An investigation into a Spatial, Iterated Prisoner's Dilemma Simulation

Marius Johannes van Laar (S2852705) & Travis Hammond (S2880024)

February 9, 2021

Abstract

We analyse the nature of agents in a spatial stochastic iterated Prisoner's Dilemma world. In particular the clustering of agents and performance of specific strategies is explored whilst parameters such as the movement rules, death threshold and reproduction threshold are varied. We find the movement mechanism has a significant impact on the results, especially on the structures agents form, whilst the death and reproduction threshold alter the specific composition of the population.

1 Introduction

The Prisoner's dilemma game (PDG) is one of the most commonly used games to explain key concepts and assumptions in Game theory. There is a well defined Nash Equilibrium which justifies selfish behaviour observed throughout nature for survival. Despite this we still see cooperation manifest in many different ways, such as at a structural level underpinning the concept of civilization or group structures. Evolution suggests the drive to pass on ones own genes is the cause of cooperation amongst species with very similar genes. Whilst this kinship is not modelled in PDG, the game can be extended in various ways which leads to behaviours that share characteristics of kinship behaviour.

The first such extension is adding an iterated element, whereby players play multiple rounds, and have a memory of the outcome of the previous rounds. As a result the strategy space increases massively. In a so-called stochastic iterated PDG, a player's strategy can be quantified by a single probability: the probability to cooperate. Hence a player who always defects will have P(C)=0. By adding memory, a player can decide on how to act based on its own or its opponents previous action, or both. Such a game is labelled memory-N, where N is the number of rounds a player remembers. In the case of it remembering only the overall interaction of the most recent round, called memory-1, its strategy is governed by a tuple of four probabilities, (P(C—CC), P(C—CD), P(C—DC), P(C—DD)). P(C—XY) is here defined as the probability that the player will cooperate in the case that in the previous round the player chose action X, and his or her opponent chose action Y. From here it can be understood that the size of the tuple grows as 4^n for memory n. As an example of a memory-1 strategy, a player who always repeats the action of their opponent is defined by the tuple (1, 0, 1, 0). This strategy is called tit-for-tat (TFT), and was the winner of Robert Axelrod's Iterated PDG tournament discussed in [1]. In this tournament, strategies played five games of 200 rounds against each other opponent.

Various lessons can be learned from this tournament. Firstly, it shows simple strategies perform better than complex ones. Specifically, it is observed that complex strategies perform better in fewer situations. An example of this is FELD versus DOWNING. In short, DOWNING estimates the probability of the opponent cooperating after a DOWNING cooperation, and after a DOWNING defection. It then plays the strategy giving it the most long term gain based on those two probabilities. As it plays more rounds, it improves it's estimation of this probability. On the other hand FELD is similar to TFT, but the probability of it cooperating after an opponent does so decreases as the game progress, starting at 100% and going down to 50% by the end of the game. Neither strategies performed well overall, however FELD achieved the highest tournament score against DOWNING. FELDs initial TFT approach led to DOWNING calculating it was best off always cooperating, as this is the most successful strategy against TFT. FELDs evolution then allowed it to defect occasionally, without DOWNING retaliating. This exploitation allowed it to score very highly against reactive maximization approaches such as DOWNING, but did poorly against simple,

	C	D
C	(1,1)	(-3,3)
D	(3,-3)	(-1,-1)

Table 1: The payoff matrix used in this simulation, reproduced from [5].

highly reciprocating strategies such as pure TFT or FRIEDMAN, which cooperates until its opponent defects, after which it will defect for the rest of the game.

Secondly, there are a few traits of strategies which are excellent predictors of final ranking in such tournaments. The first of these is niceness; strategies which cooperate on the first turn consistently finish higher than strategies which defect on the first turn. The second trait is being forgiving. The more forgiving a strategy is, the more defections it will accept before reciprocating (ie defecting too). A nice completely unforgiving strategy is FRIEDMANN, as it will defect following a defection until the end of the game. TFT is unforgiving for one move, as it only considers the last move. A more forgiving example is Tit for two Tats, which reciprocates after two sequential defections by the opponent.

This is one manifestation of the iterated form of the PDG. An alternative is to randomly shuffle opponents every round and play the game for a pre-determined number of rounds. This removes the basis on which many of the non-zero memory strategies in Axelrod's tournament are built, that of repeated interaction with the same opponent. Adding an elimination and reproduction mechanism to this game structure effectively changes the nature of the game from one of a structured football competition, to a temporal evolution of a dynamic population. Any strategies basing their decision on their previous encounter(s) are effectively playing against the population strategy average. The evolution of this population is strictly founded by the reproduction and elimination mechanism introduced in the simulation. In such a game it is possible to follow the evolution and dynamics of different strategies.

The simple iterated PDG described above is a form of compulsory play. In the real world, agents have the possibility to remove themselves from undesirable or unprofitable situations [2]. This can be incorporated into an iterated PDG by adding a spatial aspect, placing all the agents on a grid and introducing simple movement rules. Survivability is now dependent on the local population surrounding the agent, rather than random samples from the population. This has the potential to create pockets of cooperative agents which are stable against low intensity invasions from defecting agents, and potentially create fully cooperative populations [3] [4]. In this report we evaluate the effect of various parameters that arise from implementing a spatial iterated version of the Prisoners Dilemma on the degree of cooperation in the population, as well as the success of particular strategies. We introduce the relevant parameters as part of the simulation methods in section 2. In section 3 we present our results and discuss our findings in section 4. Finally, we conclude our project in section 5.

2 Simulation Methods

In our simulation we let the game play out on a toroidal square grid of length 20. The maximum occupancy of any single point on the grid is 1. All agents are memory 1, meaning their strategy is defined by the tuple (P(C—CC), P(C—CD), P(C—DC), P(C—DD)). The exact strategies counted in our model are set out in table X. The payoff matrix is shown in table 1. The score from the payoff matrix is the main quantity used throughout the simulation and is used, analogously to energy, to measure fitness. It is applied in the movement rules as well as the reproduction and elimination mechanisms. In order to ensure sufficient mixing in the population agents were given an age equal to the number of iterations they have survived. After a minimum number of iterations agents have a finite probability of being removed from the simulation, or equivalently "dying of old age". The exact calculation of this probability is described in section 2.2. Furthermore, in order to prevent overpopulation, a cost energy was added proportional to the total number of agents that is subtracted from all agents each turn.

The core of the program used in this project is the Mesa python library, which describes itself as "a modular framework for building, analyzing and visualizing agent-based models". This allowed for easy implementation of the aforementioned mechanisms, visualization in early simulations as well as running batches of simulations for data acquisition. Its documentation can be found in [6].

To initiate the simulation, 60 agents with starting energy of 1 were placed on an random, empty cell. In order to prevent bias towards any particular strategy in the initial conditions, all starting agents were assigned the strategy (0.5, 0.5, 0.5, 0.5), which means agents choose to cooperate or defect with equal probability. Each iteration consists of shuffling the list of agents, and taking following steps for each agent:

1. Test probability to die of old age; remove agent from grid and scheduler if true
2. Reproduce the agent if it has sufficient energy,
3. Increment age by 1, and reset ΔE to 0,
4. Find all nearest neighbors on adjacent grid points,
5. For each neighbor, play the PDG, add the score to ΔE and update the agent's memory.
6. Calculate the cost of living and subtract it from ΔE and add ΔE to the agent's total energy.
7. Move agent.

Note that neighbors do not benefit or lose from interactions that occur in any turn other than their own.

2.1 Movement mechanism

Three different movement rules were implemented for comparison in the early stages of the project. These are global, local-free and local-probability. Global movement always moved agents to a random free cell. This movement rule is the spatial equivalent of an iterated PDG with compulsory play against random opponents. In this case the population should tend towards an ALLD strategy as this is the Nash Equilibrium, which was also what was observed in early simulations. "Local" rules restricted moves to empty nearest neighbor sites. Agents would move to an empty site if found, or otherwise remain in place. With local-free, agents who had a ΔE of 1 or less would move locally if possible. With local-probability agents have a probability of moving equal to it's ΔE divided by -12, provided $\Delta E < 0$. The worst payoff an agent can get from four interactions is -12, and assuming the cost of living is exactly zero this probability is normalized to the range [0,1]. As the cost of living oscillates around zero, it has no net effect on the probability. Local-probability is a simple implementation to get agents to move away from unprofitable surroundings. Preliminary simulations showed this rule resulted in a much higher average number of neighbors through the appearance of a large cluster of agents, surrounded agents with a higher degree of separation from other agents. The percentage of actions in which the agent cooperated remained stable around 50%, potentially suggesting the cluster consisted of agents with a higher chance to cooperate.

2.2 Reproduction & Mutation mechanism

In order to let the population evolve from a non-biased starting group, it is necessary to add reproduction, elimination and mutation mechanics. Here we set out the reproduction and mutation mechanisms used in our simulation. The reproduction mechanism is simple: if an agent has an energy above threshold parameter p, it places a "child" on the nearest empty grid cell, which is simply a mutated copy of itself with age 0. The parent gives the child half its energy, which is at least p/2.

Each child has 20% chance of getting a mutated strategy relative to its parent. The strategy is mutated by the following algorithm:

- For each P in the strategy 4-tuple:
    1. Generate a random number in the range [-d,d], where d is the hyperparameter mutation amplitude.
    2. Add the random number to P
    3. Normalize P to the range [0,1]

By using a different random number for each P a fair distribution over the full strategy space is achieved.

2.3 Cost of living and elimination.

In order to prevent overpopulation and ensure sufficient turnover of agents throughout the simulation, we introduce some rules which determine when agents are removed from the simulation. There are two main ways for an agent to be eliminated from the simulation.

The first is by old age. Every round agents have a probability of $p = (A - T)/M$ of dying of old age. A is the age of the agent, T is the minimum age before which the probability becomes positive and M is a constant used to define the maximum lifespan $T + M$. Secondly, if an agent's energy drops below a threshold Θ it is removed from the simulation, akin to survival of the fittest. To aid the maintenance of a stable population, a cost of living proportional to the number of agents N was deducted from the energy of every agent every turn. The cost of living C is given by the equation;

$$C = k + 4R(CC)R(DC)N/A, \tag{1}$$

where k is a constant, R is the reward of the interaction identified in the brackets (for our project these values are 1 and 3 respectively if not specified) and A is the area of the grid. Throughout this project the value of k was set to -3, as this gives a cost of living C = 0 for N = 100. Since the cost of living is linearly proportional to the number of agents, agents are given a bonus energy if N is less than 100, raising the number of agents with sufficient energy to reproduce. On the other hand, if there are more than 100 agents, the cost of living suppresses reproduction and raises the death rate. In our simulation M and T were set to 100 and 50 respectively. Θ, the 'death threshold' is a hyperparameter.

Summarizing; the main parameters of interest are the reproduction threshold p, the death threshold Θ and the mutation amplitude d.

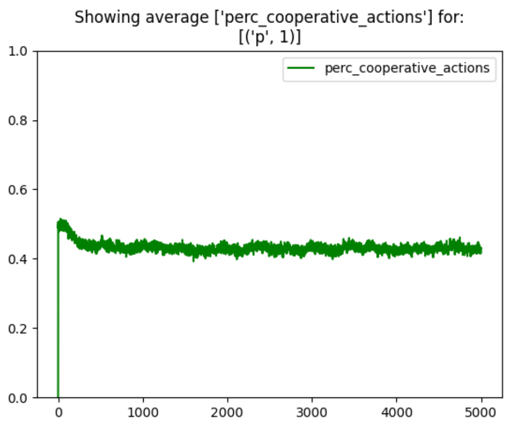
3 Results

In this section we present our results for simulations run with various input parameters. In order to reduce random variations in the results, simulations were repeated fifty times for each parameter setting. The figures presented are the average of each simulation for each time step, unless specified otherwise. To better understand the effects of each parameter, they were varied in isolation. The results are discussed in the context of the percentage of actions that were cooperative in one time step, and the distribution of strategies. Clusters and the distribution of cooperative actions are probed in the context of how many neighbors an agent has. The analysis is performed at a per simulation and per agent level. At the simulation level, the percentage of cooperative actions is compared to the number of neighbors, both averaged per time step. To get a similar picture at the agent level, at each time step of a simulation, the number of cooperative actions an agent took is compared to the number of neighbors the agent had. From the linear fit of this distribution the gradient and intercept is extracted and stored. Here we present the average gradient and intercept across the fifty simulations.

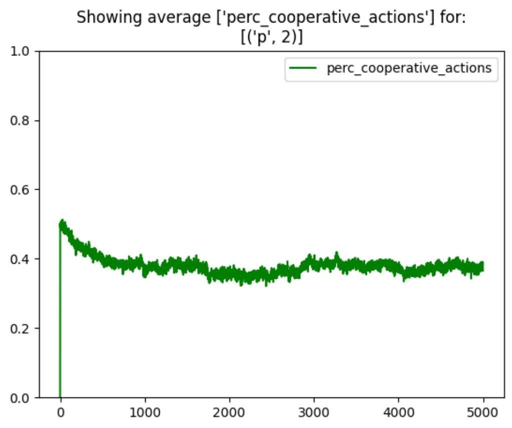
3.1 Reproduction threshold

From figure 1 it is clear that the level of cooperation in the simulation decreases as the reproduction threshold is increased. Considering the payoff matrix in table 1, it is understood that defectors can gain the same amount of energy in much fewer interactions if it meets cooperative agents compared to a cooperative agent. On top of that defective agents are suppressed much less by fellow defectors (loss of 1 unit of energy) than cooperative agents are by defectors (loss of 3 units of energy). Hence it is much easier for defective agents to reach higher reproduction thresholds, whilst the cooperative agents are more likely to drop below the death threshold before they are able to reproduce in such cases. The second observation that can be made from figure 1 is that the number of iterations taken before a steady percentage is reached increases with the production threshold. This a natural consequence of fewer agents reproducing every turn as a result of the higher threshold, meaning fewer children and mutated strategies enter the simulation per turn. As a result it takes longer for the population to properly explore the strategy space.

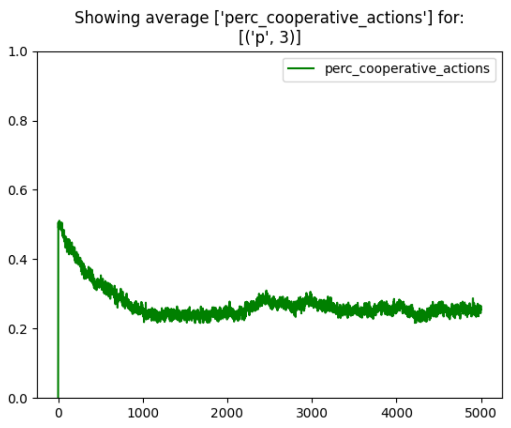
Figure 2 explores the prevalence of various strategies across the four simulations. The main three strategies observed are (0,1,0,1), (0,1,0,0) and (1,1,0,0). Only the last of these has a simple interpretation: it always repeats its last action, regardless of its opponents actions. Since the first action of any new agent is random, agents with this strategy will always repeat that first action. This strategy is mildly successful as all children whose first action is to defect either exploit cooperative agents surrounding it



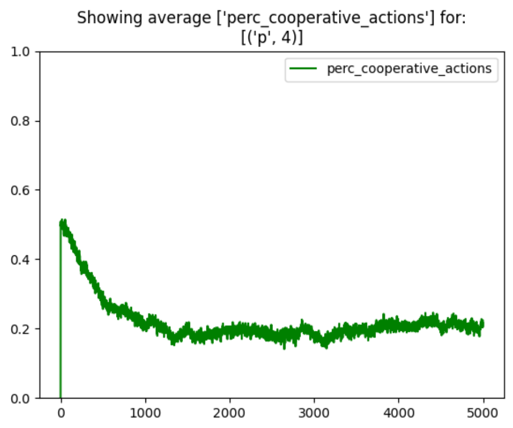
(a) p=1



(b) p=2

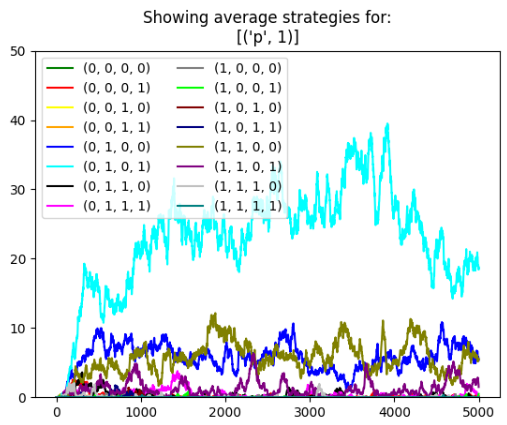


(c) p=3

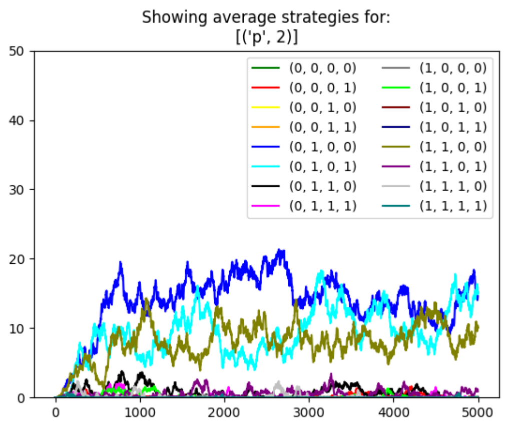


(d) p=4

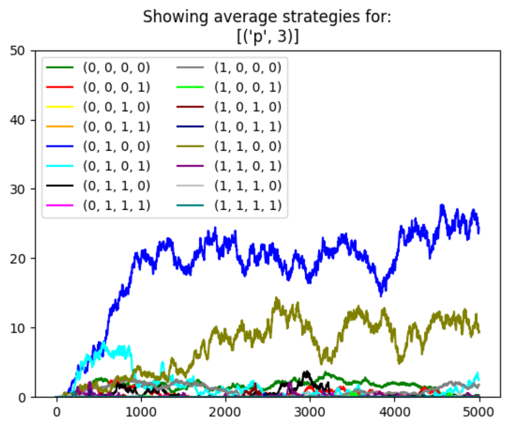
Figure 1: Plots showing the average percentage of actions that were cooperative for four different reproduction thresholds. Across all four plots the percentage initially decreases and then reaches a constant value around which small variations are observed. This constant value decreases as the reproduction parameter increases.



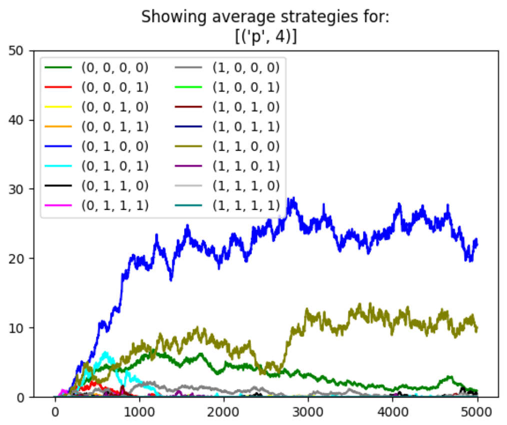
(a) p=1



(b) p=2

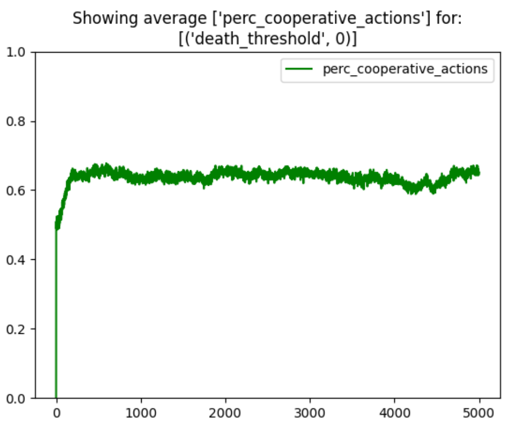


(c) p=3

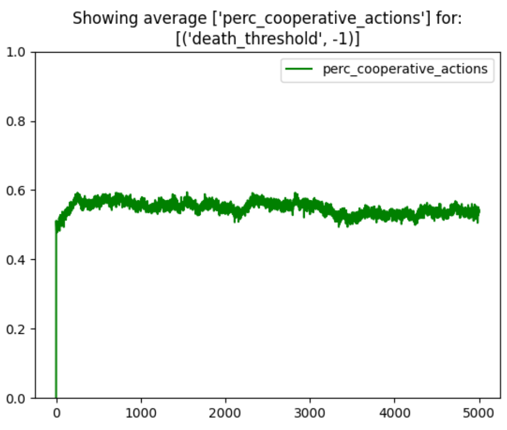


(d) p=4

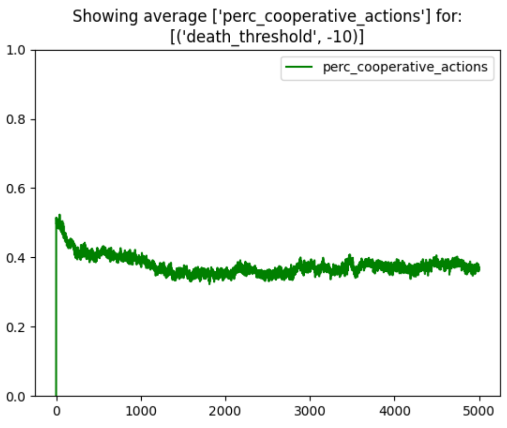
Figure 2: Plots showing the average count of each strategy for four different reproduction thresholds. The initially dominant (0,1,0,1) gets suppressed as the reproduction parameter increases, whilst the similar (0,1,0,0) becomes more dominant along with a consistent (1,1,0,0). In figure d the always defect (0,0,0,0) strategy also makes sizeable fraction of the population.



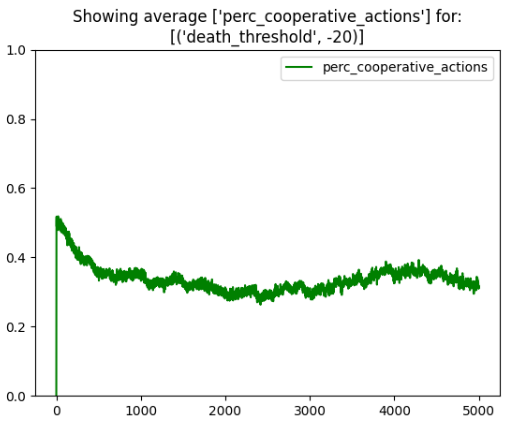
(a) Θ = 0



(b) Θ = -1



(c) Θ = -10



(d) Θ = -20

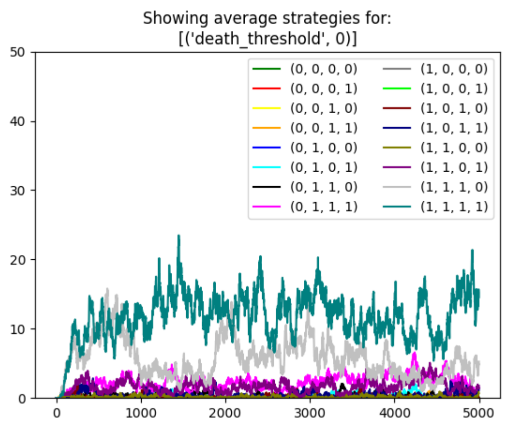
Figure 3: Plots showing the average percentage of actions that were cooperative for four different death thresholds. In all plots the percentage remains steady around some small random oscillations. In figures (a) and (b) the simulations were slightly more cooperative than random, whilst in (c) and (d) agents were in general more defective; just less than 40% of actions were cooperative.

and reproduce themselves again, or mix in with other defective agents. Children who cooperate first will only survive if surrounded by cooperative agents. The main difference between (0,1,0,1) and (0,1,0,0) is their willingness to return to cooperation after mutual defection. (0,1,0,1) will oscillate between mutual cooperation and mutual defection with itself, giving zero average energy change, whilst (0,1,0,0) will keep defecting until the self interaction is disturbed. This loop causes both agents to lose energy every turn, but is not exploitable and also exploits (0,1,0,1)s desire to return to cooperation. This effect is observed in figures 2c and 9b, where initial populations of (0,1,0,1) are killed off by the exploitative (0,1,0,0).

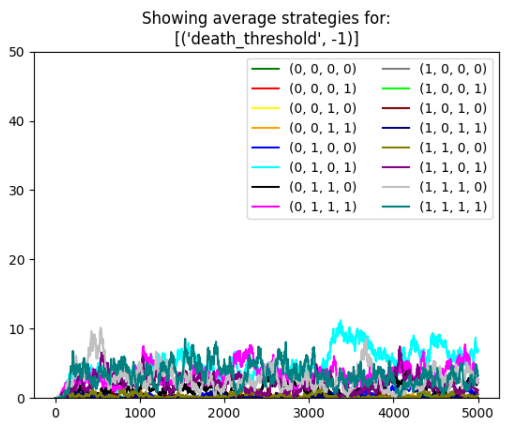
3.2 Death threshold

The death threshold is a parameter that can be understood as a measure of how forgiving the simulation is. Even with local movement there remains an element of randomness in the encounters an agent experiences. Its energy is likely to oscillate as it goes through situations where it may do very well or very badly. By lowering the death threshold an agent is able to survive a rough patch longer, and move further in the hope of finding better neighbors. This works to the benefit of both cooperative and defective agents, however, as mentioned in section 3.1, defective agents will be able to recover faster.

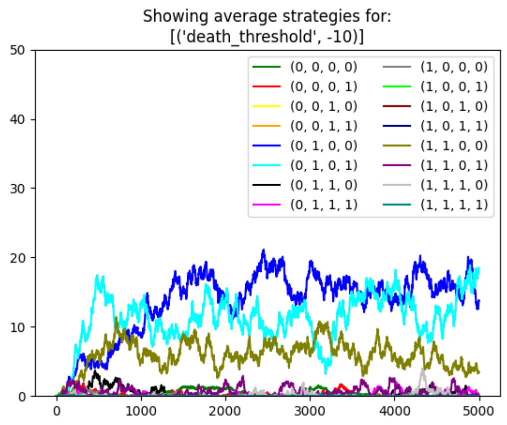
Figure 3 reinforces the workings of this dynamic. For high Θ (figure 3a and 3b) defective agents quickly find themselves below the threshold if they do not have a cooperative agent they are able to exploit near them. Communities of cooperative agents are strong and mobile enough to resist invasions by



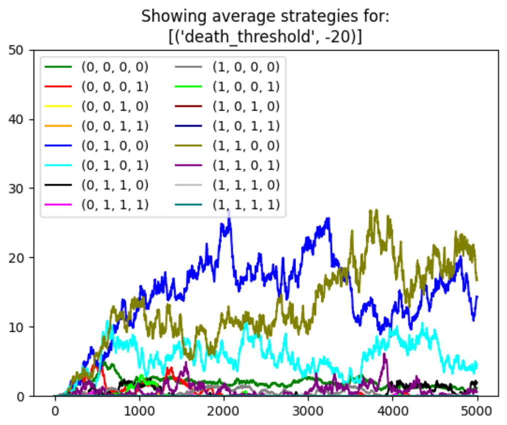
(a) Θ = 0



(b) Θ = -1



(c) Θ = -10



(d) Θ = -20

Figure 4: Graphs showing the average count of each strategy for four different death thresholds. In figure (a) cooperative strategies such as All-C (1,1,1,1) and (1,1,1,0) are most common.. In figure (b) no single strategy is more successful than the other on average, whilst in (c) and (d) we again see strategies like (0,1,0,0), (0,1,0,1) and (1,1,0,0) be the top performers.

defective agents: this is reflected in figure 4a where highly cooperative strategies are most prevalent. As the threshold Θ is lowered the percentage of actions that are cooperative decreases and even at a death threshold of -1 there is no clear advantage for any particular strategy. In figure 4b cooperative strategies boast non-zero count averages, however since the percentage of cooperative actions is only just above 50% it is clear they are not stable and unable to repel invasions by defective agents. For very low values of Θ the same defective strategies as in the simulations with high reproduction thresholds become stable.

3.3 Movement Rules

A key part of any spatial PDG is the rules dictating the movement of agents. With the aim of letting agents move away from undesirable neighbors, two movement rules were tested that move agents only if they experienced a negative change in energy in a turn. The local_prob rule better at allowing agents with both good and bad neighbors to remain in place. This is more realistic as it effectively accounts for realistic factors such as hesitancy arising due to uncertainties about the new position, or a cost of moving (not modelled in our simulation). Since agents move after interacting with their surroundings, the input parameter for both rules is the change in energy of that turn. Since agents always move if their $\Delta E < 1$, we expect more movement with this rule.

In figure 5 we see that at the simulation level, the local_free rule rule leads to a higher percentage of cooperative actions compared to the local_prob rule. This shows that it is largely beneficial for agents to

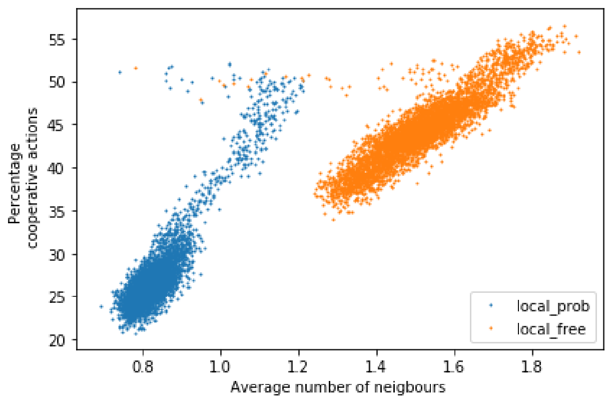


Figure 5: A scatter plot of the percentage cooperative actions against the average number of neighbours for the two different movement rules. It is observed that agents have fewer neighbors on average with the local_prob rule compared to the local_free rule, and are also less cooperative. By analysing the spread of data points, we see the cooperative actions and number of neighbours varies across a bigger range for the local_free rule.

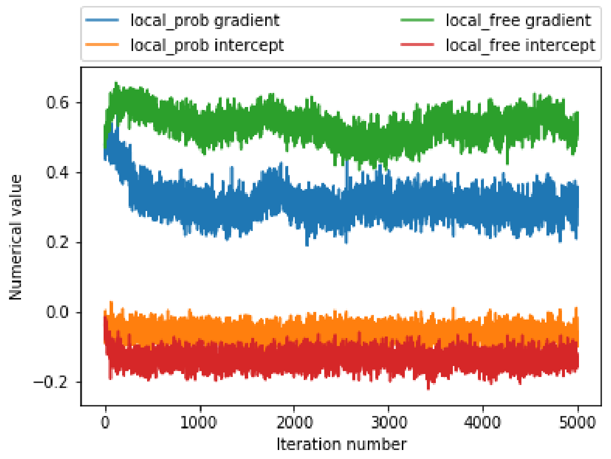
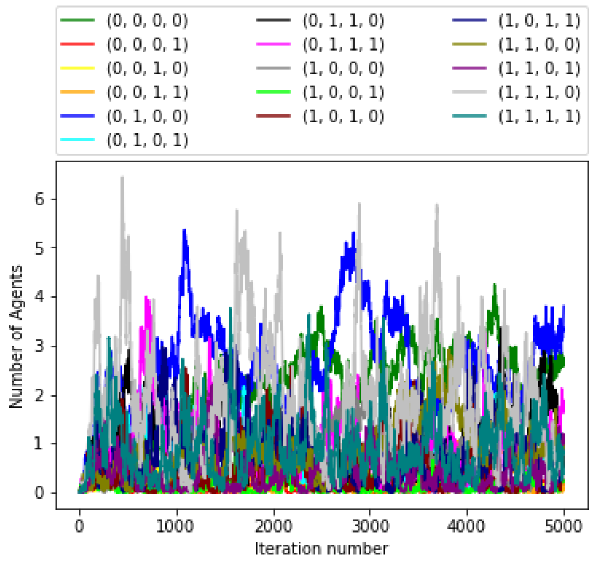


Figure 6: A plot showing the best fit values of a linear fit to the per agent data of its number of cooperative actions and number of neighbors, for the two different movement rules. The gradient is steeper for the local_free rule than the local_prob rule, which has a slightly higher intercept too.

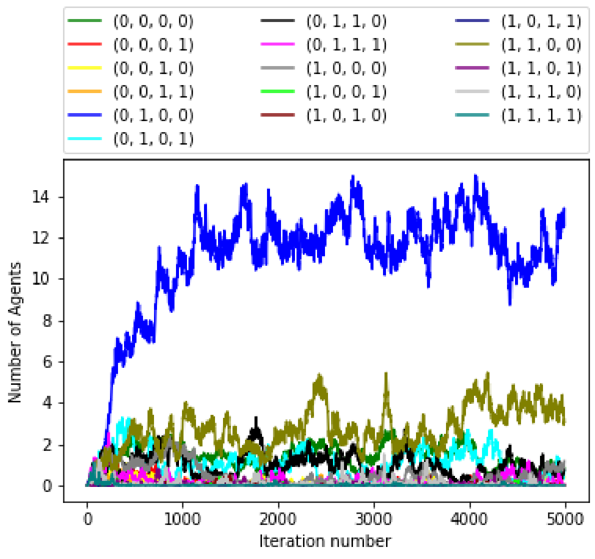
immediately retreat from sites where they experience a net energy loss, which increases the survivability of cooperative agents, and also of cooperative clusters, since we see that the average number of neighbors is consistently higher than for the local_prob simulation. The large spread of the average number of neighbors suggest the shape and compactness of clusters are very dynamic.

Considering the agent level results, in figure 6 we see that the gradient is higher for the local_free simulation, meaning there is a stronger correlation between the number of neighbors an agent has, and the number of cooperative actions the agent takes. Furthermore, the intercept is lower, meaning that agents with fewer neighbors typically have a lower percentage of cooperative actions than agents with more neighbors. To quantify these claims, we take the average gradient and intercept of the second half of the iterations: 0.55 and -0.15 roughly. This means the average number of cooperative actions for an agent is given by 0.55 $\times$ N - 0.15, where N is the number of neighbors. So if an agent has one neighbor, he would be expected to cooperate with the neighbor 0.4 times, or 40% of the time. Equivalently an agent with 3 neighbors will on average cooperate 1.35 times; this is 45% of the time. For the local_prob rule this difference is smaller due to the lower gradient and higher intercept.

In figure 7a a mixture of strongly defective and strongly cooperative are able to coexist in relatively low numbers, this is the source of why the percentage of cooperative actions remains neutral just below



(a) Local_free



(b) Local_prob

Figure 7: Two plots of the average count of all strategies for the two different movement rules. For the local_free rule there is no clear advantageous strategy, whilst for local_prob movement, (0,1,0,0) is again the most successful strategy.

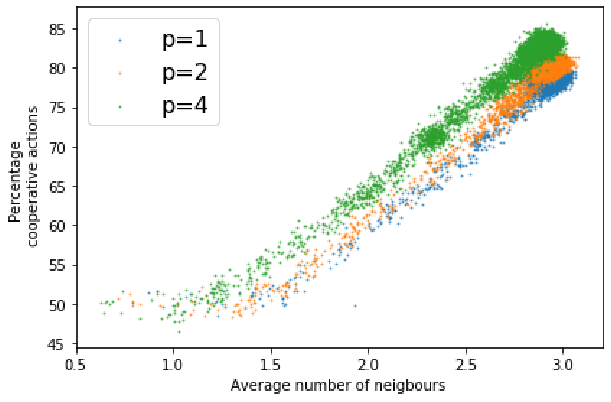


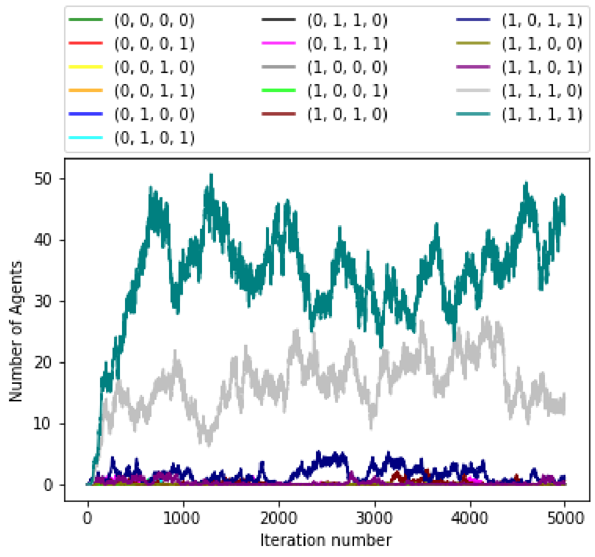
Figure 8: A scatter plot of the percentage cooperative actions against the average number of neighbours for three different reproduction thresholds in a non-zero sum simulation. All three p values have a similar distribution, although increasing the reproduction threshold increases the percentage of cooperative actions across all average number of neighbors.

50% for the local_free simulation. On the other side, in figure 7b (0,1,0,1) is again the most dominant strategy, albeit with a much lower average than seen before.

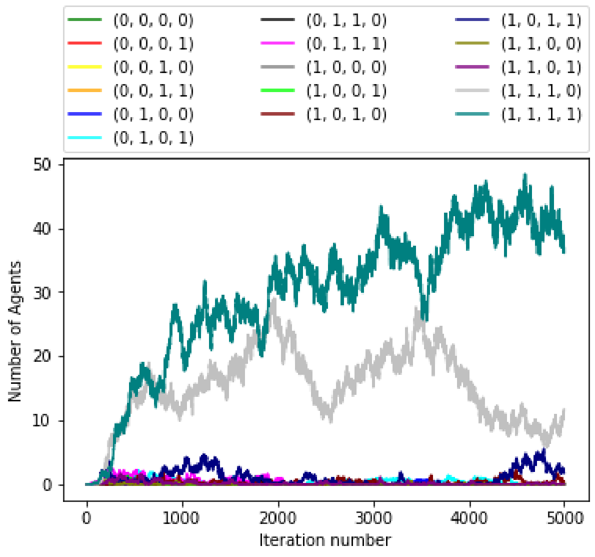
3.4 Positive sum game

As mentioned in section 3.1, the payoff matrix used so far has an inherent advantage to defectors as they have much more to gain and less to lose. In this section we explore the results of simulations run with a higher reward for mutual cooperation: 2, instead of 1. Agents will still experience a net loss of 1 unit of energy for one mutual cooperation and one "betrayal" (CD), however, two mutual co-operations is now able to offset a betrayal and still deliver a net gain of 1. Three batches of simulations were run, with the reproduction threshold varied.

From figure 8 we see the level of cooperation increases as the reproduction threshold increases. This is the opposite behaviour to that seen in section 3.1, where a higher reproduction threshold suppressed the



(a) p=1



(b) p=4

Figure 9: A plot of the strategy counts for a reproduction threshold of 1 (a) and 4 (b). In both simulations All-C (1,1,1,1) and (1,1,1,0) are the most successful strategies, and (1,0,1,1) is the only other non-suppressed strategy that is able to establish itself for a notable amount of iterations.

number of cooperative agents. Moreover, the average number of agents is much higher than that seen in for example 5. This is most likely due to one large cluster of cooperative agents forming, which will not be very dynamic as the overall energy change is likely to be positive, especially in the core. The cluster will contain a growing fraction of the population as other agents stick to the cluster.

One behaviour that does carry over independent of the dominant strategy is that the rate at which the simulation reaches a stable population increases proportionally with the reproduction threshold, as observed in figure 9.

4 Discussion

Due to the payoff matrix, the simulations largely tended to uncooperative worlds as defecting offered agents the best chance of survival. Setting low reproduction thresholds was beneficial to cooperative agents, as it becomes easier for them to create self supporting clusters whilst high densities of defective agents are self-destructive if unable to spread out. Equivalently, low reproduction thresholds in simulations with a higher reward for mutual cooperation increased the presence of defective agents since it was easier for them to reproduce and sustain themselves off clusters with high levels of cooperation. In general, higher reproduction thresholds favoured the stronger agents, acting as a form of natural selection.

Decreasing the death threshold improves the longevity of these dense areas of defective agents, and hence also suppresses the level of cooperation. The main successful strategy emerging from the simulations is (0,1,0,0), which is rather illogical strategy. It is exploitative of opposing cooperative strategies, since it will defect after CC and DC interactions, however it will not retaliate a defection in the case of a CD interaction, but rather continue cooperating in this case. This is counter intuitive since this is a self-detrimental cycle if matched with an exploitative strategy such as itself. In some sense it is greedy, hoping to be forgiven for defecting only to consequently immediately defect again. Most likely any agents getting stuck in a CD cycle died off quickly, and the agents with this strategies that survived long enough were always defecting.

Considering the movement rules, it was initially surprising to see probabilistic movement yield a less cooperative world. However when considering the reward of betrayal and the reproduction threshold, it is likely that an invasion of a cooperative cluster by a defective agent was likely snowballed by the lack of urgency of the cooperative agents to move away from the invader. Considering all simulations displayed in sections 3.1 and 3.2 were run with the local_prob rule, this movement rule likely had a significant effect on the results and acted in favour of defective strategies. When analysing the results of figure 6, there are two main results we must take forward from figure 5. Firstly, the average number of neighbors is lower with the local_prob rule. Secondly, the simulations were relatively more cooperative, yet absolutely still mostly defective, with the local_free rule. With that in mind, the distribution of number of neighbors per agent is likely sharply peaked around 1 and 2 neighbors, resulting in larger variances in the linear fit of the distribution against number of cooperative actions. These variances are in part captured by the fluctuations seen in the graph. More complex rules could include aspects such as vision, where agents are able to take into account the strategies of neighboring agents [7].

Another variable that strongly impacts results is the values of the payoff matrix. We have seen that the mutual cooperation reward being increased without affecting the Nash equilibrium still increases the level of cooperation across simulations. In figure 1 we see a difference of around 20% between the percent of cooperative actions for reproduction thresholds of 1 and 4. For the same reproduction thresholds but with a higher reward for mutual cooperation the difference is only a few percentage points, as observed in figure 8.

It is interesting to note that none of the successful cooperative strategies in ours simulations exhibit the traits found key by Axelrod, despite other spatial iterated PDGs such as [8] doing so.

5 Conclusion

In this project a simulation of a spatial iterated prisoner's dilemma game was developed. By adding a spatial element play becomes non-compulsory and agents are able to move away from unfavourable positions. The specific rules of movement have a significant impact on the outcome of a simulation, and to a lesser extent some parameters introduced to evolve the population of players. The average results of fifty simulations run for 5000 iterations are presented, and show some non-intuitive strategies that emerge as the most stable and successful. Simulations with higher degrees of cooperation typically contain clusters or networks of cooperative agents which are strong and dynamic enough to withstand encounters with aggressive defectors. This phenomenon can be quantified by considering the relation between how many cooperative actions an agent takes on any particular turn to the number of neighbors it has.

6 Declaration

This project had a perhaps unconventional distribution of the workload. Travis wrote 95% of the code, with Marius fixing a few bugs and adding some functions. Travis performed the data analysis to produce the graphs in section 3.1 and 3.2. Marius did the equivalent for the graphs in section 3.3 and 3.4. Marius wrote the entire report.

References

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

[2] E Stanley, Dan Ashlock, and Leigh Tesfatsion. Iterated prisoner's dilemma with choice and refusal of partners. 1993.

[3] Bjørn Lomborg. Nucleus and shield: The evolution of social structure in the iterated prisoner's dilemma. *American Sociological Review*, 61(2):278–307, 1996.

[4] Tomonori Morikawa, John M Orbell, and Audun S Runde. The advantage of being moderately cooperative. *American Political Science Review*, pages 601–611, 1995.

[5] Stephen Majeski, Greg Linden, Corina Linden, and Aaron Spitzer. A spatial iterated prisoners dilemma game simulation with movement. In *Simulating social phenomena*, pages 161–167. Springer, 1997.

[6] Mesa library documentation. `https://mesa.readthedocs.io/en/master/overview.html`. Accessed: 03/11/2020.

[7] Zhen Wang, Chao Yu, Guang-Hai Cui, Ya-Peng Li, and Ming-Chu Li. Evolution of cooperation in spatial iterated prisoner's dilemma games under localized extremal dynamics. *Physica A: Statistical Mechanics and its Applications*, 444:566 – 575, 2016.

[8] Patrick Grim. The greater generosity of the spatialized prisoner's dilemma. *Journal of theoretical Biology*, 173(4):353–359, 1995.