UNIVERSIDADE DE SÃO PAULO
FACULDADE DE ECONOMIA, ADMINISTRAÇÃO,
CONTABILIDADE E ATUÁRIA

Kauê Lopes de Moraes

# The Lucas Tree Model in the Age of AI: An Agent-Based Reinforcement Learning Approach

# (Modelo de Árvore de Lucas na era da IA: uma abordagem baseada em agentes e aprendizado por reforço)

São Paulo

2023

Prof. Dr. Carlos Gilberto Carlotti Júnior
Reitor da Universidade de São Paulo

Profa. Dra. Maria Dolores Montoya Diaz
Diretora da Faculdade de Economia, Administração, Contabilidade e Atuária

Prof. Dr. Claudio Ribeiro de Lucinda
Chefe do Departamento de Economia

Prof. Dr. Mauro Rodrigues Junior
Coordenador do Programa de Pós-Graduação em Economia

**Kauê Lopes de Moraes**

# The Lucas Tree Model in the Age of AI: An Agent-Based Reinforcement Learning Approach

# (Modelo de Árvore de Lucas na era da IA: uma abordagem baseada em agentes e aprendizado por reforço)

Dissertação apresentada ao Programa de Pós-Graduação em Economia do departamento de Economia da Faculdade de Economia, Administração e Contabilidade da Universidade de São Paulo, como requisito parcial para a obtenção do título de Mestre em Ciências.

Advisor: Prof. Dr. Rodrigo De Losso Da Silveira Bueno

**Original version**

**São Paulo**

**2023**

## ACKNOWLEDGEMENTS

I want to express my deepest gratitude to my family and my girlfriend, whose unconditional support and understanding have been fundamental pillars throughout this journey. You have been the driving force behind every challenge faced and every success achieved.

I extend my sincere thanks to my friends, who not only provided me with moments of relaxation and joy but also proved to be valuable confidants and collaborators in the numerous discussions about the topics addressed in this work.

A special thanks to my advisor, whose wisdom and guidance not only shaped this work but also significantly influenced my academic and personal growth.

To all of you, my heartfelt thanks for being by my side during this important stage of my life.

# RESUMO

MORAES, K. L. **The Lucas Tree Model in the Age of AI: An Agent-Based Reinforcement Learning Approach**. 2023. 66p. Dissertação(Mestrado) - Faculdade de Economia, Administração, Contabilidade e Atuária, Universidade de São Paulo, São Paulo, 2023.

Esta dissertação explora a integração da teoria econômica com técnicas avançadas de aprendizado de máquina (aprendizado por reforço), com um foco específico na modelagem baseada em agentes (ABM) para simulação de mercados financeiros. O principal objetivo é desenvolver um ambiente de simulação de mercado de ativos, fundamental para aprofundar a compreensão das dinâmicas dos mercados financeiros. Utilizando o modelo de árvore de Lucas, esta dissertação estabelece um quadro para testar e validar as técnicas de simulação desenvolvidas, dado que o modelo tem solução analítica em alguns casos. O modelo é replicado por meio de uma abordagem baseada em agentes, criando um ambiente simulado propício para gerar os dados necessários ao treinamento de modelos de inteligência artificial. O projeto computacional desenvolvido para este estudo é caracterizado pela sua flexibilidade, permitindo a exploração de diversos cenários econômicos e o relaxamento de várias hipóteses tradicionais em modelos de macro-finanças. Esta flexibilidade é crucial, pois permite abordar cenários que são desafiadores de serem tratados com métodos analíticos tradicionais. Os resultados corroboram com a eficácia da modelagem baseada em agentes na replicação do modelo econômico clássico e na geração de dados para análises mais aprofundadas. Este trabalho não apenas oferece novas perspectivas sobre o modelo de árvore de Lucas, mas também estabelece uma base para pesquisas futuras, que podem expandir e explorar outras facetas complexas dos mercados financeiros.

**Palavras-chave**: Modelo de Árvore de Lucas. Aprendizado de Máquina. Aprendizado por Reforço. Modelo Baseado em Agentes. Economia. Precificação. Finanças.

## ABSTRACT

MORAES, K. L. **The Lucas Tree Model in the Age of AI: An Agent-Based Reinforcement Learning Approach**. 2023. 66p. Dissertation (Master) - Faculdade de Economia, Administração, Contabilidade e Atuária, Universidade de São Paulo, São Paulo, 2023.

This dissertation explores the integration of economic modeling and advanced machine learning techniques (reinforcement learning), with a specific focus on agent-based modeling (ABM) for the simulation of financial markets. The main goal is to develop an asset market simulation environment, crucial for deepening our understanding of the dynamics of financial markets. Utilizing the Lucas tree model, this research establishes a framework to test and validate the developed simulation techniques, given that the model has an analytical solution in some cases. The model is replicated through an agent-based approach, creating a simulated environment conducive to generating the necessary data for training artificial intelligence models. The computational project developed for this study is characterized by its flexibility, allowing the exploration of various economic scenarios and the relaxation of several traditional hypotheses in macro-finance models. This flexibility is crucial, as it enables the addressing of scenarios that are challenging to be dealt with using traditional analytical methods. The results corroborates with the effectiveness of agent-based modeling in replicating the classical economic model and in generating data for more in-depth analyses. This work not only offers new perspectives on the Lucas tree model but also establishes a basis for future research, which can expand and explore other complex facets of financial markets.



**Keywords**: Lucas Tree Model. Machine Learning. Reinforcement Learning. Agent-based model. Economics. Asset Pricing. Finance.

## LIST OF FIGURES

## CONTENTS

# 1 INTRODUCTION

This dissertation originated from a broader initial ambition: to develop a method for deriving, directly from data, an optimal functional form for the stochastic discount factor. This dissertation represents the first step of this ambitious journey, laying the foundation upon which future investigations will be built.

The initial approach to tackling this challenge involved applying advanced machine learning techniques to identify these functional forms within a pre-defined function space. To do so, it is essential to generate a significant volume of data, preferably prices, from a simulated economy with a known stochastic discount factor. With several such simulated data sets and knowledge of the true functional forms of the stochastic factor, it becomes feasible to train an artificial intelligence model capable of correctly estimating this functional form. The main advantage of this estimation is the ability to economically interpret the relationships between variables and understand the true weights attributed to each for the correct pricing of assets.

Therefore, the focus of this dissertation is the generation of these essential data. We chose to replicate a well-established model in the economic literature, the Lucas tree model, to validate and corroborate our results. This model serves as an ideal test case for the developed methodology, providing a solid benchmark for our simulations.

The computational project implemented here is designed to be sufficiently flexible, allowing the relaxation of various traditional assumptions in macro-finance models, which often present challenges in their formulation and analytical derivation. For instance, the implementation of different behavioral strategies and beliefs among agents. The utility and versatility of this project will become evident throughout the work.

In this initial chapter, we will explore the evolution of asset pricing models, highlighting how modern techniques, especially agent-based models, can be applied to achieve similar objectives. An introduction to agent-based modeling and a brief history of the technique will establish the context for its application in the field of finance. Finally, we will describe the specific objectives of this work, framing the results presented as an integral part of a broader research line.

## 1.1 A brief historical development of Asset Pricing Models

The evolution of asset pricing techniques mirrors the intellectual progress permeating economic theories over time. Each innovation expanded the understanding of the dynamics governing capital markets, outlining the fundamental theoretical framework for asset pricing and exerting a decisive influence on the theoretical structuring of financial

markets. After decades of study, much of the research today is unified in the language of the stochastic discount factor, and one of the main tasks is understanding its properties.

In the 1960s, the advent of the Capital Asset Pricing Model (CAPM), proposed by Sharpe (1964), marked a watershed in how financial assets were evaluated and priced. By introducing the beta coefficient (a measure of an asset's reaction to market risk), the CAPM established a linear correlation between an asset's expected return and the market risk premium, quantitatively structuring the concept of systematic risk. Extensive literature corroborates various flaws in the predictions of the one-factor model (BANZ, 1981; ROSENBERG; LANSTEIN, 1985).

The first general equilibrium models for asset pricing date back to the 1970s, (STIGLITZ, 1970), (LUCAS, 1978a), and (BREEDEN, 1979), and these continue to be used as base models in macroeconomics courses to this day. The theoretical development of these models, simple and elegant, was not corroborated by stylized facts and estimates made with real data, see Hansen and Singleton (1982), Mehra and Prescott (1985), Kocherlakota (1996), leading to a vast literature that emerged to explain the discrepancies found.

In the 1990s, the publication of Fama-French's three-factor model (FAMA; FRENCH, 1993) enriched the CAPM by considering additional variables, such as company size, the book-to-market ratio, and momentum strategy. By encompassing a variety of factors, these models elucidated more accurately the variations in returns and improved asset evaluation.

The endeavor of factors continues, and today we have the so-called "Factor Zoo" (HARVEY; LIU; ZHU, 2016), where a large set of factors individually explains part of the returns in the cross-section of assets, but together have drastically reduced explanatory power and are subject to criticism for out-of-sample errors. Even considering the case of a factor that truly has explanatory power, McLean and Pontiff (2016) argues that the publication of a study reduces the explanatory power of that factor as agents learn from the information, rendering the factor no more important than the rest of the previous factors.

With the popularization of various artificial intelligence techniques (ATHEY, 2018), research in economics, more specifically in finance, is equipped with a tool that allows modeling and finding complex patterns in large volumes of data. In the second decade of this century, machine learning techniques were primarily used from a predictive perspective, without delving into deeper questions and implications of economic theory, sometimes doubling the performance of econometric techniques based on regressions (GU; KELLY; XIU, 2020). These techniques, mostly pruned of economic interpretation, attempt to find highly non-linear combinations of variables that are capable of minimizing/maximizing some metric, traditionally in finance, the expected return.

While classical econometric methodologies have been the foundation of financial

theory and have guided asset pricing practices over the years, the increasing complexity of financial markets and their interconnectedness have spurred the search for new approaches. Tools such as artificial intelligence, simulations, and complex network analysis have been explored to holistically capture market patterns and interactions, aiming to enhance not only the pricing projections of assets but also the theory itself. In recent years, machine learning techniques have gained popularity for addressing more theoretical problems in economics (ATHEY; IMBENS, 2019), particularly those related to causality (CHERNOZHUKOV *et al.*, 2018; SCHÖLKOPF, 2022; PROSPERI *et al.*, 2020).

Within the context of simulations, Agent-Based Modeling (ABM) stands out as an alternative to traditional analytical models. Starting from a purely microeconomic approach, we can simulate the behavior and learning of individual agents, which allows us to analyze patterns in macroeconomic aggregates and understand how individual decisions can lead to surprising macroeconomic consequences in highly nonlinear environments, the so-called emergent effects.

## 1.2 Agent-Based Modeling (ABM)

The origins of Agent-Based Modeling techniques can be traced back to the pioneering work of John von Neumann, one of the fathers of modern computing. Neumann (1966) introduced the concept of cellular automata, proposing systems where cells on a grid evolve according to defined rules. This idea laid the groundwork for later models that would use interactive agents in complex networks.

A significant milestone in the development of ABM came with John Conway and his famous "Game of Life" in the 1970s (GARDNER, 1970). The "Game of Life" is a classic example of a cellular automaton, where complex patterns emerge from simple rules. This game demonstrated how complex and unpredictable behaviors could arise from local interactions between agents, establishing a fundamental paradigm for agent-based modeling.

In the field of economics, one of the first and most influential agent-based models was developed by Schelling (1971). Schelling's segregation model illustrates how racial segregation can emerge in a city even when individuals have a moderate preference for neighbors of their own race. This model was crucial in demonstrating the utility of ABM in explaining complex social and economic phenomena, highlighting how the interactions of individual agents can lead to emergent macroscopic patterns.

Since then, agent-based modeling has been applied in a variety of disciplines, including economics, sociology, political science, biology, and ecology (STEINBACHER *et al.*, 2021). In economics, in particular, ABM has been used to study markets, simulate entire economies, and understand the dynamics of financial crises, among other topics

(DOSI *et al.*, 2020).

This approach has gained momentum with the increase in computational capacity and the availability of large data sets, enabling the simulation of complex systems with an ever-increasing number of agents and more detailed interactions. Agent-based modeling continues to evolve, offering valuable insights in many fields of knowledge and challenging our understanding of complex systems (MONTI *et al.*, 2023).

An agent-based model is a computational approach to simulating the actions and interactions of autonomous entities, termed "agents," in a shared environment. Each agent is characterized by a set of attributes, behavior rules, and decision-making capabilities. These characteristics can be fixed, adaptive, or learned over time. The model is defined by the following components:

- Agents: Individual entities with distinct capacities for perception, cognition, and action. They can represent individuals, groups, organizations, or any other decision-making entity. Agents are essentially defined by their behavior rules, or how they react and make decisions in response to their perceptions and the state of the environment. These rules can be deterministic or stochastic and can be based on bounded rationality, heuristics, or any other form of decision-making.

- Environment: The context in which agents operate and interact. This can include spatial features (like a geographic map or a social network), as well as environmental factors that influence agent behavior.

- Interactions: Describes how agents communicate, compete, collaborate, or influence each other. Interactions can be direct or mediated through the environment.

The evolution of the system is generally modeled in discrete steps, but it can also be continuous. At each step or time interval, agents assess their environment, make decisions, and act according to their behavior rules.

The emergent dynamics of the model result from the complex interaction between agents and their environment. Unlike traditional models based on equations of motion, ABMs allow the observation of emergent phenomena that arise from the interaction of simple rules at the agent level but can lead to complex behaviors at the system level.

## 1.3 ABM in Finance

Agent-Based Modeling (ABM) techniques have special advantages in simulating financial markets, analyzing how the decisions of different types of investors influence the formation of asset prices. By construction, this type of modeling more easily accommodates

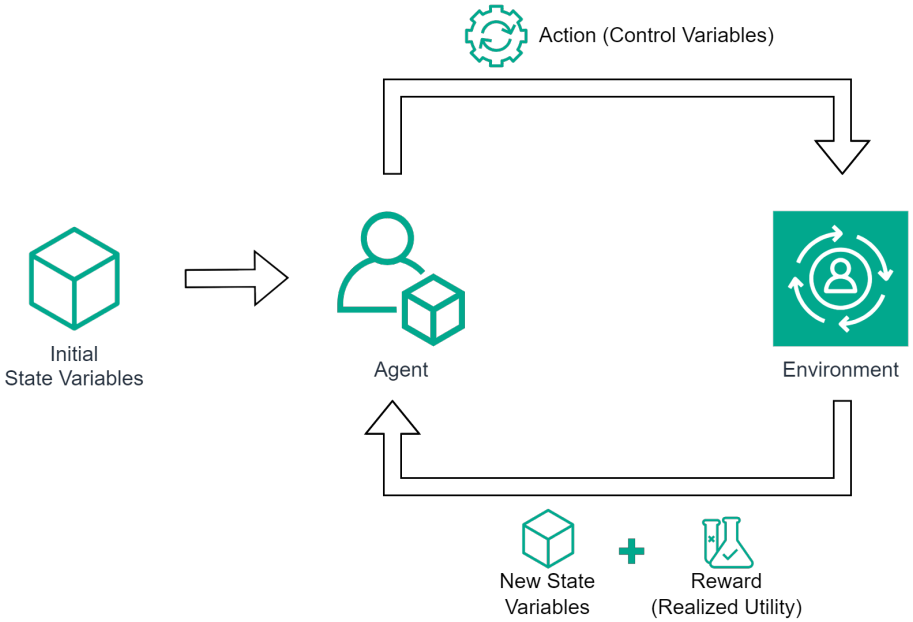


Figure 1 – General diagram of an Agent-Based Model, illustrating the flow for a single agent. This diagram depicts the various stages and interactions a single agent undergoes within the model. It includes the agent's decision-making process, interaction with the environment, response to external stimuli, and the subsequent loop of actions taken.

heterogeneous agents and more complex structures/asymmetries of information, which are very important, for example, to understand the dynamics of market pricing.

These models have been successful in studying the origins and development of financial crises (VASELLINI, 2023). By modeling interactions among entities such as banks, businesses, and consumers, ABM helps to illustrate critical factors, like market panic, non-dissipative idiosyncratic shocks, and financial interconnections, which contribute to financial instability and crises.

In this context, the Lucas tree model, traditionally used in finance, serves as an interesting backdrop for the application of agent-based modeling. The tree model, proposed by Robert Lucas in 1978, is a general equilibrium model in which agents make decisions about consumption and investment based on their expectations about the future dividends of an asset. In its original form, the model assumes homogeneity among agents and rational expectations. However, by introducing an ABM perspective, it becomes possible to explore how heterogeneity in agents' beliefs, strategies, and information limitations can influence asset pricing, market dynamics, and the formation of bubbles and crashes.

This combined approach not only enriches the understanding of financial market behavior but also opens doors to investigate how different policies and interventions might influence market outcomes in a world where agents are not always fully rational and where their complex interactions can lead to unexpected dynamics.

## 1.4 Objectives

Considering the historical and conceptual importance of the Lucas Tree Model, this study adopts it as the baseline model. The choice of this model is strategic, given its recognition and familiarity in the field of finantial economics. The intention is to use a well-established model as a starting point for introducing more advanced techniques, which are less familiar to economists, thereby creating a didactic bridge between traditional economic theories and modern methodologies. The objectives of this work are as follows:

- **General Objective:** To replicate the classic results of the Lucas Tree Model using an agent-based model and Reinforcement Learning techniques.

The objective aims to demonstrate the effectiveness of contemporary modeling approaches in the context of established economic theory, paving the way for future work that uses the model constructed here, diverging and relaxing hypotheses that are traditionally complicated to resolve analytically.

To achieve this objective, it is necessary to familiarize the reader with techniques that, although well-known in the field of computer science, have only recently been incorporated into economic literature. These fundamental aspects guide the specific objectives of the study:

1. To introduce the concept of Neural Networks as Function Approximators, establishing their relevance and applicability.
2. To explore the nuances of a Reinforcement Learning technique, connecting it with the traditional approach adopted in the recursive solution of macroeconomic models.
3. To implement the Lucas Tree Model in an agent-based context, demonstrating the feasibility and functionality of this approach.
4. To simulate the agent-based Lucas Tree Model, where agents seek to maximize their returns through Reinforcement Learning techniques, and To compare the equilibrium results obtained with the predictions of classic models, thus providing a critical evaluation of the new model solution proposed.

# 2 DEVELOPMENT

This chapter is dedicated to presenting essential technical concepts for understanding the proposed computational solution. Initially, we revisit the core aspects of the Lucas Tree Model, elucidating conventional methodologies for determining equilibrium prices. This review is crucial to establish a contrast between traditional approaches, commonly explored in macroeconomics courses, and the machine learning method that will be discussed.

Moving to a still-evolving sphere of economic modeling, which has been gaining prominence in recent publications, we outline the Agent-Based Model inspired by the seminal work of the Lucas Tree. Furthermore, we introduce the application of Q-Learning, a technique that, while sharing roots with classic dynamic programming tools, represents a foray into the realm of reinforcement learning.

Concluding, we propose an analogy between neural networks and the familiar concept of Linear Regression, which eases the transition to comprehending Deep Q-Networks, a natural evolution of Q-Learning. This dialogue between advanced machine learning techniques and traditional methods of economic analysis is the guiding thread that permeates the discussion proposed in this chapter.

## 2.1 Lucas Tree General Equilibrium Model

Before proceeding to solve the Lucas Tree Model in an agent-based format, it is essential to revisit the basic structure of the model as proposed in the seminal paper "Asset Prices in an Exchange Economy" (LUCAS, 1978b).

### 2.1.1 Brief Description of the Model

Consider a discrete-time endowment economy, denoted by $t \in 0, 1, 2, \ldots$. A continuum of identical agents, uniformly distributed over the interval $[0, 1]$. Each agent is endowed, at the initial moment $t = 0$, with a tree, which is infinitely divisible and can be traded among agents.

The trees in question represent assets that generate a "fruit" or a dividend $y_t \in \mathbb{R}$ at the beginning of each period $t$. This dividend must be entirely consumed within the period it is produced. The sequence of dividends $y_t$ is modeled as a Markov stochastic process, characterized by the following conditional probability distribution function:

$$f(y_{t+1}|y_t) \tag{2.1}$$

### 2.1.2 Assets in the Economy

In this economic context, two types of assets are identified: a risk-free asset, denoted as $b$, and a variable return asset, $s$, which symbolizes the share in the fruits generated by specific trees.

- Risk-Free Asset: $b_{t+1}$

  The risk-free asset, represented by $b_{t+1}$, guarantees a fixed payment of one unit of consumption in the subsequent period, regardless of the state of the economy. The symbol $b_{t+1}$ indicates the amount of risk-free assets acquired at time $t$ and maturing at time $t+1$. The price per unit of this asset at time $t$ is denoted by $q_t$.

- Tree Shares $s$:

  The tree shares, or $s_t$, refer to the volume of shares that generate a dividend flow at the beginning of each period $t$, equivalent to $s_t y_t$ units of consumption, which can be reinvested. The unit price of tree shares at time $t$ is represented by $p_t$.

Agents in this economy face the decision of trading immediate consumption for assets that promise a share of future dividends from the trees, implying a potentially uncertain future consumption flow. Therefore, within their budgetary constraints, agents must weigh the amount of present consumption against future consumption.

### 2.1.3 Utility of the Representative Agent

Consider that the representative agent has instant utility equal to $u(c_t)$, such that $u'(c_t) > 0 \; \forall \; c_t$, so that the preference of the representative agent $\mathbb{U}$ at $t = 0$ is modeled as follows:

$$\mathbb{U} = \mathbb{E}_0 \left[ \sum_{t=0}^{\infty} \beta^t u(c_t) \right]$$

Where $\mathbb{E}_0$ is the expectation considering the set of information available at $t = 0$, and $\beta$ is a discount factor used by the Agent such that $0 < \beta < 1$.

### 2.1.4 Agent's Budget Constraint

For an economic agent, it is essential that their consumption and investment choices align with their financial limitations. In any period $t$, an agent must obey the following budget constraint:

$$c_t + p_t(s_{t+1} - s_t) + q_t b_{t+1} \leq b_t + s_t y_t \tag{2.2}$$

And for this purpose, we can consider the values for the state variables at the initial time as known.

### 2.1.5 The Problem of the Representative Agent

Given the description of the model, the optimization problem faced by the representative agent can be formulated as follows:

$$\max_{c_t, s_{t+1}, b_{t+1}} \mathbb{E}_0 \left[ \sum_{t=0}^{\infty} \beta^t u(c_t) \right]$$
$$st. \quad c_t + p_t(s_{t+1} - s_t) + q_t b_{t+1} \leq b_t + s_t y_t$$

The objective function represents the expected present value of utility from consumption over time, where $\beta$ is the discount factor, and $u(c_t)$ is the utility function of consumption in period $t$. The budget constraint, already described earlier, ensures that the agent does not spend more than their available resources in each period.

To establish the starting point of the agents' trajectories, the following initial conditions are assumed for the state variables:

$$s_0 = 1, \quad b_0 = 0, \quad y_0 = \bar{y} \tag{2.3}$$

These conditions establish that each agent starts with one unit of the uncertain return asset ($s_0$), no risk assets ($b_0$), and a known and constant initial yield for the tree ($y_0 = \bar{y}$).

### 2.1.6 Recursive Formulation of the Representative Agent's Problem

To recursively formulate the problem of the representative agent, we first observe that the distribution of dividends $y_t$ generated by a Lucas tree follows a Markov process as defined in equation (2.1). This implies that the current state variable $y_t$ is statistically sufficient to predict $y_{t+1}$, eliminating the need to consider the history of dividends. Thus, $y_t$ is established as the only state variable necessary for the dividend series.

Furthermore, we enter period $t$ with knowledge of the inherited quantities of tree shares $s_t$ and risk-free assets $b_t$, which are also treated as state variables.

At the end of each period, the agent makes decisions, respecting the budget constraint, on how much to consume $c_t$, and how much to invest in new tree shares $s_{t+1}$ or safe assets $b_{t+1}$.

The recursive problem of the representative agent can then be expressed by the following Bellman equation:

$$V(s_t, b_t, y_t) = \max_{c_t, s_{t+1}, b_{t+1}} u(c_t) + \beta \mathbb{E}\Big[V(s_{t+1}, b_{t+1}, y_t)|y_t\Big] \\ st. \quad c_t + p_t(s_{t+1} - s_t) + q_t b_{t+1} \leq b_t + s_t y_t \tag{2.4}$$

In certain solution approaches, it is useful to incorporate the budget constraint directly into the objective function, resulting in the following alternative form of the Bellman equation:

$$V(s_t, b_t, y_t) = \max_{s_{t+1}, b_{t+1}} u(b_t + s_t y_t - p_t(s_{t+1} - s_t) - q_t b_{t+1}) + \beta \mathbb{E}\Big[V(s_{t+1}, b_{t+1}, y_{t+1})|y_t\Big] \tag{2.5}$$

## 2.2 Traditional Solution of the Model

This section explores the traditional approach to solving the Lucas model, which will serve as a basis for the method presented in the next section.

### 2.2.1 Analytical Solution for the Model's Equilibrium Price

By deriving with respect to the control variables, we obtain the first-order conditions:

$$u'(c_t)p(y_t) = \beta \mathbb{E}\Big[V_1(s_{t+1}, b_{t+1}, y_{t+1})|y_t\Big] \qquad (s_{t+1})$$
$$u'(c_t)q(y_t) = \beta \mathbb{E}\Big[V_2(s_{t+1}, b_{t+1}, y_{t+1})|y_t\Big] \qquad (b_{t+1})$$

Here, $V_i(s_{t+1}, b_{t+1}, y_{t+1})$ represents the partial derivative of the function $V$ with respect to the i-th argument. Using the envelope theorem, we can calculate the derivatives of the value function as follows:

$$V_1(s_t, b_t, y_t) = u'(c_t)(p(y_t) + y_t)$$
$$V_2(s_t, b_t, y_t) = u'(c_t)$$

Assuming that the good produced by the Lucas tree is non-storable, we have that in equilibrium $c_t = y_t$. Advancing one period and substituting in the first-order conditions, we obtain:

$$u'(y_t)p(y_t) = \beta \mathbb{E}\Big[u'(y_{t+1})(p(y_{t+1}) + y_{t+1}))|y_t\Big] \tag{2.6}$$
$$u'(y_t)q(y_t) = \beta \mathbb{E}\Big[u'(y_{t+1})|y_t\Big] \tag{2.7}$$

Solving for prices, we define the stochastic discount factor, $m_{t+1}$, of the Lucas Tree Model:

$$
\begin{aligned}
m_{t+1} &= \frac{u'(y_{t+1})}{u'(y_t)} \\
p(y_t) &= \beta \mathbb{E}\Big[m_{t+1}(p(y_{t+1}) + y_{t+1}))|y_t\Big] \\
q(y_t) &= \beta \mathbb{E}\Big[m_{t+1}|y_t\Big]
\end{aligned}
$$

The first-order conditions (2.6) reflect the marginal equality of the cost and benefit of acquiring additional assets, whether in the form of shares ($s_{t+1}$) or risk-free bonds ($b_{t+1}$). The marginal utility function $u'(c_t)$ is weighed against the prices of assets ($p(y_t)$ and $q(y_t)$), and equals the expected discounted value of tomorrow's marginal benefit. This correspondence highlights the intertemporal trade-off faced by economic agents: the pleasure of present consumption versus the benefit of future consumption provided by investment.

The derivatives of the value function, obtained through the envelope theorem, show that the marginal value of acquiring more of the uncertain asset depends not only on the current price and dividend but also on consumption. That is, there is a direct relationship between consumption and the valuation of the asset, underscoring the importance of intertemporal consumption preference.

By establishing that consumption in equilibrium equals the dividends ($c_t = y_t$), we are assuming that all the income generated is consumed, reflecting the hypothesis of the non-storability of the good. This simplifies the analysis by avoiding the need to consider optimal storage strategies, which would add another layer of complexity to the problem.

Finally, asset prices are determined by the stochastic discount factor $m_{t+1}$, which reflects the intertemporal rate of substitution of consumption. The price of the Lucas asset ($p(y_t)$) is the expected present value of future dividends plus the appreciation of the price, adjusted by the stochastic discount. On the other hand, the price of the risk-free asset ($q(y_t)$) reflects only the future expectations adjusted by the same discount factor.

These equations encapsulate the essence of the Lucas Tree Model, a cornerstone in modern financial theory. They illustrate how asset prices are influenced by the rational expectations of investors and intertemporal consumption preferences, providing a clear mechanism for asset valuation in a dynamic and uncertain environment.

### 2.2.2 Numerical Iterative Solution of the Bellman Equation

The solution presented here is resolved for a grid of values previously chosen by the economist, usually around the long-term stationary equilibrium. When setting up this grid, we encounter the first problem of the method: the choice of position, size, and precision (number of points in the same space). It's worth noting that the number of points in this grid includes the problem of the curse of dimensionality for this method.

Our goal now is to determine the functional form of the value function associated with the Bellman equation of the problem (2.5). Once this functional form is found, we will be close to completely solving the problem. This requires solving an optimization problem for the control variables to find:

$$
\begin{aligned}
s_{t+1} &= g(s_t, b_t, y_t) \\
b_{t+1} &= h(s_t, b_t, y_t)
\end{aligned}
$$

where $y_t$ is realized at the beginning of each period. With the initial values $s_0$, $b_0$, and $y_0$ given, we can solve the problem recursively, obtaining the optimal control sequences $\{s_i\}_{i=1}^{\infty}$ and $\{b_i\}_{i=1}^{\infty}$.

To derive the functional form of the value function, we will use the algorithm developed by Howard (HOWARD, 1960) in his doctoral thesis. The main points of this method will be highlighted, emphasizing the aspects relevant to the implementation proposed in this work. Complete details of the implementation and proof of convergence of the algorithm to the optimal solution are presented by Ljungqvist and Sargent (2018) in Appendix A of *Recursive Macroeconomics*.

To facilitate the fixed-point procedure, we start from **??** and consider the linearity of the expectation operator:

$$
\begin{aligned}
u'(y_t)p(y_t) &= \beta\mathbb{E}\Big[u'(y_{t+1})(p(y_{t+1}) + y_{t+1})|y_t\Big] \\
u'(y_t)p(y_t) &= \beta\mathbb{E}\Big[u'(y_{t+1})p(y_{t+1})|y_t\Big] + \beta\mathbb{E}\Big[u'(y_{t+1})y_{t+1}|y_t\Big]
\end{aligned}
$$

We define functions $f$ and $r$ as aids for the iterative procedure, so we can later recover the original functions.

$$
\begin{aligned}
f(y_t) &= u'(y_t)p(y_t) \\
r(y_{t+1}) &= \beta\mathbb{E}\Big[u'(y_{t+1})y_{t+1}|y_t\Big]
\end{aligned}
$$

By making these substitutions, we have:

$$f(y_t) = r(y_{t+1}) + \beta\mathbb{E}\Big[f(y_{t+1})|y_t\Big]$$

Note that we define the value $r(y_{t+1})$ because it remains constant throughout the iterations of the function $f(y_t)$, since $y_{t+1}$ has a known distribution at time $t$. Therefore, we can calculate its expectation and make the iterative calculation more efficient. Let's introduce the operator $T$ of the problem, such that:

$$(Tf)(y_t) = r(y_{t+1}) + \beta\mathbb{E}\Big[f(y_{t+1})|y_t\Big]$$

So that the solution $f^*(y_t)$ is a fixed point of T, meaning $Tf^*(y_t) = f^*(y_t)$. As previously indicated, in Appendix A of Recursive Macroeconomics, we find the proof for the following assertions:

- $T$ has only one fixed point.
- For any $f_0$ that respects the constraints of the problem, the sequence $T^k f_{0_k}$ converges uniformly to $f^*$, the fixed point of the operator $T$.
- The fixed point of $T$ is the true value function of the problem.

This allows us to finally state the following procedure:

**Algorithm 1** Policy/Howard Improvement Algorithm

$u_0(s_0, b_0, y_0) \leftarrow$ Random Feasible Policy
$i \leftarrow 0$
$N \leftarrow n$
**while** $i \leq N$ **do**
  $V_{u_i}(x) \leftarrow \sum_{t=0}^{\infty} \beta^t r(x, u_i(x))$
  $u_{i+1} \leftarrow \max_u r(x, u) + \beta V_{u_i}[g(x, u)]$
  $i \leftarrow i + 1$
**end while**

With this, given a sufficiently large number of iterations, we ensure that the value function will converge, within an *epsilon* $> 0$, to the real value.

It is worth noting that this solution is only possible when we specify the function $r(x, u)$ of the problem correctly. Since the model developed earlier is fully defined, we do not encounter any problems here.

### 2.2.3 Benveniste-Scheinkman Theorem

The iterative solution previously discussed requires iterating over a large set of points using the algorithm provided, a task difficult to perform manually. Therefore, we have an alternative solution that allows us to solve the problem using derivatives, which in a way, simplifies the calculations elegantly.

The Benveniste-Scheinkman Theorem establishes conditions under which the value function of an optimal control problem is differentiable. Furthermore, it specifies how the derivatives of the value function can be used to characterize the optimal policies of agents. The details and necessary conditions for applying this theorem can be found in the reference (BENVENISTE; SCHEINKMAN, 1979).

In the context of the Lucas Tree Model, we apply this theorem to the value function $V(s_t, b_t, y_t)$ defined in the recursive formulation of the representative agent's problem (2.5). According to the theorem, if the value function is sufficiently smooth and the envelope conditions are met, the partial derivatives of the value function with respect to the state variables reveal the shadow prices of these variables.

This approach, using the Benveniste-Scheinkman Theorem, allows us to follow a process similar to the one described earlier. However, instead of iterating over a grid of points, we iterate over functions. This method offers a more efficient and elegant way of finding solutions for the model, avoiding the computational complexities associated with iterating over extensive grids and allowing a more direct and analytical analysis of the optimal policies.

### 2.2.4 Critiques

The model proposed by Lucas has provided valuable insights into expectation formation and intertemporal decision-making, yet it has also given rise to various "puzzles" that have perplexed economists for decades. These puzzles, or observed inconsistencies between the model's theoretical predictions and actual empirical data, have spurred a vast literature that seeks not only to elucidate them but also to propose extensions and reformulations of the original model.

In particular, a significant point of contention concerns the assumption of rational expectations, which is central to Lucas's model. Critics argue that this assumption might be overly restrictive, overlooking the possibility of behavior constrained by bounded rationality or influenced by psychological biases (SELTEN, 1990; KAHNEMAN; TVERSKY, 1979). Additionally, the adequacy of the hypothesis of markets always being in equilibrium has been questioned, suggesting that out-of-equilibrium market dynamics might provide additional insights into phenomena such as financial crises and economic fluctuations (STIGLITZ, 2011).

Acknowledging these challenges, various research lines have been dedicated to exploring approaches that relax some of the more stringent constraints of the model (WEIL, 1989). Here, I highlight the literature on heterogeneous agent models that allow for variations in the expectations and behaviors of individuals, offering a more nuanced and potentially more realistic representation of the economy (DAVID, 2008; EBRAHIM; MATHUR, 2001).

The next section will address one of these methodologies, which relies on the idea of agent-based modeling. In this approach, we forgo the aggregation of information and simulate each agent individually. This type of approach facilitates the implementation of expectations that are not fully rational but are still systematic and predictable. This approach represents an attempt to reconcile theoretical predictions with empirical observations while maintaining consistency with fundamental behavioral principles observed in economic time series data (KAHNEMAN, 2003).

## 2.3 Agent-Based Lucas Tree Model

As previously discussed, an Agent-Based Model (ABM) revolves around three key components: Agents, Environment, and Interactions among them. Within the Lucas tree model framework, agents are conceptualized as economic entities endowed with resources, engaging in a market to set prices for financial assets. Each agent is characterized by its initial endowment and a set of potential strategies for decision-making regarding the buying and selling of assets.

The market, constituting the Environment in our ABM, is the stage where transactions take place. It is here that the 'matching' between buyers and sellers is facilitated, following predefined rules that simulate market conditions. The functioning of this environment is crucial for understanding market dynamics and the fluctuation of asset prices.

The agent-market interaction is outlined by the set of rules defining how agents access the market and conduct transactions. These rules form the backbone of the model, as they determine how information is shared, decisions are made, and prices are adjusted over time.

Contrasting with the general equilibrium approach of the initial model, the ABM modeling described here focuses on discrete episodes of interaction between agents and the market. Each episode can be viewed as a complete microcosm, where agents undergo multiple iterations, refining their strategies based on transaction history. This process is analogous to trial-and-error learning, reminiscent of the concept explored in the movie "Groundhog Day" (1993), where the repetition of events allows individuals to optimize their decisions based on previous experiences.

These trajectories of interactions are crucial as they provide agents with the opportunity to adjust their decision-making policies for control variables, considering the current state of the market. Thus, the ABM incorporates a learning component, essential for the evolution of agent strategies and the emergence of complex market patterns.

We begin the formal description of the model parts with Agents and proceed with the Market. Since agents interact only through the Market entity, describing the functioning of the market suffices to complete the mechanics of the model.

### 2.3.1 Agents

Each agent, indexed as $k$ at iteration $t$, denoted as $A_t^k$, is fully characterized by its state variables, control variables, decision policy, utility function, and idiosyncratic variables.

1. State Variables:
   - $s_t^k$ - Quantity of risky asset owned by agent $k$
   - $b_t^k$ - Quantity of risk-free asset owned by agent $k$
   - $y_t^k$ - Income of agent $k$ at iteration $t$
2. Control Variables:
   - $p_t^k$ - Price of the risky asset set by agent $k$
   - $q_t^k$ - Price of the risk-free asset set by agent $k$
3. Decision Policy:
   - $\pi^k : \quad (s_t^k, b_t^k, y_t^k) \mapsto (p_t^k, q_t^k)$ Maps the state variables to the control variables
4. Utility Function:
   - $u^k(c)$ Instantaneous utility of agent $k$
   - $U$ Overall utility of agent $k$

$$U^k = \mathbb{E}_0 \left[ \sum_{t=0}^{\infty} \beta^t u^k(c_t^k) \right]$$

5. Idiosyncratic Variables:
   - $\underline{x}^k$ - A vector of idiosyncratic variables for agent

In this study, we investigate a representative agent model; hence, idiosyncratic variables do not exist, leading to $\underline{x}^k = \{\}$.

In summary, the agents' role in the model can be outlined as follows: Based on the observation of state variables and market dynamics, agents choose the prices of assets and the amount of consumption that maximize their expected discounted utility.

### 2.3.2 Market (Environment)

The market, in our Agent-Based Model (ABM) of the Lucas Tree, serves as the central mechanism for allocating buy and sell intentions among agents, reallocating asset quantities, calculating utilities, and simulating dividend processes while maintaining the model's constraints. For instance, the market can enforce no-Ponzi conditions by penalizing agents attempting to offer negative quantities in the market, thereby disincentivizing such choices in future simulation episodes.

The market is fully characterized by the following elements:

1. $X_t$ - Vector representing the environmental state variables.
2. $\{A_t^k\}$ - The set of $N$ agents participating in the market at time $t$.
3. Update Rule
    - $\Lambda: \quad (\{A_t^k\}, X_t) \mapsto (\{A_{t+1}^k\}, X_{t+1})$ - A function mapping the current state of agents and environment to their next state

Several mechanics can be explored in the allocation of agents' offers and demands, which can include factors like bounded rationality, monopoly power, negotiation skills, etc. For this study, a straightforward mechanism for executing trades is chosen, defined as follows:

**Definition: Possível Comprador**

Given any two agents $A^1$ and $A^2$, agent $A^2$ is defined as a potential buyer of $A^1$ at price $\bar{p}$ in iteration $t$ if and only if:

- The agents set prices such that $p_t^1 \geq \bar{p} \geq p_t^2$ for the asset $s$.
- Agent $A^1$ has stock $s_t^i > \Delta s$
- Agent $A_2$ has funds $y_t^i > \Delta s \cdot \bar{p}$
- Neither agent has engaged in a trade during iteration $t$

Where $\Delta s$ represents the quantity of asset $s$ exchanged in the event of a market match.

The decision mechanism for the settlement price $\bar{p}$ between two agents can be explored in various ways, thus affecting the equilibrium outcomes of the model. One approach could be to define a bargaining power variable for each agent, influencing the price setting during matches.

The Market's responsibilities include:

- Conducting matches between agents who are potential buyers and sellers.
- Determining the settlement price $\bar{p}$ for each transaction.
- Intermediates the exchange of goods and income between matched agents.

### 2.3.3 Simulation of an Episode

With the agents and the market (environment) established, we outline the process of a simulation episode within our model. An episode corresponds to a complete realization of the proposed model. Given the infinite horizon of interactions (as indicated by the summation extending to infinity in the expected utility function of the consumer), a truncation mechanism is implemented to make simulations computationally feasible.

The Figure 2 illustrates the sequence of events for a single agent throughout an episode. Initially, at iteration $t = 0$, the agent is characterized by its initial state variables $(s_0, b_0, y_0)$. Based on this state, the agent employs its policy $\pi$ to determine the values of the control variables $(p_i, q_i, c_i)$, which are then transmitted to the environment.

The environment processes the decisions of all agents, executes the "matching" algorithms for transactions, and distributes the new state variables $(s_{i+1}, b_{i+1i}, y_{i+1})$ back to the agents, along with the calculation of the instant utility gained by each one.

Interestingly, the calculation of instant utility, or "Reward", is conducted by the market instead of individually by the agents. This convention, derived from Reinforcement Learning practices, allows for the application of negative reinforcements to deter undesirable behaviors without needing to explicitly restrict the actions available to agents. When an agent performs an improper action, the market imposes a penalty, encouraging the learning of allowed and productive strategies.

An agent-based model introduces additional dimensions of freedom, allowing for more complex and individualized interactions compared to aggregated models. However, this richness in detail comes with the cost of increased computational demand.

A key aspect of the model in question is the decision-making process of the agents, namely, how they select control variables to maximize their discounted utility over time. A conventional approach to solve this problem involves constructing a “grid” of possible actions and states, where agents iteratively refine their estimates until convergence is achieved.

However, the solution proposed in this work diverges from this traditional approach. Instead, agents use the history of previous episodes to update their action policies. This method has the advantage of not requiring the exhaustive execution of all possible state combinations to achieve convergence, thereby reducing computational costs and speeding up the optimization process.

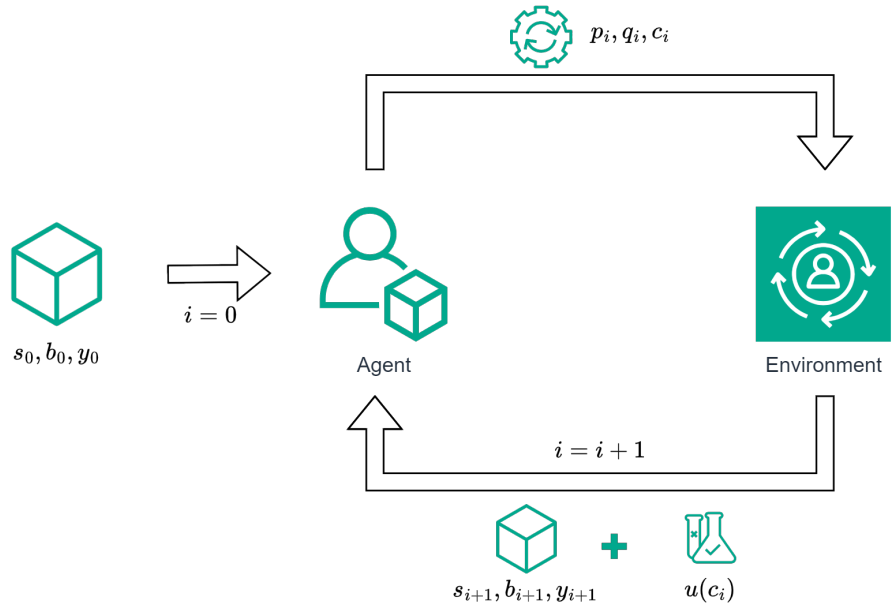


Figure 2 – Representation of an episode from the perspective of a single agent.

In the illustrated approach, each agent undergoes a series of interactions within the market environment, making strategic decisions based on both the current state and the accumulated experience from past episodes. This iterative learning and adaptation process not only enhances the realism of the simulation but also allows for the emergence of complex market dynamics that can provide deeper insights into economic behaviors and phenomena.

## 2.4 Q-Learning

The technique of Q-Learning, a form of Reinforcement Learning, has made significant strides in a wide array of complex applications, ranging from video games to advanced economic models. This technique gained notable prominence following the publication of the seminal paper "Playing Atari with Deep Reinforcement Learning" (MNIH *et al.*, 2013). Its relevance and impact were further bolstered in 2023 with the first publication of a work in a "Top 5" economics journal employing the Q-Learning methodology (JOHNSON; RHODES; WILDENBEEST, 2023), marking a significant advancement in the adoption of this approach in rigorous economic studies.

The concept of Q-Learning was introduced by Christopher Watkins in his 1989 doctoral thesis. This approach was groundbreaking for enabling agents to learn to maximize

future rewards based on iteratively updated estimates, eliminating the need for a model of the environment. This opened the door to applying reinforcement learning in problems where the environment model is unknown or too complex to be explicitly modeled.

Q-Learning is an example of Temporal Difference (TD) methods, which learn directly from raw experience without waiting for a final outcome (SUTTON; BARTO, 2018). This differs from other reinforcement learning techniques that require knowledge of the state transition model or that learn from the complete trajectory of the state to the end.

The introduction of Q-Learning marked a significant advance in the field of artificial intelligence, allowing for a more flexible and robust approach to automated learning. It was only with the advent of computational advancements and the introduction of Deep Q-Networks (DQN) that the full potential of Q-Learning was realized. The combination of Q-Learning with neural networks, known as Deep Q-Network (DQN), was first demonstrated by DeepMind in 2013, playing Atari games directly from screen pixels. This showed that the technique could outperform humans in many of these games (HESTER *et al.*, 2018), dealing with high levels of perception and control, providing a strong argument for using the same technique to simulate economic agents.

Before introducing the solution presented in the paper "Playing Atari with Deep Reinforcement Learning" (MNIH *et al.*, 2013), I will present some concepts that will serve as a basis for the final application, also taking advantage of using the terminology of Reinforcement Learning.

### 2.4.1 Quality Function

The Quality Function $Q(x, a)$ plays a crucial role in Reinforcement Learning, evaluating the expected value of being in a state $x$ (state variables) and taking an action $a$ (chosen control variables). The Quality Function $Q$ is used to estimate the value of each action from a specific state, and it will be the function estimated by the agents, with improved estimates throughout the model's training. This function is essential for guiding the agent in their decision-making, aiming to optimize future rewards, and is expressed as follows:

$$\begin{aligned} Q(x_t, a_t) &= \mathbb{E}\Big[r(x_t, a_t) + \beta V(x_{t+1})\Big] \\ &= \sum_{x_{t+1}} P(x_{t+1}|x_t, a_t)[r(x_t, x_{t+1}, a_t) + \beta V(x_{t+1})] \end{aligned}$$

where $r(x_t, a_t)$ is the reward received for taking action $a_t$ in state $x_t$, $\beta$ is the discount factor that weighs the importance of future rewards, $P(x_{t+1} \mid x_t, a_t)$ is the probability of transitioning to state $x_{t+1}$ given the current state $x_t$ and the action taken

$a_t$, and the term $V(x_{t+1})$ represents the maximum value of the Quality Function for the next state, the value function of the problem. This formulation of the Quality Function $Q$ incorporates both the immediate reward and the optimized value of future actions, forming the basis of the optimal policy that the agent seeks to find.

$$\pi(x_t) = \underset{a}{\operatorname{argmax}}\, Q(x_t, a) \tag{2.8}$$

$$V(x_t) = \max_a Q(x, a) \tag{2.9}$$

Note that, if we had the true Quality Function, in any possible state we could choose the optimal trajectory, and for that, it would be enough to solve the maximization problem (2.9), to be able to choose the best action (control variables).

### 2.4.2 Q-Learning Algorithm

Equipped with the concept of the Quality Function, we can state the algorithm responsible for the convergence of a random Quality Function $Q^0$ to its true value:

1. We start the algorithm by setting up a grid that represents the values for the random Quality Function $Q^0$, varying for each possible state variable and control variable. Since $x_0$ is known (initial state variables) and we set $i = 0$.

2. The agent will execute in the Lucas environment a sequence of $k$ iterations, in such a way as to choose the action $a_k$ by solving:

   $$a_k = \underset{a}{\operatorname{argmax}}\, Q^i(x_k, a)$$

   At the end of this stage, we observe the series:

   - $\{x_i\}_0^{k+1}$ - Sequence of States Observed by the Agent
   - $\{a_i\}_0^k$ - Sequence of Actions Taken by the Agent (Chosen Control Variables)

3. Given the sequences of control and state variables, we can calculate the realized values of $Q$, and thus update our function $Q^{i+1}$ in the following way:

   $$Q^{i+1}(x_k, a_k) = (1-\alpha)Q^i(x_k, a_k) + \alpha\Big[r_k + \beta \underset{a}{\max} Q(x_{k+1}, a)\Big] \tag{2.10}$$

   Here, $\alpha$ is the learning rate chosen for the problem, in such a way as to update the estimate of the Quality Function through a weighted average of the old value $Q^i$ and the value obtained directly from the simulation. The choice of this parameter is discussed in detail by Even-Dar and Mansour (2004).

4. We take $i = i + 1$ and return to step 2 until some convergence criterion is satisfied.

### 2.4.3 Convergence of Q-Learning

Similar to the convergence proof of the Policy Iteration algorithm, the convergence proof of Q-Learning follows these steps:

1. Show that the optimal policy is a fixed point of the Operator $H$

$$(Hq)(x,a) = \sum_{x_{t+1}} P(x_{t+1}|x_t)[r(x_t, x_{t+1}, a_t) + \beta \max_a q(x_{t+1}, a)]$$

2. Prove that the operator $H$ is a contraction.

Thus, by the Banach contraction mapping theorem, it is proven that the iterated computation of the operator $H$ converges uniformly to the unique fixed point, the optimal policy of the problem in question.

The proof can be found in more detail in Regehr and Ayoub (2021)

### 2.4.4 Curse of Dimensionality

The solution method presented relies on creating a "grid" that represents the possible state and control variables, which becomes prohibitive when dealing with a large number of variables.

The curse of dimensionality, a term coined by Bellman (1953), refers to the phenomenon where the volume of the state space exponentially increases with each added dimension, making many optimization and reinforcement learning algorithms computationally unfeasible. This challenge is particularly evident in reinforcement learning, where the value function $V$ or the quality function $Q$ must be estimated for every possible combination of state and action.

In reinforcement learning, assembling a "grid" of all possible combinations of state and control variables can quickly become impractical as the number of variables increases. The computational complexity and the amount of memory required to store and update corresponding values scale exponentially, making traditional solution methods, such as Dynamic Programming, less viable in high-dimensional environments.

$$\text{Number of points in the grid} = N^d$$

where $N$ is the number of points per dimension and $d$ is the number of dimensions. Thus, even a modest increase in the number of dimensions (variables) can lead to exponential growth in the number of required points, resulting in a computationally intractable problem.

To mitigate the curse of dimensionality, techniques such as function approximation, deep learning, and dimensionality reduction methods are often employed. These approaches seek to simplify the state space by focusing on representative features and learning more compact representations that capture the essential information for making informed decisions.

Considering the complexity of economic and financial models, the curse of dimensionality highlights the need for innovative methods that can efficiently handle large state and action spaces, enabling the practical application of reinforcement learning in complex economic environments.

### 2.4.5 Numerical Example of the Curse of Dimensionality

Consider a numerical example to illustrate the curse of dimensionality: suppose we are sampling $N$ continuous variables $\{x_i\}_{i=1}^{N} \in [0,1]^N$, taking 80% of their lengths as the sample. Let $SV$ be the sampled hypervolume of the possible $N$ variables $(0.8^N)$. We can then observe the following values for different numbers of variables:

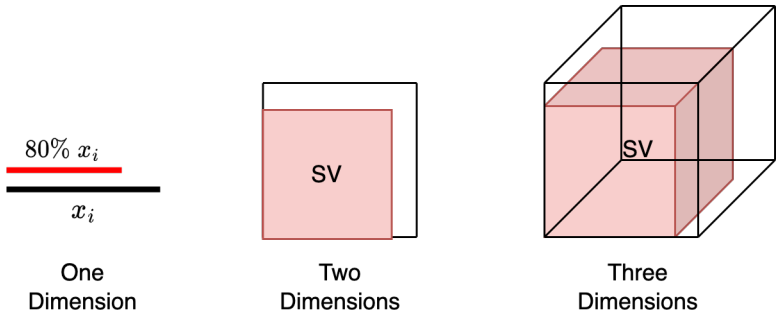


Figure 3 – Graphical representation illustrating the rapid decline in sampled hypervolume with increasing dimensions.

Notice that even when a significant portion of individual variables is sampled, the sampled hypervolume becomes negligible already with $N = 10$. In practical terms, this means that when sampling 80% of each of 10 variables individually, we have only managed to sample about 10% of all possible combinations. This example highlights how quickly the curse of dimensionality can diminish the effectiveness of sampling in high-dimensional spaces, emphasizing the challenges faced in complex model simulations and analyses.

| N | 1 | 2 | 5 | 10 | 20 | 50 |
|---|---|---|---|---|---|---|
| SV | 80% | 64% | 32% | 10% | 1% | 0.001% |

## 2.5 Function Approximators

The innovation presented in the influential paper "Playing Atari with Deep Reinforcement Learning" (MNIH *et al.*, 2013) is based on dynamic optimization and employs

Bellman's equation, addressed through the method of "Q-Learning" (as discussed in Section 2.4), to tackle the complexity of approximating the quality function $Q$ using neural networks. This approach can be seen as a solution to the curse of dimensionality, and by approximating a large set of function points with a set of base functions, we make it possible to reduce the amount of information needed to understand a process.

In this section, we aim to provide context for understanding the solution proposed for the Agent-Based Model (ABM). We will begin with an exploration of function approximators, progressing to a discussion about neural networks and how they are used to approximate unknown functions. We will conclude with the formulation of the loss function that the neural network aims to minimize and an explanation of the "Deep Q-Learning" algorithm.

### 2.5.1 History of Function Approximators

The endeavor to mathematically approximate functions began with the work of Euler (1775), where the renowned mathematician described a projective function for mapping the Russian Empire, aiming to transform meridians into parallel lines. Euler's approach required a series of intervals. Subsequently, Laplace refined this technique, achieving similar results with a finite set of points, surpassing the number of parameters in the problem (STEFFENS; ANASTASSIOU, 2006).

In 1820, Fourier introduced a general solution for function approximation, which, despite fixing a functional form, required fewer points than parameters and minimized the maximum error among various adjusted functions. Due to its elegance and pedagogical significance, Fourier's approach to decomposing functions into series will be the starting point for understanding the structure of a neural network in this text.

#### 2.5.1.1 Divertissement: Fourier Series

While studying the heat diffusion $u(x,t)$[1] in metal plates, Fourier faced a challenge: determining the initial heat distribution $u(x,0) = f(x)$ for a problem yet unresolved beyond its trivial solution. This issue was modeled by the partial differential equation known as the heat equation, here expressed for one spatial dimension:

$$\frac{\partial u(x,t)}{\partial t} = K\frac{\partial^2 u(x,t)}{\partial x^2}$$

The trivial solutions to this equation take the form:

$$u(x,t) = Ae^{-Kn^2\pi^2 t}\sin(n\pi x), \quad n \in \mathbb{N}$$

[1] $u(x,t)$ - Function modeling temperature along a spatial dimension $x$ as a function of time $t$.

By applying the initial conditions $u(x, 0) = f(x)$, we derive:

$$f(x) = A \sin(n\pi x), \quad n \in \mathbb{N}$$

Given the linearity of the differential operator, we can state, through the principle of superposition, that a linear combination of the trivial solutions also satisfies the equation. Therefore, if $f(x)$ can be expressed as a linear combination of sine functions, we have a solution for the heat equation:

$$f(x) = \sum_{i=1}^{k} A_i \sin(n_i \pi x), \quad n_i \in \mathbb{N}$$

Fourier pondered whether it was possible to approximate any periodic and odd function by an infinite series of sinusoidal terms. The affirmative response led to the creation of Fourier series, in which a function $f(x)$ can be approximated by a sum of $k$ sine functions, with the approximation becoming exact as $k \to \infty$.

The historical example of Fourier series leads us to two discussions relevant to this work. First, it illustrates a functional approximation tool, essential to science and a precursor to the modern concept of neural networks as universal function approximators.

To illustrate this, consider Fourier's solution within the structure of a neural network. Numerically, the series is truncated at a finite value $k$. Figure 4 represents the summation of sinusoidal terms as a network diagram, which anticipates the formalism used for neural networks:

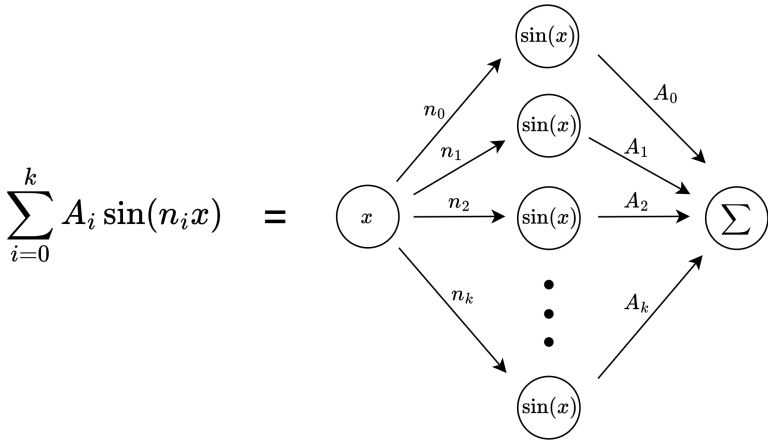


Figure 4 – Representation of the Fourier series as a network diagram. This formalism is useful for transitioning to neural networks.

In diagram 4, the edges represent multiplicative weights, and the nodes represent functions to be applied—in neural network terminology, edges are weights and nodes are

activation functions. The parameters to be adjusted, dependent on the chosen function $f(x)$, correspond to the edge weights $A_i$ and frequencies $n_i$, although in neural networks, these weights are real numbers rather than natural ones. Adjusting these weights is analogous to training a neural network.

Fourier analytically determined the coefficients $A_i$, but in more general applications, we will use the "Backpropagation" process and gradient descent to adjust the weights by minimizing a loss function, which will be detailed later in this chapter.

Finally, Fourier's approach is a classic example of the methodology described by Polya (1973), where a complex solution is constructed from simpler solutions, in this case representing an arbitrary periodic function as an infinite series of trigonometric terms.

The previously shown diagram 4, illustrating the solution of the Fourier series, may initially seem more complex compared to the classical representation in the form of summations. This is understandable, as methodologies involving neural networks transcend the idea of mere summations of functions. They are designed to capture a hierarchical composition of weighted sums and activation functions applied sequentially. This added complexity is crucial, as it endows neural networks with the capability to model complex interactions and patterns that go beyond simple linear accumulation of signals.

### 2.5.2 Neuron

While it is common to explain neural networks through bioinspired analogies, this approach can sometimes obscure their fundamental principle: the iterated composition of functions.

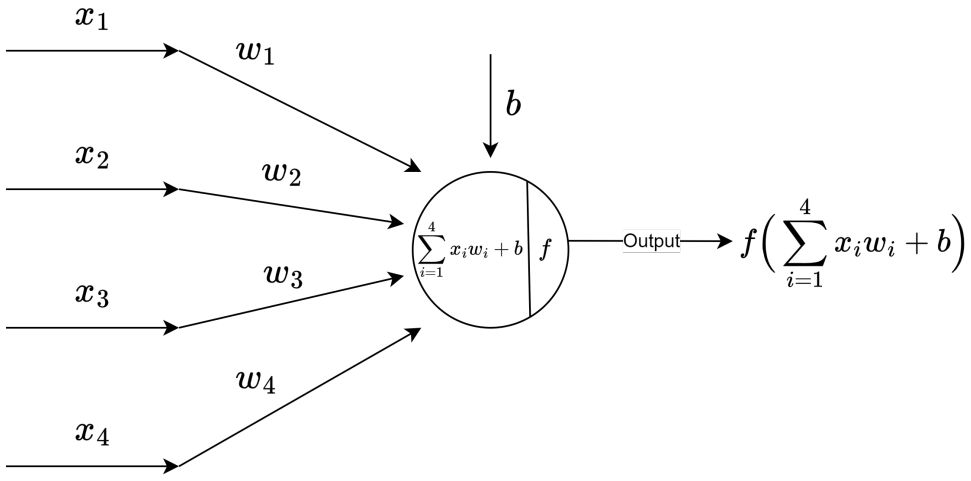


Figure 5 – Diagram illustrating the operation of a neuron unit, which consists of a set of weights $w_i$, a constant $b$, and an activation function $f$. Essentially, a neuron is just another representation of a function, or more precisely, a function from the domain $\mathbb{R}^n$ to the codomain $\mathbb{R}$.

As the fundamental unit of a neural network, the neuron can be understood in detail in Figure 5. Essentially, it represents a function whose partial derivatives can be efficiently calculated since they depend only on $f'(\cdot)$, the derivative of the activation function, which is fixed and chosen precisely for this purpose.

Observing Figure 5, if we adopt an identity activation function, $f(x) = x$, and rename the result of the calculation to $\hat{y}$, the weights $w_i$ to $\beta_i$, and the bias $b$ to $\beta_0$, we obtain the following relationship:

$$w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b = \underbrace{\beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_3 + \beta_4x_4 = \hat{y}}_{\text{Structure analogous to Linear Regression}}$$

In practical econometrics, we would start from a set of samples $(\underline{X}_i, y_i)$, check the premises for applying the ordinary least squares method, and seek to minimize the sum of squared residuals. This traditional approach typically leads to solutions of the type:

$$\beta = (X'X)^{-1}X'y$$

While the elementary case reveals an elegant solution, it faces the computational challenge of inverting the matrix $X'X$, a costly procedure of order $\mathbb{O}(n^3)$ where $n$ is the number of variables. Alternative methods such as QR decomposition or direct solving of the linear system $X'X\beta = X'y$ are often preferred. Even more efficient is the use of gradient descent, an iterative process that converges due to the linearity of $\beta_i$ and the convexity of the objective function. This technique updates $\beta_i$ in each iteration, following the negative direction of the gradient of the cost function:

$$\bar{\beta}_i \leftarrow \beta_i - \alpha\Big(\frac{\partial}{\partial\beta}(\sum(y_i - \hat{y}_i)^2\Big)$$
$$\beta_i \leftarrow \bar{\beta}_i$$

Gradient descent not only guarantees convergence in the context of ordinary least squares but is also computationally accessible and extendable to other activation functions beyond identity. By exploring different activation functions, although we may lose the guarantee of convexity of the cost function and have to settle for local minimum solutions, we gain significant flexibility in modeling complex nonlinear relationships.

Consider then a neural network $\hat{f}(X;\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the set of weights to be optimized (said to be parameterized by $\boldsymbol{\theta}$). The goal is to optimize these weights to minimize the loss function, defined as:

$$\mathcal{L}(\boldsymbol{\theta}) = \sum(\hat{f}(X_i; \boldsymbol{\theta}) - y_i)^2$$

Podemos então definir a atualização dos pesos da rede da seguinte forma:

$$\bar{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta} - \alpha \nabla \mathcal{L}(\boldsymbol{\theta})$$
$$\boldsymbol{\theta} \leftarrow \bar{\boldsymbol{\theta}}$$

Where $\nabla \mathcal{L}(\boldsymbol{\theta})$ represents the usual gradient operation of the loss function $\mathcal{L}$ evaluated at the point $\boldsymbol{\theta}$.

The optimization strategy using gradient descent was first conceived in 1847 when Louis Augustin Cauchy used it to calculate the orbit of celestial bodies (LEMARECHAL, 2012). The stochastic variant of gradient descent was first documented in 1951 by Robbins and Monro (1951) and later applied by Rosenblatt (1958) in their seminal work "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," a concept simplified in Figure 5.

It's important to note that the methods of training neural networks are so extensive that they justify a dissertation of their own. Among them, the backpropagation algorithm stands out, essential for the effectiveness of training modern neural networks. However, we will now focus on the main formulation of this work: the loss function we aim to minimize.

## 2.6 ABM Solution: The Deep Q-Network

In the quest for an effective approach to train agents in complex environments, the integration of reinforcement learning techniques with deep neural networks has emerged as a promising field of research. This method, known as the Deep Q-Network (DQN), merges the generalization capability of neural networks with the optimization framework of Q-Learning, a landmark in reinforcement learning. In this section, we explore the foundation of this approach, focusing on defining a loss function that captures the nuances of the problem at hand. The formulation of the loss function is crucial, as it guides the agent's learning process toward convergence on a policy that maximizes the accumulated reward over time, thus reflecting the quality of the decisions made by the agent.

The Deep Q-Network approach adapts the principles of Q-Learning for use with high-dimensional input spaces, where traditional methods face the curse of dimensionality. By employing neural networks as function approximators, DQNs can efficiently handle large state and action spaces, which are characteristic of many real-world problems, including complex economic models.

### 2.6.1 Formulation of the Loss Function

In the DQN framework, the agent learns to approximate the optimal action-value function, commonly known as the Q-function. This function, $Q(s, a; \boldsymbol{\theta})$, estimates the expected return for taking action $a$ in state $s$, following policy $\pi$. The agent's objective is to find the optimal policy that maximizes the expected return from each state. The loss function, therefore, is designed to measure the difference between the currently estimated Q-values and the target Q-values, which are updated using the Bellman equation.

Effectively training a neural network necessitates the definition of a loss function that quantifies the deviation of the network's predictions from the optimal conditions of the problem. We start from the classic formulation of the Q-Learning algorithm, as expressed in Equation (2.11), which is presented as follows:

$$Q^{i+1}(s_k, a_k) = (1 - \alpha)Q^i(s_k, a_k) + \alpha\Big[r_k + \beta \max_a Q(s_{k+1}, a)\Big]$$

Expanding and reorganizing the terms using the distributive of $\alpha$, we obtain:

$$\begin{aligned} Q^{i+1}(s_k, a_k) &= Q^i(s_k, a_k) + -\alpha Q^i(s_k, a_k) + \alpha\Big[r_k + \beta \max_a Q(s_{k+1}, a)\Big] \\ Q^{i+1}(s_k, a_k) &= Q^i(s_k, a_k) + \alpha\Big[\underbrace{r_k + \beta \max_a Q(s_{k+1}, a) - Q^i(s_k, a_k)}_{\text{Bellman Error}}\Big] \end{aligned}$$

We observe that if the estimated function $Q^i(s_k, a_k)$ exactly corresponds to the optimal quality function $Q^*(s_k, a_k)$ , the Bellman error would be zero. This principle guides the definition of the loss function used by Mnih *et al.* (2013) in their influential paper "Playing Atari with Deep Reinforcement Learning", where the network is trained to minimize the Bellman error, essential for the convergence of the Q-Learning algorithm when combined with deep neural networks.

Thus, we finally arrive at the loss function $\mathcal{L}$ that will be minimized during the agents' maximization process:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}\Big[\Big( \underbrace{r_k + \beta \max_a Q(s_{k+1}, a)}_{\text{Simulated Term}} - \underbrace{Q^i(s_k, a_k; \boldsymbol{\theta})}_{\text{Neural Network}} \Big)^2\Big] \tag{2.11}$$

Where the "Simulated Term" corresponds to the true value of the quality function that we aim to achieve with the simulated actions, and the "Neural Network" represents the current estimate of the $Q$ function parameterized by the weights $\boldsymbol{\theta}$. This formulation

encapsulates the essence of Deep Q-Learning, enabling the training of neural networks to approximate the optimal policy in complex decision-making environments.

To carry out the minimization process, it is necessary to simulate a complete sequence of trajectories in order to calculate the "Simulated Term":

$$r_k + \beta \max_a Q(s_{k+1}, a) = r_k + \sum_{i=1}^{w} \beta^i r_{k+i} + \beta^{w+1} \epsilon \tag{2.12}$$

Here, the $r_i$ values will be known through simulation, and the term $\epsilon$ will depend on the mechanics of the model. If we were dealing with a finite-time model, and $w$ were large enough to reach the end of the summation, we would have $\epsilon = 0$. Since agents in the Lucas model have infinite lifetimes, some estimation is required, such as using the function $Q^i(s_k, a_k; \boldsymbol{\theta})$ itself, or we could argue that the value of $\beta^{w+1}$ will be small enough to become an insignificant portion in the summation.

Through the detailed loss function, we can structure a paradigm for agent training that is both robust and adaptable. This framework not only allows for an accurate simulation of the environment through the "Simulated Term" but also integrates the continuous learning of the neural network that approximates the $Q$ function. The ability to optimize agents in a context of an infinite life horizon, as illustrated by the Lucas model, highlights the flexibility of the DQN method. While this study presents the fundamental architecture and training principles, the full potential of DQN is realized when implemented and tested against the dynamic complexity of real-world environments, paving the way for future research and practical applications.

# 3 RESULTS

This chapter is dedicated to the presentation and analysis of the results achieved throughout this dissertation. Given the intensity of the computational resources required by the simulations, the efficient structuring of the computational project assumed a critical role in the success of this study.

We proceed to describe the architecture of the neural network used to approximate the agents' value function $Q$. This section meticulously details the selection of the architecture, as well as the choice and impact of hyperparameters in the optimization process. Transparency in this step is fundamental to understanding how the decisions made in modeling influence the final results.

Finally, we present the practical results obtained during training episodes. This includes the evolution of prices determined by the agents over time and a careful analysis of how they behave and adapt their strategies in response to conditions in the simulated market. In addition, we provide insight into the challenges and obstacles faced during the prolonged training process, offering a deeper understanding of the practical and theoretical aspects involved in applying reinforcement learning to complex economic models.

This chapter not only illustrates the results achieved but also provides insights into the nuances and challenges of employing advanced machine learning techniques in economic analysis.

## 3.1 Computational Simulation Project

The implementation of a computational simulation project for this study presented unique challenges, especially in terms of flexibility and integration with advanced technologies. Although there are numerous libraries dedicated to agent-based models, none completely suited the specificities and needs of this work.

The key requirements for the development of the computational simulation project included:

- **Capability for Parallelization:** Essential for ensuring efficient execution on Graphics Processing Units (GPUs), particularly given the large number of simulations required for agent training.
- **Efficient Memory Mechanism:** An optimized memory structure was crucial for storing and prioritizing the most recent trajectories of the agents' state and control variables, a fundamental aspect for the learning dynamics of the model, as a history of trajectories is necessary for the training to occur.

- **Integration with Machine Learning Libraries:** The ability to work in conjunction with machine learning tools was indispensable for the effective minimization of the loss function through a neural network. Given the complexity of efficient optimization of neural networks, the use of existing libraries offered a practical and robust solution.

Given the absence of ready-made frameworks capable of meeting all these demands — particularly frameworks dedicated to efficient simulations and efficient integration with machine learning techniques — the decision was made to develop a custom simulation environment. This tailor-made solution was developed to fully meet the listed requirements, ensuring that the simulation environment was fully aligned with the objectives and methodology proposed for this research.

### 3.1.1 Software Architecture

For a clear understanding of the simulation project's architecture, Figure 6 presents a diagram that outlines the key components and their interactions. This diagram is essential for visualizing how the different parts of the simulation system interact and cooperate to achieve the results.

The following describes each component of the architecture, with a detailed explanation of their functions and mechanisms. The complete source code is available in the project repository.

### 3.1.2 Lucas Environment

The "Lucas Environment" component, thoroughly described in the development section, is one of the cornerstones of the project in terms of computational cost. This module manages the dynamics of trading and pricing in the market, performing the "match" between agents and ensuring compliance with the model's constraints. Conventional implementations of this environment would result in a complexity of $\mathbf{O}(n^2)$, where $n$ is the number of agents. However, the approach adopted here allows reducing this complexity to $\mathbf{O}(n)$, thanks to the choice of an efficient and simplified market mechanism in conjunction with the use of GPUs.

The missing parameters to be defined here for the agents follow the description of the Lucas model and are defined as follows:

$$u(c_t) = \frac{c_t^{1-\gamma}}{1-\gamma} \tag{3.1}$$

$$\beta = 0.95 \tag{3.2}$$

$$Y_t \sim \frac{1}{100} Lognormal(1, 1) \tag{3.3}$$

$$\tag{3.4}$$

Following part of the discussion presented by (MEHRA; PRESCOTT, 1985), we will use the same instantaneous utility function, and for an initial simulation, we will make the dividends of the tree follow a Lognormal distribution with a mean of $10^{-2}$ and a standard deviation of $10^{-4}$.

The implementation of the environment component follows the interface structure suggested by Towers *et al.* (2023), with some adaptations to consider an environment with multiple agents. Thus, we can proceed with some standard methods of the TF-Agents library (GUADARRAMA *et al.*, 2018).

### 3.1.3 The Orchestrator Class

The “Driver” component acts as the core of the simulation, managing the flow of information emitted by the environment and redirecting it to the storage system (Replay Buffer). This module also plays a crucial role in determining the appropriate times to update the agents' policies. Although the policy update can be interpreted as an update of the function $Q$, it is more practical to follow the standard terminology of the TensorFlow and TF-Agents libraries to maintain consistency and clarity in the implementation. Note that the agents' policy is always to choose the action that maximizes the function $Q$.

### 3.1.4 Replay Buffer

For efficient storage and selection of trajectories for updating the $Q$ function, the "Reverb" library (CASSIRER *et al.*, 2021) was adopted. This choice allowed the outsourcing of the complex task of managing a large volume of data, taking advantage of management policies already implemented and optimized by the library itself. The use of "Reverb" allowed us to focus efforts on other parts of the simulation, rather than on the logistics of storage and data retrieval.

### 3.1.5 Agent Policy Model

As discussed in the development chapter, the quality function $Q : \mathbb{R}^6 \mapsto \mathbb{R}^1$ was parameterized using a neural network with parameters $\boldsymbol{\theta}$, as illustrated in figure 7. The adopted neural network has a sequential architecture with four internal dense layers, each containing 50 neurons and using the RELU activation function (AGARAP, 2018). This

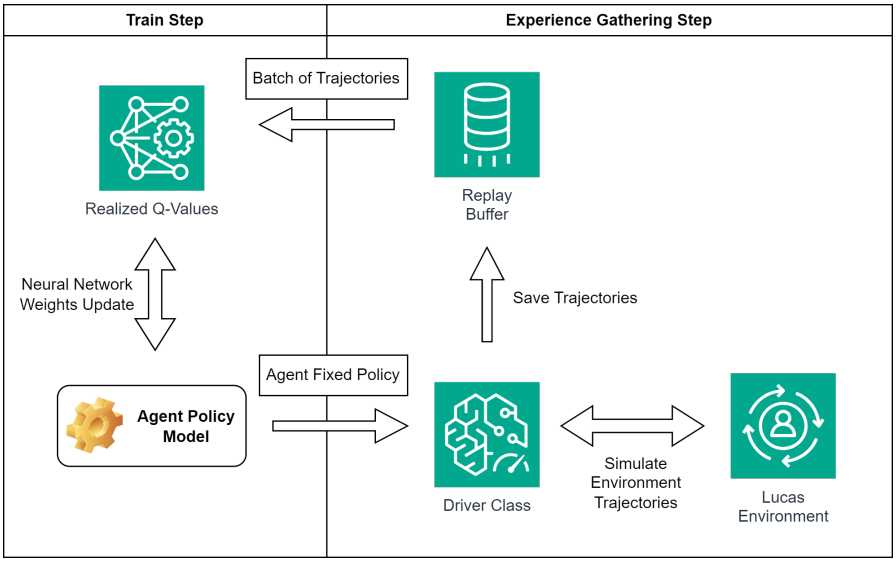


Figure 6 – General diagram of the implemented architecture. Standard terminologies from the TensorFlow and TF-Agents environments are being used to facilitate integration and familiarity. It's important to note that the division between training and experience collection, illustrated in the diagram, serves primarily for didactic purposes, highlighting that they are distinct processes. In practice, both occur simultaneously and are intrinsically interconnected in the simulation flow.

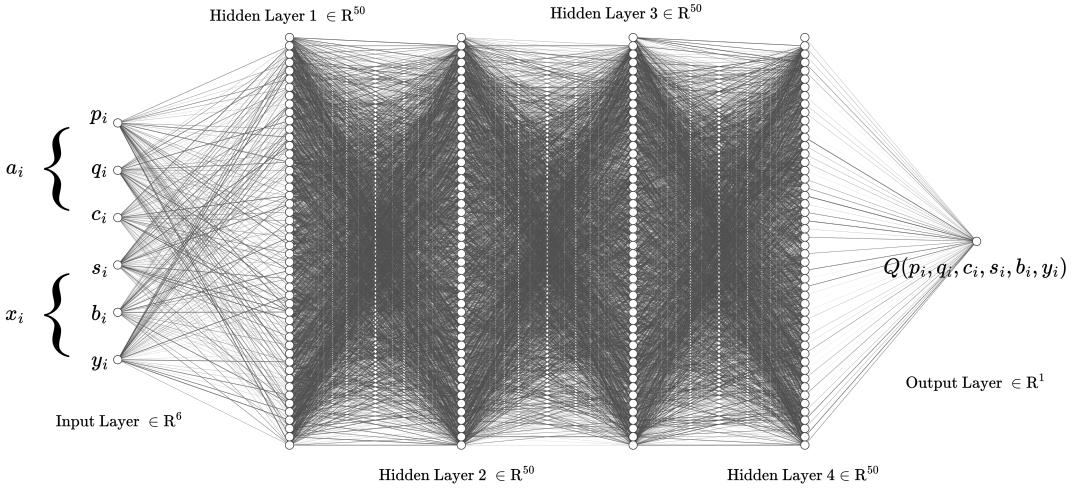


Figure 7 – Chosen Neural Network Architecture to adapt to the quality function "Q"

architecture choice aims to balance complexity and computational efficiency, providing the necessary flexibility for modeling the $Q$ function.

Analyzing this image facilitates understanding the parametrization by $\boldsymbol{\theta}$. With the architecture fixed, the focus lies on defining the weights of the connections (edges), represented by the vector $\boldsymbol{\theta}$. In a dense layer, all possible combinations of connections are made, resulting in $|\boldsymbol{\theta}| = 6 \cdot 50 + 50 \cdot 50 + 50 \cdot 50 + 50 \cdot 50 + 50 \cdot 1 = 7850$, i.e., $\boldsymbol{\theta} \in \mathbb{R}^{7850}$.

The procedures for finding the parameter vector that minimizes the loss function, although there is not always a guarantee of convergence to the global minimum, can be found in more detail in Bishop and Nasrabadi (2006).

The search for the parameter vector $\boldsymbol{\theta}$ that minimizes the loss function in a neural network is a central challenge in the field of machine learning and an area of intense research. While a global optimal solution is not always guaranteed due to the complex nature of loss functions in deep neural networks, various methods have been developed and improved to navigate the solution space efficiently. These methods seek to find a set of parameters that is good enough for practical needs, even if it is not the absolute global optimum.

In this work, we adopted the Adam optimization technique (KINGMA; BA, 2014), with a learning rate parameter of 0.001, a variant of the gradient descent method. The Adam algorithm is chosen for its efficiency and ability to iteratively adjust the network's parameters in response to the gradient of the loss function. This process seeks to minimize the error between the predictions generated by the neural network and the actual values, by continuously adjusting the weights and biases in its layers, thus enhancing the network's predictive performance.

For those interested in deepening their understanding of the theoretical foundations and optimization techniques in machine learning, the work of Bishop and Nasrabadi (2006) is a valuable reference.

## 3.2 Training Results

This section examines the behavior of agents over multiple training episodes, during which the loss function was optimized. It's important to remember, an episode is defined as a single execution of an economy, starting at time $t = 0$, with truncation at $t = 300$. These agents, equipped with their state variables, choose control variables that have the highest value for the estimated quality function $\hat{Q}$ by the neural network.

To better visualize the evolution of the agents' strategies during training, we selected a representative sample of 10 agents. Trajectories of some of their key variables are presented below, allowing a detailed analysis of the optimization process outcomes.

Note that, due to the initial random weights in the neural network, the agents exhibit initially random behaviors in the first episode. However, as episodes progress and the neural network undergoes successive updates, there is an emerging trend of agents moving towards more coherent and effective strategies in maximizing their utilities. This phenomenon illustrates the learning and adaptation process of agents in a simulated economic environment, reflecting the neural network's ability to capture and incorporate market complexities and nuances.

Throughout the development and testing of the proposed simulation, I encountered challenges related to non-convergence due to various factors, primarily linked to limitations in time and available hardware resources. These constraints necessitated the implementation of several simplifications in the model. Initially, the model was designed with 5000 agents, each having distinct risk aversion parameters, along with a more complex market mechanism to calculate aggregate consumption. However, adjustments were necessary, and the final model was resized to accommodate 1200 agents with homogeneous characteristics. Furthermore, the aggregate consumption was simplified to be represented by a lognormal return variable, without autocorrelation, to adapt the model to the operational limitations faced.

## 3.3 Stock Price Evolution Throughout Simulations

In this section, I discuss and analyze the evolution of the price $p$ of the risky asset $s$ in the Lucas Tree throughout the simulations.

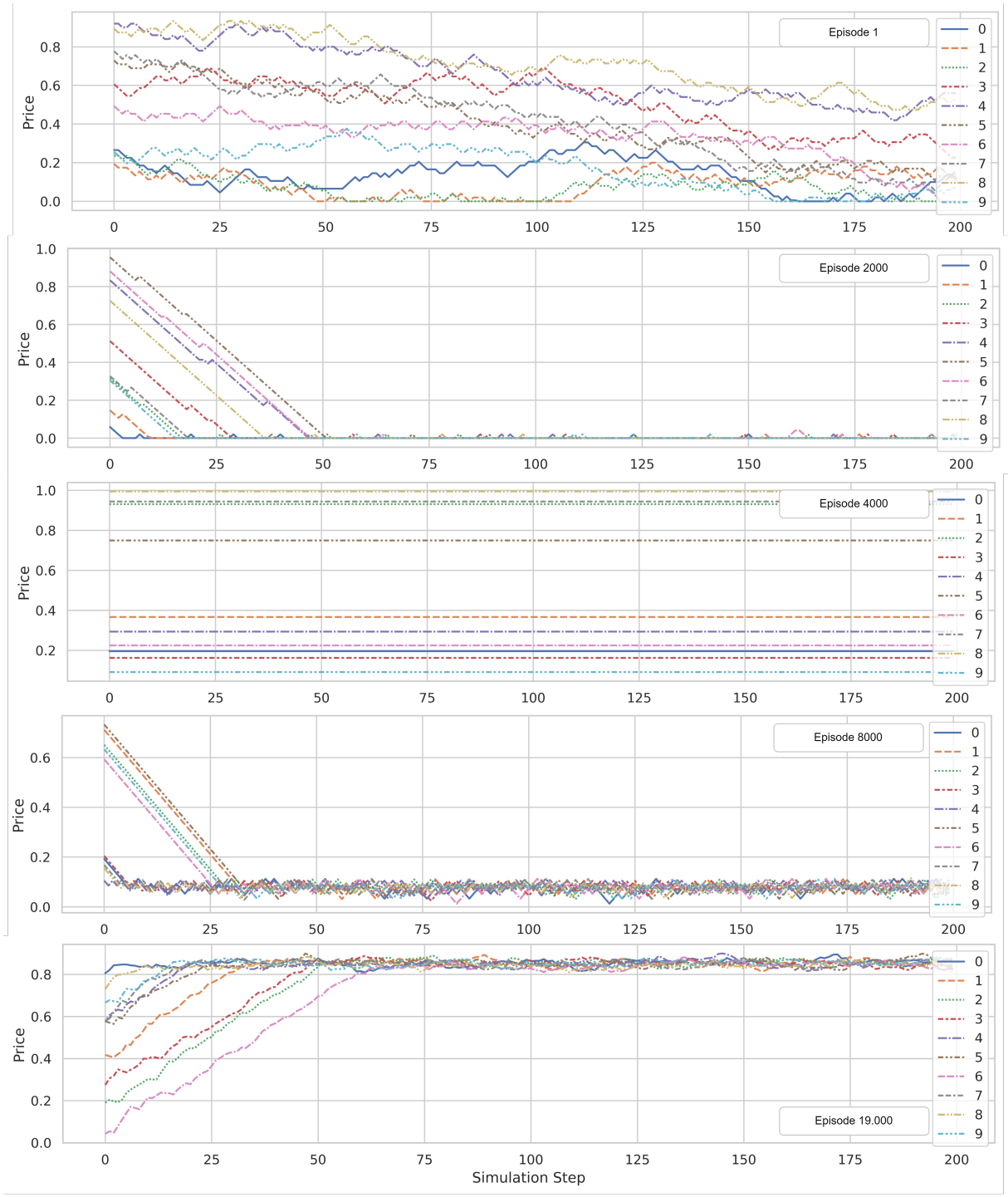


Figure 8 – Evolution of prices $p_i$ for the risky asset $s$ of 10 different agents over 19,000 complete simulations. Notice that the equilibrium result for the Lucas model for the price is 0.845, for the asset simulated with returns following a lognormal distribution with a mean of 1 and a standard deviation of 1, with a multiplicative factor of 1/100.

As illustrated in figure 8, the first episode exhibits a predicted behavior: agents, without a history of interactions with the environment, act almost randomly. In subsequent analyses, we will observe the impacts of these pricing decisions on the agents' portfolios. Let's take, for example, agents 8 (with prices higher than average) and agent 2 (with lower prices). According to the market mechanics, agent 8 is a potential buyer of agent 2, so if a match occurs between them, a trade of a quantity of asset $s$ at the average price between the two agents will happen.

After 2,000 episodes, agents adjusted their strategies to a downward price race pattern, one of the most frequently observed local minima in the simulations. Although it is premature to draw definitive conclusions at this stage, since the training process is far from any convergence, a tendency of agents to reduce prices for immediate gains from asset sales is noticeable.

By episode 4,000, the strategies are still evolving and difficult to interpret. However, by episode 8,000, a more cohesive approach emerges, with all agents converging to the same price. This indicates an optimal point in terms of absence of transactions, aligned with the ideal result of the Lucas model, although the price is still far from the theoretical optimum. The emerging strategy suggests a preference of agents to buy more of the asset at low prices, aiming to increase their future stream of discounted utilities.

From episode 10,000 onwards, I continued to monitor the simulation every 1,000 episodes. Only by episode 19,000 did the strategies begin to approach the optimal values predicted by the model. From an analytical perspective, we could argue that the state reached by episode 8,000 already represented an optimal point, a discussion that will be further explored in the following sections.

## 3.4 Evolution of Traded Stock Quantities Throughout Simulations

To complement the previous analysis focused on price evolution, this section examines the variation in the quantity of asset $s$ for the same agents discussed earlier (see figure 8).

Figure 9 presents the evolution of these quantities throughout the simulations. In the first episode, we observe a direct reflection of what was discussed in the price section. Without prior experiences, agents like Agent 8, who set higher prices, tend to buy the asset $s$ from agents with lower prices, exemplified by Agent 2.

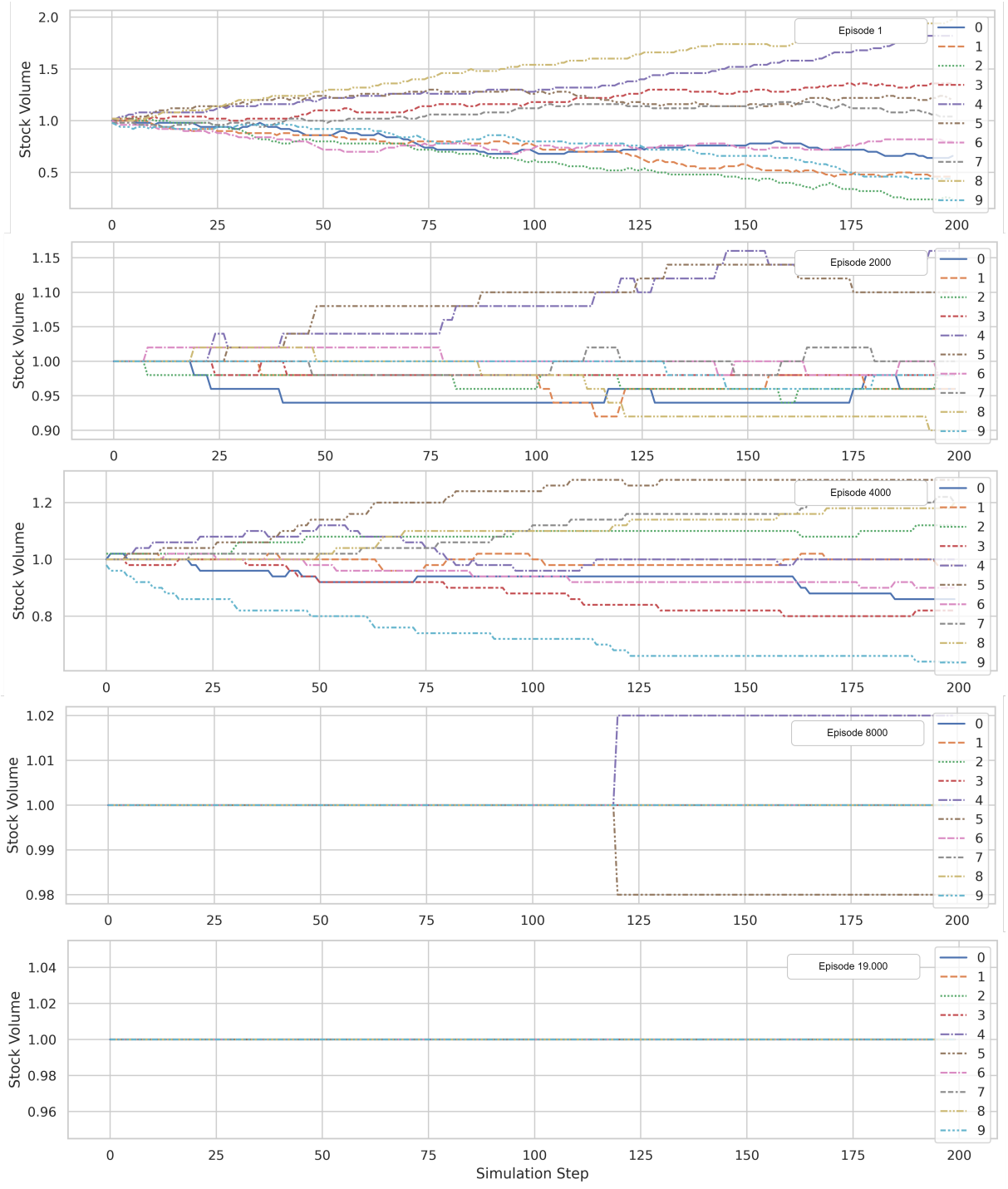


Figure 9 – Evolution of the quantities of risky asset of 10 different agents over 19,000 complete simulations, note that the quantities are normalized to correspond to the initial percentage of the agent's asset at the start of each simulation

In episodes 2000 and 4000, there is an improvement in the agents' control over their buying and selling positions, indicating a refinement in their strategies. An interesting point emerges in episode 8000, where agents show a tendency to avoid trading regardless of the proposed prices. In fact, only one exchange occurs between Agents 4 and 5, followed

by a period without further transactions. This behavior can be interpreted as optimal from the perspective of individual utilities, as suggested by the Lucas model, which advocates maintaining the initial endowment of the asset in an economy of identical agents. However, the possibility of buying remains attractive due to prices being below the optimum. The realization of purchases, though, is contingent on the willingness of another agent to sell.

By episode 19000, we reach the outcomes predicted by the Lucas model without growth. As expected, no trades occur between agents in the optimal state, reflecting the stabilization of strategies and the effectiveness of the agents' learning and adaptation process throughout the simulations.

## 3.5 Bond Variables Evolution Throughout Simulations

Let's now examine the evolution of variables associated with the risk-free asset, $b$. According to the Lucas Tree model, when assuming constant consumption, the price of the asset is given by:

$$q_i = \mathbb{E}[m_{t+1}] = \mathbb{E}[\frac{u'(\bar{c})}{\beta u'(\bar{c})}] = \mathbb{E}[\frac{1}{\beta}] = \frac{1}{\beta}$$

In the scenario of the model with a representative agent, agents are not expected to be willing to trade the risk-free asset. Therefore, in an optimal equilibrium as proposed by the Lucas model, we should observe a constant price for $q_i$ and no asset trading between agents.

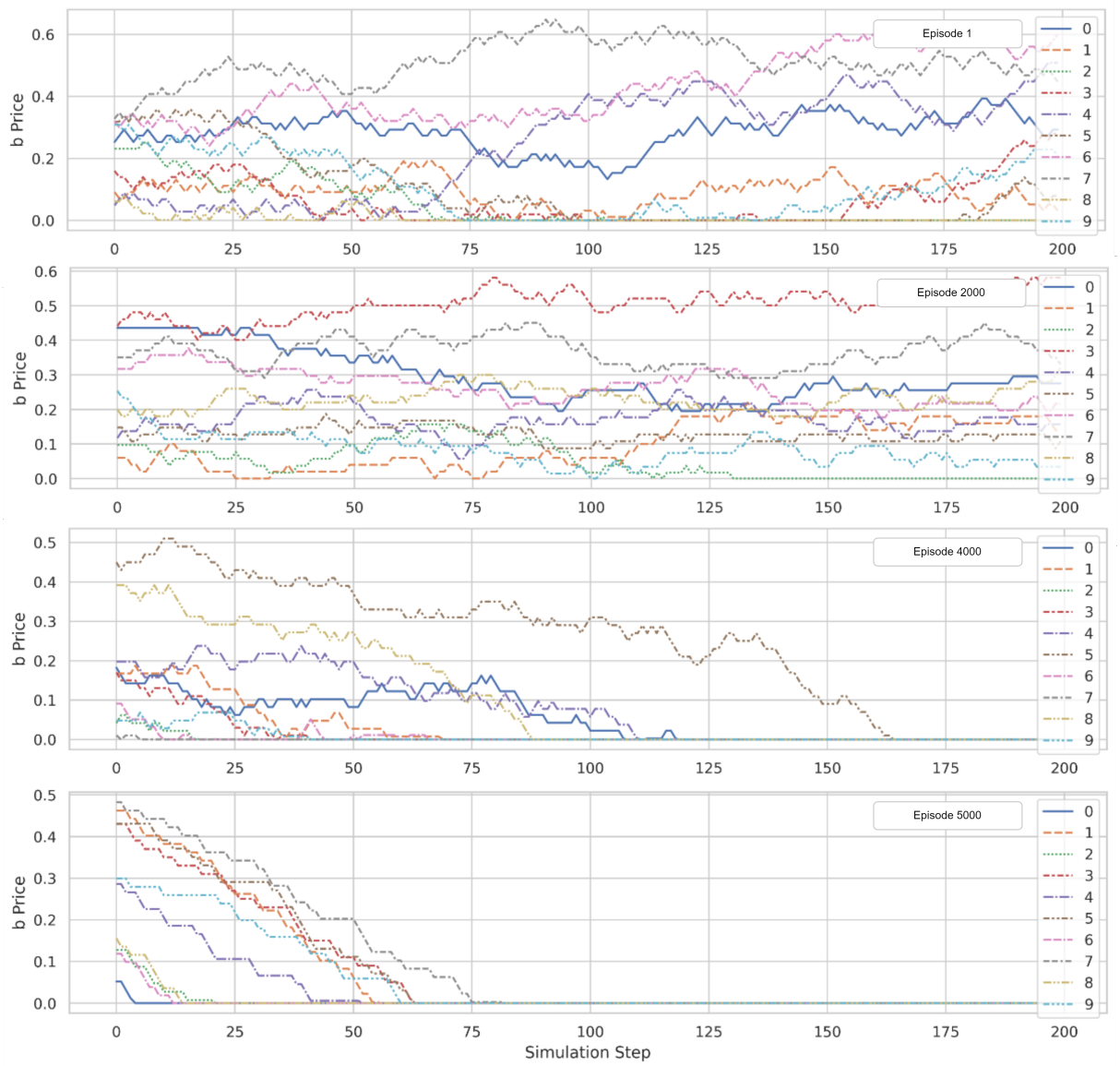


Figure 10 – Evolution of the price $q_i$ for the risk-free asset $b$ for the same 10 agents. In the optimal equilibrium of the Lucas model, with constant consumption, we would expect a constant price and no trading between agents.

Analyzing figure 10, we observe a deviation from the expected behavior. Firstly, the convergence of the agents' strategies for the risk-free asset occurs more rapidly compared to the variable return asset. This phenomenon might be attributed to the uncertainty associated with future returns, potentially increasing the number of training epochs needed for the model. Interestingly, agents do not drive the price to the theoretical optimum of the Lucas model but towards a trend of lower prices.

Considering that trades continue to occur, there is no justification for this price reduction. Once $q_i \leq 1/\beta$, agents would have a marginal return higher than the marginal cost, encouraging the purchase of the asset. However, this logic, while valid in the aggregate model, does not apply directly in the ABM context. The reason for the absence of trades

in the simulated market is similar to the equilibrium argument of the traditional model: for a trade to occur, there needs to be at least one agent willing to buy and another willing to sell. Since this is not the case, no trades occur, aligning with the predictions of the traditional model.

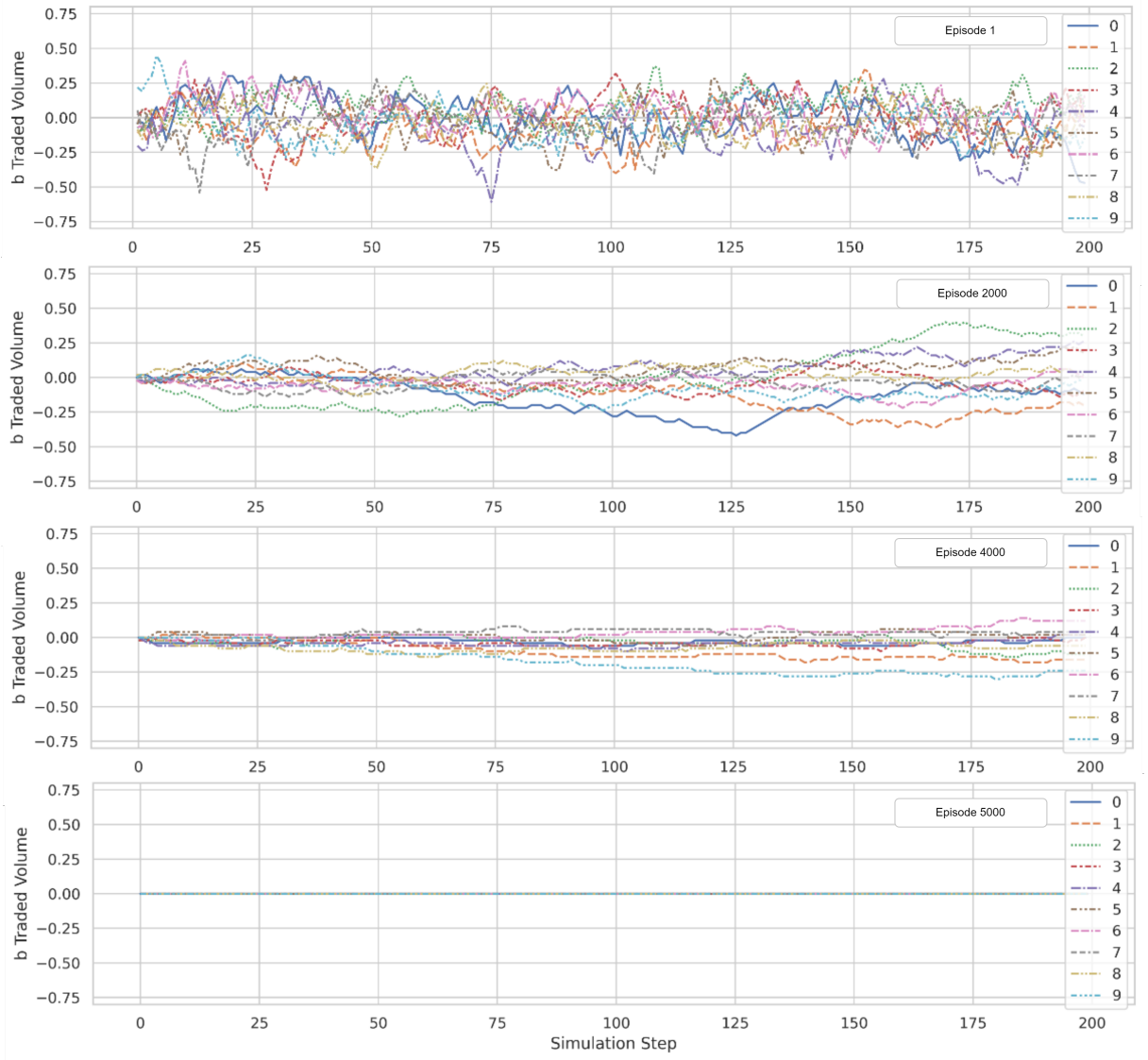


Figure 11 – Evolution of the traded volumes $q_i$ by the agents for the same 10 agents. In the optimal equilibrium of this economy, no trading of the asset occurs, as we are dealing with agents with identical preferences.

Figure 11 clearly shows how the agents converge to the optimal state of the economy. Notice that the convergence to not buying or selling the risk-free asset occurs between episodes 4000 and 5000, at least four times faster than the convergence of the price of the risky asset, which occurred between episodes 18000 and 19000.

## 3.6 Complexities of Simulations

In the field of machine learning, a recurring challenge is the fine-tuning of model hyperparameters to increase the likelihood of finding the global minimum of the loss function. This work faced such complexity: for the training series discussed, involving more than 25,000 episodes, there were at least 15 complete runs, each containing at least the same number of episodes, that did not converge to the theoretical optimum.

During the development of an artificial intelligence project, it is common to select certain parameters based on promising results reported in the literature. This includes choices related to the architecture of the neural network, such as the number of layers, activation functions, and weight initialization methods, as well as adaptations in the loss function, choice of optimizers, and definition of learning parameters. However, the formulation presented in this work has no precedent in the literature, representing an additional challenge. To achieve consistent and viable results, it was necessary to adopt a simplification: the constancy of the total product in the simulated economy. This decision was crucial to enable the location of the optimal point, considering the particularities and novelty of the approach proposed here.

# 4 CONCLUSION

## 4.1 Contributions

This dissertation contributes to the economic literature by transferring the results of the Lucas Tree model to the context of an agent-based model using reinforcement learning techniques. This approach allows for the determination of optimal utility flows without the need for explicitly deriving the model's optimization conditions, representing a significant step towards more agile and less restrictive modeling.

One of the major advancements of this study is the efficient simulation of the problem, highlighting the importance of careful implementation of market mechanisms. Although the efficiency of the simulation depends on the mechanisms used, this work sheds light on procedures that can be optimized for a wide range of applications in financial markets.

Furthermore, I consider the didactic value of the work. By elucidating the technical aspects of neural networks and reinforcement learning in language accessible to economists, I hope this study facilitates the dissemination and adoption of these techniques in other strands of economic research.

The structure of this model constitutes a platform for future research, considering the relaxation of some hypotheses, such as complete markets and rational expectations of agents. It serves as a starting point for future investigations that seek to explore and incorporate emerging variables, constantly evolving market dynamics, and the vast range of agent behaviors. Agent-based models enable simulations in economics considering microeconomic foundations without the need for aggregates, thus establishing another bridge between economic theory and practical applications.

## 4.2 Limitations and Criticisms

The adoption of reinforcement learning techniques based on "Q-Learning" in this work was guided by the technical rigor of its mathematical formulations and the familiarity that the economic community has with the Bellman equation, facilitating the didactic process of its explanation. However, it is important to note that despite their theoretical robustness, these techniques do not necessarily reflect the state of the art in terms of performance or efficiency compared to more recent techniques. In this sense, the scope of the current study could be expanded and refined for applications that forgo extensive demonstrations and justifications of convergence, focusing exclusively on optimization and the minimization of specific metrics.

Furthermore, when evaluating the effectiveness of algorithms employing Bellman error, it is necessary to consider that there are significant controversies regarding its efficiency as proxy variables in the search for the loss function's minimum (FUJIMOTO *et al.*, 2022). Given the slow convergence rate of these algorithms, recent research indicates that Bellman error may not be the best indicator for identifying the optimum of the loss function.

## 4.3 Recommendations for Future Research

### 4.3.1 Study of Hyperparameter Relationships with Convergence Speeds

Future research could delve deeper into the influence of hyperparameters on the convergence speed of agent-based models, particularly in reinforcement learning contexts. Identifying an optimal set of hyperparameters is crucial to balance the model's generalization capability and its speed in adapting to new market conditions. Systematic studies varying these hyperparameters could uncover important insights for constructing more efficient and accurate models.

### 4.3.2 Effects of Bargaining Power

The bargaining power of agents, as buyers and sellers in the asset market, can significantly impact price formation. Subsequent research could explicitly model bargaining power and study its effects on market dynamics. This might include how changes in market structures or regulations impact agents' negotiating power and, in turn, asset prices.

### 4.3.3 Informational Asymmetry

Informational asymmetries are common in financial markets and can lead to significant inefficiencies. Future work could incorporate different levels of information among agents and study how this affects asset pricing and market efficiency. Moreover, it would be interesting to investigate strategies that agents develop to overcome or exploit these asymmetries. Note that this approach allows us to study strategies that spontaneously emerge among agents, a possibility only viable through the chosen modeling strategy.

### 4.3.4 Agent Heterogeneity

Although this study simulated agents individually, we ended up choosing equal parameters for all agents to follow the premises of the Lucas model. That said, there is a wide range of heterogeneity that can be explored. Future research could include a variety of agents' preferences, constraints, and expectations, allowing a deeper understanding of how the diversity of agents affects market dynamics.

### 4.3.5 Behavioral Heuristics

The complexity of investor behavior is not fully captured by traditional rational models. Future studies could delve into the behavioral heuristics that investors use in decision-making, such as loss aversion, overconfidence, and anchoring. Modeling these heuristics within the ABM framework could provide a more realistic representation of how markets operate and evolve under actual conditions.

## 4.4 Conclusion

This dissertation ventured into the intersection of traditional asset pricing theory with modern and evolving computational techniques, particularly agent-based modeling (ABM) applied to the renowned Lucas Tree model featuring autonomous agents optimizing their own utility.

The results demonstrate that ABM methodologies are capable of achieving outcomes consistent with the theoretical optima of the traditional model. Beyond these results, they can provide deeper insights into the workings of financial markets, highlighting the importance of considering behavioral nuances and the adaptability of agents in a dynamic environment. The integration of reinforcement learning techniques allowed the simulated agents to adjust their strategies in response to market changes, a crucial aspect in finance that static models fail to adequately accommodate. This fact also allows us to abstract any optimization mechanics of the agent, only controlling the flow of information received by the agents.

Although the scope of this research was necessarily limited, particularly by the goal of replicating a classic model for representative agent finance, it is straightforward to modify the already implemented project to relax hypotheses and conduct other experiments. However, depending on the relaxed hypothesis, there would no longer be a closed result to compare to, and we would be in a somewhat unknown field, precisely why this work chose to replicate a known result.

The methodological and data limitations point to future research opportunities, suggesting the application of this model to different sets of rules and expansion to include multiple forms of agent learning. I cite here as a primary opportunity the simulation of heterogeneous agents and also different market mechanics. Future studies can benefit from applying this model in realistic market contexts, allowing finance practitioners a richer understanding of asset price determinants and market behavior, and perhaps the replication of known and yet unexplained stylized facts.

In conclusion, the financial modeling techniques suggested here (Agent-Based Modeling with Deep Reinforcement Learning) solve a particular case of one of the main models of theoretical finance. The transition from a purely analytical to a more empirical

and behavioral paradigm, while challenging, is a natural evolution in response to the complexity of financial markets. Through continuous methodological adaptation and innovation, we can enhance the understanding of asset pricing in real economies.

# REFERENCES


AGARAP, A. F. Deep learning using rectified linear units (relu). **arXiv preprint arXiv:1803.08375**, 2018.

ATHEY, S. The impact of machine learning on economics. *In*: **The economics of artificial intelligence: An agenda**. [*S.l.: s.n.*]: University of Chicago Press, 2018. p. 507–547.

ATHEY, S.; IMBENS, G. W. Machine learning methods that economists should know about. **Annual Review of Economics**, Annual Reviews, v. 11, p. 685–725, 2019.

BANZ, R. W. The relationship between return and market value of common stocks. **Journal of Financial Economics**, v. 9, n. 1, p. 3–18, 1981. ISSN 0304-405X. Available at: https://www.sciencedirect.com/science/article/pii/0304405X81900180.

BELLMAN, R. E. **An Introduction to the Theory of Dynamic Programming**. Santa Monica, CA: RAND Corporation, 1953.

BENVENISTE, L. M.; SCHEINKMAN, J. A. On the differentiability of the value function in dynamic models of economics. **Econometrica**, [Wiley, Econometric Society], v. 47, n. 3, p. 727–732, 1979. ISSN 00129682, 14680262. Available at: http://www.jstor.org/stable/1910417.

BISHOP, C. M.; NASRABADI, N. M. **Pattern recognition and machine learning**. [*S.l.: s.n.*]: Springer, 2006. v. 4.

BREEDEN, D. T. An intertemporal asset pricing model with stochastic consumption and investment opportunities. **Journal of Financial Economics**, v. 7, n. 3, p. 265–296, 1979. ISSN 0304-405X. Available at: https://www.sciencedirect.com/science/article/pii/0304405X79900163.

CASSIRER, A. *et al.* **Reverb: A Framework For Experience Replay**. 2021.

CHERNOZHUKOV, V. *et al.* **Double/debiased machine learning for treatment and structural parameters**. [*S.l.: s.n.*]: Oxford University Press Oxford, UK, 2018.

DAVID, A. Heterogeneous beliefs, speculation, and the equity premium. **The Journal of Finance**, Wiley Online Library, v. 63, n. 1, p. 41–83, 2008.

DOSI, G. *et al.* Rational heuristics? expectations and behaviors in evolving economies with heterogeneous interacting agents. **Economic Inquiry**, Wiley Online Library, v. 58, n. 3, p. 1487–1516, 2020.

EBRAHIM, M. S.; MATHUR, I. Investor heterogeneity, market segmentation, leverage and the equity premium puzzle. **Journal of banking & finance**, Elsevier, v. 25, n. 10, p. 1897–1919, 2001.

EULER, L. De proiectione geographica deslisliana in mappa generali imperii russici usitata. **Acta Academiae Scientiarum Imperialis Petropolitanae**, p. 288–297, 01 1775.

EVEN-DAR, E.; MANSOUR, Y. Learning rates for q-learning. **J. Mach. Learn. Res.**, v. 5, p. 1–25, 2004. Available at: https://api.semanticscholar.org/CorpusID:1750497.

FAMA, E. F.; FRENCH, K. R. Common risk factors in the returns on stocks and bonds. **Journal of Financial Economics**, v. 33, n. 1, p. 3–56, 1993. ISSN 0304-405X. Available at: https://www.sciencedirect.com/science/article/pii/0304405X93900235.

FUJIMOTO, S. *et al.* **Why Should I Trust You, Bellman? The Bellman Error is a Poor Replacement for Value Error**. 2022.

GARDNER, M. Mathematical games. **Scientific American**, Scientific American, a division of Nature America, Inc., v. 223, n. 4, p. 120–123, 1970. ISSN 00368733, 19467087. Available at: http://www.jstor.org/stable/24927642.

GU, S.; KELLY, B.; XIU, D. Empirical asset pricing via machine learning. **The Review of Financial Studies**, Oxford University Press, v. 33, n. 5, p. 2223–2273, 2020.

GUADARRAMA, S. *et al.* **TF-Agents: A library for Reinforcement Learning in TensorFlow**. 2018. https://github.com/tensorflow/agents. [Online; accessed 25-June-2019]. Available at: https://github.com/tensorflow/agents.

HANSEN, L. P.; SINGLETON, K. J. Generalized instrumental variables estimation of nonlinear rational expectations models. **Econometrica**, [Wiley, Econometric Society], v. 50, n. 5, p. 1269–1286, 1982. ISSN 00129682, 14680262. Available at: http://www.jstor.org/stable/1911873.

HARVEY, C. R.; LIU, Y.; ZHU, H. ... and the cross-section of expected returns. **The Review of Financial Studies**, Oxford University Press, v. 29, n. 1, p. 5–68, 2016.

HESTER, T. *et al.* Deep q-learning from demonstrations. *In*: **Proceedings of the AAAI conference on artificial intelligence**. [*S.l.: s.n.*], 2018. v. 32, n. 1.

HOWARD, R. A. Dynamic programming and markov processes. John Wiley, 1960.

JOHNSON, J. P.; RHODES, A.; WILDENBEEST, M. Platform design when sellers use pricing algorithms. **Econometrica**, v. 91, n. 5, p. 1841–1879, 2023. Available at: https://onlinelibrary.wiley.com/doi/abs/10.3982/ECTA19978.

KAHNEMAN, D. Maps of bounded rationality: Psychology for behavioral economics. **American economic review**, American Economic Association, v. 93, n. 5, p. 1449–1475, 2003.

KAHNEMAN, D.; TVERSKY, A. Prospect theory: An analysis of decision under risk. **Econometrica**, [Wiley, Econometric Society], v. 47, n. 2, p. 263–291, 1979. ISSN 00129682, 14680262. Available at: http://www.jstor.org/stable/1914185.

KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. **arXiv preprint arXiv:1412.6980**, 2014.

KOCHERLAKOTA, N. R. The Equity Premium: It's Still a Puzzle. **Journal of Economic Literature**, v. 34, n. 1, p. 42–71, March 1996. Available at: https://ideas.repec.org/a/aea/jeclit/v34y1996i1p42-71.html.

LEMARECHAL, C. Cauchy and the gradient method. **Doc Math Extra**, v. 251, n. 254, p. 10, 2012.

LJUNGQVIST, L.; SARGENT, T. J. **Recursive macroeconomic theory**. [*S.l.: s.n.*]: MIT press, 2018.

LUCAS, R. E. Asset prices in an exchange economy. **Econometrica**, [Wiley, Econometric Society], v. 46, n. 6, p. 1429–1445, 1978. ISSN 00129682, 14680262. Available at: http://www.jstor.org/stable/1913837.

LUCAS, R. E. Asset prices in an exchange economy. **Econometrica**, [Wiley, Econometric Society], v. 46, n. 6, p. 1429–1445, 1978. ISSN 00129682, 14680262. Available at: http://www.jstor.org/stable/1913837.

MCLEAN, R. D.; PONTIFF, J. Does academic research destroy stock return predictability? **The Journal of Finance**, Wiley Online Library, v. 71, n. 1, p. 5–32, 2016.

MEHRA, R.; PRESCOTT, E. The equity premium: A puzzle. **Journal of Monetary Economics**, v. 15, n. 2, p. 145–161, 1985. Available at: https://EconPapers.repec.org/RePEc:eee:moneco:v:15:y:1985:i:2:p:145-161.

MNIH, V. *et al.* Playing atari with deep reinforcement learning. **CoRR**, abs/1312.5602, 2013. Available at: http://arxiv.org/abs/1312.5602.

MONTI, C. *et al.* On learning agent-based models from data. **Scientific Reports**, Nature Publishing Group UK London, v. 13, n. 1, p. 9268, 2023.

NEUMANN, J. v. Theory of self-reproducing automata. **Edited by Arthur W. Burks**, 1966.

POLYA, G. **How to solve it: A new aspect of mathematical method**. 2. ed. [*S.l.: s.n.*]: Princeton university press, 1973.

PROSPERI, M. *et al.* Causal inference and counterfactual prediction in machine learning for actionable healthcare. **Nature Machine Intelligence**, Nature Publishing Group UK London, v. 2, n. 7, p. 369–375, 2020.

REGEHR, M. T.; AYOUB, A. **An Elementary Proof that Q-learning Converges Almost Surely**. 2021.

ROBBINS, H.; MONRO, S. A Stochastic Approximation Method. **The Annals of Mathematical Statistics**, Institute of Mathematical Statistics, v. 22, n. 3, p. 400 – 407, 1951. Available at: https://doi.org/10.1214/aoms/1177729586.

ROSENBERG, K. R. B.; LANSTEIN, R. Persuasive evidence of. **The Journal of Portfolio Management**, Princeton University Press, p. 48, 1985.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

SCHELLING, T. C. Dynamic models of segregation. **Journal of mathematical sociology**, Taylor & Francis, v. 1, n. 2, p. 143–186, 1971.

SCHÖLKOPF, B. Causality for machine learning. *In*: **Probabilistic and Causal Inference: The Works of Judea Pearl**. [*S.l.: s.n.*], 2022. p. 765–804.

SELTEN, R. Bounded rationality. **Journal of Institutional and Theoretical Economics (JITE) / Zeitschrift für die gesamte Staatswissenschaft**, Mohr Siebeck GmbH  Co. KG, v. 146, n. 4, p. 649–658, 1990. ISSN 09324569. Available at: http://www.jstor.org/stable/40751353.

SHARPE, W. F. Capital asset prices: A theory of market equilibrium under conditions of risk*. **The Journal of Finance**, v. 19, n. 3, p. 425–442, 1964. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1964.tb02865.x.

STEFFENS, K.-G.; ANASTASSIOU, G. The history of approximation theory. from euler to bernstein. **The History of Approximation Theory: From Euler to Bernstein**, p. 1–219, 01 2006.

STEINBACHER, M. *et al.* Advances in the agent-based modeling of economic and social behavior. **SN Business & Economics**, Springer, v. 1, n. 7, p. 99, 2021.

STIGLITZ, J. E. A Consumption-Oriented Theory of the Demand for Financial Assets and the Term Structure of Interest Rates1. **The Review of Economic Studies**, v. 37, n. 3, p. 321–351, 07 1970. ISSN 0034-6527. Available at: https://doi.org/10.2307/2296724.

STIGLITZ, J. E. Rethinking macroeconomics: What failed, and how to repair it. **Journal of the European Economic Association**, Oxford University Press, v. 9, n. 4, p. 591–645, 2011. ISSN 15424766, 15424774. Available at: http://www.jstor.org/stable/25836083.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. [*S.l.: s.n.*]: MIT press, 2018.

TOWERS, M. *et al.* **Gymnasium**. Zenodo, 2023. Available at: https://zenodo.org/record/8127025.

VASELLINI, R. Abm applications to financial markets. *In*: **AI in the Financial Markets: New Algorithms and Solutions**. [*S.l.: s.n.*]: Springer, 2023. p. 73–83.

WEIL, P. The equity premium puzzle and the risk-free rate puzzle. **Journal of monetary economics**, Elsevier, v. 24, n. 3, p. 401–421, 1989.