Agent-based Systems

Paolo Turrini

🏠 www.dcs.warwick.ac.uk/~pturrini ✉ p.turrini@warwick.ac.uk

The Plan

- **Logical Agents [Week 1-2]**
    - Knowledge, Preferences, Strategies and how to reason.
- **Decision Theory [Week 3]**
    - Probabilistic Beliefs and Expected Utility.
- **Game Theory [Week 4-5]**
    - Extensive Games and Opponent Modelling.
- **Learning Agents [Week 6-7]**
    - Markov Decision Processes, (Multi-Agent) Learning.
- **Collective Decision-Making [Week 8]**
    - Cooperation and Social Choice
- **Social Agents [Week 9]**
    - Matching, Social Networks.

Multi-agent Learning

In this lecture

We are going to look at Markov Decision Processes with many agents:

- The idea is that agents interact repeatedly and learn each others' strategies
- Connection with game theory
- Revisiting learning as evolutionary process

A great survey:

D. Bloembergen, K. Tuyls, D. Hennes & M. Kaisers
Evolutionary Dynamics of Multi-Agent Learning: A Survey
Journal of Artificial Intelligence Research, 2015.

MAS learning

- We have seen how reinforcement learning informs decision-making via self-play;
- However, we have also seen how it works:
  - with one agent only;
  - with a static environment;

Having multiple interacting agents makes the problem so much more difficult.

Now what?

MAS learning

Imagine a population playing a normal form game repeatedly.

1. Each individual is associated with one strategy for the infinitely repeated game (e.g., always cooperate in a PD).
2. At each time step individuals are paired with each other, randomly.
3. They all play the strategy the are associated with in the beginning.
4. Their payoff determines their reproductive success at the next round.
5. The population changes accordingly, with possibly some mutation.
6. The game is played again.

Mutants

Suppose a population playing strategy s is invaded by mutants, playing s'.

Let $\epsilon \in [0,1]$ be the proportion of such mutants.

The expected payoff of a non-mutant is

$$\epsilon u(s,s') + (1-\epsilon)u(s,s)$$

This is the payoff I get by playing a type times the probability to play them!
and for a mutant? It's the inverse

$$\epsilon u(s',s') + (1-\epsilon)u(s',s)$$

Evolutionary Stable Strategies

Definition

A strategy s is **evolutionarily stable** if for all $s' \neq s$ there exists a $\delta \in [0, 1]$ such that for all $\epsilon \in [0, 1]$ with $\epsilon < \delta$ we have that:

$$\epsilon u(s, s') + (1 - \epsilon)u(s, s) > \epsilon u(s', s') + (1 - \epsilon)u(s', s)$$

What does this definition say?

Notice the "limit" idea...

ESS: the general definition

Let $f(x, y)$ be the expected fitness (=payoff) of strategy x against strategy y. Then x is evolutionarily stable iff, for any mutant strategy y, the following hold:

- $f(x, x) \geqslant f(y, x)$
- if $f(x, x) = f(x, y)$ then $f(x, y) > f(y, y)$

What do these conditions say?

ESS and NE

Observe:

Proposition

Each ESS is a NE.

‘always defect’ in a Prisoner’s Dilemma is evolutionarily stable.

With no invasion, ‘always cooperate’ also is and everyone is so much better off.

But small invasions are problematic...

The evolution of trust

http://ncase.me/trust/

Replicator Dynamics

Now we analyse the dynamics more in general.
We introduce to steps:

Selection Each strategy reproduces, depending on the payoff obtained.

Mutation There is a percentage of invaders, at each round.

These **replicator dynamics** highlight the role of selection, it describes how systems consisting of different strategies change over time. They are usually formalised as a system of differential equations which describe

- the payoffs for each interaction
- the state of the population as the probability distribution of all different types.

Replicator Dynamics

We start with a population $\mathbf{x}$, which represents the probability distribution (= the proportion) of different player types (=strategies).

We are interested in the population change (=the evolution steps). This is written as

$$\dot{x}_i = x_i[f_i(\mathbf{x}) - \overline{f}(\mathbf{x})]$$

where:

- f is a fitness (=payoff) function, applied to a specific player
- $\overline{f}$ and average fitness function for the population as a whole.

Replicator Dynamics

In a two-player game, each player is described by his own evolving population, and at every iteration one individual of each type is selected to play.

This means that the fitness of each type depends on the population distribution of the co-player, i.e., the two populations are co-evolving.

If populations $\mathbf{x}$ and $\mathbf{y}$ have payoff matrices $\mathbf{A}$ and $\mathbf{B}$, we can write the expected fitness of player i of population $\mathbf{x}$ as

$$f_i(\mathbf{x}) = \sum_j a_{ij} y_j = (\mathbf{A}\mathbf{y})_i$$

similarly, the average fitness would be

$$\bar{f}(\mathbf{x}) = \sum_i x_i \sum_j a_{ij} y_j = \mathbf{x}^\top \mathbf{A}\mathbf{y}$$

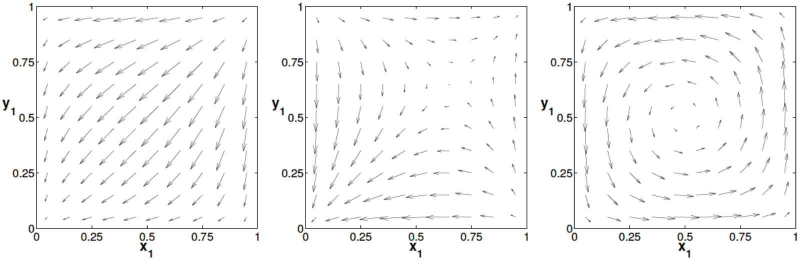
Replicator Dynamics

The general form of population change is given by

$$\dot{x}_i = x_i[(\mathbf{Ay})_\mathbf{i} - \mathbf{x}^\top \mathbf{Ay}]$$

$$\dot{y}_i = y_i[(\mathbf{x}^\top \mathbf{B})_\mathbf{i} - \mathbf{x}^\top \mathbf{By}]$$

Replicator Dynamics



The replicator dynamics, plotted in the unit simplex, for the prisoner's dilemma (left), the stag hunt (center), and matching pennies (right)

MA-RL

We look at a general model of MDPs played by many agents. This means:

- transitions take into account everyone's choices
- we look at how to generalise the one-agent models
- we exlore the connection with repeated games

A very good survey

Bloembergen et al.
Evolutionary Dynamics of Multi-Agent Learning: A Survey
Journal of Artificial Intelligence Research (2015)

General MDPs

Like in a game with start with choices, once per agent.

$$A = A_1 \times A_2 \times \cdots \times A_n$$

Rather than actions, like in an MDP, now we have profiles of actions, like in a game.

Idea: agents play a game, look at what everyone did, and then play again.

Both the transitions and the rewards depend on what everyone does.

$r : S \times A \times S \to \mathbb{R}$ associates a reward to each transition.

$P : S \times A \times S \to [0, 1]$ associates a probability to each transition.

This is very general and encodes games played on graphs. Transitions are defined as triples made by the initial state, the choice profile and the final state.

Notice: Repeated games are just a special case, where we need only one state.

Different learners

There are many ways to model learning in this model. We look at two:

- Independent Learners. They only look at their own action and the reward they get.
- Joint Action Learner. They look at the other agents as well.
- Gradient-based optimisation. In between the two.

Connecting RL with GT

Multi-agent learning and evolutionary game theory share important connections, as they both deal with the strategic adaptation of boundedly rational agents in uncertain environments.

The intuitive connection was made formal with the proof that the continuous time limit of *Cross learning* converges to the replicator dynamics (Börgers & Sarin, 1997)

T. Börgers and R. Sarin
Learning through reinforcement and replicator dynamics.
Journal of Economic Theory, 1997.

Cross Learning

Cross learning is one of the most basic forms of stateless reinforcement learning, which updates policy π, based on the reward r received after taking action j, as follows:

$$\pi(i) \leftarrow \pi(i) + \begin{cases} r - \pi(i)r & \text{if } i = j \\ -\pi(i)r & \text{otherwise} \end{cases}$$

A valid policy is ensured by the update rule as long as the rewards are normalised, i.e., $r \in [0, 1]$.

J. G. Cross
A stochastic learning model of economic behavior.
The Quarterly Journal of Economics, 1973

Cross Learning and policy change

We can estimate the expected change of policy $E(\Delta\pi(i))$ (Börgers & Sarin, 1997).

The probability $\pi(i)$ of action i is affected both if i is selected and if another action j is selected.

Let $E_i[r]$ be the expected reward after taking action i. Then...

$$E(\Delta\pi(i)) = \pi(i)[E_i[r] - \pi(i)E_i[r]] + \sum_{j\neq i} \pi_j[-E_j[r]\pi(i)] = \pi_i[E_i[r] - \sum_j \pi_j E_j[r]]$$

Cross Learning and replicator dynamics

Assuming the learner takes infinitesimally small update steps, the continuous time limit of the previous equation can be rewritten as

$$\pi_{t+\delta}(i) = \pi_t(i) + \delta \Delta \pi_t(i)$$

With $\delta \to 0$ this yields a continuous time system which can be expressed as a partial differential equation

$$\dot{\pi} = \pi(i)[E_i[r] - \sum_j \pi_j E_j(r)]$$

Cross Learning and replicator dynamics

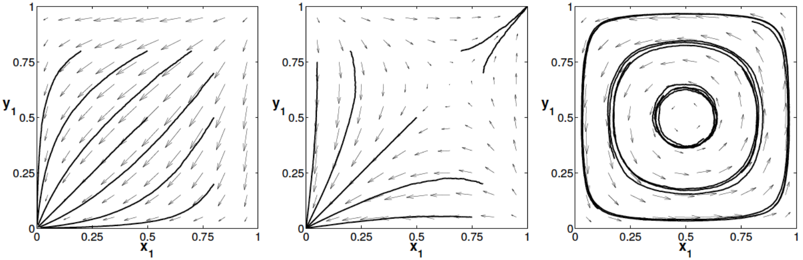
In a two-persons normal form game we can simply write the probability as a mixed strategy. Given the payoff matrices **A** and **B** and policies **x** and **y** for the two players respectively, this yields:

$$\dot{x}_i = x_i[(\mathbf{A}\mathbf{y})_i - \mathbf{x}^\top \mathbf{A}\mathbf{y}]$$

$$\dot{y}_i = y_i[(\mathbf{x}^\top \mathbf{B})_i - \mathbf{x}^\top \mathbf{B}\mathbf{y}]$$

which are exactly the multi-population replicator dynamics.

Learning and replicator dynamics



Policy traces of Cross learning, plotted on the unit simplex and overlaid on the replicator dynamics, for the prisoner's dilemma (left), stag hunt (centre) and matching pennies (right).

Many RL algorithms

Cross learning is a simple method. A number of other RL algorithms have been developed for learning in normal form games

- Q-learning (Tuyls et al. 2003)
- FAQ-learning (Kaisers and Tuyls 2010)
- Regret Minimisation (Klos et al. 2010)
- Lenient FAQ-learning (Panait et al 2008)
- Gradient ascent (Kaisers et al. 2012)

Will they find the "right" strategies? If so, how fast?

Learning in Normal Form Games

Repeated normal form games can be used as a testbed for multi-agent learning. They are stateless and agents choose from a finite set of actions at each time step. This simplifies the analysis heavily.

We focus on two-player two-action games, which simplifies the analysis even more. Here the learning dynamics can be fully represented by the pair $(\dot{x}, \dot{y})$, which denotes the probability of both learners to choose the first action.

Cross Learning in 2x2 games

Let $\mathbf{h} = (1, -1), \mathbf{x} = (x, 1 - x), \mathbf{y} = (y, 1 - y)$.
For Cross learning $(\dot{x}, \dot{y})$ are updated as follows.

$\dot{x} = x[\mathbf{Ay}_1 - \mathbf{x}^\mathbf{T}\mathbf{Ay}] = x(1 - x)[y - a_{11} - a_{12} - a_{21} + a_{22}] = x(1 - x)[y\mathbf{hAh}^\mathbf{T} + a_{12} - a_{22}]$
where a_{12}, a_{22} are elements of the payoff matrix $\mathbf{A}$.

To simplify the notation, we write $\overline{\delta} = \mathbf{Ay}^\top{}_1 - \mathbf{Ay}^\top{}_2 = y\mathbf{hAh}^\mathbf{T} + a_{12} - a_{22}$ to denote the gradient so that Cross learning dynamics can be rewritten as $\dot{x} = x(1 - x)\overline{\delta}$

FAQ dynamics

Frequency-adjusted Q-learning (FAQ) mimics simultaneous action updates by modulating the Q-learning update rule inversely proportional to x_i.

In 2x2 games this simplifies to:

$$\dot{x} = \alpha x(1-x)[\frac{\overline{\delta}}{\tau} - log\frac{x}{1-x}]$$

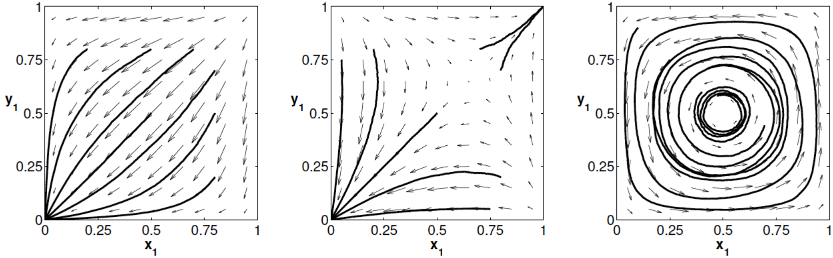
RM

The dynamics of RM are slightly more complex, as the denominator depends on which action gives the highest reward. This can be derived from the gradient: the first action will be maximal if $\overline{\delta} < 0$.

So with RM this simplifies to

$$\dot{x} = \alpha x(1-x)\overline{\delta} \times \begin{cases} (1+\alpha x\overline{\delta})^{-1} & \text{if } \overline{\delta} < 0 \\ (1-\alpha(1-x)\overline{\delta})^{-1} & \textit{otherwise} \end{cases}$$

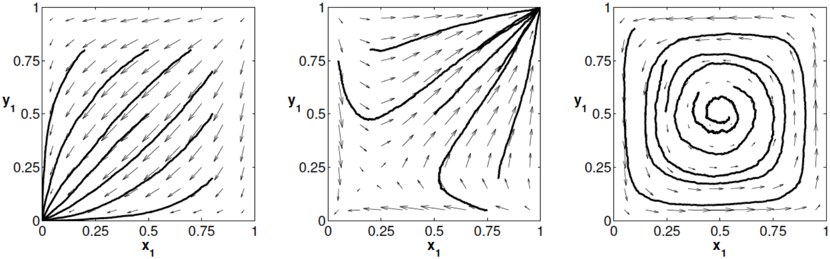
FAQ



Policy learning plotted on the unit simplex for PD, SH, MP

Whereas the dynamics of these different algorithms are similar in their convergence behaviour when only one equilibrium is present, as is the case in the prisoner's dilemma and matching pennies, in the stag hunt differences can be observed.

LFAQ

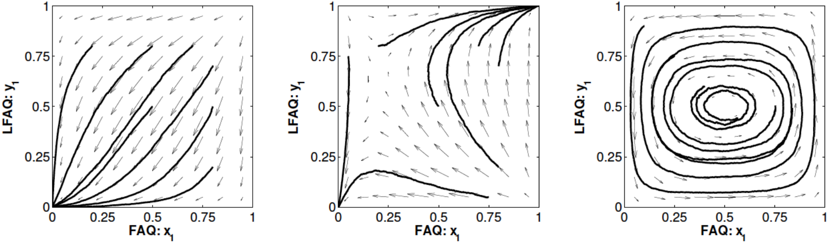


Policy learning plotted on the unit simplex for PD, SH, MP

The notion of leniency, introduced to overcome convergence to suboptimal equilibria, works to drive the learning process towards the optimal outcome of the game

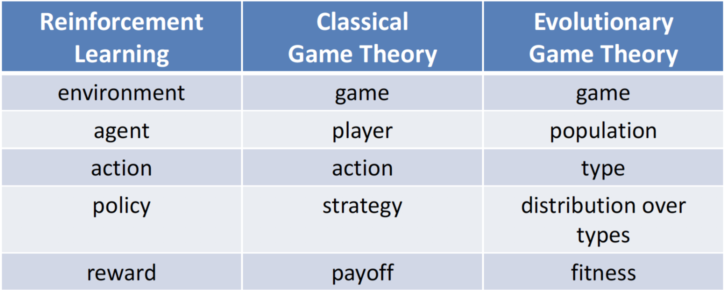
(L)FAQ does spyral inwards towards the single Nash equilibrium at $(1/2, 1/2)$ in MP, which is which is not evolutionarily stable in the classical replicator dynamics model.

FAQ vs LFAQ



Policy learning plotted on the unit simplex for PD, SH, MP and overlaid This represents two players having different learning models.

Conclusion



Reinforcement Learning	Classical Game Theory	Evolutionary Game Theory
environment	game	game
agent	player	population
action	action	type
policy	strategy	distribution over types
reward	payoff	fitness

Learning algorithms converge to the unique equilibria but exhibit a number of differences with multiple equilibria.

We have seen the connection with repeated games and replicator dynamics.

What next? Cooperation