partition sampling performed worse than rejection sampling globally.

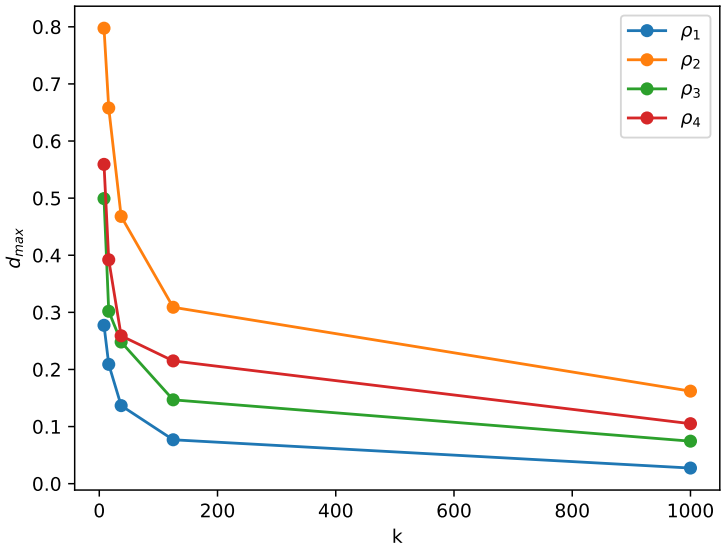
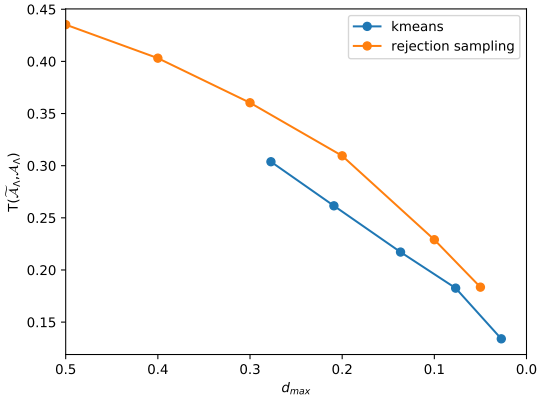
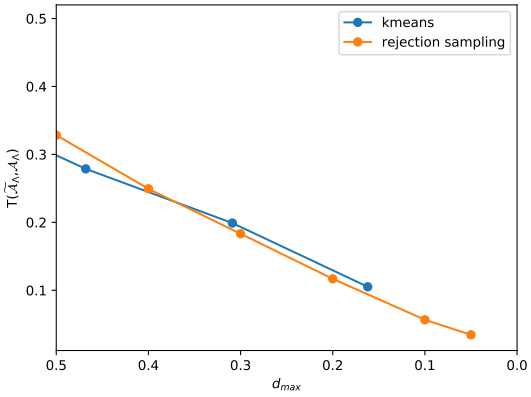


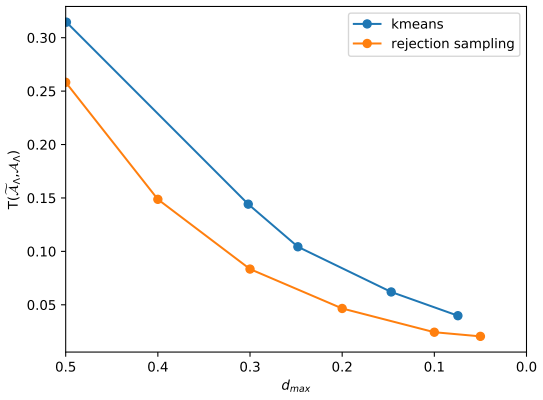
Figure 5.4.7: Graphs of $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ versus d_{max} for the case of the blurry detector coarse-graining map, given the macrostates $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$.



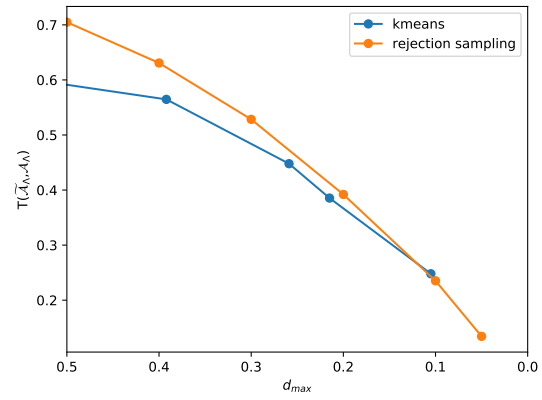
(a) $\rho_1 = |1\rangle\langle 1|$



(b) $\rho_2 = |0\rangle\langle 0|$]



(c) $\rho_3 = \frac{\mathbb{1}_2}{2}$



(d) $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$

Figure 5.4.9: Graphs of d_{max} versus $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ for states $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$ for the rejection sampling and the partition sampling method using k-means++, for the

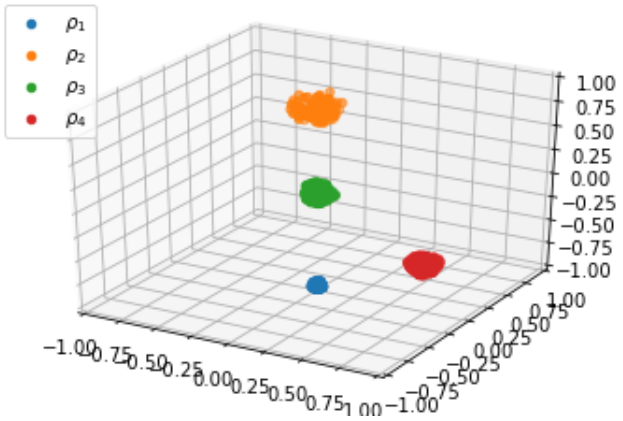


Figure 5.4.8: Partitions for which the macrostates $\rho_1 = |1\rangle\langle 1|$, $\rho_2 = |0\rangle\langle 0|$, $\rho_3 = \frac{\mathbb{1}_2}{2}$, and $\rho_4 = \begin{bmatrix} 0.2 & 0.4 \\ 0.4 & 0.8 \end{bmatrix}$ were found to belong to, for $k = 1000$. It is possible to see that the partition for the least probable macrostate, ρ_2, is larger than the partition for the most probable macrostate, ρ_1.

5.4.3 Conclusions on the partition sampling method

From these results, it is possible to conclude that the k-means algorithm has the potential of performing better and faster than rejection sampling. However, there are some problems and limitations that must be taken into consideration. First, since the k-means++ algorithm seeks to minimize the inertia ϕ 5.2.1 of the feature space, the partitions it generates are smaller in regions in which the density of datapoints is greater, in comparison to places in which they are sparse. Although this is good in the sense that it improves the quality of the approximation for the most commonly observed macrostates, the precision we can obtain for macrostates that are less likely to result from applying the coarse-graining map to a random quantum state is somewhat limited. Secondly, the quality of the approximation given by partition sampling is heavily dependent on the geometric location of the macrostate inside its partition. This limits its ability to be applied as a general method, as there will always be states which fall on the borders of a given partition. It is possible to mitigate this effect by increasing the value of k, thereby making the partitions smaller in general. It would be interesting to run this algorithm on a computer cluster in order to see if these limitations can be overcome when using large values of k.

After analyzing these results, I became interested in attempting to devise a method which would be more effective at sampling random quantum states that are mapped onto less probable macrostates. The preliminary results of this study will be shown in the next chapter.

Chapter 6

VAE-inspired sampling

6.1 Why use variational autoencoders?

A significant challenge for both methods explored in the previous chapters is that, for some physical scenarios, sets $\widetilde{\Omega}_\Lambda$ which attempt to numerically approximate $\Omega_\Lambda(\rho) = \{\psi_i \in \mathcal{L}(\mathcal{H}_B) | \Lambda(\psi_i) = \rho\}$ can have vastly different sizes for different macrostates ρ. Recall that Λ is a coarse graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \rightarrow \mathcal{L}(\mathcal{H}_d)$. When uniformly sampling from $\mathcal{L}(\mathcal{H}_D)$, states ψ_i such that, when one applies the coarse-graining map Λ result in a macrostate ρ that is unlikely to be observed compared to the other macrostates will be underrepresented in the data. In the case of rejection sampling, this makes the algorithm computationally slow for certain choices of Λ and ρ. Partition sampling can help to increase the speed of the calculations for these states, but it does not offer great improvements to the quality of the approximation.

One way to avoid this problem and create an algorithm that could more effectively approximate $\Omega_\Lambda(\rho)$ for any given ρ would be to create an algorithm that could sample microstates directly from the set itself. In order to do this, one would have to know the conditional probability distribution $\mathrm{p}(\psi|\rho)$. This probability distribution can be understood as the probability of a certain microstate ψ given that we have observed the coarse-grained macrostate to be ρ.

This is exactly the type of problem that generative machine learning models attempt to solve [32]. Given an observed variable x that is a random sample from an unknown underlying process with a probability distribution $\mathrm{p}(x)$, generative models attempt to approximate this distribution with a model $\mathrm{p}_\theta(x)$, where θ are the parameters of said model. Learning is the process by which the parameters of the model are adjusted in order for $\mathrm{p}_\theta(x)$ to approximate the true probability distribution. Generative models can include latent variables. As the name suggests, latent variables refer to variables that are a part of the model but which are not observed directly. In this work, I shall denote latent variables as h. If the joint probability distribution between both observed and latent variables given by the model is written $\mathrm{p}_\theta(x, h)$, then the marginal distribution over the

observed variables is:

$$\mathrm{p}_\theta(x) = \int \mathrm{p}_\theta(x, h)\, \mathrm{d}h. \tag{6.1.1}$$

Typically, the marginal distribution is intractible due to the integral in equation 6.1.1 not having an analytic solution or efficient estimator [32]. This intractibility is related to that of the posterior distribution $\mathrm{p}_\theta(h|x)$. This is because these distributions are related through the identity:

$$\mathrm{p}_\theta(h|x) = \frac{\mathrm{p}_\theta(x, h)}{\mathrm{p}_\theta(x)}. \tag{6.1.2}$$

Variational autoencoders, introduced in [33], deal with this problem by introducing a parametric inference model $\mathrm{q}_\phi(h|x)$ which is called the encoder. Here, ϕ denotes the parameters of the encoder model. This model attempts to approximate the posterior $\mathrm{p}_\theta(h|x)$.

As mentioned in chapter 3, the physical interpretation of generalized statistical ensembles is related to how physics is carried out in practice. Physical laws are established by carrying out experiments a large number of times. Scientists can control the macroscopic variables of the experimental setup, however, the microscopic state of the system varies. In the method proposed in [13], the way to determine the best assignment for the microscopic state of a system with a known macroscopic description is by averaging over all microscopic configurations that satisfy the macroscopic constraints. If it were possible to successfully approximate the conditional probability $\mathrm{p}(\psi|\rho)$, I could sample a large number of states from this distribution and calculate their average in order to approximate the assignment $\mathcal{A}_\Lambda$.

Given this, I decided to study the possibility of using a VAE-inspired model to attempt to compute a probability distribution $\mathrm{q}_\phi(\psi|\rho)$ which could approximate $\mathrm{p}(\psi|\rho)$. This research is still being carried out. In this work, I will present the idea of the VAE-Inspired sampling algorithm, which I plan to continue to develop during my doctoral research. Before doing so, I will present to the reader the main concepts on neural networks that they will need in order to understand this work.

6.2 Neural networks

6.2.1 Perceptrons, sigmoid neurons and neural networks

The perceptron is the simplest type of neural network, and can be thought of as a building block for more complex ones. It is composed of a single neuron, which takes n binary inputs and produces a single binary output. The inputs are usually represented in the form of a vector $\vec{x} = [x_1, x_2, \ldots, x_n]$. Each input x_i is weighted by a certain value w_i. These weights are also

represented by a vector $\vec{w} = [w_1, w_2, \ldots, w_n]$. The output is calculated by taking the weighted sum of the inputs, which can be expressed as the dot product $\vec{w} \cdot \vec{x}$, and checking if it is less than or greater than some threshold value [34], i.e.,

$$\text{output} = \begin{cases} 0 & \text{if} \quad \vec{w} \cdot \vec{x} \leq \text{threshold} \\ 1 & \text{if} \quad \vec{w} \cdot \vec{x} > \text{threshold} \end{cases} . \tag{6.2.1}$$

Machine learning textbooks usually express this condition using a term called the bias, denoted b. The value of the bias is equal to minus the value of the threshold and equation 6.2.1 is written as:

$$\text{output} = \begin{cases} 0 & \text{if} \quad \vec{w} \cdot \vec{x} + b \leq 0 \\ 1 & \text{if} \quad \vec{w} \cdot \vec{x} + b > 0 \end{cases} . \tag{6.2.2}$$

The equation can be simplified further by defining the quantity $z \equiv \vec{w} \cdot \vec{x} + b$. Determining the output of the perceptron is equivalent to applying the Heaviside step function to z. The function applied to z in order to determine the output of a neuron is called its activation function. In the case of the perceptron, the activation function is the Heaviside step function. Different types of neural networks use other activation funtions. The type of neuron most commonly used in modern neural networks, the sigmoid neuron, uses as its activation function the sigmoid function:

$$f(z) = \frac{1}{1 + e^{-z}}. \tag{6.2.3}$$

Unlike the perceptron, the sigmoid neuron's output is not binary. Instead, it outputs a real number between 0 and 1. The reason the sigmoid function is a good activation function is because it maintains the property that $f(z) \approx 1$ for $z \to \infty$, and $f(z) \approx 0$ when $z \to -\infty$. However, it is not discontinuous like the step function. This allows one to change the output by incremental amounts through slight alterations in the weights and biases. For the neural networks described in this text, we shall assume that the activation function used in each neuron is the sigmoid.

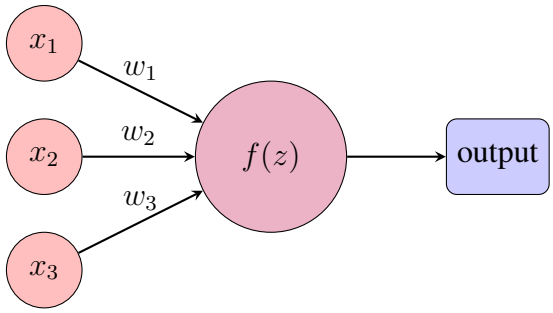
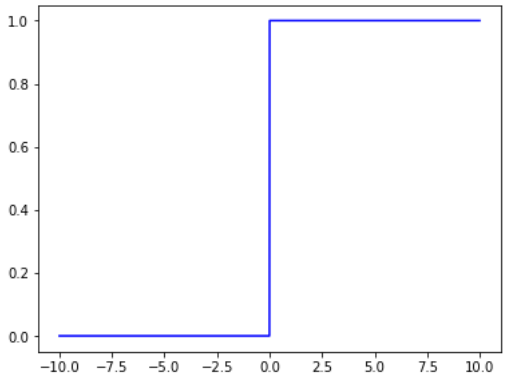
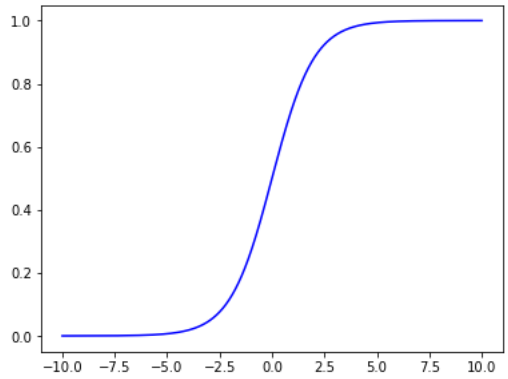


Figure 6.2.1: Graphical representation of a single neuron in a neural network. In this figure, there are three inputs, x_1, x_2 and x_3, with weights w_1,w_2 and w_3, respectively. However, this can be generalized for an arbitrary number of inputs and corresponding weights. The symbol $f(z)$ represents the application of the activation function on $z = \vec{w} \cdot \vec{x} + b$, where $\vec{w} = [w_1, w_2, w_3]$ and $\vec{x} = [x_1, x_2, x_3]$.



(a) Step function



(b) Sigmoid function

Figure 6.2.2: Graphs depicting the step function and the sigmoid function. The shape of the sigmoid function is a smoothed out version of that of the step function, which leads to it being effective as an activation function for cases in which one does not want output values to be binary.

Most neural networks are not made up of a single neuron. Neurons are organized in layers which are connected to each other. The first layer of the network is called the input layer, and it holds the input values $\vec{x}$. The last of the layers is called the output layer. As the name suggests, it returns the output of the neural network. There can be an arbitrary number of layers between these two. These are called the hidden layers. Figure 6.2.3 shows an example of a neural network and highlights the different types of layers in them.

Each of the hidden layers takes the weighted sum of the outputs of the previous layer and applies to it an activation function. The resulting outputs are called the activations and denoted a. The activations are refered to with two indices. The activation a^l_j refers to the activation of the j-th neuron in the l-th layer. These serve as inputs for the next layer. Each activation can be written as follows:

$$a^l_j = f\left(\sum_k w^l_{kj} a^{l-1}_k + b^l_j\right). \tag{6.2.4}$$

In this expression, w^l_{kj} denotes the weight for the connection from the k-th neuron in the $(l-1)$-th layer to the j-th neuron in the l-th layer. Furthermore, b^l_j refers to the bias of the j-th neuron in the l-th layer. In order to simplify this expression, it is useful to consider a weight matrix w^l composed of the weights that connect the $(l-1)$-th layer to the l-th layer. Similarly, we can define a bias vector b^l, and an activation vector a^l, composed, respectively, of the values of b^l_j and a^l_j for a given layer l. Using this notation, it is possible to rewrite equation 6.2.4 as follows:

$$a^l = f(w^l a^{l-1} + b^l). \tag{6.2.5}$$

The quantity $z^l \equiv w^l a^{l-1} + b^l$ is called the weighted input to the neurons in layer l, and equation 6.2.5 is sometimes written as $a^l = f(z^l)$. Neural networks are interesting to use for learning tasks because, by adjusting the weights and biases of the neurons in the network, one can express a great different number of functions [34]. The process through which the algorithm learns the optimal weights and biases that will enable it to perform a certain task is called training. In the following subsection, I will present a brief explanation on how training works.

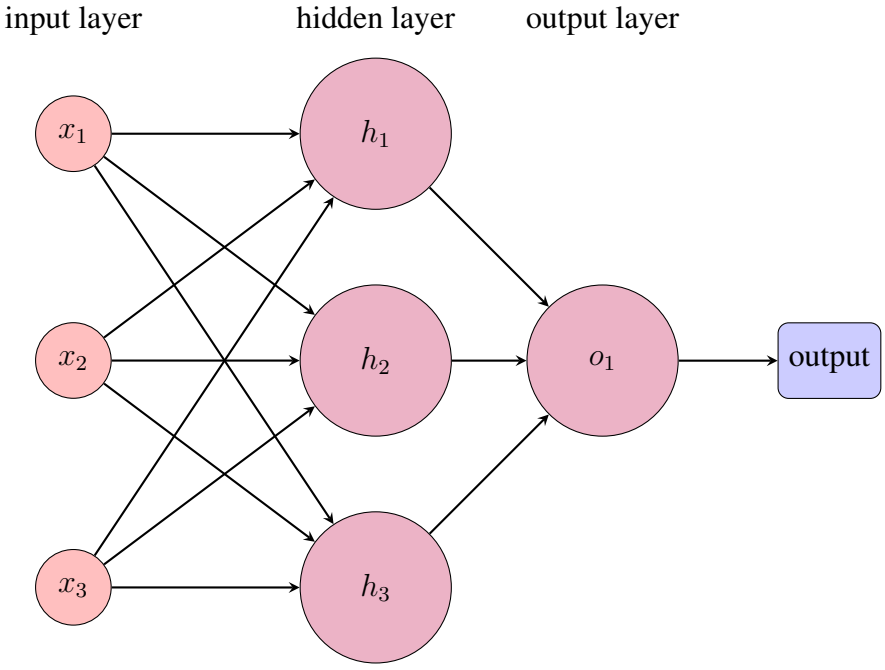


Figure 6.2.3: Graphical representation of a neural network containing one hidden layer. Here, the output layer cointains only one neuron. However, it is possible to have multiple neurons in the output layer. The outputs of the neurons in the hidden layer act as input values for the neuron in the output layer.

6.2.2 Training a neural network

In order to facilitate understanding, I will explain how the training process works in supervised learning tasks before talking about how this task is performed for the VAE. Supervised learning happens when the training data that the algorithm is given consists of a set of examples $\vec{x}$ and a set of expected outputs $\vec{y}$. The expected outputs $\vec{y}$ are usually called the labels.

Let a^L be the vector of outputs from a neural network, where L denotes the last layer. A possible function that quantifies how close a^L is to the desired output is:

$$C(w, b) \equiv \frac{1}{2n} \sum_x |y(x) - a^L|^2, \tag{6.2.6}$$

where w is the collection of all of the weights, b is the collection of all biases and n is the total num-

ber of training inputs. Note that $\vec{a}$ depends on w and b, although this dependence is not explicitly shown. The function C is called the quadratic cost function or the mean squared error (MSE). From equation 6.2.6 it is possible to see that $C \to 0$ as $\vec{y}(x) \to \vec{a}$. Hence, the objective of the training algorithm is to minimize the cost function as a function of the weights and biases. The idea of this method is that, once these optimal values of w and b are found, one can input unlabeled data into the neural network and obtain reliable predictions. In order reduce the number of variables used in some equations, I will consider that each of the layers of the neural network has an extra neuron that is permanently set to 1. The biases b can be interpreted as the weights of those extra neurons. Therefore, during the remainder of the section I will refer to the parameters of the neural network as the weights.

The algorithm used to minimize the cost function is called gradient descent. It is based on changing the weights in incremental amounts for every element of the training data that the algorithm processes. To find the direction of this change, one calculates the gradient of the cost function with respect to its weights, for each time the algorithm is run. We know from calculus that the gradient will point towards the direction of steepest ascent. Therefore, the gradient descent mechanism adjusts the weights in the direction of $-\nabla C(w)$.

The change applied to w for each iteration of the algorithm can be written as:

$$\Delta w = -\eta \cdot \nabla C(w). \tag{6.2.7}$$

The parameter η is called the learning rate, and is always positive. It determines the size of the incremental change applied to the parameters for each iteration of the algorithm. In other words, for each training example, the weights are updated in the following manner:

$$w^l_{kj} \to w^l_{kj} - \eta \frac{\partial C(\nu)}{\partial w^l_{kj}}. \tag{6.2.8}$$

When there is a very large number of training examples, however, gradient descent can be computationally slow. Therefore, usually what is used is a variation of the method called stochastic gradient descent. Stochastic gradient descent estimates the value of $\nabla C(w)$ by computing it not over all of the training data, but over a small number of randomly chosen training inputs, called a batch. By averaging over the batch, one can acquire a good estimate for $\nabla C(w)$.

The partial derivative indicated in equation 6.2.8 can be calculated analytically using the chain rule. Let us denote the last layer of the neural network by the letter L, so that the output of the j-th neuron of the last layer is denoted a^L_j. The cost function for a single training example can be written as:

$$C(w) = \frac{1}{2}\left(y - a^L_j\right)^2, \tag{6.2.9}$$

where $a_j^L = f(z_j^L) = f\left(\sum_k w_{kj}^L a_k^{L-1} + b_j^L\right)$. The derivative of $C(w)$ with respect to w_{kj}^L is

$$\frac{\partial C}{\partial w_{kj}^L} = \left(y - a_j^L\right) f'\left(z_j^L\right) \frac{\partial z_j^L}{\partial w_{kj}^L}. \tag{6.2.10}$$

Evaluating the derivative of z_j^L with respect to w_{kj}^L, we obtain:

$$\frac{\partial z_j^L}{\partial w_{kj}^L} = \sum_k \frac{\partial z_j^L}{\partial a_k^{L-1}} \frac{\partial a_k^{L-1}}{\partial w_{kj}^L} = \sum_k w_{jk}^L f'\left(z_k^{L-1}\right) \frac{\partial z_k^{L-1}}{\partial w_{kj}^L}. \tag{6.2.11}$$

From this, we see that there is a recursion relation between the derivatives and it is possible to keep applying the chain rule until we reach the first layer of the network. Because of this property, the algorithm for calculating the partial derivatives of the cost function is called backpropagation.

A schematic representation of the gradient descent method can be seen in figure 6.2.4. The figure shows one of the limitations of the method, which is that, depending in the initial values of the parameters of the neural network and on the choice of learning rate, it is possible for the algorithm to become "stuck" in a local minima. This is a potential source of problems for training the neural network.

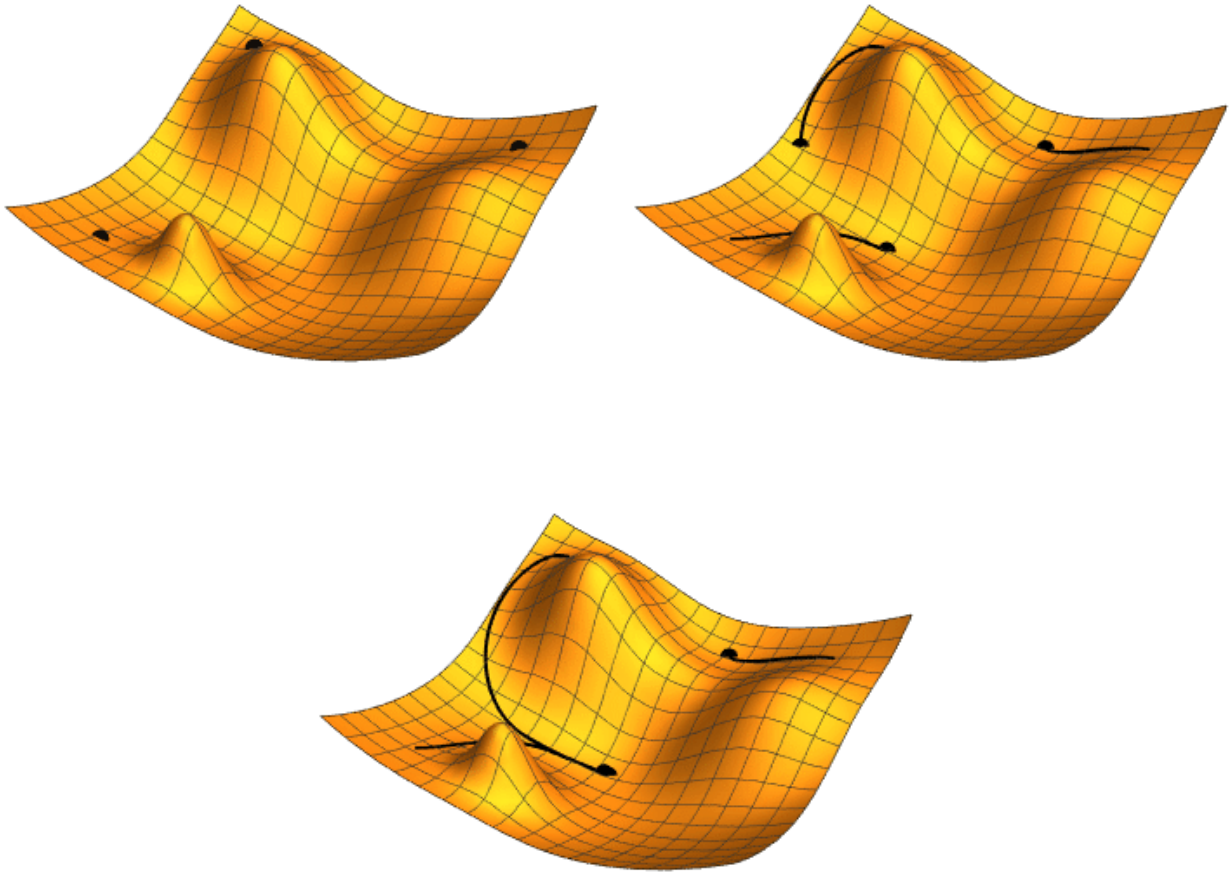


Figure 6.2.4: Figure illustrating gradient descent with 3 different initial conditions. Note that, depending on the initial conditions and the size of the training step used, it is possible for the algorithm to become "stuck" in a local minima. When this happens, the cost function does not reach its absolute minimum. All images shown here were created by Jacopo Bertolotti and distributed under a CC0 1.0 license.

6.3 Variational autoencoders

As mentioned in section 6.1, variational autoencoders attempt to learn the probability of some latent variable $\vec{h}$ given some observed variable $\vec{x}$, denoted $\mathrm{p}_\theta(h|x)$, where θ are the parameters of the model. Note that, a priori, we do not know the values of the latent variables $\vec{h}$. Therefore, it is not possible to generate a set of labels in order to train the algorithm in a supervised way, as described in the previous section. Instead, the neural network needs to be trained in an unsupervised manner. In order to understand how VAEs manage to do this, we must first talk about the structure of the VAE and the role of each of its components.

Structurally, VAEs are composed of a pair of connected neural networks — the encoder and the

decoder. The encoder is a neural network that takes the input data and generates a vector with a set of means, $\vec{\mu}$, and a set of standard deviations, $\vec{\sigma}$, for each of the latent variables. This is usually called the latent representation of the data. These means and standard deviations are used to sample the elements of h, which are used as the input for the decoder. The objective of the decoder is then to reconstruct $\vec{x}$ given $\vec{h}$. A schematic representation of this can be seen in figure 6.3.1.

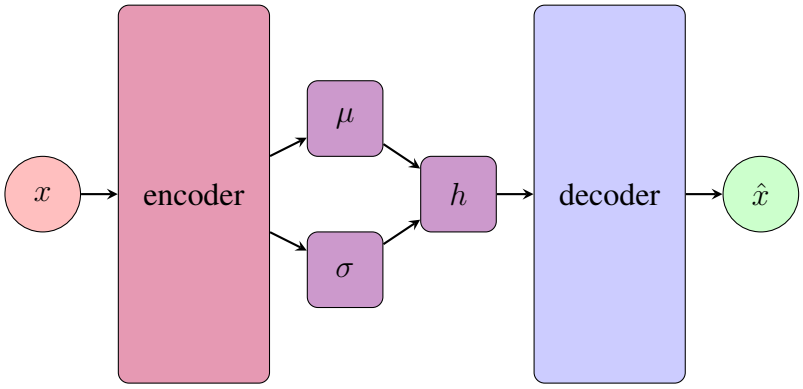


Figure 6.3.1: Visual representation of the VAE. The encoder learns the propability distribution of the latent variables $\mathrm{p}_\phi(h|x)$. This probability distribution is parametrized by a vector of variances μ and a vector of standard deviations σ.

In order to sample $\vec{h}$, the encoder computes a probability distribution $\mathrm{q}_\phi(h|x)$ which approximates $\mathrm{p}_\theta(h|x)$. The decoder computes $\mathrm{p}_\theta(x|h)$. The optimization objective of the VAE is to maximize the evidence lower bound (ELBO), defined as [32]:

$$\mathcal{L}_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(h|x)}\left[\log\left[\frac{p_\theta(x,h)}{q_\phi(h|x)}\right]\right], \tag{6.3.1}$$

where $\mathbb{E}$ denotes the expected value of the argument. The ELBO and the marginal likelihood $\mathrm{p}_\theta(x)$ are related in the following way:

$$\log \mathrm{p}_\theta(x) = \mathcal{L}_{\theta,\phi}(x) + D_{KL}(q_\phi(h|x)||p_\theta(h|x)). \tag{6.3.2}$$

A more detailed derivation of this relation can be see in [32]. The quantity $D_{KL}(q_\phi(h|x)||p_\theta(h|x))$ is the Kullback-Leibler (KL) divergence between $q_\phi(z|x)$ and $p_\theta(z|x)$, defined as:

$$D_{KL}(q_\phi(h|x)||p_\theta(h|x)) = \mathbb{E}_{q_\phi(h|x)}\left[\log\left[\frac{q_\phi(h|x)}{p_\theta(h|x)}\right]\right], \tag{6.3.3}$$

and can be understood as a measure of how much one probability distribution differs from another [35]. Looking at equation 6.3.2, one can see that maximizing the ELBO is beneficial in two ways. First, it approximately maximizes the marginal likelihood, which in turn makes the generative model better. Secondly, it minimizes the KL divergence between $q_\phi(h|x)$ and $p_\theta(h|x)$, which means

that $q_\phi(h|x)$ approximates $p_\theta(h|x)$ more accurately.

Notice that, since $\vec{h}$ is the result of stochastic sampling, and not a deterministic operation, we cannot apply backpropagation through the layers in the way shown in the previous section. In order to deal with this problem, $\vec{h}$ is reparameterized in the following manner:

$$\vec{h} = \vec{\mu} + \vec{\sigma} \odot \alpha, \tag{6.3.4}$$

where α is a random noise sample and the symbol $\odot$ denotes element-wise multiplication. This way, all of the layers in the autoencoder become deterministic and backpropagation can be applied, allowing for the network to be trained. An illustration of this trick is shown in figure 6.3.2.

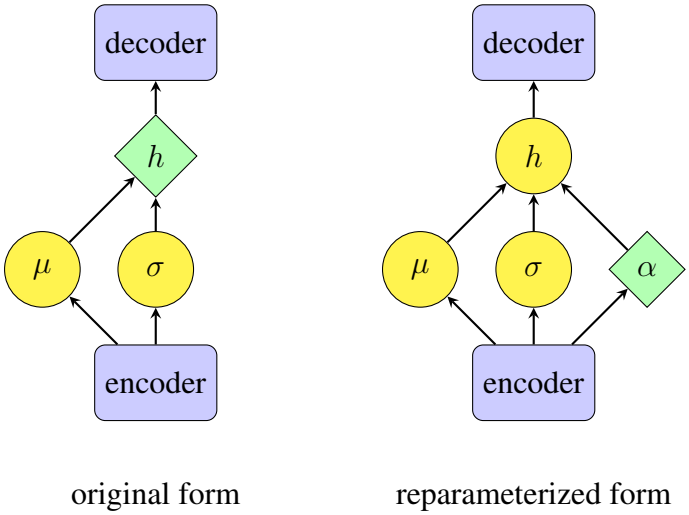


Figure 6.3.2: Illustration of the reparameterization trick. Deterministic nodes are represented by circles and stochastic nodes are represented by diamonds. By reparemeterizing h as a function of μ and σ, using the random variable α, it is possible to turn h into a "deterministic node" through which backpropagation can be applied.

6.4 VAE-Inspired sampling algorithm

As mentioned in the introduction of this chapter, I am interested in finding a way to estimate $\mathrm{p}(\psi|\rho)$, the probability of a certain microstate ψ given a coarse-grained state ρ. In the physical context explored by this work, the density matrix ρ is the observed variable, while the microstate ψ is a sample from the probability distribution $\mathrm{p}(\psi|\rho)$. Unlike in the general problem explored by the VAE, the probability distribution $\mathrm{p}(\rho|\psi)$ in this case is known exactly. We know that, given a state ψ, the macrostate ρ will be determined by application of the coarse-graining map $\Lambda : \mathcal{L}(\mathcal{H}_D) \rightarrow \mathcal{L}(\mathcal{H}_d)$. Hence, the structure proposed for the VAE-Inspired sampling algorithm is as follows. A vector $\vec{x}$ parameterizing ρ is input into a multilayered neural network like the VAE's encoder and

outputs a vector of means $\vec{\mu}$ and strandard deviations $\vec{\sigma}$ from which a vector $\vec{h}$ is sampled. The vector $\vec{h}$ is then represented as a quantum state $|\psi\rangle \in \mathcal{H}_D$. Then, the coarse-graining map Λ is applied to $\psi = |\psi\rangle\langle\psi|$ in order to reconstruct the input. A schematic representation of this can be seen in figure 6.4.1.

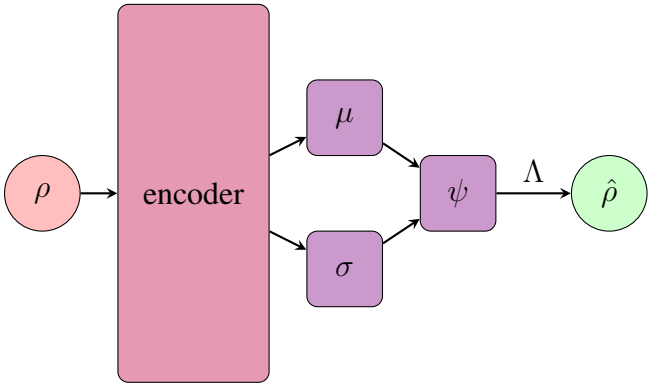


Figure 6.4.1: Schematic representation of the VAE-Inspired algorithm. The coarse-graining map Λ performs the decoder's role of reconstructing the input.

Once this model is trained, I will be able to use the encoder-style neural network in order to sample microstates ψ_i that approximate the set $\Omega_\Lambda(\rho)$ defined in equation 3.0.2. The approximate assignment $\widetilde{\mathcal{A}}_\Lambda$ can then be calculated by taking the average over $\widetilde{\Omega}_\Lambda(\rho)$.

In order for the algorithm to work, it is essential that the output of the encoder, $\vec{h}$, be a valid representation of a quantum state. The vector $\vec{h}$ is composed of real numbers, and its dimension, denoted L is usually called the latent dimension in the context of machine learning. It is possible to represent a pure $|\psi\rangle \in \mathcal{H}_D$ as a vector of this sort by interpreting each of the components of $\vec{h}$ as an angle parameterizing a sphere of dimension $2^{N+1} - 1$. The reason for this will be seen as follows.

In terms of a general basis $|j_1, j_2, ..., j_N\rangle$, an N q-bit quantum state can be written as:

$$|\psi\rangle = \sum_{j_1,j_2,...,j_N=0}^{1} C_{j_1,j_2,...,j_N} |j_1, j_2, ..., j_N\rangle , \tag{6.4.1}$$

where:

$$\sum_{j_1,j_2,...,j_N=0}^{1} |C_{j_1,j_2,...,j_N}|^2 = 1. \tag{6.4.2}$$

The number of coefficients $C_{j_1,j_2,...,j_N}$ that describe an N q-bit quantum state is 2^N. Each coefficient is a complex number that can be written as:

$$C_{j_1,j_2,...,j_N} = a_{j_1,j_2,...,j_N} + ib_{j_1,j_2,...,j_N}. \tag{6.4.3}$$

And the normalization condition 6.4.2 can be rewritten as:

$$\sum_{j_1,j_2,\ldots,j_N=0}^{1} |a_{j_1,j_2,\ldots,j_N}|^2 + |b_{j_1,j_2,\ldots,j_N}|^2 = 1 \tag{6.4.4}$$

Given this, we can see that any N qubit quantum state can be specified using 2^{N+1} real numbers that satisfy equation 6.4.4. It is possible to interpret this set of 2^{N+1} real numbers as a set of coordinates that determine a point on a sphere of dimension $2^{N+1} - 1$ and radius 1. Using spherical coordinates [36], we write:

$$\begin{aligned}
a_{0,0,\ldots,0} &= \cos\theta_1 \\
b_{0,0,\ldots,0} &= \sin\theta_1 \cos\theta_2 \\
&\vdots \\
a_{1,1,\ldots,1} &= \sin\theta_1 \ldots \sin\theta_{N-2} \cos\theta_{2^{N+1}-1} \\
b_{1,1,\ldots,1} &= \sin\theta_1 \ldots \sin\theta_{N-2} \sin\theta_{2^{N+1}-1},
\end{aligned}$$

where $\theta_1, \theta_2, \ldots, \theta_{N-2}$ range over $[0, \pi]$ radians and ϕ_{N-1} ranges over $[0, 2\pi]$ radians.

Therefore, if we choose our latent dimention so that $L = 2^{N+1}$ and set the components of $\vec{h}$ to be in the appropriate ranges for each of the angles θ_i. An obvious drawback of this is that the dimension of the latent description will increase exponentially with the number of qubits of the microscopic states. The effects of this will need to be studied in more detail in the future. However, for an initial version of the algorithm, it is possible to study microscopic descriptions with a small amount of qubits in order to avoid this issue.

So far, the greatest challenge to implementing this algorithm is finding the best way to train the model. Although the VAE-inspired algorithm has already been constructed, early tests showed that it was not possible to get the error between the input and the reconstructed output to converge. In other words, the VAE is not learning the best values for the parameters of the network. It is possible that the cost function that is being applied is not adequate given the adaptations that have been performed to the structure of the VAE. Another consideration is that the initial values of the parameters and the learning rate are set to values that lead to the gradient descent algorithm getting "stuck" in a local minima. For my doctoral research, I plan to continue to study neural networks in order to diagnose this problem.

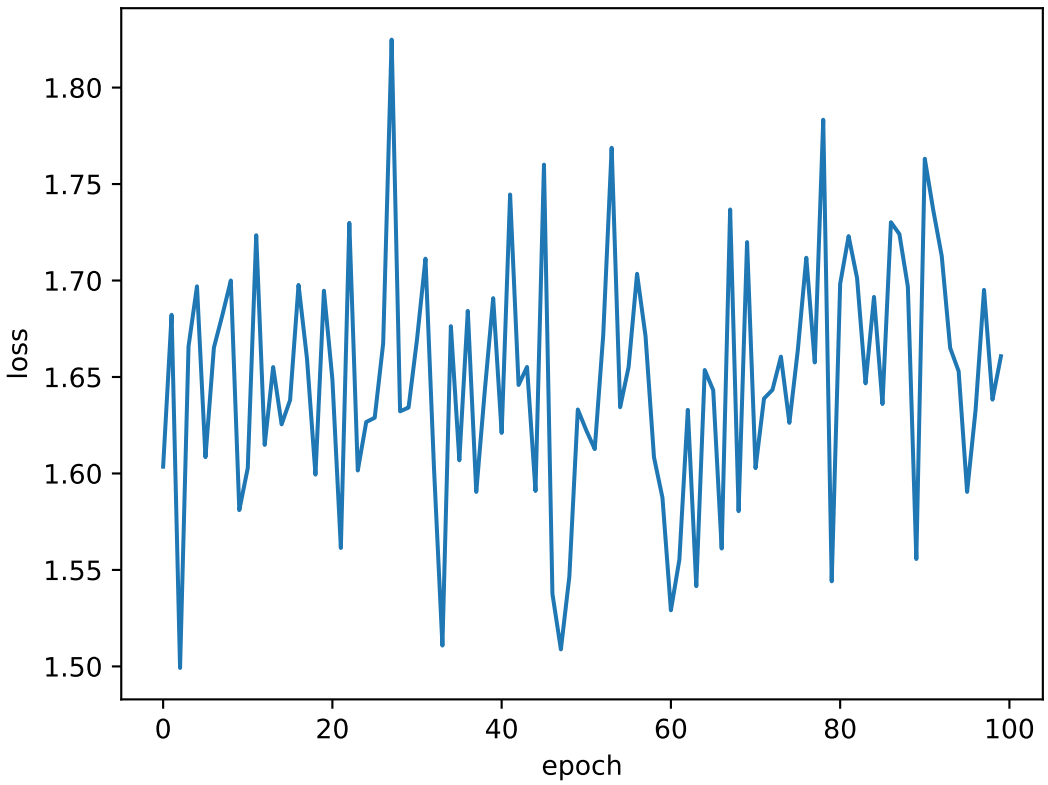


Figure 6.4.2: Graph showing the reconstruction loss — the error between the input and the reconstructed output, as defined by equation 6.2.6 — for each epoch of training. The reconstruction loss does not converge and instead fluctuates as the training is performed, which means that the program is not learning how to reconstruct the data.

Chapter 7

Conclusion

One of the greatest challenges of physics is developing a framework that can describe natural phenomena on all scales. In statistical mechanics, the connection between the macroscopic laws of thermodynamics and the microscopic laws of classical or quantum physics are connected through the concept of statistical ensembles. A statistical ensemble is an infinitely large set of microscopic systems which share the same macroscopic properties, such as, for example, a fixed energy or temperature. This reflects the reality of the how physics is developed in practice, with experiments being performed many times by researchers in order to establish a physical law. Experimentalists may be able to control the macroscopic variables in their experimental setups precisely, but due to uncertainty, the microscopic description of equivalent experimental preparations is bound to vary. Recent results in the field of quantum information theory have provided new perspectives on how to use this concept to assign a quantum microstate to a system given a coarse-grained description. One considers a generalized statistical ensemble, where what determines if the states satisfy the macroscopic conditions is a coarse-graining map [11, 12]. The best assignment for the system is defined as the average over all microstates in the set $\Omega_\Lambda(\rho) = \{\psi \in \mathcal{L}(\mathcal{H}_D)|\Lambda(\psi) = \rho\}$[13]. Using these results, this work sought out to explore the possibility of applying machine learning techniques to numerically approximate the best assignment for the microscopic state of a system given partial information.

In order to form a basis for comparision, I first developed a program that did not use machine learning methods, the rejection sampling algorithm. Given a macrostate $\rho \in \mathcal{L}(\mathcal{H}_d)$, this program uniformly samples microstates $\psi_i \in \mathcal{L}(\mathcal{H}_\mathcal{D})$, applies to them a coarse-graining map Λ and checks if the trace distance $T(\Lambda(\psi_i), \rho)$ is bellow a certain threshold value ϵ. After the resulting dataset $\widetilde{\Omega}_\Lambda$ achieves a sufficiently large size, the program calculates the average over the set, which is the approximate assignment $\widetilde{\mathcal{A}}_\Lambda$. In order to gain a more quantitative understanding of the performance of the algorithm, I applied it to coarse-graining maps for which the analytical result for the assignment $\mathcal{A}_\Lambda$ is known, such as the partial trace and the blurry detector coarse-graining map

[11, 12, 13]. This allowed me to use the trace distance $T(\widetilde{\mathcal{A}}_\Lambda, \mathcal{A}_\Lambda)$ as a metric of error.

The results of the rejection sampling algorithm allowed me to gain intuition about the maps I was working with, as well as some of the properties of the systems they describe. In order to be able to visualize the states $\Lambda(\psi_i)$ in 3D space, I decided to work with macrostates $\rho \in \mathcal{L}(\mathcal{H}_2)$. In the case of the partial trace mapping from $\mathcal{L}(\mathcal{H}_4) \to \mathcal{L}(\mathcal{H}_2)$, we saw that the performance of the algorithm was related to the degree of purity of the macrostate ρ. This originates from the fact that applying the partial trace to a two qubit state will only yield a pure state if there is no entanglement, i.e., the system is in a product state. States with higher purity needed to be submitted to stricter conditions in order to achieve measures of error of the same order as the maximally mixed state. This corroborates results such as [27], which show that entangled states are more probable to be obtained through random sampling in $\mathcal{L}(\mathcal{H}_4)$ in relation to non-entangled ones. Another factor that lead the program to perform better on mixed states, for both the partial trace and the blurry detector map, was related to the geometry of the sets $\Lambda \circ \widetilde{\Omega}_\Lambda$, where symbol $\circ$ denotes the element-wise application of Λ on $\widetilde{\Omega}_\Lambda$. For mixed states, as $\epsilon \to 0$, the shape of the dataset containing the states that satisfied the threshold conditions approximates that of a ball of radius ϵ. This radial symmetry allowed for the errors of the method to cancel each other out in part, resulting in a better performance. Pure states, which are situated on the surface of the Bloch sphere, do not benefit from this property. Comparing the performance of the algorithm on macrostates $|0\rangle\langle 0|$ and $|1\rangle\langle 1|$, it was found that, despite the program requiring a large amount of time to calculate $\widetilde{\mathcal{A}}_\Lambda(|0\rangle\langle 0|)$, the error was generally smaller than for $\widetilde{\mathcal{A}}_\Lambda(|1\rangle\langle 1|)$.

The partition sampling algorithm aimed to reduce the amount of time necessary to calculate $\widetilde{\mathcal{A}}_\Lambda$. The method consists of creating a large dataset of uniformly sampled microstates $D = \{\psi_i \in \mathcal{L}(\mathcal{H}_\mathcal{D})\}$, applying a coarse-graining map to each one, and creating a dataset $\Lambda \circ D$. Next, the unsupervised machine learning algorithm k-means++ divides the states in $\Lambda \circ D$ into clusters. Each cluster has a center associated to it, and a point is said to belong to a given cluster if it is closer to the center of that cluster than to any of the other centers. This process divides the space $\mathcal{L}(\mathcal{H}_d)$ into a k number of partitions. Given that the macrostate ρ belongs to a partition K_n, the set $\widetilde{\Omega}_\Lambda$ is defined as the set of states ψ such that $\Lambda(\psi) \in K_n$. The approximate assignment $\widetilde{\mathcal{A}}_\Lambda(\rho)$, as before, is the average over $\widetilde{\Omega}_\Lambda$. The reason the k-means++ method was chosen for partition sampling is because its objective is to minimize the inertia of the data points. One of the consequences of this is that, when the states in $\Lambda \circ D$ are not uniformly distributed in $\mathcal{L}(\mathcal{H}_d)$, k-means++ will tend to create smaller partitions where the data concentration is higher, and larger partitions where the data is sparse. This makes the predictions of the algorithm more precise for macrostates located in densely populated regions of $\Lambda \circ D$. This is physically interesting because those regions correspond to the macrostates which are more likely to be observed in any given experiment. Hence, partitioning using k-means++ allows for a greater precision in calculating

$\widetilde{\mathcal{A}}_\Lambda(\rho)$ for the most commonly observed macrostates.

In order to compare partition sampling with the rejection sampling approach, I applied it to the same coarse-graining maps and macrostates used in the previous case. For the partial trace, partition sampling did not perform as well as rejection sampling on the states that were analyzed. It is possible that this is because most of the macrostates chosen were situated in close proximity to one another on the Bloch sphere and, therefore, fell into the same partition and were assigned the same average microstate. It is likely that the method has to be applied with a larger number of partitions in order to be able to discriminate between macrostates that are closer to one another. This would require more powerful computational resources than I currently have access to, as, although partition sampling is a faster method than rejection sampling, it requires more memory. For the blurry detector map, partition sampling performed better for three out of four of the macrostates investigated. The error associated to $\widetilde{\mathcal{A}}_\Lambda(|0\rangle\langle 0|)$ for both methods was similar. Unlike in the rejection sampling case, the error for $\widetilde{\mathcal{A}}_\Lambda(|1\rangle\langle 1|)$ was not larger than for $\widetilde{\mathcal{A}}_\Lambda(|0\rangle\langle 0|)$. Instead, the quality of the approximation was similar for both of these macrostates.

The last algorithm that was proposed in this work was the VAE-Inspired sampling algorithm. Variational autoencoders are neural network based machine learning algorithms that aim to approximate the probability distribution of latent variables given some observed data. The idea of studying variational autoencoders came from the fact that VAEs are being used today to learn the probability distributions of quantum states [14]. Hence, it could be possible to use a similar approach to learn the conditional probability distribution $\mathrm{p}(\psi|\rho)$ of a microstate ψ given a macrostate ρ. Then, for any given ρ, we could sample microstates from $\mathrm{p}(\psi|\rho)$ to create a dataset $\widetilde{\Omega}_\Lambda$. The numerically calculated assignment $\widetilde{\mathcal{A}}_\Lambda$ would, once again, be the average of $\widetilde{\Omega}_\Lambda$.

The VAE-Inspired algorithm is structured in the following manner: a multilayered neural network called the encoder takes ρ as an input and generates a probability distribution from which we sample a microstate ψ. The network is trained so that the difference between $\Lambda(\psi)$ and ρ is minimized. Unfortunately, during my early tests of this algorithm, I was unable to minimize the difference between $\Lambda(\psi)$ and ρ. During my doctoral research, I plan to study VAEs in more detail in order to diagnose the cause of this problem.

In conclusion, there is still much to be explored on the subject of applying machine learning algorithms to physics. In this work, I attempted to explore some of these applications in order to gain knowledge of these new techniques that are being developed. For my Ph.D. project I intend to study not only variational autoencoders, but other modern machine learning methods, such as generative adversarial networks (GANs). It is possible that these techniques could be applied not only to the coarse-graining problem to this work, but to inverse problems in general.

Bibliography

[1] John Ziman. *Reliable Knowledge: An exploration of the Grounds for Belief in Science*. Cambridge University Press, 1978.

[2] Josiah Willard Gibbs. *Elementary Principles in Statistical Mechanics: Developed with Especial Reference to the Rational Foundation of Thermodynamics*. Cambridge Library Collection - Mathematics. Cambridge University Press, 2010.

[3] Mehran Kardar. *Statistical Physics of Particles*. Cambridge University Press, 2007.

[4] Daniel Lidar. Lecture notes on the theory of open quantum systems, 02 2019.

[5] Ángel Rivas, Susana F Huelga, and Martin B Plenio. Quantum non-markovianity: characterization, quantification and detection. *Reports on Progress in Physics*, 77(9):094001, Aug 2014.

[6] Heinz-Peter Breuer, Elsi-Mari Laine, Jyrki Piilo, and Bassano Vacchini. Colloquium: Non-markovian dynamics in open quantum systems. *Rev. Mod. Phys.*, 88:021002, Apr 2016.

[7] Inés de Vega and Daniel Alonso. Dynamics of non-markovian open quantum systems. *Rev. Mod. Phys.*, 89:015001, Jan 2017.

[8] Fagner M. Paula, Paola C. Obando, and Marcelo S. Sarandy. Non-markovianity through multipartite correlation measures. *Phys. Rev. A*, 93:042337, Apr 2016.

[9] Felix A. Pollock, César Rodríguez-Rosario, Thomas Frauenheim, Mauro Paternostro, and Kavan Modi. Non-markovian quantum processes: Complete framework and efficient characterization. *Phys. Rev. A*, 97:012127, Jan 2018.

[10] Nadja K. Bernardes, Álvaro Cuevas, Adeline Orieux, Carlos Monken, Paolo Mataloni, Fabio Sciarrino, and Marcelo Santos. Experimental observation of weak non-markovianity. *Scientific Reports*, 5, 04 2015.

[11] Cristhiano Duarte, Gabriel Dias Carvalho, Nadja K. Bernardes, and Fernando de Melo. Emerging dynamics arising from coarse-grained quantum systems. *Phys. Rev. A*, 96:032113, Sep 2017.

[12] Pedro Silva Correia and Fernando de Melo. Spin-entanglement wave in a coarse-grained optical lattice. *Phys. Rev. A*, 100:022334, Aug 2019.

[13] Pedro Silva Correia, Paola Concha Obando, Raúl O. Vallejos, and Fernando de Melo. Macro-to-micro quantum mapping and the emergence of nonlinearity. *Physical Review A*, 103(5), May 2021.

[14] Andrea Rocchetto, Edward Grant, Sergii Strelchuk, Giuseppe Carleo, and Simone Severini. Learning hard quantum distributions with variational autoencoders. *npj Quantum Information*, 4(28), 2018.

[15] Ilia A. Luchnikov, Alexander Ryzhov, Pieter-Jan Stas, Sergey N. Filippov, and Henni Ouerdane. Variational autoencoder reconstruction of complex many-body physics. *Entropy*, 21(11):1091, Nov 2019.

[16] John Watrous. *The Theory of Quantum Information*. Cambridge University Press, USA, 1st edition, 2018.

[17] Michael M. Wolf. The theory of quantum information, 2012.

[18] Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. 01 2006.

[19] Christian Gross and Immanuel Bloch. Quantum simulations with ultracold atoms in optical lattices. *Science*, 357:995–1001, 09 2017.

[20] Immanuel Bloch. Quantum coherence and entanglement with ultracold atoms in optical lattices. *Nature*, 453:1016–22, 07 2008.

[21] Takeshi Fukuhara, Sebastian Hild, Johannes Zeiher, Peter Schauß, Immanuel Bloch, Manuel Endres, and Christian Gross. Spatially resolved detection of a spin-entanglement wave in a bose-hubbard chain. *Phys. Rev. Lett.*, 115:035302, Jul 2015.

[22] Sherson J. F., Weitenberg C., Endres M., Cheneau M., Bloch I., and Kuhr S. Single-atom-resolved fluorescence imaging of an atomic mott insulator. *Nature*, 467:68–72, 2010.

[23] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 201.

[24] J.R. Johansson, P.D. Nation, and Franco Nori. Qutip 2: A python framework for the dynamics of open quantum systems. *Computer Physics Communications*, 184(4):1234 – 1240, 2013.

[25] Roger Balian. *From Microphysics to Macrophysics*. 01 2007.

[26] J. Sakurai and Jim Napolitano. *Modern Quantum Mechanics*. 09 2017.

[27] Patrick Hayden, Debbie Leung, and Andreas Winter. Aspects of generic entanglement. *Communications in Mathematical Physics*, 265, 08 2004.

[28] S. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[29] Eric W. Weisstein. "voronoi diagram." from mathworld–a wolfram web resource.

[30] David Arthur and S. Vassilvitskii. k-means++: the advantages of careful seeding. in proceedings of the eighteenth annual acm-siam symposium on discrete algorithms, new orleans, siam, pp. 1027-1035. *K-Means++: The Advantages of Careful Seeding*, pages 1027–1035, 01 2007.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[32] Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *Foundations and Trends in Machine Learning*, 12(4):307–392, 2019.

[33] Diederik Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 12 2013.

[34] Michael A. Nielsen. Neural networks and deep learning, 2018.

[35] Wikipedia contributors. Kullback-leibler divergence — Wikipedia, the free encyclopedia, 2021. [Online; accessed 2-July-2021].

[36] Wikipedia contributors. n-sphere — Wikipedia, the free encyclopedia, 2020. [Online; accessed 8-October-2020].