The Prisoner's Dilemma

Can a Q-learning agent learn to play?

Irene Ferfoglia
UniTS - Reinforcement Learning exam
18.07.2022

The agenda

- Prisoner’s Dilemma
  - Introduction
  - Formalisation
  - Player’s strategies
- Q-Learning
  - TD learning
  - Policy control in TD(0)
  - The algorithm
  - Convergence
- Results

The Prisoner's Dilemma

Introduction

- Two prisoners with no means of speaking to each other
- Each can either cooperate with the other or give them up to the police
- They don't know each other's action
- No loyalty to each other: objective is to maximise their own reward

Multiple games

If the two players

- play more than once
- remember previous actions of their opponent
- change strategy accordingly

→ Iterated Prisoner's Dilemma

Formalisation

- Adversarial bandits problem
- Two choices of actions: Cooperate (C) or Defect (D)
- Four states: (A’s previous action, B’s previous action)
- Rewards: defined in a payoff matrix

Payoff matrix

A \ B	Cooperate	Defect
Cooperate	(R,R)	(S,T)
Defect	(T,S)	(P,P)

- $T > R > P > S\,, \quad 2R > S + T$
- For each player it is beneficial to defect: Nash equilibrium
- **BUT** highest mutual payoff is with cooperation

Payoff matrix

A \ B	Cooperate	Defect
Cooperate	(3,3)	(0,5)
Defect	(5,0)	(1,1)

- $5 > 3 > 1 > 0\,, \quad 6 > 5$
- For each player it is beneficial to defect: Nash equilibrium
- **BUT** highest mutual payoff is with cooperation

Strategies

- Always cooperate
- Always defect
- Random action
- Tit-for-Tat: copy opponent's previous action

Q-Learning

Temporal-Difference (TD) learning

- Learn from experience, without a model of the environment, its rewards and next-state probability distributions
- Update estimates before final outcome is known (bootstrap)
- Simplest TD method, **TD(0)**:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[R_{t+1} + \gamma V(R_{t+1}) - V(S_t)\right]$$

Policy control in TD(0)

SARSA	Q-Learning
$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[R_{t+1} + \gamma\, \underline{Q(S_{t+1}, A_{t+1})} - Q(S_t, A_t)]$	$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\,[R_{t+1} + \gamma\, \underline{\max_a Q(S_{t+1}, a)} - Q(S_t, A_t)]$
On-policy	Off-policy
Learns near-optimal policy while exploring	Directly learns optimal policy
More conservative	More aggressive

Q-Learning algorithm

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
    Initialize S
    Loop for each step of episode:
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$
    until S is terminal

Requirements for convergence

- Every state-action pair continues to be visited
- Learning rate decreases over time
- Stationary and Markovian environment
- Problem! when vs Q-learner, environment is not stationary

ε-greedy policy

- Simple way to achieve balance of exploration and exploitation
- $0 \leq \varepsilon \leq 1$

$$A_t = \begin{cases} \max_a Q(s, a) & \text{with probability } 1 - \varepsilon \\ \text{random action} & \text{with probability } \varepsilon \end{cases}$$

- Can be decaying by a factor

The choice of α

- TD algorithm convergence criterion:

$$\sum_t \alpha_t = \infty \qquad \sum_t \alpha_t^2 < \infty$$

So α is set to $\dfrac{1}{t+1}$.

Results

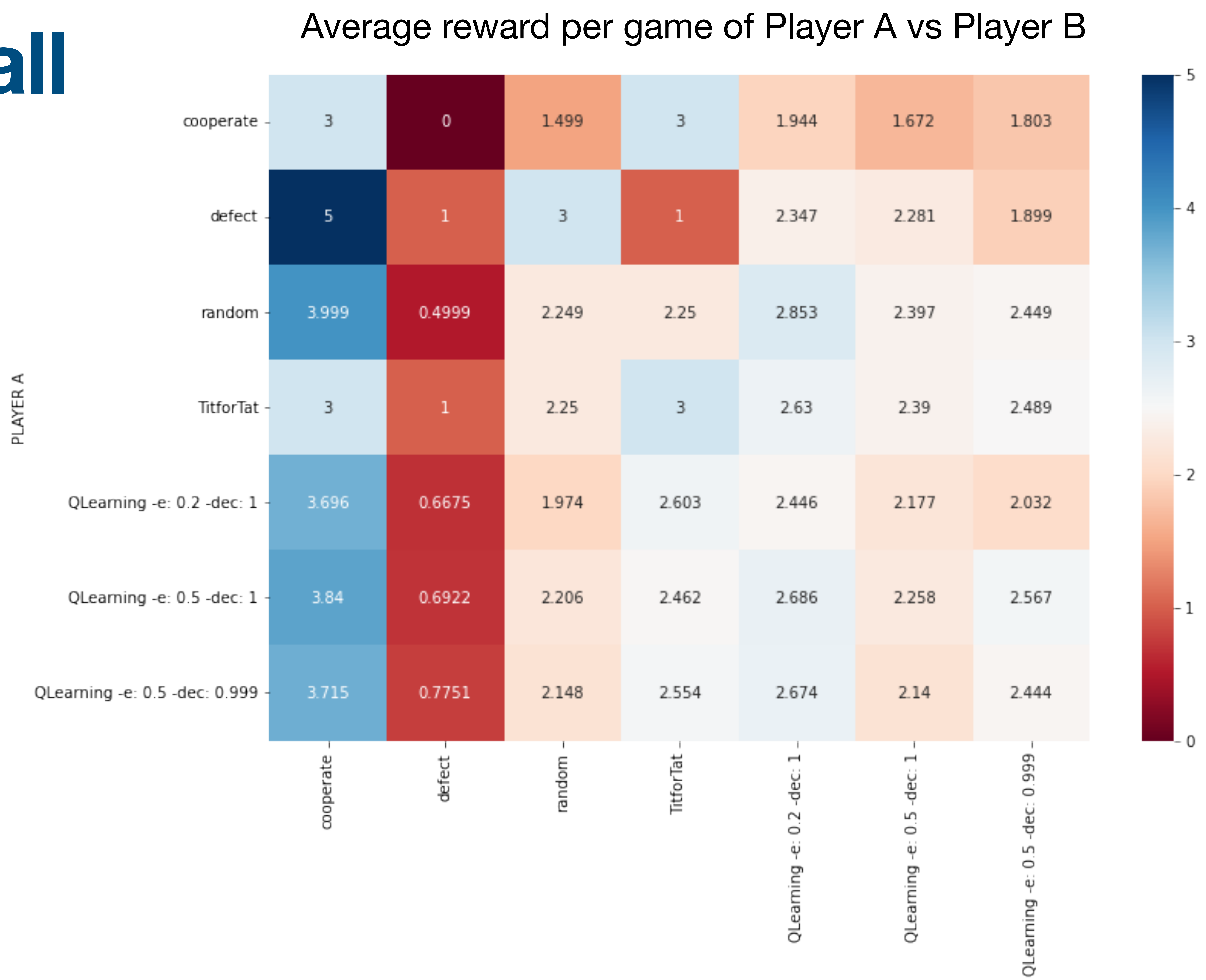
Can a Q-learning agent learn to play:

- vs a fixed strategy?
- vs another Q-learning agent?

Set Up

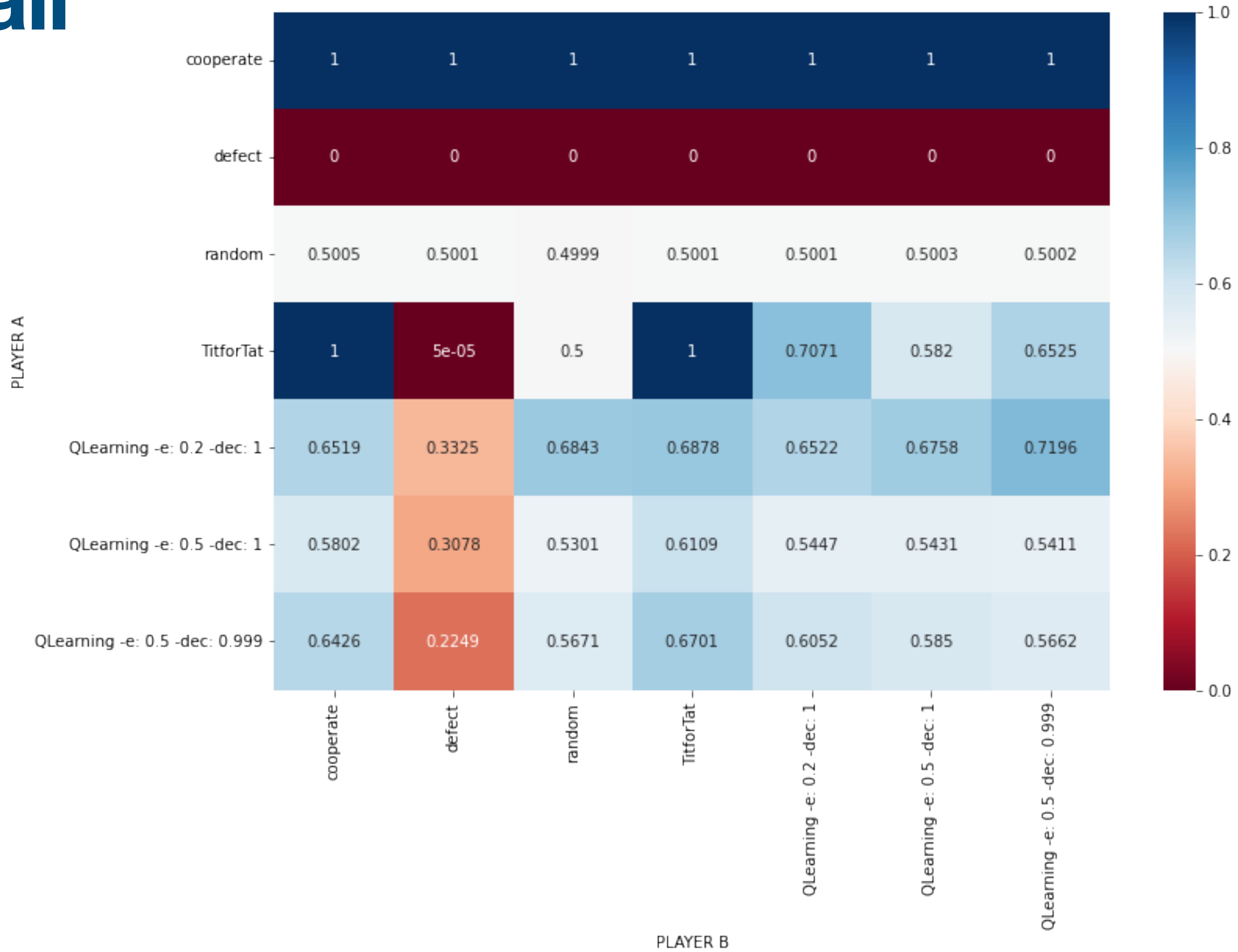
- 100 independent meetings, to avoid random uncertainty
- 10’000 games each meeting
- Q learner with varying epsilons and decays
- Strategies are unknown to one another
- Objective: learn optimal strategy to maximise own avg reward

All vs all

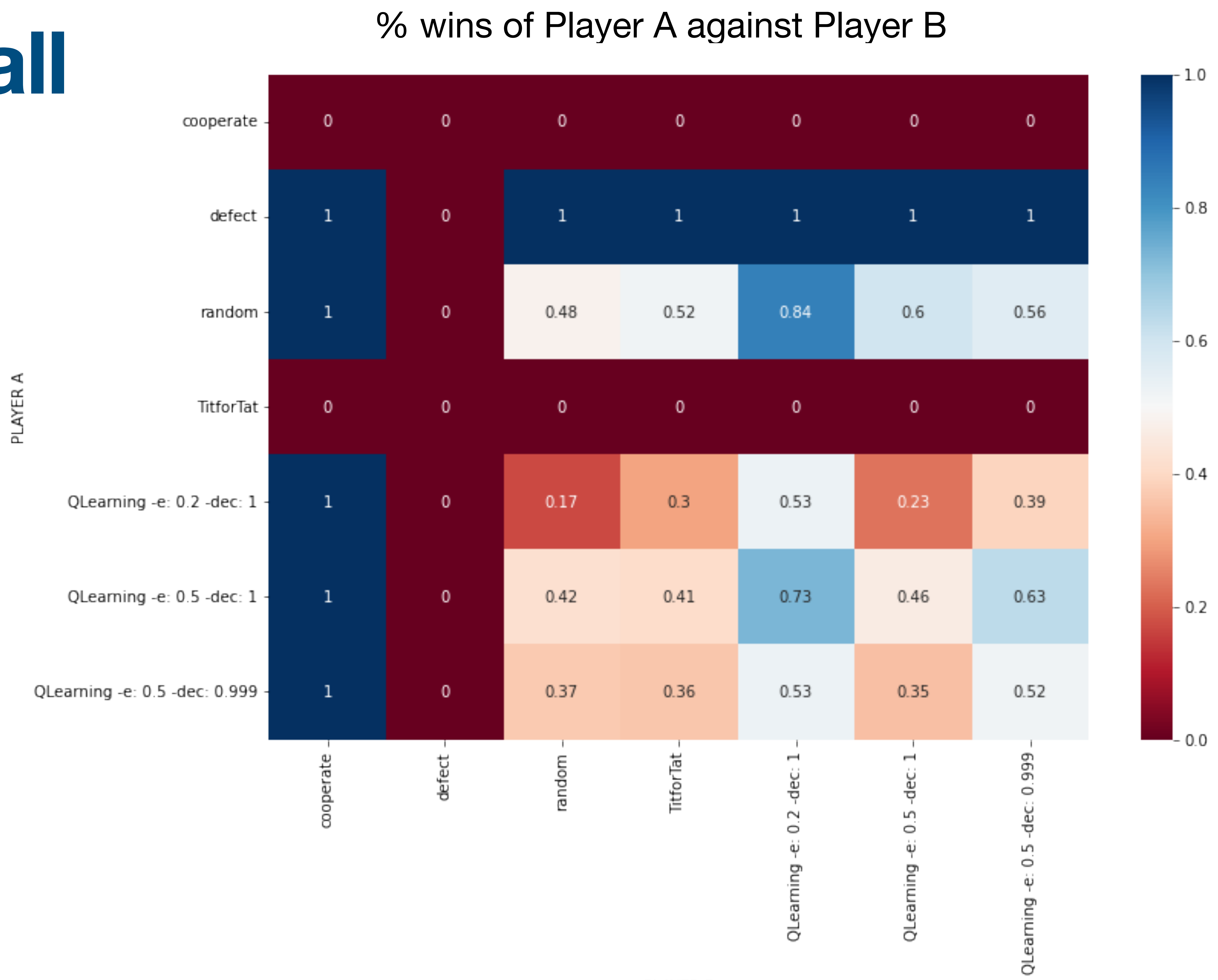

Average reward per game of Player A vs Player B
PLAYER A
PLAYER B
cooperate
defect
random
TitforTat
QLearning -e: 0.2 -dec: 1
QLearning -e: 0.5 -dec: 1
QLearning -e: 0.5 -dec: 0.999
3 0 1.499 3 1.944 1.672 1.803
5 1 3 1 2.347 2.281 1.899
3.999 0.4999 2.249 2.25 2.853 2.397 2.449
3 1 2.25 3 2.63 2.39 2.489
3.696 0.6675 1.974 2.603 2.446 2.177 2.032
3.84 0.6922 2.206 2.462 2.686 2.258 2.567
3.715 0.7751 2.148 2.554 2.674 2.14 2.444
0
1
2
3
4
5

All vs all

Average % of cooperations per game of Player A vs Player B



All vs all


% wins of Player A against Player B
PLAYER A
PLAYER B
cooperate
defect
random
TitforTat
QLearning -e: 0.2 -dec: 1
QLearning -e: 0.5 -dec: 1
QLearning -e: 0.5 -dec: 0.999
cooperate 0 0 0 0 0 0 0
defect 1 0 1 1 1 1 1
random 1 0 0.48 0.52 0.84 0.6 0.56
TitforTat 0 0 0 0 0 0 0
QLearning -e: 0.2 -dec: 1 1 0 0.17 0.3 0.53 0.23 0.39
QLearning -e: 0.5 -dec: 1 1 0 0.42 0.41 0.73 0.46 0.63
QLearning -e: 0.5 -dec: 0.999 1 0 0.37 0.36 0.53 0.35 0.52
1.0
0.8
0.6
0.4
0.2
0.0

Conclusions

Can a Q-learning agent learn to play the Iterated Prisoner's Dilemma…

- Vs a fixed strategy?
- Vs another Q-learning agent?

Kind of

Thank you!