Using agent-based simulations to verify when and if the mathematical predictions of Santos, Santos and Pacheco hold true

Contents

1 Abstract

In order to understand our behaviour, our willingness and ability to cooperate with those around us, it is vital that we can create simulations to model the dynamics of populations and the interactions of agents within them. It is also important to understand the way cooperation and defection evolve in both small and large populations. The main research paper on which this project is based is "*Social Norms of Cooperation in Small-Scale Societies*" by Santos, Santos, and Pacheco. In the paper a number of simulation parameters and methods are introduced and those specifications are replicated in this project in order to verify the results of the simulations performed in the paper.

2 Introduction

The primary aim of this project is to replicate the findings of Santos, Santos, and Pacheco which will allow further exploration into more detailed models of the prisoner's dilemma game within populations. The paper outlines simulation parameters based on equations to model the process in which the prisoner's dilemma game is played over time in a population. In order to allow for greater research to be done in this area based on the simulation developed to verify the mathematical model developed by Santos et al. it is necessary that the simulation can be recreated and tested under similar variables and constraints to verify their results.

3 Literature Review

4 Methodology

The application developed was initially programmed in both native Python as well as C++ in an effort to determine the most optimal medium to utilise for further development of the project. The code itself was based on the pseudocode provided by Santos, Santos, Pacheco in the supporting material of their paper. In developing both the C++ and Python programs, the focus initially was on optimization of the program due to the intractability of the algorithm as defined in the paper for large population sizes (Z), large generation numbers (G) and large numbers of runs (R). The total time complexity for any given simulation being $\mathrm{O}(Z^2 \cdot G \cdot R)$. Utilising numpy [3] a number of scalar optimisations were able to increase the efficiency of the program in its Python implementation.
The method of vectorizing the program took was aimed at parallelizing the stage at which one individual interacted with a number of other individuals in order to determine its average fitness over a number of interactions. All optimizations made at any point in the algorithm were tested to ensure complete correctness and verify the algorithm itself had been altered functionally in any way.

4.1 Vectorization of the fitness function

The fitness function, the function to determine the average fitness of an individual over the course of a series of prisoner's dilemma games with other individual in the population, was the aim of the optimization. The vectorization was required to compute a series of vectors in order to computer the payoff for a given agent X and update the reputations of X and a vector of other agents (the tournament vector). The following computations were required:

1. Vector of actions taken by a given agent X against each opposing individual in the tournament vector.
2. Vector of actions taken by each agent in the tournament vector.
3. Vector of the reputation of X after each interaction.
4. Vector of the reputation of each agent after each interaction with X.

The equations defining the actions of both agent X (C_x) and each agent in the tournament vector (C_y) are defined as:
A_i = the vector of size two defining the possible actions of agent i against another agent Y, this vector contains the action if Y has reputation G and if Y has reputation B.
$(A_i)_G$ = the action of agent X if the reputation of some agent Y is G (0 is defection, 1 is cooperation).
$(A_i)_B$ = the action of agent X if the reputation of some agent Y is B (0 is defection, 1 is cooperation).
R = the reputations of the agents defined in the tournament vector (0 is B, 1 is G).
R_x = the reputation of agent X (0 is B, 1 is G).

$$\overrightarrow{C_x} = (A_x)_G \cdot \overrightarrow{R} + (A_x)_B \cdot (1 - \overrightarrow{R})$$
$$\overrightarrow{C_y} = \overrightarrow{(A_y)_G} \cdot \overrightarrow{R_x} + \overrightarrow{(A_y)_B} \cdot (1 - \overrightarrow{R_x})$$

The constraints of this vectorization however are such that for any given tournament vector there must be no duplicates. Due to the parallel nature of the interactions, interactions are performed with incorrect or antiquated knowledge of the reputation of an agent who appears more than once. In order to perform the fitness function correctly with duplicate agents in the tournament sample the algorithm must iterate through each agent. Thus the vectorization requires that a given tournament sample must not contain duplicate agents.
The
The parallelization of reputation update was done utilizing *numpy* features.

4.2 Analysis of optimization

The optimizations made to increase the program's time complexity were measured utilizing *Jupyter Notebook* to measure improvements in execution time for given functions.

5 Results

6 Discussion

7 Extension

The method by which Santos, Santos and Pacheco incorporate private assessment errors of the reputation of any other agent in the simulation involves the use of a single variable χ which determines this error for all agents. The error χ determines the probability that a given agent will wrongly assess the reputation of another agent in a given interaction. The extension on this project aimed to investigate the effect of dynamic assessment errors and the delay of reputation information propagation on rates of cooperation. The process by which this variable assessment error is incorporated into the simulation is outlined in *Methodology (7.1)*

While it is expected that the cooperation index for smaller population sizes will stay relatively constant for a given social norm in comparison to a constant private assessment error, for larger population sizes, this propagation delay may have a greater impact on the rate of cooperation.

7.1 Methodology

The likelihood of wrongly assessing the reputation of a given individual is defined as a function of the time since the last update of that individual's reputation. In turn this process emulates the delay that exists between the creation of new information about an agent's reputation and when any other agent in the population comes into that knowledge. The way that this is done, while maintaining the stochastic framework of the simulation itself, is by utilizing a function for the private assessment error for a given individual i (denoted further as χ_i) and keeping track only of the number of time steps (interactions, denoted as t_i) since agent-i's reputation was *changed*. While we may update an agent's reputation after every interaction, t_i is only reset to 0 when R_i has been changed to a value other than its current value.
The private assessment error the function is defined as:
χ_{min} = the minimum private assessment error.
Z = the population size.
R_s = the rate of information spread, $0 < R_s \leq 1$. R_s of value $\boldsymbol{a}$ signifies that at each interaction every agent with knowledge of agent-i's reputation will give this information to another agent with probability $\boldsymbol{a}$.

$$\chi(t) = 1 - \chi_{min} - \frac{(1 + R_s)^t}{Z}$$
$$\chi_i = \mathbf{max}(\chi(t_i), \chi_{min})$$

8 Conclusion

References

[1] Martin A. Novak, Karl Sigmund, (2005) *Evolution of indirect reciprocity.* Nature.

[2] Fernando P. Santos, Francisco C. Santos, Jorge M. Pacheco, (2016) *Social Norms of Cooperation in Small-Scale Societies.* PLOS Computational Biology.

[3] Numpy Developers, (2016) *Numpy.* `http://www.numpy.org/`.

[4] Continuum Analytics, Inc., (2016) *Anaconda and Anaconda Accelerate.* `https://docs.continuum.io/`.

[5] Hisashi Ohtsuki, Yoh Iwasa, (2005) *The leading eight: Social norms that can maintain cooperation by indirect reciprocity.* Journal of Theoretical Biology.

[6] Olof Leimar, Peter Hammerstein, (2001) *Evolution of cooperation through indirect reciprocity.* Proceedings of the Royal Society B.