Masterarbeit
Zur Erlangung des akademischen Grades Master of Science (M.Sc.)

Analysis of Multi-Agent Reinforcement Learning from a Statistical Physics Perspective

eingereicht von: David Goll

Gutachter/innen: Prof. Dr. Dr. h.c. mult. Jürgen Kurths
Dr. Jobst Heitzig

Eingereicht am Institut für Physik der Humboldt-Universität zu Berlin am: 31.01.2025

Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen, Bilder sowie die Nutzung von Künstlicher Intelligenz für die Erstellung von Texten und Abbildungen, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, ...

Note on the Use of AI-Based Tools

During the preparation of this thesis, LLM-based AI tools were used to improve the linguistic quality of the text. All sentences and paragraphs were independently written and composed by me, while AI tools were exclusively employed for improving spelling, punctuation, translation, and language refinement. No content was generated by AI.

David Goll
31.01.2025

Abstract

Multi-Agent Reinforcement Learning involves interacting agents whose learning processes are coupled through their shared environment, giving rise to emergent, collective dynamics that are sensitive to initial conditions and parameter variations. This thesis explores how a statistical physics perspective can be applied to illuminate the mechanisms governing collective behaviour, levering in particular the toolset of dynamical systems theory. By constructing deterministic approximation models of stochastic algorithms, this approach has uncovered some of the underlying dynamics. Nonetheless, even in the simple independent Q-learning algorithm with a Boltzmann exploration policy, significant discrepancies arise between the actual dynamics and previous approximation models. It is clarified why these models actually do not approximate the original algorithm but interesting variants, which simplify the learning dynamics. To resolve the inconsistencies, a new approximation model is proposed, which explicitly incorporates agents' update frequencies and demonstrates good agreement with the stochastic dynamics of the real system. The model's utility is showcased by applying it to the question of spontaneous cooperation in social dilemmas. In the Prisoner's Dilemma, it reveals that mutual cooperation is merely a metastable transient phase and not a true equilibrium, making it exploitable. Furthermore, a systematic analysis shows that increasing the discount factor exacerbates a "moving target" problem, preventing convergence to a joint policy by inducing oscillations. The oscillations arise from a supercritical Neimark–Sacker bifurcation, where the unique stable fixed point of the learning dynamics transitions into an unstable focus surrounded by a stable limit cycle. These phenomena are observed not only for independent learning but also in memory-one joint-action Q-learning on the iterated Prisoner's Dilemma. Overall, these results demonstrate that even in trivial two-agent, two-action games, basic algorithms like Q-learning can exhibit complex and unstable learning dynamics.

Zusammenfassung

Multi-Agent Reinforcement Learning behandelt die gekoppelten Lernprozesse von mehereren interagierenden Agenten. Die daraus resultierende emergente Lerndynamik reagiert empfindlich auf Variationen von Anfangsbedingungen und Parameter. Diese Arbeit untersucht, wie eine Statistische Physik Perspektive dabei helfen kann, die zugrundeliegenden Mechanismen besser zu verstehen. Hierfür werden insbesondere die mathematischen Methoden der Nichtlinearen Dynamik angewendet. Durch die Konstruktion deterministischer Näherungsmodelle stochastischer Algorithmen hat dieser Ansatz bereits einige der zugrunde liegenden Dynamiken aufgedeckt. Dennoch treten selbst im relativ simplen Q-Learning-Algorithmus mit einer Boltzmann-policy erhebliche Diskrepanzen zwischen der tatsächlichen Dynamik und früheren Näherungsmodellen auf. In dieser Arbeit wird erläutert, warum diese Modelle tatsächlich interessante Varianten approximieren und die Lerndynamik vereinfachen, anstatt den originalen Algorithmus zu beschreiben. Um diese Inkonsistenzen zu beheben, wird ein neues Näherungsmodell vorgeschlagen, welches die Aktualisierungsfrequenzen der Agenten explizit einbezieht, und mit der beobachteten Dynamik des realen Systems übereinstimmt. Die Nützlichkeit des Modells wird bezüglich der Frage, warum spontane Kooperation beim Q-learning in sozialen Dilemmata auftritt, demonstriert. Im Gefangenendilemma zeigt das Modell, dass dieses Verhalten lediglich eine transiente Übergangsphase und kein echtes Gleichgewicht ist. Darüber hinaus zeigt eine Stabilitäts Analyse des Modells, dass eine Erhöhung des Diskontierungsfaktors das “moving target” Problem verschärft und Oszillationen hervorruft, die Konvergenz zu einer gemeinsamen Strategie verhindert. Eine superkritischen Neimark-Sacker-Bifurkation transformiert den stabilen Fokus der Lerndynamik in einen instabilen Fokus, welcher von einem stabilen Grenzzyklus umgeben ist. Diese Phänomene werden nicht nur bei independent Q-learning, sondern auch bei memory-one joint-action Q-learning des iterierten Gefangenendilemmas beobachtet. Insgesamt zeigen diese Ergebnisse, dass selbst in trivialen Spielen mit zwei Agenten und zwei Aktionen einfache Algorithmen wie Q-Learning komplexe und instabile Lerndynamiken aufweisen können.

Contents

Chapter 1

Introduction

Reinforcement Learning (RL) [1] is a foundational machine learning approach where an agent aims to learn optimal behaviour through trial-and-error interactions with its environment. It has established a strong theoretical foundation and has been successfully applied in diverse domains, including playing video games at a superhuman level [2, 3], mastering complex board games like Go [4], and optimising decision-making in robotics and finance. Algorithmic learning excels in tasks where a single agent operates in a well-defined environment with a clear success metric to optimise. However, real-world problems often involve multiple agents interacting in shared environments, where individual interests may misalign with the collective good, and success may not be easily defined [5, 6].

One such challenge, arguably the most critical humanity faces in the 21st century, is the question how to address climate change [7]. Despite the escalating dangers, policymakers around the world remain painstakingly slow in developing and enforcing global regulatory frameworks capable of mitigating—let alone avoiding—the involved consequences [8]. From a political science perspective this slow progress hardly comes as a surprise: self-interested, irrational and shortsighted agents (e.g. national governments) must learn to cooperate in a global collective action problem [9–11].

In this context, Multi-Agent Reinforcement Learning (MARL) [12] bears potential as a modelling tool to understand how adapting agents act and interact in response to different incentives and constraints [13]. It has been suggested that insights derived from such models could inform the design of frameworks that enhance cooperative behaviour, such as in formalising procedures for international climate agreements [14, 15].

However, extending RL into multi-agent settings introduces unique challenges and complexities not encountered in single-agent RL. MARL, characterised by multiple learning agents adapting continuously to each other and a dynamic environment, exhibit complex learning dynamics that are difficult to interpret and predict [16, 17]. This lack of transparency poses significant risks, particularly when deployed in high-stakes environments [18, 19]. Incidents such as the 2010 Flash Crash[1] [20, 21], serve as a reminder that even interactions between individually simple algorithms can produce complicated and unexpected effects, let alone those involving capable AI agents [18].

The set of conventional techniques in the field of machine learning, focused on numerical analysis and improving performance of specific algorithms [17], is not able to provide a coherent picture of the macroscopic learning dynamics, as these exhibit emergent phenomena that cannot be easily deduced from constituents. This lack of understanding is unfortunate, given that behavioural patterns of a wide range of potentially interesting scenarios—ranging from applications in biology to economics & sociology—could well be encapsulated within this class of model systems [22].

As such, it appears vital to complement conventional machine learning research on MARL with a statistical physics perspective. Statistical physics emerged as a framework to explain macroscopic physical properties—such as temperature, pressure, and entropy—in terms of the collective behaviour of microscopic constituents governed by probability distributions. Rooted

[1] The 2010 Flash Crash was a sudden and severe stock market downturn that occurred in a matter of minutes, driven by an unintended feedback loop between multiple automated trading algorithms.

in the development of classical thermodynamics, its principles have since then been successfully adapted to study diverse phenomena, from phase transitions in ferromagnets [23] and Brownian motion in liquids/gases [24] to complex systems such as social networks [25], ecosystems [26], financial markets [27], and artificial intelligence [28].

While classical thermodynamics is primarily concerned with thermodynamic equilibrium, statistical physics also encompasses the study of non-equilibrium phenomena like diffusion and transport processes in chemical reactions. In this context, dynamical systems theory provides mathematical tools for analysing the time-evolution of non-equilibrium systems, such as by identifying its attractors.

In this thesis, we explore how a statistical physics and in particular a dynamical systems perspective can be beneficial to study MARL. By approximating the stochastic algorithms as deterministic dynamical equations, this approach enables a rigorous analysis of the underlying dynamics and the emergent behaviour. Even in simple two-agent games with basic learning algorithms, such approximation models have revealed a variety of learning dynamics, ranging from convergence to multiple fixed point attractors [29], periodic oscillations [30], and even chaotic behaviour [31], highlighting the complexity and diversity of possible outcomes in MARL. However, deriving these models involves unrealistic assumptions that fail to capture the even more complex actual dynamics. We demonstrate that, even in the simplest case of independent Q-learning, significant discrepancies arise between the algorithm's behaviour and previous approximation models. A new deterministic approximation model is then proposed, showing good agreement with the real system's stochastic dynamics, and the learning dynamics are analysed on the paradigmatic example of the Prisoner's Dilemma.

This thesis provides a comprehensive overview of the fundamentals of MARL while emphasising both the benefits and challenges of adopting a statistical physics perspective. To this end, chapter 2 introduces foundational concepts of reinforcement learning, key principles of game theory and evolutionary game theory, and the unique challenges posed by multi-agent learning. The chapter concludes with a brief introduction to dynamical systems theory and related work, establishing a bridge to the main part of the thesis.

In Chapter 3, we apply this perspective in a detailed case study. The results of this chapter are currently under review for publication [32]. Specifically, we construct a deterministic approximation model of independent Q-learning in single-state environments. The Prisoner's Dilemma serves as a paradigmatic example to demonstrate the model's applicability, enabling an in-depth analysis of how parameters and initial conditions influence the learning dynamics. The analysis shows that even this seemingly trivial case gives rise to intricate phenomena, such as transient dynamics that persist for billions of time steps and bifurcations that prevent convergence to a joint policy. We discuss the implications of the findings regarding how MARL can be used as a modelling tool in complex scenarios and how to ensure meaningful interpretations of results.

Finally, Chapter 4 explores potential extensions to more complex algorithms and game settings, offering directions for future research and broader applications.

The key contributions of this thesis can thus be summarised as follows:

- By providing a historical overview of approximation methods for independent Q-learning in single-state environments, we clarify why previous models fail to capture the original algorithm and instead approximate modified variants.
- We propose a deterministic approximation model that aligns well with the actual learning behaviour observed.
- We demonstrate the presence of prolonged transient phases in the dynamics, which result from heterogeneous update frequencies.
- We prove that seemingly stable cooperative behaviour over billions of time steps in the Prisoner's Dilemma is not an equilibrium solution of the dynamics, but merely a transient phenomenon.
- Through a stability analysis of the new model, we show that the observed oscillations for high discount factor values result from a supercritical Neimark-Sacker bifurcation, driven by a "moving target" problem.

Chapter 2

Background

This chapter builds a comprehensive overview of Multi-Agent Reinforcement Learning (MARL). Section 1 starts with single-agent Reinforcement Learning (RL), explaining concepts such as agents, environments, rewards, and how to define optimal policies. Section 2 explores key elements of game theory and evolutionary game theory, which are instrumental in studying multi-agent interactions. Building on these foundations, section 3 introduces the basic concepts of MARL and highlights the unique challenges inherent to this field. The chapter closes with a short introduction to dynamical systems theory and a concise review of its application in MARL research.

The first three sections are based on and follow the structure of chapters 2–6 of Albrecht et al.'s (2024) recently published textbook [12], which is anticipated to become a foundational reference for the fundamentals of MARL. The first section about single-agent RL also draws on principles from chapters 3 and 6 of Sutton and Barto's (2018) renowned textbook [1].

The mathematical notation follows the conventions outlined in [1]. This notation is extended to encompass multi-agent interactions, drawing on insights from [12]. Random variables are represented by capital letters, such as the state S_t, action A_t, and reward R_t at time t, while their specific realisations are denoted by lowercase letters (e.g., s_t, a_t, r_t).

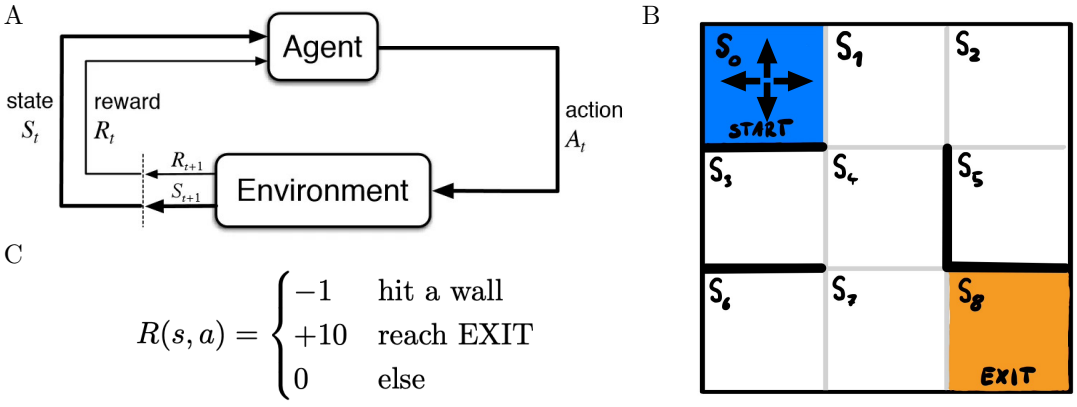


Figure 2.1: (A): The agent–environment interaction in a Markov decision process (adapted from [1]). (B): 2D maze environment with pure transition probabilities and reward function (C). Starting at s_0 the goal of an RL agent is to find the optimal policy of navigating to the exit, s_8.

2.1 Single-Agent Reinforcement Learning

In the domain of machine learning, RL distinguishes itself by focusing on sequential decision-making and learning from non-supervised feedback. At its essence, RL revolves around a singular agent's interaction with its environment, where the agent learns to make decisions through trial and error, in order to achieve a goal. In most cases this goal is specified as maximising an expected cumulative reward. The interaction of the learning agent and its environment happens over a sequence of discrete time steps. Actions are the possible choices of the agent (e.g. move left, right, up, down for a robot in a 2D grid maze illustrated in figure 2.1). States are a property of the environment (e.g. the coordinates of the cell the robot is in). Based on the observations the agent receives about the actual state by the environment, it selects an action and receives feedback in form of a numerical reward (e.g. +10 points for reaching the exit of the maze). The rewards indicate the value of an action in a given state and are the basis for evaluating the policy of the agent. A policy is a function of observations and defines a stochastic rule by which the agent chooses its action, typically based on the observed states.

2.1.1 Markov Decision Process

A fundamental concept in RL is the *Markov Decision Process* (MDP), which provides a formal framework for modelling sequential decision problems.

Definition 2.1.1 (finite Markov Decision Process (MDP))**.** A finite MDP consists of:

- Finite set of states $\mathcal{S}$ and a subset of terminal states $\overline{\mathcal{S}} \subset \mathcal{S}$
- Finite set of actions $\mathcal{A}$
- Reward function $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$
- State transition probability function $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ such that

$$\forall s \in \mathcal{S}, a \in \mathcal{A} : \sum_{s' \in \mathcal{S}} T(s'|a, s) = 1 \tag{2.1}$$

- Initial state distribution $\mu : \mathcal{S} \to [0, 1]$ so that $\sum_{s \in \mathcal{S}} \mu(s) = 1$ and $\forall s \in \overline{\mathcal{S}} : \mu(s) = 0$.

The initial state S_0 is sampled from $\mathcal{S}\setminus\overline{\mathcal{S}}$ according to μ. At each time step t the agent receives an observation from the environment, giving it full information of the state S_t the environment is in. Based on this observation and its current policy π_t, the agent chooses an action A_t from its action space $\mathcal{A}$.[1] The state transition probability function then determines the next state S_{t+1} and the agent receives a numerical reward $R_t = R(S_t, A_t, S_{t+1})$. Such an MDP produces the sequence

$$S_0, A_0, R_0, S_1, A_1, R_1, S_2, A_2, R_2, ...$$

which is continued until a maximal time step t_{max} or a terminal state is reached. An *episode* is one realised run of this stochastic process. A specific realisation is denoted by lowercase letters:

$$s_0, a_0, r_0, s_1, a_1, r_1, s_2, a_2, r_2, ...$$

In a Markov decision process, the probability $p(s_{t+1}, r_t|s_t, a_t)$ of the agent to receive the current reward $r_t = R(s_t, a_t, s_{t+1})$ and transition to the next state s_{t+1} is determined by the current state s_t and action a_t alone,

$$p(s_{t+1}, r_t|s_t, a_t, s_{t-1}, a_{t-1}, ...) = p(s_{t+1}, r_t|s_t, a_t). \tag{2.2}$$

In an MDP the agent has complete knowledge of the state of the environment. In many application however it is unrealistic to assume that the agent receives the full information. In these cases, the system can be modelled by a *partially observable Markov decision process*, where the environment dynamics are determined by an MDP, but the agent only receives partial or noisy information about the true state of the environment.

The goal of an RL agent is to learn a policy that maximises its total return, defined as the cumulative reward over all time steps of one episode. However, this may not be possible as the return is in general a stochastic value. Instead, the *expected* return can be maximised, assuming that the agent follows the policy π to choose its action and that the state transitions are governed by T. Still, for non-terminating MDPs the expected return may be infinite, which is undesirable as a metric of performance. To guarantee finite returns in all finite MDPs, a discount factor $\gamma \in [0, 1)$ is introduced to define the *discounted return*,

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k} = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + ..., \tag{2.3}$$

and the *expected discounted return* under policy π,

$$\mathbb{E}_\pi[G_t] = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} \right], \tag{2.4}$$

with the convention that $R_t = 0$ for $t \geq t_{max}$. To shorten the notation, in the remainder of this thesis, 'return' refers to the discounted return unless stated explicitly otherwise. The same goes for the expected return.

The discount factor γ plays a crucial role in RL. It determines the weight γ^k assigned to rewards received k steps into the future. An agent with a discount factor close to zero prioritises immediate rewards and places minimal value on future rewards. Conversely, a discount factor close to 1 encourages the agent to consider long-term rewards when making decisions. Note, that the discount factor is forbidden to be equal to 1, as it would result in an infinite return. The discount factor is a hyperparameter critical to the agent's success. It is not learned by the agent. The appropriate value depends on the specific environment and learning task.

Equipped with a metric to define success of an agent, we can now define an optimal policy.

[1] For sake of notation we assume the action space to be equal for all states, although this is not a necessary restriction.

2.1.2 Optimal Policy and Bellman Equations

Value functions are a fundamental concept of RL, their estimation is at the core of most algorithms [1]. The *state-value function* v_π estimates how valuable it is for the agent to be in a given state s, assuming it continues to choose actions under the fixed policy π, which is a probabilistic mapping from states to actions. The notion "how valuable" is defined numerically as the expected return given a certain state,

$$\begin{aligned} v_\pi(s) &:= \mathbb{E}_\pi[G_t|S_t = s] \\ &= \mathbb{E}_\pi[R_t + \gamma G_{t+1}|S_t = s] \\ &= \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma\mathbb{E}_\pi[G_{t+1}|S_{t+1} = s']\Big] \\ &= \sum_{a\in\mathcal{A}} \pi(a|s) \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma v_\pi(s')\Big]. \end{aligned} \tag{2.5}$$

Note that in (2.5) the recursive property

$$\begin{aligned} G_t &= R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + ... \\ &= R_t + \gamma G_{t+1} \end{aligned}$$

is used. Similar to the state-value function, we can define the *action-value function* $q_\pi(s,a)$ as the expected return of choosing action a in state s and continuing with policy π,

$$\begin{aligned} q_\pi(s,a) &:= \mathbb{E}_\pi[G_t|S_t = s, A_t = a] \\ &= \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma v_\pi(s')\Big] \\ &= \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma \sum_{a'\in\mathcal{A}} \pi(a'|s') q_\pi(s',a')\Big]. \end{aligned} \tag{2.6}$$

Equation (2.5) and (2.6) are called *Bellman equations* after Richard Bellman and his work on dynamic programming in the 1950's [33]. The Bellman equations allow to define an optimal policy π_* for a finite MDP. A policy π is called *optimal* if its expected return for all states is greater or equal to the expected return of all other policies π'. The optimal value functions are therefore defined as

$$v_*(s) := \max_\pi v_\pi(s), \forall s \in \mathcal{S}, \tag{2.7}$$

$$q_*(s,a) := \max_\pi q_\pi(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}. \tag{2.8}$$

We can write the value functions of optimal policies, the so-called *Bellman optimality equations*, as

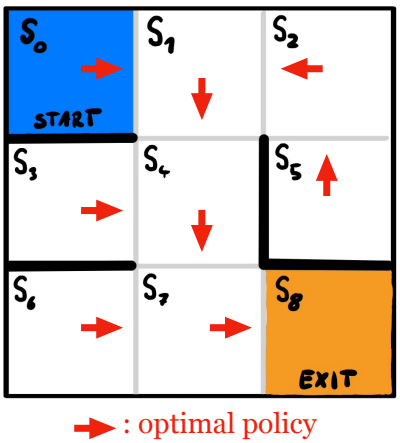
$$v_*(s) = \max_{a\in\mathcal{A}} \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma v_*(s')\Big], \tag{2.9}$$

$$q_*(s,a) = \sum_{s'\in\mathcal{S}} T(s'|s,a)\Big[R(s,a,s') + \gamma \max_{a'\in\mathcal{A}} q_*(s',a')\Big]. \tag{2.10}$$

The Bellman optimality equations form a system of z nonlinear equations with z being the number of states (or state-action pairs) [12]. The optimal value function v_* (or q_*) is the unique solution to this system. Once the optimal value function is known, it is straightforward to derive the optimal policy π_* from it. For the action-value function, any policy that assigns arbitrary probabilities to actions with the maximum value q_* for the given state and a probability of zero to the rest, is an optimal policy. Hence, there can be multiple optimal policies for the unique optimal value function. If the maximum is unique, the optimal policy can be written as

$$\pi_*(s) = \arg\max_{a\in\mathcal{A}} q_*(s,a). \tag{2.11}$$

For the exemplary 2D grid maze, presented in figure 2.1, one can calculate the optimal policy via backward value iteration to deduct the optimal policy (see figure 2.2).



State \ Actions	→	↓	←	↑
State 0	5.12	3.1	3.1	3.1
State 1	5.12	6.4	5.12	4.12
State 2	3.1	4.1	5.12	3.1
State 3	6.4	4.12	4.12	4.12
State 4	5.4	8	5.12	5.12
State 5	2.1	2.3	2.3	4.1
State 6	8	5.4	5.4	5.4
State 7	10	7	6.4	6.4
State 8	0	0	0	0

: max value

Figure 2.2: Optimal policy for the 2D maze environment presented in figure 2.1, calculated via backward value iteration of eq. 2.10 for a discount factor of $\gamma = 0.8$.

2.1.3 Tabular Reinforcement Learning

Tabular RL encompasses methods in which an agent preserves its knowledge about the environment within a so called Q-table. This table stores the estimated quality $Q(s, a)$ of taking action a in a given state s. Through trial and error, the agent updates its Q-table, an estimation of the true action-value function, based on rewards.

While powerful for simple environments, tabular RL struggles with complex ones. The Q-table becomes massive as the number of states and actions grows. This phenomenon was termed *curse of dimensionality* by Richard E. Bellman himself, when considering problems in dynamic programming [33]. Even if storage space would not be an issue, the computation time necessary for learning most likely is. Most algorithms require that the estimation of the Q-values are updated multiple times, rendering tabular methods often impractical for tasks with a high number of states/actions. For example, the number of legal Go board positions, estimated at 10^{170} [34], is finite yet vastly more numerous than the estimated number of atoms in the observable universe, 10^{80} [35]. It is a prime example where a tabular approach is of no use.

Despite this limitation, tabular RL forms the foundation for understanding state-of-the-art RL techniques, such as deep RL, which has driven the remarkable breakthroughs in reinforcement learning over the past decade [2–4]. It provides a clear and transparent framework for grasping core concepts like exploration (trying new actions), exploitation (using known good actions), and value estimation (judging the worth of different choices). Our focus in this thesis will be solely on tabular RL techniques.

In the pursuit of an optimal policy, RL agents face the challenge of balancing exploration and exploitation. Exploration involves trying new actions to gather information about the environment and improve the agent's understanding of the state-action space. Exploitation refers to leveraging the agent's learned knowledge to select actions that are likely to yield high returns based on past experiences. The most common selection strategies are the epsilon-greedy policy and the Boltzmann policy.

The *epsilon-greedy policy* selects the greedy action with the highest estimated value with a probability of $(1 - \epsilon)$ (exploitation), but with a small probability ϵ it occasionally explores all actions randomly to discover potentially better strategies. It is defined as

$$\pi(a|s) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{A}|} & \text{if } a = \arg\max_{a' \in \mathcal{A}} Q(s, a') \\ \frac{\epsilon}{|\mathcal{A}|} & \text{otherwise} \end{cases}, \tag{2.12}$$

where:

- $\pi(a|s)$ is the probability of selecting action a given state s,

- ϵ is the exploration rate,
- $|\mathcal{A}|$ is the number of possible actions.

The *Boltzmann policy*, also known as a softmax policy in the machine learning community, uses a softmax function to convert the estimated action values into a probability distribution, where actions with higher values are more likely to be selected, but all actions have a non-zero probability of being chosen. The Boltzmann policy is defined as

$$\pi(a|s) = \frac{e^{Q(s,a)/T}}{\sum_{a' \in \mathcal{A}} e^{Q(s,a')/T}}, \tag{2.13}$$

where:

- $\pi(a|s)$ is the probability of selecting action a given state s,
- $Q(s, a)$ is the estimated value of taking action a in state s,
- T is called the *temperature* parameter in analogy to thermodynamics.

This corresponds to the Boltzmann distribution for a statistical mechanical system with energy $-Q(s, a)$ at temperature T (disregarding the Boltzmann constant k). Note however, that although the term 'temperature' is borrowed from statistical mechanics, here it does not describe a measurable physical quantity. In physical systems, temperature represents the measure of the average kinetic energy of particles in a system. In RL, it is a dimensionless parameter to control the level of exploration. Higher temperatures lead to more uniform action selection, while lower temperatures favor exploitation of actions with higher values.

In this thesis, we focus on Boltzmann exploration due to its ability to capture continuous changes within the policy space. Unlike epsilon-greedy, which makes discrete exploration-exploitation decisions based only on the maximum of the Q-values, Boltzmann exploration uses a continuous distribution based on actual values. This allows for a smoother transition between exploration and exploitation as Q-values are updated, making it ideal for investigating the nuances of policy changes over time.

2.1.4 Temporal Difference Learning

The focus of this thesis will be on a specific algorithm called Q-learning, which belongs to a a class of algorithms termed *Temporal Difference* (TD) learning. This class of RL algorithms is particularly compelling for several reasons. TD methods are widely adopted in the machine learning community due to their effectiveness in tabular RL settings [1]. Moreover, TD algorithms are rooted in the temporal difference model of dopamine feedback observed in animal learning [36, 37], offering insights not only for a machine learning perspective but also for a broader biological and social learning context.

TD algorithms learn optimal policies by bootstrapping from the current estimate of the value function. Bootstrapping refers here to the process that a value estimate for the state-action pair $Q(s, a)$ is updated using value estimates of other state-actions pairs $Q(s', a')$. In TD learning, the agent updates its estimates based on experiences from interactions with the environment, formalised as Markov chains with a reward process,

$$...S_t, A_t, R_t, S_{t+1}, A_{t+1}, R_{t+1}, S_{t+2}, A_{t+2}, R_{t+2}, ...$$

An experience could be a concrete quadruple (s_t, a_t, r_t, s_{t+s}) that make up the transition from one particular state-action pair to another. TD algorithms employ the subsequent general update rule to acquire estimates of the action-value functions:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \cdot \underbrace{[\tilde{Q}_t - Q(s_t, a_t)]}_{\substack{\text{temporal difference error} \\ D_t}} . \tag{2.14}$$

$Q(s_t, a_t)$ is the current estimate of the action-value function of the state action pair (s_t, a_t) sampled at time step t. As the learning process goes on, the estimation will be more and more refined.

The step size parameter $\alpha \in [0, 1)$ is called *learning rate*. Out of convention, we keep this naming but elaborate a bit on the meaning behind it. The term 'rate' implies a measure of change over time, emphasising how quickly the agent updates its knowledge or adjusts its behaviour in response to new experiences. In physical systems a rate typically refers to a quantity with units such as [time^{-1}] or [distance^{-1}], representing a rate of change of a quantity over time or space. In contrast, the learning rate here is dimensionless and serves as a scaling factor for the magnitude of parameter updates, rather than representing a rate of change in a physical quantity. Despite this small inaccuracy, "learning rate" has become the standard term used in the context of machine learning and optimisation algorithms.

The *temporal difference error* D_t is the difference at time t between the *temporal difference target* $\tilde{Q}_t$ and the current estimate. $\tilde{Q}_t$ is constructed based on the experience samples and depends on the concrete algorithm.

Q-Learning

Q-Learning [38] (algorithm 1), introduced by Watkins and Dayan in 1992, is one of the most widely used TD algorithms and is considered as one of the early breakthroughs in RL [1]. The temporal difference target of Q-learning,

$$\tilde{Q}_t(r_t, s_{t+1}) = r_t + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a'), \tag{2.15}$$

is based on the Bellman optimality equation 2.10, substituting the summation over states s' and the reward function $R(s, a, s')$ with the respective elements from the quadruple (s_t, a_t, r_t, s_{t+1}) [12]. The update rule of Q-Learning reads

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \Big[r_t + \gamma \max_{a' \in \mathcal{A}} Q(s_{t+1}, a') - Q(s_t, a_t) \Big]. \tag{2.16}$$

It was shown that this converges to the optimal action value function so long as all actions are repeatedly sampled in all states an infinite number of times and the learning rate α is reduced over time in a way that fulfils the following conditions [39]:

$$\forall s \in \mathcal{S}, a \in \mathcal{A}: \quad \sum_{k=0}^{\infty} \alpha_k(s, a) = \infty, \qquad \sum_{k=0}^{\infty} \alpha_k(s, a)^2 < \infty,$$

where the index k denotes the k-th selection of action a in state s. The learning rate function $\alpha_k(s, a) = 1/k$ satisfies these conditions. It can be motivated by the fact that the arithmetic mean of the first k elements of a sequence of numbers x_i fulfils the recursion equation $\langle x_i \rangle_{i=1}^{k} = \langle x_i \rangle_{i=1}^{k-1} + \frac{1}{k}\big(x_k - \langle x_i \rangle_{i=1}^{k-1}\big)$. By contrast, a constant learning rate does *not* fulfil the conditions. Nonetheless, a constant rate is often preferred in practice due to its ease of implementation and faster learning performance, although it lacks a theoretical proof of convergence.

As mentioned, this thesis focuses on Q-learning, the most prominent TD algorithm. For a broader perspective, we note the existence of other variants, such as SARSA and Expected SARSA.

SARSA

The SARSA algorithm [40], owes its name to the experience quintuple $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$ it uses in its update rule,

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \Big[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \Big]. \tag{2.17}$$

Note the change of the estimation of the future state-action value compared to Q-Learning. In SARSA, the future state-action value $Q(s_{t+1}, a_{t+1})$ is estimated based on the actual action a_{t+1}

Algorithm 1: Q-Learning with Boltzmann Policy

Input: State space $\mathcal{S}$ with a subset of terminal states $\overline{\mathcal{S}} \subset \mathcal{S}$, action space $\mathcal{A}$, learning rate α, discount factor γ, temperature parameter T
Output: Learned Q-values $Q(s, a)$
Initialise $Q(s, a)$ arbitrarily for all $s \in \mathcal{S}, a \in \mathcal{A}$, except $Q(s \in \bar{\mathcal{S}}, \cdot) = 0$
while *not converged* **do**
  Choose initial state s
  while *not reached terminal state* **do**
    Choose action a with Boltzmann policy:
      $\pi(a|s) \leftarrow \frac{e^{Q(s,a)/T}}{\sum_{a'} e^{Q(s,a')/T}}$ for all $a \in \mathcal{A}$
      $a \sim \pi(\cdot|s)$
    Take action a, observe reward r and next state s'
    Update Q-value:
      $Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$
    Update state: $s \leftarrow s'$
  end
end

taken in the next step under the current policy. It is hence called an *on-policy* method. In Q-Learning, an *off-policy* method, the estimation of the optimal future value $\max_{a' \in \mathcal{A}} Q(s_{t+1}, a')$ is independent of the policy being followed. Q-Learning learns from the maximum Q-value of the next state, irrespective of whether the action in that state will be chosen by its current policy or not.

Expected SARSA

Expected SARSA is a variant of the TD algorithm that estimates the value of state-action pairs by considering the expected value of future actions under the current policy. It computes the expected value by averaging over all possible actions of the next state, weighted by their probabilities under the current policy:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_t + \gamma \sum_{a' \in \mathcal{A}} \pi(a'|s_{t+1}) Q(s_{t+1}, a') - Q(s_t, a_t) \right] \tag{2.18}$$

Given the next state, the algorithm follows a trajectory akin to the expected movement of SARSA, thereby reducing variance attributed to the stochasticity of A_{t+1}, albeit at the expense of heightened computational complexity. Although its estimation of the future value is based on the current policy, the actual action taken in the next step might differ, making it in general an off-policy method.

2.2 Game Theory

Building upon the concepts of single-agent RL explored in the previous section, we now turn our focus to multi-agent scenarios. In these environments, agents not only interact with their surroundings but also with each other. Analysing such interactions forms the core of game theory, a field of mathematics that studies the strategic decision-making of rational interacting agents [41].

This section introduces the fundamental concepts of game theory that are relevant to MARL and this thesis. It is structured into three parts. The first and second part define games and their solution concepts, based on the discussion of chapter 3 and 4 in [12]. The final part extends to evolutionary game theory, which proves to be particularly relevant for analysing MARL from a dynamical systems perspective (see Section 2.4), following the introduction outlined in [42].

To ensure clarity, we primarily adhere to the naming conventions of the RL community. This means using terms like 'agent' instead of 'player', 'reward' instead of 'payoff', and 'policy' instead of 'strategy'. However, we may occasionally use terminology from game theory or evolutionary game theory when the context necessitates it. For instance, the concept of a 'game' in game theory aligns with the RL concept of an 'environment'. We will employ both terms to highlight the different perspectives these communities offer. A table summarising synonymous terms, is provided in table 2.2.

Throughout this work, the superscript index i denotes a specific agent, while $-i$ represents its opponent(s).

2.2.1 Games

Normal-Form Games

Normal-form games represent the foundation for analysing multi-agent interactions.

Definition 2.2.1 (Normal-form game)**.** A *normal-form game* is defined as a tuple $\Gamma = (\mathcal{I}, (\mathcal{A}^i)_{i\in\mathcal{I}}, (R^i)_{i\in\mathcal{I}})$, consisting of:

- a finite set of agents $\mathcal{I} = \{1, 2, \ldots, N\}$, with $N \in \mathbb{N}$,
- a finite set of actions $\mathcal{A}^i$ for each agent i,
- individual reward functions $R^i : \boldsymbol{\mathcal{A}} \to \mathbb{R}$, where $\boldsymbol{\mathcal{A}} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N = \prod_i \mathcal{A}^i$ is the joint action space.

The process of a normal-form game is as follows: Each agent $i \in \mathcal{I}$ samples an action $A^i \in \mathcal{A}^i$ according to its potentially probabilistic policy. Based on the joint action $\boldsymbol{A} = (A^1, ..., A^N)$, each agent receives an individual reward $R^i(\boldsymbol{A})$. In the context of MARL, a normal-form game can be regarded as a multi-agent environment with a single non-terminal state and a single terminal state. After playing one round, the game transitions to the terminal state and is finished.

A defining characteristic of a normal-form game lies in its representation of all individual reward functions $R^i : \boldsymbol{\mathcal{A}} \to \mathbb{R}$ in a tensor $\mathbf{R} \in \mathbb{R}^{|\mathcal{A}^1|\times\cdots\times|\mathcal{A}^N|}$, where each entry corresponds to the joint reward for a specific combination of actions. This representation does not inherently require agents to choose their actions simultaneously, but it emphasises a static structure, where the sequence of decisions is not explicitly modelled.[2]

In the case of two agents, the reward tensor, technically a bimatrix, is commonly referred to as a "reward matrix" in practice, and the normal-form game is termed *matrix game*. Three examples of such matrix games are shown in table 2.1. Here, agent 1 chooses a row, while agent 2 selects a column. The matrix entries, ordered pairs denoted as (r^1, r^2), represent the rewards each agent receives for a specific joint action. Note that the superscript is an index for the agent and not an exponent.

[2] In contrast, extensive-form games represent the decision-making process through a tree-like structure, capturing the sequential nature of actions. In these games, agents make decisions at different points in time, with information about previous actions explicitly displayed along the branches of the tree.

	C	D
C	3,3	0,5
D	5,0	1,1

(a) Prisoner's Dilemma

	H	T
H	1,-1	-1,1
T	-1,1	1,-1

(b) Matching Pennies

	S	H
S	4,4	1,3
H	3,1	3,3

(c) Stag Hunt

	B	S
B	3,2	0,0
S	0,0	2,3

(d) Bach-Stravinsky

Table 2.1: Comparison of Game Matrices: Each matrix represents a different two-agent normal-form game with two actions. The entries in the matrix represent the reward pair for each possible combination of agent choices. The first element of the pair indicates the reward for the row agent, the second element indicates the reward for the column agent. (a) Prisoner's Dilemma, a well-studied example of the collective action problem. (b) Matching Pennies, a zero-sum game where the optimal policy of agents is to choose randomly (c) Stag Hunt, a game where agents must choose between individually safer options or riskier collective cooperation for higher rewards. (d) Bach-Stravinsky (also called Battle of Sexes), a coordination game where agents must coordinate their choices to achieve a mutually beneficial outcome.

The matrix 2.1a depicts the *Prisoner's Dilemma*, a well-studied social dilemma. Each agent can choose to cooperate (C) or to defect (D). While mutual cooperation yields the second-highest reward for both ($r^i = 3$), each agent is individually incentivised to defect as it is the "dominant action", guaranteeing a higher reward compared to cooperating ($1 > 0, 5 > 3$). In fact, the joint action (D,D) is the *Nash equilibrium* of this game. A Nash equilibrium is a set of policies where no agent can unilaterally deviate from their policy to achieve a better outcome, given the policies chosen by the other agents (for a formal definition see section 2.2.2).

Repeated Normal-form Games

Normal-form games model single interactions between agents. To explore several sequential interactions of the same type, one introduces *repeated normal-form games*.[3] Here, the same normal-form game is played repeatedly for a finite or infinite number of times, which might induce different dynamics. The sequential interaction allows for the emergence of policies that differ from those in a single instance.

For example, this distinction transforms the Prisoner's Dilemma from a one-shot social dilemma into a strategic game of repeated interactions, called *iterated Prisoner's Dilemma*, where cooperation can emerge as a rational choice under certain conditions [43]. While in the one-shot game, rational self-interest dictates defection as the dominant policy, the iterated nature of the repeated game introduces the possibility of reciprocal cooperation. Agents may adopt cooperative policies as a means of fostering long-term mutually beneficial relationships, leveraging the threat of punishment for defection to incentivise cooperation and thereby achieve higher overall returns.

Given a normal-form game Γ, a repeated normal-form game Γ' plays Γ over $t_{max} \in [0, \infty)$ time steps. Γ is then called the *base game* of the repeated game Γ'. The total return, the cumulative rewards from each interaction, is now the metric by which an agent evaluates its performance. In repeated games, agents can leverage their past experiences to inform their current actions through access to the complete joint-action history, on which their policy is now conditioned. Even though agents are in general able to use the information of the complete history, policies might actually only use a fraction of it to inform their action choice. The famous "Tit-for-Tat" strategy in the iterated Prisoner's Dilemma exemplifies this concept [43]. Starting with cooperation in the first round, it then conditions its action solely on the opponent's most recent choice, cooperating if they cooperated and defecting if they defected.

[3]The terms *repeated* and *iterated* are often used interchangeably to describe games where a normal-form game is played multiple times. In this thesis, we treat them as synonymous.

It is important to note, that "endgame effects" may occur in *finitely* repeated games, if agents have access to the information of how many rounds are left to play. For instance, in the iterated Prisoner's Dilemma, if the number of iterations is predetermined and the agents know how many, the rational strategy is conditioned on this information. As both agents anticipate the other's defection in the last round, there's no incentive to cooperate in the second-to-last round either. This logic extends backward throughout the interaction, suggesting a rational choice of defection throughout the entire sequence [44]. Due to this characteristic of fixed-horizon games, the focus of this thesis is on repeated games with indefinite horizons. The agents are unaware of the number of rounds left to play, and are unable to adapt their policy to this information.

Infinitely repeated games introduce a discount factor γ similar to RL, reflecting the diminishing value of future rewards. Besides the former written aspect that a discount factor could reflect the preference of an agent to value long-term future rewards, it may also be interpreted in a different way: $(1-\gamma)$ can be seen as the (perceived) probability in each time step for the game to terminate. Such a game would still count as "infinite" since any number of maximal time step t_{max} has a non-vanishing probability of occurring [12].

All games investigated in this thesis are restricted to repeated normal-form games. However, for the sake of completeness, we include a brief discussion of more complex game forms, such as stochastic and partially observable stochastic games. These game forms are mentioned to provide context and highlight potential extensions.

Stochastic Games

A repeated normal-form game is still static in the sense that it has only a single non-terminal environmental state, defined by the base game in which the agents stay throughout the game, and a single terminal state to which the environment transitions when the game ends. Moving beyond the static nature, *stochastic games* introduce a dynamic environment. In stochastic games, the environment can be in one of multiple non-terminal states. The rules defining the transition probabilities in-between states, may be based both on the actions taken by agents and a random chance element. Stochastic games allow agents to observe the full environmental state and the complete joint-action history. These games can terminate after reaching a terminal state, a predefined number of steps, or continue indefinitely. Notably, both repeated normal-form games (multiple agents with a single environment state) and Markov Decision Processes (single agent with multiple environmental states) are special cases of stochastic games.

Definition 2.2.2 (Stochastic game)**.** A *stochastic game* consists of:

- a finite set of agents $\mathcal{I} = \{1, 2, \ldots, N\}$, with $N \in \mathbb{N}$,
- a finite set of states $\mathcal{S}$, with a subset of terminal states $\overline{\mathcal{S}} \subset \mathcal{S}$,
- a finite set of actions $\mathcal{A}^i$ for each agent i,
- individual reward functions $R^i : \mathcal{S} \times \boldsymbol{\mathcal{A}} \times \mathcal{S} \to \mathbb{R}$, where $\boldsymbol{\mathcal{A}} = \mathcal{A}^1 \times \cdots \times \mathcal{A}^N = \prod_i \mathcal{A}^i$ is the joint action space.
- a state transition function $T : \mathcal{S} \times \boldsymbol{\mathcal{A}} \times \mathcal{S} \to [0, 1]$ such that

$$\forall s \in \mathcal{S}, \mathbf{a} \in \boldsymbol{\mathcal{A}} : \sum_{s' \in \mathcal{S}} T(s'|\mathbf{a}, s) = 1 \tag{2.19}$$

- Initial state distribution $\mu : \mathcal{S} \to [0, 1]$ such that $\sum_{s \in \mathcal{S}} \mu(s) = 1$, $\forall s \in \overline{\mathcal{S}} : \mu(s) = 0$.

Partially Observable Stochastic Games

Partially observable stochastic games (POSGs) further increase the complexity by introducing limitations on how agents perceive the environment. In POSGs, agents don't have direct access to the true environmental state. Instead, they rely on individual observations generated through

observation functions. These functions depend on the actual state and the past joint actions of all agents. This injects a layer of uncertainty, as agents must make decisions based on potentially incomplete and noisy information. All the game models covered previously, including MDPs, are special cases of POSGs.

2.2.2 Solution Concepts

So far we have introduced the individual goals of agents (to maximise their expected return) and multi-agent environments, referred to as games, which define the interaction rules. In this section we will look at the system as a whole and define solution concepts, analysing what kind of behaviour is to be expected under the game-theoretic assumption that agents behave rationally. The underlying questions here are: Does a solution necessarily exist within a game? Is a solution unique or are there multiple, perhaps even an infinite number?

In essence, a solution is a joint policy $\boldsymbol{\pi} = (\pi^1, \ldots, \pi^N)$ that satisfies specific criteria. These criteria are based on the expected returns each agent would receive under this joint policy and the relationships between those returns. Note, that the solution concepts and their properties discussed here assume finite game models, meaning a finite number of agents and finite state, action, and observation spaces. Taken together, the combination of a game model and a solution concept defines a learning problem in MARL [12].

Definition of the Expected Return in the Context of Multi-Agent Systems

To define a solution concept, we first need to define the expected return given a joint policy. We base our definition on chapter 4.1 in [12] and restrict it to stochastic games.[4] For single-agent RL we defined the reward function in section 2.1.1 as

$$R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}.$$

and the expected return in time step t as

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k}.$$

Taking into account that for multi-agent systems, the rewards are in general a function depending on the joint action $\boldsymbol{A} \in \boldsymbol{\mathcal{A}}$ of all agents, we introduce

$$\boldsymbol{R} : \mathcal{S} \times \boldsymbol{\mathcal{A}} \times \mathcal{S} \to \mathbb{R}^N,$$

where $R^i(S, \boldsymbol{A}, S')$ is the individual reward of agent i. Further, let

$$H_t := (S_0, \boldsymbol{A}_0, ..., S_{t-1}, \boldsymbol{A}_{t-1}, S_t) \tag{2.20}$$

denote the *full history* up to time t.[5] Note that histories do not have to contain rewards, as this information can be recovered if $\boldsymbol{R}$ is known, given the sequence of states and actions. $\mathcal{H}_t$ is the set of full histories up to t. A realised history h_t is denoted in lowercase. Now we can define the full return for agent i, given the complete history $H_{t_{max}}$ as

$$G^i(H_{t_{max}}) := \sum_{k=0}^{t_{max}-1} \gamma^k R^i(S_k, \boldsymbol{A}_k, S_{k+1}), \tag{2.21}$$

and the expected return for a given joint policy as

$$\begin{aligned} g^i(\boldsymbol{\pi}) &:= \mathbb{E}_{H_{t_{max}} \sim (\mu, \boldsymbol{\pi}, T)} \left[G^i(H_{t_{max}}) \right] \\ &= \sum_{h_{t_{max}} \in \mathcal{H}_{t_{max}}} Pr(H_{t_{max}} = h_{t_{max}} | \boldsymbol{\pi}) G^i(h_{t_{max}}), \end{aligned} \tag{2.22}$$

[4] For a definition encompassing PSOGs, see chapter 4.1 in [12]

[5] From a physics perspective, it might be more appropriate to use the term 'trajectory' instead of 'history' to denote the path followed by a system through phase-space and time. This change in terminology would more clearly emphasise that we are referencing a hypothetical path along which the game could be played, rather than past instances in a temporal sense. However, we adhere to the term 'history' for consistency with the convention used in [12].

where the probability for a specific complete history $h_{t_{max}}$ to occur under the joint policy is

$$Pr(h_{t_{max}}|\boldsymbol{\pi}) = \mu(s_0) \prod_{t=0}^{t_{max}-1} \boldsymbol{\pi}(s_t, \boldsymbol{a}_t) T(s_{t+1}|s_t, \boldsymbol{a}_t). \tag{2.23}$$

Having established an expression for (2.22), we can now define solution concepts.

Best Response

The concept of a *best response* revolves around identifying the most advantageous policy for an agent given the policies of others. A best response policy is one that yields the highest expected return for an agent, taking into account the policies chosen by all other agents. In short, the best response reflects an agent's rational reaction to the policies of others.

Definition 2.2.3 (Best Response)**.** Let Γ be a game with a finite set of $N \in \mathbb{N}$ agents $\mathcal{I}$, a set of finite actions for each agent $\mathcal{A}^i$, and a reward function for each agent $R^i : \boldsymbol{\mathcal{A}} \to \mathbb{R}$, where $\boldsymbol{\mathcal{A}}$ denotes the joint action space. A joint policy $\boldsymbol{\pi} = (\pi^1, ..., \pi^N)$ in this context is denoted by the tuple (π^i, π^{-i}) where $\pi^i \in \Pi^i$ represents the policy chosen by agent i and π^{-i} represents the joint policies of all other agents except agent i. Π^i is the set of all possible policies of agent i. The set of *Best Response policies* $BR^i(\pi^{-i})$ of agent i to π^{-i} is defined as

$$BR^i(\pi^{-i}) = \arg\max_{\pi^i} g^i(\pi^i, \pi^{-i}). \tag{2.24}$$

This set might be singular or contain a whole continuum of policies.

Nash Equilibrium

One of the most central solution concepts in game theory is the *Nash equilibrium*. Introduced by John Nash in his famous 1950 paper *"Equilibrium Points in N-Person Games"*, it represents a joint policy where no agent has an incentive to deviate from its current policy, assuming the policies of all other agents remain fixed. Each individual policy in a Nash equilibrium is the best response to the other agents' policies in that equilibrium.

Definition 2.2.4 (Nash Equilibrium)**.** Let Γ be a game according to def. 2.2.3. The joint policy $\boldsymbol{\pi}$ is a called a *Nash equilibrium* if

$$\forall i, \pi^{i'} : g^i(\pi^{i'}, \pi^{-i}) \leq g^i(\boldsymbol{\pi}). \tag{2.25}$$

Nash proved that at least one such equilibrium exists for any game with a finite set of agents and a finite set of pure actions. There can be multiple or even infinitely many Nash equilibria, but at least one is always guaranteed. For example, consider the (non-repeated) matrix games depicted in table 2.1.

The only Nash equilibrium of the Prisoner's Dilemma is (D, D), as D is the strictly dominant action. Regardless of what the other agent chooses, defection outperforms cooperation. Although collectively, they would be better off if they both cooperate, each agent could improve its reward by changing to defection. If both defect, they cannot improve by changing to cooperation.

Matching Pennies has no pure Nash equilibrium. "Pure" refers to a policy which assigns a probability of 1 to a certain action. Instead, the unique equilibrium of this game is a joint policy in which each player randomly selects H or T, each with a probability of 0.5.

Bach-Stravinsky has three Nash equilibria. Two pure ones, where both agents choose the same action, and one probabilistic equilibrium. If the row-agent chooses B with a probability of 3/5 and the column-agent plays S with a probability of 3/5, neither party can improve, given the policy of the other. This can be calculated by considering what policy one agent has to play to render its opponent indifferent to choosing one option over the other. The probability of agents miscoordinating is 13/25, resulting in an expected return of 6/5 for each agent, which is lower than the payoff of 2 from their less favoured pure strategy equilibrium.

The symmetric Stag Hunt game also has three Nash equilibria. The two pure ones and one probabilistic, $(\pi_S^1, \pi_S^2) = (2/3, 2/3)$.

These simple examples demonstrate that there exists always at least one Nash equilibrium for finite games, possibly in probabilistic policies. Games may have a unique equilibrium solution, or they may have multiple (even infinitely many) equilibria that can yield different expected returns for the agents. Thus, learning an equilibrium solutions in a multi-agent setting is not necessarily the same as maximising the expected returns for all agents.

This prompts significant questions for MARL: Does the collective learning process reliably converge toward a Nash equilibrium, and if multiple exist, which one does it converge to? Furthermore, is it possible for MARL to discover novel Nash equilibria in more complex game structures where traditional approaches might fall short? In the next section, we review a branch of game theory that has inspired the design of several MARL algorithms aimed at addressing these questions.

2.2.3 Evolutionary Game Theory

This section follows closely the introduction to evolutionary game theory outlined in [42].

Unlike classical game theory, which assumes perfect knowledge and rationality in order to deduce optimal policies, *evolutionary game theory* leverages biological concepts like natural selection and mutation [46]. This approach focuses on the replicator dynamics (2.26), which describe how populations of individuals evolve over time based on their fitness in interactions.

Each individual belongs to a specific type. In each interaction, individuals are randomly paired and their reproductive success is determined by their fitness, which results from these interactions. Individuals with higher fitness compared to the population average see their population share increase, while those with lower fitness become less prevalent. This population evolution can be modelled by the state vector $\mathbf{x} = (x_1, x_2, ..., x_n)$, with $0 \leq x_a \leq 1$ for all a and $\sum_{a=1}^{n} x_a = 1$, representing the fraction of the population belonging to each of n types. The fitness of type a is given by the fitness function $f_a : \mathbb{R}^n \to \mathbb{R}$. The average fitness of the population $\overline{f} : \mathbb{R}^n \to \mathbb{R}$ is given by $\overline{f}(\mathbf{x}) = \sum_{b=1}^{n} x_b f_b(\mathbf{x})$. The population change over time is governed by the replicator dynamics,

$$\dot{x_a} = x_a \left[f_a(\mathbf{x}) - \overline{f}(\mathbf{x}) \right]. \tag{2.26}$$

Beyond describing the change over time of a large population of individuals, this model can also be interpreted differently in the context of normal form games. Here, the population vector $\mathbf{x}$ represents the policy $\boldsymbol{\pi}$ of a single agent, with x_a corresponding to the probability π_a of choosing the pure action a. Under this interpretation, the replicator dynamics illustrate how the agent's policy evolves through repeated gameplay and iterative adjustments.

For a two-agent two-action game, each agent i is described by its own population. During each game iteration, one individual from each agent's population is drawn according to its population distribution $\boldsymbol{\pi}^i = (\pi_1^i, \pi_2^i)$ to play its action in the "role" of that agent in an interaction with the other drawn individual. The fitness of each type π_a^i now depends on the population distribution $\boldsymbol{\pi}^{-i}$ of the opponent $-i$, meaning that the two populations are co-evolving.

Consider the Prisoner's Dilemma for instance. The reward tensor is given by

$$\mathbf{R} = \begin{pmatrix} R_{CC}^1, R_{CC}^2 & R_{CD}^1, R_{CD}^2 \\ R_{DC}^1, R_{DC}^2 & R_{DD}^1, R_{DD}^2 \end{pmatrix}. \tag{2.27}$$

The expected reward and average policy reward of action $a \in \{C, D\}$ for agent 1 under $\boldsymbol{\pi}^1$ can be viewed as the fitness and average fitness, and expressed as

$$f_a(\pi^1) = \sum_b \pi_b^2 R_{ab}^1,$$
$$\overline{f}(\pi^1) = \sum_a \pi_a^1 \sum_b \pi_b^2 R_{ab}^1.$$

Similar expressions hold for agent 2. The corresponding replicator dynamics can then be used to analyse the policy space. Figure 2.3 shows the vector fields of the policies for the games of table 2.1. It reveals the stability of the Nash equilibria derived in section 2.2.4.

To provide a broader overview, we note that evolutionary game theory introduces an additional solution concept: In *symmetric* two-player games, where both agents have the same action space and the bimatrix is invariant under exchanging rows and columns and first and second entries, $R^1(a^1, a^2) = R^2(a^2, a^1)$, evolutionary game theory also studies an alternative dynamical model in which both players are drawn at random from *the same* population, and consequently the type distribution in that population is described by a *single* policy π. That model can then be used to further refine the concept of Nash equilibria with the notion of *evolutionarily stable policies* [42, 46].[6]

Definition 2.2.5 (Evolutionarily Stable Policy)**.** An *evolutionarily stable policy* for a symmetric two-player game, denoted by $\pi = (\pi_1, \pi_2, ..., \pi_n)$, is a policy that is immune to invasion by mutant policies in the context of evolutionary game theory. Formally, for any mutant policy π', the following conditions must hold:

1. $f(\pi, \pi) \geq f(\pi', \pi)$, i.e., mutants perform no better than incumbents against incumbents
2. If $f(\pi, \pi) = f(\pi', \pi)$, then $f(\pi, \pi') > f(\pi', \pi')$, i.e., mutants perform worse than incumbents when playing against mutants

where $f(\pi, \pi')$ denotes the fitness or expected reward of policy π when playing against policy π'.

Importantly, every evolutionarily stable policy is also an asymptotically stable fixed point of the replicator dynamics [47]. The replicator dynamics of the Prisoner's Dilemma lead to the (D,D) equilibrium, which is both a Nash equilibrium and an evolutionarily stable policy. The dynamics of the Matching Pennies game exhibit closed orbits around the mixed Nash equilibrium (0.5, 0.5), which is only Lyapunov-stable but not asymptotically stable, and is thus not an evolutionarily stable policy. The policies stay on a closed orbit and never converge to the Nash equilibrium, unless they already start there. In Stag Hunt, both pure equilibria are evolutionarily stable policies, but the mixed Nash equilibrium is not, as it is not asymptotically stable, in fact not even Lyapunov-stable but rather a saddle point of the dynamics. Since the Bach-Stravinsky game is not symmetric, evolutionary stable policies according to definition 2.2.5 are not defined. Nonetheless, we note that the mixed equilibrium is a saddle point of the replicator dynamics.

[6] Note that in the literature they are usually termed *evolutionarily stable strategies*. We make the change to 'policies' to stay consistent with the naming conventions throughout this thesis. The relationship between terminology used in reinforcement learning, game theory, and evolutionary game theory is presented in table 2.2.

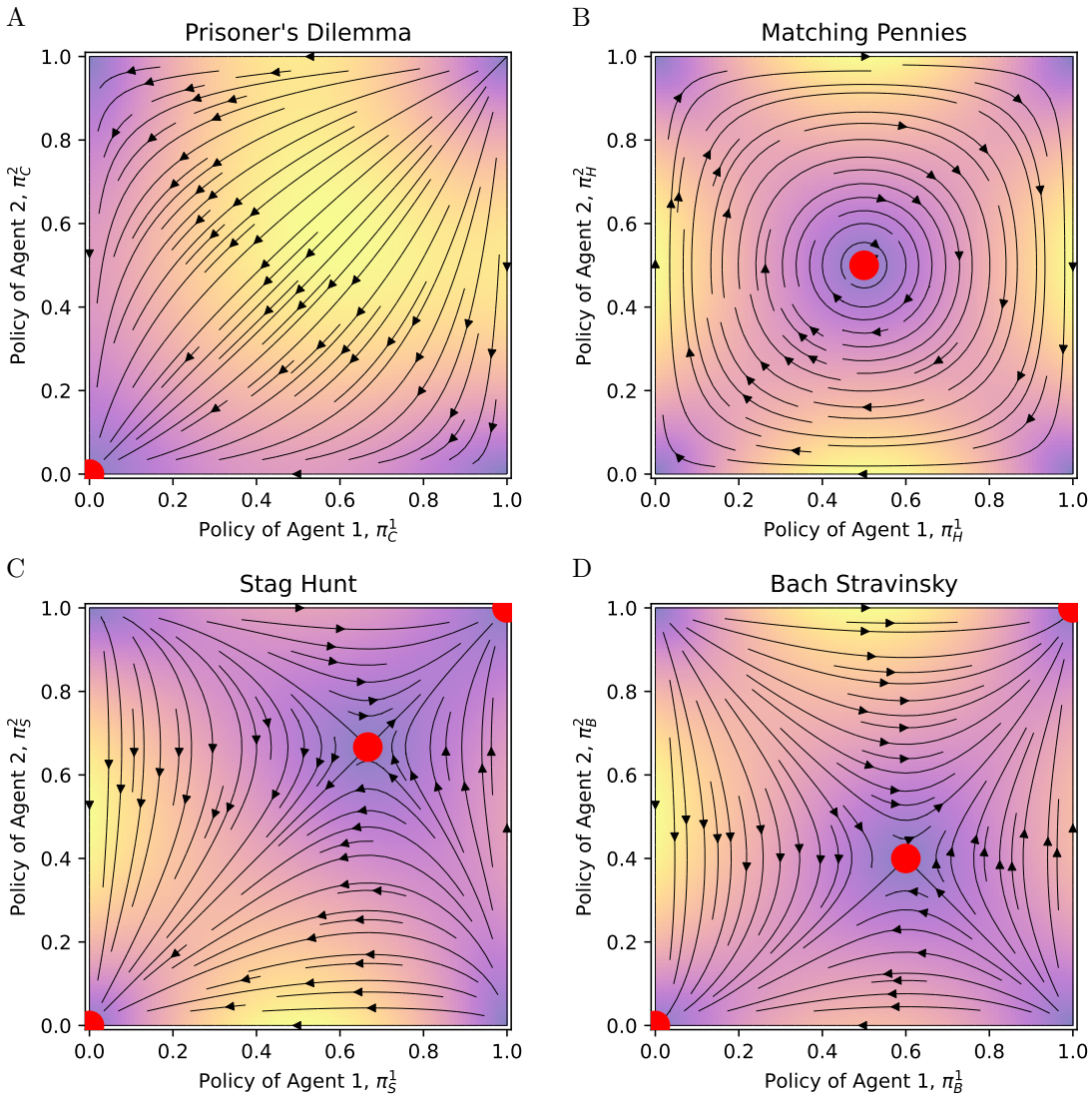


Figure 2.3: Replicator dynamics in policy space for the matrix games defined in table 2.1. The abscissa is the probability of the first agent to choose the first action, the ordinate is the probability of the second agent to choose the first action. The colour gradient shade indicates the magnitude of policy change, normalised from zero (purple) to the maximum (yellow). The streamlines show the direction of the policy change. The red dots show the Nash Equilibria of the games. (A): Prisoner's Dilemma. NE: (0,0) (B): Matching Pennies. NE: (1/2, 1/2) (C): Stag Hunt. NE: (0,0), (1,1), (2/3, 2/3) (D): Bach-Stravinsky. NE: (0,0), (1,1), (3/5,2/5)

Reinforcement Learning	Game Theory	Evol. Game Theory
environment	game	game
agent	player	population
action	action	type
prob. π_a of choosing a	prob. π_a of choosing a	population share of type a
policy π	strategy π	distribution over types
reward, return	payoff, utility	fitness
deterministic π	pure π	monomorphic population
probabilistic π	mixed π	polymorphic population
joint π	π profile	population state
exploration-exploitation	(risk-reward)	mutation-selection
(learning rate)	(adaptation rate)	(mutation rate)

Table 2.2: Comparison of terms in Reinforcement Learning, Game Theory, and Evolutionary Game Theory. Parentheses indicate an analogy rather than an equivalence in terminology. Adapted from [12], [42] and [48].

2.3 Multi-Agent Reinforcement Learning (MARL)

In the first section, we examined classical RL in single-agent environments, where a singular agent learns an optimal policy through iterative interactions with an environment to achieve predefined goals. Then, we have explored games as a formal representation of interactions among multiple agents, along with solution concepts that define optimal behaviour under the assumption of rationality. Here, we now introduce the ability of "learning", or rather adapting, to sequentially interacting agents, marking the transition to Multi-Agent Reinforcement Learning (MARL). Unlike the typical game-theoretic assumption of perfect information, agents have no prior knowledge of the game. They can only acquire information about the environment and other agents through interactions.

In this section, we define a general learning process in a multi-agent environment and present two fundamental MARL methods, *independent* and *joint-action* learning. Further, we address the challenges and complexities that arise from having multiple learning agents instead of one.

2.3.1 General Learning Process

We start by reproducing the definition of a general learning process in MARL as described in [12], which involves a game model, multiple time scales, data, an algorithm and a chosen learning goal:

Game Model: The game model defines the environment in which the agents interact with each other.

Time Scales: In MARL, several key terms are commonly used to describe the different time scales of the learning process. Although the fundamental terminology is well-established, the specific interpretation can vary. To ensure clarity, we define the following terms:

- *time step* t: A time step is a single instance of interaction between the agents and the environment.
- *episode* e: An episode is a sequence of time steps that starts from an initial state and ends when a terminal state is reached. It represents one complete trial of the agents interacting with the environment. After each episode, the agents may reset to a new initial state to begin a new episode.
- *update interval* u: The period between policy updates, which can span several time steps or an entire episode. For example, in *batch learning* [49], agents collect experiences over

multiple interactions and perform summary statistics on this data to update their policies. An agent navigating a maze might update its policy only after reaching a terminal state and evaluating its overall performance for the entire episode. In general, each agent may have its own update interval, updating independently or in coordination with other agents depending on the learning algorithm used. In this thesis, unless specified otherwise, we consider policy updates that occur at every time step t.

- A *run* typically refers to the entire process of training the agents, which may include many episodes. It encompasses the overall learning experience, from the beginning of training until a certain stopping criterion is met, such as achieving a desired level of performance or reaching a maximum number of episodes.

Data: The data

$$\mathcal{D}_z \doteq \{H_{t_u} | u = 1, ..., z\}, \quad z \geq 0$$

is used for the training algorithm. It consists of z many histories H_{t_u}, each produced by a joint policy $\boldsymbol{\pi}_u$ used during a particular update interval u. Note that this definition only allows for agents with symmetric update intervals. An extension of the notation to individual update intervals is possible but would exceed the need for this thesis.

Learning Algorithm: The learning algorithm $\mathbb{L}$ takes the collected data $\mathcal{D}_z$ and current joint policy $\boldsymbol{\pi}_z$, and produces a new joint policy,

$$\boldsymbol{\pi}_{z+1} = \mathbb{L}(\mathcal{D}_z, \boldsymbol{\pi}_z) \tag{2.28}$$

The initial policy $\boldsymbol{\pi}_0$ is predetermined, typically set to random.

Learning Goal: The goal of learning is to attain a joint policy $\boldsymbol{\pi}_*$ which fulfills criteria usually specified by a chosen solution concept like a Nash equilibrium. When evaluating learning performance in MARL algorithms, the key measure we focus on in this thesis is how a joint policy approaches a solution concept $\boldsymbol{\pi}_*$ of the game in the limit of infinitely available data,

$$\lim_{z \to \infty} \boldsymbol{\pi}_z = \boldsymbol{\pi}_*. \tag{2.29}$$

In real-world applications, gathering infinite data isn't feasible. Typically, learning terminates after reaching a predetermined limit on resources, such as the total number of time steps, or when changes in the policy fall below a predefined threshold—e.g., through annealing of the learning rate or the exploration parameter (e.g., epsilon or temperature) [44]. In practice, the learning process is typically halted when policy adjustments appear stable, marked by minimal variations and consistent behaviour over numerous time steps.

However, as we demonstrate in chapter 3, it is essential to exercise caution when interpreting such outcomes. One must scrutinise whether the learning dynamics truly reach an equilibrium or merely stabilises at an meta-stable state[7] that may eventually transition given sufficient time.

2.3.2 MARL Methods: Independent and Joint-Action Learning

The MARL methods used in this thesis fall under two categories: *independent learning* and *joint-action learning*. Additionally, when all agents use the same algorithm (and the same type of exploration policy, same learning parameters etc.), the method is given the attribute *self-play*. Conversely, *mixed-play* occurs when different learning algorithms are utilised by the agents. All methods used in this thesis are self-play methods.

Independent Learning: The *independent learning* approach [50] naturally extends single-agent algorithms to multi-agent settings by treating each agent's learning as an isolated process. The processes are "independent" insofar as they are only indirectly linked to other agents' learning processes via the shared environment. Each agent i learns its policy π^i based solely on its

[7] Note that 'state' does not refer here to environmental or internal states of the RL process but to dynamical states of the learning process.

Algorithm 2: Independent Q-Learning with Boltzmann Policy

Input: State space $\mathcal{S}$, action space $\mathcal{A}^i$, learning rate α^i, discount factor γ^i, temperature parameter T^i for each agent i
Output: Learned Q-values $Q^i(s,a)$ for each agent i
Initialise $Q^i(s,a)$ arbitrarily for all $s \in \mathcal{S}, a \in \mathcal{A}^i$, except $Q(s \in \bar{\mathcal{S}}, \cdot) = 0$, for each agent i
while *not converged* **do**
  Choose initial state s
  while *not reached terminal state* **do**
    for *each agent* i **do**
      Choose action a^i with Boltzmann policy:
      $\pi^i(a^i|s) \leftarrow \frac{e^{Q^i(s,a^i)/T}}{\sum_{b^i} e^{Q^i(s,b^i)/T}}$ for all $a^i \in \mathcal{A}^i$
      $a^i \sim \pi^i(\cdot|s)$
    end
    Take joint action $\boldsymbol{a} = (a^1, a^2, ..., a^n)$
    for *each agent* i **do**
      Observe own reward r^i and next state s'
      Update Q-value:
      $Q^i(s,a^i) \leftarrow Q^i(s,a^i) + \alpha^i \left[r^i + \gamma^i \max_{b^i} Q^i(s',b^i) - Q^i(s,a^i) \right]$
    end
    Update state: $s \leftarrow s'$
  end
end

own local history, without using information about other agents. This means that an agent only observes its own past observations, actions, and rewards, but not, for example, the past actions of other agents.

An example of a scenario where each agent uses the same Q-learning algorithm, known as independent Q-Learning, is provided in algorithm 2.

A key benefit of independent learning is the ease with which single-agent learning algorithms can be adapted for multi-agent environments, ensuring that the system can scale with the number of agents. From each agent's perspective, the actions and policies of other agents are simply perceived as part of an evolving environment. However, a drawback is the lack of any explicit coordination mechanism among the agents.

Joint-Action Learning: Unlike independent learners, which disregard the existence of other agents, *joint-action learners* specifically consider them [51]. A key advantage of joint-action learning lies in its ability to facilitate coordination among agents. However, this approach suffers from a scalability issue. The complexity of the algorithm increases exponentially with the number of agents, making it computationally challenging for large agent populations.

The most fundamental framework for illustrating how joint-action learning algorithms work lies in repeated normal-form games (detailed in Section 2.2.1). These games offer a static environment where agents interact repeatedly. At each time step, the environment grants agents access to the complete joint-action history $h_t \in \mathcal{H}$, from which they extract individual observations $o_t^i = f(h_t)$ by an internal function $f : \mathcal{H} \to \mathcal{O}$. Based on these observations, an agent determines its policy $\pi^i(\cdot|o_t)$. While observations function similarly to states as defined in the single-agent case (as discussed in Section 2.1), it's important to note that they are internal to each agent and are not properties of the environment. Therefore, they are also referred to as *internal states* to distinguish them from the external states that describe the environment. Each agent may have its own individual internal states.

To clarify the concept of internal states, consider the example of the Iterated Prisoner's Dilemma. The number of internal states of a Tit-for-Tat agent is two because only the opponents past action is of interest, not the agents own. In general, a policy might consider the past m

Algorithm 3: Joint-Action Q-Learning with Boltzmann Policy for Repeated Normal Form Games

Input: Observation space $\mathcal{O}^i$, action space $\mathcal{A}^i$, learning rate α^i, discount factor γ^i, temperature parameter T^i for each agent i, maximum time step t_{max}
Output: Learned Q-values $Q^i(o^i, a)$ for each agent i
Initialise $Q^i(o, a)$ arbitrarily for all $o \in \mathcal{O}^i, a \in \mathcal{A}^i$ and for each agent i
for *each agent* i **do**
  Choose initial observations o^i
end
$t = 0$
while *not reached* t_{max} **do**
  for *each agent* i **do**
    Choose action a^i with Boltzmann policy:
    $\pi^i(a^i|o^i) \leftarrow \frac{e^{Q^i(o^i,a^i)/T}}{\sum_{b^i} e^{Q^i(o^i,b^i)/T}}$ for all $a^i \in \mathcal{A}^i$
    $a^i \sim \pi^i(\cdot|o^i)$
  end
  Take joint action $\boldsymbol{a} = (a^1, a^2, ..., a^n)$;
  for *each agent* i **do**
    Observe own reward r^i and next observation $o^{i'}$
    Update Q-value:
    $Q^i(o^i, a^i) \leftarrow Q^i(o^i, a^i) + \alpha^i \left[r^i + \gamma^i \max_{b^i} Q^i(o^{i'}, b^i) - Q^i(o^i, a^i) \right]$
    Update observation: $o^i \leftarrow o^{i'}$
  end
  $t \leftarrow t + 1$
end

recent joint actions $(\boldsymbol{A}_{t-m}, ..., \boldsymbol{A}_{t-1})$ to decide on cooperation or defection. For a memory length[8] of $m = 1$, so only the most recent interaction, this results in four internal states,

$$DD, DC, CD, CC,$$

one for each possible combination of actions. For $m = 2$, there are already 16 possible combinations (DD-DD, DD-DC, ...). Here we can see the curse of dimensionality come into play which was mentioned in Section 2.1.3. The number of possible internal states z blows up as the memory length increases with

$$z = \big((\#\text{actions per agent})^{\#\text{agents}}\big)^{\#\text{memory length}} = (2^2)^m.$$

Alternatively, observations of the joint-action history could utilise summary statistics, like the number of times other agents cooperated in the past m interactions, to construct more abstract internal states.

Each agent might use different observations. For instance, one agent might use the information of the joint actions taken in the past two time steps, whereas its opponent has a memory length of only one time step, giving the first agent an information advantage about the second.

A pseudo code for joint-action Q-Learning for a repeated normal form game is given by algorithm 3. Note that it is similar to algorithm 2, with the difference that the internal observations o^i take the place of the environmental states s.

[8]The term 'memory' might be somewhat misleading. Describing an agent as having zero 'memory'—corresponding to independent Q-learning—could incorrectly suggest an inability to retain past experiences. However, this is not the case in ongoing learning processes, where agents implicitly store information from past interactions in their learned policies. Here, 'memory' refers to the agent's ability to use explicit information from prior observations to refine its decision-making process. For instance, an agent with a memory length of one can develop a more nuanced and refined policy compared to an agent with no access to past observations. While a more precise term might be 'observation length', we adhere to the convention of using 'memory'.

2.3.3 Challenges of MARL

MARL algorithms are faced with unique challenges that go beyond those encountered in single-agent settings, due to the interactions among multiple learning agents. Among others, the most pertinent challenges are outlined below.

Non-Stationarity and the Moving-Target Problem: A stochastic process $\{X_t\}_{t\in\mathbb{N}_0}$ is considered stationary if the joint probability distribution of $(X_{t_1+\tau}, \ldots, X_{t_n+\tau})$ is the same as that of $(X_{t_1}, \ldots, X_{t_n})$ for all $t_1, \ldots, t_n, \tau$. This implies that the process exhibits consistent behaviour over time. In the context of RL, an environment is non-stationary if the rules governing state transitions and rewards change over time. The optimal policy derived for one set of environment conditions may become suboptimal if these conditions change. This means that the learning target for an agent is constantly changing, known also as the *moving target problem*.

Single-agent temporal difference algorithms are guaranteed to converge to an optimal policy, provided that the environment is Markovian (and thus, in particular, stationary) and the agent is permitted to explore a sufficient range of actions (see Section 2.1.4). However, when multiple adaptive agents interact, each agent's learning process alters the environment as perceived by other agents, making it non-stationary. This disrupts the Markov property and the convergence guarantees of single-agent algorithms do not hold in MARL. Consequently, each agent faces a moving target problem, where the learning objectives are continually influenced by the evolving policies of other agents [52].

This raises important questions: Can multiple RL agents still converge to a solution over time? If so, under what conditions? And if not, what factors contribute to the failure of convergence?

Equilibrium Selection: Recall from Section 2.2.2 that games can have multiple solutions, each offering potentially different expected returns to the agents. The notion of equilibrium selection addresses the challenge of which equilibrium agents will adopt in a game with multiple possibilities [53]. As an observer it might not be transparent why a MARL process might converge to one equilibrium over the other and what are the underlying dynamics governing the outcomes.

Scalability: A major hurdle in MARL is achieving efficient scalability to a large number of agents. This challenge stems from the curse of dimensionality, where the number of possible joint actions grows exponentially with the number of agents. This exponential growth significantly impacts the algorithms in various ways. Algorithms relying on joint action-values, such as the observation of previous played joint actions in Repeated Normal-form Games, face a double burden: an exponential increase in both the space required to represent the Q-values and the number of observations needed to learn effectively. But even approaches that are not based on joint values (e.g., independent learning) are not immune. More agents exacerbates the perceived environment non-stationarity, as each additional agent introduces another dynamic element, requiring other agents to constantly adapt, making the learning process more complex.

2.4 Dynamical Systems Theory — the Linkage between MARL and Physics

Having established the basics of MARL, we shift to its analysis through a dynamical systems theory perspective. Dynamical systems theory focuses on how systems evolve over time in response to internal and external influences. It encompasses the study of systems governed by both linear and nonlinear differential (or difference) equations, with nonlinear dynamics representing the more complex and arguably more intriguing field. Unlike linear systems, which adhere to the principles of superposition and proportionality, nonlinear systems exhibit intricate and often unexpected behaviours due to the interplay of multiple factors and feedback mechanisms.

Central to dynamical systems theory is the concept of attractors, which represent the long-term behaviour of a system as it evolve towards equilibrium, recurring patterns, or chaos. Attractors can take various forms, such as fixed points, limit cycles, and strange (chaotic and non-chaotic) attractors. Structural stability analysis aims to determine a system's attractors by examining how small perturbations from an equilibrium solution evolve, and to identify potential bifurcations that may signal stability transitions.

A fundamental tool in this study is a linear stability analysis, which examines the dynamics in the close vicinity of a fixed point by linearising the system's equations through a first-order Taylor expansion. This approximation yields the Jacobi matrix, which quantifies how small perturbations evolve over time. The stability of the fixed point is determined by the eigenvalues of the Jacobian: for discrete-time systems, the fixed point is stable if all eigenvalues have magnitudes less than 1. Note, however, that if any eigenvalue has magnitude exactly 1, the linear analysis may fail to capture the system's behaviour, requiring higher-order methods. Such non-hyperbolic fixed points indicate more complex dynamics like bifurcations or center-type dynamics. For a more detailed introduction, we refer to [54] or any other standard textbook on the subject.

Dynamical systems theory finds applications in a wide array of scientific disciplines, including physics, chemistry, geophysics, neuroscience, social sciences, ecology, economics, evolutionary game theory, control theory, and machine learning [54]. In the realm of physics, it is employed across diverse fields, ranging from the microscopic domain of quantum mechanics to the cosmic scales of astrophysics. In quantum mechanics, it underpins the understanding of complex quantum phenomena such as quantum chaos [55]. In physical chemistry and biophysics, nonlinear models are pivotal in explaining diverse oscillatory systems, such as the Belousov–Zhabotinsky chemical reaction [56] and neuron spiking dynamics [57]. In climate science, it offers insights into the dynamics of weather patterns, fluid dynamics, and climate change, including the identification of tipping points that can lead to abrupt and irreversible shifts in the climate system [58]. In astrophysics, nonlinear dynamics is applied to study phenomena like chaotic motion in multi-body systems and galactic dynamics among others [59]. These diverse applications underscore the versatility and significance of dynamical systems theory in understanding and predicting the behaviour of complex systems across various scales and disciplines.

In the context of MARL, dynamical systems theory provides a framework for analysing the underlying learning dynamics. The interdependence of agents and the inherent stochasticity of the algorithms in MARL often lead to complex emergent dynamics. Here, a complementary perspective from dynamical systems theory has proven beneficial [22, 30, 31, 42, 48, 60–79]. To this end, the stochastic algorithms are approximated in deterministic dynamical equations[9], which allows for a concise interpretation and provides a framework for a rigorous analysis of the effects of parameters and initial conditions. By illuminating the mechanisms driving collective behaviour, a dynamical systems theory approach advances the development of more effective and transparent learning algorithms. The aspect of transparency is particularly critical, as it addresses concerns about the potential risks associated with deploying complex machine learning algorithms in real-world decision-making scenarios where their underlying dynamics may not be fully understood [18].

Most existing deterministic approximation models of MARL algorithms are focused on tabular temporal difference learning using Boltzmann policies and are limited to single-state environ-

[9] Note that dynamical systems theory can also be applied to stochastic systems, though this is generally more complex.

ments (e.g., independent Q-Learning applied to normal-form games). In particular, a common approach is to take the continuous-time limit and link the resulting ordinary differential equations to the replicator dynamics of evolutionary game theory [30, 31, 48, 60, 61] (section 2.2.3). Based on these replicator-type models, Barfuss et al. presented a methodological extension to multi-state environments (e.g. joint-action Q-learning on repeated normal-form games or independent Q-learning on stochastic games) in 2019.

Beyond Boltzmann policies, which offer advantages for dynamical systems analysis due to their continuity in the Q-value space, some work also explores epsilon-greedy policies. For instance, Wunder et al. conducted a comprehensive study on epsilon-greedy Q-Learning applied to various normal-form game classes [67]. Their work provides convergence proofs (or lack thereof) for each class, demonstrating a range of behaviours from rapid convergence to stable oscillations.

Furthermore, it has been shown that not only in complex games [73] but even in seemingly simple games like "Rock, Paper, Scissors" [31], deterministic chaos may emerge from the approximated dynamics, raising concerns for the application of MARL algorithms on critical real-world tasks. Such chaotic dynamics may lead to unpredictable and potentially unstable behaviour, posing a challenge for the development of robust and reliable MARL methods. But even in systems with stable non-chaotic attractors, studies have emphasised the distinction between approximated dynamics and actual behaviour of stochastic algorithms [22]. Relaxing the assumptions that allow to take a deterministic limit can significantly alter the learning dynamics, leading to phenomena like noise-induced oscillations that prevent convergence to a joint policy [30, 69].

In the next chapter, we work out in detail how a deterministic approximation model of a MARL algorithm may—or may not—be constructed, compare the behaviour of stochastic algorithms with their deterministic approximation models, and demonstrate how algorithmic details may induce unexpected behaviour.

Chapter 3

Deterministic Approximation Model of Independent Q-learning in Single-State Environments

The text and content of this chapter are largely identical to [32], currently under review, which is co-authored by myself, my supervisor Jobst Heitzig, and Wolfram Barfuss.

In this chapter, we demonstrate the utility of a dynamical systems theory approach for understanding the emergent behaviour in MARL. Using the example of independent Q-learning in single-state environments, we propose a deterministic approximation model that encapsulates the learning dynamics, and facilitates a rigorous analysis of how parameters and initial conditions influence the learning process.

In section 3.1, we reintroduce the independent Q-learning algorithm and the specific setup under study.

We then provide a historical overview of related work in section 3.2. In particular we review previous models, the FAQL [48] and the BQL model [74], and elaborate why they actually approximate interesting variants of independent Q-learning, rather than describing the fundamentally more complex dynamics of the original algorithm. To highlight the stylised discrepancies, we compare the deterministic dynamics with the actual stochastic learning process in the context of the Prisoner's Dilemma.

In section 3.3, we propose an alternative deterministic approximation model for single-state environments and demonstrate its effectiveness in capturing the emergent behaviour of independent Q-learning. Our model explains why agents appear to "learn" to spontaneously cooperate over extended periods in the Prisoner's Dilemma and shows that this behaviour is not a true equilibrium but merely a metastable phase of the dynamics. It further shows how stable oscillations arise from the moving-target problem, preventing convergence under certain parameter settings, by inducing a supercritical Neimark-Sacker bifurcation.

We conclude with a discussion of the broader implications of our findings, emphasising the limitations of the independent learning approach and the need for caution when interpreting results.

Change of Notation — Emphasis on Time Evolution: To reflect the emphasis on time evolution, we change our notation for the remainder of this thesis to show the time dependency of Q and π in the argument and the state/action indices in the subscript.

3.1 Method

We study the simplest possible multi-agent setup: two agents interacting in a single-state environment, playing the Prisoner's Dilemma as the most paradigmatic example game. The Prisoner's Dilemma, a social dilemma characterised by a single Nash equilibrium where both agents defect, along with its iterated version, has been extensively studied in the MARL community [29, 44, 48, 64, 66, 68, 80–84]. The reward tensor for the game is given by

$$\mathbf{R} = \begin{pmatrix} R^1_{CC}, R^2_{CC} & R^1_{CD}, R^2_{CD} \\ R^1_{DC}, R^2_{DC} & R^1_{DD}, R^2_{DD} \end{pmatrix} = \begin{pmatrix} 3,3 & 0,5 \\ 5,0 & 1,1 \end{pmatrix}. \tag{3.1}$$

The agents can either choose to cooperate (C) or to defect (D). The superscript index i denotes an agent, while $-i$ represents its opponent. Random variables are denoted by uppercase letters (e.g. action A^i), their specific instances in lowercase (e.g. a^i), and tensors in boldface (e.g. joint action $\mathbf{A}$). At each time step t, each agent i chooses an action $A^i(t) = a^i \in \mathcal{A}^i$, where $\mathcal{A}^i = \{C, D\}$, receives a reward $R^i_{\mathbf{A}(t)}$ based on the joint action $\mathbf{A}(t) = (A^i(t), A^{-i}(t)) \in \mathcal{A}$, where $\mathcal{A} = \mathcal{A}^1 \times \mathcal{A}^2$, and updates its policy accordingly. The process repeats until a terminal time step is reached.

Formally, the environment consists of a single *non-terminal* state, s_0, defined by (3.1). After each time step, the environment transitions back to s_0. The learning process concludes at the terminal time step, at which point the environment transitions to a *terminal* state, $s_{terminal}$. By definition, no rewards are provided in the terminal state, and agents remain there indefinitely [12]. We note that this setup corresponds to the game-theoretic definition of a repeated normal-form game (section 2.2.1), where agents do *not* condition their policies on past interactions.

The agents adapt to new information via independent Q-learning (algorithm 4). The state-action value estimate $Q^i_{a^i}(t)$, called Q-value, represents how much agent i values action a^i at time t. The stochastic update rule reads

$$Q^i_{a^i}(t+1) = Q^i_{a^i}(t) + \alpha^i \delta_{A^i(t)a^i} \left[R^i_{\mathbf{A}(t)} + \gamma^i \max_{b^i \in \mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t) \right], \tag{3.2}$$

where $\alpha^i \in [0, 1)$ is called the agent's *learning rate* and $\gamma^i \in [0, 1)$ its *discount factor*. The policy $\pi^i_{a^i}(t)$ is the probability of agent i to choose action a^i at time t. It is drawn from a Boltzmann distribution

$$\pi^i_{a^i}(t) := f(Q^i(t), a^i), \tag{3.3}$$

$$f(Q^i, a^i) := \frac{\exp[Q^i_{a^i}/T^i]}{\sum_{b^i \in \mathcal{A}^i} \exp[Q^i_{b^i}/T^i]}. \tag{3.4}$$

where $T^i \in (0, \infty)$ is called *temperature* in analogy to statistical physics. Before we continue, we want to remark some comments on this setup and why it is of interest.

In the independent learning approach (section 2.3.2), each agent perceives the evolving policies of other agents as part of its "effective" environment, rendering it non-stationary. The independent learning approach is thus incompatible with the Markov Decision Process framework on which most convergence guarantees in single-agent algorithms rely (section 2.3.3). Despite this lack of convergence guarantees, independent learning is often used in practice due to its adaptability and scalability with the number of agents [17, 85]. These simple algorithms serve as crucial baselines in MARL research and can achieve individual cumulative rewards that are comparable with more sophisticated state-of-the-art methods [12, 86].

In the original single-agent Q-learning algorithm [39], the discount factor γ^i is a hyperparameter that determines an agent's preference for future state values in *multi-state* environments. The necessity of including a discount factor in a *single-state* environment, as considered here, is therefore debatable. Some studies effectively set $\gamma^i = 0$ by defining the environment to transition into a terminal state after each round [70, 71, 76, 79]. Others define the environment as static yet repetitive and keep the term involving γ [48, 67, 80, 87–89]. To preserve the original algorithm's core structure—where the term involving γ^i is a defining feature—we consider a repetitive environment and retain the discount factor. Given that the agents lack knowledge of when the game

Algorithm 4: Independent Q-learning with Boltzmann policy in a single-state environment

Input: Action space $\mathcal{A}^i$, learning rate α^i, discount factor γ^i, temperature parameter T^i for each agent i, common environment E
Output: Learned Q-values $Q^i_{a^i}$ for each agent i
Initialise $Q^i_{a^i}$ arbitrarily for all $a^i \in \mathcal{A}^i$ for each agent i
while *not reached terminal time step* **do**
  for *each agent* i **do**
    Choose action a^i with Boltzmann policy:
    $\pi^i_{a^i} \leftarrow \frac{e^{Q^i_{a^i}/T^i}}{\sum_{b^i} e^{Q^i_{b^i}/T^i}}$ for all $a^i \in \mathcal{A}^i$
    $a^i \sim \pi^i_{a^i}$
  end
  Take joint action $\mathbf{a} = (a^1, a^2, ..., a^n)$ in the environment E
  for *each agent* i **do**
    Observe own reward r^i in the environment E
    Update Q-value of chosen action:
    $Q^i_{a^i} \leftarrow Q^i_{a^i} + \alpha^i \left[r^i + \gamma^i \max_{b^i} Q^i_{b^i} - Q^i_{a^i} \right]$
  end
end

will end, our framework is consistent with the common interpretation of γ^i to be the agent's belief about the probability that the game continues in the next time step.

The Boltzmann policy function (section 2.1.3) is chosen over common alternatives like epsilon-greedy because it uses a smooth probability distribution based on Q-values rather than discrete choices. Some studies suggest this mechanism aligns with human and animal decision-making in competitive and observational learning tasks [90, 91]. The temperature parameter $T^i > 0$ regulates the exploration-exploitation trade-off: higher T^i promotes exploration by equalising probabilities, while lower T^i emphasises exploitation of actions with higher Q-values. As $T^i \to 0$, the agent converges to a pure policy. We keep the temperature constant throughout the learning process, rather than annealing it [44], to simplify the process and enhance the interpretability of the results.

The outcome of a Q-learning process is typically interpreted as a pure policy: the action with the maximum Q-value in a given state is regarded as the "learned" action. However, in this work, we focus on the dynamics of the learning process itself, interpreting the Boltzmann distribution as the "learned" policy at any time t, as it reflects the agent's probabilistic decision-making process. Our primary interest lies in understanding the long-term behaviour of the learning process as a function of parameters and initial conditions.

The dynamical state of the system is fully described by the four-dimensional vector in Q-space, $\mathbf{Q}(t) := (Q^1_C(t), Q^1_D(t), Q^2_C(t), Q^2_D(t))$. The four Q-values are the fundamental dynamical variables, evolving according to (3.2). In contrast, the joint policy $\boldsymbol{\pi}(t)$ resides in a two-dimensional subspace due to the normalisation constraint, $\sum_{a^i \in \mathcal{A}^i} \pi^i_{a^i}(t) = 1$, for all i and t. At any time t, this subspace is represented by $\boldsymbol{\pi}_C(t) = (\pi^1_C(t), \pi^2_C(t))$, capturing agents' cooperation probabilities. Thus, the Q-space encodes the full state of the system, while the policy space offers a lower-dimensional *projection*, indicating what the agents will actually do.

It is important to note that the use of the Kronecker delta $\delta_{A^i(t)a^i}$ in the update rule (3.2) implies that *only* the Q-value of the action $A^i(t)$ played at time t by agent i is updated, while the remaining Q-values retain their current values. As will be shown in the next section, this feature is critical to the algorithm's structure; neglecting or modifying it results in dynamics that diverge from the original formulation [39], that is commonly used [1, 12].

3.2 Previous Deterministic Models: Historical Context, Methodologies and Pitfalls

As discussed in section 2.4, the primary goal of constructing approximation models of MARL systems is to transform the stochastic algorithms into deterministic dynamical equations, either in discrete or continuous time, which enables a convenient analysis. A secondary goal may be to simplify the system to its essentials. For independent Q-learning, this could mean to reduce the dynamics from higher-dimensional Q-space into lower-dimensional policy space. In this section, we present two existing approximation models that take this step and explain why, in doing so, they deviate from the classic incremental algorithm—defined by (3.2)—and instead represent modified variants.

3.2.1 Cross-learning (CL) Model

Much of the research on multi-agent learning from a dynamical systems perspective builds on foundational work by Börgers and Sarin from 1997 [60], who first showed a formal relation between the replicator dynamics of evolutionary game theory (section 2.2.3) and MARL. Specifically, they proved that *Cross Learning* [92] converges to the replicator dynamics in the continuous-time limit.

Unlike Q-learning which updates estimates of the action-value function $Q^i_{a^i}(t)$, Cross Learning directly updates the action probabilities $\pi^i_{a^i}(t)$. This eliminates the need for a separate policy function for action selection, leading to a more intuitive connection to the replicator dynamics. In a two-agent setting, the update rule reads

$$\pi^i_{a^i}(t+1) = \pi^i_{a^i}(t) + \alpha^i \begin{cases} R^i_{\mathbf{A}(t)} - \pi^i_{a^i}(t) R^i_{\mathbf{A}(t)} & \text{if } A^i(t) = a^i \\ -\pi^i_{a^i}(t) R^i_{\mathbf{A}(t)} & \text{otherwise.} \end{cases} \tag{3.5}$$

The continuous-time limit is constructed by segmenting the time into intervals Δt, substituting the discrete time steps of the stochastic processes $(t+1)$ with $(t+\Delta t)$ and the learning rate α in the update rule with $\alpha \Delta t$, and taking the limit $\Delta t \to 0$. The limit transforms (3.5) into the deterministic coupled differential equations

$$\frac{d}{dt}\pi^i_{a^i}(t) = \alpha \pi^i_{a^i}(t) \left[\mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{a^i A^{-i}(t)} - \sum_{b^i \in \mathcal{A}^i} \pi^i_{b^i}(t) \mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{b^i A^{-i}(t)} \right], \tag{3.6}$$

which are a form of the replicator dynamics (2.26) of evolutionary game theory [42] (section 2.2.3). As can be seen in figure 3.1, learning trajectories of independent Cross learning follow the streamlines of the replicator dynamics (figure 2.3) for small learning rates ($\alpha = 0.001$).

3.2.2 Frequency-Adjusted Q-learning (FAQL) Model

Building on the foundation of [60], Tuyls et al. in 2003—and similarly Sato and Crutchfield for a slightly different variant—applied this approach to independent Q-learning using a Boltzmann policy in single-state environments. The authors proposed that the time evolution of an agent's policy in the continuous-time limit can be approximated by the deterministic replicator equation

$$\begin{aligned} \frac{d}{dt}\pi^i_{a^i}(t) = \frac{\alpha^i}{T^i}\pi^i_{a^i}(t) & \underbrace{\left(\mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{a^i A^{-i}(t)} - \sum_{b^i \in \mathcal{A}^i} \mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{b^i A^{-i}(t)} \right)}_{\text{exploitation}} \\ - \alpha^i \pi^i_{a^i}(t) & \underbrace{\left(\ln \pi^i_{a^i}(t) - \sum_{b^i \in \mathcal{A}^i} \pi^i_{b^i}(t) \ln \pi^i_{b^i}(t) \right)}_{\text{exploration}}. \end{aligned} \tag{3.7}$$

But in the derivation, it was implicitly assumed[1] that *all* Q-values are updated at each time step, effectively treating the update rule (3.2) as if the Kronecker delta $\delta_{A^i(t),a^i}$ were absent. This

[1] This non-trivial assumption is not explicitly stated.

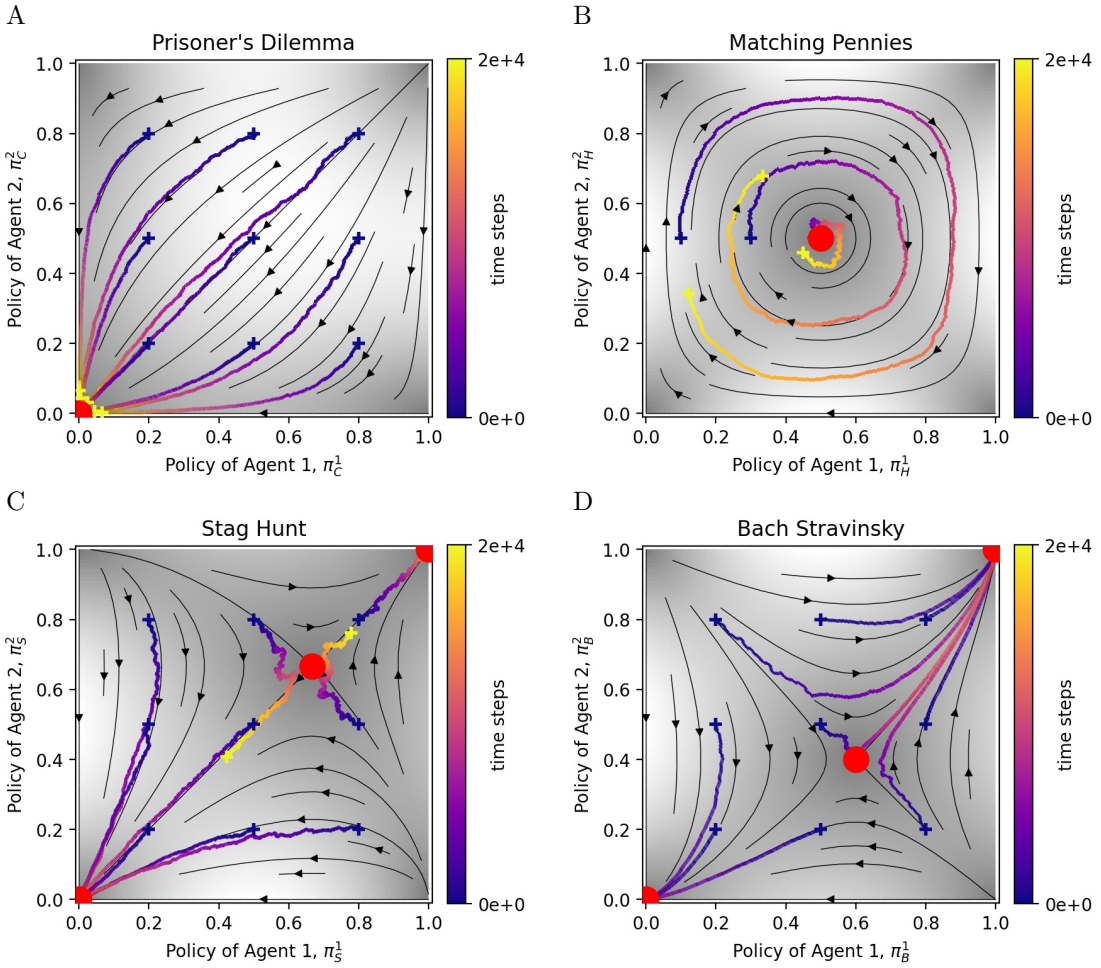


Figure 3.1: Policy trajectories of a single run of independent Cross learning, defined by (3.5), for $\alpha^i = 0.001$, overlaid on the replicator dynamics from figure 2.3 (in grey shade) for the games defined in table 2.1. The crosses indicate the start and end position, the colour gradient from purple to yellow indicates the time evolution. The trajectories follow the streamlines of the replicator dynamics, albeit with stochastic noise. The red dots show the Nash Equilibria of the games. (A) Prisoner's Dilemma. (B) Matching Pennies. (C) Stag Hunt. (D) Bach Stravinsky.

assumption allows to simplify the dynamics to the lower-dimensional policy space and eliminates all terms involving the discount factor γ^i. While this may aid theoretical analysis, it introduces significant discrepancies between the model and actual dynamics [80].

However, the model aligns well with a modified variant of Q-learning, termed '*frequency-adjusted* Q-learning' (FAQL). Originally applied by Leslie and Collins in 2005, Kaisers and Tuyls defined and termed it a separate algorithm in 2010, after identifying the discrepancies to be caused by the update frequencies. But rather than revising the approximation model to reflect actual learning dynamics, they adjusted the algorithm itself to match the simplified model dynamics, arguing that this adaptation yields more favourable and stable outcomes. For clarity, we therefore refer to the simplified model (3.7) as the 'FAQL model' throughout the remainder of this work.

The FAQL algorithm [80] smooths the learning process by scaling the learning rate with the inverse of the update frequency, $1/\pi^i_{a^i}(t)$, which effectively diminishes the influence of the Kronecker delta in the derivation of (3.7). Additionally, it introduces a new hyperparameter, $\beta^i \in [0, 1)$, which modifies the update rule[2] to

$$Q^i_{a^i}(t+1) = Q^i_{a^i}(t) + \alpha^i \min\left(\frac{\beta^i}{\pi^i_{a^i}(t)}, 1\right) \delta_{A^i(t)a^i} \left[R^i_{\mathbf{A}(t)} + \gamma^i \max_{b^i \in \mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t)\right]. \tag{3.8}$$

Relating the Exploration-Exploitation Mechanism to the Principle of Free Energy Besides motivating a novel algorithm, the work of [48] made another valuable contribution. A crucial finding is that the derived equations can be decomposed into terms for exploitation (selection following the replicator dynamics) and exploration (mutation through randomisation based on the Boltzmann mechanism). Equation (3.7) can also be interpreted through the lens of thermodynamics, drawing parallels between selection and energy, and mutation and entropy. The mutation term can be further decomposed into two components: the entropy of a single policy, $\ln \pi^i_{a^i}(t)$, and the entropy of the entire population, $\sum_{b^i} \pi^i_{b^i}(t) \ln \pi^i_{b^i}(t)$. This perspective suggests that mutation is driven by the difference in entropy between an individual policy and the overall population entropy [42, 48]. A similar relation of the dynamics of Temporal Difference Learning to the thermodynamical concept of free energy has also been suggested by [70, 71] and [22]. We further note that a free energy minimisation principle has been proposed as a framework for modelling perception and learning in neurobiological theories [93]. This "free energy principle" posits that biological systems minimise a measure, termed "free energy" in analogy to thermodynamics, representing the difference between predicted and actual sensory inputs, to maintain stability and guide perception, action, and learning.

3.2.3 Batch Q-Learning (BQL) Model

In 2019, Barfuss et al. extended previous deterministic models of MARL, such as the FAQL model, which were focused so far on single-state environments, to encompass *multi-state* environments with time discounting [74]. While the title of the publication might suggest the model represents classic, incremental Temporal Difference (TD) learning—of which Q-learning is a specific case—it actually approximates a *batch* version of TD-learning rather than incremental TD-learning. In batch learning [49], the timescales of interaction and adaptation are separated. This approach allows agents to adapt based on aggregated experiences rather than individual interactions. Here we follow the definition of [74], restrict it to the single-state setup as defined in section 3.1 and discuss its deterministic approximation.

In batch independent Q-learning, the agents interact $K \in \mathbb{N}_+$ times under the *constant* joint policy $\boldsymbol{\pi}(t)$. The information from these interactions are stored inside a batch of size K. At the update step $(t+K)$, the agents then use the sample average of the gathered experience to update their Q-values and subsequently the joint policy $\boldsymbol{\pi}(t+K)$. With a minor abuse of notation to

[2] The minimum ensures that the effective learning rate does not exceed one.

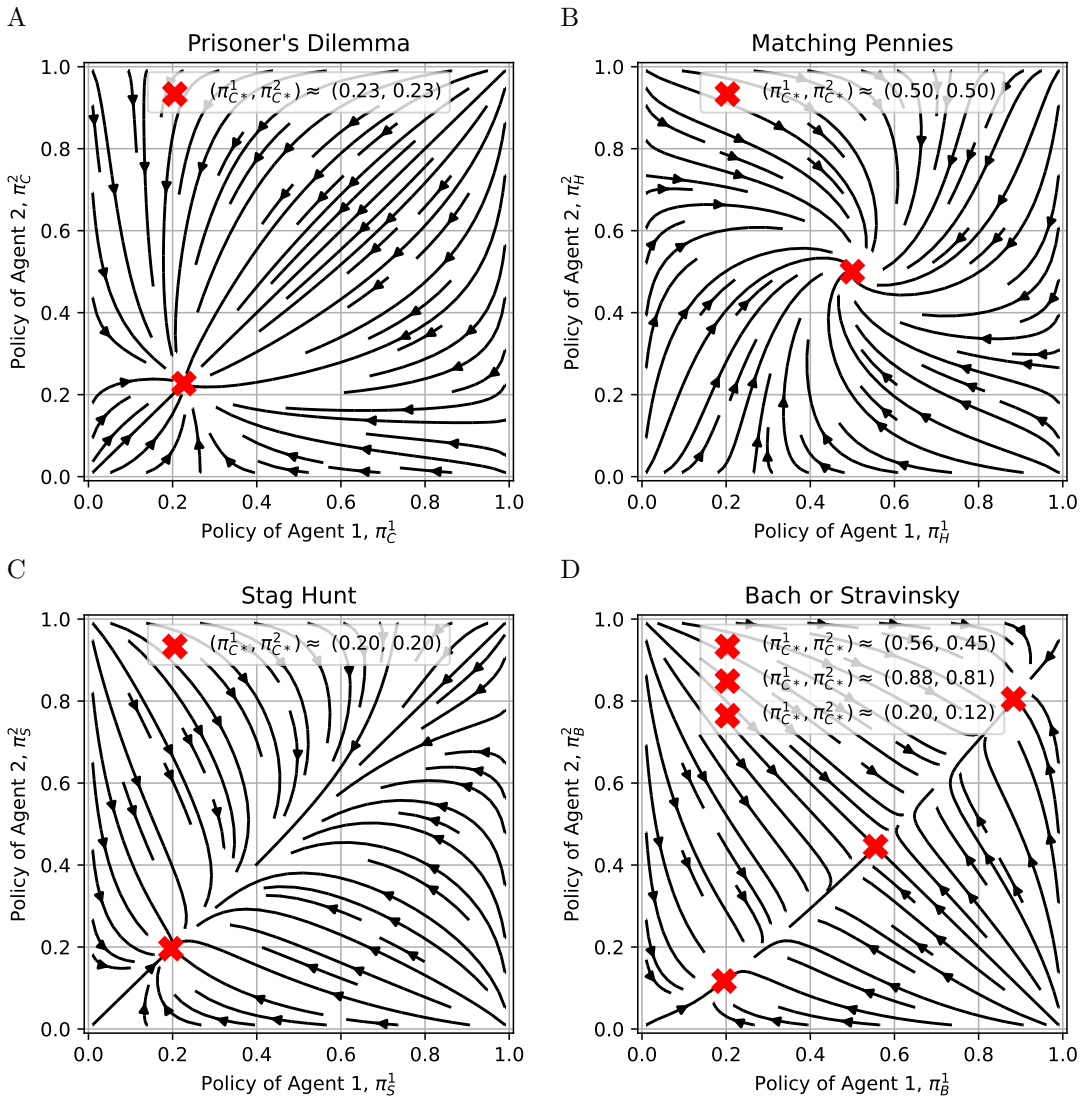


Figure 3.2: Policy space of the FAQL model, defined by (3.7), for the games defined in table 2.1 for $T^i = 1, \alpha^i = 0.01$. The red crosses indicate the fixed points of the dynamics. A comparison with the replicator dynamics of the CL model (figure 3.1) shows, that the exploration term in (3.7) changes the position of the equilibria from pure to probabilistic policies. Note also that while Matching Pennies exhibit stable limit cycles around the Nash equilibrium in the replicator dynamics, the equilibrium is a stable focus in the FAQL model dynamics. The dynamics in the Stag Hunt game depict only one fixed point, compared to the three in the replicator dynamics.

improve readability, (3.2) is modified to

$$Q^i_{a^i}(t+K) = Q^i_{a^i}(t) + \alpha^i D^i_{a^i,\mathbf{A}(t),\dots,\mathbf{A}(t+K),Q^i(t)}, \tag{3.9}$$

$$D^i_{a^i,\mathbf{A}(t),\dots,\mathbf{A}(t+K),Q^i(t)} := \frac{1}{K_{a^i}} \sum_{k=0}^{K-1} \delta_{A^i(t+k)a^i} \left[R^i_{\mathbf{A}(t+k)} + \gamma^i \max_{b^i \in \mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t) \right], \tag{3.10}$$

where $K_{a^i} := \max\left(1, \sum_{k=0}^{K-1} \delta_{A^i(t+k)a^i}\right)$ denotes the number of times agent i played action a^i. To avoid division of zero, $K_{a^i} := 1$ if the action a^i was never played. For a batch size of $K = 1$, batch Q-learning is equal to regular Q-learning. Note however that for $K > 1$, batch learning allows to update *multiple* Q-values per agent per update step—all Q-values whose actions were played in the batch.

In the infinite batch limit $K \to \infty$ (and subsequently $K_{a^i} \to \infty$), the stochastic batch temporal difference error (3.10) becomes almost surely (a.s.) deterministic because of the law of large numbers. It can be written in dependence of all Q-values at time t as

$$\begin{aligned}
D^i_{a^i,\mathbf{Q}(t)} &:= \lim_{K\to\infty} D^i_{a^i,\mathbf{A}(t),\dots,\mathbf{A}(t+K),Q^i(t)} \\
&\overset{\text{a.s.}}{=} \mathbb{E}_{A^i(t)=a^i, A^{-i}(t)\sim\pi^{-i}(t)} \left(\delta_{A^i(t)a^i} \left[R^i_{\mathbf{A}(t)} + \gamma^i \max_{b^i\in\mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t) \right] \right) \\
&= \mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{a^i A^{-i}(t)} + \gamma^i \max_{b^i\in\mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t) \\
&= \mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)} R^i_{a^i A^{-i}(t)} + \underbrace{\gamma^i \max_{b^i\in\mathcal{A}^i} Q^i_{b^i}(t)}_{\text{constant in } a^i} \\
&\quad - T^i \ln \pi^i_{a^i}(t) - \underbrace{T^i \ln \sum_{b^i\in\mathcal{A}^i} \exp[Q^i_{b^i}(t)/T^i]}_{\text{constant in } a^i},
\end{aligned} \tag{3.11}$$

where the last two terms are the inverse of (3.3). The deterministic update rule for the Q-values in the separated *update* timescale u then reads

$$Q^i_{a^i}(u+1) = Q^i_{a^i}(u) + \alpha^i D^i_{a^i,\mathbf{Q}(u)}. \tag{3.12}$$

Inserting (3.12) into (3.3) returns a deterministic update rule for the policy,

$$\begin{aligned}
\pi^i_{a^i}(u+1) &= \frac{\exp[Q^i_{a^i}(u+1)/T^i]}{\sum_{b^i\in\mathcal{A}^i} \exp[Q^i_{b^i}(u+1)/T^i]} \\
&= \frac{\exp[Q^i_{a^i}(u)/T^i] \exp[\alpha^i D^i_{a^i,\mathbf{Q}(u)}/T^i]}{\sum_{b^i\in\mathcal{A}^i} \exp[Q^i_{b^i}(u)/T^i] \exp[\alpha^i D^i_{b^i,\mathbf{Q}(u)}/T^i]} \\
&= \frac{\pi^i_{a^i}(u) \exp[\alpha^i D^i_{a^i,\mathbf{Q}(u)}/T^i]}{\sum_{b^i\in\mathcal{A}^i} \pi^i_{b^i}(u) \exp[\alpha^i D^i_{b^i,\mathbf{Q}(u)}/T^i]}.
\end{aligned} \tag{3.13}$$

As it is, (3.13) depends on the higher-dimensional vector $\mathbf{Q}(u)$. To have an approximation that conveniently reduces the learning dynamics to the lower-dimensional policy space, (3.13) needs to be expressed purely in terms of $\boldsymbol{\pi}(u)$. Luckily, one can make use of the fact that (3.13) is invariant under adding terms to $D^i_{a^i,\mathbf{Q}(u)}$ that are constant in a^i, such as the last term of (3.11). Equation (3.13) thus simplifies to

$$\pi^i_{a^i}(u+1) = \frac{\pi^i_{a^i}(u) \exp[\alpha^i D^i_{a^i,\boldsymbol{\pi}(u)}/T^i]}{\sum_{b^i\in\mathcal{A}^i} \pi^i_{b^i}(u) \exp[\alpha^i D^i_{b^i,\boldsymbol{\pi}(u)}/T^i]}, \tag{3.14}$$

where

$$D^i_{a^i,\boldsymbol{\pi}(u)} := \mathbb{E}_{A^{-i}(u)\sim\pi^{-i}(u)} R^i_{a^i A^{-i}(u)} - T^i \ln \pi^i_{a^i}(u). \tag{3.15}$$

Note that for single-state environments, the second term including the discount factor γ^i also vanishes in the derivation of (3.14), as it is constant in action. No matter which actions the

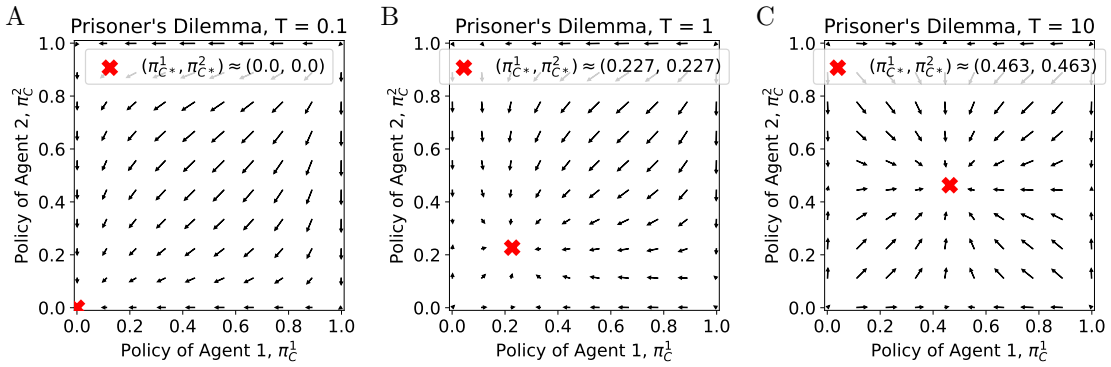


Figure 3.3: Policy space of the BQL model, defined by (3.14), for the Prisoner's Dilemma for $\alpha^i = 0.01$ and three different temperature values. Red crosses indicate the stable fixed point, ranging from $(0, 0)$ for $T \to 0$ to $(0.5, 0.5)$ for $T \to \infty$.

agents choose, the environment transitions back to the same unique non-terminal state. This means that in the limit $K \to \infty$, the dynamics are *independent* of the discount factor. As in the FAQL model, this is again due to the implicit assumption that all Q-values get updated simultaneously.

Barfuss demonstrated good agreement of (3.14) with actual behaviour for $K \approx 10^3 - 10^4$, but not for smaller K-values [22]. To emphasise its distinction from incremental Q-learning, we will refer to this model throughout this work as the 'Batch Q-Learning' (BQL) model. In single-state environments, the FAQL model corresponds to the continuous-time limit of the BQL model—hence, we also collectively refer to them as the 'FAQL/BQL model'.

A fixed point policy $\boldsymbol{\pi}_*$ of (3.14) can be determined by finding the roots of (3.15) for all i, a^i. After normalisation, this results in the two-dimensional system of equations

$$\pi^i_{a^i *} = \frac{\exp[\mathbb{E}_{A^{-i} \sim \pi_*^{-i}} R^i_{a^i A^{-i}} / T]}{\sum_{b^i \in \mathcal{A}^i} \exp[\mathbb{E}_{A^{-i} \sim \pi_*^{-i}} R^i_{b^i A^{-i}} / T]}. \tag{3.16}$$

Figure 3.3 displays the dynamics of the BQL model on the Prisoner's Dilemma for three temperature values. A linear stability analysis (box 3.2.3) shows that the system has a unique stable fixed point, ranging from $\lim_{T\to\infty} \pi^i_{C*} = 0.5$ to $\lim_{T\to 0} \pi^i_{C*} = 0$, which is the Nash equilibrium.

The fixed point equation (3.16) can also be interpreted outside the learning context as defining a "soft" version of Nash equilibrium based on a form of bounded rationality rather than full rationality: if the equation is fulfilled, both players do not maximise but "soft maximise" their reward under the correct assumption that the other player does likewise, by playing the corresponding Boltzmann policy. In behavioural game theory, this form of equilibrium is called 'Logit Quantal Response equilibrium' [94]. As experimental evidence from humans suggest that indeed boundedly rational human decisions sometimes approximate such soft equilibria [95], the question of whether MARL algorithms converge to such points as well is an important plausibility check.

In summary, the dynamics of both previous approximation models, FAQL and BQL, exhibit the following key characteristics:

1. A fixed point of the dynamics is a boundedly rational strategic equilibrium.
2. They are fully described within the lower-dimensional policy space.
3. For single-state environments, they are independent of the discount factor γ^i—all terms including γ^i vanish in the derivation.

Box 3.2.3: Linear Stability Analysis of the BQL model

A linear stability analysis of the fixed point equations (3.16) involves analysing the eigenvalues of the Jacobian matrix at this point. To this end, we solve the two-dimensional system of equations (3.16) numerically using the `fsolve` function from Python's SciPy library, to calculate the fixed point. To determine its stability, we calculate the Jacobian,

$$J = \begin{pmatrix} \partial_{\pi_C^1}\pi_C^1 & \partial_{\pi_C^2}\pi_C^1 \\ \partial_{\pi_C^1}\pi_C^2 & \partial_{\pi_C^2}\pi_C^2 \end{pmatrix} = \begin{pmatrix} 0 & -\frac{p_{\pi^2}^1 q_{\pi^2}^1}{T[p_{\pi^2}^1+q_{\pi^2}^1]^2} \\ -\frac{p_{\pi^1}^2 q_{\pi^1}^2}{T[p_{\pi^1}^2+q_{\pi^1}^2]^2} & 0 \end{pmatrix}, \tag{3.17}$$

where

$$p_{\pi^{-i}}^i := \exp[\mathbb{E}_{A^{-i}\sim\pi^{-i}} R_{a^i=C,A^{-i}}^i/T],$$
$$q_{\pi^{-i}}^i := \exp[\mathbb{E}_{A^{-i}\sim\pi^{-i}} R_{a^i=D,A^{-i}}^i/T].$$

Note that the prefactor -1 in (3.17) comes from

$$R_{a^i=C,a^{-i}=C}^i - R_{a^i=C,a^{-i}=D}^i - R_{a^i=D,a^{-i}=C}^i + R_{a^i=D,a^{-i}=D}^i = 3-0-5+1=-1$$

We calculate the Eigenvalues λ_n of the Jacobi matrix numerically with the function `numpy.linalg.eig` from Python's NumPy library. Since all eigenvalues are $|\lambda_n| < 1$ for all temperatures (see figure 3.5.G), we deduct the discrete-time fixed point to be a stable node.

3.2.4 Comparison between the FAQL/BQL Model and Independent Q-learning

In this section, we compare the FAQL/BQL model to stochastic realisations of independent Q-learning. We aim to answer whether the approximated dynamics still capture the core principles of Q-learning in a multi-agent setting, or if the inherent assumptions cause the model to deviate significantly from the classic incremental algorithm.

For convenience, we consider α^i, γ^i, T^i to be homogeneous in all experiments throughout the remainder of this thesis. We set the learning rate to $\alpha = 0.01$ and the temperature to $T = 1$, as we are mainly interested on the effect of the discount factor γ.

To compare the algorithm, where learning occurs in Q-space, with the FAQL/BQL model, which describes learning in policy space, it is essential to understand how Q-values translate into policies via the Boltzmann mechanism and vice versa. Note that an agent's probability to cooperate, π_C^i, does *not* depend on the absolute Q^i-values but only on their difference, $\Delta Q^i := Q_D^i - Q_C^i$, due to

$$\pi_C^i = \frac{e^{Q_C^i/T}}{e^{Q_C^i/T} + e^{(Q_C^i+\Delta Q^i)/T}} = \frac{1}{1+e^{\Delta Q^i/T}}. \tag{3.18}$$

Thus, a joint policy does *not* correspond to a single point in Q-space but an affine subspace. To study the influence of any initial policy $\boldsymbol{\pi}(0)$ on the algorithmic learning process, we first need to specify initial $\mathbf{Q}(0)$-values which fulfil $\boldsymbol{\pi}(0) = f(\mathbf{Q}(0))$. To this end, we define for any given $\pi_C^i(0)$:

$$\begin{aligned} Q_C^i(0) &:= Q_{base} - \frac{\Delta Q^i(\pi_C^i(0))}{2}, \\ Q_D^i(0) &:= Q_{base} + \frac{\Delta Q^i(\pi_C^i(0))}{2}, \end{aligned} \tag{3.19}$$

where Q_{base} is a parameter that governs the overall initial level of Q-values.

Figure 3.4 illustrates the time evolution of single runs of independent Q-learning for $Q_{base} = 0$, two values of the discount factor and two different initial joint policies.

For $\gamma = 0$ (figure 3.4.A and 3.4.C), the FAQL/BQL model, depicted in dotted lines, shows good agreement with the stochastic process (besides the slower timing in figure 3.4.C).

However, for $\gamma = 0.8$ (figure 3.4.B and 3.4.D), the model does not capture the behaviour at all. For both initial conditions, after the first few hundred time steps, the policy trajectories settle into metastable phases where they remain for an extended period. After a very long time, the behaviour undergoes a drastic shift, and the policies transition into a sustained oscillatory pattern that persists indefinitely. In figure 3.4.B, this transition occurs after approximately 70 thousand steps. For $(\pi_C^1(0), \pi_C^2(0)) = (0.9, 0.7)$, the shift is even more pronounced. Initially, the policies seem to converge on mutual cooperation, which appears to contradict individually rational behaviour. However, after about *two million* steps, the trajectories descend into the same asymmetric metastable phase observed for $(\pi_C^1(0), \pi_C^2(0)) = (0.5, 0.48)$, before ultimately transitioning into the indefinite oscillations. In stark contrast, the FAQL/BQL model predicts fundamentally simpler behaviour: as shown by the dotted lines, their policy trajectories quickly converge to a joint policy within just a few hundred steps. That the previous models do not describe actual Q-learning can also be seen in figure 3.5.

Figure 3.5.I shows averaged policy trajectories of Q-learning over five runs for two different initialisation approaches and two different values of γ. For $Q_{base} = \min(\mathbf{R})/(1-\gamma) = 0$, the trajectories deviate from the model, following the edges of the policy space instead. For $Q_{base} = \max(\mathbf{R})/(1-\gamma)$, the trajectories initially cluster near the center of the policy space. For $\gamma = 0$, although the trajectories differ from the FAQL/BQL model, they eventually equilibrate around the fixed point, regardless of initialisation. However, for $\gamma = 0.8$ the trajectories fall into indefinite oscillations, which are not centred around the fixed point. In figure 3.5.B, some trajectories appear to converge to mutual cooperation in the depicted time span of 1×10^5 steps. But as mentioned above, these phases are only metastable. Given sufficient time, the trajectories eventually transition to the same oscillatory pattern observed in other trajectories. Notably, these metastable phases do *not* occur for trajectories initialised at $Q_{base} = 25$.

In summary, the stylised discrepancies are:

1. Whereas the FAQL/BQL model dynamics converge to a single Logit Quantal Response equilibrium in the Prisoner's Dilemma after a couple of hundred steps, actual independent Q-learning does *not* necessarily converge to any strategic equilibrium and may instead settle into oscillations that emerge only after millions of steps.

2. Whereas the FAQL/BQL model reside in the lower-dimensional policy space, actual independent Q-learning dynamics *cannot* be reduced from the higher-dimensional Q-space: the initialisation (Q_{base}) matters.

3. Whereas the FAQL/BQL model is independent of the discount factor in single-state environments, actual independent Q-learning dynamics are clearly influenced by changes in γ and exhibit fundamentally different behaviour.

For a comparison of the FAQL/BQL model with *frequency-adjusted* independent Q-learning, see appendix A.1.

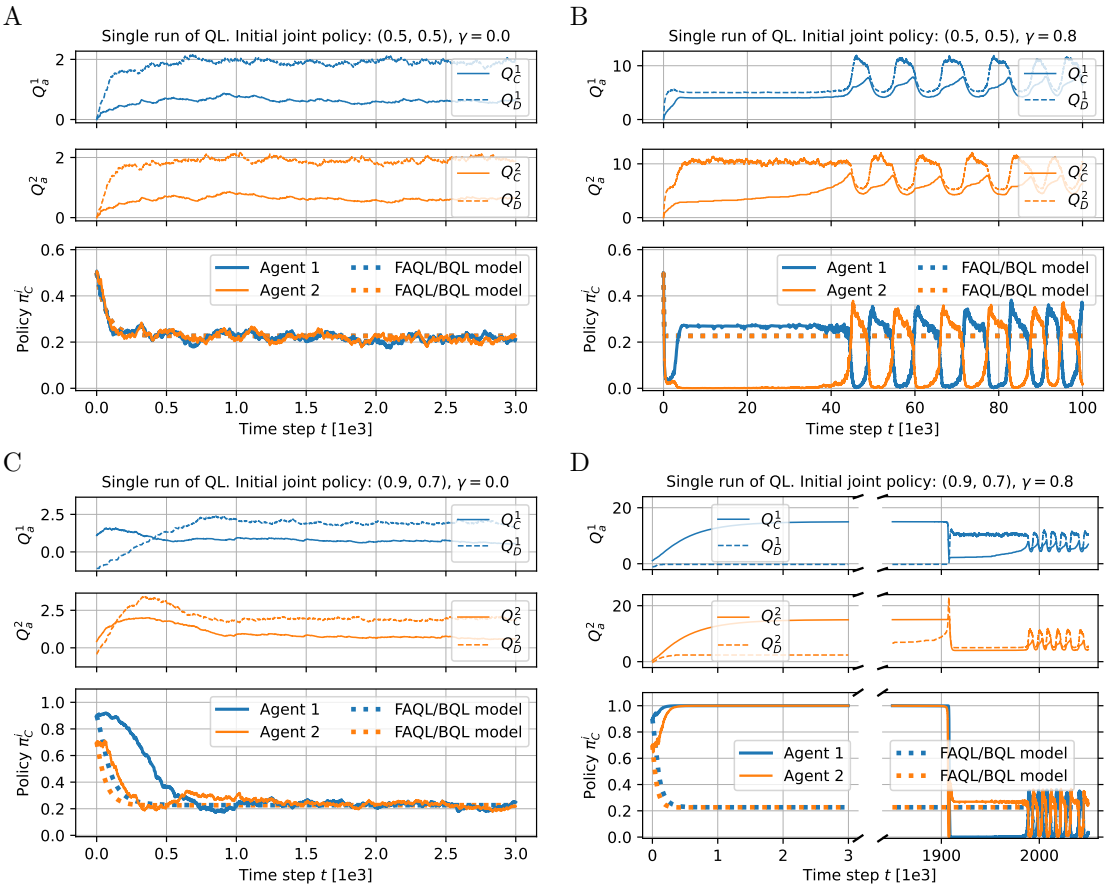


Figure 3.4: Time evolution of single runs of independent Q-learning for different initial conditions and discount factor values and $T = 1, \alpha = 0.01$, compared with the policy trajectories of the FAQL/BQL model. Note that the depicted runs represent single instances of a stochastic process. Timings and trajectories vary across different runs. The first two subplots in each panel show the evolution of the Q-values $(Q_C^1, Q_D^1, Q_C^2, Q_D^2)$, while the third subplot illustrates the resulting probabilities of cooperation (π_C^1, π_C^2). The dotted policy trajectories represent previous approximation methods: FAQL, defined by (3.7), and BQL, defined by (3.14). The left panels (A, C) depict $\gamma = 0$, the right panels (B, D) $\gamma = 0.8$. The top panels (A, B) depict an initial joint policy $(\pi_C^1, \pi_C^2) = (0.5, 0.48)$, corresponding to Q-values $(0, 0, -0.04, 0.04)$ via (3.19). The lower panels (B, D) show an initial joint policy $(\pi_C^1, \pi_C^2) = (0.9, 0.7)$, corresponding to Q-values $(1.1, -1.1, 0.4, -0.4)$ via (3.19).

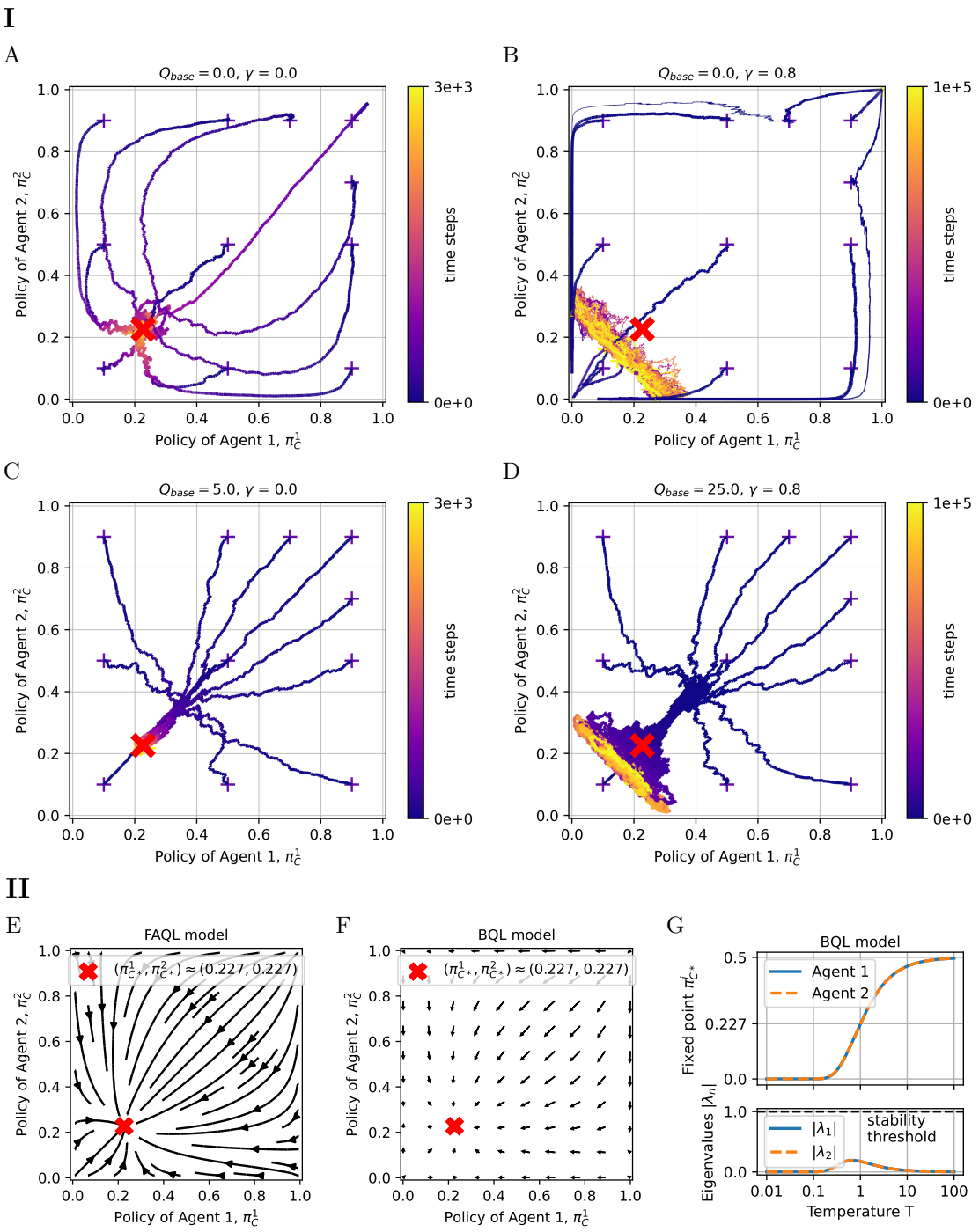


Figure 3.5: Comparison between averaged policy trajectories of independent Q-learning on the Prisoner's Dilemma (**I**) and previous deterministic models (**II**). **I:** Top panels (A, B): $Q_{base} = \min(\mathbf{R})/(1-\gamma)$. Bottom panels (C, D): $Q_{base} = \max(\mathbf{R})/(1-\gamma)$. Left panels (A, C): $\gamma = 0$. Right panels (B, D): $\gamma = 0.8$. For each initialisation, five runs are executed. The trajectories from the same initialisation are grouped based on their final location in policy space (below or above the diagonal from (0,1) to (1,0)), and the mean of each group is plotted. Line thickness indicates the proportion of runs in each group. The colour gradient (purple to yellow) indicates time evolution. The red cross marks the fixed point of the FAQL/BQL model. Note that for $Q_{base} = 0$ and $\gamma = 0.8$, some trajectories initialised in the top right appear to converge to the metastable phase of mutual cooperation in the depicted time span of 1×10^5 steps. **II:** Vector fields of previous models for $T = 1$ and $\alpha = 0.01$. E: FAQL model in continuous time, defined by (3.7). F: BQL model in discrete time, defined by (3.14). G: Stability analysis of the BQL model. It has a unique symmetric fixed point $\boldsymbol{\pi}_* > 0$, depending on the temperature $T > 0$. All absolute eigenvalues of the Jacobian at π_{C*}^i are below 1, indicating a stable node.

3.3 A Choice-Probability-Aware Model of Independent Q-learning

The discrepancies between the FAQL/BQL model and independent Q-learning arise from the implicit assumption in the former that *all* Q-values are updated at each step. Some researchers recognised the need to consider update frequencies but modified the algorithm to fit the model, rather than adjusting the model itself [63, 74, 80]. While some publications [16, 22, 42, 63, 80] acknowledged that the FAQL model [48] does not accurately represent actual independent Q-learning, others—including [70, 71], and more recent works like [79] and [89]—do not mention this discrepancy, potentially overlooking its implications. Recently in 2022, Hu et al. proposed an adjusted "continuity equation model" of independent Q-learning in large-scale multi-agent systems modelled as population games [76]. However, their model is limited to the case $\gamma = 0$. Thus, we cannot apply it to explain all of the stylised discrepancies from above.

Here, we propose an approximation model for independent Q-learning in a single-state, repeated environment, with discounting but no memory, as defined in section 3.1. We show that all stylised discrepancies between actual independent Q-learning and previous approximation models can be explained by adjusting the previous models' update frequencies to be proportional to the current agent's policies. Our primary focus is then to demonstrate and rigorously prove that heterogeneous update frequencies can fundamentally alter a system's behaviour, emphasising the need for caution when using MARL as a modelling tool.

We construct our deterministic approximation by isolating the dynamics between consecutive time steps. At each step, we study the expectation of the next step given the current values. The model stays in discrete-time, aligning closer with the inherent nature of computer simulations. Importantly, this approach replaces the Kronecker delta $\delta_{A^i(t)a^i}$ in (3.2) with the probability—or update frequency—$\pi^i_{a^i}(t)$, leading to

$$\begin{aligned}\mathbb{E}_{\mathbf{A}(t)\sim\boldsymbol{\pi}(t)}[Q^i_{a^i}(t+1) \mid Q^i(t)] = Q^i_{a^i}(t) \\ + \alpha\pi^i_{a^i}(t)\left[\mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)}R^i_{a^iA^{-i}(t)} + \gamma \max_{b^i\in\mathcal{A}^i} Q^i_{b^i}(t) - Q^i_{a^i}(t)\right].\end{aligned} \tag{3.20}$$

It is *not possible* to reduce these dynamics into the lower-dimensional policy space, as done in the FAQL/BQL model. When attempting to transform (3.20) into ΔQ-space for the Prisoner's Dilemma,

$$\begin{aligned}\mathbb{E}_{\mathbf{A}(t)\sim\boldsymbol{\pi}^{-i}(t)}\Delta Q^i(t+1) &= \mathbb{E}_{\mathbf{A}(t)\sim\boldsymbol{\pi}^{-i}(t)}Q^i_D(t+1) - \mathbb{E}_{\mathbf{A}(t)\sim\boldsymbol{\pi}^{-i}(t)}Q^i_C(t+1) \\ &= \Delta Q^i(t) + \alpha\Bigg[\mathbb{E}_{A^{-i}(t)\sim\pi^{-i}(t)}\left(\pi^i_D(t)R^i_{a^i=D,A^{-i}(t)} - \pi^i_C(t)R^i_{a^i=C,A^{-i}(t)}\right) \\ &\quad + \left(\pi^i_D(t) - \pi^i_C(t)\right)\gamma\max_{b^i\in\mathcal{A}^i} Q^i_{b^i}(t) - \pi^i_D(t)Q^i_D(t) + \pi^i_C(t)Q^i_C(t)\Bigg],\end{aligned}$$

it becomes apparent, that the last three terms cannot be expressed in terms of ΔQ^i because of the update frequencies $\pi^i_{a^i}(t) \neq 1$.

3.3.1 Results

We analyse the dynamics of (3.20) on the Prisoner's Dilemma. Although the 4D dynamics cannot be reduced to the 2D policy space, a *projection* can still be illustrated (figure 3.6). For reasonably small learning rates ($\alpha = 0.01$), a comparison with the averaged trajectories of Q-learning (figure 3.5) demonstrates that (3.20) captures the observed complexities.

For $\gamma = 0$, all trajectories converge to the fixed point of the FAQL/BQL model, $\pi^i_{C*} \approx 0.227$. In contrast, for $\gamma = 0.8$, the behaviour depends on the initial policies: symmetric initial policies converge to π^i_{C*} while asymmetric initial policies lead to oscillatory dynamics. Note that in figure 3.5.B, the trajectory starting at the symmetric initial condition $\pi^i_C = 0.9$ remains at mutual cooperation for up to two million steps, seemingly contradicting the statement just made. However, after an astonishing four *billion* steps, it finally converges to π^i_{C*}. These phenomena are readily explained through a stability analysis of our model, offering an efficient approach while avoiding the ambiguities of interpreting single trajectories of specific parameter conditions.

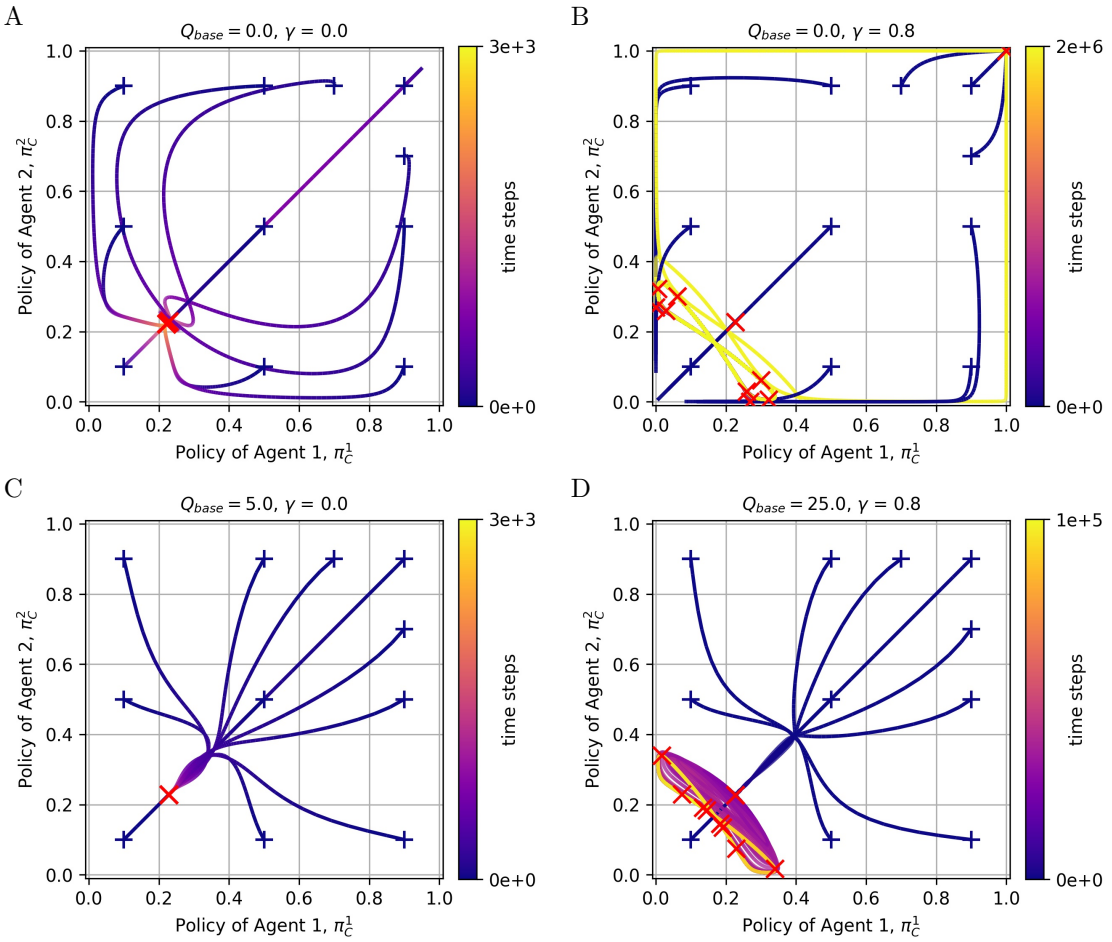


Figure 3.6: Projection of our 4D deterministic approximation model of independent Q-learning on the Prisoner's Dilemma, defined by (3.20), into 2D policy space for $T = 1$, $\alpha = 0.01$, and different values of γ and Q_{base}. The colour gradient (purple to yellow) represents time evolution. The end point of each trajectory is indicated by a red cross. Top panels (A, B): $Q_{base} = \min(\mathbf{R})/(1-\gamma)$. Bottom panels (C, D): $Q_{base} = \max(\mathbf{R})/(1-\gamma)$. Left panels (A, C): $\gamma = 0$. Right panels (B, D): $\gamma = 0.8$. Note that in panel B, the trajectory initialised at $\pi_C^i(0) = 0.9$ eventually converges to the fixed point $\pi_{C*}^i \approx 0.227$, but only after 4×10^7 steps, far beyond the depicted 2×10^6 steps.

Box 3.3.1: Linear Stability Analysis of our Model

To analyse the stability of the four dimensional fixed point $\mathbf{Q}_* = (Q^1_{C*}, Q^1_{D*}, Q^2_{C*}, Q^2_{D*})$, given by (3.21), we need to analyse the eigenvalues of the Jacobi matrix at this point. If we take into account that for the Prisoner's Dilemma, $Q^i_{C*} < Q^i_{D*}$ holds at $\mathbf{Q}_*$, the maximum term of (3.20) reduces to $\max(Q^i_{C*}Q^i_{D*}) = Q^i_{D*}$. We can therefore simplify (3.20) at the fixed point $\mathbf{Q}_*$ to

$$\mathbb{E}_{\mathbf{A}(t)\sim\boldsymbol{\pi}(t)}[Q^i_{a^i}(t+1) \mid Q^i_*(t)] = Q^i_{a^i*}(t) + \alpha\pi^i_{a^i*}(t)\left[\mathbb{E}_{A^{-i}(t)\sim\pi^{-i}_*(t)} R^i_{a^iA^{-i}(t)} + \gamma Q^i_{D*} - Q^i_{a^i*}\right].$$

To shorten the notation, we omit the dependencies and the fixed point subscript index $*$ in the following, and make use of the relations

$$\partial_{Q^i_C}\pi^i_C = \partial_{Q^i_D}\pi^i_D = \frac{e^{(Q^i_C+Q^i_D)/T}}{T(e^{Q^i_C/T}+e^{Q^i_D/T})^2},$$
$$\partial_{Q^i_D}\pi^i_C = -\partial_{Q^i_C}\pi^i_C = -\partial_{Q^i_D}\pi^i_D = \partial_{Q^i_C}\pi^i_D,$$

where $\pi^i_{a^i}$ is given by (3.3). To shorten the notation further, we introduce

$$\begin{aligned}
f^i &:= \alpha\partial_{Q^i_C}\pi^i_C\left[\pi^{-i}_C R^i_{a^i=C,a^{-i}=C} + (1-\pi^{-i}_C)R^i_{a^i=C,a^{-i}=D} + \gamma Q^i_D - Q^i_C\right],\\
g^i &:= \alpha\pi^i_C\partial_{Q^{-i}_C}\pi^{-i}_C\left[R^i_{a^i=C,a^{-i}=C} - R^i_{a^i=C,a^{-i}=D}\right],\\
h^i &:= \alpha\partial_{Q^i_C}\pi^i_C\left[\pi^{-i}_C R^i_{a^i=D,a^{-i}=C} + (1-\pi^{-i}_C)R^i_{a^i=D,a^{-i}=D} - (1-\gamma)Q^i_D\right],\\
k^i &:= \alpha(1-\pi^i_C)\partial_{Q^{-i}_C}\pi^{-i}_C\left[R^i_{a^i=D,a^{-i}=C} - R^i_{a^i=D,a^{-i}=D}\right],
\end{aligned}$$

which help to write the Jacobi matrix at the fixed point as

$$\begin{pmatrix}
f^i - \alpha\pi^i_C + 1 & -f^i + \alpha\gamma\pi^i_C & g^i & -g^i\\
-h^i & h^i - \alpha(1-\gamma)(1-\pi^i_C) + 1 & k^i & -k^i\\
g^{-i} & -g^{-i} & f^{-i} - \alpha\pi^{-i}_C + 1 & -f^{-i} + \alpha\gamma\pi^{-i}_C\\
k^{-i} & -k^{-i} & -h^{-i} & h^{-i} - \alpha(1-\gamma)(1-\pi^{-i}_C) + 1
\end{pmatrix}.$$

We solve the eigenvalues of the Jacobi matrix at the fixed point (3.21) numerically with the function `numpy.linalg.eig` from Python's NumPy library. The absolute eigenvalues are plotted against the discount factor in figure 3.7 for three different temperature values.

Stability Analysis The four-dimensional fixed point $\mathbf{Q}_*$ of (3.20) is obtained by finding the roots of the second term for all i, a^i. The coupled equations read

$$\begin{aligned}
Q^i_{a^i*} &:= \mathbb{E}_{A^{-i}\sim\pi^{-i}_*} R^i_{a^iA^{-i}} + \gamma \max_{b^i\in\mathcal{A}^i} Q^i_{b^i*}\\
&= \mathbb{E}_{A^{-i}\sim\pi^{-i}_*} R^i_{a^iA^{-i}} + \gamma \max_{b^i\in\mathcal{A}^i} \sum_{k=0}^{\infty}\gamma^k \mathbb{E}_{A^{-i}\sim\pi^{-i}_*} R^i_{b^iA^{-i}}\\
&= \mathbb{E}_{A^{-i}\sim\pi^{-i}_*} R^i_{a^iA^{-i}} + \underbrace{\frac{\gamma}{1-\gamma}\max_{b^i\in\mathcal{A}^i}\mathbb{E}_{A^{-i}\sim\pi^{-i}_*} R^i_{b^iA^{-i}}}_{\text{constant in } a^i},
\end{aligned} \tag{3.21}$$

where the final line applies the geometric series formula under the condition $|\gamma| < 1$. Note that in the translation of $\mathbf{Q}_*$ to $\boldsymbol{\pi}_*$ via the Boltzmann function (3.3), the second term of (3.21) is irrelevant as it is an offset constant in a^i and only the differences of the Q-values matter. This means that a fixed point of the dynamics described by (3.14) is also a fixed point of (3.20) in *policy space*, and vice versa. So why does the new model behave so differently?

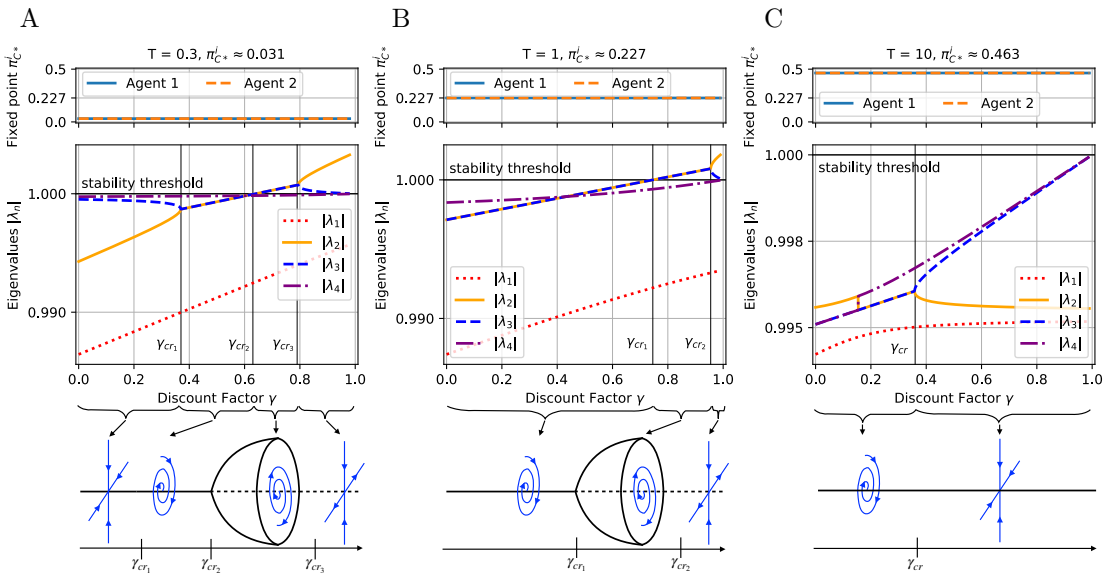


Figure 3.7: Stability analysis of our model, defined by (3.20), for $\alpha = 0.01$ and three different temperature values: $T = 0.3$ (A), $T = 1$ (B), and $T = 10$ (C). The deterministic 4D system shares the same unique symmetric fixed point $\boldsymbol{\pi}_* = \boldsymbol{\pi}(\mathbf{Q}_*)$ in *policy space* as the 2D FAQL/BQL model (figure 3.5). The first row shows the position of the 4D fixed point $\mathbf{Q}_*$, defined by (3.21), in 2D policy space. Specifically, it illustrates how the projected equilibrium policy $\pi^i_{C*} := \pi^i_C(\mathbf{Q}_*)$ is not affected by the discount factor. The second row shows the absolute eigenvalues of the Jacobian matrix at the 4D fixed point $\mathbf{Q}_*$ as a function of γ, with the stability threshold ($|\lambda| = 1$) highlighted. It demonstrates that although the position of the fixed point in policy space remains unaffected by γ, its stability properties changes. For instance, at $T = 1$, the dynamics undergoes a supercritical Neimark-Sacker bifurcation at $\gamma_{cr_1} \approx 0.75$. The third row provides schematic lower-dimensional representations of the corresponding dynamical regimes for different ranges of γ, illustrating transitions between stability, oscillatory dynamics, and divergence.

The key lies in *stability*. Although both models share the same unique fixed point in policy space, their stability properties differ. While it is a 2D stable node for all values of T and all values of γ in the BQL and FAQ model, it is more nuanced in the new 4D model. For $T = 1$, a linear stability analysis (see box 3.3.1 and figure 3.7.B) reveals that the fixed point is a stable focus attractor for $\gamma \lesssim 0.75$, meaning that eventually all trajectories converge to the fixed point. But at $\gamma_{cr_1} \approx 0.75$, the system undergoes a supercritical Neimark-Sacker bifurcation[3]. This turns the stable focus into an unstable focus, around which a stable limit cycle emerges. All trajectories with asymmetric initial conditions in policy space, even with minimal deviation, converge to the limit cycle instead of the fixed point. This describes the oscillations observed for $\gamma = 0.8$ in figure 3.4 and 3.5. For $\gamma \gtrsim 0.95$, the unstable focus turns into a saddle node.

Figure 3.8 depicts these different dynamical regimes in 4D Q-space by plotting projections into 2D policy space and a constructed 3D space, defined by the basis vectors $\mathbf{q}_1 = (1, -1, 0, 0)$, $\mathbf{q}_2 = (0, 0, 1, -1)$, and $\mathbf{q}_3 = (1, 1, -1, -1)$. The first two dimensions are the ΔQ^i-values, the third dimension indicates difference between agents. Note again that the trajectory initialised at $\pi^i_C(0) = 0.9$ for $\gamma = 0.97$ remains at mutual cooperation ($\pi^i_C \approx 1$) within any finite number of steps feasible for computational simulation. However, the equations show that this is *not* a true fixed point.[4]

So far we limited the discussion to $T = 1$. As noted in section 3.2.3 (figure 3.3), the position of the fixed point in policy space changes with varying T. The stability analysis (figure 3.7.A and 3.7.C) further reveals that the effect of the discount factor γ also varies for different T.

[3] A Neimark-Sacker bifurcation is the discrete-time equivalent of an Andronov-Hopf bifurcation.
[4] Technically, $\pi^i_C = 1$ would be a fixed point, but any finite $T > 0$ prohibits pure policies.

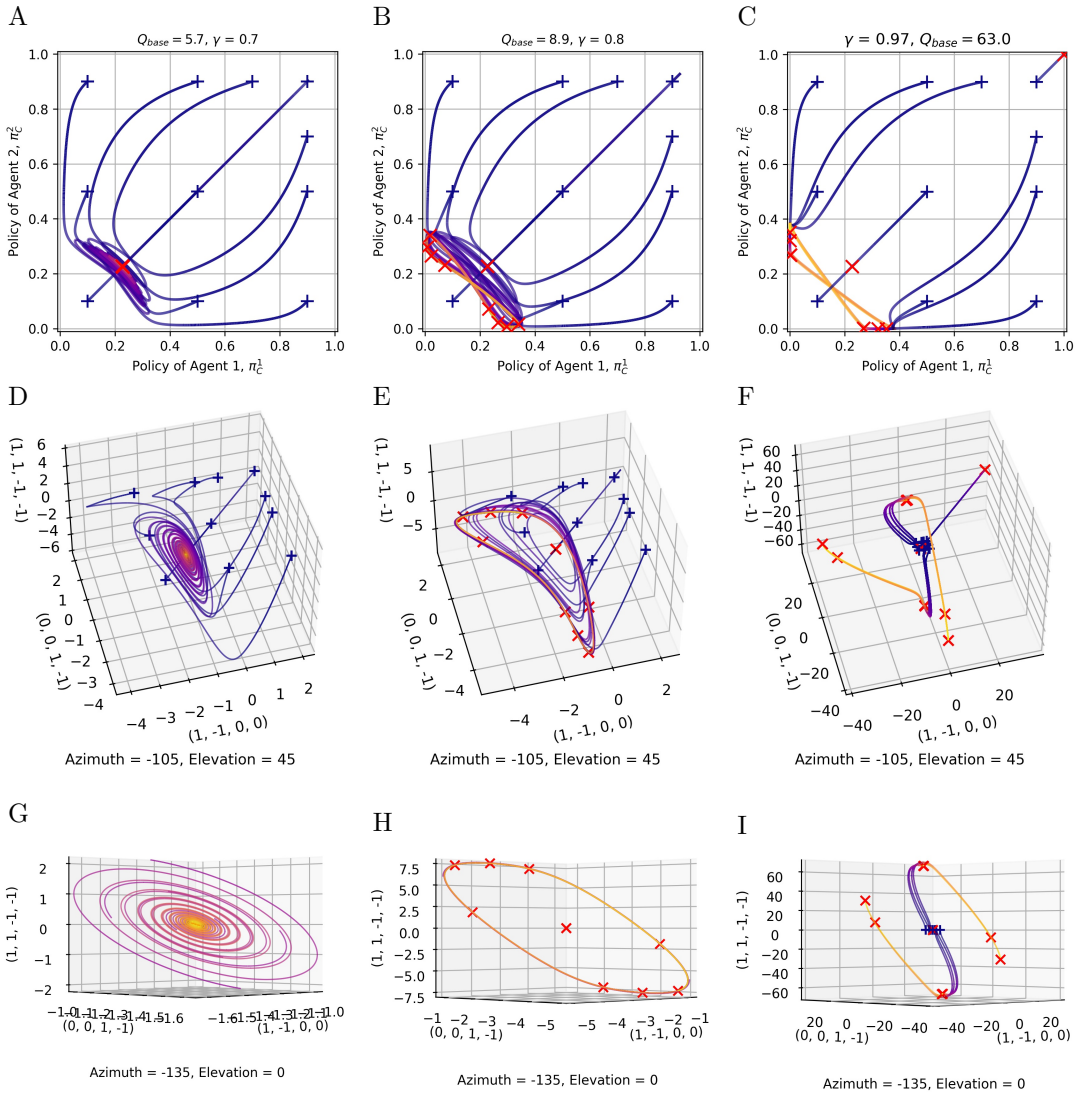


Figure 3.8: Projection of 4D deterministic dynamics of independent Q-learning on the Prisoner's Dilemma, defined by (3.20), for $T = 1$, $\alpha = 0.01$ and different values of γ. Left panels (A, D, G): $\gamma = 0.7$. Middle panels (B, E, F): $\gamma = 0.8$. Right panels (G, H, I): $\gamma = 0.97$. All trajectories are initialised around the fixed point Q-values, defined by (3.21): $Q_{base} = Q_{C*} + \Delta Q_*/2$. The colour gradient (purple to yellow) represents time evolution over 3×10^4 steps. The end point of each trajectory is indicated by a red cross. Top panels (A, B, C): Projection of 4D dynamics into 2D policy space. Middle panels (D, E, F): Projection into a 3D space defined by the basis vectors $\mathbf{q}_1 = (1, -1, 0, 0)$, $\mathbf{q}_2 = (0, 0, 1, -1)$, and $\mathbf{q}_3 = (1, 1, -1, -1)$. The first two dimensions represent the ΔQ^i-values, while the third dimension captures the difference between agents. Bottom panels (G, H, I): Projection into the same 3D space, viewed from a different angle. For $\gamma = 0.7$ and $\gamma = 0.8$, only the last two-thirds of the time evolution are shown for clarity. For $\gamma = 0.7$, the unique fixed point π^i_{C*} is a stable focus. For $\gamma = 0.8$, it is an unstable focus surrounded by a stable limit cycle for all asymmetric joint policies. For $\gamma = 0.97$, it is a saddle point, with stable eigenvectors projected onto the diagonal of the policy space and unstable eigenvectors directed perpendicular to it. The trajectory initialised at $\pi^i_C(0) = 0.9$ remains at mutual cooperation ($\pi^i_C \approx 1$) within any finite number of steps feasible for computational simulation. Note however that the equations show that this is *not* a true fixed point and pure policies are prohibited due to $T > 0$.

3.3.2 Cause of Metastable Phases and Oscillations

With the deterministic equation (3.21) established for calculating an agent's target values based on its opponent's policy, we now examine the underlying causes of the metastable phases and oscillations observed in figure 3.4, which are matched by our model (see figure 3.9). Our discussion focuses on the trajectory starting from the initial policy $(\pi_C^1(0), \pi_C^2(0)) = (0.5, 0.48)$, though similar reasoning applies to all other initial conditions.

Note that our model cannot precisely capture the exact timing of specific runs due to the inherent randomness and sensitivity to initial actions, but it effectively captures the overall timescales of the stochastic system.

Metastable Phases: Starting at zero, all Q-values grow. Since defection yields higher rewards than cooperation, the growth rate of Q_D^i is higher than of Q_C^i. This in turn causes the difference $\Delta Q^i = Q_D^i - Q_C^i$ to increase, resulting in a fast decline of the probability to cooperate. The policy of the second agent declines slightly faster than the first, approaching $\pi_C^2 \approx 0$. At this point, given π_C^2, Agent 1's corresponding target values, calculated using (3.21), are

$$Q_{C,target}^1 = 0 + \frac{\gamma}{1-\gamma} = 4,$$
$$Q_{D,target}^1 = 1 + \frac{\gamma}{1-\gamma} = 5,$$

resulting in $\pi_C^1 \approx 0.27$. In return, given π_C^1, Agent 2's target values are $Q_{C,target}^2 \approx 9.1$ and $Q_{D,target}^2 \approx 10.4$.

Since Agent 2 primarily defects, Q_D^2 updates frequently and reaches its target quickly, while Q_C^2 lags due to infrequent updates, keeping π_C^2 near zero. This metastable phase persists until Q_C^2 receives enough updates to approach its target. Over time, Q_C^2 gradually catches up, closing the gap ΔQ^2, and the assumption $\pi_C^2 \approx 0$ no longer holds.

Oscillations: As π_C^2 grows, the expected rewards and hence also the target values of agent 1 grow. But again, due to the asymmetric update frequency, Q_D^1 increases much faster than Q_C^1. As a result, the policy π_C^1 plummets close to zero. This has the effect that the target values of agent 2 now decrease drastically, closing the ΔQ^2 gap even further. As a result, π_C^2 grows rapidly. Now, the roles of agent 1 and 2 are swapped and the process begins all over again, albeit with a shorter period. An oscillating pattern emerges.

The oscillations can be understood as a feedback loop in which the agents' adaptations consistently lag behind the changes of their effective environment. This phenomenon, known as the moving target problem in RL [1] (section 2.3.3), poses a significant challenge in MARL [12, 16].

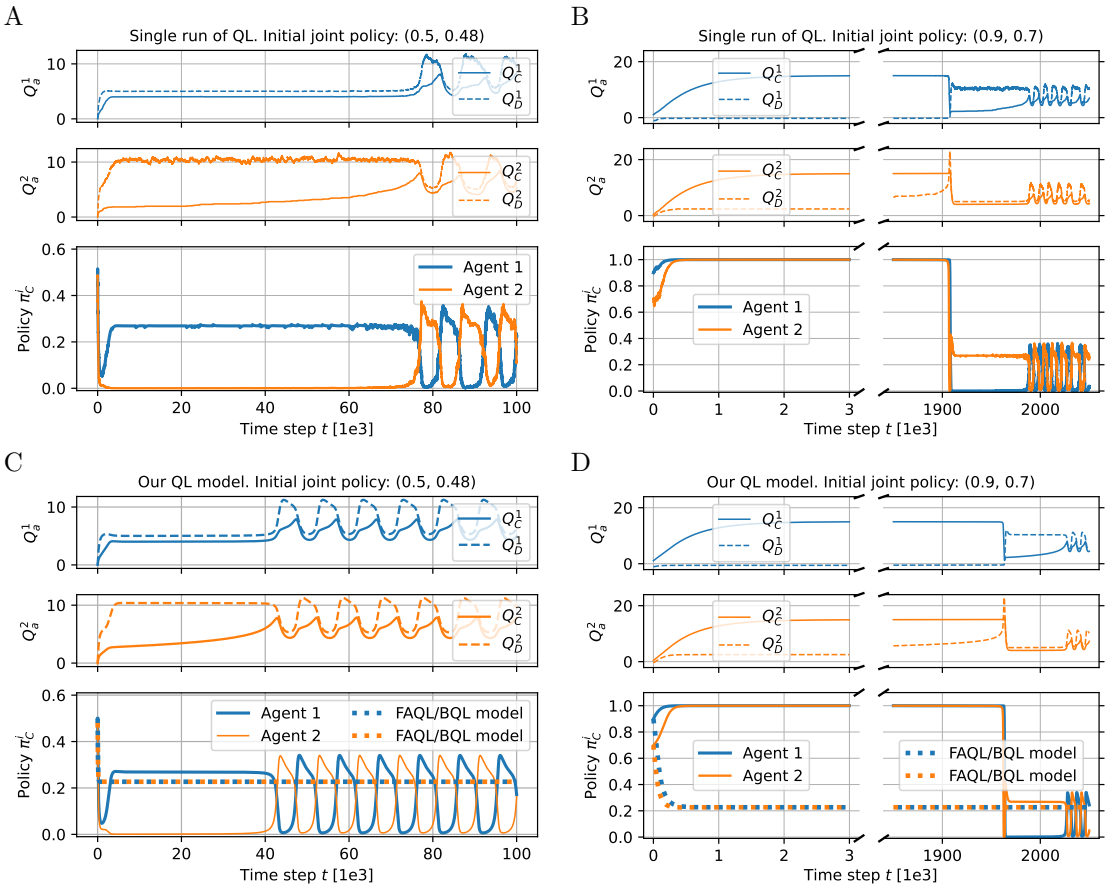


Figure 3.9: Comparison between a single run of independent Q-learning on the Prisoner's Dilemma (top panels: A, B) and our deterministic approximation model (bottom panels: C, D), defined by (3.20), for $T = 1$, $\alpha = 0.01$, $\gamma = 0.8$, $Q_{base} = 0$. Note that the depicted runs in A and B represent single instances of a stochastic process. Timings and trajectories vary across different runs. The first two subplots in each panel show the evolution of the Q-values $(Q_C^1, Q_D^1, Q_C^2, Q_D^2)$, while the third subplot illustrates the resulting probabilities of cooperation (π_C^1, π_C^2). The dotted policy trajectories in C and D represent previous approximation methods: FAQL, defined by (3.7), and BQL, defined by (3.14). The left panels (A, C) depict an initial joint policy $(\pi_C^1, \pi_C^2) = (0.5, 0.48)$, corresponding to Q-values $(0, 0, -0.04, 0.04)$ via (3.19). The right panels (B, C) show an initial joint policy $(\pi_C^1, \pi_C^2) = (0.9, 0.7)$, corresponding to Q-values $(1.1, -1.1, 0.4, -0.4)$ via (3.19).

3.4 Discussion

This chapter presented a historical overview of approximation methods of independent Q-learning with a Boltzmann policy in single-state environments. A new deterministic approximation model was proposed, which shows good agreement with the behaviour observed in experiments of the stochastic process. In particular, we demonstrated the occurrence of metastable phases in the Prisoner's Dilemma due to heterogeneous update frequencies and showed, that the observed oscillations for high discount factor values result from a supercritical Neimark-Sacker bifurcation in the 4D dynamics.

Our analysis underscores the importance of accounting for Q-value update frequencies to understand independent Q-learning dynamics. By incorporating these frequencies, our deterministic approximation captures behaviours that simpler policy-space approximations like the BQL/FAQL model cannot describe.

This distinction becomes particularly evident in the example of the Prisoner's Dilemma, where we have shown that the resulting 4D dynamics not only exhibit different transient dynamics than the 2D FAQL/BQL model, but can also prevent convergence to a joint policy, by altering the stability properties of equilibria. It is therefore crucial to recognise that the FAQL and BQL models do not represent classical, incremental Q-learning but rather specifically modified variants—a non-trivial nuance sometimes overlooked in the literature.

Our case study illustrates how using a Boltzmann policy with independent Q-learning can induce metastable phases by causing update frequencies to approach zero. As a result, the time required for the corresponding Q-value to reach its target can far exceed any realistic number of learning steps. These metastable phases could therefore easily be mistaken for equilibrium dynamics, posing a risk of misinterpretation. This highlights the importance of examining all dynamic variables (e.g., all Q-values) rather than focusing solely on the target variables of interest (e.g., the policy), as only a few variables might display perceptible drift during a metastable phase that indicates the instability [96].

We demonstrated this issue by addressing the question under what conditions independent incremental algorithms spontaneously "learn" to cooperate in social dilemmas [97]. Specifically, we showed that what might initially appear as stable cooperative behaviour in the Prisoner's Dilemma—seemingly contradicting the rationale of strategic interactions—is, in fact, a prolonged transient phase of the Q-learning process rather than an equilibrium. While such misinterpretations are relatively easy to avoid in simple environments like the Prisoner's Dilemma, they become far more challenging in complex environments with many agents, actions, and multiple equilibria. In this regard, a complementary dynamical systems perspective can be helpful.

Further, we showed how the moving target problem can cause stable oscillations, which prevent convergence to a joint policy. In this specific case study, this phenomenon is tied with higher values of the discount factor γ, which induce a Neimark-Sacker bifurcation. Nevertheless, the moving target problem is not unique to scenarios with $\gamma > 0$. In other settings, such as a public goods game with multiple agents, where $\gamma = 0$, different mechanisms can similarly intensify this issue, resulting in comparable phenomena. Although algorithmic adjustments such as batch learning, frequency-adjusted updates, or adopting alternative policy mechanisms (e.g. epsilon-greedy) can help mitigate the moving target problem, these are, essentially, *symptomatic* treatments. The underlying root cause of oscillatory or even more complex behaviour lies in the *non-stationarity* of the effective environment for each agent, a fundamental challenge in MARL [17].

If the independent Q-learning algorithm is interpreted as a model of actual learning processes occurring in humans or other organisms, the described complex dynamics should be considered interesting features worthy of further study. Most of the time, however, MARL algorithms are not meant as a model of something but as a numerical tool for finding certain types of strategic equilibria (such as Quantal Response Equilibria). For that application, the described complex dynamics should rather be considered a bug than a feature as it makes the MARL tool less useful. In that context, addressing the non-stationarity challenge is crucial for developing scalable MARL algorithms with robust convergence guarantees, which remains an open research problem [12]. Our model can serve as a valuable tool for future work in this regard.

While our deterministic model effectively captures the general behaviour of independent Q-learning, it has notable limitations. Specifically, the model cannot replicate the exact timing or outcomes of individual runs due to the inherent randomness and sensitivity to initial conditions present in the true dynamics. To address this, extending the model to include a noise term—converting the deterministic ordinary difference equations into stochastic difference equations—could enhance its fidelity. Such an approach would better account for the stochastic nature of these algorithms, leading to improved predictions of key phenomena, such as exit times from metastable phases and the average periods of oscillations.

It is also important to note that the model is not derived from a formal mathematical limit (e.g., the infinite batch-size limit of the BQL model). Instead, it is constructed by isolating the dynamics between consecutive time steps and studying the deterministic expectation of the next step based on the current values. While this approach yields good approximations for the games analysed in this thesis[5], it may overlook finer details in more complex games.

Looking ahead, future research could focus on extending our approximate model to multi-state environments, partially observable stochastic games and other Temporal-Difference learning algorithms, broadening its applicability to more complex settings. In the next chapter, we provide a brief outlook by presenting evidence that the insights gained in this chapter also apply to multi-state environments.

[5]The dynamics of the remaining games of table 2.1 are depicted in figure A.4 in appendix A.4.

Chapter 4

Outlook: Joint-Action Q-learning on the Iterated Prisoner's Dilemma

In the previous chapter, we focused on single-state environments, demonstrating that even in these simplified scenarios, multi-agent learning can exhibit intricate and often unexpected dynamics. By restricting the scope to single-state settings, we were able to isolate and analyse the foundational aspects of independent Q-learning. This chapter builds upon that foundation by expanding the discussion to multi-state environments, specifically through an investigation of joint-action Q-learning applied to the memory-one iterated Prisoner's Dilemma.

The primary goal of this short chapter is to establish the broader applicability of the insights derived from the single-state environment and to give an outlook of possible extensions in future work. We demonstrate that the dynamics observed in independent Q-learning persist in more intricate settings where state transitions play a role. In particular, we compare the behaviour of memory-one algorithms in the iterated Prisoner's Dilemma to the memory-zero dynamics explored previously.

4.1 Method

In chapter 3, we chose independent Q-learning on the two-agent Prisoner's Dilemma as the simplest and most paradigmatic MARL system. Here, we now increase the complexity incrementally, and allow the agents to use joint-action Q-learning on the iterated Prisoner's Dilemma with a memory length of one (algorithm 5).

Formally, the environment in this setup is defined by four environmental states, which are the four possible outcomes of the previous round (CC, CD, DC, DD).[1] At each time step, the agents observe the current state, select an action based on this information, and update their policy according to the reward they receive. The reward tensor is identical for all states $s \in \{CC, CD, DC, DD\}$:

$$\mathbf{R}_s = \begin{pmatrix} 3,3 & 0,5 \\ 5,0 & 1,1 \end{pmatrix}. \tag{4.1}$$

But in contrast to independent Q-learning, agents in this setup can now condition their policies on the joint action of the previous time step, making their policies and Q-values explicitly dependent on the state s. This raises the question of whether granting agents this capability is sufficient to enable the emergence of more sophisticated policies within the social dilemma.

Algorithm 5: Joint-Action Q-Learning with Memory-One Boltzmann Policy for Two-Agent Repeated Normal Form Games

Input: Action space $\mathcal{A}^i$, learning rate α^i, discount factor γ^i, temperature parameter T^i for each agent i, maximum time step t_{max}
Output: Learned Q-values Q^i_{s,a^i} for each agent i
Initialise Q^i_{s,a^i} arbitrarily for all $s \in \mathcal{S} = \mathcal{A}^1 \times \mathcal{A}^2$, $a^i \in \mathcal{A}^i$ and for each agent i
Initialise state s arbitrarily
$t = 0$
while *not reached* t_{max} **do**
  for *each agent* i **do**
    Choose action a^i with Boltzmann policy:
    $\pi^i(a^i|s) \leftarrow \frac{e^{Q^i_{s,a^i}/T^i}}{\sum_{b^i} e^{Q^i_{s,b^i}/T^i}}$ for all $a^i \in \mathcal{A}^i$
    $a^i \sim \pi^i(\cdot|s)$
  end
  Take joint action $\mathbf{a} = (a^1, a^2)$;
  for *each agent* i **do**
    Observe own reward r^i and next state s'
    Update Q-value:
    $Q^i_{s,a^i} \leftarrow Q^i_{s,a^i} + \alpha^i \left[r^i + \gamma^i \max_{b^i} Q^i_{s,b^i} - Q^i_{s,a^i} \right]$
  end
  Update state: $s \leftarrow s'$
  $t \leftarrow t + 1$
end

For the dynamics of *frequency-adjusted* joint-action Q-learning, see appendix A.2.

[1] In section 2.3.2, we defined joint-action learning to be based on individual observations and not on environmental states. Here however, since both agents in this setup always share the same observation (i.e., the previous joint action), we equate the individual observations to an environmental state s. If agents had differing memory lengths, this simplification would not be possible, and their observations would need to be treated as distinct internal states.

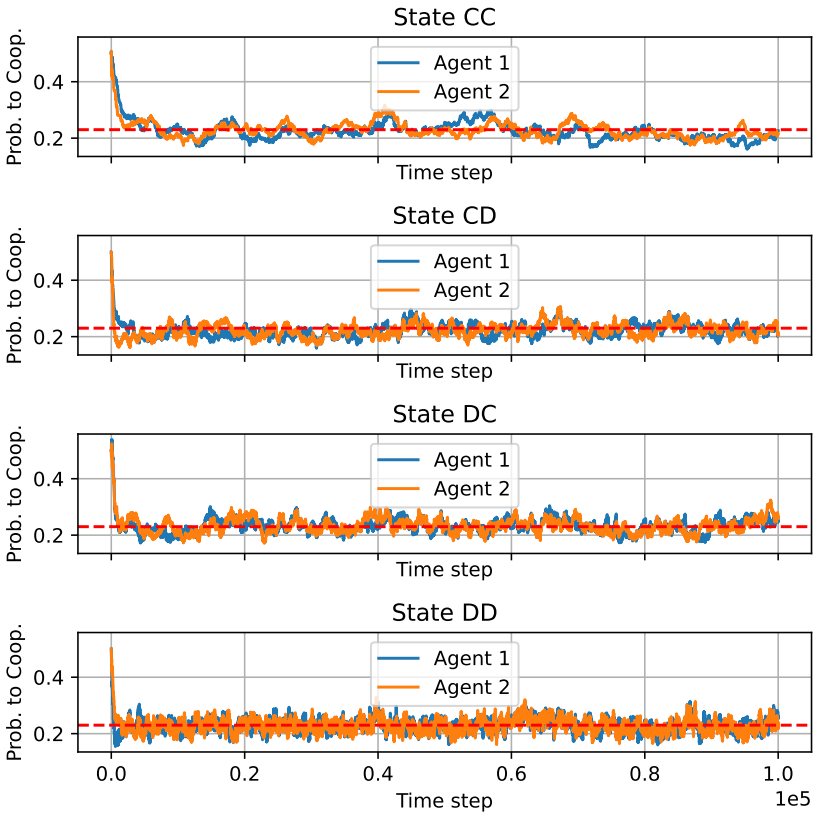


Figure 4.1: Time evolution of the policies of joint-action Q-learning (algorithm 5) on the iterated Prisoner's Dilemma with memory one for $T = 1, \alpha = 0.01, \gamma = 0$. All Q-values are initialised at zero. The dashed red line indicates the fixed point policy ($\pi^i_{C*} \approx 0.227$) of the memory-zero deterministic model, defined by (3.20).

4.2 First Results

For convenience, we set $T^i = 1, \alpha^i = 0.01$ and $Q^i_{s,a^i}(t = 0) = 0$, for all i, s, a^i, and investigate two values for the discount factor: $\gamma = 0$ and $\gamma = 0.8$. These parameter values ensure consistency with the setup of the prior chapter.

Discount factor $\gamma = 0$: Figure 4.1 illustrates the time evolution of the policy space for a single run with $\gamma = 0$. Across all four states, the policy trajectories resemble the dynamics observed in the single-state memory-zero setup from the previous chapter. After a couple of thousand time steps, the trajectories stabilise near $\pi^i_{\cdot,C} \approx 0.227$. Despite this stabilisation, the stochastic nature of action selection—driven by the constant temperature and learning rate—ensures that trajectories continue to fluctuate around this equilibrium, but without straying afar.

Discount factor $\gamma = 0.8$: In figure 4.2, we also observe the same qualitative behaviour as for the memory-one iterated Prisoner's Dilemma (see figure 3.9), but distributed over all four states. In the state DD we observe the familiar oscillations (see figure 3.9). The asymmetric states CD and DC also exhibit oscillatory patterns, with the distinction that the oscillations of the cooperative agent start earlier than of the defective agent. In the state CC, both agents seem to learn to defect after $\sim 3 \times 10^5$ steps. Given enough time however, the trajectories also begin to oscillate (see figure A.3 in the appendix).

In essence, these results suggest that the agents do not learn a more sophisticated policy for the given parameter and initial conditions. Instead, their long-term behaviour remains similar to the memory-zero setup.

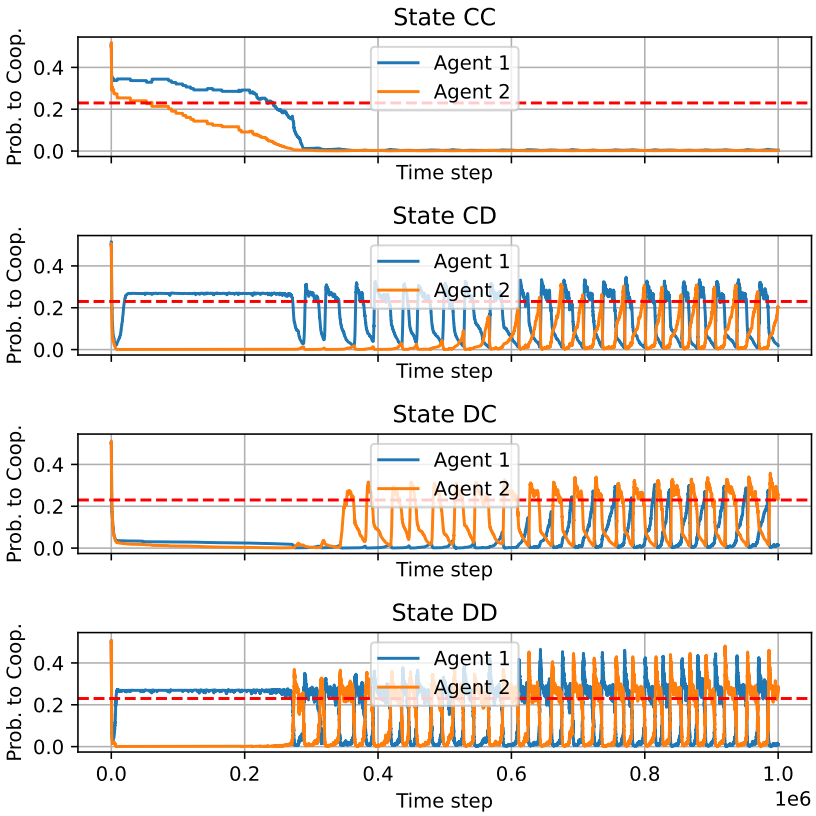


Figure 4.2: Time evolution of the policies of joint-action Q-learning (algorithm 5) on the iterated Prisoner's Dilemma with memory one for $T = 1, \alpha = 0.01, \gamma = 0.8$. All Q-values are initialised at zero. The dashed red line indicates the fixed point policy ($\pi^i_{C*} \approx 0.227$) of the memory-zero deterministic model, defined by (3.20).

4.3 Discussion

To fully understand the observed dynamics, one would need to examine the time evolution of all Q-values across all states, resulting in 16 variables, and analyse their relationships. For brevity, we limit our discussion to the key observation that, while the dynamics in multi-state environments retain the characteristics observed in the zero-memory single-state setup of chapter 3, new complexities emerge. These arise from the interplay between multiple states, which introduces further nuances to the collective learning process. Nevertheless, the core dynamics can largely be understood by analysing the simpler memory-zero case.

Chapter 5

Conclusion

This thesis explored how Multi-Agent Reinforcement Learning (MARL) can be analysed from a statistical physics perspective, leveraging in particular the tools of dynamical systems theory. MARL is a growing interdisciplinary field with increasing relevance in real-world applications. However, the multi-agent component introduces various challenges, including complicating the design of scalable algorithms and making learning dynamics significantly harder to evaluate and interpret compared to single-agent settings. By approximating the dynamics of the stochastic algorithms via deterministic equations, this study demonstrates how the tools of dynamical systems theory help uncover the collective learning dynamics.

Our case study on independent Q-learning in two-agent single-state games clarified why existing approximation models, such as the FAQL and BQL model, fail to capture the learning dynamics of actual Q-learning, instead representing simplified and idealised variants. These models assume replicator-type learning dynamics, which neglects the incremental nature of the algorithm and assumes that all Q-values get updated with perfect information.

By explicitly accounting for agents' update frequencies, we proposed a new approximation model, that showed good agreement with actual simulations. We showed that the resulting dynamics are fundamentally more complex and cannot be reduced from the higher-dimensional Q-space into the lower-dimensional policy space.

At the paradigmatic example of the Prisoner's Dilemma, we demonstrated how an approximation model can be used to efficiently analyse the effect of initial conditions and parameter choices on the collective learning behaviour. It revealed how the Boltzmann exploration policy can cause prolonged transient dynamics that might persist for billions of time steps, potentially being misinterpreted as equilibria.

Furthermore, it demonstrated that the moving target problem—agents trying to adapt to other agents' policies that are themselves changing on the same time scale—can be exacerbated by certain parameter settings to such an extent that it alters the stability of the equilibrium solution. Specifically, we showed how increasing the discount factor induces a supercritical Neimark–Sacker bifurcation, transforming the fixed point attractor into a stable limit cycle.

While numerical simulations alone might suffice to draw similar conclusions for simple games such as the Prisoner's Dilemma, this quickly becomes impractical for more complex systems featuring additional actions or agents. In these cases, without an understanding of the underlying dynamics, one could easily risk misinterpreting the outcomes of the learning process, such as mistaking transient phenomena for equilibria. In this regard, our case study demonstrated how a deterministic approximation model provides an elegant framework to illuminate the collective dynamics.

However, it is equally important to recognise the limitations of this approach. As the number of actions and agents increases, the complexity of the system grows exponentially, leading to an explosion in the number of equations required to describe it. This makes the construction of approximations challenging and renders analytical solutions infeasible. Moreover, approximation models should always be used in conjunction with numerical simulations to ensure that their predictions are validated against observed data.

As demonstrated in this thesis, even the simplest learning algorithm, Q-learning, when applied to the most basic game, the Prisoner's Dilemma, can exhibit surprisingly complex learning dynamics. This raises critical questions: If such simple case studies produce dynamics that are highly sensitive to initial conditions and parameter choices, how can MARL be reliably used to model complex scenarios where traditional methods fail? How can it be ensured that outcomes can be meaningfully interpreted? While MARL offers an appealing framework for modelling collective behaviour, its inherent intricacies can further challenge the clarity and reliability of analysis.

Overall, we conclude that adopting a statistical physics perspective is a valuable approach to addressing these challenges. One promising avenue for future research is extending the preliminary analysis of chapter 4, classifying the dynamical regimes in Boltzmann joint-action Q-learning, with a focus on the interplay between initial conditions and agents' memory length. This aligns with recent work by Meylahn and Janssen, who conducted a similar analysis for epsilon-greedy joint-action Q-learning [82]. Another direction could involve studying large population games, drawing inspiration from Wang et al.'s analysis of the regret minimisation algorithm using the master equation approach from statistical physics [98].

Finally, it is important to acknowledge that all methodologies discussed in this thesis are restricted to tabular algorithms. Given that the significant breakthroughs in MARL in recent years were driven by the incorporation of deep learning methods, future research should explore how a statistical physics perspective can be applied also to deep learning-based MARL, ensuring that the field remains aligned with state-of-the-art developments.

Bibliography

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html. 4, 6, 7, 9, 11, 12, 32, 48

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602. 4, 10

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 4

[4] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529 (7587):484–489, 2016. 4, 10

[5] Allan Dafoe, Edward Hughes, Yoram Bachrach, Tantum Collins, Kevin R McKee, Joel Z Leibo, Kate Larson, and Thore Graepel. Open problems in cooperative ai. *arXiv preprint arXiv:2012.08630*, 2020. 4

[6] Allan Dafoe, Yoram Bachrach, Gillian Hadfield, Eric Horvitz, Kate Larson, and Thore Graepel. Cooperative ai: machines must learn to find common ground. *Nature*, 593:33–36, 2021. doi: 10.1038/d41586-021-01170-0. 4

[7] William Nordhaus. Climate change: The ultimate challenge for economics. *American Economic Review*, 109(6):1991–2014, 2019. 4

[8] Katherine Richardson, Will Steffen, Wolfgang Lucht, Jørgen Bendtsen, Sarah E Cornell, Jonathan F Donges, Markus Drüke, Ingo Fetzer, Govindasamy Bala, Werner Von Bloh, et al. Earth beyond six of nine planetary boundaries. *Science advances*, 9(37):eadh2458, 2023. 4

[9] Paul G Harris. Collective action on climate change: The logic of regime failure. *Nat. Resources J.*, 47:195, 2007. 4

[10] Robert O Keohane and David G Victor. Cooperation and discord in global climate policy. *Nature Climate Change*, 6(6):570–575, 2016.

[11] Stefano Carattini, Simon Levin, and Alessandro Tavoni. Cooperation in the climate commons. *Review of environmental economics and policy*, 2019. 4

[12] Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. *Multi-Agent Reinforcement Learning: Foundations and Modern Approaches*. MIT Press, 2024. URL https://www.marl-book.com. 4, 6, 9, 12, 14, 16, 17, 22, 31, 32, 48, 50

[13] Lieve Helsen. Tackling climate change with machine learning. In *Conference on Neural Information Processing System (NeurIPS)-panelist workshop, Location: online*, 2021. 4

[14] Tianyu Zhang, Andrew Williams, Soham Phade, Sunil Srinivasa, Yang Zhang, Prateek Gupta, Yoshua Bengio, and Stephan Zheng. Ai for global climate cooperation: Modeling global climate negotiations, agreements, and long-term cooperation in rice-n, 2022. URL https://arxiv.org/abs/2208.07004. 4

[15] James Rudd-Jones, Fiona Thendean, and María Pérez-Ortiz. Crafting desirable climate trajectories with rl explored socio-environmental simulations, 2024. URL https://arxiv.org/abs/2410.07287. 4

[16] Pablo Hernandez-Leal, Michael Kaisers, Tim Baarslag, and Enrique Munoz de Cote. A survey of learning in multiagent environments: Dealing with non-stationarity, 2019. URL https://arxiv.org/abs/1707.09183. 4, 43, 48

[17] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33 (6):750–797, 2019. 4, 31, 50

[18] Nick Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, Oxford, 2014. ISBN 978-0-19-967811-2. 4, 27

[19] James Johnson. Artificial intelligence, drone swarming and escalation risks in future warfare. *The RUSI Journal*, 165(2):26–36, 2020. 4

[20] Andrei Kirilenko, Albert S Kyle, Mehrdad Samadi, and Tugkan Tuzun. The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, 72(3):967–998, 2017. 4

[21] Antonio Majdandzic, Boris Podobnik, Sergey V Buldyrev, Dror Y Kenett, Shlomo Havlin, and H Eugene Stanley. Spontaneous recovery in dynamical networks. *Nature Physics*, 10 (1):34–38, 2014. 4

[22] Wolfgang Barfuss. Dynamical systems as a level of cognitive analysis of multi-agent learning. *Neural Computing and Applications*, 34:1653–1671, 2022. doi: 10.1007/s00521-021-06117-0. URL https://doi.org/10.1007/s00521-021-06117-0. 4, 27, 28, 35, 38, 43

[23] Lars Onsager. Crystal statistics. i. a two-dimensional model with an order-disorder transition. *Physical Review*, 65(3-4):117, 1944. 5

[24] Joseph Klafter and Igor M Sokolov. *First steps in random walks: from tools to applications*. OUP Oxford, 2011. 5

[25] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002. 5

[26] Luis Gómez-Nava, Robert T Lange, Pascal P Klamser, Juliane Lukas, Lenin Arias-Rodriguez, David Bierbach, Jens Krause, Henning Sprekeler, and Pawel Romanczuk. Fish shoals resemble a stochastic excitable system driven by environmental perturbations. *Nature Physics*, 19(5):663–669, 2023. URL https://doi.org/10.1038/s41567-022-01916-1. 5

[27] Thomas Lux. Applications of statistical physics in finance and economics. In *Handbook of research on complexity*. Edward Elgar Publishing, 2009. 5

[28] Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 11(1):501–528, 2020. 5

[29] Michael Kaisers and Karl Tuyls. Faq-learning in matrix games: Demonstrating convergence near nash equilibria, and bifurcation of attractors in the battle of sexes. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011. 5, 31

[30] Tobias Galla. Intrinsic noise in game dynamical learning. *Phys. Rev. Lett.*, 103:198702, Nov 2009. doi: 10.1103/PhysRevLett.103.198702. URL https://link.aps.org/doi/10.1103/PhysRevLett.103.198702. 5, 27, 28

[31] Yuzuru Sato, Eizo Akiyama, and J. Doyne Farmer. Chaos in learning a simple two-person game. *Proceedings of the National Academy of Sciences*, 99(7):4748–4751, 2002. doi: 10.1073/pnas.032086299. URL https://www.pnas.org/doi/abs/10.1073/pnas.032086299. 5, 27, 28

[32] David Goll, Jobst Heitzig, and Wolfram Barfuss. Deterministic model of incremental multi-agent boltzmann q-learning: Transient cooperation, metastability, and oscillations, 2024. URL https://arxiv.org/abs/2501.00160. 5, 30

[33] Richard Bellman. *Dynamic Programming*. Dover Publications, 1957. ISBN 9780486428093. 9, 10

[34] John Tromp and Gunnar Farnebäck. Combinatorics of go. https://tromp.github.io, January 2016. Archived (PDF) from the original on January 25, 2016. Retrieved June 17, 2020. 10

[35] Melvin M Vopson. Estimation of the information contained in the visible matter of the universe. *AIP Advances*, 11(10), 2021. 10

[36] Wolfram Schultz, Peter Dayan, and P. Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997. doi: 10.1126/science.275.5306.1593. URL https://www.science.org/doi/abs/10.1126/science.275.5306.1593. 11

[37] Peter Dayan and Yael Niv. Reinforcement learning: the good, the bad and the ugly. *Current opinion in neurobiology*, 18(2):185–196, 2008. 11

[38] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Oxford, 1989. 12

[39] Christopher Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3):279–292, 1992. ISSN 1573-0565. doi: 10.1007/BF00992698. 12, 31, 32

[40] G. A. Rummery and M. Niranjan. On-line Q-learning using connectionist systems. Technical Report TR 166, Cambridge University Engineering Department, Cambridge, England, 1994. 12

[41] John Von Neumann and Oskar Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ, USA, 1944. 14

[42] Daan Bloembergen, Karl Tuyls, Daniel Hennes, and Michael Kaisers. Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research*, 53:659–697, 2015. 14, 19, 20, 22, 27, 33, 35, 43

[43] Robert Axelrod and William D. Hamilton. The evolution of cooperation. *Science*, 211 (4489):1390–1396, 1981. doi: 10.1126/science.7466396. URL https://www.science.org/doi/abs/10.1126/science.7466396. 15

[44] Tuomas W. Sandholm and Robert H. Crites. Multiagent reinforcement learning in the iterated prisoner's dilemma. *Biosystems*, 37(1):147–166, 1996. ISSN 0303-2647. doi: https://doi.org/10.1016/0303-2647(95)01551-5. URL https://www.sciencedirect.com/science/article/pii/0303264795015515. 16, 23, 31, 32

[45] John F Nash. Equilibrium points in n-person games. *Proc Natl Acad Sci U S A*, 36(1): 48–49, 1950. doi: 10.1073/pnas.36.1.48. 18

[46] J. Smith and G. R. Price. The logic of animal conflict. *Nature*, 246(5427):15–18, 1973. doi: 10.1038/246015a0. 19, 20

[47] Jörgen W Weibull. *Evolutionary game theory*. MIT press, 1997. 20

[48] Karl Tuyls, Katja Verbeeck, and Tom Lenaerts. A selection-mutation model for q-learning in multi-agent systems. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '03, page 693–700, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581136838. doi: 10.1145/860575.860687. URL https://doi.org/10.1145/860575.860687. 22, 27, 28, 30, 31, 33, 35, 43

[49] Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Batch Reinforcement Learning*, pages 45–73. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-27645-3. doi: 10.1007/978-3-642-27645-3_2. URL https://doi.org/10.1007/978-3-642-27645-3_2. 22, 35

[50] Ming Tan. *Multi-agent reinforcement learning: independent vs. cooperative agents*, page 487–494. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997. ISBN 1558604952. 23

[51] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. *AAAI/IAAI*, 1998(746-752):2, 1998. 24

[52] Karl Tuyls and Gerhard Weiss. Multiagent learning: Basics, challenges, and prospects. *Ai Magazine*, 33(3):41–41, 2012. 26

[53] John C. Harsanyi and Reinhard Selten. *A General Theory of Equilibrium Selection in Games*. Number 0262582384 in MIT Press Books. The MIT Press, December 1988. ISBN ARRAY(0x55c22ec0). URL https://ideas.repec.org/b/mtp/titles/0262582384.html. 26

[54] Steven H. Strogatz. *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*. CRC Press, 2nd edition, 2015. doi: 10.1201/9780429492563. URL https://doi.org/10.1201/9780429492563. 27

[55] Yurii Bolotin, Anatoli Tur, and Vladimir Yanovsky. *Chaos: Concepts, control and constructive use*. Springer, 2009. 27

[56] Niall Shanks. Modeling biological systems: the belousov–zhabotinsky reaction. *Foundations of Chemistry*, 3(1):33–53, 2001. ISSN 1572-8463. doi: 10.1023/A:1011434929814. URL https://doi.org/10.1023/A:1011434929814. 27

[57] Carmen Rocsoreanu, Adelina Georgescu, and Nicolaie Giurgiteanu. *The FitzHugh-Nagumo model: bifurcation and dynamics*, volume 10. Springer Science & Business Media, 2012. 27

[58] N. Wunderling, A. S. von der Heydt, Y. Aksenov, S. Barker, R. Bastiaansen, V. Brovkin, M. Brunetti, V. Couplet, T. Kleinen, C. H. Lear, J. Lohmann, R. M. Roman-Cuesta, S. Sinet, D. Swingedouw, R. Winkelmann, P. Anand, J. Barichivich, S. Bathiany, M. Baudena, J. T. Bruun, C. M. Chiessi, H. K. Coxall, D. Docquier, J. F. Donges, S. K. J. Falkena, A. K. Klose, D. Obura, J. Rocha, S. Rynders, N. J. Steinert, and M. Willeit. Climate tipping point interactions and cascades: a review. *Earth System Dynamics*, 15(1):41–74, 2024. doi: 10.5194/esd-15-41-2024. URL https://esd.copernicus.org/articles/15/41/2024/. 27

[59] P.C.H Martens. Applications of non-linear methods in astronomy. *Physics Reports*, 115(6): 315–378, 1984. ISSN 0370-1573. doi: https://doi.org/10.1016/0370-1573(84)90184-4. URL https://www.sciencedirect.com/science/article/pii/0370157384901844. 27

[60] Tilman Börgers and Rajiv Sarin. Learning through reinforcement and replicator dynamics. *Journal of Economic Theory*, 77(1):1–14, 1997. ISSN 0022-0531. doi: https://doi.org/10.1006/jeth.1997.2319. URL https://www.sciencedirect.com/science/article/pii/S002205319792319X. 27, 28, 33

[61] Yuzuru Sato and James P. Crutchfield. Coupled replicator equations for the dynamics of learning in multiagent systems. *Physical Review E*, 67(1), January 2003. ISSN 1095-3787. doi: 10.1103/physreve.67.015206. URL http://dx.doi.org/10.1103/PhysRevE.67.015206. 28, 33

[62] Yuzuru Sato, Eizo Akiyama, and James P. Crutchfield. Stability and diversity in collective adaptation. *Physica D: Nonlinear Phenomena*, 210(1–2):21–57, October 2005. ISSN 0167-2789. doi: 10.1016/j.physd.2005.06.031. URL http://dx.doi.org/10.1016/j.physd.2005.06.031.

[63] David S Leslie and Edmund J Collins. Individual q-learning in normal form games. *SIAM Journal on Control and Optimization*, 44(2):495–514, 2005. 35, 43

[64] Segismundo S Izquierdo, Luis R Izquierdo, and Nicholas M Gotts. Reinforcement learning dynamics in social dilemmas. *Journal of Artificial Societies and Social Simulation*, 11(2):1, 2008. 31

[65] Drew Fudenberg and David K Levine. Learning and equilibrium. *Annu. Rev. Econ.*, 1(1): 385–420, 2009.

[66] Naoki Masuda and Hisashi Ohtsuki. A theoretical analysis of temporal difference learning in the iterated prisoner's dilemma game. *Bulletin of mathematical biology*, 71:1818–1850, 2009. 31

[67] Michael Wunder, Michael L Littman, and Monica Babes. Classes of multiagent q-learning dynamics with epsilon-greedy exploration. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1167–1174, 2010. 28, 31

[68] Naoki Masuda and Mitsuhiro Nakamura. Numerical analysis of a reinforcement learning model with the dynamic aspiration level in the iterated prisoner's dilemma. *Journal of theoretical biology*, 278(1):55–62, 2011. 31

[69] Tobias Galla. Cycles of cooperation and defection in imperfect learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(08):P08007, aug 2011. doi: 10.1088/1742-5468/2011/08/P08007. URL https://dx.doi.org/10.1088/1742-5468/2011/08/P08007. 28

[70] Ardeshir Kianercy and Aram Galstyan. Dynamics of boltzmann q learning in two-player two-action games. *Phys. Rev. E*, 85:041145, Apr 2012. doi: 10.1103/PhysRevE.85.041145. URL https://link.aps.org/doi/10.1103/PhysRevE.85.041145. 31, 35, 43

[71] Aram Galstyan. Continuous strategy replicator dynamics for multi-agent q-learning. *Autonomous agents and multi-agent systems*, 26:37–53, 2013. 31, 35, 43

[72] Tobias Galla and J. Doyne Farmer. Complex dynamics in learning complicated games. *Proceedings of the National Academy of Sciences*, 110(4):1232–1236, 2013. doi: 10.1073/pnas.1109672110. URL https://www.pnas.org/doi/abs/10.1073/pnas.1109672110.

[73] James B. T. Sanders, J. Doyne Farmer, and Tobias Galla. The prevalence of chaotic dynamics in games with many players. *Scientific Reports*, 8(1):4902, 2018. ISSN 2045-2322. doi: 10.1038/s41598-018-22013-5. URL https://doi.org/10.1038/s41598-018-22013-5. 28

[74] Wolfram Barfuss, Jonathan F. Donges, and Jürgen Kurths. Deterministic limit of temporal difference reinforcement learning for stochastic games. *Phys. Rev. E*, 99:043305, Apr 2019. doi: 10.1103/PhysRevE.99.043305. URL https://link.aps.org/doi/10.1103/PhysRevE.99.043305. 28, 30, 35, 43

[75] Shuyue Hu, Chin-wing Leung, and Ho-fung Leung. Modelling the dynamics of multiagent q-learning in repeated symmetric games: a mean field theoretic approach. *Advances in Neural Information Processing Systems*, 32, 2019.

[76] Shuyue Hu, Chin-Wing Leung, Ho-fung Leung, and Harold Soh. The dynamics of q-learning in population games: A physics-inspired continuity equation model. *arXiv preprint arXiv:2203.01500*, 2022. 31, 43

[77] Chen Chu, Yong Li, Jinzhuo Liu, Shuyue Hu, Xuelong Li, and Zhen Wang. A formal model for multiagent q-learning dynamics on regular graphs. In *IJCAI*, pages 194–200, 2022.

[78] Wolfram Barfuss and Richard P Mann. Modeling the effects of environmental and perceptual uncertainty using deterministic reinforcement learning dynamics with partial observability. *Physical Review E*, 105(3):034409, 2022.

[79] Stefanos Leonardos and Georgios Piliouras. Exploration-exploitation in multi-agent learning: Catastrophe theory meets game theory. *Artificial Intelligence*, 304:103653, 2022. 27, 31, 43

[80] Michael Kaisers and Karl Tuyls. Frequency adjusted multi-agent q-learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 309–316, 2010. 31, 35, 43

[81] Lu Wang, Danyang Jia, Long Zhang, Peican Zhu, Matja**v**z Perc, Lei Shi, and Zhen Wang. Lévy noise promotes cooperation in the prisoner's dilemma game with reinforcement learning. *Nonlinear Dynamics*, 108(2):1837–1845, 2022.

[82] Janusz M Meylahn and Lars Janssen. Limiting dynamics for q-learning with memory one in symmetric two-player, two-action games. *Complexity*, 2022(1):4830491, 2022. 57

[83] Quentin Bertrand, Juan Duque, Emilio Calvano, and Gauthier Gidel. Q-learners can provably collude in the iterated prisoner's dilemma. *arXiv preprint arXiv:2312.08484*, 2023.

[84] Yuki Usui and Masahiko Ueda. Symmetric equilibrium of multi-agent reinforcement learning in repeated prisoner's dilemma. *Applied Mathematics and Computation*, 409:126370, 2021. 31

[85] Laetitia Matignon, Guillaume J Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012. 31

[86] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869*, 2020. 31

[87] Monica Babes, Michael Wunder, and Michael Littman. Q-learning in two-player two-action games. In *Proc. AAMAS*, pages 1–6, 2009. 31

[88] Johannes Zschache. Melioration learning in iterated public goods games: The impact of exploratory noise. *The Journal of Mathematical Sociology*, 42(1):1–16, 2018.

[89] Brian Mintz and Feng Fu. Evolutionary multi-agent reinforcement learning in group social dilemmas. *arXiv preprint arXiv:2411.10459*, 2024. 31, 43

[90] Daeyeol Lee, Michelle L Conroy, Benjamin P McGreevy, and Dominic J Barraclough. Reinforcement learning and decision making in monkeys during a competitive game. *Cognitive brain research*, 22(1):45–58, 2004. 32

[91] Soyoun Kim, Jaewon Hwang, Hyojung Seo, and Daeyeol Lee. Valuation of uncertain and delayed rewards in primate prefrontal cortex. *Neural Networks*, 22(3):294–304, 2009. 32

[92] John G. Cross. A Stochastic Learning Model of Economic Behavior*. *The Quarterly Journal of Economics*, 87(2):239–266, 05 1973. ISSN 0033-5533. doi: 10.2307/1882186. URL `https://doi.org/10.2307/1882186`. 33

[93] Karl Friston. The free-energy principle: a unified brain theory? *Nature reviews neuroscience*, 11(2):127–138, 2010. 35

[94] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995. 38

[95] Bouke Klein Teeselink, Dennie van Dolder, Martijn J van den Assem, and Jason D Dana. High-stakes failures of backward induction. *Games and Economic Behavior*, 2024. 38

[96] Tim Kittel, Jobst Heitzig, Kevin Webster, and Jürgen Kurths. Timing of transients: quantifying reaching times and transient behavior in complex systems. *New Journal of Physics*, 19(8):083005, 2017. 50

[97] Wolfram Barfuss and Janusz M. Meylahn. Intrinsic fluctuations of reinforcement learning promote cooperation. *Scientific Reports*, 13(1), January 2023. ISSN 2045-2322. doi: 10.1038/s41598-023-27672-7. URL https://www.nature.com/articles/s41598-023-27672-7. Number: 1 Publisher: Nature Publishing Group. 50

[98] Zhen Wang, Chunjiang Mu, Shuyue Hu, Chen Chu, and Xuelong Li. Modelling the dynamics of regret minimization in large agent populations: a master equation approach. In *IJCAI*, pages 534–540, 2022. 57

Appendix A

Appendix

A.1 FAQL on Single-state Prisoner's Dilemma

Figure A.1 shows the good agreement of the FAQL/BQL model with stochastic realisations of frequency-adjusted Q-learning, defined by (3.8), on the Prisoner's Dilemma, irregardless of Q-value initialisations (Q_{base}) or discount factor values (γ).

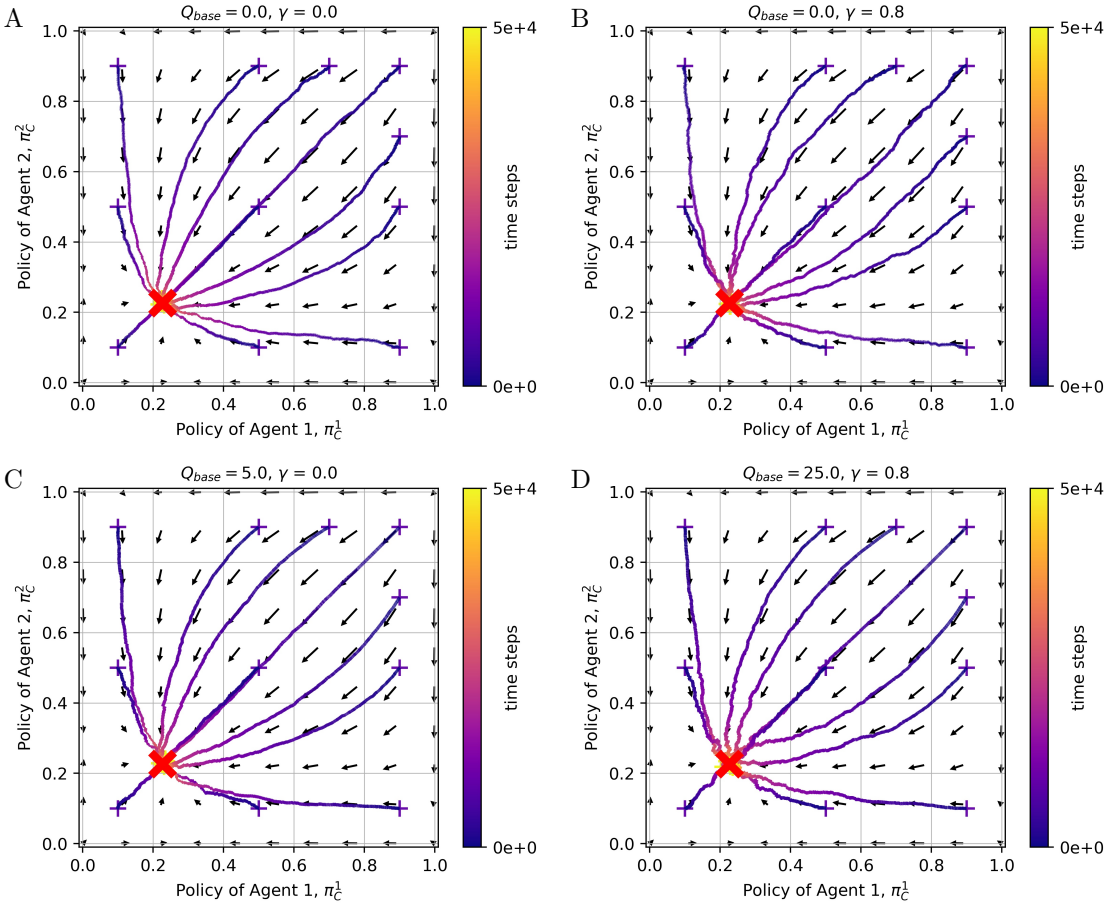


Figure A.1: Comparison between policy trajectories of a single run of independent frequency-adjusted Q-learning ($T = 1, \alpha = 0.01, \beta = 0.01$) on the Prisoner's Dilemma and the FAQL/BQL model, as done in figure 3.5. Top panels (A, B): $Q_{base} = \min(\mathbf{R})/(1-\gamma)$. Bottom panels (C, D): $Q_{base} = \max(\mathbf{R})/(1-\gamma)$. Left panels (A, C): $\gamma = 0$. Right panels (B, D): $\gamma = 0.8$.

A.2 FAQL on Memory-One Iterated Prisoner's Dilemma

Figure A.2 shows the smoother learning dynamics of frequency-adjusted joint-action Q-learning (algorithm 5 paired with the update rule (3.8)), compared to the classic algorithm (figure 4.1 and 4.2). After an initial period, the oscillations of the policies in A.2.B relax to the fixed point of the memory-zero FAQL/BQL model dynamics.

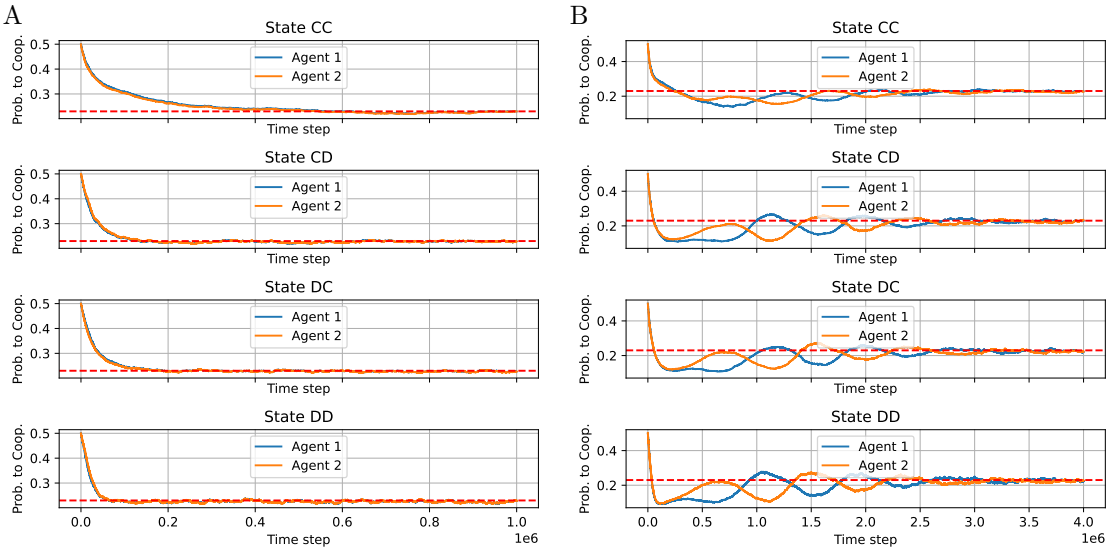


Figure A.2: Time evolution of the policies of frequency-adjusted joint-action Q-learning on the iterated Prisoner's Dilemma with memory one for $T = 1, \alpha = 0.01$ and two different discount factor values. All Q-values are initialised at zero. The dashed red line indicates the fixed point policy ($\pi^i_{C*} \approx 0.227$) of the FAQL/BQL model for the memory-zero Prisoner's Dilemma. (A): $\gamma = 0$ (B): $\gamma = 0.8$

A.3 Joint-Action Q-learning: State CC

Figure A.3 shows that eventually, after about 5 million time steps, the policy trajectories in the state CC also begin to fluctuate.

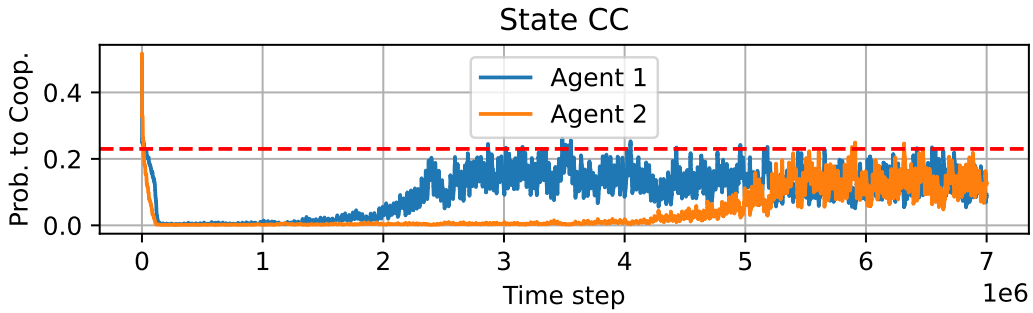


Figure A.3: Extended time evolution of the policies of Joint-Action Q-learning (algorithm 5) on the Iterated Prisoner's Dilemma with memory one for $T = 1, \alpha = 0.01, \gamma = 0.8$ for the state CC. All Q-values are initialised at zero. The dashed red line indicates the fixed point policy ($\pi^i_{C*} \approx 0.227$) of the FAQL/BQL model for the memory-zero Prisoner's Dilemma.

A.4 Deterministic Learning Dynamics of other Games

Figure A.4 depicts the deterministic dynamics of (3.20) for the remaining games of table 2.1 for $\gamma = 0$.

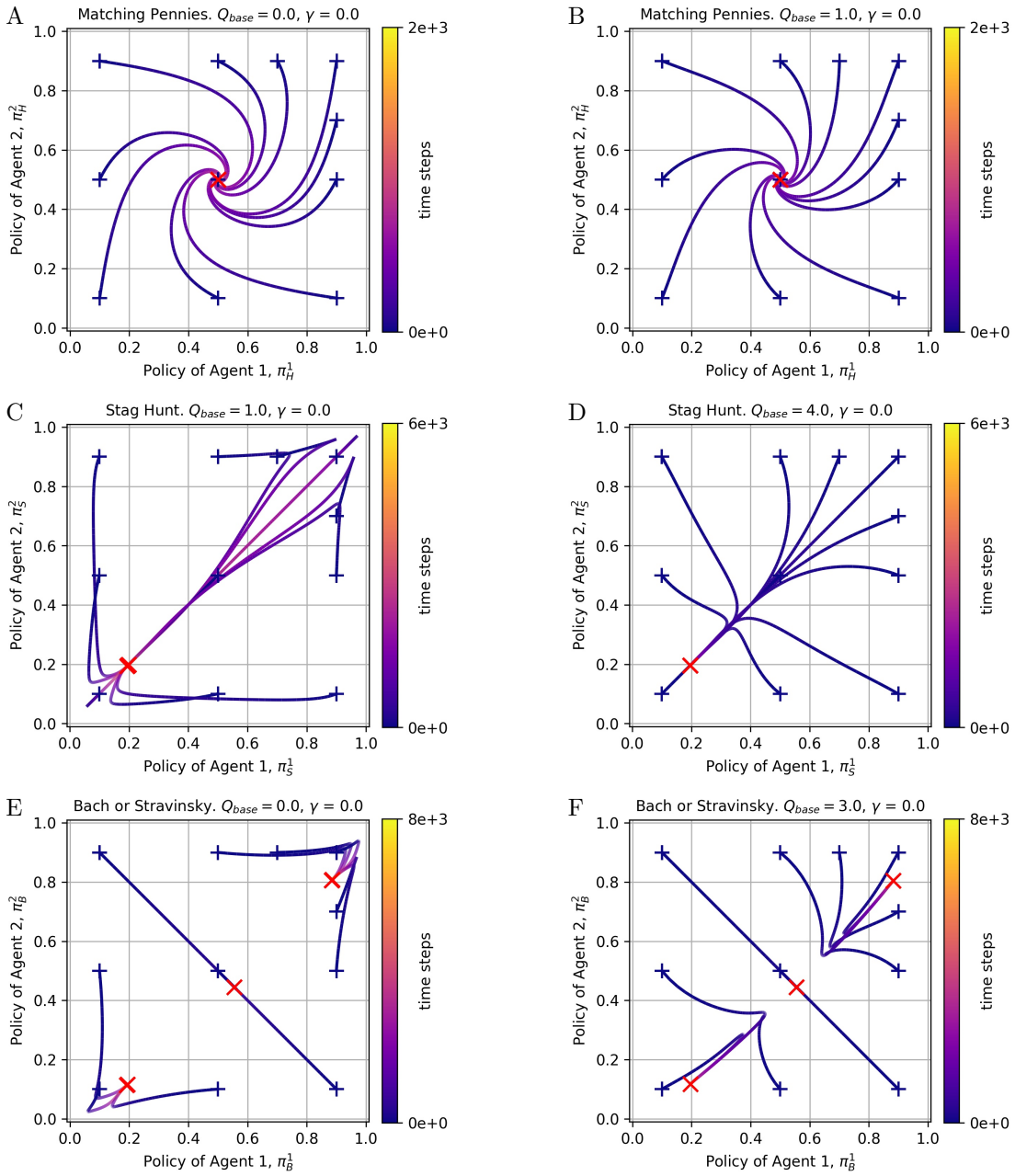


Figure A.4: Projection of our 4D deterministic model of independent Q-learning, defined by (3.20), into 2D policy space for $T = 1$, $\alpha = 0.01$, $\gamma = 0$ similar to figure 3.6. Left panels (A, C, E): $Q_{base} = \min(\mathbf{R})/(1-\gamma)$. Right panels (B, D, F): $Q_{base} = \min(\mathbf{R})/(1-\gamma)$.

Code availability All computer code is made publicly available on https://github.com/golldavid.