MATHEMATICAL MODELING OF REINFORCEMENT LEARNING IN MULTI-AGENT SYSTEMS

AIDAN ANDREWS MATH 495 FINAL

Aidan S. Andrews
MATH 495
University Of Illinois Urbana-Champaign
Champaign, IL 61820
aidansa2@illinois.edu

April 30 - May 19, 2025

ABSTRACT

In this paper, I construct and analyze mathematical models of reinforcement learning (RL) with an emphasis on agentic and multi-agent systems. I will be using "We" to be more formal in this paper, however I am still the sole author. We begin by formulating RL in rigorous terms, defining Markov Decision Processes (MDPs) as the foundation and reviewing fundamental solution methods such as value iteration and Q-learning. We then survey interaction structures in multi-agent systems – including fully cooperative, fully competitive, and mixed scenarios – and discuss how each can be represented by continuous-time or stochastic models (ODEs, PDEs, SDEs). In particular, we explore networked and mean-field formulations that approximate large-agent interactions, and consider spatial diffusion models for agents in continuous space. Building on these insights, we propose a novel mathematical model that bridges RL with multi-agent dynamics: a system of nonlinear differential equations coupling RL value learning with multi-agent strategy evolution. This model extends classical frameworks (replicator equations, Lotka–Volterra predator-prey, epidemic models) to an RL context. We derive equilibrium conditions corresponding to learned joint behaviors (e.g. Nash equilibria or coordinated policies), analyze their stability via linearization, and identify bifurcations such as the onset of oscillatory learning or multiple stable outcomes as system parameters (like learning rate or reward structure) vary. The analysis yields theoretical insight into emergent coordination, cycling behaviors, and convergence in multi-agent learning. We conclude by discussing implications for the design of multi-agent systems, emphasizing how rigorous dynamical modeling can inform stable learning algorithms and predict complex phenomena (like chaos or convergence) in multi-agent reinforcement learning.

Keywords Reinforcement Learning · Multi-Agent Systems · Dynamical Systems · Differential Equations · Stability Analysis

1 Introduction

Reinforcement learning (RL) provides a formal framework for sequential decision-making in uncertain environments Wikipedia [2023a]. In a typical RL setting, an *agent* interacts with an *environment* over discrete time steps. At each step, the agent observes a state, takes an action, and receives a reward signal, with the goal of maximizing cumulative reward over time. Mathematically, this interaction is encapsulated by a Markov Decision Process (MDP). An MDP is defined by a tuple (S, A, P, R): S is the set of states; A is the set of actions; $P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a)$ is the state transition probability for taking action a in state s; and $R_a(s, s')$ is the immediate reward obtained when transitioning from s to s' under action a Wikipedia [2023a]. The agent's objective is to learn an optimal *policy* $\pi : S \to A$ that maximizes expected cumulative reward (often discounted) from each state. Formally, given a policy π, the *value function* at state s is the expected discounted return $V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, \pi\right]$, where $0 \leq \gamma < 1$ is a discount

factor and r_t is the reward at time t Wikipedia [2023b]. An optimal policy π^* maximizes this value for all states, and its value function $V^*(s) = \max_\pi V^\pi(s)$ satisfies the *Bellman optimality equation* Wikipedia [2023a]:

$$V^*(s) \;=\; \max_{a\in A} \mathbb{E}\big[\, R(s,a,s') + \gamma\, V^*(s')\,\big] \;=\; \max_{a\in A} \sum_{s'\in S} P(s'|s,a)\,\Big(R(s,a,s') + \gamma\, V^*(s')\Big)\,. \tag{1}$$

Dynamic programming algorithms are used to compute V^* and π^*. **Value iteration** applies the Bellman update iteratively to converge to the optimal value function. **Policy iteration** alternates between evaluating a given policy and improving it, and also converges to π^*. In contrast, **Q-learning** is a model-free RL algorithm that directly learns the optimal *action-value function* $Q^*(s,a)$ (the expected return for taking action a in state s and following the optimal policy thereafter). The Q-learning update rule is given by:

$$Q(s_t,a_t) \;\leftarrow\; Q(s_t,a_t) + \alpha\Big(r_{t+1} + \gamma\, \max_{a'} Q(s_{t+1},a') \;-\; Q(s_t,a_t)\Big)\,, \tag{2}$$

where α is a learning rate GeeksforGeeks [2023]. This stochastic iterative update provably converges to the optimal Q^* under standard conditions, and the greedy policy $a_t = \arg\max_a Q(s_t,a)$ converges to the optimal policy.

While single-agent RL has a well-developed theory, **multi-agent reinforcement learning (MARL)** presents additional challenges. In MARL, multiple agents coexist in a shared environment, each receiving (potentially different) rewards and updating its policy. This setting can be modeled as a stochastic game (also called a Markov game) – a multi-agent extension of an MDP where the transition probabilities and rewards depend on the joint actions of all agents. Formally, for N agents indexed by $i \in 1,\dots,N$, we define state space S and individual action sets A_i for each agent. At each step, all agents choose actions $\mathbf{a} = (a_1,\dots,a_N)$, resulting in a transition $s \to s'$ with probability $P_{\mathbf{a}}(s,s')$ and a reward $R^i_{\mathbf{a}}(s,s')$ for each agent i Wikipedia [2023b]. Each agent seeks to maximize its own cumulative reward. This framework subsumes many scenarios: from fully cooperative (agents share the same reward function) to fully competitive (one agent's gain is another's loss) to general mixed motives.

Multi-agent environments give rise to rich dynamics. Agents are concurrently learning and the environment includes other learners, making the system *non-stationary* from any single agent's perspective. The outcome is often conceptualized in game-theoretic terms: agents seek an equilibrium strategy profile (e.g. a Nash equilibrium) where no agent can unilaterally improve its reward. However, learning algorithms might not always converge to equilibrium; they may oscillate or exhibit complex behavior. To understand and predict such behaviors, we turn to **mathematical modeling** of multi-agent learning dynamics, using tools from dynamical systems (ordinary and partial differential equations, stochastic processes). By analyzing these models, we can characterize conditions for convergence or instability, draw analogies to well-studied dynamical systems (like predator-prey or epidemic models), and potentially design better multi-agent learning algorithms.

In this paper, we develop continuous-time mathematical models for multi-agent RL and analyze their behavior. In Section 2, we establish the mathematical foundations of single-agent RL (MDPs, Bellman equations, Q-learning) that will underlie our multi-agent extensions. Section 3 surveys different multi-agent interaction structures – **pure cooperation**, **pure competition**, and **mixed** – and examines which of these lend themselves to tractable mathematical modeling via ODEs, PDEs or SDEs. We discuss networked vs. mean-field and spatial formulations of multi-agent dynamics. In Section 4, we propose a novel dynamical model that bridges RL with multi-agent interactions: specifically, a set of coupled nonlinear ODEs inspired by replicator equations and other biological dynamics, which incorporates the learning processes of agents Sato and Crutchfield [2003]. We draw analogies between components of our model and classical models (predator-prey, replicator, SIR, etc.) to build intuition. In Section 5, we analyze the proposed model: we derive equilibrium conditions corresponding to stationary policies or strategy profiles, perform local stability analysis (identifying eigenvalues that indicate convergence or divergence), and explore how varying key parameters induces bifurcations in the system's qualitative behavior. Finally, Section 6 concludes with implications of this analysis for multi-agent system design – for example, how reward structures or learning rates might be tuned to ensure coordination and avoid unstable outcomes – and outlines directions for further theoretical research bridging RL and dynamical systems.

Throughout, we adopt a first-person scholarly perspective, reflecting on the modeling choices and analytical results. Our aim is to provide a self-contained yet comprehensive treatment that rigorously connects reinforcement learning with multi-agent dynamical systems theory.

2 Background: Reinforcement Learning and Markov Decision Processes

Markov Decision Processes (MDPs). As introduced above, an MDP provides a mathematical formalism for an agent's sequential decision making under uncertainty. Formally, an MDP is (S, A, P, R, γ), where S is the state space and A is the action space (both can be finite or infinite sets). At each discrete time step t, the agent observes state $s_t \in S$ and chooses an action $a_t \in A$. The environment then transitions to a new state $s_{t+1} \sim P(\cdot \mid s_t, a_t)$ according to the transition probability distribution P, and the agent receives a reward $r_{t+1} = R(s_t, a_t, s_{t+1})$. The Markov property implies that the distribution of s_{t+1} depends only on the current state and action, not on the past history. The process continues for an infinite horizon (or until some terminal condition). The discount factor $0 \leq \gamma < 1$ weights future rewards geometrically, ensuring that the infinite sum of rewards converges.

The goal in an MDP is to find an optimal *policy* $\pi^* : S \to A$ that maximizes the agent's expected cumulative discounted reward. The optimal value function $V^*(s) = \max_\pi \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, \pi]$ satisfies the Bellman optimality condition given earlier. This can be derived from the principle of optimality: the value of a state under an optimal policy must equal the immediate reward from the best action plus the discounted value of the next state Bellman [1957]. Solving the Bellman equation directly (a system of $|S|$ nonlinear equations) is often done iteratively via **value iteration**: one initializes $V_0(s)$ arbitrarily (e.g. zero) and iteratively applies the update

$$V_{k+1}(s) \leftarrow \max_{a \in A} \sum_{s'} P(s'|s, a) \Big(R(s, a, s') + \gamma V_k(s') \Big), \tag{3}$$

for all s. This update is a contraction mapping in the supremum norm (with factor $\gamma < 1$), guaranteeing convergence to the unique fixed point V^*.

Once V^* is found, an optimal policy is obtained by choosing in each state an action that achieves the max in the Bellman equation. Alternatively, one can compute the optimal *Q-value* function $Q^*(s, a) = \mathbb{E}[r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a]$. $Q^*(s, a)$ satisfies the Bellman equation in its action-value form: $Q^*(s, a) = \mathbb{E}_{s'}[R(s, a, s') + \gamma \max_{a'} Q^*(s', a')]$. The advantage of Q-values is that a policy can be directly obtained by $a^*(s) = \arg\max_a Q^*(s, a)$, and one can learn Q^* without a model of P or R.

Q-learning. Q-learning is a classic model-free RL algorithm that uses temporal-difference updates to learn $Q^*(s, a)$ online. As the agent interacts with the environment, it updates its estimate $Q(s, a)$ for the state-action pair (s, a) experienced. The update rule was given above:

$$Q_{\text{new}}(s_t, a_t) = Q(s_t, a_t) + \alpha \Big(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \Big). \tag{4}$$

This can be seen as stochastically approximating the Bellman optimality operator. In the limit of small learning rate α and sufficient exploration of state-action pairs, $Q(s, a)$ will converge to $Q^*(s, a)$. The term in parentheses is the temporal difference (TD) error. Many RL algorithms (SARSA, TD(λ), policy gradient methods) exist, but Q-learning provides a simple and concrete example we will extend to multi-agent settings.

We note that one can interpret the learning process in continuous time as well. In the limit $\alpha \to 0$, the difference equation for Q can be rescaled to a differential equation (this is an example of viewing learning dynamics through the lens of **ODE methods** for stochastic approximation) Sato and Crutchfield [2003]. Roughly, one obtains $\dot{Q}(s, a) = \mathbb{E}[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s, a)]$, where the expectation is over the randomness in state transitions and possibly exploration. The stable equilibrium of this ODE corresponds to the Bellman equation solution Q^*. This continuous-time perspective will be useful when we consider multi-agent learning dynamics, as it allows applying dynamical systems theory (fixed points, stability, etc.) to the learning process itself.

Policy iteration and actor-critic. Another class of RL methods (important in multi-agent contexts as well) are policy-based methods. These maintain an explicit policy π_θ (often parameterized by θ) and update it in the direction of performance improvement, e.g. via gradient ascent on V^π or related objectives. In continuous time, such updates can be described by differential equations on the policy parameters or directly on the policy distribution (leading to replicator-like equations under certain parameterizations, as we will see). The **actor-critic** approach couples a policy (actor) update with a value function (critic) update. This two-timescale process can also be studied with ODEs or stochastic differential equations: the critic evaluation is often faster, and the actor update slower, leading to a quasi-steady-state analysis where the actor's ODE sees the critic converged to an approximate value function.

This background in single-agent RL provides the groundwork. Next, we turn to multi-agent systems, where multiple such learning processes occur simultaneously and interact.

3 Interaction Structures in Multi-Agent Systems

In a multi-agent system, each agent i has its own policy π_i, value function V_i, and possibly its own state (in decentralized settings) or a shared state (in fully observable games). Agents may cooperate or compete, and the overall dynamics depend critically on the *interaction structure*. We categorize interactions into three canonical types Wikipedia [2023b]:

- **Pure Competition (Zero-Sum Games):** Two (or more) agents have perfectly opposing goals. In the two-player zero-sum case, the sum of rewards is zero, so one agent's reward is the negation of the other's. Chess and Go are classic examples: one player's win is the other's loss. In such settings, each agent aims to maximize its reward which equates to minimizing the other's reward. The game has a *value* v and an equilibrium where one agent's optimal policy and the other's optimal counter-policy yield payoff v to one and $-v$ to the other Wikipedia [2023b]. A Nash equilibrium in zero-sum games is a saddle point that can often be found via solving a minimax problem. Traditional results from game theory (von Neumann's minimax theorem) guarantee the existence of an equilibrium in mixed strategies. From a learning perspective, however, finding these equilibria via naive RL is non-trivial – dynamics can oscillate as each agent keeps adapting to the other. We will see that competitive learning dynamics can be analogous to *predator-prey cycles* or oscillatory behavior. Indeed, continuous-time models of learning in zero-sum games have revealed cycles and even chaos (e.g. the classic rock–paper–scissors game yields periodic or chaotic orbits under certain learning rules Sato and Crutchfield [2003]). We will model a two-agent zero-sum scenario as a case study for our ODE framework.
- **Pure Cooperation (Team Games):** All agents share an identical reward function, so their interests are perfectly aligned Wikipedia [2023b]. A fully cooperative multi-agent problem can be viewed as a single-agent problem on the joint state-action space of all agents (though with exponential growth in complexity as agent count increases). Examples include agents working together to reach a common goal, or multiple robots collaboratively moving a object. In such settings, an optimal joint policy maximizes the common return; any learning algorithm that converges to optimal single-agent behavior on the joint space will yield a globally optimal solution. However, decentralized learning (where each agent updates its own policy based on local observations) can be tricky: agents must somehow coordinate updates since each agent's actions affect the others' rewards. Nevertheless, cooperative settings are generally more benign for learning dynamics – since all agents seek the same objective, one doesn't expect sustained oscillations or divergent behavior caused purely by reward conflicts. Mathematical modeling often leverages potential games or consensus dynamics: if a common potential function (analogous to a global value function) exists, the learning dynamics can sometimes be derived as gradient ascent on that potential. For example, in a cooperative control problem one can define a Lyapunov function that all agents collectively seek to minimize, yielding convergent dynamics. We will discuss how cooperative multi-agent learning might be cast as a gradient system (a dissipative system that converges to an optimum), and how concepts like **logistic growth** could describe the improvement of group performance over time (e.g. a sigmoidal learning curve where initial progress is slow, mid-phase is rapid, and then performance saturates at optimum – analogous to logistic population growth with a carrying capacity).
- **Mixed-Sum (General-Sum) Settings:** These are intermediate scenarios that combine elements of competition and cooperation Wikipedia [2023b]. Most real-world multi-agent problems are of this nature: agents have partly overlapping interests. For instance, autonomous cars on a road all want to reach their destinations (common interest in avoiding collisions and traffic jams), but each also wants to minimize its own travel time (individual interest). Such situations can be modeled by general-sum games, which may have multiple Nash equilibria, Pareto optima that differ from Nash equilibria, and the possibility of *social dilemmas* (e.g. the prisoner's dilemma where individual rationality leads to collectively suboptimal outcomes). Mixed settings are the most challenging for analysis because the dynamics can exhibit multiple attractors or complex oscillatory patterns depending on the exact payoff structure. For example, in an iterated prisoner's dilemma setting with adaptive strategies, one might see the emergence of cooperation or defection depending on initial conditions – reminiscent of **bistability** in dynamical systems (two stable equilibria with an unstable threshold in between). In replicator dynamics (from evolutionary game theory) for general-sum games, one often finds that if a game has multiple Nash equilibria, some may be stable and others unstable, and the system's long-term behavior depends on initial strategy distributions (analogous to how an SIR model's outcome can depend on initial infected fraction if there's a herd immunity threshold, or how a bistable chemical reaction depends on initial concentration relative to a threshold).

In summary, cooperative games often allow a potential function formulation (leading to gradient-like dynamics), competitive games often lead to Hamiltonian or oscillatory dynamics (conservative or marginally stable cycles in the replicator equation for zero-sum games), and mixed games can exhibit a mix of these behaviors (with possible multiple equilibria and complex basins of attraction). A key question is which of these structures lend themselves to *meaningful mathematical modeling* with differential equations or similar tools.

Networked vs. mean-field interactions: Another important aspect is how agents interact (who interacts with whom). In *fully-connected* scenarios, every agent's actions affect every other (as in a game where all players compete or cooperate together). In other cases, interactions are local or networked – e.g. each agent interacts only with its neighbors on a graph (communication network, physical proximity, etc.). Network structure can be represented by coupling in the equations: each agent i might have state $x_i(t)$ and its dynamics $\dot{x}_i$ depends on neighbors j in some graph. For a large number of agents on a network, analyzing N coupled ODEs can be difficult. One approach is a **mean-field approximation**: if the network is large and well-mixed (or random regular), one might assume each agent interacts with an average effect of others. In the limit $N \to \infty$, this leads to *mean-field games (MFG)* Wikipedia [2023c] or *mean-field control*, where a *representative agent* interacts with the statistical distribution of the population. Mathematically, mean-field game theory yields a description in terms of coupled PDEs: typically a Hamilton–Jacobi–Bellman (HJB) equation for the value function of a representative agent and a Fokker–Planck (or Kolmogorov) equation for the evolution of the population state distribution Wikipedia [2023c]. This is a powerful framework because it collapses an N-agent problem to a pair of PDEs whose dimensionality does not grow with N. For example, one equation describes how agents optimally respond (optimal control), and the other describes how the population's state distribution flows over time according to those responses Wikipedia [2023c]. Mean-field models are analytically tractable in certain regimes and can capture emergent aggregate behavior (like convergence to a mean-field equilibrium which approximates a large-N Nash equilibrium).

However, mean-field models assume a kind of homogeneity (agents are exchangeable and interactions are symmetric). If the network is highly structured or the number of agents is not asymptotically large, one might have to analyze the full network dynamics. In such cases, one can sometimes still derive reduced models: e.g. for a large random network, use a graphon (continuum limit of graphs) to derive a *graphon mean-field* model, or for spatial lattices, use PDEs with diffusion to approximate local interaction spread.

Spatial diffusion models: If agents move in continuous space and interact locally (e.g. predators and prey moving in a plane, or agents spreading information), one can incorporate spatial variables into the model. For example, one could have agent density $u(t, x)$ that satisfies a reaction-diffusion PDE: the reaction term accounts for local interaction dynamics (learning, strategy update, reproduction, etc.) and the diffusion term represents movement or spatial spread. In evolutionary game dynamics, *reaction-diffusion replicator equations* have been studied, where spatial pattern formation can occur due to a Turing instability if cooperators and defectors segregate, for instance. In a multi-agent RL context, spatial models might describe how behaviors or strategies propagate through space as agents physically move and learn. For instance, consider an epidemic analogy: suppose a certain effective behavior (strategy) is initially adopted by a few agents in a spatial domain and can spread to others via local interaction (agents observe or learn from neighbors). This could be modeled similarly to an epidemic **SIR model**: with S = susceptible (agents not using the strategy), I = infected (agents currently trying the new strategy), R = removed or fully convinced (agents permanently adopting the strategy). The transitions $S \to I \to R$ could be driven by local learning events instead of virus transmission. This is a *metaphorical* use of SIR dynamics in a multi-agent learning context, illustrating how ideas from epidemic modeling might inform how quickly a new strategy percolates through an agent population, or how stubbornness (analogous to immunity) might prevent adoption.

In summary, mathematical modeling of multi-agent systems can take several forms:

- **Coupled ODEs** for small N (each agent or agent group represented by state variables). These ODEs might be derived as the limit of discrete-time update algorithms (like Q-learning or policy gradient) as step-size $\to 0$. They allow analysis of equilibria (which often correspond to game-theoretic solution concepts like Nash or correlated equilibria) and stability (convergence of learning or oscillatory divergence).
- **Stochastic differential equations (SDEs)** if one retains some randomness (e.g. due to finite learning rate or exploration noise). For instance, one can model Q-learning with constant α as a stochastic approximation that tracks an ODE with added noise. Techniques from stochastic stability can then be applied.
- **Mean-field PDEs** for large populations, yielding a deterministic description of the distribution of agents over states or strategies. These are useful for emergent behavior in large systems, and can sometimes be analyzed using tools from PDE theory (fixed points, stability, traveling waves for strategy invasion, etc.).
- **Network/graph models**, which may be reducible to low-dimensional ODEs in special cases (e.g. all-to-all symmetric interactions) or require numerical analysis in general.

Each approach has trade-offs in tractability versus fidelity. In the next section, we propose a specific continuous-time model that aims to *bridge RL principles with multi-agent dynamics*, drawing inspiration from the above frameworks (especially replicator dynamics and ODE approximations of Q-learning). The model will then be analyzed using stability and bifurcation methods akin to those used in nonlinear ODE systems from mathematical biology.

4 A Dynamical Systems Model for Multi-Agent Reinforcement Learning

We now propose a mathematical model that integrates reinforcement learning with multi-agent interaction dynamics. Our model considers a set of agents learning and interacting, and represents their learning update rules and strategy adaptations as a system of coupled nonlinear ODEs. The design of the model is guided by two requirements: (1) it should reduce to known dynamics (like the replicator equation or gradient ascent) in special cases, to build trust that it captures sensible behavior; (2) it should incorporate key RL elements such as value estimation and reward feedback, distinguishing it from a purely evolutionary or ecological dynamics model.

Model setup: For clarity, we start with a two-agent system (extensions to N agents will be discussed qualitatively). Consider two agents, X and Y, playing a repeated game (or equivalently, navigating a shared environment). Each has a set of possible actions. For simplicity, let each agent have N discrete actions (the extension to continuous action spaces is possible via differential inclusions or by considering limit $N \to \infty$ as a continuum of strategies). Let $x_i(t)$ denote the probability that agent X chooses action i at time t, and $y_j(t)$ the probability that agent Y chooses action j. Thus $x(t) = (x_1, \dots, x_N)$ is X's mixed strategy at time t, and $y(t)$ is Y's strategy; we have $\sum_i x_i = \sum_j y_j = 1$ at all times. These probabilities evolve as the agents learn from rewards.

We also introduce *value estimates* or *reward memories* for each action. Let $Q_i^X(t)$ be agent X's current estimate of the value (cumulative reward potential) of playing action i, and similarly $Q_j^Y(t)$ for agent Y and action j. These Q-values are reminiscent of Q-learning, except that here we have a stateless (single-state repeated game) scenario for simplicity – the only dynamic element is strategy probabilities. If the environment has state, one could imagine each state has its own Q values and the analysis would be more complex; focusing on a single state or symmetric repeated interaction allows us to highlight the learning dynamics themselves.

The coupled ODE model is as follows:

- **Value estimate dynamics:** When agent X plays action i and Y plays action j, suppose X receives reward R_{ij}^X and Y receives R_{ij}^Y (this defines the payoff bi-matrix of the game). We model the evolution of Q_i^X as a leaky integration of received rewards:

$$\dot{Q}_i^X \;=\; R_{ij}^X(t) \;-\; \alpha_X \, Q_i^X \,, \tag{5}$$

  and similarly $\dot{Q}_j^Y = R_{ij}^Y(t) - \alpha_Y Q_j^Y$. Here $R_{ij}^X(t)$ means the reward obtained at time t when X took action i and Y took action j. In expectation, $R_{ij}^X(t)$ would be R_{ij}^X weighted by the probability that those actions were taken. The parameter $\alpha_X \in [0, 1)$ is a *memory decay rate* for agent X. If $\alpha_X = 0$, Q_i^X integrates (sums up) all rewards obtained when action i was played (i.e. *perfect memory* of past payoffs); if $\alpha_X > 0$, past rewards gradually fade, which means the agent gives more weight to recent outcomes. This ODE is a continuous-time limit of a Q-learning update: indeed, the discrete version was $Q_i^X(n+1) - Q_i^X(n) = R_{ij}^X - \alpha_X Q_i^X(n)$, which corresponds to a learning rate of 1 for the immediate reward and a decay α_X for the old value (thus implicitly $\gamma = 0$ here for simplicity since we treat each stage game independently – extension to cumulative reward would introduce a γ term as well).

- **Strategy dynamics:** How do the mixed strategies $x(t)$ and $y(t)$ change over time? We posit that agents adjust their action probabilities in the direction of actions that have higher **estimated** value. A common choice (in evolutionary dynamics and learning models) is a **softmax** or logit rule: the probability of an action is proportional to $\exp(\beta \times \text{estimated value})$, where β is a positive *intensity of selection* or learning sensitivity parameter. In discrete time, the model by Sato et al. used exactly this: after updating Q, agent X chooses its next action i with probability

$$x_i = \frac{\exp(\beta_X Q_i^X)}{\sum_k \exp(\beta_X Q_k^X)} \,, \tag{6}$$

  and similarly for Y. In continuous time, one can derive the resulting ODE for x_i by differentiating this softmax and plugging in $\dot{Q}_i^X$. Alternatively, one can write a first-order ODE that captures the intuition: *increase the probability of actions with above-average value, and decrease the probability of actions with below-average value*. This leads to a form of the **replicator equation**. Specifically, we can write:

$$\dot{x}_i \;=\; \beta_X \, x_i \left(Q_i^X \;-\; \sum_k x_k \, Q_k^X \right), \tag{7}$$

  for each $i = 1, \dots, N$. Here $\sum_k x_k Q_k^X$ is the current average value estimate under X's strategy (i.e. the expected Q-value at time t). Thus if action i's value Q_i^X exceeds the average, $\dot{x}_i > 0$ (its probability increases),

and if it is below average, $\dot{x}_i < 0$ (probability decreases). The factor x_i in front ensures probabilities remain between 0 and 1 and sums to 1 (this is identical in form to the replicator dynamics from evolutionary game theory, except using *learned* values Q in place of true payoff). Similarly for agent Y:

$$\dot{y}_j \;=\; \beta_Y\, y_j \left(Q_j^Y \;-\; \sum_k y_k\, Q_k^Y \right). \tag{8}$$

Putting these pieces together, the full model for the two-agent learning system is a set of $2N + 2N = 4N$ first-order ODEs (for $i = 1, \ldots, N$ and $j = 1, \ldots, N$):

$$\begin{aligned}
\dot{Q}_i^X &= \mathbb{E}[R_{ij}^X] - \alpha_X Q_i^X, \quad i = 1, \ldots, N, \\
\dot{Q}_j^Y &= \mathbb{E}[R_{ij}^Y] - \alpha_Y Q_j^Y, \quad j = 1, \ldots, N, \\
\dot{x}_i &= \beta_X x_i \Big(Q_i^X - \sum_k x_k Q_k^X \Big), \quad i = 1, \ldots, N, \\
\dot{y}_j &= \beta_Y y_j \Big(Q_j^Y - \sum_k y_k Q_k^Y \Big), \quad j = 1, \ldots, N.
\end{aligned} \tag{9}$$

where $\mathbb{E}[R_{ij}^X]$ denotes the expected reward to X when X plays action i and Y plays according to y_j. If we assume that at any given moment, the probability that Y plays j is $y_j(t)$, then $\mathbb{E}[R_{ij}^X] = \sum_j y_j R_{ij}^X$. In other words, $\dot{Q}_i^X = \sum_j y_j R_{ij}^X - \alpha_X Q_i^X$. This is the expected immediate reward for action i minus the decay. Similarly $\dot{Q}_j^Y = \sum_i x_i R_{ij}^Y - \alpha_Y Q_j^Y$.

This system can be recognized as a form of **coupled replicator–learning dynamics**. In the special case where $\alpha_X = \alpha_Y = 0$ (no forgetting) and β_X, β_Y are small (so that Q changes slowly relative to strategy adaptation), the Q values will accumulate the average payoffs. If we start at $Q_i^X(0) = 0$, then $Q_i^X(t)$ will simply be the time integral of rewards gained by playing i. In a stationary scenario, $Q_i^X(t) \to$ (average payoff per round) $\times\, t$, which grows unbounded – thus in practice one would normalize or focus on the differences $Q_i^X - Q_k^X$. In fact, Sato *et al.* show that with perfect memory, the above equations reduce to the *multi-population replicator equation* for the game Sato and Crutchfield [2003]. Intuitively, if agents remember all past rewards, their *rate of increase* of preference for an action is proportional to how much better that action's total payoff has been compared to others – which is exactly the evolutionary replicator dynamic interpretation. In this limit, if the game is zero-sum, the joint dynamics of (x, y) follow a *conservative* flow (like a Hamiltonian system) that can cycle or even behave chaotically (as reported for the rock–paper–scissors game) Sato and Crutchfield [2003].

On the other hand, for $\alpha_X, \alpha_Y > 0$, the Q-values represent more recent performance. This introduces *dissipation* into the system – old information decays, so the system effectively has a "forgetting" or "learning rate" that can dampen oscillations. Sato & Crutchfield (2003) note that with memory decay, the system can display a full range of nonlinear behaviors including limit cycles and chaos, but generally it becomes dissipative rather than conservative Sato and Crutchfield [2003]. In practice, α serves to stabilize learning in many cases: it is akin to lowering the effective weight of long-past experiences, which can help when the environment (or other agent) is non-stationary.

The parameters β_X, β_Y control how aggressively the strategies follow the estimated values. In the limit $\beta \to \infty$, the softmax becomes a best-response (choose the current best estimated action with probability 1). In that case, the dynamics become discontinuous (instantaneous switching when an action becomes estimated as better than another) Sato and Crutchfield [2003]. For analysis, it's easier to consider β finite so that strategies adjust smoothly. If β is very small, strategies change only gradually (almost random exploration) Sato and Crutchfield [2003]. Our model assumes a moderate β such that strategy adaptation is responsive but smooth.

Relation to classic models: We highlight analogies between this model and familiar dynamical systems:

- The (x_i) and (y_j) equations are in the form of **replicator equations**, a hallmark of evolutionary game dynamics. In fact, if Q_i^X were the true payoff against $y(t)$ at time t, $\dot{x}_i = \beta_X x_i [U_i^X(y) - \bar{U}^X(y)]$ (where $U_i^X(y) = \sum_j y_j R_{ij}^X$ is the expected payoff for X using i against Y's mixed strategy y, and $\bar{U}^X = \sum_i x_i U_i^X$ is the average payoff) would be the standard replicator equation for agent X. In our case Q_i^X is a *learned estimate* of $U_i^X(y)$, and α, β govern how closely Q tracks the actual payoffs and how quickly strategies respond. Thus our model can be seen as a generalization of replicator dynamics to include learning inertia.
- The two-agent replicator dynamics in a zero-sum game is formally analogous to the **Lotka–Volterra** predator-prey equations in certain cases. For example, the classical rock–paper–scissors cyclical dynamic is topologically

similar to a limit cycle in predator-prey systems (with three species cycling). In our model, when $\alpha = 0$ and for zero-sum games, we get a non-dissipative system that often has closed orbits (neutrally stable cycles) rather than asymptotically stable equilibria – reminiscent of a center in Hamiltonian systems (predator-prey is a special case with an invariant loop). When $\alpha > 0$ introduces dissipation, these cycles can become spirals converging to equilibrium (a Hopf bifurcation, as we discuss later, where the fixed point goes from neutral to attracting).

- If the game is fully cooperative, $R_{ij}^X = R_{ij}^Y$ is a common reward. In that case, effectively Q^X and Q^Y will track the same values (assuming similar α), and x and y should both move towards favoring the actions that maximize that common value. One can show that in this case, x and y will tend to synchronize (especially if initialized similarly). The dynamics could then be reduced to a single replicator equation for the common interest, plus the value learning. This scenario might have a Lyapunov function (e.g. the common expected return) that increases over time, meaning the learning dynamics are **gradient-like**. This would guarantee convergence to an optimum (no oscillation). Such behavior is analogous to **logistic growth** in that the performance (common payoff) might sigmoidally increase over time as agents converge to the optimal joint strategy.
- The introduction of memory decay α is analogous to friction or damping in physics, which often stabilizes a system Sato and Crutchfield [2003]. Without decay, replicator dynamics in a zero-sum game is like a neutrally stable center (undamped oscillator). With decay, we have a damped oscillator that can settle to equilibrium (if the equilibrium is stable). This is directly analogous to damped predator-prey systems (where prey growth and predator decline are no longer perfectly balanced, so eventually the system settles to a fixed point instead of sustaining cycles). In dynamical systems terms, as α increases from 0, a pair of purely imaginary eigenvalues (at $\alpha = 0$ in zero-sum game) move into the left half-plane, inducing a Hopf bifurcation that turns the center into a stable spiral (if the game is zero-sum). We will elaborate on stability conditions below.
- One can also interpret the system in a higher-level manner: it is a **two-time-scale system** if β (strategy update rate) is high relative to α (value update/decay rate). If strategies adapt quickly and Q changes slowly, then at any instant the strategies are near a best response to the current Q values. In the limit $\beta \to \infty$ (best-response dynamics) and small α, this approximates **fictitious play** in game theory (where each agent best-responds to the average of the other's past play). Fictitious play is known to converge in some classes of games (e.g. zero-sum, potential games) and not in others. Our model can thus mimic fictitious play with a smoothing parameter. If instead α is small (so memory is long) but β is moderate, the Q values might evolve on a slower time scale capturing the long-term payoffs, while the strategies rapidly oscillate Sato and Crutchfield [2003] – a separation of time scales approach might analyze the fast strategy dynamics assuming Q quasi-static, and then the slow drift of Q.

Having defined the model, we next analyze its equilibria, stability, and bifurcations. We emphasize that while the model is inspired by prior work (notably Sato & Crutchfield), we frame the analysis in a broader context, highlighting analogies to course topics like predator-prey and SIR models where appropriate.

5 Analysis of the Proposed Model

Equilibria: Equilibria of the ODE system correspond to fixed points of the learning dynamics. An equilibrium requires $\dot{x}_i = 0, \dot{y}_j = 0, \dot{Q}_i^X = 0, \dot{Q}_j^Y = 0$ for all i, j. From $\dot{Q}_i^X = 0$ we get $\sum_j y_j R_{ij}^X - \alpha_X Q_i^X = 0$. In vector form, $\dot{\mathbf{Q}}^X = 0$ implies

$$\mathbf{R}^X \, y = \alpha_X \, \mathbf{Q}^X, \tag{10}$$

where $\mathbf{R}^X$ is the payoff matrix for X (of size $N \times N$, with entries R_{ij}^X). Similarly $\mathbf{R}^Y x = \alpha_Y \mathbf{Q}^Y$. Meanwhile, $\dot{x}_i = 0$ implies either $x_i = 0$ or Q_i^X equals the weighted average $\sum_k x_k Q_k^X$; in other words, at equilibrium all actions used with non-zero probability must have equal estimated value (otherwise probability would shift). In game theory terms, this is the condition of a *best-response mixed strategy*: if an action has higher value than another, the lower one should have zero probability (since otherwise the strategy would adjust). Thus, at equilibrium, the support of x (the set of actions with $x_i > 0$) should be composed of actions that all have equal Q_i^X, and any action not in the support must have a Q^X that is *no greater* than those in the support (so that the agent is indifferent or the unused actions are worse). The same holds for y with respect to $\mathbf{Q}^Y$.

Combining these conditions, one can see that an equilibrium of the learning dynamics corresponds to a *Nash equilibrium* of the underlying game, under the assumption that the Q-values correctly reflect the actual payoffs. If the learning has converged, Q_i^X should equal the true expected payoff of action i against Y's equilibrium strategy (because $\mathbf{R}^X y = \alpha_X \mathbf{Q}^X$ and at equilibrium Q^X is stationary, if $\alpha_X > 0$ we must have $Q_i^X = (\mathbf{R}^X y)_i / \alpha_X$; if $\alpha_X = 0$, then

the equation says $\mathbf{R}^X y = 0$, which in zero-sum games is consistent with equilibrium payoffs, but in general $\alpha = 0$ just means Q can drift). Essentially, the condition reduces to: x and y form a Nash equilibrium of the matrix game defined by R^X and R^Y, and Q^X, Q^Y are proportional to the payoff vectors $R^X y$ and $(R^Y)^\top x$ respectively. If $\alpha > 0$, the proportionality constant doesn't matter for the strategy indifference condition, so Q values simply adjust their scale to satisfy the equilibrium equations.

For concreteness, consider a simple example: a **two-action** game ($N = 2$). Let X's actions be A and B, and Y's actions C and D. Let the payoffs be given by matrices (for X and Y):

$$R^X = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \qquad R^Y = \begin{pmatrix} a' & b' \\ c' & d' \end{pmatrix}, \tag{11}$$

where $R^X_{11} = a$ is the reward to X if X plays A and Y plays C, etc., and $R^Y_{11} = a'$ is the reward to Y in that case. A mixed strategy for X is characterized by probability x of A (and $1 - x$ of B), and for Y by y of C (and $1 - y$ of D). A Nash equilibrium (x^*, y^*) in such a 2×2 game can be found by indifference conditions: X is indifferent between A and B, and Y indifferent between C and D (if both mix). That yields two linear equations:

$$a\, y^* + b\,(1 - y^*) = c\, y^* + d\,(1 - y^*) \tag{12}$$

for X's indifference, and

$$a'\, x^* + c'\,(1 - x^*) = b'\, x^* + d'\,(1 - x^*) \tag{13}$$

for Y's. These can be solved for x^*, y^* (assuming an interior equilibrium exists). Our equilibrium conditions for the ODE system reduce exactly to these when $\dot{x} = \dot{y} = 0$ with $x, y \in (0, 1)$. The Q values at equilibrium would satisfy $Q^X_A = Q^X_B$ and $Q^Y_C = Q^Y_D$, reflecting the indifference.

Trivial equilibria also exist: for example, pure strategy equilibria where $x_i = 1$ for some i and $y_j = 1$ for some j, and those actions are mutual best responses. In the learning dynamics, those correspond to absorbing states if the learning steers the system into one of those corners.

Stability analysis: We examine the local stability of an equilibrium by linearizing the ODEs around that point. This leads to a Jacobian matrix that can be analyzed for eigenvalues with positive or negative real parts. Because of the structure of our equations, analytical results are known from evolutionary game theory for the replicator part. We supplement that with the effect of the Q-value coupling.

Consider first a *fully cooperative* scenario. In this case, the game has a single maximizer of common reward (assuming a unique optimum). The equilibrium will be at the joint action (or mixed strategy) that maximizes $R^X = R^Y$ for both. We expect this equilibrium to be **stable** under learning. In fact, one can often construct a Lyapunov function in cooperative games. For instance, the total expected reward $U(x, y) = x^\top R^X y$ can serve as a Lyapunov function: our dynamics (with $\alpha > 0$) should increase this quantity until it reaches a maximum at equilibrium. Intuitively, if all agents share a goal, any small deviation from the optimal policy will be corrected by learning, pushing the system back toward optimum. This is akin to a **stable node** in the phase space (all eigenvalues of Jacobian have negative real parts). We can draw analogy to a single-species logistic growth: the system will settle to a fixed point (the optimal performance), with no oscillations. Indeed, for small perturbations, one can show the eigenvalues are real and negative if the game's Hessian (in a continuous strategy formulation) is negative definite at the optimum (i.e. a local maximum of the potential).

In a **zero-sum competitive** scenario, analysis shows a stark contrast. The Nash equilibrium of a zero-sum game is typically a *saddle point* in payoff. In replicator dynamics with *zero* memory decay ($\alpha = 0$) and symmetric learning rates, this equilibrium is *neutrally stable*: small perturbations neither grow nor decay but lead to orbits around the equilibrium (closed orbits conserving a "Hamiltonian" quantity). This is analogous to the center of the predator-prey system (the classical Lotka–Volterra predator-prey equations have a family of closed orbits around a neutrally stable equilibrium). If we add a bit of memory decay ($\alpha > 0$), the equilibrium can become either stable or unstable depending on details, but generally in two-player zero-sum, it tends to be **stable** (damped). Why? Because in zero-sum, the Nash equilibrium is often *unique* and attracts strategies under many learning dynamics when there is some damping. Our model's damping comes from α. For small α, one might get a **spiral** trajectory that converges to the equilibrium (eigenvalues are complex with negative real part, signifying a decaying oscillation). The presence of oscillatory convergence is a signature of a Hopf bifurcation as α crosses from 0 to positive – at $\alpha = 0$ we had a pair of imaginary eigenvalues, for $\alpha > 0$ they become negative real. This is a **supercritical Hopf bifurcation** resulting in the disappearance of the neutral limit cycle and emergence of a stable focus at the equilibrium. If α is too large (meaning the agent forgets extremely fast, effectively chasing a moving target of immediate rewards), it could destabilize learning by making the system too reactive (though in our model α only enters linearly, so unbounded α would just force Q to 0 quickly and the strategy would wander randomly – not exactly an instability but a degenerate behavior). Typically, a moderate α is stabilizing.

Now, what about **mixed-motive games**? These can be the most interesting. We may have multiple Nash equilibria; the learning dynamics could converge to one or oscillate between them or exhibit more complex attractors. A classic example is the **Stag Hunt** (a coordination game with two pure Nash equilibria: one payoff-dominant but risky, one safe but lower payoff). In replicator dynamics, Stag Hunt yields two stable fixed points (the two pure equilibria) and an unstable mixed equilibrium between them (a saddle) – this is analogous to a **saddle-node or transcritical bifurcation** in the payoff parameter space: as payoffs change such that the risk of stag hunt decreases, the unstable equilibrium collides with one stable equilibrium and only the high-payoff one remains stable. In our learning model, similar behavior would occur: if the game has that structure, the learning ODEs will have those two stable attractors. Small initial differences decide which equilibrium the system converges to. From a dynamical systems viewpoint, the presence of multiple attractors means the system is **not globally stable** – initial conditions matter. Bifurcations occur when a parameter (say, a reward value or an external incentive) changes the stability of equilibria. For instance, if we introduce a slight preference for one equilibrium (like a payoff bonus for coordinating on the higher-payoff outcome), at some threshold the basin of attraction for the high payoff equilibrium will suddenly expand, and beyond that threshold the low-payoff equilibrium might become unstable (a tipping point leading to everyone coordinating on the high payoff – analogous to how raising infection rate past a threshold in SIR causes an epidemic to take off).

Another interesting case is **cyclic games** like rock–paper–scissors (RPS), which are not zero-sum but have cycling behavior. RPS replicator dynamics yields a *heteroclinic cycle* in some variants or a neutrally stable orbit in the zero-sum variant. Our model with $\alpha > 0$ would introduce damping in RPS, typically resulting in a unique interior equilibrium that is *asymptotically stable* (damped spiral inward) or *limit cycle* (if there's sustained oscillation due to nonlinearity). There is known research showing that simple learning algorithms in RPS either converge to the mixed equilibrium or cycle around it depending on the algorithm's characteristics – our model could exhibit either, but by adjusting parameters we might see a Hopf bifurcation from convergence to limit cycle. Specifically, if agents learn too aggressively (high β) and with too little memory decay (low α), they might overshoot and create a persistent cycle. If they are more cautious (lower β or sufficient α), they might converge.

Bifurcations: From the above discussion, the main bifurcation parameters in our model could be:

- The memory decay rates α_X, α_Y: $\alpha = 0$ often marks a transition from conservative to dissipative dynamics. In zero-sum games, $\alpha = 0$ gave neutral cycles, $\alpha > 0$ gives decaying cycles (Hopf bifurcation at 0). In other game types, α can affect whether an equilibrium is stable or if oscillations occur.
- The learning sensitivity parameters β_X, β_Y: As these increase, the system becomes more nonlinear and prone to oscillation. There could be a critical β at which a stable equilibrium loses stability and a limit cycle emerges (Hopf bifurcation). For example, in a two-agent game that is a perturbed zero-sum, low β might converge to equilibrium, but beyond a threshold, the fast response causes perpetual cycling.
- The reward/payoff parameters: Changing payoffs can change the game type (e.g. from a coordination game to an anti-coordination game). Bifurcations like saddle-node (when two equilibria collide and annihilate each other), transcritical (equilibria exchanging stability), or pitchfork (symmetric games where an equilibrium splits into multiple as symmetry is broken) can occur in the space of payoff parameters. For instance, continuously turning a game from cooperation towards competition (interpolating between identical rewards vs. opposite rewards) could lead to a bifurcation from a single stable equilibrium (fully cooperative solution) to a regime of oscillations (as the game becomes competitive).
- Number of agents: While not a continuous parameter, increasing the number of agents and their network structure can also induce qualitative changes. In mean-field limits, one can sometimes observe phase transitions as density or connectivity passes thresholds (analogous to percolation or epidemic threshold).

To illustrate one concrete bifurcation: consider a game of *hawk-dove* (a.k.a. chicken, an anti-coordination game) parameterized by a risk-reward parameter. In hawk-dove, there is a mixed Nash equilibrium which is stable under replicator dynamics. If the value of the resource (reward for winning) increases relative to cost of conflict, at some point the mixed equilibrium might destabilize and the system may tend toward an all-hawk or all-dove equilibrium (or vice versa). This could manifest as a pitchfork bifurcation in a symmetric version of the model. Similarly, adding memory decay to a cycling system is a Hopf bifurcation as described.

Spatial and network considerations: If we extend the model spatially (imagine each point in space has a local proportion of strategies), then pattern formation could occur. A uniform equilibrium strategy might become unstable to spatial perturbations if, for example, cooperators and defectors segregate to exploit each other – analogous to Turing patterns in reaction-diffusion systems. That would be an interesting spatial bifurcation: uniform equilibrium loses stability and a heterogeneous pattern (perhaps oscillating in space or forming stationary stripes or spots) emerges. Networks add another layer: the graph topology could cause certain modes of the system (eigenvectors of a network Laplacian, say) to destabilize, leading to clusters of different behaviors.

Delving into those is beyond our scope, but the key insight is that the **mathematical tractability** generally decreases as we move from ODE (few agents or symmetric mean-field) to large network (many differential equations). Mean-field and symmetry assumptions recover tractability via PDEs or reduced ODEs. For the purposes of our analysis, we mostly considered symmetric two-agent interactions or homogeneous populations. These already demonstrate the rich range of dynamics possible (convergence, oscillation, chaos, multiple attractors), tying them to known dynamical phenomena in mathematical biology and physics.

6 Implications and Conclusions

We presented a preliminary mathematical analysis of reinforcement learning in multi-agent systems by marrying RL concepts with dynamical systems modeling. By starting from a rigorous definition of RL through MDPs, Bellman equations, and Q-learning, we established the baseline for how learning dynamics can be captured in equations. We then extended this perspective to multi-agent interactions, identifying that the nature of agent objectives (cooperative vs. competitive) fundamentally influences the qualitative behavior of learning dynamics – much as it does in game theory.

Our proposed ODE model provides a bridge between **RL algorithms** and **continuous-time evolutionary dynamics**. In special cases, it reproduces known results: for example, we showed that in the continuous-time limit, independent reinforcement learners can yield replicator dynamics as a special case. This not only offers theoretical confirmation (as in Sato & Crutchfield's work Sato and Crutchfield [2003]) but also gives us a language to describe multi-agent learning using the rich terminology of dynamical systems (fixed points, cycles, basins of attraction, etc.). The inclusion of learning-specific parameters like memory decay and learning rate allowed us to draw analogies to physical systems with damping, and to identify conditions for stability (e.g. memory decay tends to stabilize competitive learning Sato and Crutchfield [2003]).

Theoretical implications: Our analysis suggests that techniques from nonlinear dynamics – such as Lyapunov functions and bifurcation theory – can be applied to study convergence of multi-agent RL. For instance, in purely cooperative settings one might construct a global Lyapunov function (the team's return) to prove convergence of learning to an optimal joint policy Bellman [1957]. In competitive settings, understanding the oscillatory solutions via Hopf bifurcation theory could inform the design of algorithms that avoid or exploit these oscillations Sato and Crutchfield [2003]. An example is recent research on "conservative" or "cycle-targeting" algorithms in game learning, which deliberately add terms to break cycles and induce convergence; our model provides a testing ground for such ideas, since adding a small perturbation to the equations (like a trembling-hand strategy or exploration) could eliminate unstable cycles and select an equilibrium (similar to how perturbations of a Hamiltonian system can lead to damping).

Moreover, by drawing parallels to ecosystems (predator-prey) and epidemics (SIR), we highlight that multi-agent learning can exhibit *emergent phenomena* like **cycling** (arms races), **collapse** (if learning overshoots, akin to predator extinction if prey are over-exploited), or **phase transitions** (sudden shifts in collective behavior as a parameter crosses a threshold). These analogies are not just metaphorical; they indicate that mathematical tools developed in those domains (e.g. stability of limit cycles in Lotka–Volterra, threshold theorems in epidemics) might be repurposed to analyze multi-agent RL. For example, the concept of an epidemic threshold in SIR (basic reproduction number R_0) might translate to a threshold in influence or learning rate above which a new strategy "infects" the whole population of agents.

Practical implications: For system designers, understanding these dynamics is crucial. If one naively deploys independent Q-learners in a competitive multi-agent environment, the outcome may be cyclic or chaotic, as our analysis (and prior studies) warn Sato and Crutchfield [2003]. This could mean non-convergence in training (policies keep changing) or highly unpredictable behavior. Our study suggests a few remedies, cast in mathematical terms:

- Introduce **damping** or **learning rate schedules** (analogous to increasing α or decreasing β) to ensure stability Sato and Crutchfield [2003]. In practice, this might mean gradually reducing step sizes, or adding regularization to action updates (e.g. entropy bonuses which make policies softer, effectively limiting β).

- Design the reward structure to avoid perverse incentives that create multiple equilibria or cycles. For instance, if a slight modification of rewards can turn a harmful mixed-game into a potential game (with a known Lyapunov function), then learning will be smoother Sato and Crutchfield [2003]. This is related to mechanism design: shaping payoffs so that the game has an efficient equilibrium that is learnable.

- Use **coordination protocols** between agents in cooperative settings to break symmetry and ensure the team converges to the desired equilibrium (avoiding, say, getting stuck in a poorer convention out of many possible). In dynamical terms, this is like selecting the basin of attraction: initial or external guidance can steer the system towards the globally optimal attractor.

Our model also provides insight into **convergence rates**. For example, in cooperative scenarios we might estimate how fast the system approaches the optimum by analyzing the eigenvalues of the Jacobian at the equilibrium Sato and Crutchfield [2003] (related to second derivative of the potential). In oscillatory cases, the frequency of cycles can be derived (in RPS, e.g., the frequency is related to payoff differences). Such quantitative predictions could guide hyperparameter choices (learning rates relative to environment episode length, etc.).

Limitations and future work: Being a preliminary theoretical study, our model made several simplifying assumptions: two agents, discrete actions, and a single-state repeated interaction. Realistic RL scenarios often involve large state spaces and many agents. Extending the analysis to those cases is challenging but conceivable via mean-field theory and numerical bifurcation analysis. Another limitation is the assumption of fully rational update forms (softmax response). Future work could incorporate other learning rules (e.g. ϵ-greedy exploration, actor-critic with function approximators) and investigate if similar dynamical patterns hold. There is also scope to validate these predictions in simulation, bridging theory with empirical RL training runs.

In conclusion, by casting multi-agent reinforcement learning in the language of dynamical systems, we obtain a richer understanding of how individual learning rules scale to complex interactions. We showed that cooperative systems behave like well-behaved gradient flows, competitive systems can cycle akin to predator-prey, and general systems can have multiple attractors reminiscent of bistable populations or epidemics with multiple outcomes. These insights are more than analogies – they are mathematically grounded connections that allow us to apply decades of knowledge from those domains to emerging AI systems. Our work lays a foundation for systematically designing multi-agent RL algorithms that are **stable, efficient, and predictable** by leveraging this cross-disciplinary mathematical framework.

References

Wikipedia. Markov decision process. `https://en.wikipedia.org/wiki/Markov_decision_process`, 2023a.

Wikipedia. Multi-agent reinforcement learning. `https://en.wikipedia.org/wiki/Multi-agent_reinforcement_learning`, 2023b.

GeeksforGeeks. Differences between q-learning and sarsa. `https://www.geeksforgeeks.org/differences-between-q-learning-and-sarsa/`, 2023.

Yuzuru Sato and James P. Crutchfield. Coupled replicator equations for the dynamics of learning in multiagent systems. *Physical Review E*, 67(1):015206, 2003.

Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

Wikipedia. Mean-field game theory. `https://en.wikipedia.org/wiki/Mean-field_game_theory`, 2023c.