Modelling and simulating the evolution of cooperation

Author(s): Alberto Pérez de Alba Ortíz

Edited by: Valeria Krzhizhanovskaya, Eline Kempkes, Jennifer van Langen, Lex Bolt, Priyank Venkatesh, Sophie Engels.

Important

- **Work in pairs** for this assignment.
- This assignment is significantly larger than the previous. **Start on time**.
- The main point of this course and assignment is the experiments you perform. Design and do something interesting: be creative. The assignment is **not** a complete description nor a recipe for a guaranteed 10.
- Same as in previous assignments, you **must** use the GUI in a logical way and properly visualise your experiments.
- The report and experiments are important. Take them seriously.

Problem statement

Evolution is driven by natural selection, survival of the fittest and *competition* between "selfish genes" to make it to the next generation. However, *cooperation* is common between individuals of the same species—e.g., from colonies to herds—and even across different species—e.g, symbiosis. One big question is: *How does such cooperative behaviour emerge in the midst of competition?* While evolutionary experiments with microorganisms can reach millions of generations in a few years, we would prefer to have some insights before this assignment's deadline. Moreover, we would like to have a generic model for evolution, which is not specific to a particular species. In this assignment, you will create a model to simulate the dynamics of competition/cooperation across generations, based on game theoretic assumptions. You will use this model to, for example, evaluate the population growth of species with different game strategies, evaluate the effect of changes in the game rules on said population dynamics, and evolutionarily design new optimal strategies. We also expect you to be creative and implement other interesting experiments that come to your mind.

The Model

How can one model competition and cooperation dynamics? In their breakthrough papers, Axelrod and Hamilton take a game theoretic perspective to explain *the evolution of cooperation* [1, 2]. The key idea is to model the interaction between organisms as an *iterated* game, which mimics how organisms typically encounter each other more than once in their ecosystems. In this context, each organism, i.e., *player*, can develop a *strategy* based on their previous interactions with another given player. A strategy is then a decision rule based on the interaction history so far. The chosen game to model competition and cooperation dynamics is the iterated Prisoner's Dilemma.

The *one-shot* Prisoner's Dilemma is a well-known thought experiment in which two individuals—with no means of communication—can choose to either *cooperate* for mutual benefit or *defect* for individual benefit [3]. In short, two criminals, A and B, are arrested and imprisoned. The police do not have enough evidence to convict them both for the main charge, but there is enough evidence for a lesser charge, which is punished with 1 year of jail time. The police offer a deal to each prisoner: they can testify against the other prisoner for the main charge and walk free from the minor charge (defect), while the other prisoner will be convicted for 3 years. However, if both prisoners testify against each other, they will both be convicted for 2 years. If both remain silent (cooperate) they both go to jail for 1 year only. The prisoners cannot communicate with each other, and their decision is irrevocable once taken. This yields the next possible scenarios (see Table 1):

1. Neither A nor B testify against each other. They both serve 1 year in prison.
2. A testifies against B but B remains silent. A walks free while B serves 3 years in prison.
3. B testifies against A but A remains silent. B walks free while A serves 3 years in prison.
4. A and B both testify against each other. They both serve 2 years in prison.

The outcomes of the Prisoner's Dilemma—which are often used to model real-world situations in politics, economy, ecology, etc.—are typically summarised in payoff tables. In this case, we use the negative of the number of years in jail, since prison time is presumably unpleasant.

	B remains silent (cooperates)	B testifies (defects)
A remains silent (cooperates)	$P_A = -1; P_B = -1$	$P_A = -3; P_B = 0$
A testifies (defects)	$P_A = 0; P_B = -3$	$P_A = -2; P_B = -2$

Table 1: Prisoner's Dilemma payoff table

In the one-shot Prisoner's Dilemma, defecting is always the best option for either prisoner, regardless of what the other one chooses. For example, if A chooses to cooperate, B can still minimise its own sentence by defecting, and the same applies if A chooses to defect. The same occurs in repeated scenarios when the players know the total number of future reencounters. The best strategy for the last encounter is to defect, and then also for the second-to-last, third-to-last, etc. However, an iterated Prisoner's Dilemma with an *unknown number of reencounters* gives rise to a much richer set of possible strategies.

A strategy defines if a player prefers to initially cooperate or defect, and how their behaviour changes in function of what the other player does. For example, extreme strategies would be to always cooperate, or to always defect. But there are a plethora of options in the middle. For example, one could choose to cooperate initially, and, as soon as the other player defects, then defect for the rest of the encounters, i.e., "hold a grudge". Of course, the model assumes that the strategic decisions are taken simultaneously at discrete time intervals, and that the strategies remain unchanged for the whole game. Axelrod and Hamilton held an iterated Prisoner's Dilemma tournament between 14 strategies submitted by game theorists from economics, sociology, political science, and mathematics. The tournament involved 200 encounters between each pair of players, who did not know the total number of encounters beforehand. The payoff matrix (with a positive reward) is shown in Table 2.

Some of the strategies submitted to Axelrod and Hamilton's tournament were rather complicated, involving even Bayesian inference. However, the winner was an extremely simple strategy, known

	B cooperates	B defects
A cooperates	$P_A = 3; P_B = 3$	$P_A = 0; P_B = 5$
A defects	$P_A = 5; P_B = 0$	$P_A = 1; P_B = 1$

Table 2: Axelrod and Hamilton's tournament payoff table

as *Tit for Tat*, which is based on cooperating in the first encounter and then simply reciprocating whatever the other player did on the previous move. Note that Tit for tat cannot beat another player, it can at best tie. However, in the grand scheme of things, it maximises its score for the whole tournament. Tit for tat is *nice* (cooperates initially), *retaliating* (defects when provoked), but also *forgiving* (does not retaliate indefinitely). These seem to be evolutionarily advantageous qualities in Axelrod and Hamilton's tournament. In this assignment, you will implement your own tournament and submit your own strategies. We will ask you to propose 10 original strategies in total, 5 by each of member of the pair. Can you beat the current champion?

The iterated Prisoner's Dilemma strategies can model the responsiveness of biological entities as simple as bacteria, which are able to chemically affect and be affected by other microorganisms, or as complicated as humans, who use their brains to craft intricate plans. Moreover, the model can even cover interactions between different species, e.g., bacteria in the gut microbiota and humans, both of which have evolved cooperation.

The genetic algorithm

One possible bias of Axelrod and Hamilton's tournament, and also of your own, is that all the strategies are submitted by humans. What if we let natural selection produce new strategies? To this end, we can use *Evolutionary Computation*, a family of algorithms that draws inspiration from darwinian evolution [4]. We will focus on *Genetic Algorithms* [5].

Genetic Algorithms are commonly used in global optimization or search applications. They rely on key evolutionarily inspired concepts, such as genes, selection, crossover, mutation, etc. In short, one performs the following steps:

0. Encoding: Before we begin, we must translate the solutions to our problem, i.e., the strategies, into genetic material, also called genes or chromosomes, that can be easily read, copied, and combined by our algorithm.
1. Initialization: We begin by initialising a population of proposed solutions for the problem at hand. In our case, this would be a population of new strategies.
2. Selection: One proceeds to select the fittest strategies from the population at the current generation. For example, we could introduce these new strategies in a tournament, and determine their fitness based on their final accumulated payoff.
3. Crossover: The genetic information of the fittest strategies is combined to produce a new population, which we expect to inherit the high performance.
4. Mutation: Some random changes are introduced in the new population's genes. This prevents the algorithm from lagging in local minima.
5. Perform Selection on the new population, and continue iterating until convergence.

For this assignment, you have to think about how to implement each of these steps. For example, you can encode strategies as rule tables [6]. In a rule table we can write all possibilities for the previous N moves, and define the next move in function of the interaction history. Here we show Tit for tat's rule table, which only considers one previous move. Note that the first row corresponds to the first move, for which there is no history. Defecting is represented with a D, and cooperating with a C.

Previous move self	Previous move other player	Next move self
N/A	N/A	C
C	C	C
D	D	D
C	D	D
D	C	C

Table 3: Tit for tat rule table

Initialising a population then implies generating many variations of such tables. Keep in mind that more previous moves can be considered, deriving in more combinations and larger tables. Extremely large tables, with a number of previous moves comparable to the length of the game, are **not** advised for this assignment. You should also think about how to rank and select the different strategies. For example, you can have each strategy in your population participate in a tournament against your original 10 strategies (and we recommend also including Tit for tat as an additional 11th strategy for benchmark). Then, assuming a population of 20, you will first generate 20 strategies at random, i,e., 20 random rule tables. Then, you will run 20 tournaments, and each tournament will include one member of the random population of 20, your original 10 strategies, and Tit for tat. The tournament can be played with the payoff matrix in Table 2, or one of your own design. Based on the outcome of these tournaments, you can choose the top 10 or top 5 members of the population of 20 with the highest accumulated payoff for crossover. Presumably, they will do very badly at the beginning, but do not be discouraged. Evolution takes time. Most importantly, you should implement a crossover mechanism in which you mix the rows (i.e., genes) from the best performing rule tables and produce the next generation of strategies. Maybe you decide to choose two parents randomly, and then take the first half of the rules from one parent, and the second half from the other parent. Or perhaps the number of rows passed to the next generation by each parent can depend on their fitness, i.e., accumulated payoff. Additionally, you should implement mutation, e.g., by randomly changing a certain number of rows. Be mindful of not mutating too many strategies, or randomness will take over the optimization process. Then, you can start selection for the new generation. Finally, you should define some criteria, e.g., a measure of convergence, to stop the genetic algorithm.

Tuning the model and the genetic algorithm

In this case you will use your model to simulate a real process in the natural world. The parameters of your model should therefore be realistic and make sense. Adjusting the parameters to reality is called tuning a model and you should perform sensible tuning of your model.

To ease fitting you would do well to organise your parameters in a fitting data structure. A dictionary or a file would be logical options but make sure that your parameters are easily adjustable during the grading process (ideally, through the GUI as well with the defaults set to your experimental parameters). What makes a suitable value for a parameter is dictated by reality: how do species thrive in real life.

Make sure your payoff tables, strategies and genetic algorithms are reasonable. Include proper references to what you based your parameters on. Failing to cite your sources makes your experiments untrustworthy. Keep in mind that it will be unlikely to find sources that tell you exactly what parameters you should use. Most likely, you will find sources that describe indirect aspects, e.g., typical population ratios or interactions in biology. Your job is then to create a model with parameters that embody these circumstances and fit it with the right parameter values such that it approximates 'real life'.

Eventually your model should yield a relatively stable prevalence state. You should note the parameters required to reach this stable state and set them as the default values for your model (i.e., when the GUI is launched). A plot showing that your model settles on this stable state is very useful.

Experiments and Analysis

The most important part of this assignment is the conclusions you can draw. To draw interesting conclusions you need good experiments performed on a solid model. Now that you have a model mimicking reality to a reasonable degree you can perform your experiments.

You are relatively free in the experiments you conduct, but make sure they are reasonable, statistically significant and interesting. You should **at least**:

1. Propose 10 original strategies (rule tables), i.e., each of you needs to come up with 5 novel strategies.
2. Produce plots and analyses about the effect of having more/fewer reencounters, different payoff tables, etc.
3. Evolve 1 new strategy with a genetic algorithm by making it compete with your original 10 strategies.

We recommend first completing points 1 and 2. Once the model is settled, you can proceed to evolve a new strategy. It is **not** a requirement that the evolved strategy outperforms the previous ones. What matters is your work and analyses. We also recommend including Tit for tat as a benchmark in your tournaments.

You can evaluate the performance of each strategy according to several criteria. For example:

- Initial viability: ability to get a foothold in a competitive environment.
- Robustness: thrive in an environment with other players with varied strategies.
- Stability: once established, resist the invasion of new strategies.

Bare Minimal Requirements

Make sure your results contain at the very least two insightful plots that summarise your findings. More plots are welcome but it is also not desirable to produce a hundred plots that contain the same information. You should also write a report about your findings (5-7 pages). In this report, you describe how you built and tuned your model, logically interpret your figures and draw sensible conclusions from them. Keep your report concise (this is up to your own judgement and depends on the scale of your experiments) and make sure you cite all the sources you used.

Assignment

Submit to Canvas at least the following:

1. A PDF file containing your report with the following main headers (see above sections for details on their contents):

   a. Model definition and implementation.

   d. Genetic algorithm setup and implementation.

   c. Tuning the model and the genetic algorithm.

   d. Experiments and analysis.

2. Your implementation. The code ought to be functional and properly documented. Your code must run successfully without alterations and there should be a script to automatically generate your plots. The plotting script can be in a different file. The GUI must be executable, its defaults should be set to the parameters you used in your report.

References

[1] Robert Axelrod. Effective choice in the prisoner's dilemma. *Journal of conflict resolution*, 24:3–25, 1980.

[2] Robert Axelrod and William D Hamilton. The evolution of cooperation. *Science*, 211:1390–1396, 1981.

[3] William Poundstone. *Prisoner's dilemma: John von Neumann, game theory, and the puzzle of the bomb.* Anchor, 1993.

[4] Thomas Bäck, David B Fogel, and Zbigniew Michalewicz. *Handbook of evolutionary computation.* IOP Publishing, 1997.

[5] Melanie Mitchell. *An introduction to genetic algorithms.* MIT Press, 1998.

[6] Adnan Haider. Using genetic algorithms to develop strategies for the prisoners dilemma. *Asian Journal of Information Technology*, 5:866–871, 2005.