# Limiting dynamics for Q-learning with memory one in two-player, two-action games

J. M. Meylahn [123]

July 29, 2021

**Abstract**

We develop a computational method to identify all pure strategy equilibrium points in the strategy space of the two-player, two-action repeated games played by Q-learners with one period memory. In order to approximate the dynamics of these Q-learners, we construct a graph of pure strategy mutual best-responses. We apply this method to the iterated prisoner's dilemma and find that there are *exactly* three absorbing states. By analyzing the graph for various values of the discount factor, we find that, in addition to the absorbing states, limit cycles become possible. We confirm our results using numerical simulations. *Key words and phrases.* Q-learning; multi-agent learning; iterated prisoner's dilemma; absorbing states.

*Acknowledgment.* This work is supported by the Dutch Institute for Emergent Phenomena (DIEP) cluster at the University of Amsterdam.

[1]Dutch Institute of Emergent Phenomena, University of Amsterdam, The Netherlands.

[2]Korteweg-de Vries Institute for Mathematics, University of Amsterdam, The Netherlands.

[3]Informatics Institute, University of Amsterdam, The Netherlands.

*Contact: j.m.meylahn[at]uva.nl

# 1 Introduction

As algorithms are increasingly employed in a variety of settings, more situations will arise in which algorithms interact. In the case of (reinforcement) learning algorithms, what this interaction will lead to is not easily predictable. This is due to the algorithms being designed for single-agent settings (or stationary environments). The theoretical foundations for reinforcement learning algorithms in multi-agent settings are thin, with very few convergence results [4].

This article aims at contributing to these theoretical foundations in the case of two-player, two-action games in two ways: (1) by developing a computational method for determining limiting dynamics of Q-learning algorithms with one period memory and (2) by constructing a representation of deterministic one-period memory Q-learning dynamics. We demonstrate the methods using the iterated prisoner's dilemma (IPD) [13].

The IPD is one of the paradigmatic model for studying the emergence of cooperation in social dilemma's [9], pricing duopolies [15] and other economics games [12]. The model's simplicity allows for rigorous analysis, giving insights that can be heuristically extended to more realistic settings. Two simple extensions to the basic model which make significant steps toward becoming realistic while keeping the model tractable are to include a basic learning mechanism for the players and to give the players a period one memory.

In realistic games, the players of the game have to learn their optimal strategy by learning what the payoffs of the game are and how their opponent is behaving. Games in which both players are learning fall into the field of multi-agent learning (MAL), in which much recent work has focused on dealing with the learning pathologies that typically arise in MAL (see[10] and references therein). For surveys of MAL see [5, 7, 11, 18]

One of the distinctions that can be made in the MAL literature is that between single-state and multi-state settings. A multi-state setting can be thought of as a situation where multiple games are being played, with probabilistic transitions occurring between the games [6]. The most simple multi-state settings is the one where the players have a period one memory. This means that the state the game is in is just the action pair played in the previous round, which both players can observe. This is economically relevant since many of the classical strategies developed for the IPD require a one period memory [1]. The Tit-For-Tat strategy, for example, imitates the action of the opponent in the previous round. This is the setting we will consider in this paper.

Among the most studied reinforcement learning algorithms for MAL is Q-learning [16]. This is an algorithm designed for Markov decision processes for single agents. In multi-agent settings, convergence of Q-learning is typically poor due to the many pathologies that can occur [10]. A promising method for overcoming these pathologies by dealing with the non-stationarity of MAL is to learn in batches. Both agents keep their strategy fixed during the batch in order to collect data in a stationary environment and then simultaneously update their Q-values at the end of the batch. In [3] the authors show that this leads to deterministic dynamics in the infinite batch size limit. The limiting dynamics are a good approximation for large, but finite batch sizes.

In this paper, we develop a computational method for determining the absorbing states of batch Q-learning with period one memory in two-player, two-action iterated games. These games have been classified for Q-learning without memory in [17]. The method allows us to plot what we call a pure mutual best-response (pmBR) graph, which gives the

strategy space dynamics in the infinite batch size limit. We focus on the iterated prisoner's dilemma and show that the absorbing states identified in [14] are the *only* absorbing states. In addition to the absorbing sates, we find that limit cycles are possible in the pmBR graph.

The absorbing states identified in this way for batch Q-learning must necessarily be the same as the absorbing states for *any successful* (in the sense of solving the Bellman equations) Q-learning algorithm that converges to a fixed strategy (for example by having a decreasing exploration rate).

In Section 2 we identify notation, define the model and introduce Q-learning. In Section 3 we describe the methods we develop. In Section 4 and 5 we give the theoretical and numerical results respectively. Finally, in Section 6 we conclude and discuss future research.

# 2 Setting

## 2.1 Model

We consider the same setting as [14] and we will employ broadly similar notation for ease of reference. The setting is that of an IPD played by two Q-learners, labeled $a \in \{1, 2\}$ and using the convention $-a = \{1, 2\} \backslash a$, with one period memory. Each Q-learner has a choice between two actions: defect (D) or cooperate (C). We denote the actions by $\sigma_a \in \{C, D\}$. Due to the one period memory this becomes a multi-state game with the possible states being $\boldsymbol{\sigma} = (\sigma_1, \sigma_2) \in \{(D, D), (D, C), (C, D), (C, C)\}$, where the first component of the vector denotes the action of player one and the second component denotes the action of player two. The payoff (or reward) for player $a$ when the game is in state $\boldsymbol{\sigma}$ is denoted by $r_a(\boldsymbol{\sigma})$ and takes the following form for the IPD:

$$\begin{pmatrix} r_1(1), r_2(1) & r_1(2), r_2(2) \\ r_1(3), r_2(3) & r_1(4), r_2(4) \end{pmatrix} = \begin{pmatrix} p, p & t, s \\ s, t & r, r \end{pmatrix}, \tag{2.1}$$

where we have enumerated the states as $(D, D) \to 1, (D, C) \to 2, (C, D) \to 3$, and $(C, C) \to 4$. In addition, we make the standard assumptions that $t > r > p > s$ and $2r > t + s$. When considering examples, we will use the normalized version where $r = 1$ and $p = 0$.

## 2.2 Learning

Each player chooses their actions according to a strategy $\pi_a$, which in the Q-learning with period one memory setting is the conditional probability of playing action $\sigma_a$ given that the game is in state $\boldsymbol{\sigma}'$, i.e. $\pi_a(\sigma_a|\boldsymbol{\sigma}')$.

The strategies of the players are updated through a reinforcement learning process. We specifically consider Q-learning type algorithms, which are designed to learn the actions that maximize the discounted future rewards, i.e.

$$\sum_{n=1}^{\infty} \delta^n r_a(\boldsymbol{\sigma}(n)), \tag{2.2}$$

with $0 < \delta < 1$ being the discount factor and $n$ denoting the time.

It has been shown that maximizing (2.2) can be achieved by solving

$$Q_{a,\boldsymbol{\sigma}(t),\sigma_a(t)} = \mathbb{E}[r_a(\boldsymbol{\sigma}(t+1))|\boldsymbol{\sigma}(t), \sigma_a(t)] + \delta \mathbb{E}\Big[ \max_{i \in \{C,D\}} \{Q_{a,\boldsymbol{\sigma}(t+1),i}\} |\ \boldsymbol{\sigma}(t), \sigma_a(t) \Big]. \tag{2.3}$$

Note that since the actions taken in the previous round determine the current state, we have used the notation that $\boldsymbol{\sigma}(t) = (\sigma_1(t-1), \sigma_2(t-1))$. The first term on the right-hand side is the expected reward from taking action $\sigma_a(t)$ while in state $\boldsymbol{\sigma}(t)$ and the second term is the discounted expected maximum reward of behaving optimally (according to the current estimate) in the state $\boldsymbol{\sigma}(t+1)$ reached by taking action $\sigma_a(t)$ in state $\boldsymbol{\sigma}(t)$. The Q-function defined by (2.3) is related to Bellman's value function by $V_a(\boldsymbol{\sigma}) = \max_{i\in\{C,D\}}\{Q_{a,\boldsymbol{\sigma},i}\}$.

The Q-learner $a$ solves (2.3) by initializing a Q-matrix

$$\begin{pmatrix} Q_{a,1,D} & Q_{a,1,C} \\ Q_{a,2,D} & Q_{a,2,C} \\ Q_{a,3,D} & Q_{a,3,C} \\ Q_{a,4,D} & Q_{a,4,C} \end{pmatrix} \tag{2.4}$$

and updating the entries according to

$$Q_{a,\boldsymbol{\sigma},\sigma_a}(t+1) = (1-\alpha(t))Q_{a,\boldsymbol{\sigma},\sigma_a}(t) + \alpha(t)\Big[r_a(\boldsymbol{\sigma}(t+1)) + \delta \max_{i\in\{C,D\}}\{Q_{a,\boldsymbol{\sigma}(t+1),i}\}\Big], \tag{2.5}$$

where

$$\alpha(t) = \begin{cases} \alpha & \text{if } (\boldsymbol{\sigma}, \sigma_a) = (\boldsymbol{\sigma}(t), \sigma_a(t)) \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

is the learning rate. Note that $r_a(\boldsymbol{\sigma}(t+1))$ is the reward obtained in period $t$ (i.e. the reward of taking action $\sigma_a(t)$ while in state $\boldsymbol{\sigma}(t)$, which leads to state $\boldsymbol{\sigma}(t+1)$). In stationary environments the iterative process defined by (2.5) has been shown to converge under some general conditions, for example, in [8].

In contrast to the standard multi-state setting, the transitions between states here happen deterministically given the actions of the players. This means that the only stochasticity comes from the action-selection mechanism, i.e. the strategies. Such an action-selection mechanism must take into account the exploration-exploitation trade-off, where exploratory actions are conducive to learning and exploitatory actions maximize the reward. The methods used in Q-learning for this balancing act can be split into two main categories: the $\epsilon$-greedy mechanisms and the softmax or Boltzman mechanisms. We will focus on mechanisms of the first type, as these are more commonly used in practice due to the ease with which hyperparameters of the algorithms are tuned in this case. We further distinguish between $\epsilon$-greedy mechanisms that either have a constant $\epsilon$ or a time dependent $\epsilon(n)$, which decreases in time.

With an $\epsilon$-greedy action-selection mechanism, the player chooses the action which it considers to be maximizing its reward with probability $1-\epsilon(n)$ and chooses an action uniformly at random with probability $\epsilon(n)$.

Q-learning converges to a solution of the Bellman equations in stationary environments. A Q-learner in a multi-agent setting is however facing a non-stationary environment since the strategy of the opponent changes in time. In the case of a decreasing $\epsilon(n)$, both agents will eventually find themselves in an environment that is stationary, since the opponent's strategy converges to a deterministic (pure) strategy.

In order to identify equilibrium points of multi-agent Q-learning, it thus makes sense to study the dynamics of Q-learning type algorithms that are focused on learning pure strategies, since the equilibrium points should be the same. To this end, [14] conceive of a

Q-learning algorithm in which the players alternate between either learning with the use of Q-learning or keeping their strategy fixed. This ensures that the environment is stationary during the learning process for both players. In this way, the players sequentially learn a best-response to a pure strategy.

As an alternative algorithm that has been suggested for multi-agent settings, we consider a batch learning algorithm as in [3], in which both players act according to a fixed strategy for the duration of the batch and then update their strategies simultaneously. In the infinite batch size limit, this leads to both players simultaneously taking a step towards the best-response given the strategy of the opponent. The absorbing states of this algorithm are the same as the absorbing states of the algorithm in [14].

# 3 Method

## 3.1 Self-consistent solutions to the Bellman equations

If we restrict ourselves to pure strategies, each player can pick one of two actions for each of the four states, leading to a total of $2^4$ strategies. As a result, the total number of pure strategy *pairs* is $2^8 = 256$. It is not feasible to repeat the calculations in [14] performed for the 16 symmetric strategy pairs for all possible pairs in order to identify asymmetric equilibria. This process can however be automated using software such as Mathematica.

To this end, we identify each strategy pair with an 8-bit digit. The first four bits encode the strategy of player one, and the last four bits the strategy of player two. A zero in the first entry means that $Q_{1,1,D} > Q_{1,1,C}$, while a one means that $Q_{1,1,D} < Q_{1,1,C}$. The digit $(0,0,0,0,0,0,0,0)$, for example, represents the strategy pair where both players play according to the all defect (All-D) strategy. The digit $(1,0,0,1,1,0,0,1)$ in contrast represents the strategy pair where both players play according to the Win-Stay-Loose-Shift (WSLS) strategy.

Each 8-bit digit leads to a system of sixteen equations which can be solved simultaneously for the sixteen Q-values. The Q-values are then expressed in terms of the model parameters $t, r, p$ and $s$. The 8-bit digit however also encodes an assumption on the inequalities obtaining between the Q-values. By using the Boole function in Mathematica we can check if the expressions for the Q-values satisfy these inequalities given the assumptions on the model parameters. In this way, we can determine whether there is a valid solution to the Bellman equations given the strategy pair for all possible strategy pairs.

This can be done for all two-player, two-action repeated games. To illustrate the computation, we will consider the IPD where both players are playing the all defect strategy. The Bellman equations (2.3) for player 1 can be written in terms of indicator functions

when the considering pure strategies. The equations for firm 1 become

$$\begin{aligned}
Q_{1,1,D} =& \mathbb{1}_{\{Q_{2,1,D}>Q_{2,1,C}\}}(p+\delta Q_{1,1,D}\mathbb{1}_{\{Q_{1,1,D}>Q_{1,1,C}\}}+\delta Q_{1,1,C}\mathbb{1}_{\{Q_{1,1,D}<Q_{1,1,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,1,D}<Q_{2,1,C}\}}(t+\delta Q_{1,2,D}\mathbb{1}_{\{Q_{1,2,D}>Q_{1,2,C}\}}+\delta Q_{1,2,C}\mathbb{1}_{\{Q_{1,2,D}<Q_{1,2,C}\}}) \quad (3.1)\\
Q_{1,1,C} =& \mathbb{1}_{\{Q_{2,1,D}>Q_{2,1,C}\}}(s+\delta Q_{1,3,D}\mathbb{1}_{\{Q_{1,3,D}>Q_{1,3,C}\}}+\delta Q_{1,3,C}\mathbb{1}_{\{Q_{1,3,D}<Q_{1,3,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,1,D}<Q_{2,1,C}\}}(r+\delta Q_{1,4,D}\mathbb{1}_{\{Q_{1,4,D}>Q_{1,4,C}\}}+\delta Q_{1,4,C}\mathbb{1}_{\{Q_{1,4,D}<Q_{1,4,C}\}}) \quad (3.2)\\
Q_{1,2,D} =& \mathbb{1}_{\{Q_{2,2,D}>Q_{2,2,C}\}}(p+\delta Q_{1,1,D}\mathbb{1}_{\{Q_{1,1,D}>Q_{1,1,C}\}}+\delta Q_{1,1,C}\mathbb{1}_{\{Q_{1,1,D}<Q_{1,1,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,2,D}<Q_{2,2,C}\}}(t+\delta Q_{1,2,D}\mathbb{1}_{\{Q_{1,2,D}>Q_{1,2,C}\}}+\delta Q_{1,2,C}\mathbb{1}_{\{Q_{1,2,D}<Q_{1,2,C}\}}) \quad (3.3)\\
Q_{1,2,C} =& \mathbb{1}_{\{Q_{2,2,D}>Q_{2,2,C}\}}(s+\delta Q_{1,3,D}\mathbb{1}_{\{Q_{1,3,D}>Q_{1,3,C}\}}+\delta Q_{1,3,C}\mathbb{1}_{\{Q_{1,3,D}<Q_{1,3,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,2,D}<Q_{2,2,C}\}}(r+\delta Q_{1,4,D}\mathbb{1}_{\{Q_{1,4,D}>Q_{1,4,C}\}}+\delta Q_{1,4,C}\mathbb{1}_{\{Q_{1,4,D}<Q_{1,4,C}\}}) \quad (3.4)\\
Q_{1,3,D} =& \mathbb{1}_{\{Q_{2,3,D}>Q_{2,3,C}\}}(p+\delta Q_{1,1,D}\mathbb{1}_{\{Q_{1,1,D}>Q_{1,1,C}\}}+\delta Q_{1,1,C}\mathbb{1}_{\{Q_{1,1,D}<Q_{1,1,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,3,D}<Q_{2,3,C}\}}(t+\delta Q_{1,2,D}\mathbb{1}_{\{Q_{1,2,D}>Q_{1,2,C}\}}+\delta Q_{1,2,C}\mathbb{1}_{\{Q_{1,2,D}<Q_{1,2,C}\}}) \quad (3.5)\\
Q_{1,3,C} =& \mathbb{1}_{\{Q_{2,3,D}>Q_{2,3,C}\}}(s+\delta Q_{1,3,D}\mathbb{1}_{\{Q_{1,3,D}>Q_{1,3,C}\}}+\delta Q_{1,3,C}\mathbb{1}_{\{Q_{1,3,D}<Q_{1,3,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,3,D}<Q_{2,3,C}\}}(r+\delta Q_{1,4,D}\mathbb{1}_{\{Q_{1,4,D}>Q_{1,4,C}\}}+\delta Q_{1,4,C}\mathbb{1}_{\{Q_{1,4,D}<Q_{1,4,C}\}}) \quad (3.6)\\
Q_{1,4,D} =& \mathbb{1}_{\{Q_{2,4,D}>Q_{2,4,C}\}}(p+\delta Q_{1,1,D}\mathbb{1}_{\{Q_{1,1,D}>Q_{1,1,C}\}}+\delta Q_{1,1,C}\mathbb{1}_{\{Q_{1,1,D}<Q_{1,1,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,4,D}<Q_{2,4,C}\}}(t+\delta Q_{1,2,D}\mathbb{1}_{\{Q_{1,2,D}>Q_{1,2,C}\}}+\delta Q_{1,2,C}\mathbb{1}_{\{Q_{1,2,D}<Q_{1,2,C}\}}) \quad (3.7)\\
Q_{1,4,C} =& \mathbb{1}_{\{Q_{2,4,D}>Q_{2,4,C}\}}(s+\delta Q_{1,3,D}\mathbb{1}_{\{Q_{1,3,D}>Q_{1,3,C}\}}+\delta Q_{1,3,C}\mathbb{1}_{\{Q_{1,3,D}<Q_{1,3,C}\}}) \\
&+\mathbb{1}_{\{Q_{2,4,D}<Q_{2,4,C}\}}(r+\delta Q_{1,4,D}\mathbb{1}_{\{Q_{1,4,D}>Q_{1,4,C}\}}+\delta Q_{1,4,C}\mathbb{1}_{\{Q_{1,4,D}<Q_{1,4,C}\}}). \quad (3.8)
\end{aligned}$$

These are reduced to

$$\begin{gathered}
Q_{1,1,D}=p+\delta Q_{1,1,D},\ Q_{1,1,C}=s+\delta Q_{1,3,D},\ Q_{1,2,D}=p+\delta Q_{1,1,D},\ Q_{1,2,C}=s+\delta Q_{1,3,D},\\
Q_{1,3,D}=p+\delta Q_{1,1,D},\ Q_{1,3,C}=s+\delta Q_{1,3,D},\ Q_{1,4,D}=p+\delta Q_{1,1,D},\ Q_{1,4,C}=s+\delta Q_{1,3,D}
\end{gathered} \quad (3.9)$$

when considering the strategy pair of both players using the All-D strategy. Solving (3.9) leads to

$$Q_{1,1,D}=Q_{1,2,D}=Q_{1,3,D}=Q_{1,4,D}=\frac{p}{1-\delta} \quad (3.10)$$

$$Q_{1,1,C}=Q_{1,2,C}=Q_{1,3,C}=Q_{1,4,C}=s+\frac{\delta p}{1-\delta}. \quad (3.11)$$

Since this is a symmetric strategy pair, the solutions to the Bellman equations for player 2 are the same.

To see if this is a valid solution or not, we check that the inequalities of the assumed strategy pair are satisfiable by the model parameters. In this case, the inequalities require that

$$Q_{a,i,D}>Q_{a,i,C} \quad \text{for all } a\in\{1,2\} \text{ and } i\in\{1,2,3,4\}, \quad (3.12)$$

so that we must have

$$\frac{p}{1-\delta}>s+\frac{\delta p}{1-\delta}, \quad (3.13)$$

or equivalently

$$p(1-\delta)>s(1-\delta). \quad (3.14)$$

This is always satisfied given that we assume that $p>s$.

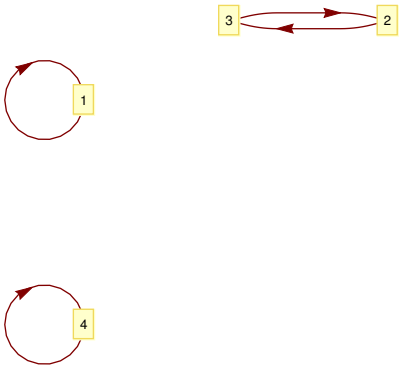


Figure 1: Game state transitions of the symmetric TFT pure strategy pair

### 3.2 Pure mutual best-response

Using the same encoding as before, but now for a single player (i.e. a 4-bit digit), we can calculate the Q-values that are a best-response to the pure strategy encoded by the 4-bit digit. These Q-values can be translated into a 4-bit digit. Given these best-responses we can construct a directed graph between the 256 8-bit digits (nodes) where a directed edge means that the first four bits of a target node contain a best-response of player one against the pure strategy of player two given by the last four bits of the source node and vice versa for the remaining two four bit digits.

We call the dynamics represented by the graph constructed in this way the pure mutual best-response dynamics (pmBR). The solutions to the Bellman equations identified in the previous subsection will appear as absorbing nodes in the pmBR graph.

### 3.3 From strategies to actions

Knowing which strategy pair is being played is still not necessarily informative as to which actions will be taken by the algorithms. As an example, consider the symmetric strategy pair of both players using the Tit-For-Tat strategy. If the system starts in the CC or DD state, it will remain there until one of the players explores and picks D or C, respectively. Then the system will oscillate between the DC and CD states until one of the players again explores to synchronize their action with that of the opponent. This means that the state graph of the game has three disconnected components. Transitions between components occur only due to exploration.

The vector for this state is given by $(0, 1, 0, 1, 0, 0, 1, 1)$ (note that states 2 and 3 require opposite responses from players 1 and 2). The transitions between states of the game when both players play a pure strategy are shown in Figure 1.

We can include the transitions due to exploration as follows. Let the probability that no one explores be $(1-\epsilon)(1-\epsilon) = A$, the probability that player 1 explores be $\epsilon(1-\epsilon) = B$ (this is the same as the probability that player 2 explores) and the probability that both explore be $\epsilon^2 = C$. Then the transition (stochastic) matrix for the game with the symmetric

TFT strategy pair and $\epsilon$-greedy exploration is given by

$$P = \begin{pmatrix} A+B+\frac{C}{4} & \frac{B}{2}+\frac{C}{4} & \frac{B}{2}+\frac{C}{4} & \frac{C}{4} \\ \frac{B}{2}+\frac{C}{4} & \frac{C}{4} & A+B+\frac{C}{4} & \frac{B}{2}+\frac{C}{4} \\ \frac{B}{2}+\frac{C}{4} & A+B+\frac{C}{4} & \frac{C}{4} & \frac{B}{2}+\frac{C}{4} \\ \frac{C}{4} & \frac{B}{2}+\frac{C}{4} & \frac{B}{2}+\frac{C}{4} & A+B+\frac{C}{4} \end{pmatrix}. \tag{3.15}$$

The eigenvector of $P$ with eigenvalue 1 will give the stationary distribution on the game states and therefore tell us how much time is spent in each state. For this matrix the stationary distribution is the uniform distribution on the states, i.e., $(1/4, 1/4, 1/4, 1/4)$. This means that the game spends equal amounts of time at each state.

A similar calculation yields the following stationary distribution for the symmetric GT strategy pair:

$$\mathcal{N}\left(\frac{4-5\epsilon-2\epsilon^2}{\epsilon} \quad 2-\epsilon \quad 2-\epsilon \quad 1\right), \tag{3.16}$$

where $\mathcal{N}$ is a normalizing constant. For $\epsilon = 0.1$ this is $(0.997, 0.054, 0.054, 0.028)$, which shows that even though coordination is possible in the GT symmetric strategy pair it occurs very little (as evidence by the 0.028 probability of being in state 4).

By calculating the stationary distributions of the pure strategy pairs, we can identify which states the game will spend the most time in, giving us an idea of the actions used and the expected payoff.

# 4 Results

## 4.1 Absorbing strategy pairs

By automating the process of solving the Bellman equations and self-consistently checking that the resulting Q-values satisfy the assumptions on the model parameters [1], we find that the symmetric solutions found by [14] are the only possible solutions. This means that there are no asymmetric strategy pairs that solve the Bellman equations in this setting.

The only possible pure strategy solutions pairs are the symmetric all defect (All-D) strategy, Grimm Trigger (GT) strategy and Win-Stay-Loose-Shift (WSLS) strategy (the last strategy is also known as the one-period-punishment strategy or the Pavlov strategy). The All-D strategy is always possible, while there are restrictions on the values of the model parameters for when the other two are possible. This is summarized in Table 1.

| **Strategy** | **Conditions** |
|---|---|
| All D | None |
| GT | $\delta > \frac{t-r}{t-p}$ |
| WSLS | $\delta > \frac{t-r}{r-p}$ |

Table 1: All possible strategies solving the Bellman equations self-consistently.

To illustrate what the conditions on the parameters imply, we plot the phase diagram in the normalized PD (i.e. $r = 1$ and $p = 0$) in Figure 2. As expected, we see that increasing $\delta$

[1] MATHEMATICA code available on request.

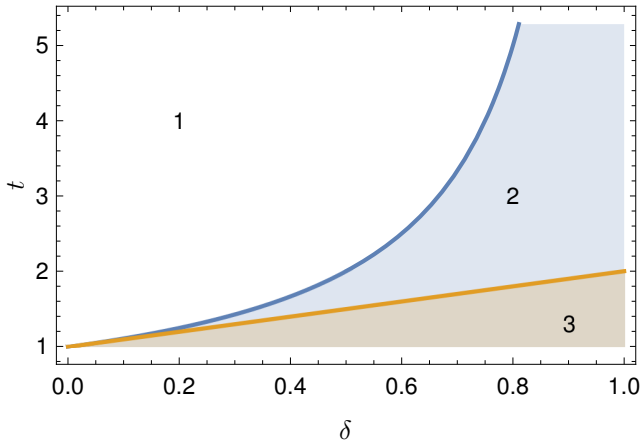


Figure 2: Phase diagram for the normalized prisoner's dilemma. The numbers indicate the number of solutions possible in that region. In the white region only the All-D solution is possible, in the blue region both the All-D and the GT solutions are possible and in the orange region All-D, GT and WSLS are possible.

(the parameter controlling how much the players value future rewards) increases the number of possible solutions. In the limit as $t$ goes to infinity, the line determining the existence of the GT strategy goes to one. This means that for any given $t$, there exists a $\delta$ for which this is a solution. For the line determining the existence of the WSLS strategy there is a critical value of $t$ ($t_c = 2$ in the normalized prisoner's dilemma) above which there is no value of $\delta$ for which the WSLS strategy is a solution.

By applying the methods of Section 3.3 we can identify what action pairs are most likely to be played in the three absorbing states given an exploration rate $\epsilon$ (taken to be 0.1 unless stated otherwise). In both the All-D and GT strategy pair, the DD action pair is played more than 99% of the time. The WSLS strategy pair, in contrast, results in the CC action being played more than 99% of the time. Based on this, the WSLS strategy pair is the only cooperative outcome, with both All-D and GT being effectively competitive outcomes.

### 4.2 Pure best-response dynamics

By the method outlined in Section 3.2, we obtain directed graphs for various values of $\delta$ representing the pmBR dynamics[2]. We do this for the normalized version of the prisoner's dilemma so that $t = 1.5$, $r = 1$, $p = 0$, $s = -0.5$ and for three values of $\delta$, namely $\delta = 0.2$ (Figure 3), $\delta = 0.4$ (Figure 4) and $\delta = 0.5$ (Figure 5). This covers the three regions shown in the phase diagram Figure 2.

In the small $\delta$ case (Figure 3) we find that all nodes lead to the absorbing node 1, which represents the symmetric All-D strategy pair. Interestingly, node 1 is at most two edges away from any other node.

For medium $\delta$ (Figure 4) we see that the symmetric GT strategy pair represented by node 18 becomes a possible solution in addition to the symmetric All-D strategy pair, but that it is an isolated node in the pmBR dynamics. Surprisingly, we find a limit cycle

[2]MATHEMATICA code available on request.

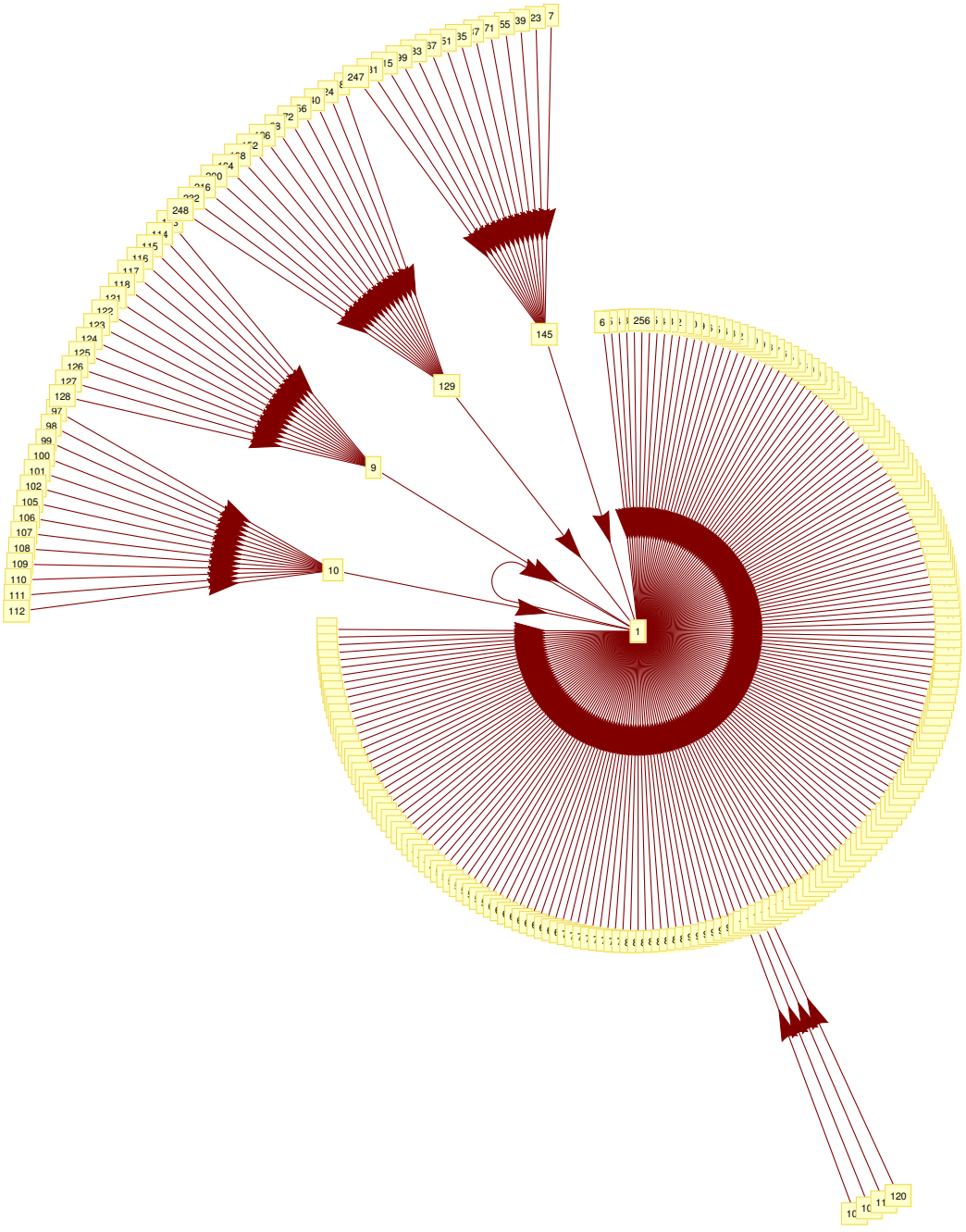


Figure 3: pmBR graph when $t = 1.5$, $r = 1$, $p = 0$, $s = -0.5$ and $\delta = 0.2$.

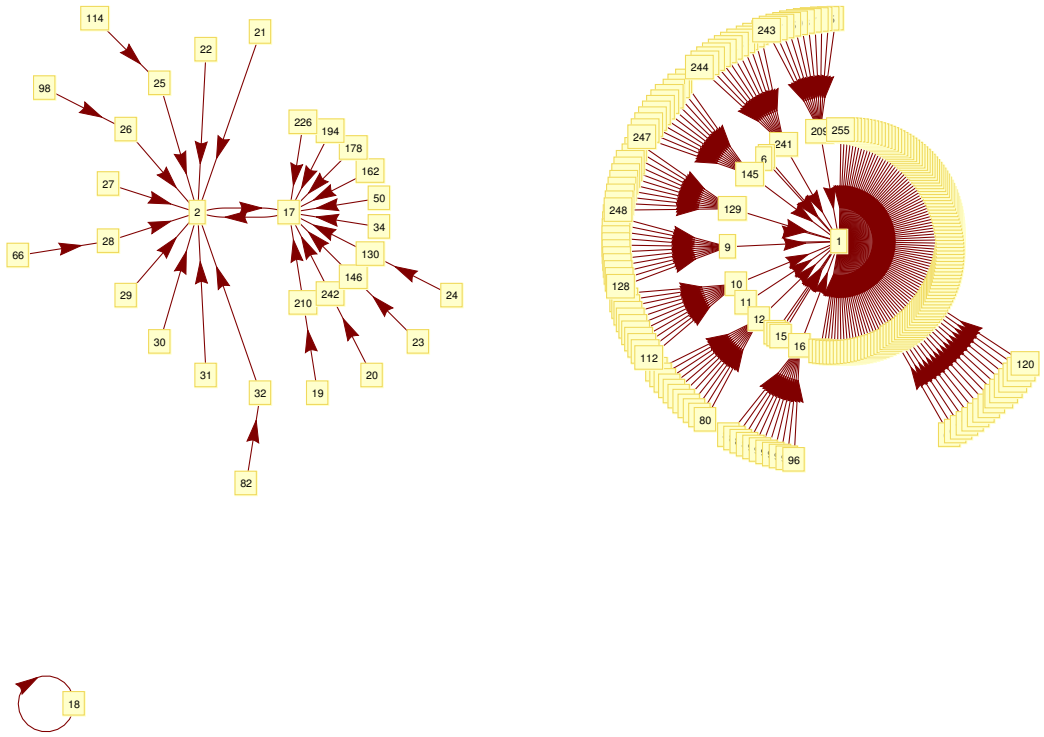


Figure 4: pmBR graph when $t = 1.5$, $r = 1$, $p = 0$, $s = -0.5$ and $\delta = 0.4$.

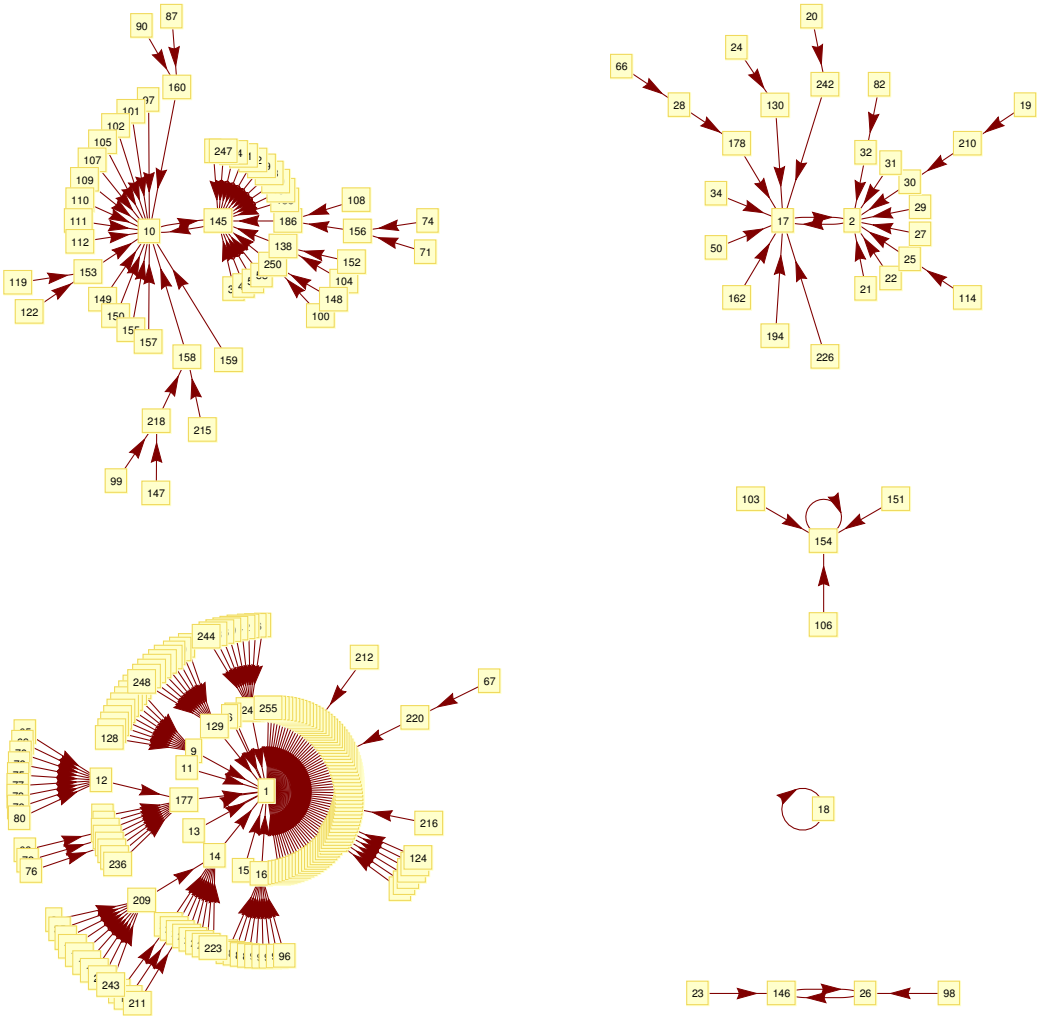


Figure 5: pmBR graph when $t = 1.5$, $r = 1$, $p = 0$, $s = -0.5$ and $\delta = 0.8$.

between nodes 2 and 17. These nodes represent the two strategy pairs where one of the players uses the All-D strategy and the other uses the GT strategy.

For large $\delta$ (Figure 5) we see all three of the possible absorbing strategy pairs with node 154 representing the symmetric WSLS strategy pair. In this case there are three limit cycles: between nodes 2 and 17, nodes 10 and 145 and nodes 26 and 146. The first limit cycle is the same as in the medium $\delta$ case, the second (between nodes 10 and 145) is between the two nodes that represent the strategy pairs where one player is using the All-D strategy and the other is using the WSLS strategy and the last (between nodes 26 and 146) is between the two nodes that represent the strategy pairs where one player is using the GT strategy and the other is using the WSLS strategy.

In addition, we see that the finer structure of the sub-networks changes as the model parameters are changed. The existence of the absorbing states and limit cycles is however robust.

# 5 Numerical results

In this section, we confirm our results by sampling single Batch Q-learning updates for increasing batch size, in order to show that the update converges to a pure strategy best-response as the batch size grows to infinity.

We will make use of the Q-learning variant of the Q-learning algorithm introduced in [2, 3] (specifically Algorithm 1 in [2]). Since we will only be simulating a single batch and are interested in the direction in which the Q-value update following the batch takes the learner, we will set the learning rate $\alpha = 1$ for the update. In order to estimate the Q-value of the next state in the post-batch update, the algorithm needs to estimate the Q-values during the batch. We introduce an auxiliary learning rate $\alpha_{\text{aux}} = 0.2$ for these estimates so that there are two learning rates, one for learning during the batch and one for the post-batch update.

Since the environment is fixed for both players during a batch, we can rely on convergence results from the single-agent Q-learning literature. This is what [2, 3] do to show that the infinite batch size limit gives rise to deterministic dynamics. A condition for this to occur is for the system to be ergodic during the batch, so that each state-action pair is visited infinitely often. In order to satisfy this condition while ensuring that the environment is that of playing against a fixed strategy opponent, we choose a constant (but small) exploration rate. In the case of the experiments of Table 2 we set $\epsilon = 0.1$.

We are interested in how the combination of fixed strategies being played during a batch lead to an update that is a best-response. Since the initial Q-values influence the estimate of the discounted future Q-value, we initialize the Q-values uniformly at random over an interval close to the correct Q-values. This ensures that we sample uniformly from the fixed strategies, but reduce the influence of the magnitude of the initial condition.

In Table 2 we see that the percentage increases with the batch size. A batch size of 16 000 basically guarantees a step in the direction of a best-response. This means that the dynamics of a Batch version of Q-learning, with a learning rate of $\alpha = 1.0$ for the post-batch update and a batch size of 16 000 should be well approximated by the pmBR graphs developed in this paper.

Furthermore, since the standard Q-learning algorithm is a batched algorithm with batch

size one, the goal of an update in Q-learning is to make a small step in the direction of the pmBR. Understanding these idealized dynamics better can thus lead to insight into the true dynamics.

| **Batch Size** | **WSLS** | **GT** | **All-D** |
|---|---|---|---|
| 1000 | 56 | 55 | 77 |
| 2000 | 65 | 74 | 86 |
| 4000 | 74 | 78 | 88 |
| 8000 | 82 | 89 | 94 |
| 16000 | 92 | 94 | 99 |

Table 2: Percentage of best-response updates in a sample of 100 experiments with increasing batch size. Here the best-response is always the same as the fixed strategy of the opponent, i.e., WSLS is a best-response to WSLS etc.

# 6 Conclusion

We have developed a computational method to identify absorbing states for two-player, two-action repeated games played by Q-learners with memory one. For the IPD we have shown that the three symmetric solutions to the Bellman equations identified by [14] (All-D, GT and WSLS) are the only pure strategy pair solutions possible.

Furthermore, we have developed a graphical representation of the dynamics one would expect under simultaneous batch Q-learning algorithms with infinite batch size. The representation shows that there are limit cycles possible in the dynamics. This is an artifact of the Q-value updates taking place simultaneously in Batch Q-learning, as opposed to the sequential Q-value updates in the algorithm proposed in [14].

The absorbing states identified using these methods can be used to establish (rather stringent) learning goals for Q-learning algorithms with memory in these settings. The advantage of using these as learning goals is that the pmBR dynamics could provide a method for establishing convergence results which would not be possible under weaker learning goals.

The methods developed here lend themselves to extensions such as considering two period memory algorithms, considering three action games or placing alternative conditions on the model parameters (such as in the stag hunt, chicken or snow drift games).

## References


[1] Axelrod, R., and Hamilton, W.D. (1981) *The evolution of cooperation*, Science 211 (4489) 1390–1396 .

[2] Barfuss, W. (2020) *Reinforcement learning dynamics in the infinite memory limit* in Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (International Foundation for Autonomous Agents and Multiagent Systems, 2020)

[3] Barfuss, W., Donges, J.F., and Kurths, J. (2019) *Deterministic limit of temporal difference reinforcement learning for stochastic games*, Phys. Rev. E 99, 043305.

[4] Busoniu, L., Babuška, R., and De Schutter, B. (2008). *A comprehensive survey of multiagent reinforcement learning.* IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 38(2), 156–172.

[5] Buşoniu, L., Babuška, R., and De Schutter, B. (2010). *Multi-agent reinforcement learning: An overview.* Innovations in multi-agent systems and applications-1, 183–221.

[6] Hennes, D., Tuyls, K., and Rauterberg, M. (2008). *Formalizing multi-state learning dynamics.* In 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology 2,266–272.

[7] Hernandez-Leal, P., Kartal, B., and Taylor, M. E. (2019). *A survey and critique of multiagent deep reinforcement learning.* Autonomous Agents and Multi-Agent Systems, 33(6), 750–797.

[8] Jaakkola, T., Jordan, M. I., and Singh, S. P. (1994). *On the convergence of stochastic iterative dynamic programming algorithms.* Neural computation, 6(6), 1185–1201.

[9] Macy, M. W., and Flache, A. (2002). *Learning dynamics in social dilemmas.* Proceedings of the National Academy of Sciences, 99(suppl 3), 7229–7236.

[10] Palmer, G. *Independent learning approaches: Overcoming multi-agent learning pathologies in team-games*, PhD Thesis (2020).

[11] Panait, L., and Luke, S. (2005). *Cooperative multi-agent learning: The state of the art.* Autonomous agents and multi-agent systems, 11(3), 387–434.

[12] Rabin, M. (1993). *Incorporating fairness into game theory and economics.* The American economic review, 1281–1302.

[13] Rapoport, A., Chammah, A. M., and Orwant, C. J. (1965). *Prisoner's dilemma: A study in conflict and cooperation* (Vol. 165). University of Michigan press.

[14] Usui, Y. and Ueda, M. (2021) *Symmetric equilibrium of multi-agent reinforcement learning in repeated prisoner's dilemma*, Applied Mathematics and Computation 409, 126370.

[15] Waltman, L., and Kaymak, U. (2008). *Q-learning agents in a Cournot oligopoly model.* Journal of Economic Dynamics and Control, 32(10), 3275–3293.

[16] Watkins, C. J., and Dayan, P. (1992). *Q-learning.* Machine learning, 8(3-4), 279–292.

[17] Wunder, M., Littman, M. L., and Babes, M. (2010, January). *Classes of multiagent Q-learning dynamics with epsilon-greedy exploration.* In ICML.

[18] Zhang, K., Yang, Z., and Başar, T. (2021). *Multi-agent reinforcement learning: A selective overview of theories and algorithms.* Handbook of Reinforcement Learning and Control, 321–384.