Master in Quantum Science and Technology

Quantum Artificial Life: simulating Lotka-Volterra equations

Final Thesis

Student: Álvaro Ruiz García
Tutor: Lucas Chibebe Céleri

September 2020

Abstract

Between the many branches of knowledge where quantum computing can play an important role, in this project we have focused on approaching population ecology (the study of the evolution of biological populations) using quantum algorithms.

For this, we have related two concepts: on one hand, we studied about the Lotka-Volterra equations, the most important predator-prey model. On the other hand, we proposed a Quantum Artificial Life model. Such model is strongly based on a map that we can build between an alive individual and a qubit. Individuals are represented by qubits, which interact between them and the environment.

We focused on trying to find out if this model follows the Lotka-Volterra equations. Here, the small amount of available qubits has been a major limitation. This restriction motivated us to build a classical algorithm that simulates the behaviour of every mechanism (interactions, self-replications, etc.) of the quantum model without using a quantum computer. This classical algorithm can perform simulations including large number of individuals. Then, it allowed us to state that our quantum model follows the results of Lotka-Volterra equations in the correct limit.

With the support of this classical simulation, we built the associated quantum algorithm. Besides the limited amount of qubits, the need of ancillary qubits and knowing the state of the system at all steps also constituted important issues for the quantum simulation. Despite of this, we achieved to perform a relatively small simulation on the *IBM Qasm* simulator with debatable results.

Proposed improvements on our algorithm will be mainly based on avoiding to measure and destroy the quantum system at each step of the simulation.

Contents

Chapter 1

Introduction

Since the invention of the first computers machines, computer hardware has grown in power at an amazing pace. *Intel* co-founder Gordon Moore described in 1965 [1] this growth in what has become known as Moore's law, which states that computer power will double every two years.

However, Moore's law may be approaching its limit due to one main reason: the growth of computational power is related to the skill of building smaller microchips. But these cannot reduce indefinitely their size due to the fact that quantum effects, such as tunneling, appear in the scale of nanometers, preventing the microchip from working properly [2, 3]. So the union between quantum physics and computer science appeared to be imminent.

Fundamental ideas of quantum computation rose during the 1980s. Richard Feynmann, in his famous lecture in 1981 [4] highlighted the limitations of simulating nature with classical computers. According to Feynmann, we were not simulating quantum systems but making some kind of classical and numerical approximation. In fact, the problem of simulating the state of a system consisting of several particles becomes intractable when we have to classically track the probabilities of all the particles. But quantum systems, like qubits, can track these probabilities because it is in their very nature to do so [5].

Yet quantum computation soon started to enlarge its frontiers beyond the simulation of quantum systems. Three main quantum algorithms were proposed during the 1990s, proving that quantum computers could perform several tasks more efficiently than classical computers.

The first algorithm is the Deutsch algorithm (1992), which can characterize

a unknown binary function $f : \{0, 1\} \to \{0, 1\}$ only with 2 measurements while a classical computer needs 4 [6]. This algorithm was also the inspiration for the well-known Shor's algorithm proposed in 1995 by Peter Shor [7]. Shor's algorithm can perform more efficiently than any known classical algorithm the factorization of large numbers into prime factors. The security of the largely used RSA cryptosystem relies on the hardness of this computational task. The third important quantum algorithm proposed was the Grover data search algorithm (1996) [8] with a huge range of potential applications [2].

In the last years, the research on the potential applications of quantum computer has increased exponentially. An uncountable number of potential applications is being tested at different stages, including cybersecurity [9], financial modeling [10], Artificial Intelligence [11], materials science [12] and optimization problems [13] among many more [14].

Neuroscience is one of the most active research lines in biology including quantum computation in its progress [15]. The idea that brain's functioning can be explained within the framework of quantum mechanics [16,17] lead to many quantum computing applications in neuroscience such as genetics and sequence analysis [18] or genomics [19].

Quantum computing can also be helpful in many optimization problems in biology. It is worth mentioning how the ground-state energy of a protein is obtained using adiabatic quantum computation in [20].

We can say that the present thesis approaches a different area of biology. What we did is to apply quantum computation to computational ecology. Thus, we aim to observe the evolution of populations given a quantum-based model of life. Also, we aim to model this evolution through the well-known Lotka-Volterra equations.

The thesis is structured in 4 essential chapter. First, in chapter 2 we will introduce Lotka-Volterra equations, the most important model of prey-predator systems. Here we will find out about the origin and meaning of these equations, next to a mathematical analysis and the behaviour of its solutions.

Once we have a deeper understanding of Lotka-Volterra model, we will proceed to explain about Quantum Artificial Life in chapter 3. Here we will discover the idea and motivation beyond this topic. Also, we will briefly describe which are the current models of Quantum Artificial Life and their ben-

efits and limitations. We introduce some modifications to this model in order to improve it.

Most important, we will report the state-of-the-art of this topic in order to have a deeper view on the objectives of this thesis. Our goal can be stated as exploring whether if our Quantum Artificial model follows the characteristics of a Lotka-Volterra prey-predator system as the number of individuals increase.

Due to our limitation on the number of qubits, and consequently on the number of individuals, in order to support our ideas we will first make a classical approach detailed in chapter 4. Here we will explain how this classical program behaves identically to a quantum program, avoiding limitations on the number of individuals and thus achieving results that prove how our quantum model has the characteristics of a Lotka-Volterra system.

Then, we will explain how to perform an implementation of the model for quantum computers at chapter 5. Due to the limited number of qubits of public access quantum computers our executions are constrained to a single simulator: the IBM *Qasm Simulator*. Results of this chapter will be far away from the ones obtained in previous chapter due to our computational limitations. However, some sketches of predator-prey Lotka-Volterra behaviour will be observed here.

To conclude, we will present a short summary of the results and make some general comments in the final chapter 6. Also, new paths will be suggested in order to solve the weak points of our algorithm.

Chapter 2

Lotka-Volterra equations

In this chapter we will put into context the origins of the well-known Lotka-Volterra equations, followed by an analysis of its physical meaning and mathematical solution

2.1 Brief history of mathematical ecology models

Mathematical ecology is the scientific field which models the evolution of size of populations in their ecosystems. First worth-mentioning publication in this field was made in 1798 by Thomas R. Malthus [21]. In his book *An Essay on the Principle of Population* he presented an equation describing the population growth, known as *Malthusian growth model*, which is no other thing than an exponential law, according to

$$\frac{\mathrm{d}P}{\mathrm{d}t} = rP(t), \tag{2.1}$$

where $P(t)$ is the population of the considered species and r a positive growth ratio. Despite the simplicity of the model, it influenced several researchers of 19th century, such as Darwin's *On the Origin of Species* work.

The well-known Lotka-Volterra equations are the next remarkable model of population evolution. They were proposed independently by Alfred J. Lotka in 1925 and Vito Volterra in 1926 [22].

year	1914	1915	1916	1917	1918	1919	1920	1921	1922	1923
predator %	11.9	21.4	22.1	21.2	36.4	27.3	16.0	15.9	14.8	10.7

Table 2.1: Percentage of predator species over all species fished each year. [23]

On the one hand, Vito Volterra got his conclusions based on the observations of his son-in-law, the biologist Umberto D'Ancona. D'Ancona organized the data of total caught species of fish during World War I and next years, organizing in predator fishes (such as sharks, rays, etc.) and prey fishes [23,24]. Percentage of predator species fished is shown in table 2.1 where we can observe that predators species increased its number during a period of reduced fishing activity such as the war period (1914-1918). This increasing of the predator fishes proportion encouraged Volterra to model a system such benefits prey population in case an external agent (such as fishing activity) appears. At the end of next section we will show how Lotka-Volterra equations satisfy this property.

On the other hand, Alfred J. Lotka initially proposed the same set of equations in the theory of autocatalytic chemical reactions in 1910. Later, in 1920, Lotka used the same equations to characterize prey-predator systems [25].

2.2 The Lotka-Volterra model

Lotka-Volterra equations are a pair of first order non-linear differential equations given by

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= \alpha x - \beta xy \text{ and} \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= \delta xy - \gamma y, \end{aligned} \tag{2.2}$$

where x is the number of preys, y is the number of predators and $\{\alpha, \beta, \delta, \gamma\}$ are four real parameters which describe the nature of the system.

To understand the physical meaning of this model, we have to assume an ecosystem where preys have infinite supply of food and predators depend on the interactions with other preys in order to reproduce and achieve the survival of the species.

Thus, in the absence of predators, preys will reproduce uncontrollably and

the population will increase exponentially (Malthusian behaviour) with the ratio α: $\dot{x}(t) = \alpha x \rightarrow x(t) = x(0)e^{\alpha t}$. On the other hand, in absence of preys, the number of predators would exponentially decrease with ratio γ due to the absence of food: $\dot{y}(t) = -\gamma y \rightarrow y(t) = y(0)e^{-\gamma t}$.

The other two terms in the equations represent the effect on interactions. Since the number of interactions increase with the size of both populations, these terms will be proportional to both x and y. This term will be positive for predators (δxy) since they benefit from interactions and negative for preys ($-\beta xy$) since they get damaged in interactions.

The solution's behaviour can be qualitatively explained in the following way: an increase of the number of preys causes an increase of the amount of available food for predators, which consequently increases its population. This causes that preys get damaged due to interactions so their population decreases. Consequently, predators will have less food available so its population will also decrease. Then, the number of preys can start increasing again due to the lack of predators and the cycle starts again. Thus, as we shall see more formally in the following, solutions will be periodically oscillating functions of time.

Now that we have a qualitative idea of the system, let us take a deep look into the solutions of the Lotka-Volterra model. There are two fixed points in time: the trivial one, $\{x = 0, y = 0\}$ and the non-trivial one

$$x = \frac{\gamma}{\delta}, \quad y = \frac{\alpha}{\beta}. \tag{2.3}$$

Since the sets $\{x(t) = x_0 e^{\alpha t}, y(t) = 0\}$ and $\{x(t) = 0, y(t) = y(0)e^{-\gamma t}\}$ are solutions contained in the x and y axis, any solution with $x_0 > 0$ and $y_0 > 0$ will be contained in the $\{x(t) > 0, y(t) > 0\}$ region of the XY plane for any $t > 0$ [23].

By dividing Eqs. (2.2) we get

$$\frac{\mathrm{d}y}{\mathrm{d}x} = \frac{-\gamma y + \delta xy}{\alpha x - \beta xy} = \frac{y(-\gamma + \delta x)}{x(\alpha - \beta y)}. \tag{2.4}$$

This equation can be separated resulting in

$$\frac{\mathrm{d}y(\alpha - \beta y)}{y} = \frac{\mathrm{d}x(-\gamma + \delta x)}{x}, \tag{2.5}$$

which can be integrated as

$$\alpha \ln y - \beta y + \gamma \ln x - \delta x = k \tag{2.6}$$

for some integration constant k. Applying the exponential on both sides and redefining the constant ($e^k \to k$) we get

$$y^\alpha e^{-\beta y} x^\gamma e^{-\delta x} = f(y)g(x) = k, \tag{2.7}$$

where we defined

$$\begin{aligned} f(y) =& y^\alpha e^{-\beta y} \\ g(x) =& x^\gamma e^{-\delta x} \end{aligned} \tag{2.8}$$

Note that these function (f and g) non-negative with $f(0) = g(0) = 0$ and $f(y \to \infty) = g(x \to \infty) = 0$, with only one maximum at $y_{\max} = \alpha/\beta$, $x_{\max} = \gamma/\delta$. The behaviour of these functions is shown in Fig. 2.1.

Let us define $M_f = f(y_{\max})$ and $M_g = g(x_{\max})$. Assuming we are not at the fixed point $\{x = x_{\max}, y = y_{\max}\}$ we will have $k < M_f M_g$ or what is the same, $k = \lambda M_g$, with $\lambda < M_f$. Of course, the value of λ is defined by the initial conditions ($\lambda = f(y_0)g(x_0)/M_g$)

Now, we can rewrite Eq. (2.7) as

$$g(x) = \frac{\lambda}{f(y)} M_g. \tag{2.9}$$

Therefore, for y values such $f(y) < \lambda$ there will be no solution for x. This means y-solution is contained between y_1 and y_2, the two solutions of $f(y) = \lambda$ (shaded zone at Fig. 2.1). As long as y approaches y_1 or y_2, x trajectory will approach γ/δ. Same reasoning can be done defining $k = \lambda' M_f$ to state than x-solution is bounded between the two solutions of $g(x) = \lambda'$ and y approaches its central α/β value when x approach to x_1 and x_2 values. Therefore, trajectories are cyclical on the $x > 0$, $y > 0$ zone; provided that we are not on the fixed point $\{x_0 = \gamma/\delta, y_0 = \alpha/\beta\}$ and $\{x_{1,2}, y_{1,2}\}$ there are return points [23, 26, 27]. In Fig. 2.2 we can observe these cyclic trajectories surrounding the fixed point.

Although there exists no analytical solutions for the functions $x(t)$ and $y(t)$, we can get an analytical estimation for initial values close to the fixed point. We will assume $x = \gamma/\delta + \Delta x$ and $y = \alpha/\beta + \Delta y$ with $\Delta x << x$ and

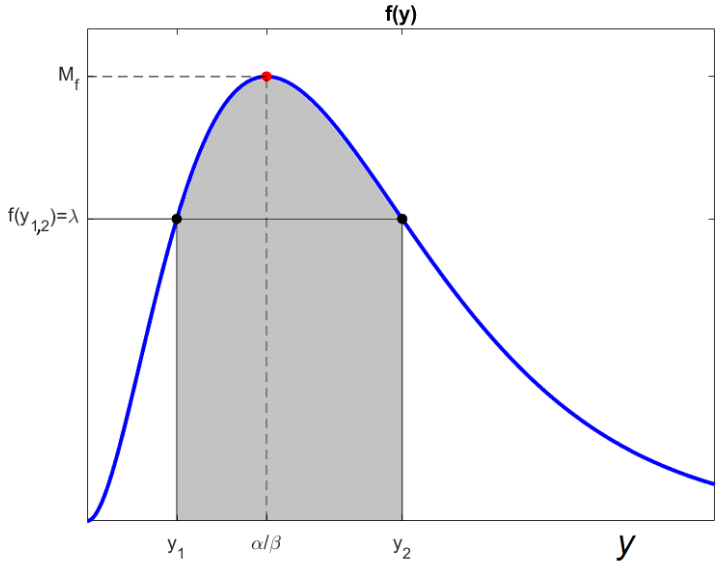

Figure 2.1: Representative plot of $f(y)$. Similar behaviour holds for $g(x)$, but with peak at γ/δ.

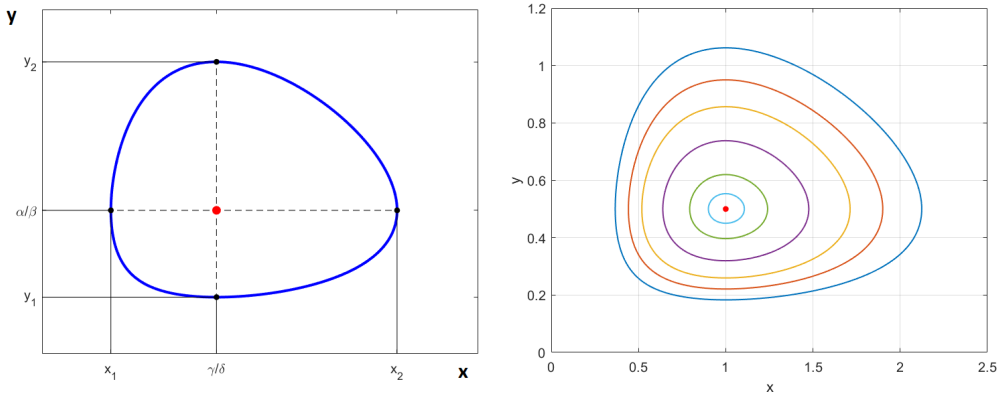

Figure 2.2: At left, cyclic representative trajectory. At right, several solutions with different initial conditions for given parameters $\alpha = 1, \beta = 2, \gamma = 1, \delta = 1$.

$\Delta y << y$. Neglecting terms proportional to $\mathcal{O}(\Delta x \Delta y)$, we can rewrite the

Lotka-Volterra equations (2.2) as

$$\begin{aligned} \Delta\dot{x} &\simeq -\frac{\beta\gamma}{\delta}\Delta y \\ \Delta\dot{y} &\simeq \frac{\delta\alpha}{\beta}\Delta x \end{aligned} \tag{2.10}$$

Differentiating again we get:

$$\begin{aligned} \Delta\ddot{x} &= -\alpha\gamma\Delta x \\ \Delta\ddot{y} &= -\alpha\gamma\Delta y \end{aligned} \tag{2.11}$$

so we can observe that populations oscillate harmonically around the fixed point with $\omega^2 = \alpha\gamma$.

For bigger deviations of the fixed point we can check numerically that oscillations become non-harmonic but period is still the same $T = \frac{2\pi}{\sqrt{\alpha\gamma}}$ as shown in figure (2.3) [26].

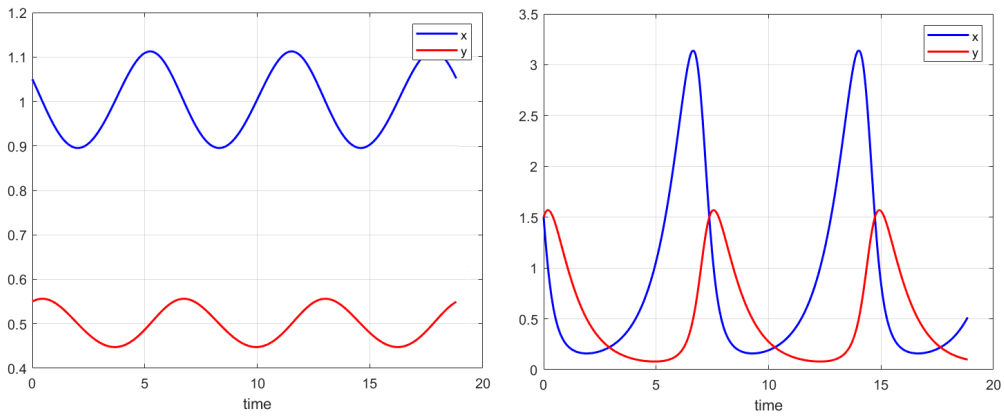

Figure 2.3: On the left panel the (almost) harmonic solutions of L-V equations near the fixed point $x_0 = 1, y_0 = 0.5$ is displayed. On the right, the non-harmonic oscillations due to initial conditions not close to the fixed point. Chosen parameters were $\alpha = 1$, $\beta = 2$, $\gamma = 1$ and $\delta = 1$.

Now that we have a deeper understanding of the Lotka-Volterra solutions we can explain D'Ancona observations mentioned at previous section. D'Acona data reflected average number of individuals during long (yearly) periods. For

example, isolating $x(t)$ from the second of Eqs. (2.2) it is straightforward to see that average over a period is

$$\bar{x} = \frac{1}{T}\int_0^T x(t)\mathrm{d}t = \frac{1}{T}\int_0^T \Big(\frac{1}{\delta}\frac{\dot{y}}{y} + \frac{\gamma}{\delta}\Big)\mathrm{d}t = \frac{\gamma}{\delta}, \tag{2.12}$$

where we cancelled the first term using the fact that

$$\int_0^T \frac{\dot{y}}{y}\mathrm{d}t = \ln(y(T)) - \ln(y(0)) = 0. \tag{2.13}$$

Similarly, we get that $\bar{y} = \alpha/\beta$ implying that the average number of individuals are equilibrium points.

Now, let us assume that we can model fishing as a factor that decrease both preys and predators with same ratio ϵ, so L-V equations become

$$\begin{aligned} \frac{\mathrm{d}x}{\mathrm{d}t} &= \alpha x - \beta xy - \epsilon x = \alpha' x - \beta xy \ \text{ and} \\ \frac{\mathrm{d}y}{\mathrm{d}t} &= \delta xy - \gamma y - \epsilon y = \delta xy - \gamma' y, \end{aligned} \tag{2.14}$$

which are same equations with $\alpha' = \alpha - \epsilon$ and $\gamma' = \gamma + \epsilon$. Therefore, as long as ϵ is small enough to not change the α' sign, we conclude that new average prey population $\bar{x}' = \gamma'/\delta > \bar{x}$ benefits of the fishing activity while on the contrary predator population $\bar{y}' = \alpha'/\delta < \bar{y}$ benefits of the lack of fishing activity. So Lotka-Volterra model match perfectly with D'Ancona observations.

Provided all the above information about Lotka-Volterra population evolution, now we will introduce a concrete model for life. This means that we will present a model in which the interaction between individuals and the environment as well as with other individuals is explicitly discussed described and we will discuss whether the populations of this model will follow the Lotka-Volterra equations. As we will see, the particular characteristic of this model is that is nature is purely quantum, so we shall name it as a Quantum Life Model.

Chapter 3

Quantum Artificial Life

In this chapter we will introduce our life model, starting with the motivation and idea behind Quantum Life. We will also describe current Quantum Artificial Life models so that we have an overall view of their advantages and limitations.This general view will provide us a better understanding of the goals and perspectives of this thesis.

3.1 What is Quantum Artificial Life?

Quantum Artificial Life is a brand new research topic on the field of quantum computation. The idea between this topic is to make use of quantum mechanics to improve life simulations model.

For this, we will construct an analogy between what we consider an alive individual in any life simulation model and the qubit, the fundamental element of quantum computing. Definition of Life is a very active topic of discussion, since it can be interpreted from the point of view of so many branches of knowledge including biology, physics or philosophy [28, 29].

It is worth to mention that from a thermodynamic point of view, we can consider alive a physical system which is able to decrease their internal entropy at the expense of matter or energy taken in from the environment. Also, we can interpret the death as the reaching of thermodynamic equilibrium between the organism and the environment [30].

Of course there are many extensive definitions from each branch of biology

but we will just take the simplest characteristic of them in order to construct our analogy as follows.

- Alive individuals are born. This is represented by the initialization of the qubits (individuals) in a predefined initial state.
- Alive individuals interact with the environment and progressively age. This is represented by the loss of information of qubits due to decoherence.
- Alive individuals interact with other individuals. In a very similar way, qubits interchange information using many-qubits gates.
- Alive individuals self-replicate. Likewise, qubits can copy partial information to other qubits.
- Alive individuals eventually die. Similarly, in a quantum computer qubits reach equilibrium state after some finite amount of time.

These considerations will be in the core of our life model.

3.2 Quantum Artificial Life previous models

So far, we gave a general vision of the analogies that can motivate a Quantum Artificial Life model. In order to construct our own model, let us first take a look of what has been done in this topic.

This project is strongly based on the work described in Refs. [31] and [32]. The model described in these publications can be interpreted at the starting point of the model that will be proposed in the present work. Also modifications on the model proposed in Ref. [33] has played a special relevance for this project, as we shall see in next sections.

In order to understand our model it is necessary to briefly explain the model presented in Ref. [31]. The essential features of that model can be states as follows

- **Life unit**. Individuals are described by two-qubits states: **genotype** and **phenotype**. First one stores the genetic information of the individuals, which is transmitted from generation to generation. Environment

cannot affect the state of genotype but it can suffer random mutations. The second one, phenotype, suffers the action from the environment, and it gets modified with interactions with other individual and the passage of time. When an individual is created, the information is copied from the genotype to the phenotype as we will see in section 3.2.1.

- **Death**. Essential property of life. In this model, the essential quantum observable is $\langle\sigma_z\rangle$, being σ_z the Pauli matrix. However, we will use a convention which is different for most standard definition of σ_z:

$$\sigma_z = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.1}$$

This matrix representation is based on the usual computational basis:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{3.2}$$

This non-standard notation is more consistent with the physical situation on an actual computer, where $|1\rangle$ state is a more energetic state than $|0\rangle$. This will be specially relevant at some points of this project

Individual is considered dead once $\langle\sigma_z\rangle_p$ (on the phenotype subspace) reaches a certain minimal value $\langle\sigma_z\rangle_p < \sigma_{death}$[1]

- **Preys and predators**. We have two kinds of species. Preys are born with greater $\langle\sigma_z\rangle$ than predators. This causes that predators' life expectancy is shorter than preys' life expectancy. If the predators' newborn ratio is shorter enough, predator species' survival will rely on the interaction with preys.

- **Self-replication**. Again, essential characteristic of any life model. This model uses asexual reproduction based on the copy of partial information as we will see in more details later.

[1] Actually in Ref. [31] and [32] the convention used for σ_z is different, so individuals are dead when they reach a maximal value $\langle\sigma_z\rangle_p > \sigma_{death}$. However, we are still describing the exactly same model with a slightly different notation.

- **Environment**. Passage of time must affect the phenotypes of individuals. A Lindblad master equation is used for this purpose, since this equation describes most of the open quantum system dynamics [36].

In the next sections we will take a further look into these features of the model. Also, we will explain some of the disadvantages of the original model and some first improvements that has been proposed for it.

3.2.1 Self-replication

The non-cloning theorem states the non-existence of an unitary operator that could copy an arbitrary qubit state into a blank state, thus making two copies of the same unknown state. Such impossible operator would be perfect for this model so that we could use it to replicate the progenitor's genotype into the progeny's genotype. Also this would allow us to copy then information from the new born's genotype to its phenotype.

However, there exist the possibility of partially copying the quantum information from one qubit to the other. Specifically, we are interested in producing a copy only of the expectation value of an operator $\mathcal{O}$.

The mathematical formalism is as follows [35]: given a quantum state ρ, the expectation value of an observable $\langle\mathcal{O}\rangle_\rho$ and another blank state ρ_e, there exists a partial cloning unitary operator, $U(\mathcal{O},\rho)$ which acts on the product state $\rho\otimes\rho_e$ such as

$$\langle\mathcal{O}\rangle_\rho = \mathrm{Tr}(\mathcal{O}\rho) = \langle\mathcal{O}\otimes\mathbb{1}\rangle_{U(\rho\otimes\rho_e)U^\dagger} = \langle\mathbb{1}\otimes\mathcal{O}\rangle_{U(\rho\otimes\rho_e)U^\dagger}\,. \tag{3.3}$$

This means that, in the new state $U\left(\rho\otimes\rho_e\right)U^\dagger$, the result of measuring $\langle\mathcal{O}\rangle$ on first or on the second subsystem will be the same. Of course this is a copy of the expectation value of a specific operator, and for other operators we will need a different U gate.

The general expression for this partial cloning gate is given in Ref. [35], yet it will not be necessary for our purposes. We aim to copy the $\langle\sigma_z\rangle$ value between two qubits (two $2-$dimensional systems). For this case, the partial

cloning gate is the C-NOT

$$U_{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{3.4}$$

whose matrix representation is based on the product state of the aforementioned computational basis.

It is straightforward to visualize the $\langle \sigma_z \rangle$ cloning with C-NOT gate using pure states $\rho = |\psi\rangle \langle\psi|$. Let us consider a general state $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$ that we aim to copy into the $|0\rangle$ blank state. Applying the C-NOT gate we obtain

$$|\psi'\rangle = U_{CNOT} |\psi\rangle |0\rangle = \alpha |00\rangle + \beta |11\rangle . \tag{3.5}$$

Now we can easily see that regardless of which of the two qubits we measure, the result will be the same

$$\langle \sigma_z \otimes \mathbb{1} \rangle_{|\psi'\rangle} = \langle \mathbb{1} \otimes \sigma_z \rangle_{|\psi'\rangle} = -|\alpha|^2 + |\beta|^2. \tag{3.6}$$

To conclude, a C-NOT gate can copy the information regarding the expectation value of σ_z, which is our relevant observable. Thus, we shall use this gate in order to both copying the genotype state to the phenotype one and also the progenitor's genotype into the newborn's genotype.

3.2.2 Environment (aging)

Lindblad master equation

As we mentioned earlier, the passage of time will be modeled by making use of a Lindblad master equation [36]

$$\frac{d\rho}{dt} = \mathcal{L}\rho = \gamma\Big(\sigma\rho\sigma^\dagger - \frac{1}{2}\sigma^\dagger\sigma\rho - \frac{1}{2}\rho\sigma\sigma^\dagger\Big) \tag{3.7}$$

where $\sigma \equiv |0\rangle \langle 1|$ and $\gamma > 0$ is a decay parameter.

A complete derivation of Lindblad equation can be found in Ref. [37, 38]. For our purposes, we will just remark the most important assumptions needed to apply Lindblad equation so that we can justify its appearance in our model.

We shall assume a small system ρ in presence of a environment ρ_{env} with infinite degrees of freedom. Coupling is assumed to be small and both systems are initially uncorrelated, so we can write them as product states $\rho(0) \otimes \rho_{\text{env}}$. Also, we will assume zero temperature $T = 0K$ and we will accept that system has no memory and thus we can make use of Markovian approximation [36, 38, 39].

All this assumptions reflect the physical reality of a quantum computer, a system constituted by few degrees of freedom (qubits) an a environment that weekly interacts with qubits causing decoherence.

As we will see in the following, the evolution of $\langle\sigma_z(t)\rangle$ under this equation implies an exponential decay. Given a general 2-level quantum state, represented by the density matrix

$$\rho(t) = \begin{pmatrix} 1 - a(t) & b(t) \\ b^*(t) & a(t) \end{pmatrix}, \tag{3.8}$$

along with the initial condition $\langle\sigma_z(0)\rangle = \text{Tr}\{\sigma_z\rho_0\} = 2a_0 - 1$, the master equation takes the form

$$\dot{\rho} = \begin{pmatrix} 1 - \dot{a} & \dot{b} \\ \dot{b}^* & \dot{a} \end{pmatrix} = \gamma \begin{pmatrix} a & -b/2 \\ -b^*/2 & -a \end{pmatrix}. \tag{3.9}$$

The solution of this equation can be found to be

$$\rho(t) = \begin{pmatrix} 1 - a_0 e^{-\gamma t} & b_0 e^{-\gamma t/2} \\ b_0^* e^{-\gamma t/2} & a_0 e^{-\gamma t} \end{pmatrix}. \tag{3.10}$$

Consequently, the evolution of $\langle\sigma_z(t)\rangle$ is given by

$$\langle\sigma_z(t)\rangle = 2a_0 e^{-\gamma t} - 1 \tag{3.11}$$

which indicates a decay of the initial value to the ground-state value $\langle\sigma_z(t \to \infty)\rangle = -1$.

The main problem here is the fact that quantum algorithms works in discrete steps, so we need to find how to discretize this time evolution in order to build its quantum simulation. The solution presented in Ref. [32] is based on a rotation than can simulate this decay of $\langle\sigma_z\rangle$ only for specific states. Therefore, we will reject this time-passing mechanism because it requires the knowledge of the previous state of the phenotype in order to apply an specific rotation that can reduce its $\langle\sigma_z\rangle$ value.

Amplitude-damping channel

Despite modelling aging using Lindblad equation is reasonable, we need a concrete method to implement this aging in specific qubits. This would be achieved using the operator sum representation of the dynamics. The following matrices, called Kraus operators [3]

$$A_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-\eta} \end{pmatrix} \tag{3.12}$$
$$A_1 = \begin{pmatrix} 0 & \sqrt{\eta} \\ 0 & 0 \end{pmatrix},$$

with $0 \leq \eta \leq 1$ represent the same evolution for the $\rho(t)$ as the Lindblad master equationin the form

$$\mathcal{E}_{AD}(\rho) = A_0 \rho A_0^\dagger + A_1 \rho A_1^\dagger. \tag{3.13}$$

This operation is also called quantum amplitude-damping channel, and it was proposed in Ref. [33].

Let us compute the result of the amplitude-damping channel acting on a general state in the form

$$\rho = \begin{pmatrix} 1-a & b \\ b^* & a \end{pmatrix}. \tag{3.14}$$

Applying Eq. (3.13) and simplifying we obtain

$$\mathcal{E}_{AD}(\rho) = \begin{pmatrix} 1-a(1-\eta) & b\sqrt{1-\eta} \\ b^*\sqrt{1-\eta} & a(1-\eta). \end{pmatrix} \tag{3.15}$$

We can compare this result with the solution of the Lindblad equation given in Eq. (3.10) and notice that both results are the same by making the following identification

$$1 - \eta = e^{-\gamma t}. \tag{3.16}$$

Therefore, this mechanism allows us to perform a discrete time-passing step equivalent to a time $\gamma t = -\log(1-\eta)$ on the Lindblad equation. Note that for a small time steps $\gamma t, \eta << 1$ both time parameter and quantum channel parameter are proportional: $\eta \simeq \gamma t$.

Quantum channels are not unitary operations, so there not exists any gate than can compute this operation by itself. Yet they can be implemented making use of an ancillary qubit as shown in Sec. 5.1. This can be a computational problem since we will need to use this quantum channel several times but it is still our best choice. Therefore, we will use this method as our mechanism to simulate the passage of time.

3.2.3 Environment (interactions between individuals)

Discarding first proposal

In Ref. [31] the proposal for the interaction gate between two individuals is the following one: a controlled unitary operation which uses genotypes of the individuals as control qubits and phenotypes as target qubits. When control qubits are equal, the target qubits remain unchanged. On the contrary, when control qubits are different, target qubits are exchanged as shown in table 3.1.

in	out	in	out	in	out	in	out
$\|0000\rangle$	$\|0000\rangle$	$\|0100\rangle$	$\|0100\rangle$	$\|1000\rangle$	$\|1000\rangle$	$\|1100\rangle$	$\|0110\rangle$
$\|0001\rangle$	$\|0001\rangle$	$\|0101\rangle$	$\|0101\rangle$	$\|1001\rangle$	$\|0011\rangle$	$\|1101\rangle$	$\|1101\rangle$
$\|0010\rangle$	$\|0010\rangle$	$\|0110\rangle$	$\|1100\rangle$	$\|1010\rangle$	$\|1010\rangle$	$\|1110\rangle$	$\|1110\rangle$
$\|0011\rangle$	$\|1001\rangle$	$\|0111\rangle$	$\|0111\rangle$	$\|1011\rangle$	$\|1011\rangle$	$\|1111\rangle$	$\|1111\rangle$

Table 3.1: Truth table of the interaction gate proposed in Refs. [31,32], where control qubits are the second and fourth ones and target qubits first and third ones.

The purpose of this gate is that when two qubits represent individuals of same species (same genotype) they do not interact, but when they represent individual of different species (prey and predator) the interaction benefit the species with low $\langle\sigma_z\rangle$ (predator) while harming the species with high $\langle\sigma_z\rangle$ (prey).

But once again, this mechanism cannot be used for a system including several individuals repeatedly interacting between them. The reason is that interchanging phenotypes is not a realistic prey-predator interaction mechanisms since it does not necessarily always benefit to predators while damaging

preys. For example, if a prey and a predator that have previously interacted, interact again, they will interchange its phenotypes a second time, in which case the effect of two interactions will simply cancel. This need to be fixed and we will see how do do this in the next section.

Two amplitude-damping channels

Since previous mechanism does not represent a realistic model, we need to explore new possibilities. Once again, in Ref. [33] it was proposed using quantum channels as a solution. The idea is simple, in order to decrease prey's $\langle\sigma_z\rangle$ value we will use the amplitude-damping channel already explained in Sec. (3.2.2) with different parameter η'. So the effect of this new channel, as explained before, is to reduce the $\langle\sigma_z\rangle_{prey}$ in the following way

$$\langle\sigma_z\rangle_{prey} = 2a - 1 \rightarrow \langle\sigma_z\rangle'_{prey} = 2a(1-\eta') - 1. \tag{3.17}$$

In order to increase predators' $\langle\sigma_z\rangle$ value we will use a similar quantum channel (let us name it as *inverse* amplitude damping) based on the matrices

$$B_0 = \begin{pmatrix} \sqrt{1-\eta'} & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and}$$
$$B_1 = \begin{pmatrix} 0 & 0 \\ \sqrt{\eta'} & 0 \end{pmatrix}. \tag{3.18}$$

Note that these two matrices are the same as the ones in Eq. (3.12). In order to visualize the results in a more intuitive way, now we consider the general state ρ written as

$$\rho = \begin{pmatrix} a & b \\ b^* & 1-a \end{pmatrix}, \tag{3.19}$$

so that initially, $\langle\sigma_z\rangle_{predator} = 1 - 2a$.

The result of applying this new operation can be computed by means of $\mathcal{E}_{IAD}(\rho) = B_0\rho B_0^\dagger + B_1\rho B_1^\dagger$, resulting in

$$\mathcal{E}_{IAD}(\rho) = \begin{pmatrix} a(1-\eta') & b\sqrt{1-\eta'} \\ b^*\sqrt{1-\eta'} & 1-a(1-\eta') \end{pmatrix}. \tag{3.20}$$

Thus, the evolution of $\langle\sigma_z\rangle_{predator}$ is given by

$$\langle\sigma_z\rangle_{predator} = 1 - 2a \rightarrow 1 - 2a(1-\eta). \tag{3.21}$$

Therefore, we can see that $\langle\sigma_z\rangle$ is actually increasing by a factor $(1-\eta')$ under the action of this channel. $\langle\sigma_z\rangle$ will get closer to 1 as long as η' gets closer to 1.

3.2.4 Death

As mentioned earlier, this model considers that individuals are alive as long as its phenotype $\langle\sigma_z\rangle$ is greater than some threshold value σ_{death} (see footnote 1). As we will see in the next section, this mechanism has both advantages and inconveniences.

The main advantage is that this mechanism can be related with the nature of a qubit. Since qubits are physical systems affected by decoherence (environment) this death mechanism can be analogous to the physical situation of a qubit getting affected by physical phenomena such as decay which causes a natural decreasing of the $\langle\sigma_z\rangle$. We will try to take profit of this situation in further sections.

The main disadvantage is the fact that we need to know how many individuals are alive in each step and, thus we need to measure these values in each step of the quantum computation. But quantum physics only allows us to measure $\langle\sigma_z\rangle$ through the destruction of the quantum state, which interrupts the execution. Fixing this problem will require more computational resources.

3.3 Space and structure of the simulation

Once proposed a mechanism for individuals to interact, self-replicate, and die we need to define the space where these individuals can be placed and distribute interactions.

The proposal is simple (also based in Ref. [33] with some modifications): we set a 2-dimensional grid where individuals can be in one of the cells. We will initially distribute the individuals randomly on the lattice. In each step of our simulation, individuals will randomly move to one of the 8 adjacent cells

or stay in the same cell, with equal probability. Movement is aborted if cell was already occupied. Also, movements to a cell outside the limit of the lattice are corrected to the closest cell inside the lattice.

Two individuals interact when they are placed in adjacent cells, as shown in Fig. (3.1).

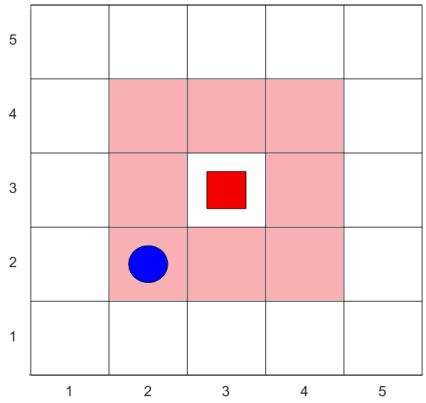

Figure 3.1: Grid with a predator (red square) and a prey (blue circle) interacting. Red shaded cells form the interacting field of the predator

Also, there will be a certain probability for each alive individual to self-replicate after each step. If this happens, self-replication mechanism will be executed and a new individual will be randomly assigned to a free cell.

Note that our spatial dynamics is purely classical, and grid must be programmed by classical means. This indicates the unavoidable combination between quantum and classical computing that will characterize our simulations.

3.4 State-of-the-art of Quantum Artificial Life and new objectives

Now that we have an overall view of the model, we need to know at which point is the development of this model.

As we have already mentioned, Quantum Artificial Life is a new-born topic in quantum computation and Ref. [31–35] are the only publications where

we can find something about this model. In Ref. [35] the authors provide a deeper view into the mathematical formalism beyond the partial-cloning method. In Ref. [31] the aforementioned basis of the model is set, while in Ref. [32] a first implementation on a IBM Quantum Computer is performed. This implementation consists on the simplest Quantum Artificial Life that we can design: just two individuals are initialized, an interaction between them is performed, followed by the simulation of the passage of time. There is no spatial dynamics in this first approach.

In Refs. [33] and [34] a few more steps were taken. Besides the improved features already mentioned, in Ref. [33] a more complex system is numerically simulated. Here a dynamic system were individuals are placed in a grid is used and the number of individuals is increased up to 4 individuals. However, simulation is purely numerical, assembling density matrices. Thus, there is no implementation on a quantum computer.

Having an overall view of the topic allow us to clearly state our goals. Due to the huge number of qubits that it would require, it is not reasonable to say that our goal is to implement a prey-predator system making use of a quantum computer that reproduces Lotka-Volterra equations in the appropriate limit. We expect the Lotka-Volterra dynamics to emerge when both the number of individuals and the number of steps are large, as we will discuss later. Such system would require a number of qubits out of the current quantum computation resources.

But we want to do some approach to this objective, so we could say that **our goal is to check whether it is possible to observe a Lotka-Volterra prey-predator system in a large enough quantum computer**. For this, we will make use of the following classical means

- First, a classical algorithm where the essential characteristics of the model is presented, i.e., the classical algorithm performs all the mechanisms of our model but classically imitating the action of these mechanisms on the $\langle \sigma_z \rangle$ of each specie. We do not worry about the efficiency of the algorithm here.

  This algorithm will be optimized so that we can reach large number of individuals ($\sim 10^3$) and steps ($\sim 10^3$) using reasonable computational times so we can check if your model will indeed work on a quantum

computer.

- Second, the *IBM Qasm* Simulator, which is a classical simulator that allow us to implement noise models on our simulations so that our results will be closer to a real quantum execution. This simulator only allows us to use up to 32 qubits so we are still strongly limited on the number of individuals and the results will not reproduce Lotka-Volterra equations, as expected.

Chapter 4

Classical simulation

Before we design an implementation of our model on a quantum computer, we will make a first classical approach. So we will build a classical program that simulates the behaviour of the $\langle \sigma_z \rangle$ value for each individual, without using the quantum formalism. This will allow us to simulate a much larger number of individuals than the number that quantum computation can allow us.

In this chapter major decisions regarding the quantum simulation will be taken, such as removing the genotypes from our system or simplifying the predator-prey interaction.

4.1 Basis of the simulation

Our aim is to check whether the model that we will use for the quantum simulations can properly replicate the Lotka-Volterra equations. So this program will replicate some of the variables that are relevant for the quantum code. Then, we will use a string σ_z variable which is analogue to the expected value $\langle \sigma_z \rangle$ of each qubit, and gives us the information whether individuals are alive or not. Damaging of individuals decrease the σ_z value, and individuals will be dead when $\sigma_z < \sigma_{death}$ for a certain σ_{death}. Key points of simulation are

- We have a 2-dimensional grid where individuals are located. Each step they move randomly to a contiguous cell.
- In each step there is a certain probability for each individual to self-replicate.

- Self-replication probability for preys is constant (parameter α of LV equation) but self-replication probability for predators is proportional to the number of preys (parameter β of LV equations)

- In each step all individuals are damaged by the passage of time.

- If two individuals are in contiguous cells, they will interact and this will damage the prey. Finally, we decide not to increase σ_z value of predator at each interaction since we already increase its born ratio with the number of preys. Thus, the behaviour of the βxy term of Lotka-Volterra equations is already implemented here. We tried also increasing predators' σ_z each interaction, with almost identical results.

- For this section, we decided to remove the genotypes from the model and thus, renounce to observe natural evolution and focus on populations evolution. Self-replication is performed similarly, except that initial phenotype value is the same for all individuals of the same species. In the last section, we will implement genotypes and discuss its contribution to the model

4.2 How aging and damaging are simulated

We want this program to reproduces the results that the quantum simulation will provide. As we explained in Sec. 3.2.2, the quantum code makes use of several **amplitude damping** quantum channels which allow us to decrease the σ_z value by a factor of $e^{-\gamma_i}$, where γ_i is the damping factor[1]. We will use a parameter γ_{Time} when the channel is used for simulate aging and a different value γ_{Feed} when the channel represents a feeding interaction.

As we explained, on the quantum part we will use the following convention: $\langle 0|\,\sigma_z\,|0\rangle = -1$ and $\langle 1|\,\sigma_z\,|1\rangle = 1$, so our analogous variable σ_z needs to be $-1 \leq \sigma_z \leq 1$. Consequently, the function that decreases the value cannot be such simple as $\sigma_z \to \sigma_z e^{-\gamma_i}$ since we will decrease until $\sigma_z = 0$ which is not the minimal value.

[1]For simplicity, we redefined the factor γt to a single dimensionless factor γ since a single parameter completely characterizes the channel.

To properly perform the decreasing of σ_z by some factor we first need to linearly re-scale σ_z into a σ'_z such $0 \leq \sigma'_z \leq 1$, and then revert the change. This is required so that if we apply the quantum channel several times our σ_z will approach to the minimal value, -1.

The relation between the two conventions of σ_z is simply given by

$$\sigma'_z = (\sigma_z + 1)/2, \quad \sigma_z = 2\sigma'_z - 1. \tag{4.1}$$

So, the function that decreases σ'_z due to time or feeding will be

$$f'_{\gamma_i}(\sigma') = \sigma' e^{-\gamma_i}, \tag{4.2}$$

where i can refer to *feed* or *time* depending on which channel we are using. Then, the function that we will use for decreasing σ_z is

$$f_{\gamma_i}(\sigma_z) = 2 \times \left(\frac{\sigma_z + 1}{2} e^{-\gamma_i}\right) - 1 \tag{4.3}$$

Obviously, if we repeatedly apply this function we get a bigger decreasing. Applying n times this function is equivalent to apply it once with $n\gamma$ decay factor as $f^n_{\gamma_i}(\sigma_z) = f_{(n\gamma_i)}(\sigma_z)$.

Figure 4.1 shows how an individual with initial $\sigma_{z,0}$ is affected when this function is repeatedly applied.

4.3 Functioning of the program

Let us explain in a schematic way how the program works step by step.

- Set the parameters in the model (size of grid, number of initial individuals, self-replication probabilities, γ_i, number of steps, etc.).
- Randomly distribute individuals on the grid.
- In each step we perform the following actions:
  - To check, for each individual, if there are interactions (prey and predator in contiguous cell) and decrease prey's σ_z value using previously explained function $f_{\gamma_{\text{Feed}}}$.
  - To simulate the passage of time by decreasing all σ_z values making use of $f_{\gamma_{\text{Time}}}$.

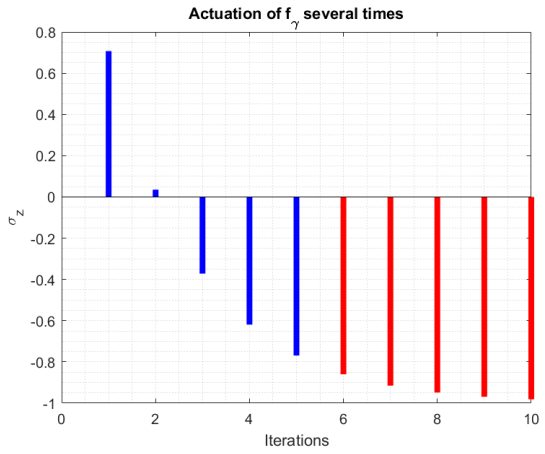

Figure 4.1: Effect of applying f_γ with $\gamma = 0.5$ several times on an individual with $\sigma_{z,0} = 0.7$. Red lines denotes that individual will be dead setting $\sigma_{death} = -0.8$.

- To check which individuals are dead and remove them from the grid.
- With a certain probability, each individual can self-replicate and create another individual of same species.
- To store the number of alive individuals.
- Randomly move individuals into a contiguous cell.

- Finally, we visualize the results. The program allows us to visualize the grid evolution as in shown in Fig. (4.2)

Of course we have to understand that Lotka-Volterra equations are real variable equations while the number of individuals is integer. Also, this simulation is based on random movements so this randomness can make results differ from predictions. The effects of these two factors will be smaller as long as we increase the number of individuals of or simulation, the limit where we expect to observe the emergence of Lotka-Volterra behaviour.

One of the main manifestation of these problems was the extinction of the populations. Since Lotka-Volterra solutions can reach minimal values really close to zero, the randomness of the simulation could cause that some of the populations eventually vanish. This situation is not reversible since, once

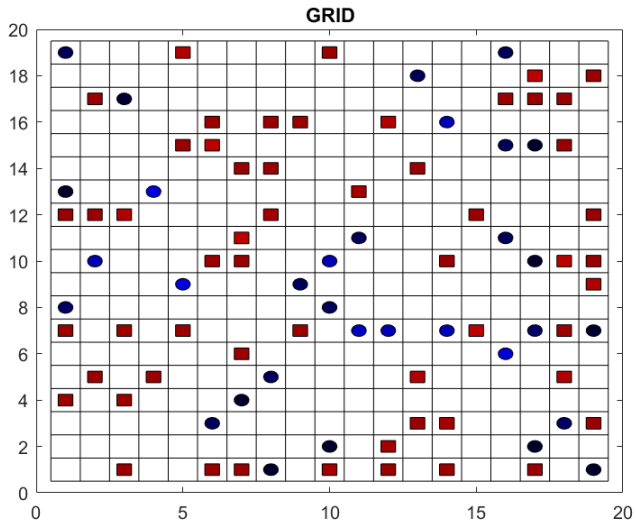

Figure 4.2: Screenshot of the grid in one step of a small simulation. Red squares represent predators while blue circles represent preys. Brighter colour indicates individual with bigger σ_z value

a population has extinguish there is no probability of self-replication. This motivated us to **avoid the last individual of each population to die**. So the program allows an individual to be alive with $\sigma_z < \sigma_{death}$ until a new one is born.

4.4 Results of simulations

We made several combinations of the set of parameters so that we could approach a behaviour as similar as possible as the Lotka-Volterra solutions. A similar behaviour (i.e. increasing the number of preys leads to an increase in that of the predators, which leads to a decreasing of the preys population and so on) can be observed for all choices of parameters. But this Lotka-Volterra-like behaviour does not always match exactly Lotka-Volterra solutions.

Figure (4.3) shows a comparison between the results of our simulation and Lotka-Volterra solutions. As we can see, both plots seems to fit properly, but in the simulation peaks may be not so regularly distributed and cycles are

Simulation parameters		Solution parameters	
preys newborn probability	0.15	α	0.07
predators newborn probability	8E-4	δ	5E-4
feeding channel parameter γ_f	0.507	β	3.5E-4
time channel parameter γ_t	0.059	γ	0.165

Table 4.1: Parameters used in the simulation showed in Figs. 4.3 and 4.4.

not exactly the same. Differences can be more visible in the implicit plot of Fig. (4.4), where we observe that simulated behaviour is not purely cyclical as in the analytical solutions. The parameters used in both simulations and solutions are displayed in Table 4.1.

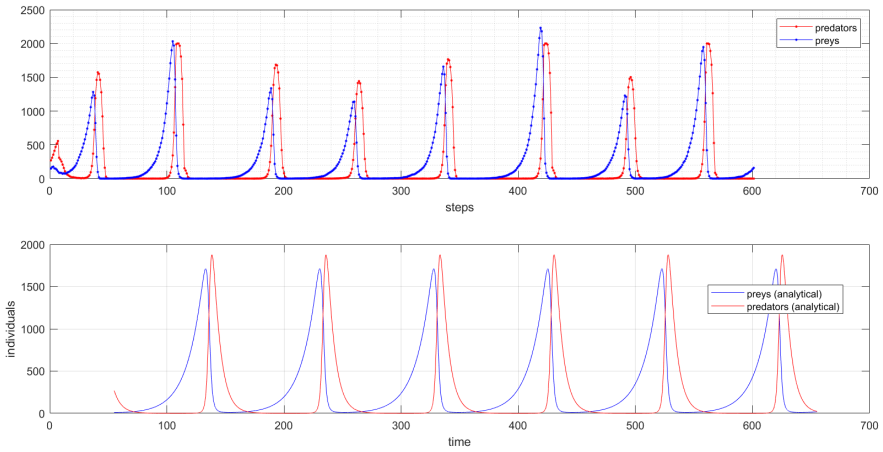

Figure 4.3: Comparison between the simulation and Lotka-Volterra solutions.

There are several reasons that prevent us to use formal methods such as χ^2, in order to obtain the best fit and analyze its reliability.

First of all, the solutions are not analytical. For example, χ^2 function for preys, given a set of results $\{x_i\}$ with no error is [40]

$$\chi^2 = \sum_i \left(x(t_i) - x_i\right)^2 . \tag{4.4}$$

Minimizing this function requires to numerically solve LV equation for the preys $x(t)$ for some unknown parameters $\alpha, \beta, \gamma, \delta$ each time we evaluate χ^2

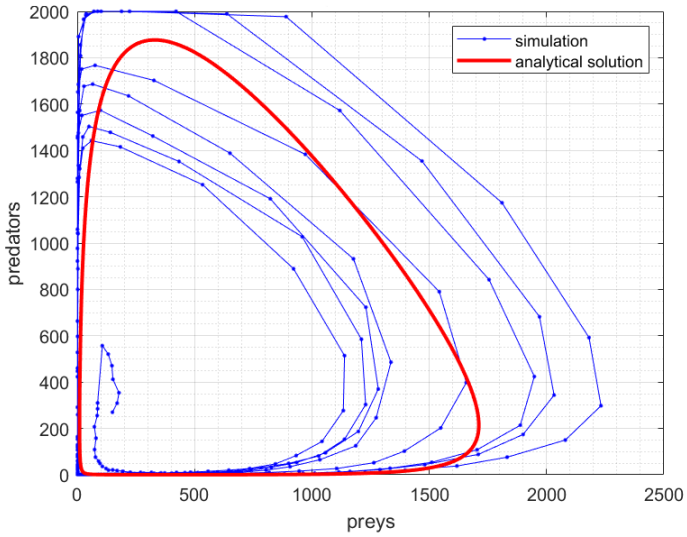

Figure 4.4: Comparison between simulation and Lotka-Volterra solutions, implicit plots.

looking for a minimum. This leads to long computational times looking for the optimal parameters.

Yet even ignoring this problem, χ^2 test may not be a good method for obtaining the best parameters beyond the simulation, since the small randomness of simulation can introduce fluctuations on the period of the solutions. This can desynchronize the oscillations of simulations and LV solutions and consequently, reasonable values of parameters may have a bad result with this test while some solutions which do not reflect the actual behaviour can have lower χ^2 values, as we can see in Fig. 4.5.

4.5 Reducing the number of individuals

Since the objective of this classical simulation is to explore the behaviour that our quantum life model will have, we will try to reduce the number of individuals so that our future quantum simulation can use less qubits. In order to achieve this, we will present the results for several configuration of the parameters, by decreasing the number of individuals each time.

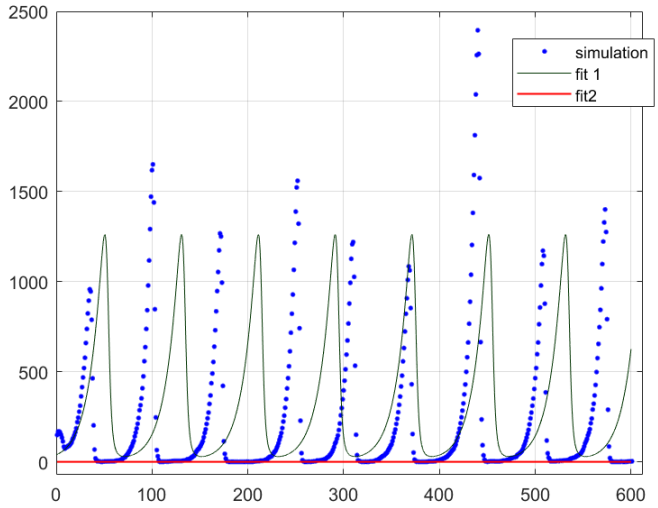

Figure 4.5: Results of the simulation along with a fit that reflects the simulation behaviour (green) and a trivial non-representative solution (red). Fit 1 has $\chi^2 \simeq 1.38 \times 10^5$ and fit 2 has $\chi^2 \simeq 1.37 \times 10^5$.

In order to reduce the maximal number of individuals we need to modify several parameters: first, the grid size must be reduced so that the number of interactions decreases proportionally and we can compare distinct simulations. Also, we needed to reduce prey's newborns ratio α and initial number of individuals. By trial-and-error we decided also to increase predators self-replication ratio (since predators self-replication probability needs to be multiplied by the number of preys and the number of preys is decreasing, increasing this parameter makes the system more stable). Different parameters of 4 simulations with progressively less individuals are shown in Table 4.2.

Of course, as we previously explained, a big number of individuals reduces the randomness of the system so that we expect that reducing the number of individuals should lead to a worst result. This is in fact observed in Fig. 4.6 where we show a comparison between this four simulations with progressively less individuals.

We must note that, in Fig. 4.6, we always chose one of the best results after performing several simulations. This allowed us to observe that the less individuals we use, bigger differences are observed in the simulations with same

Simulation	a	b	c	d
preys newborn probability	0.15	0.07	0.05	0.05
predators newborn probability	8E-4	1.5E-3	3E-3	6E-3
feeding channel parameter η_f	0.507	0.507	0.507	0.507
time channel parameter η_t	0.059	0.059	0.059	0.059
initial number of preys	150	20	20	20
initial number of predators	270	60	60	20
lattice size	60×60	30×30	30×30	15×15

Table 4.2: Parameters used in the simulations shown in Fig. 4.6.

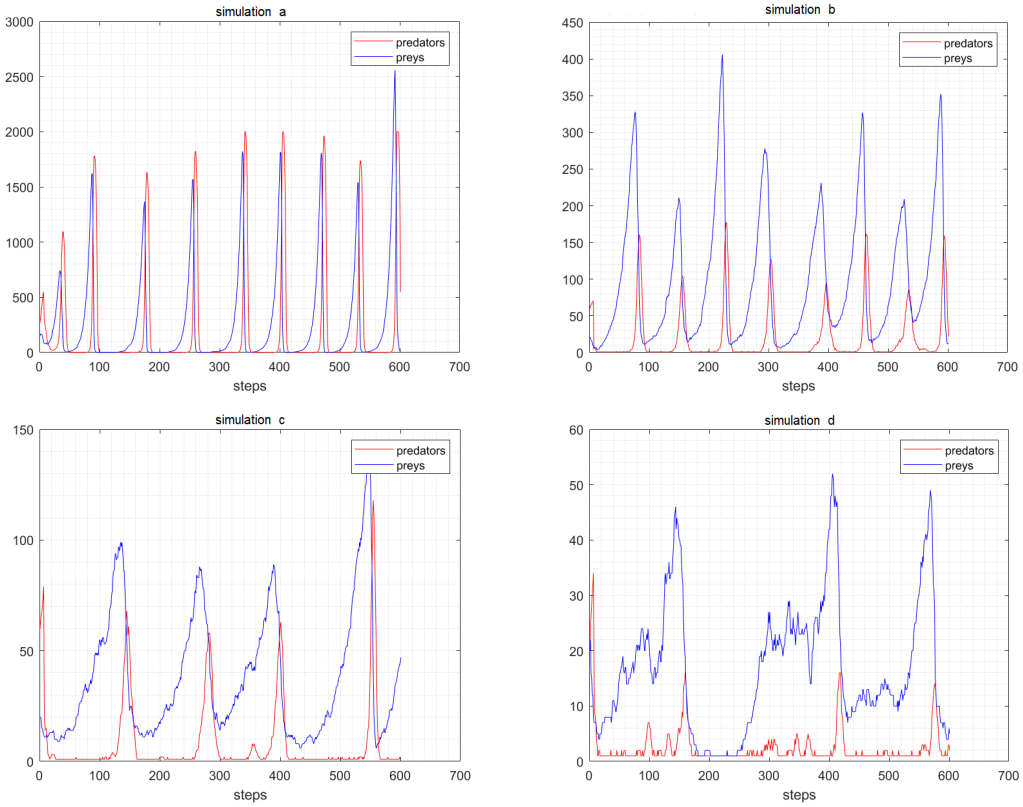

Figure 4.6: From left to right and up to down, comparison of the four progressively smaller simulations with parameters shown in Table 4.2.

parameters.

4.6 Genotypes and natural evolution

We also built a version of the simulation which includes genotypes.

Functioning is really similar, but in each step we perform a random mutation on genotypes. This mutation consist of adding a random number $r \in [-0.01, 0.01]$ to the genotype. Mutation is only allowed if resulting σ_z of the phenotype is in the range $\sigma_{death} < \sigma_{phenotype} < 1$. Thus, when a new individual is born its genotype and initial phenotype will be a copy of the father's genotype.

The results (shown in Fig. 4.7) are essentially the same when it comes to the population evolution, and we can observe a really slow increase in the mean value of the population genotype as we have new generations of individuals.

Two conclusions can be drawn from this discussion

- We need a large number of steps to observe natural evolution. This and the fact that we need to double the number of qubits forces us to discard this option for the simulation on a quantum computer.

- Our results indicate a slow tendency of increasing average population genotype σ_z value, for both phenotypes and genotypes. This means that, as long as the individuals are evolving, they become healthier, since new individuals are born with a larger σ_z on phenotype, which gives them more life expectancy.

4.7 Classical simulation: Conclusions

As we showed, it is possible to simulate a system using our model which accords fairly to the Lotka-Volterra. But the level of precision that we aim to achieve can limit us due to the need of a higher number of individuals. Making use of thousand of individuals allows us to achieve excellent results, but this will not be possible in the quantum simulation. We can notice that for a few tens of individuals (~ 70 individuals) the dynamics is quite irregular but we can still observe a Lotka-Volterra-behaviour. This number of individuals is still out of our quantum computational possibilities. So we can conclude that our goal of observing Lotka-Volterra on the quantum simulation is very hard, demanding

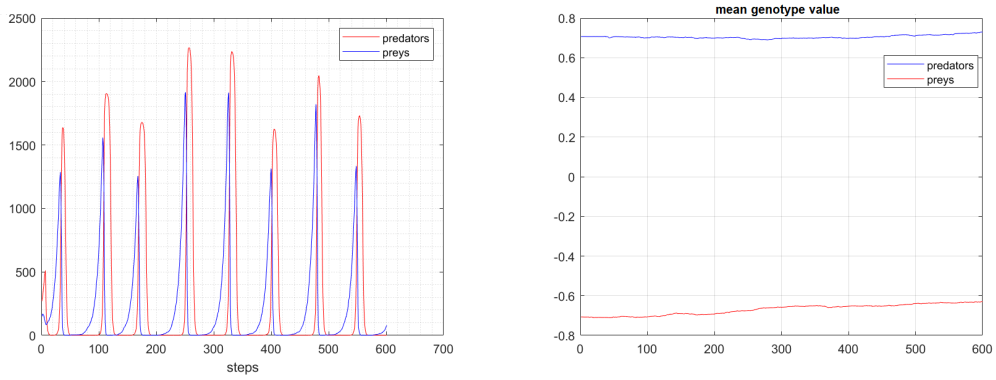

Figure 4.7: Simulation results. In the first figure, we observe that the evolution of the populations is really similar to the first example shown in Fig. 4.6. In the second figure, we show how the average genotype of the population slowly increases in time.

improvements in the quantum hardware. However, we can still observe clues of some similar behaviour, indicating that the theoretical proposal actually works.

Regarding genotypes, we can also state that our model follows a Darwinian evolution. Yet this evolution has a characteristic time considerably longer than the evolution times considered here. Therefore, adding Darwinian characteristics to the model will be discarded in the next section so we can focus on the proper population evolution.

Chapter 5

Quantum Simulation

Since we now have a better perspective of what should we expect, let us focus on design an algorithm that we can execute on a quantum computer.

First, we will explain in detail how we shall implement each of the aforementioned mechanisms of our model. Also, major contributions to the model of this project are explained in this chapter, since they mainly consist of approaches designed to make possible the simulation of the model on a quantum computer.

Then, we will have to glue the mechanism together with the spatial dynamics in order to construct the whole prey-predator system simulation.

As we explained at Sec. 3.4, at the moment our only possibility is to execute our algorithm on a simulator. So we will focus some details of the explanation thinking about the *Qasm Simulator* by IBM, although every step can be extrapolated to any of the most common quantum computers.

5.1 Aging

As we explained in Secs. 3.2.2 and 4.2, our idea is to make use of a quantum amplitude-damping channel in order to simulate the passage of time due to discrete steps.

Since quantum channels are not unitary operations they cannot be directly implemented as a quantum gate: they require the use of ancillary qubits. The simplest implementation can be found in Ref. [3]. It consist of a controlled-y

rotation by an angle (θ_t) followed by a C-NOT gate, as shown in Fig. (5.1).

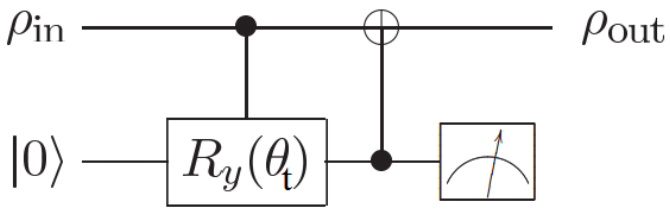

Figure 5.1: Quantum circuit for implementing the amplitude-damping channel [3].

It can be shown [3] that this circuit performs the quantum operation given in Eq. (3.13) with $\eta = \sin^2(\theta_t/2)$.

However, our model needs to simulate aging to each phenotype at each step. For a system with N individuals with a maximum of S steps, this would require the number of $N \times S$ ancillary qubits. So our computational limit will be reached incredibly soon. This is the main motivation for the following proposal.

5.1.1 Taking profit of the natural decay of the qubits

Qubits are generally physical systems where $|1\rangle$ represents the excited state which naturally decays to the ground state $|0\rangle$. Thus, if we keep the circuit static (without applying any gates) for a certain time t of the order of the decay time of the qubit ($T1$), we will observe a very similar behaviour of the one produced by the amplitude-damping channel. This coherence time is characteristic of each qubit. We should expect similar coherence times for all the qubits of a quantum computer although some quantum computers have considerably different $T1$ for each qubit. However, it would be interesting to interpret the differences of $T1$ times between qubits as the fact that different individuals die at different ages due to several causes (diseases, accidents, etc...) and the average $T1$ would be the average life expectancy.

It is worth also mentioning the natural dephasing which causes the randomization of the relative phase between the components $|0\rangle$ and $|1\rangle$ of the qubit state. This process is characterized by the $T2$ time. Yet this phenomena

has no special relevance for us since we are measuring $\langle\sigma_z\rangle$ operator which does not take into account dephasing between $|0\rangle$ and $|1\rangle$ states.

Coherence times of some of the IBM quantum computers are shown in Table 5.1. They are calibrated daily [41], but the order of magnitude does not change.

Quantum Computer	# qubits	$T1$: max-min-average ($\pm 1\mu s$)	$T2$: average ($\pm 1\mu s$)
IBM Q Melbourne	14	19-102-57	59
IBM Q5 Rome	5	70-111-91	117
IBM Q5 London	5	61-107-75	85
IBM Q Yorktown	5	44-72-58	62
IBM Q Armonk	1	195	148

Table 5.1: Coherence times of some of the public access IBM quantum computers. Based on [42] but updated using data from [41] (11/06/2020).

Also, this idea is extremely coherent with the philosophy of Quantum Artificial Life: we are taking profit of how time progressively destroy quantum information to precisely simulate how time progressively destroy life. So this mechanism is included in the very nature of quantum physics.

Thus, applying several identity gates which makes no other thing than delay the execution of the program we will get a natural decay of the state $|1\rangle$ towards the ground state $|0\rangle$. The number of identity gates needed will depend on the run time of each gate (which also depend on the calibration) and the strength of the decay that we want to observe (related with parameter η of amplitude-damping channel).

5.1.2 Noise Models in Qasm Simulator

As we have already mentioned, we will need a large amount of qubits. *IBM Q Melbourne* is the biggest public access IBM quantum computer, but its decay times are extremely short and very different between all the qubits. This confirms us the fact that we are restricted to use the *IBM Qasm* Simulator. Quantum simulators are generally ideal simulators which introduce no noise to our computations, so this idea of taking profit of the natural decay may not be applicable to our simulations. Luckily, *IBM Qasm* Simulator allows

us to include customized error models into our simulations [43], which can be extremely helpful for us. Among all the possible error models that we can implement, such as phase damping or thermal relaxation, we will choose the amplitude-damping. This means that when we implement this model, the decay to the ground $|0\rangle$ state will be modelled through an amplitude-damping in certain gates but it will not require an extra qubit to implement the quantum channel. So we will get same results with no need of using ancillary qubits.

Also, we have plenty of freedom applying this noise model: we can choose which gates and which qubits will be subjected to this error. Of course, we can also choose the error strength through the amplitude-damping parameter η. We will implement the error on phenotypes qubits and on identity gates. Thus, our time-passing simulation will consist of just identity gates in the phenotype qubits with a noise model parametrized by our chosen value of η.

Of course, in a real quantum computer the practical situation will be quite different: we would have to calibrate how many identity gates should we implement in order to achieve a proper decay in phenotypes qubit. But the final result will be similar. Furthermore, if we are implementing genotypes in our model, same decay will be produced on the qubits representing them, and we do not want genotypes to be affected by the environment. This last issue motivated us to propose the solution of next section.

5.1.3 Proposal to avoid damping at genotype

We know that quantum computation is performed in parallel on all qubits, so the aforementioned mechanism of delay the computation on some qubits will affect all qubits, including the genotypes. Genotypes are modelled so that they do not suffer other change than random mutations. Before we explain our proposed solution, let us clarify two points:

- It is not our goal in this project to implement simulations including genotypes. This proposal is explained for completeness and to show that (ignoring our lack of quantum computational resources) it is possible to implement a model including genotypes using our time-passing mechanism.

- Our proposal has been tested on the simulator. Implementation on a

quantum computer may require further calculations and tests in order to calibrate the number and positions of the gates.

Given that a time-passing operation is given by T identity gates (here, we have chosen $T = 6$ since we can choose the effect of identity gates using previous model), we can insert in the genotype 2 NOT gates between all these identity gates: first NOT gate will be placed close to the beginning and second NOT gate at the end. Of course, at the phenotype we will implement the same number of identity gates with no NOT gates inserted between them. The idea is that, before we place first NOT gate, the physical sate had decay about half of the total decay that we should expect. Then, the other half of the process is performed between NOT gates, which leads to the inverse behaviour. Thus, second part of the process causes an increasing of the $\langle \sigma_z \rangle$ of the state that cancels the first decay.

In Fig. 5.2 our positioning of gates is showed and how this achieves to leave the genotype's $\langle \sigma_z \rangle$ unmodified. An amplitude-damping noise model was implemented using a parameter $\eta = 0.06$.

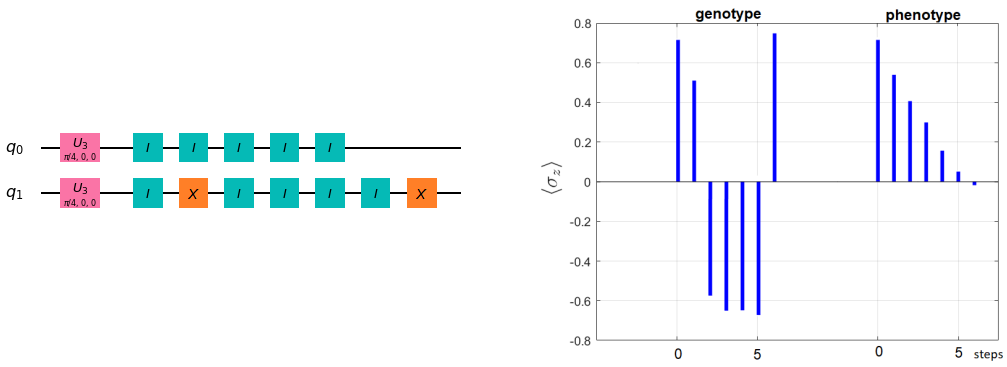

Figure 5.2: At left, gates distribution for genotype (q_1) and phenotype (q_0) are shown. At right, we first observe the $\langle \sigma_z \rangle$ genotype evolution each step: NOT gates cause bounces and finally we obtain (almost) same initial value. On the other hand, we can then observe that phenotype $\langle \sigma_z \rangle$ is progressively decaying each step

At the end, the genotype's $\langle \sigma_z \rangle$ final vale is almost the same as initially, in such a way that the quantum computation will not interfere with the results.

Small and random variations on the genotype can be perfectly justified in a Darwinian model as random mutations. In Fig. 5.3 we show how the genotype's and phenotype's $\langle\sigma_z\rangle$ evolve after 20 steps, just like the one shown in Fig. 5.2.

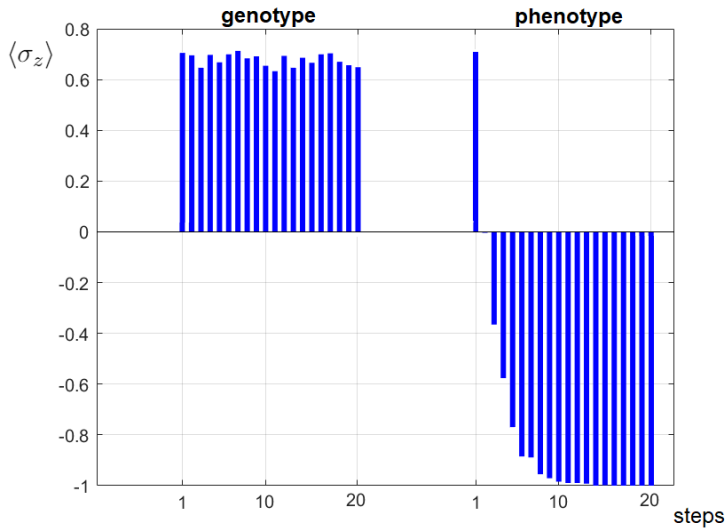

Figure 5.3: Results after 20 iterations of the quantum circuit shown in Fig. 5.2. We can observe how the genotype value (first 20 bars) is almost constant, suffering small random mutations, while the phenotype (last 20 bars) decreases exponentially.

5.2 Interaction between individuals

Once again, extending the idea in Ref. [33], we will implement this mechanism by making use of quantum channels. In the initial propose, two quantum channels were necessary: the first one, the already known amplitude-damping channel was implemented at prey's phenotype in order to reduce its $\langle\sigma_z\rangle$, while the second one —inverse amplitude-damping— works in a similar way to increase $\langle\sigma_z\rangle$ value, as explained in Sec. 3.2.3. The implementation of this second channel can be performed just realizing that the matrices describing it (Eq. (3.18)) are exactly the same as the usual channel, but reversing the order of the states $|0\rangle$ and $|1\rangle$. This allows us to use the same implementation as

in the amplitude-damping channel but adding two NOT gates to the target qubit before and after the circuit, as shown in Fig. 5.4.

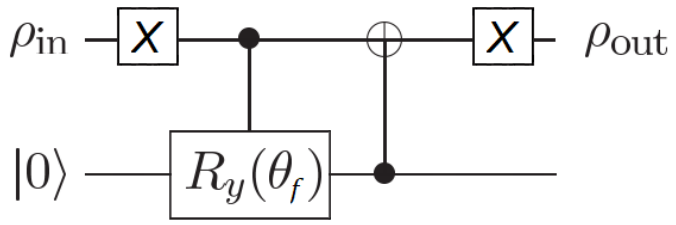

Figure 5.4: Inverse amplitude damping. Now damping parameter will be $\gamma_f = \sin^2(\theta_f/2)$

Of course, a formal proof that this circuit actually performs inverse amplitude-damping operation can be worked out, but since NOT gates just reverse the states $|0\rangle$ and $|1\rangle$, this reasoning is enough to support our claim.

However, as we showed in Sec. 4.1, it is not necessary to increase preys' $\langle\sigma_z\rangle$ since Lotka-Volterra behaviour will be already implemented by making new-born probability proportional to the number of preys. So, we will just use the standard amplitude-damping for prey's phenotype.

Of course we could use the same mechanism as in time-passing to decrease $\langle\sigma_z\rangle$, but we finally decided not to due it because of the following reasons:

- The time-passing mechanism takes place in every step for every individual, while feeding interaction is more occasional.
- The feeding interaction should be much stronger than time-passing interaction, so we will need an extraordinary amount of identity gates to cause such big time delay. Also, it is performed between only 2 individuals, which may lead to huge computational times and other type of noise problems.

So we will use an ancillary qubit each time two individuals are interacting. IBM *Qasm* Simulator allows us to restore qubits to the $|0\rangle$ state in the middle of the implementation of the circuit. It is true that, nowadays, this is not possible in a real quantum computer, but we shall use it in our simulations to only employ one ancillary qubit, thus saving computational resources.

5.3 Self-replication

Self-replication is performed by copying the state from progenitor's phenotype to the descendant phenotype, or if we include genotypes in our model, from progenitor's genotype to the descendant genotype and then from descendant genotype to its phenotype. Partial state copying is performed using a C-NOT gate as explained in Sec. 3.2.1.

Self-replication has a certain probability of occurrence for every individual, in each step, as explained in the classical program section. This probability is constant for the preys while, in the case of the predators, it is proportional to the number of alive preys.

5.4 Death

As we mentioned while explaining the model, individuals are dead at the point its phenotype $\langle\sigma_z\rangle$ reaches a minimal value σ_{death}. Yet here it comes what is the main problem of Quantum Artificial Life: we have to know at each step whether individuals are alive or not in order to move them at the grid, interact, self-replicate, etc. But quantum computation only allows us to know the state of a qubit through the measurement of the qubits, which destroys the state of the system. So a quantum life simulation cannot be simply executed and look for the final results.

Our solution is the following one: each step we perform all life protocols that happened in previous steps such interactions, time-passing, and self-replication and then, we measure the circuit. We store the data and, by classical means we check which individuals are dead, i.e., we check whether any phenotype qubit has $\langle\sigma_z\rangle < \sigma_{death}$. If an individual is dead, we remove this individual from our system, which means, this individual will not interact or self-replicate again and its qubits will be used for further computations[1].

Then, we will repeat the exact same quantum computation but adding a new step (which may contain another interaction, self-replication, death),

[1]Once again, when implementing model on *Qasm Simulator* we will be able to restore qubits of death individuals in order to use them again, although this is not the general situation in a real quantum computer

but excluding previously dead qubits. After this new execution, the system is measured and data is again analyzed.

Of course this whole procedure of measuring data at each step and executing each time all steps until the current one, will noticeably delay computations, but is the only way to check at each step which individuals are still alive.

5.5 A complete quantum simulation

We have built a code which allows us to execute a complete scenario of quantum life including all the previously described mechanisms. This code combines scripts using QISKIT (the open IBM Q python library) and MATLAB ®. First, let us highlight and clarify some points about our final simulation:

- Individuals move randomly in a 2D classical grid as explained in Sec. 3.3.
- We also prevent last individual from each specie to be dead as shown and justified in Sec. 4.3.
- Since we have a limited amount of qubits it is unavoidable to set a maximal bound on the number of individuals. If the number of individuals of a specie is maximal, there will not be newborns.
- Finally, we will not be implementing genotypes since we do not have enough resources to observe natural evolution through this program.

The whole functioning of the program combines both quantum and classical simulation due to the aforementioned fact that we must measure after each new step and decide the next step using the information of the previous step. For this, it is important to mention the **agenda** variable. It consist of a string were each element is an event. Here, every event that take place at the simulation is reported and saved. Possible reported events are:

- Feeding: it saves which individuals have interacted and at which step.
- Death: it saves which individuals have died at each step.

- Newborns: it saves in each step, which alive individual is the father and which free qubit we will use as newborn individual.

The agenda variable is essential to combine classical and quantum computation. Figure 5.5 schematically explains the general behaviour of the program.

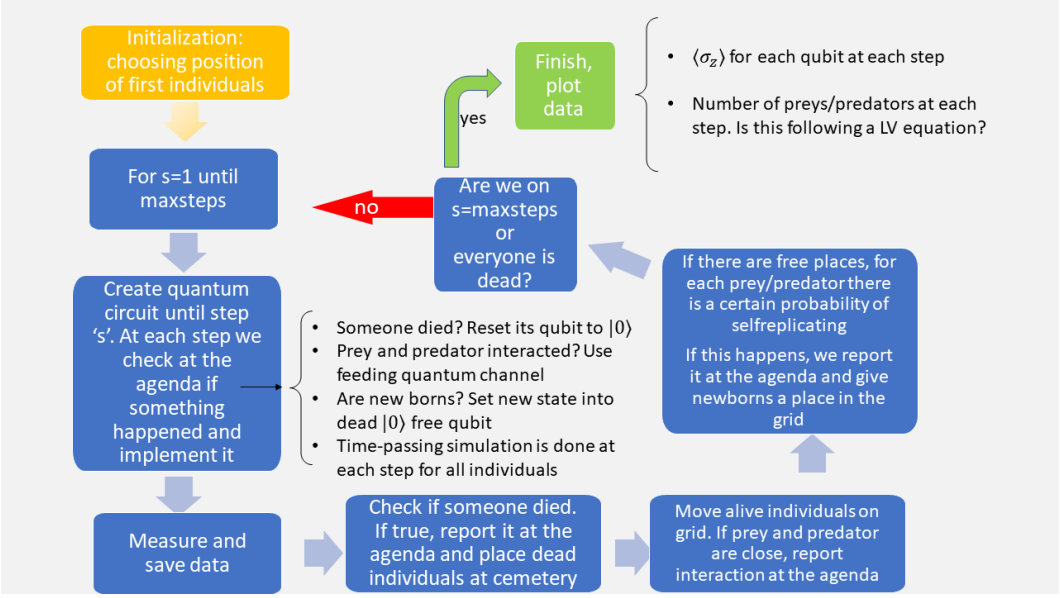

Figure 5.5: Scheme of the simulation.

After each simulation, we can observe all the data concerning the simulation, including the position of each qubit at each time and its $\langle \sigma_z \rangle$.

5.6 Results of quantum simulations

As we have explained, all simulations were performed at the *IBM Qasm* Simulator. Our main limitation using this simulator is not the limited of 32 qubits, but our own computational resources. Working with a standard personal computer, we cannot expect to compute in a affordable amount of time anything involving more than 20 qubits.

So most of the time invested in this project was dedicated to a trial-and-error work in two directions: on one hand, looking for some proper parameters

(of course classical simulations helped us since we used really similar parameters) and on the other hand, making the most of our computational resources and exploring which could be the limit of our simulations.

Finally, after several days of computations we achieve to put our limit in 17 qubits, using 1 qubit as ancillary, 9 qubits as prey's phenotypes and 7 qubits as predator's phenotypes. The parameters used for one of the best results obtained after several executions are shown in Table 5.2. In Fig. 5.6 we can check different results using these parameters.

Parameter	Value
σ_{death}	-0.8
Lattice dimensions	10×10
Prey's self-replication probability	0.2
Predator's self-replication ratio probability	0.1
Feeding channel parameter η_f	0.059
Time channel parameter η_t	0.507

Table 5.2: Parameters used in the simulation.

Although the graph of our results is not very conclusive, we can observe some signs of a Lotka-Volterra behaviour. In other words, we can see how a large amount of predators causes a drop on the number of preys and consequently also on the number of predators. After a some steps, preys number start to increase causing also an increase on the number of predators. Of course this behaviour is considerably vague in our results, due to the huge randomness of the system. As we expected, the small amount of individuals makes that randomness affecting the processes of interaction and self replication plays such an important role on the system. Also, we can observe that simulations are plenty different one from other.

However, our results from Chapter 4 already showed us that Lotka-Volterra dynamics emerged for larger number of individuals, having progressively worse and more random results when reducing the number of individuals and steps. So, classical simulations can endorse the thesis that our quantum model would follow Lotka-Volterra equations when considerably increasing the size of the simulation.

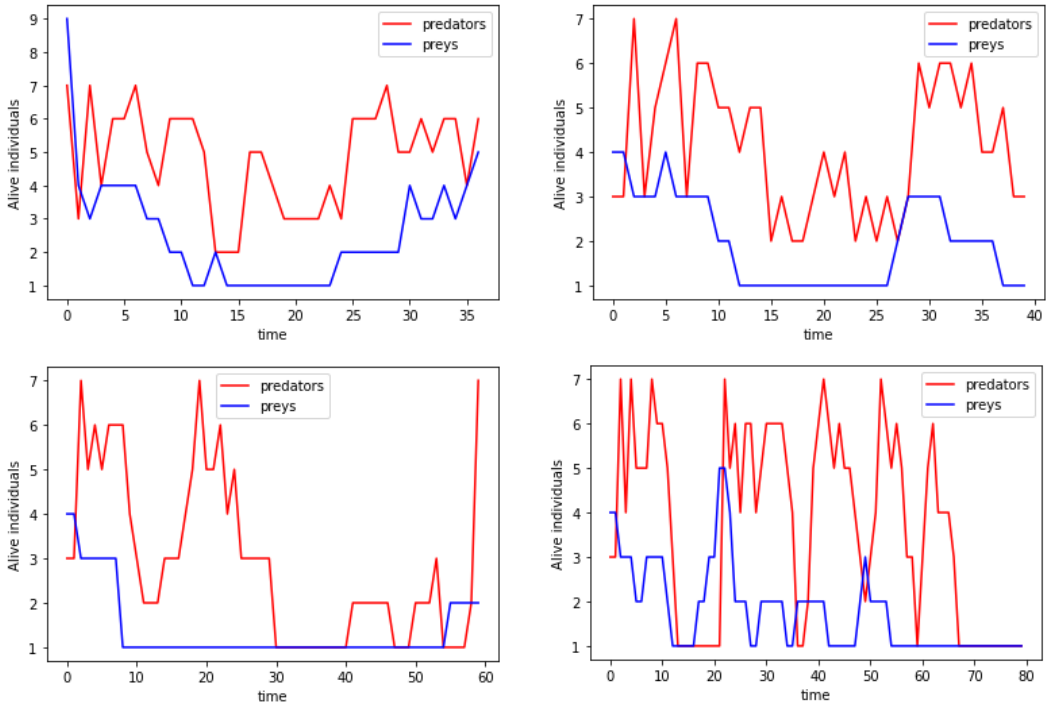

Figure 5.6: Results of population's evolution of the simulation for 4 different executions using, respectively 35, 40, 60 and 80 steps.

Chapter 6

Conclusions

In order to properly understand the advances of our work, it is important to know where was the starting point of Quantum Artificial Life, which was explained in Sec. 3.4 and the huge difficulties of observing a quantum system with the characteristics of a Lotka-Volterra model.

First, we can say that classical simulation emulating the quantum model was considerably successful: we replicated in a reliable way the mechanisms of our quantum model (self-replicating, interacting with both environment and other individuals and dying) into a classical program, in order to explode the capacity of the model. Results were pretty successful when a large number of individuals (in the order of few thousand) was implemented in the simulation. This large simulation followed in a really reasonable way the Lotka-Volterra equations. Also, a model using genotypes was implemented obtaining an slow increase on both species genotype. This prove us that genotypes model is a Darwinian model since it evolves according to natural selection.

Yet simulations are based on random factors, such as the random movement in the grid and the self-replication probability and this randomness becomes more evident when we decrease the number of individuals, and results starts to progressively differ of Lotka-Volterra solutions.

Secondly, we could say that the results of our quantum simulation where what we ought to expect: simulations are ruled by our own limitations so properties that we can observe are just sketches of what we would like to observe. This means, we do not truly observe a Lotka-Volterra behaviour on our system but we can sketch this behaviour by identifying certain tendencies

on the population to increase or decrease depending on the other population according to Lotka-Volterra solutions.

However, classical simulation works as a support since they prove that our quantum model could achieve same —or really similar— results provided that we have an enough large number of qubits available. Obviously, computational resources are nowadays the main limitations of Quantum Artificial Life. However, we could analyze classical vs quantum models ignoring this fact so that we can make a fair comparison.

On the one hand, the main advantage of quantum models is the natural way in which qubits adapt to individuals: using the decoherence of the system as a time-passing mechanism and gates errors or dephasing as a random mutations (when our model includes genotypes) provides us a optimal platform for including these mechanisms. Furthermore, partial-cloning mechanism is extremely optimized since only one C-NOT gate is required in order to copy the $\langle \sigma_z \rangle$ value from the progenitor to the son. These advantages translate into a optimization of quantum life model simulations in front of classical simulations.

Yet on the other hand, there are two big points in favour or classical simulations: The main one, the fact that quantum measurement destroys the current system, does not allow to know the state of the system at intermediate steps. This problem could be ignored if we are only interested in the final state of the system —which is not our case, since our goal was to observe a Lotka-Volterra behaviour so we needed to know the state at each step— and the solution that we devised requires more computational time: provided that a classical computation of n steps takes a time $\mathcal{O}(n)$, a quantum simulation that needs to measure at each step an repeat the simulation until next step is equivalent to perform $\sum_{k=1}^{n} k = n(n+1)/2$ steps, so quantum computation takes a $\mathcal{O}(n^2)$ time.

Also, the fact that we cannot reset qubits in most quantum computers plays an especial relevant role since qubits representing death individuals and qubits used in quantum channels are not used again. This can increase extremely fast the number of qubits that we need for our simulation, since every step newborn individuals and new interactions will take a brand new qubit.

Let us now give some perspectives that could be of great interest for future

research.

One simple idea that could strongly optimize the number of qubits used is to use previously dead qubits as a new individuals, since they can be considered after a long time in the ground state $|0\rangle$.

Also, we could avoid our great weakness (measuring system multiple times at each step) using weak measures that do not destroy the system or entangling ancillary qubits with an individual qubit so that we get information of the individual at a precise step.

Another different but interesting approach might be to study the thermodynamics of the system. Since life is constantly fighting against second thermodynamics law, a balance of the entropy of the system could be a great source of information about the simulation.

So as a final conclusion, we would say that Artificial Life is branch of computational science were quantum supremacy can be achieved, yet it is considerably more far away than other branches of computational science.

Bibliography

[1] Gordon E. Moore, *Cramming more components onto integrated circuits* Electronics Magazine, Volume 38, Number 8, April 1965

[2] V. Moret Bonillo, *Principios Fundamentales de Computación Cuántica* (2013). Faculty of Informatics, University of A Coruña. Computation Department.

[3] M. A. Nielsen, I. L. Chuan, *Quantum computation and quantum information* (2010). Cambridge University Press.

[4] R. Feynmann, *Simulating physics with computers* (1982). International Journal of Theoretical Physics, Vol 21, Nos. 6/7.

[5] J. J. Fernandez, *Richard Feynman and the birth of quantum computing* (January 2018). Medium. https://medium.com/quantum1net/richard-feynman-and-the-birth-of-quantum-computing-6fe4a0f5fcc7

[6] J. Pade, *Quantum mechanics for Pedestrians 2* (2018). Springer.

[7] P. Shor, *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer* (1994). IEEE Computer Society Press, pp. 124–134.

[8] Lov K. Grover, *A fast quantum mechanical algorithm for database search* Annual ACM Symposium on the Theory of Computing (May 1996)

[9] Sreeja Chowdhury et al., *Physical Security in the Post-quantum Era: A Survey on Side-channel Analysis, Random Number Generators, and Physically Unclonable Functions* May 2020 https://arxiv.org/abs/2005.04344

[10] Román Orús, Samuel Mugel, Enrique Lizaso, *Quantum Computing for Finance: Overview and prospects* Reviews in Physics Volume 4, November 2019

[11] Vedran Dunjko, Hans J Briegel, *Machine learning & artificial intelligence in the quantum domain: a review of recent progress* Reports on Progress in Physics, Volume 81, Number 7. June 2018

[12] James N. Eckstein, Jeremy Levy, *Materials issues for quantum computation* Cambridge University Press. Volume 38, Issue 10, October 2013

[13] Nikolaj Moll et al., *Quantum optimization using variational algorithms on near-term quantum devices* October 2017 https://arxiv.org/abs/1710.01022

[14] Bhupesh Bishnoi, *Quantum-Computation and Applications* June 2020 https://arxiv.org/abs/2006.02799

[15] Prashant S. Emani et al., *Quantum Computing at the Frontiers of Biological Sciences* November 2019 https://arxiv.org/abs/1911.07127

[16] Christoph Simon, *Can quantum physics help solve the hard problem of consciousness? A hypothesis based on entangled spins and photons* September 2018 https://arxiv.org/abs/1809.03490

[17] B. Adams, F. Petruccione, *Quantum effects in the brain: A review* October 2019 https://arxiv.org/abs/1910.08423

[18] A. Layeb, S. Meshoul, M. Batouche, *Multiple sequence alignment by quantum genetic algorithm* IEEE Xplore, June 2006

[19] A.S. Boev et al., *Genome assembly using quantum and quantum-inspired annealing* April 2020 https://arxiv.org/abs/2004.06719

[20] A. Perdomo-Ortiz et al., *Finding low-energy conformations of lattice protein models by quantum annealing* Nature Scientific Reports 2, 571, August 2012

[21] Thomas Robert Malthus, *An Essay on the Principle of Population As It Affects the Future Improvement of Society.* Retrieved 20 June 2015. via Internet Archive

[22] Mira-Cirstiana Anisiu, *Lotka, Volterra and their model*, Didactica Mathematica, Vol. 32(2014)

[23] Martin Braun, *Differential Equations and Their Applications. An introduction to Applied Mathematics* Springer (1993)

[24] S. Kingsland, *Modeling Nature: Episodes in the History of Population Ecology.* University of Chicago Press (1995)

[25] N. S. Goel et al., *On the Volterra and Other Non-Linear Models of Interacting Populations.* Academic Press (1971)

[26] F. Hoppensteadt, *Predator-prey model.* Scholarpedia 1 (2006)

[27] Chris Prior, *Lotka-Volterra,* Department of Mathematical Sciences, Durham University

https://www.maths.dur.ac.uk/~ktch24/term1Notes(10).pdf

[28] Santiago Sánchez-Migallón, *La escurridiza definición de vida,* Xataka, October 2015.

https://www.xataka.com/investigacion/la-escurridiza-definicion-de-vida

[29] Claus Emmeche, *Defining Life, Explaining Emergence.* Niels Bohr Institute, March 2012.

https://www.nbi.dk/~emmeche/cePubl/97e.defLife.v3f.html

[30] Erwin Schrodinger, *What is Life?*, Cambridge University Press, 1944.

[31] Unai Alvarez-Rodriguez, Mikel Sanz, Lucas Lamata & Enrique Solano, *Artificial Life in Quantum Technologies*, Nature Sci. Rep. 6 (2016)

[32] Unai Alvarez-Rodriguez, Mikel Sanz, Lucas Lamata & Enrique Solano, *Quantum Artificial Life in an IBM Quantum Computer*, Nature Sci. Rep. 7 (2018)

[33] Jon Cipitria Herrero, *Predator-Prey Behaviour in Quantum Artificial Life with Quantum Reinforcement Learning*, Final Grade Thesis, UPV-EHU, 2018.

[34] Asier Izquierdo, *Classical and Quantum Artificial Life*, Final Grade Thesis, UPV-EHU, 2019.

[35] Unai Alvarez-Rodriguez, Mikel Sanz, Lucas Lamata & Enrique Solano, *Biomimetic Cloning of Quantum Observables*, Nature Sci. Rep. 4 (2014)

[36] H.P. Breuer, F. Petruccione, *The Theory of Open Quantum Systems.* Oxford University Press (2006).

[37] Philip Pearle, *Simple Derivation of the Lindblad Equation.* April 2012 https://arxiv.org/abs/1204.2016

[38] Michel Le Bellac, *Quantum physics.* Cambridge University Press, 2006.

[39] C. C. Gerry, P. L. Knight, *Introductory Quantum Optics.* Cambridge University Press, 2005.

[40] R. J. Barlow, *Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences.* Hoboken: Wiley, 2013.

[41] IBM Quantum Experience

https://quantum-computing.ibm.com

[42] Quantum Computing Report. *Qubit quality,* July 2020.

https://quantumcomputingreport.com/qubit-quality/

[43] Qiskit documentation, *Noise models,* June 2020.

https://qiskit.org/documentation/apidoc/aer_noise.html

Codes and Files

In the following repository, most relevant codes of this project have been deposited:

https://github.com/alvaroesalvaroo/Quantum-Artificial-Life-Files