Universität Bayreuth

Sommersemester 2018

Bachelor Thesis

Lehrstuhl: VWL IV - Mikroökonomie

Betreuerin: Eleni Milona

Reinforcement learning and the provision of public goods

An evaluation of a reinforcement learning model in large public goods games

Vorgelegt von: Gideon Stein

MtrklNr: 1360095

Studiengang: P&E

8. Fachsemester

Email: gideon-stein@gmx.de

Abstract:

Public goods games are one of the most discussed topics in game theory. Since real-life agents behave vastly different to the solutions of classic game theory given the payoff specifications of the game, alternative ways to describe and explain such behavior are high in demand. This paper is concerned with one specific new approach for human behavior in public good games. Learning models. Specifically, this paper will answer the question whether reinforcement learning, one specific type of learning model, is able to describe the behavior of humans accurately. The focus is on games with 40 to 100 players and multiple rounds of play. In order to achieve this, the results of multiple different real-life experimental public goods games will be compared to the results of an implementation of a specific reinforcement learning model taken from (Roth and Erev,1995). Since the experimental studies only provide the aggregated data, the comparison will be executed based on this data. The implementation for this undertaking will be created in Python. The complete code will be provided with this paper to comprehend the results optimally.

Table of contents

Reinforcement learning and the provision of public goods – By Gideon Stein

I. Introduction

How do people interact with each other in competitive environments? This is one of the core questions of game theory and experimental economics. The answer to this question obviously depends on the kind of environment that is looked at. One of the most analyzed and modified environments is the public goods game, an environment, in which people can provide money to a public good. This good will then later be payed out equally to the agents after it was multiplied by some factor x. Various modifications to this basic structure allows this game to describe a number of real-life environments. Included in these modifications are, for example, the payout factor, the possibility to punish other people and multiple rounds of play. Surprisingly, people tend, over all these different modifications, to differ from the game theoretic solution for this game. Considering that, scientists have tried to provide alternatives to explain the behavior of humans in these games. A number of different directions were taken in this attempt. There are various explanations reaching from reciprocity considerations and payoff alterations over irrationality to learning behavior models. This paper will take a deeper look at the latest attempt. The idea behind these models is, that people don't act rationally, but learn about the game while they play it, and then update their behavior accordingly to their new information. Many different models were created to depict this idea. They differ in parameters, information gathering and also information usage. This paper will focus on one commonly used type of learning model: Reinforcement learning. The results of the model will be compared to real life data to understand to which level aggregated human behavior in public good games can be described by the model. Settings with a large number of players will be focused. To do this, this model will be implemented as a python program which than can predict data. Before that however, the general outline of this paper will be provided. The exact definition of public good games and the fraction of these games that will be focused on in this paper will be stated. After that, the game theoretic solution to these games will be shown as well. In the next step, the observations of human behavior in public good games will be summed up. Based on that, the general baseline for this paper is set. Knowing all that, the model that will be used in this paper will be defined and implemented, and their parameters will be explained. This paper will have a specific chapter that focuses on the implementation of the model. This will secure, that the implementation works correctly, and their predictions are according to the theoretic model. Also, this might help other authors to implement and test their own models. After the implementation is finished, the model will be optimized to describe the experimental data as good as possible. The results of this process will then be stated and summarized. General statements, conditions and limits of the accuracy of the model will be formulated and at the end of this paper. As a bonus, the implementation of the model that is used will be provided freely for all readers. This secures that results and statements can be comprehended easier and readers can test out things for themselves. Also, the open implementation of the models might help future research on this topic.

1.1 Method outline

To test the hypothesis, whether a reinforcement learning model can accurately describe the aggregated behavior of humans in public goods games, it is essential to outline the method that will be used. The results of this paper will be based on a graphical comparison of the model results and actual human behavior in laboratory settings. The two papers (Weimann et al., 2012) and (Isaac and Walker, 1988) will provide this laboratory data. Unfortunately, the complete data of these experiments is not included in the papers, but only the aggregated data in the form of graphs. These graphs state values like average contribution or free-riding behavior for all players and rounds. To evaluate the reinforcement learning model, the same graphs will be created for the results of the model and the resemblance of these graphs will be analyzed. Based on this comparison, the leading question of this paper will be answered. The execution of this process has three steps. First, the reinforcement learning model must be implemented, which will happen in Python. The python program will output the various graphs that are needed for the comparison. After that, various parameters of the model must be optimized in order to make use of the full potential of the model. Finally, the fitted model can be compared to the experimental data. Obviously, an only graphical comparison is not optimal. A set of statistical methods would deliver more precise answer to the leading question. However, the results of this paper should not be dismissed based on this factor. A strong resemblance in values like "average contribution" delivers undeniable evidence in favor of the hypothesis of this paper.

1.2 General setting

Public goods games are a group of games, that received a lot of attention in game theory and experimental economics over the last years. Due to its broad definition, there are many variants of the basic game. However, the basics are mostly the same. The game is centered around the problem of providing a so called public good to a group of players. Every player has a given amount of money or tokens at the begin of the game (C). They then can decide which amount of money they want to contribute to the public good (C_p). All the money that was contributed will then be add up and multiplied by a factor between 1 and <N (factor smaller than the number of players N) (M). This sum will then be split among all players equally. Two things are important here. Players keep the rest of their money that they didn't contribute for themselves. The received amount of money of a player is therefore the total amount of money that was contributed divided by N plus the money that he kept for himself. Also, players get their equal share of the public good, even if they didn't contribute anything to it. This behavior is often called "free-riding" in the literature. Formally the payoff (P) in every round for a player i is:

$$P_i = C_i - C_{pi} + (M*(C_{pi} + \sum C_{pj})/N)$$

where $\sum C_{pj}$ *is the sum of all other contributions to the public good and* $M < N$.

Until now, the game is a "single-shot" game, which means it is only played once, and the received amount of money will be payed out after the rounds ends. However, many experiments choose to play multiple rounds to see what happens to the behavior of the players over time. The most unintuitively results are produced in such experiments. Additionally, to the variability of rounds to

play, there are a number of variables that can also vary from game to game. Important ones are group size, multiplication factor of the public good or the MPCR (Marginal per capital return) of the game and the amount of information that is given to every player. Of course, the behavior of players varies with these components. This is important to note, since the goal of the paper is to implement a learning model that can describe real life behavior. To do that effectively, it is important to know the settings of the game and corresponding behavior of players that is to be described.

While the most variables are very intuitive, the concept of MPCR is slightly more complicated. The MPCR of a game denotes how much additional return a single player will get if he provides one additional money unit to the public good while everything else stays the same. Formally, MPCR is defined as:

MPCR = M/N

So, a MPCR of e.g. 0.2 denotes that if a player provides an additional money unit, he will get back 0.2 money units from his contribution assuming the rest of the players don't change their contribution.

1.3 Specific setting

As it was stated earlier, this paper will focus on describing behavior specifically for large groups. To be even more precise the number of players (N) will be between 40 and 100. The experimental results of (Weimann et al., 2012) and (Isaac and Walker, 1988) will be used as the basis of comparison for this intent. The experiment features a setting with up to 100 people and 10 rounds of play. Additionally, the MPCR lies between 0.02 and 0.7. The bigger MPCR's come from Isaac and Walker's experiments, where as the lower MPCR's come from the experiments of Weimann et al.. Players don't have additional information about the contributions of specific players. These specific settings of the game will also be the settings for the games that are simulated by the reinforcement learning model. All the mentioned variables will be equal.

II. Theoretic foundation

Given that that setup of the game is now clear, it is important to understand why the prediction of behavior in public goods games is such a prominent field of study. The first and foremost reason for that seems to be the fact, that the prediction of behavior for players given classic game theory differs from actual observed behavior. Assuming correctly specified payoffs, classic game theory has a hard time explaining human behavior in these games. Other explanations of behavior can therefore support the understanding of these games. To set the general framework for this paper the discrepancy between game theoretic solutions and human behavior will be precisely formulated in this section. The internalization of this discrepancy is important to understand the aim of this paper and helps with the evaluation of the reinforcement learning model.

2.1 Game theoretic solution

How should rational players act in a public goods game on a theoretic level? This question is studied by classic game theory. The answer to this is rather clear. It is traditionally answered by the calculating the Nash Equilibrium (NE), a procedure that is one of the fundamental solution concepts of game theory. A Nash Equilibrium is a strategy profile for a game, where no player has an incentive to one-sidedly deviate from his strategy. In other words: Once the players all begin to play their strategies that are included in the NE, no one can raise his expected utility by changing his strategy one-sidedly. Rational players are assumed to play a Nash equilibrium if it exists since it can't be exploited by the opponents. To make it even more clear: "A Nash equilibrium is an action profile a^* with the property that no player i can do better by choosing an action different from a^*_i, given that every other player j adheres to a^*_j" (Osborne,2004, p.20). In the case of public good games, the Nash equilibrium is best shown on a very simple 2-player normal form public goods game: Consider a game with two players where every player has 1$ at the beginning of the game. Both players have the option to contribute their Dollar to the public good or keep it for themselves. After both players have decided on their play, the contributed money will be multiplied by 1.5 and distributed between both players equally. This game has the following normal form:

	Cooperate	Don't cooperate
Cooperate	1.5, 1.5	0.75, 1.75
Don't cooperate	1.75, 0.75	1, 1

Table 1: public goods game with two players

In this form, the Nash equilibrium is easily spotted. For both players "Don't' cooperate" is a strategy that has no alternative that has a better payoff no matter the opponent's strategy. The strategy profile ("Don't cooperate", "Don't cooperate") therefore fulfills the requirements for a Nash equilibrium. Additionally, to that, ("Don't cooperate", "Don't cooperate") is the only NE in the game. Another way to look at this is, that ("Don't cooperate") is a dominant strategy for both players. Therefore, it should be played by rational players. While the game theoretic solution for this simple game is clear, it is now required to apply these results to more general public goods games. In principle there are three dimensions in which this normal form of a public goods game can vary from the basic game. Rounds of play, number of strategies and number of players. For all three dimensions it can be shown that the unique NE does not allow strategies that cooperate at any time. The basic game that was analyzed until now is a so called "single-shot game". This means it is only played once. However, public goods games are often played multiple rounds. How does this affect the NE? For repeated games, repetition effects must be considered. However, since classic public goods games feature anonymous contributions (only the aggregated contribution is known but not the individual contribution) and the players cannot draw any conclusions about the other players from their information, these effects can be discarded. Since the players cannot communicate with each other or form coalitions, the last round is equal to a single shot game. Using backward induction this effect transfers itself to all other rounds. All rounds will be played as single shot games which requires the players to not cooperate in every round. Since repetition is not included in the basic game itself, the NE from the single shot game will simply be played in every round. Another way of thinking about this is to look for dominant strategies. As long as no conclusions about the contributions of other players can be drawn, playing "Don't cooperate" in every round of the game is still a dominant strategy over all other alternatives (since there are no repetition effects) no matter

the number of repetitions. The following normal form game should illustrate this for 2 rounds of play. In order to secure comprehensibility of this normal form game, the strategies will only be displayed in curtailed form. While the true normal form of this game includes eight different strategies for every player, the different off-equilibrium paths are not relevant for the example since they produce no new payoff outcomes. They will therefore be ignored. This shortening does only prevent duplicate cells and does not alter the NE of the game.

C = cooperate DC = Don't cooperate	Round1:C, Round2: C	C, DC	DC, C	DC, DC
C, C	3, 3	2.25,3.25	2.25,3.25	1.5,3.5
C, DC	3.25,2.25	2.5,2.5	2.5,2.5	1.75,2.75
DC, C	3.25,2.25	2.5,2.5	2.5,2.5	1.75,2.75
DC, DC	3.5,1.5	2.75,1.75	2.75,1.75	2.2

Table 2: public goods game for two rounds of play

In order to produce the same results for any number of rounds, for the two-player case, one restriction has to be considered. In order to prevent the possibility to draw conclusions about the opponent, the payoffs of the game will not be announced after every round, but only after the end of the game. This prevents the players to be able to calculate the exact contribution of the other player based on the payoff of a round. As long as this restriction is fulfilled, the unique NE cannot include strategies that include cooperation.

The second dimension that can vary is the number of strategies. In many public good games, players have not only the option to contribute all or nothing of their money to the public good, but often can contribute any part of their money to the public good. E.g. players have 10$ at the start of the game and can contribute anything between 0$ and 10$ to the public good. How do these additional strategies affect the NE of the basic game? Additional strategies don't alter the Nash equilibrium. As DC dominates C, it also dominates every other strategy that contributes to the public good. The following normal form game should illustrate this for 3$ budget and a public good multiplicator of 1.5.

	Contribute 0$	Contribute 1$	Contribute 2$	Contribute 3$
Contribute 0$	3, 3	3.75,2.75	4.5,2.5	5.25,2.25
Contribute 1$	2.75,3.75	3.5,3.5	4.25,3.25	5,3
Contribute 2$	2.5,4.5	3.25,4.25	4,4	4.75,3.75
Contribute 3$	2.25,5.25	3,5	3.75,4.75	4.5,4.5

Table 3: public goods game with four options

As It can be seen, the strategy "contribute 0$" still dominates all other strategies. This holds no matter how many different distribution strategies there are. The NE of the basic game is therefore the same.

The third dimension that can differ is the number of players in the game. Many public goods experiments don't restrict themselves on only two players but include groups of multiple players that all have the same basic option to contribute to the public good. Surely this should alter the NE of the basic game. In fact, formally there is a different NE, since it must include more strategies. However, what is also true is, that the strategy DC is still dominant over all other strategies, concerning the actions of all other players. The following normal form game should again illustrate this for 3 players with binary choices and a public good multiplicator of 1.5.

Cooperate (strategy of Player 3)			Don't cooperate (strategy of Player 3)		
	Cooperate	Don't cooperate		Cooperate	Don't cooperate
Cooperate	1.5,1.5,1.5	1,2,1	Cooperate	1,1,2	0.5,1.5,1.5
Don't cooperate	2,1,1	1.5,1.5,0.5	Don't cooperate	1.5,0.5,1.5	1,1,1

Table 3 and 4: public goods game with three players

After calculating the expected utility of every strategy, "Don't cooperate" is still the best option for all players. Given that, the NE of a standard public good game can be stated like this:

For a standard public goods game with n players, the unique NE of the game only includes strategies that never allow cooperation.

Noteworthy is the fact that this proposal assumes a MPCR of <1. If this is not the case, players get more return if they contribute everything and the NE is different. With this definition, the classical game theoretic solution for general public goods games is clear. Given rationality assumptions, every player should not contribute anything to the public good no matter how many rounds the game is played and how many players are participating in the game. Or formulated differently: "In the public goods game (PGG), the only Nash equilibrium (NE) that is based on monetary considerations is for all players to free ride." (Dong et al. ,2016, p.1).

Given this solution, it is now time to look at actual human behavior and whether it resembles the predictions from above.

2.2 Human behavior in public goods games

Obviously, the exact human behavior that occurs in public goods experiments widely varies due to the distinct nature of the experiments. Factors like the public good multiplicator, punishment options (which is strictly speaking a different game), knowledge about the outcome and rounds to play

greatly influence the contributions to the public good game. In general, however, concerning the public goods setting that was described above, there are two main trends that are essential. (Chaudhuri, 2011, p.2) states these two as following: 1. “In one-shot versions of the public goods game, there is much more contribution than predicted in the Nash equilibrium of the game.” According to (Chaudhuri,2011), the level of contribution ranges from 40% to 60% of the maximum contribution in these one-shot games. An observation that is counterintuitive to the game theoretic solution from above. And 2. “if the players interact repeatedly over a number of rounds then contributions often start out at between 40% and 60% of the social optimum and decline steadily over time as more and more players choose to “free ride.”” (Chaudhuri,2011, p.2). Additionally, to that, in many experiments, the public good provision never goes to zero but remains at a level close but over zero. This was for example shown by the experiment stated in (Isaac and Walker, 1988). Again, this is different from the game theoretic solution from above. What this means is, that humans do not seem to behave according to the solutions of classic game theory. Two explanations are possible. On the one hand, humans might not behave rational and are therefore not describable when using classic game theoretic assumptions. On the other hand, the payoff specifications of the game might not be equal to the payoffs in the actual game. In any case, learning models can provide a valuable addition to the understanding of these games. Whether the decision mechanism of humans can be represented as a form of a learning model will be the question of the next chapters. However, before this process can be started, there are three more points worth mentioning concerning human behavior in these experiments. The first thing that is important to realize, is that the contribution levels are aggregated values consisting of the decisions of all players. A contribution level of 40% does not mean, that the individual players all contribute on similar levels. Contribution levels can have a high standard deviation. E.g. it is easily imaginable that a group consists of players has 6 “free-riders” and 4 “unconditional cooperators” which would lead to a 40% contribution level. This is important to keep in mind while looking at contribution levels. Individual contributions therefore often do not steadily decline, like the aggregation suggests, but can easily go from 100% to 0%. A prediction of individual behavior might require a whole different model than the model concerning the aggregated results. Secondly, recent literature suggests that: “…there are distinct types of players in such games who differ in their social preferences and/or their beliefs about their peers,…”(Chaudhuri, 2011, p.3), which is also to be kept in mind when analyzing aggregated contribution levels. There seem to be three types of players. Unconditional cooperators which always contribute a high amount to the public good, unconditional free-riders which always free-ride and conditional cooperators which cooperators based on the behavior of the other players. Normally a large part of the players in public goods games can be classified as conditional cooperator. Instead of modelling every agent in the same way, an inclusion of such classes might improve models significantly. The third point that is worth mentioning is the fact, that contribution levels are not negatively correlated with the group size but might even be positively correlated. “contrary to intuition larger groups are no worse—and may even be better—at providing the public good than smaller ones.” (Chaudhuri, 2011, p.3), “On the contrary, groups of size 40 and 100 provided the public good more efficiently than groups of size 4 and 10” (Isaac and Walker,1988, p.1). Since this paper is mainly concerned with bigger group sizes, this may be useful to keep in mind while evaluating the models.

III. Introduction of the experimental studies

Before explaining the model that is used in this paper, it is worth mentioning the reason behind choosing the two mentioned studies and the specifics under which the results of the experiments that are used in this paper were conducted. Since experiments with such large group sizes are rather complicated to implement, certain modifications were added to make the experiments possible. These modifications should be kept in mind throughout this paper. While there is a vast amount of different public goods experiments, experiments with a high number of players are rather rare. This is due to the organizational effort and the monetary expenses of such experiments. Considering the aim of this paper, there were therefore not too many different studies to choose from. Despite that, the two studies have two features that make them especially fitting for this paper. Firstly, both studies use no modifications to the classic public goods game. This is necessary to hold the results of this paper as general as possible. Secondly the two studies feature a large difference in MPCR which is essential to depict a large number of different games.

3.1 Weimann et al.'s experimental study

The goal of this study was amongst others to check for group-size effects and MPCR effects in large groups. To achieve this, six different experiments with 10 rounds each were executed over the internet. For every experiment there were eight groups that played the exact same game. Four locations in Germany were connected via skype which gave the participants the opportunity to see each other. Contrary to (Isaac and Walker, 1988) this means that all experiments were finished in a single session and decision were made simultaneously. Two of the experiments were used to secure that the connection of different laboratories had no impact on the result of the game. Groups of eight people which played the game locally were compared to groups of eight that played the game over the internet and no significant difference was found. The other four experiments were 60 and 100 players games with a MPCR of 0.02 or 0.04. These MPCRs were used since MPCR effects of very small MPCRs were tested. Contrary to (Isaac and Walker, 1988), The participants were rewarded with real money. A total number of 2840 subjects participated in the experiment and they received on average a payoff of 15.36€. Due to missing participants, the last experiment could not reach the full group size of 100 people. Therefore, the groups of the game with 100 players and a MPCR of 0.04 has only on average 95.5 players. While this might have an effect on the comparison of the two 100 players experiments in the study, it is irrelevant for the purpose of this paper. All 4 large group experiments (60,100 players and MPCR of 0.02 and 0.04) will be compared to corresponding results of the reinforcement learning model.

3.2 Isaac and Walker's experimental study

The goal of this large-scale experiment was amongst other things to test for group size effects in public goods games. To test this, different experiments were conducted with varying amounts of players ranging from 4 to 100 and with 10 rounds of play. In order to handle such a large number of experiments and participants, some modifications were added to the games that are typically not included in laboratory public goods games. First of all, all games were carried out via computers. Participants were able to choose their play in every round via a program at the computers. Due to the large number of participants, the majority of games were played in multiple sessions over multiple days. Therefore, decisions were not made simultaneously. Since the multiple session

experiments required participants to decide on plays on multiple days, there was a possibility for participants not showing up for some sessions. To prevent missing data, missing decisions were interpreted as no contribution to the public good. Participants for the experiments were students that were enrolled from different microeconomic classes. To reward these students, all multi session experiments offered not cash rewards, but additional credit points for the students. Students could therefore boost their grades in the specific classes. Whether this leads to different behavior compared to the monetary rewards was tested and no significant difference was found. “For a specific group size and MPCR, the aggregate pattern of token allocations in the VCM-MS-XC environment and the VCM-SS-$ environment studied by IW are very similar.” (Isaac and Walker, 1988, p. 10). VCM-MS-XC stands for MS extra credit reward and VCM-SS-$ stands for single session monetary rewards. For the goal of this paper, only the results of the large-scale experiments with 40 and 100 players will be used to compare it to the results of the reinforcement learning model. Here, two different MPCR levels are available. Both levels (0,3 and 0,75) will be simulated by the reinforcement learning model.

IV. Introduction to Learning models in general

Before defining the specific learning model that is used in this paper, it is worth to talk about learning models in games in general. Especially two questions arise when first dealing with learning models. Why are learning models used to analyze games, and what kind of learning models exist currently?

Classic non-cooperative game theory is mainly concerned with equilibria in games. Especially Nash equilibria are often considered the solution to a game by classic game theory. Although these results include a considerable amount of information and provide a very essential way of analyzing a game, numerous problems occur when using this way of thinking. The equilibrium solutions of games are often not observed in reality. This makes it hard to use these solution concepts to describe and predict actual behavior and cut back the usability in situations in which actual humans are included. The most prominent explanation for this phenomenon is, that the assumption that classic game theory makes are way to strong. “One traditional explanation of equilibrium is that it results from analysis and introspection by the players in a situation where the rules of the game, the rationality of the players, and the players’ payoff functions are all common knowledge. Both conceptually and empirically, these theories have many problems.” (Fudenberg et al., 1998, p.1). This is the first reason to consider learning models in games. Learning models make way less assumptions than classic game theory. The agents in these models are therefore in many situations closer to actual human agents while also resulting in equilibria as well. Based on that, they produce solutions to the games that are more similar to actual human behavior and are therefore more relevant in real-life applications. Secondly, classic game theory struggles with equilibrium selection. In games with multiple equilibria it is not clear which solution should be played. Learning models however, can often give an answer on which equilibrium will be played given certain initial parameters. The third and final reason to consider learning models in games is the fact that they can give an inside on the dynamic processes which occurs in a game. In real-life games with multiple round of play, it is often observed that agents often play strategies close or equal to the NE in later rounds but deviate from the NE in earlier rounds of the game. Learning models can provide a way to accurately describe such observations.

Based on these reasons, learning models provide a perspective on games different from classic game theory that can lead to a better understanding and better descriptions of the dynamics and the outcomes of games.

Now that it is clear that learning models have a high utility when analyzing games, it is also necessary to mention the most common types of them. When looking at the corresponding literature, there are three types of learning models. Models that use the replicator dynamic, reinforcement learning models and belief learning models. Almost all learning models for games can be classified as one of these categories, although different models use various additional parameters. The core ideas of these models should therefore be clear. With that knowledge, the model that is used in this paper can be categorized better. Additionally, it will be easier to realize the parameters that are included in the reinforcement learning model of this paper.

4.1 Replicator dynamic models

The first model type that finds applications in various games are replicator dynamic models. The core of this model is the replicator equation. An equation that origins from evolutionary biology but found its way into game theory and especially evolutionary game theory. Suppose there is a population of players that can only play one specific strategy each and there are different shares of the population that play different strategies. Over multiple rounds the players will get randomly matched against each other and get a payoff based on a payoff matrix. Given that payoff, the fitness of a specific player type can be calculated by comparing it with the average fitness of the population. This fitness determines how the shares of different strategies change in the next round. An over-average fitness leads to an increase of the share of the specific strategy in the population. An under-average fitness leads to a decrease. Over time, strategies with a high fitness tend to be played by bigger and bigger shares of the population.[1] The basic replicator equation is defined as:

$$\dot{X}_A = (F_A(X) - \emptyset(X)) * X_A$$ *where* $\dot{X}_A$ *is the growth rate of the a strategy A,* X_A *the current share of the strategy A in the population (0,1),* F_A *denotes the fitness of a player type A depending on the population shares* $\boldsymbol{X}$ *(vector) and* $\emptyset$ *denotes the average fitness of the population depending on the population shares* $\boldsymbol{X}$*.*

For a complete derivation of this equation see (Weibull, 1995). The changes of the shares of the population can be interpreted as a learning process. The population learns which strategies lead to more success and adapts them. On the first sight, this learning is different to belief and reinforcement learning on a fundamental level. While in reinforcement and belief learning models, the individual players learn while they play the game, in replicator dynamics there is no individual learning.

Players in standard replicator dynamic models are programmed to only play one specific strategy and can't change. Based on their success their strategies reproduce with different rates which leads to the learning process described above. This learning however is only on an aggregated level. This fact makes it hard to model games in which the players are fixed and play for multiple rounds. However, with different interpretations of the populations, this is also possible. Instead of looking at a population as an actual population of players, one also can interpret it as a population of strategies in the mind a single player.

1. However, fitness of strategies might change over time as well, since the payoff is dependent on the prevalence of different strategies in the population.

Or formulated differently: “The change which the “population of ideas” in the decision maker’s mind undergoes may be analogous to biological evolution” (Börgers and Sarin, 1997, p.1). The shares of the strategies then change based on their fitness. Replicator dynamic models are very flexible models which can be fitted to many different games and have the possibility to be expanded in various ways. Interesting examples for this are (Hauert et al.,2002) and (Börgers and Sarin,1997).

4.2 Belief learning

The second model type that is widely used is belief learning. The basic idea behind belief learning is defined as following: “…, belief learning refers to models in which players are engaged in a dynamic game and each player optimizes,…, with respect to a prediction rule that gives a forecast of next period opponent behavior as a function of the current history” (Durlauf and Blume, 2010, p.9). What this means is that an agent that acts according to a belief learning rule is doing three things. 1. Forming beliefs about the actions of the opponent players for the next round based on the knowledge of their plays in previous rounds. 2. Choosing the best action based on the formed belief about the other players and a predefined decision rule 3. Receiving the payoff for the current round and adding the new observed actions of the opponents to the knowledge. These three steps are repeated as long as the algorithm interacts with his opponents and has to make decisions. Two ways are possible in which belief learning models can differ from each other. While additional parameters are a important criterion for differences between models, they are not the most striking source for differences. What is far more relevant is the definition of the belief forming function and the decision rule. While most belief learning models use maximizing of expected utility based on the current beliefs as the decision rule, they often differ in the belief forming function. One of the main concerns is e.g. how older evidence should affect the current beliefs. Cournot learning and fictious play are a prime example for this debate. While in Cournot learning only the last round effects the current belief of a player, fictious play calculates the average of all evidence that is available. Obviously both rules create vastly different results while both being valid belief forming functions for a belief learning model. Additionally, much more complicated belief forming functions can be imagined. When modeling a game with a belief learning model, each player is represented by an independent belief learning algorithm which plays accordingly to the beliefs that it formed based on previous rounds of the game. In every round, the player receives new information about his opponents and he learns about the game. This is a key difference between belief learning models and replicator dynamic models. All players learn individually for themselves. Interesting implementations are (Cheung and Friedmann,1997) or (Nyarko and Schotter,2002).

4.3 Reinforcement learning

The third and final type of learning models is reinforcement learning. While the core idea of the model might seem a little bit primitive, it performs very good in various environments.[2] Again many different models can be defined as reinforcement learning models. They share one defining idea. Stated by (Rouse,2018, p.1): “Reinforcement learning is a training method based on rewarding desired behaviors and/or punishing undesired ones.” What this means is, that a reinforcement learning algorithm learns about its environment by acting and receiving the return of that action.

2. As long as they are stable. When the game environment is changing rapidly or continuous, reinforcement learning might perform way worse than the other two models.

In a game, every player is modeled by its own reinforcement learning algorithm, which is a similar feature to the belief learning models. At the beginning of the game, all possible actions have the same probability to be played. After choosing an action for the current round, the algorithm checks whether the action was bad according to a so-called reward function. If It was, the action will be negatively reinforced, and the action will be played less in the future. If the action was good according to the same function, the action will be positively reinforced, and the action will be played more often in the future. The more actions the algorithm performs, the better will he be informed about its environment, and ideally perform better according to his reward function. To put it even simpler: The algorithm is learning about its environment by trying out different strategies. To specify the reinforcement learning that is used in a specific game, one now has to define the reward function and the method how the probabilities of all actions are getting altered when receiving new information. Often, the reward function is just defined as the costs minus the return of an action, while propensity stacks are used to accord for changes in the probabilities of the different strategies. Additionally, different parameters can be added to further expand the basic model. While (Roth and Erev,1997) is the most prominent implementation of such a model in the field of game theory its applications reach far beyond the field of game theory. A good example how powerful these models can be is (Mnih et al.,2013) where a simple reinforcement learning model is learning to play video games on its own.

V. The specific reinforcement learning model

5.1 Roth and Erev's model definition

The model of (Roth and Erev,1995) is a model that uses reinforcement learning to determine the actions for all players in different games (mostly 2-player normal form games). However, the model has additional parameters that expand the basic idea of reinforcement learning. The reward function of the model is very simple. Since the model is initially used to play normal form games, the reward for an action is defined by the corresponding payoff in the normal form of the game. In the case of a public goods game with many actions and many players, the reward function is still simple and was defined as the following: The reward R of an action A is the difference between the amount of money that was provided to the public good (PG), and the return from the public good (RPG). Or formally:

R(A) = RPG – PG.

Note that R(A) can be negative which corresponds to a negative reinforcement. The method on how the probabilities for all actions get altered in this model is slightly more complicated, however still very intuitive. Every action has a so called "propensity stack" which represents the sum of all reinforcements. The propensity stack has an entry for each action in the game. Every time the player plays a certain action, the reinforcement value will be added to the corresponding stack. To calculate the probability for a specific action in the current round, the propensity stack for that action will be divided by the sum of all propensity stacks. In order to perform this calculation in the first round, the propensities have to have an initial value that is not zero. Additionally, propensity Stacks cannot be

negative. If a reinforcement is leading to a negative stack for a specific action, the stack will be set to 0. The idea behind this calculation is that the propensity stacks of different actions grow or shrink faster than others because of their rewards. This means: "…pure strategies which have been played and have met with success tend over time to be played with greater frequency than those which have met with less success" (Roth and Erev, 1995, p.172). Therefore, with enough rounds, the reinforcement model should tend towards a Nash Equilibrium if such exists. This mechanism can also be described formally:

A number $q_{nk}(t)(\leq 0)$ *describes the propensity of a player n for the period t for the action k that is included in the action space of n. If k is played and achieved a reward of x, then the propensity is updated as following:* $q_{nk}(t+1) = q_{nk}(t) + x$ *while for all other actions y:* $q_{ny}(t+1) = q_{ny}(t)$*. The probability that player n plays the action k in the period t is described as* $p_{nk}(t) = q_{nk}(t) / (\sum (q_{ny}(t)) + q_{nk}(t))\ (0,1)$*.*

In this basic form, it is noteworthy that x has a decreasing impact on the probabilities. "a payoff of x from playing pure strategy k at time t has a bigger effect on $p_{nk}(t)$ when t is small than when t is large" (Roth and Erev,1995,p.172). This is due to the fact, that propensity stacks tend to get bigger over time and the additional value of x diminishes. This however can be altered by one of the parameters that are introduced now.

Three parameters are introduced to the basic model in (Roth and Erev,1995). These parameters alter the probabilities and the propensities in one way or another to assure better predictions in various games. There is a so called "cutoff" parameter (μ), an "experimentation" parameter (Ɛ) and a "forgetting" parameter (ϕ). They are built into the equations of the model and alter the results. In principal they can be chosen freely which can lead to massive differences in the output of the model. A future challenge will therefore be to optimize these parameters in terms of resemblance. First off, the "cutoff" parameter is a value that restricts actions with very small probabilities. If an action has a probability that is lower than the "cutoff" parameter, its probability and its propensity will instead be set to 0. There are a couple of ideas that stand behind this parameter. However, the biggest two of them are probably to speed up convergence and speeding up the computation of the model. The "cutoff" parameter can lie in the range from 0 to 1 while high values are rather counterintuitive and probably prevent the model to deliver good predictions. Additionally, high "cutoff" values lead to a higher variance since the probability of good strategies might be reduced to 0 by accident. Because of these two disadvantages, a rather low "cutoff" value is optimal. Formally:

If $p_{nk}(t) = q_{nk}(t) / (\sum (q_{ny}(t)) + q_{nk}(t)) < \mu$ *then* $p_{nk}(t) = q_{nk}(t) = 0$ *for* $\mu\ (0,1)$

The second addition is the so called "experimentation" parameter. Its purpose is to promote local experimentation of different actions. The idea behind it is, that if an action A leads to a positive reward X, X is not completely added to the propensity stack of A but a small portion of X equal to the "experimentation" parameter is added to the neighbors (the most similar actions) of this action. The value of the "cutoff" parameter can again range from 0 to 1, Where 0 states that 0% of X will be awarded to neighbors and 1 states that 100% of X will be rewarded to neighbors. Behind the idea of

local experimentation, the parameter can represent error of players and also prevent the model from converging to only play one specific action (given the "cutoff" parameter is active). Again, small values for the "experimentation" value deliver the best predictions. Formally:

"$q_{nk}(t+1) = q_{nk}(t) + (1 - \varepsilon)x$. The remaining quantity εx will be added to the propensities to play the strategies adjacent to strategy k" (Roth and Erev, 1995, p.175) for ε (0,1)

The third parameter that is introduced in (Roth and Erev,1995), is the "forgetting" parameter. Its purpose is to enable to model to forget old information. This is achieved by multiplying all propensities with (1 - "forgetting" parameter) after every round. While doing this, it is guaranteed, that the propensities do not steadily grow as more and more rounds are played. Therefore, new rewards always remain significant. Again, the value of the parameter can range between 0 and 1 where 0 means that no information is forgotten while values close to 1 means that almost all information from the past is forgotten. Small values for the parameter are again be the most promising ones concerning the predictions. Formally*:*

"At the end of each period t, each propensity $q_{nk}(t)$ obtained as before is multiplied by $I - \phi$,…" (Roth and Erev, 1995, p.175) for ϕ (0,1)

With the basic model and the additional parameters clear, only the implementation will show whether it can describe human behavior in specific public goods games.

5.2 Implementation of the model

The implementation of the reinforcement learning model is the central part of this paper. With it, it is possible to compare the results of the model with actual human behavior in an efficient way. Based on this comparison, the model can be evaluated. This new knowledge then can be used to answer the key question of this paper. Since this paper is not directly concerned with the implementation of the models but rather with the results of this implementation, there will be no direct references to the code. Additionally, the explanation of the different steps that were carried out to create the implementation will be structured in a way, that people without a background in programming can also understand it. The code of the implementations was created in the Software "JupyterNotebook", a basic and widely used development environment for Python, that makes it very easy to structure the program. It gives the developer the possibility to easily write commentary between code and structures and the whole code in the form of a sheet-style document. Also, code can be written in code boxes that can be executed separately and provide a good way to structure the program in individual functional code snippets. The different parts and calculations were organized with the help of this tool. This should maximize the ability of the reader to comprehend the implementations if they choose to take a closer look of the code file. The code will be freely

available on GitHUB. When explaining different parts of the implementation, the corresponding code boxes will be named to draw a connection between this paper and its code.

In order to maximize the comprehensibility and the maintainability of the implementation, the different calculations of the model were divided into functional parts. At the end of this process, any number of public goods games can be simulated, and the results are illustrated in the form of graphs. The first step of the implementation is the initiation of the different parameters and the creation of the initial propensity stack. If needed, parameters can be altered during this step. The propensity stack is initialized as an AxB matrix where A is the number of possible actions and B is the number of players in the game. This matrix will be altered during the computational process, by selecting the specific element in the matrix and replacing it with a new value (boxes 1-3). After the initiation, there are mainly three calculations that define the core of the implementation. The calculation of the actions for each player and the current round, the execution of the public goods game and the updating process of the players depending on the outcome of the game. Calculating the actions for all the players based on their propensities is carried out by transforming the propensity stacks of a player into probabilities according to the definition of the model. After that, a random action will be drawn based on the just calculated probabilities. The actions of all players will then be saved into a list (boxes 6). With this list the public goods game can be executed. The game puts out the money that every receives back after the game ends (box 7). With the outputs of the first two steps, the propensity stacks of all players can be updated. The required payoff of the actions will be calculated by subtraction the action from the received money in the current round. During the update process of the propensities, the "cutoff parameter" and the "experimentation parameter" are included. After a single propensity is updated, it is checked whether its current corresponding probability is below the cutoff. If this is the case, the propensity stack will be set to 0. Furthermore, the payoff will not be completely added to the corresponding propensity stack, but a small part equal to the "experimentation parameter" is added to the neighbor actions. This is however only the case when the payoff is positive. This is due to the fact that negative experimentation can potentially lead to probability problems (e.g. negative probabilities). Furthermore, this is also consistent with the formulation "-strategies close to a high probability strategy must also be played with a positive probability that is bounded away from zero." (Roth and Erev, 1995, p.175). Negative probability effects are not considered in this formulation (box 8). After the main updating process is finished, the third parameter will be applied to the propensity stacks. Here all stacks are simply multiplied by the "forgetting parameter" (box 9). After this step, a single round of the public goods game is over. Since multiple rounds are played in the space of one simulation, all stated code blocks expect the initiation are formulated as functions. This way the functions can be executed multiple times which happens in box 10. Box 10 executes all main functions as often as there are rounds and puts out all plays of the game of all players sorted in a list. Furthermore, it is itself a function that can be executed multiple times. This is essential when applying the Monte-Carlo method (box 16). Boxes 12-15 are used to draw the results as graphs (average contribution and free-riding). Additionally, the individual behavior of every player can be looked at. Boxes 17 and 18 include the code for creating the graphs for a monte-Carlo simulation. All simulations are again drawn in a single graph. After this step, a number of different code boxes include tools to analyze the results of the model and optimize the parameters of the model. From box 23 the rest of the implementation is used for optimization. Finally, it is useful to mention that there are some small addition functions that support the usability

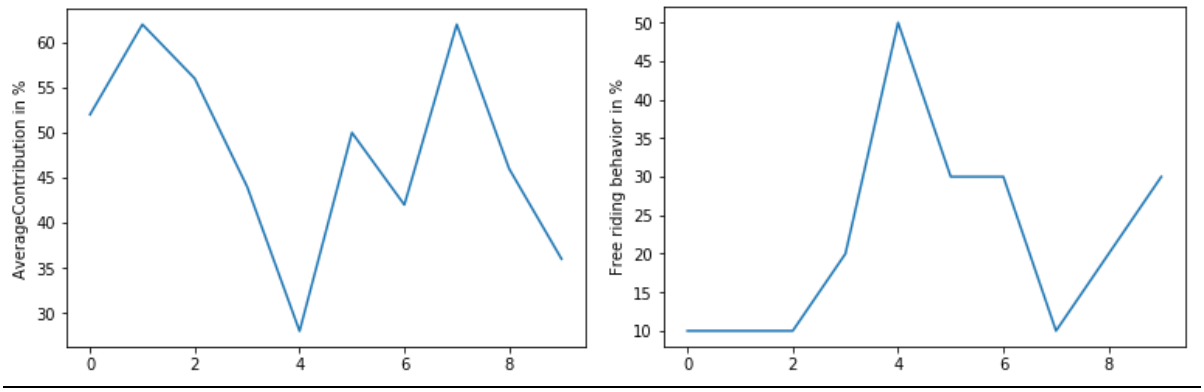
of the implementation. The most noteworthy is the function that resets all propensity stacks after a simulation is finished (box5). Since the stacks are only initiated in the beginning of the program this is necessary to execute multiple simulations successively. Otherwise the propensities would be adopted from the previous game. With the implementation clear, it is now to see whether it can deliver good predictions for actual experiments.

5.3 Test run of the model implementation

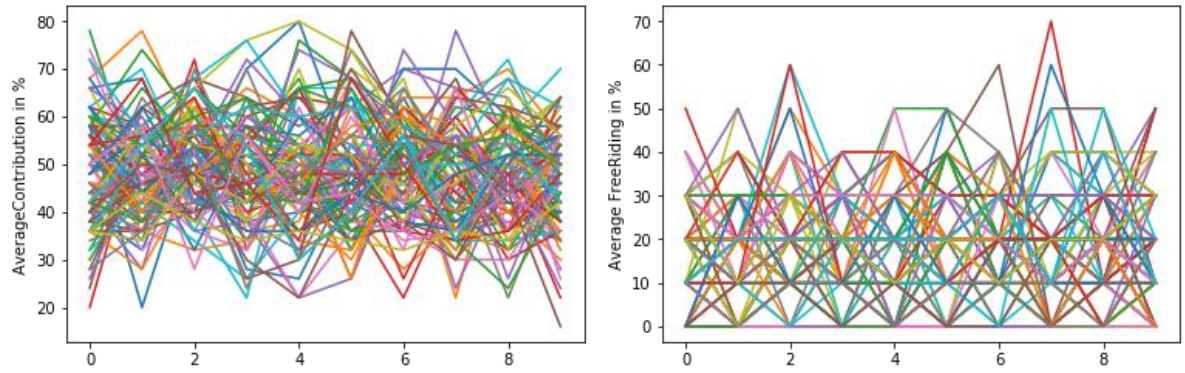
Before trying to predict the specific experiment data with the help of the model, it Is necessary to execute a small test run to secure the functionality of the model. For a small game with 10 players, 10 rounds, a MPCR of 0.5, 5 money units to distribute and the remaining parameters set according to one of the suggestions of (Roth and Erev, 1997, p.177) (The initial propensities will be set to 10 and the experimentation model with forgetting will be used which means $\mu = 0$, $\varepsilon = 0.05$ and $\phi = 0.001$.) Since there is no data for this initial simulation, the parameters can and will not be optimized for the model. It is important to keep in mind, that the results of this simulation are not important, but it should simply secure that there are no bugs in the implementation and the results that are needed for further steps are successfully provided at the end of the implementation.

After executing this first simulation with the parameters and variables set as described above, no bugs were encountered. Additionally, the implementation successfully provides the graphs that are needed for further steps. To analyze the results of the simulation, 4 different graphs will be considered. Two graphs for single simulations and two graphs which show the iterations of a monte-Carlo simulation for the game. Since the implementation makes it easy to show 100 graphs in one picture, the actual iterations will be depicted instead of average values. The idea behind this is that this gives a better look on the behavior of the model. The first important graph is the average contribution of all players for all rounds. It gives a good first impression what the players in the model are doing and whether they tend to learn over time. Graph 1 shows the results for the test run of the model. On the first view, there seems to be no tendency towards contributing less to the public good. Therefore, there can be no learning tendency attributed to the test run model. Since probabilities are a large part of the model, this first observation should be validated by looking at multiple executions of the model. Graph 3 shows a monte-Carlo simulation for 10 games. The graph confirms that the first impression was correct. The players in the model do not seem to tend to contribute less. A learning behavior can not be attributed to the players based on the average contribution. Graph 2 includes information about free-riding behavior over the rounds of the game. This graph can be seen as an addition to graph one. It gives additional information over the model which can be used to compare it to real-life data. When the players in the model are learning, the free-riding behavior should increase over the span of the game. Graph 3 gives no evidence for that, which is in accordance with the result shown in graph 1. The last graph that is considered contains the free riding behavior over time for a monte-Carlo simulation. The reason to consider this graph is again based on the fact that probability plays a large role in the model, and multiple simulations give a better picture on the general tendency of the model. Graph 4 shows these results for the test run. Again, the first impression of the single simulation was confirmed. There is no visible tendency towards more free-riding.

Based on the results of the test run, it can be concluded that the implementation of the model works fine and delivers all data that is relevant for the comparison to real-life experimental data. Furthermore, the model does a poor job in showing any sign of learning behavior in the test run. Since the parameters and the variables of the game were set rather arbitrary and there was no optimizing of the parameters, this should come as no surprise. As it was mentioned above, this was not the goal of this test run anyways. Better results will occur when fitting the parameters for the game variables of the real-life experiments in the next chapter.



Graph 1 & 2: Results of the test run for a single simulation



Graph 3 & 4: Results of the test run for a monte-Carlo simulation with 100 iterations

VI. Evaluation of the model

6.1 Comparison of the model with Weimann et al.'s experimental results

After executing the first test run, it is now time to use the model to predict the first experimental data. Weimann et al.'s experiments consist of 6 different experiments from which four are relevant

for the purpose of this paper. Two experiments with 100 players and two experiments with 60 players, each with a MPCR of 0.02 or 0.04. The model will predict all 4 of these experiments. The basis of comparison will be the two graphs that Weimann et al. included in his paper (graph 5 and 6). They show the average contribution of the players for all 4 games (Weimann et al., 2012, p.4), and the percentage of free-riding behavior during all 4 games (Weimann et al., 2012, p.4). While these graphs form the basis of comparison, there are some characteristics of them that are worth mentioning, since they give the final comparison more depth. First, the initial contribution level depends on the number of players and the MPCR of the game. All experiments show a contribution level of under 50% which drops to under 10% until round ten. Additionally, the average contribution does not fall to 0% which is contrary to the game theoretic solution of the game. The free riding behavior starts between 10 and 30% depending on the game and increases up to 70%. Again, the game theoretic solution would predict 100% free-riding behavior which is not observed in the experiment. Finally, all curves are strictly monotonic. These characteristics and the graphs will now be used to compare it to the results of the reinforcement learning model. This comparison will happen in two steps. First, the model will be executed four times for the four different experiments. This first execution will feature initial parameter values which are not optimized. The first results will be compared to the real- life data, and differences and similarities will be marked. Based on that first comparison, the second execution of the model will feature a parameter fit which aims on reducing the differences observed on the first execution round. The four parameters that are available for a fit are μ, Ɛ, ϕ and the initial propensities. After the fit is finished, the second results will again be compared to the real-life data. The question whether the reinforcement learning model can describe Weimann et al.'s real-life data will be answered based on this final comparison.

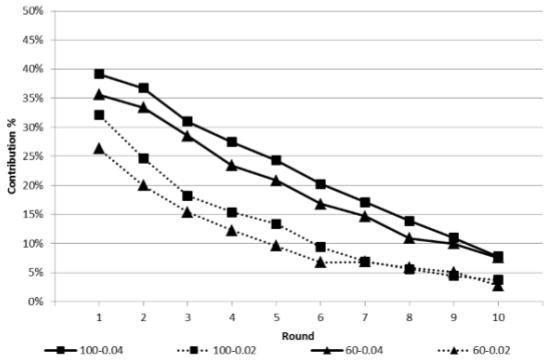


Figure 3: Average contributions to the public good in treatments *T3 - T6.*

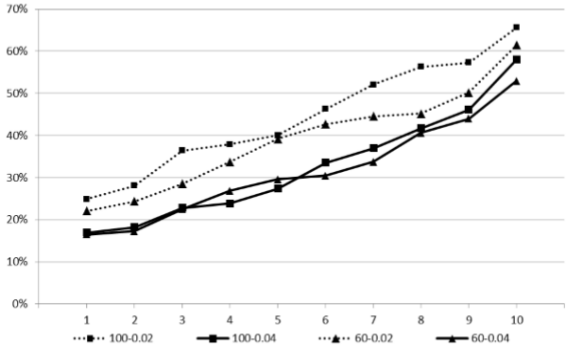
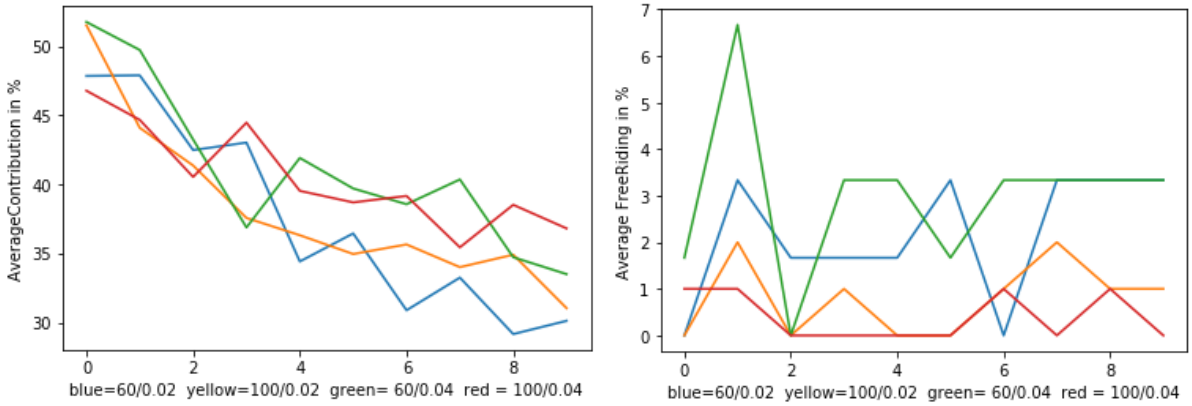


Figure 4: Percentage of strict free riding in *T3 - T6.*

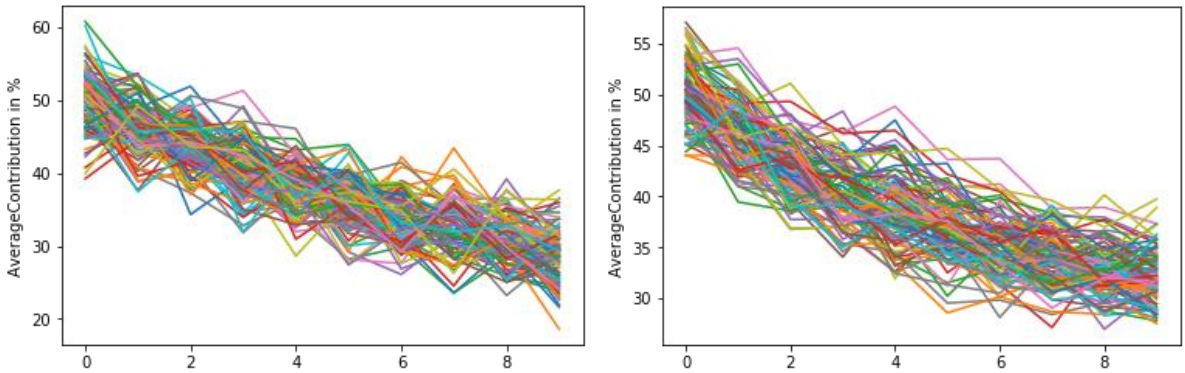
Graph 5 & 6: Weimann et al.'s results for average contribution and complete free-riding (Weimann et al.,2013,p.5)

6.1.1 The first execution for Weimann et al.*'s data*

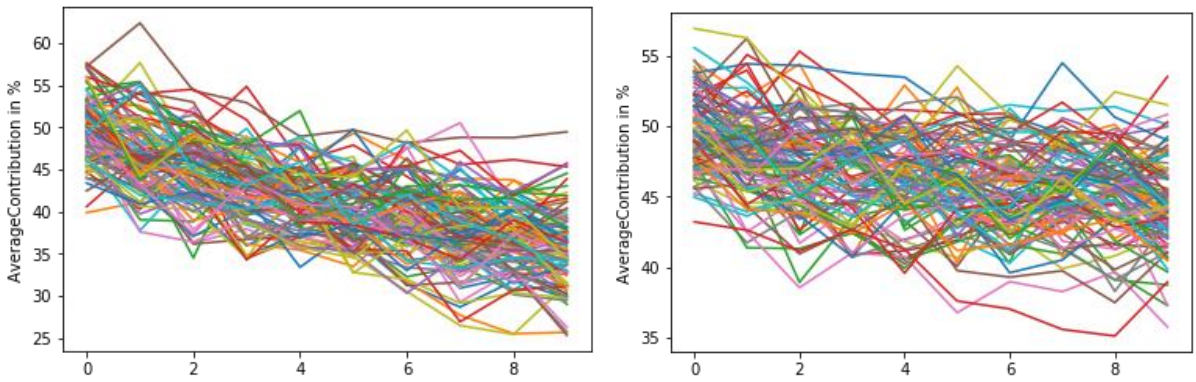
For the four experiments with the variables N = 60 or 100, rounds of play = 10, 120 money units, MPCR = 0.02 or 0.04 and parameter values of μ = 0, Ɛ = 0.05 and ϕ = 0.05 and initial Propensities of 1 for all plays the following results were observed: Graph 7 shows the average contribution for all 4 games. Graph 8 shows the free-riding percentage for all 4 games. Additionally Graphs 9-12 show a monte-Carlo simulation for the 4 games each with 10 iterations of the model.



Graph 7 & 8: Results for a single simulation of all four experiments



Graph 9 & 10: Results of a monte-Carlo simulation with 100 iterations for the experiments 60/0.02 (left) and 100/0.02 (right) without parameter optimization



Graph 11 & 12: Results of a monte-Carlo simulation with 100 iterations for the experiments 60/0.02 (left) and 100/0.02 (right) without parameter optimization

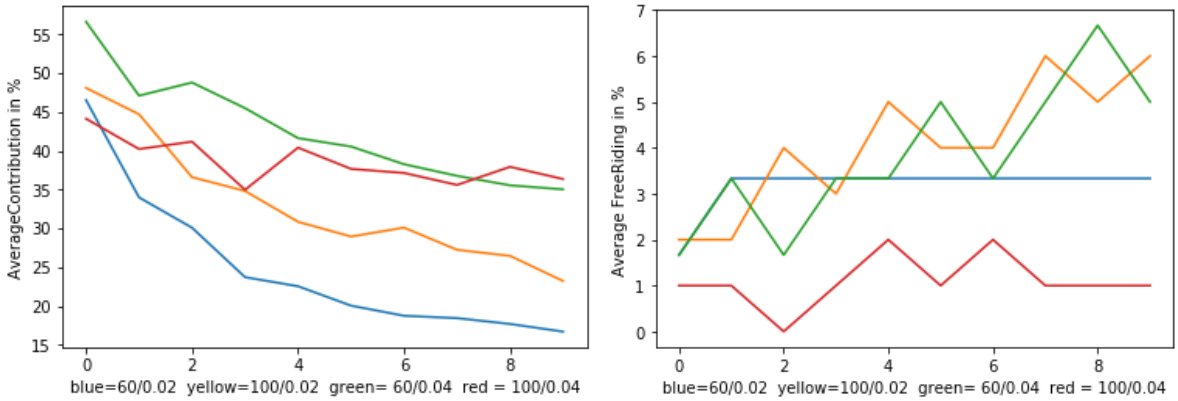
First, contrary to the test run, the model shows a learning behavior for the given variables and parameters for all 4 games. All four simulations show a decline in average contribution which means players in the game contribute less and less every round. While this can be counted as a success on its own, the model is still far away from describing the actual behavior of the experiments. Four main differences to Weimann et al.'s data can be observed. 1. The average contribution to the public good is to high in the first round and to high in the last round. Additionally, there is no difference in contribution for the first round between the models. They are all centered around 50% contribution.[3] 2. The average contribution is not declining monotonic but often has rounds in which the contribution is higher than the round before. Only over the whole period of the game, the average contribution is declining. A fact that does not occur in real-life public goods experiments. 3. The free riding behavior of the model is extremely different compared to Weimann et al.'s data. While Weimann et al. reports a certain percentage of players that already free ride in the first round, and a strong increase in free-riding over the course of the game, the free riding behavior is the model is nowhere close to that the model shows no increase in free-riding behavior over the game. Additionally, the free-riding behavior is much lower than in the experiments of Weimann et al.. What this shows is that the model only has a tendency to choose lower contribution plays, but the transition to free-riding seems to be longer than in real-life experiments. 4. The learning process of the model seems to slow down when the multiplier in the public good increases. The simulations with a higher public good multiplier have a flatter average contribution curve. The learning process seems to slow down. This is problematic since all of Weimann et al.'s experiments report a considerable decline in public good contribution over the course of the game. This means that the model in this initial state gets worse for games with higher public goods multipliers

Considering these 4 differences, the model results for the four experiments of Weimann et al. are similar to the real-life data, but still not good enough to actually talk of predictions. The second execution of the model will try to improve the model concerning these differences.

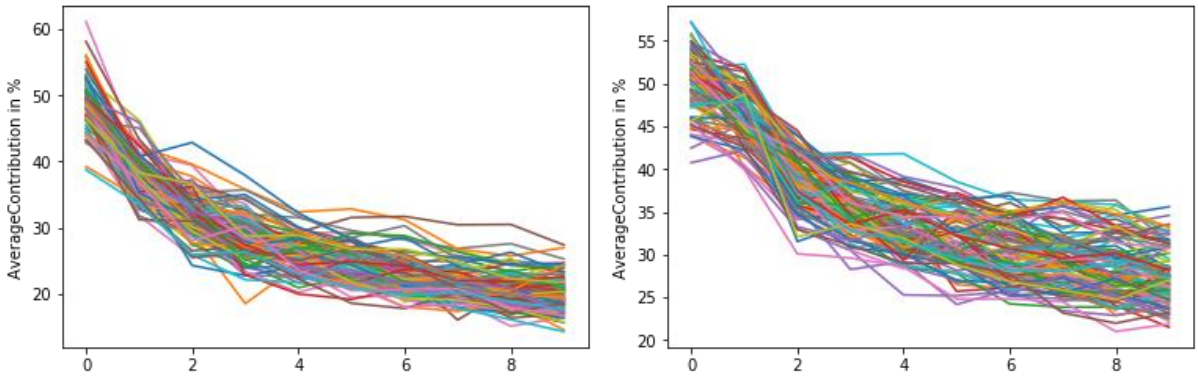
6.1.2 The second execution of the model

During the optimizing of the model parameters, various combinations were executed. The following configurations of parameters proved themselves as the most successful: For all 4 experiments, the best values for ε and ϕ were $\varepsilon = 0.05$ and $\phi = 0.04$. Noteworthy is that both parameters had no significant impact on the model when keeping them at values between 0 and 0.5. The initial propensity was set to 1.0 for the experiments 60/0.02, 100/0.02 and 60/0.04. For the experiment 100/0.04, an initial propensity of 3.0 lead to better results. Finally, μ was set to 0.007 for the 60/0.02 experiment, 0.004 for the 100/0.02 experiment, 0.002 for the 60/0.04 experiment and 0.0001 for the 100/0.04 experiment. μ was overserved as the parameter with the most positive effect on the model. Graph 13-18 show the results for these parameter settings. To answer to which extend the model was able to describe Weimann et al.'s data, it will be analyzed to which extend the four differences of the first run of the model were resolved.

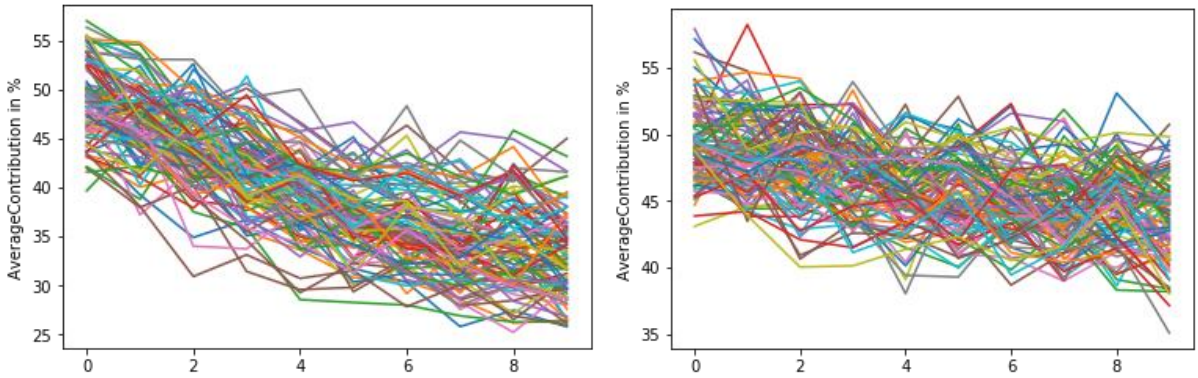
3. Since all initial propensities are equal, this should be no surprise.



Graph 13 & 14: Results of a single simulation for all four experiments with parameter optimization



Graph 15 & 16: Results of a monte-Carlo simulation with 100 iterations for the experiments 60/0.02 (left) and 100/0.02 (right) with parameter



Graph 17 & 18: Results of a monte-Carlo simulation with 100 iterations for the experiments 60/0.04 (left) and 100/0.04 (right)

Difference 1: Contribution levels:

Except for experiment four, the contribution levels were slightly lowered over the course of the game. Especially in experiment one and two, the average contribution has a similar trend to the real-life data. Despite that, the contribution levels for the first round are still at 50% which is higher compared to the real-life data for all experiments and especially for experiment one and two. Additionally, the contribution levels are still to high compared to Weimann et al.'s experiments and the contribution in the last round is way to high for all experiments.

Difference 2: monotonic declining graphs

For experiments three and four, no improvement was achieved. The graphs of the model still include many rounds with higher contributions than the round before. For the experiments one and two however, significant improvements can be observed. For experiment two. The graphs of the simulation show much less jumps. For experiment one, the average contribution is very close to monotonic declining with few exceptions (Graph 8). Experiment one and two therefore came close to predicting Weimann et al.'s experimental data.

Difference 3: free-riding behavior:

While graph ten shows that there was a slight improvement concerning the tendency towards more complete free-riding during the game, the increase and the level of complete-free riding are far off compared to Weimann et al.'s experiments. While Weimann et al.'s experiments report up to 65% free-riding in the last round of the game, the model produces not even 10% free-riding in the last round. Additionally, the graphs for Weimann et al.'s experiments are monotonic increasing. The graphs of the model are not.

Difference 4: Public good multiplier

The observation from the first run of the game was confirmed. While the fitting process improved especially experiment one, the improvements that where achieved decline with an increasing public good multiplier. For experiment four which has the highest multiplier (100*0.04) almost no improvements where achieved. And the predictions of the model remained very bad. Additionally, the range of possible parameter values were narrowed down as well. For many values, no learning behavior at all was observed. The public goods multiplier seems to be negatively correlated with the learning behavior of the model and restricts the effects of the additional parameters.

Concerning the results of the second run, it was shown that the model can reliably describe the two experiments of Weimann et al.'s with the lowest public good multiplier. For the experiment three and four, the results of the model are not that close to the real-life data. This can be translated to the model describing experiments with a low public good multiplier. For higher multiplier experiments however, the model failed to predict public goods experiments. Too many fundamental differences between model results and real-life data were observed (difference 1-4).

6.2 Comparison of the model with Isaac and Walker's experimental results

After the reinforcement learning model was compared the first dataset with not too much success, it is now time to compare the model to Isaac and Walker's experimental data. Isaac and Walker's study included 12 different experiments from which four are relevant for the purpose of this paper. Two experiments with 100 players and two experiments with 40 players with an MPCR of 0.3 or 0.75. The model will predict all 4 of these experiments. Since Isaac and Walker's paper does not include

information about the free-riding behavior during the course of the game, the basis of comparison will be the two graphs (graph 19 and 20) that show the average contribution of the players for games with a MPCR of 0.3 and 0.75. for different number of players. Sadly, these graphs also include the results for experiments with fewer players. These results have to be ignored. As before, some characteristics of these graphs are worth mentioning. While the contribution levels start under 50% for all four experiments, contrary to Weimann et al.'s results, the average comparison does not drop below 35% for the last round of the game. A result that is highly inconsistent with the game theoretic solution of the game. Another interesting fact is, that the curves are not strictly monotonic but include ups and downs. Another feature that is contrary to Weimann et al.'s experimental results. These characteristics and the graphs will again be used to compare it to the results of the reinforcement learning model. The comparison will again happen in the same two steps that were described before. The parameters that are available for the fit are still μ, ε, ϕ and the initial propensities. The question whether the reinforcement learning model can describe Isaac and Walker's real-life data will be answered based on this final comparison.

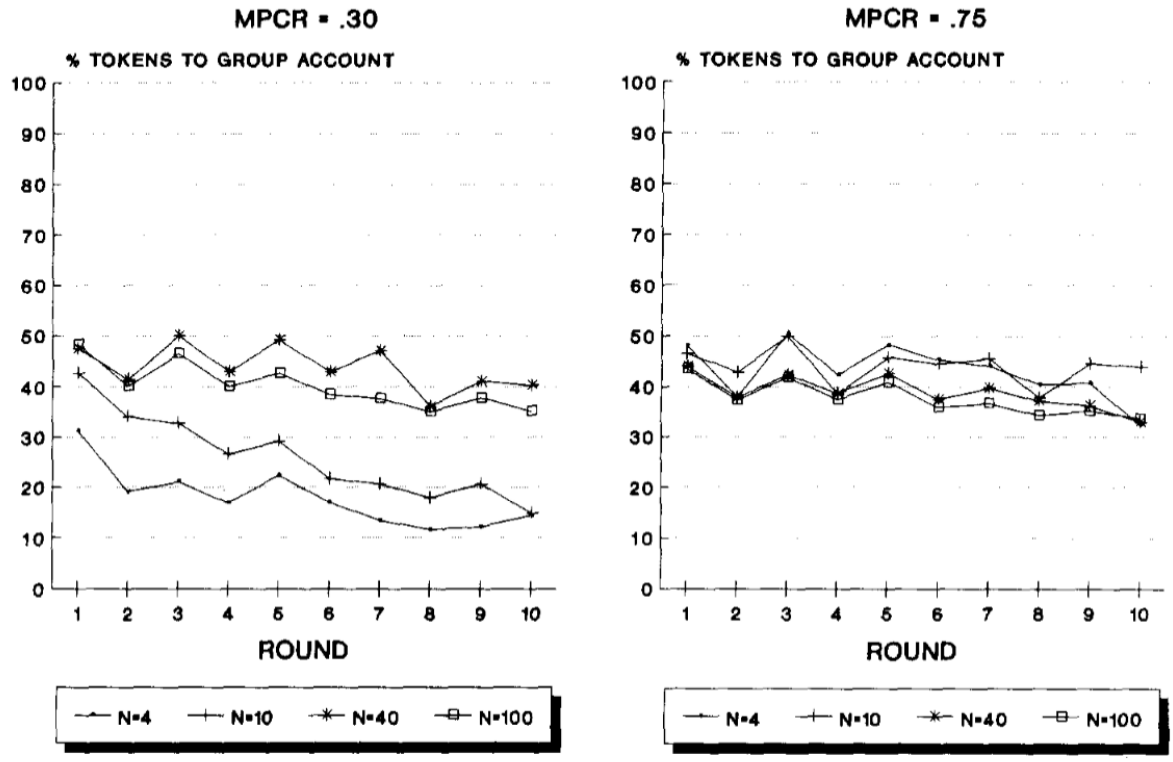
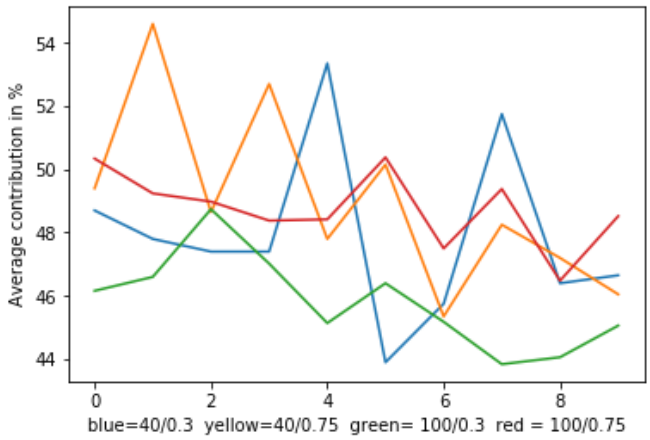


Fig. 6. Group size comparison for high (0.75) and low (0.30) *MPCR* cells.

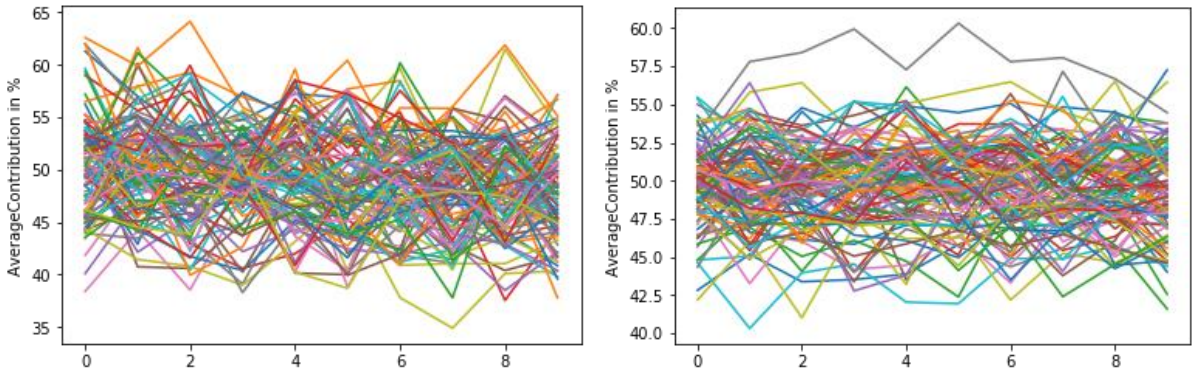
Graph 19 & 20: Isaac and Walker's results for average contribution in both experiments (Isaac and Walker,1999,p.4)

6.2.1 The first execution for Isaac and Walker's data

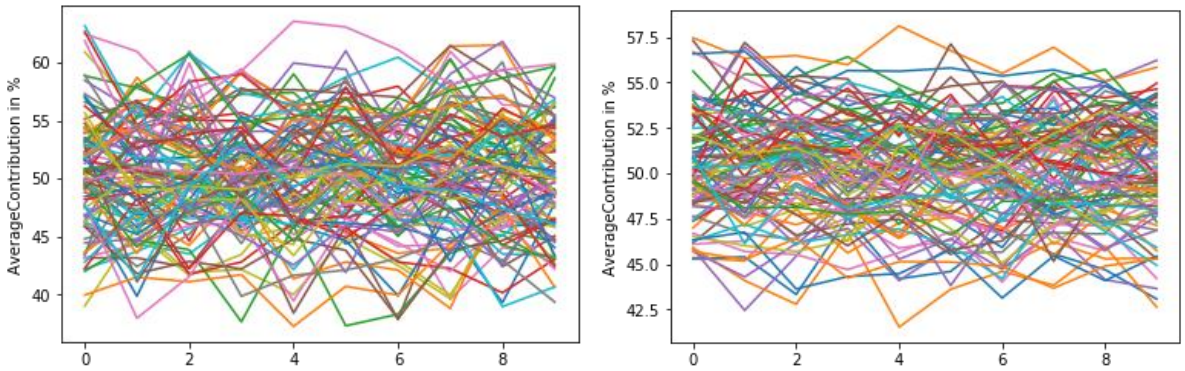
For the four experiments with the variables N = 40 or 100, rounds of play = 10, 50 money units, MPCR = 0.3 or 0.75 and parameter values of $\mu = 0$, $\varepsilon = 0.05$ and $\phi = 0.05$ and initial Propensities of 10 for all plays the following results were observed: Graph 15 shows the average contribution for all 4 games. Additionally Graphs 21-25 show a monte-Carlo simulation for the 4 games each with 10 iterations of the model.



Graph 21: Results for a single simulation of all four experiments



Graph 22 & 23: Results of a monte-Carlo simulation with 100 iterations
for the experiments 40/0.3 (left) and 100/0.3 (right)



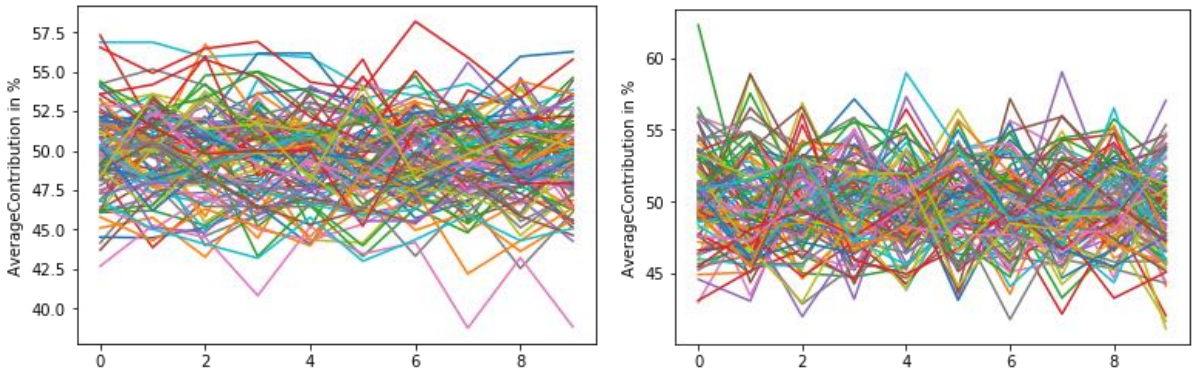
Graph 24 & 25: Results of a monte-Carlo simulation with 100 iterations
for the experiments 40/0.75 (left) and 100/0.75 (right)

Surprisingly, the model does not show any form of declining average contribution. This means the players in the model do not seem to learn about the game. The four monte-Carlo simulations show that there is no observable decline in average contribution for the model. This is the first and most important difference between model and real-life data since Isaac and Walker's experiments report a slight decline of average public good contribution for all four experiments. Additionally, the contribution levels of the first round are again on average around 50% which is higher than in Isaac and Walker's experiments. The model results in non-monotonic graphs which is equal to experiments. For free-riding behavior, no comparisons can be made. Noteworthy is the fact, that the public good multiplier is very high in these for experiments which is again correlated with an even lower decline of average contribution and an even lower learning rate (Or even no learning rate) compared to the 100/0.04 experiment of Weimann et al..

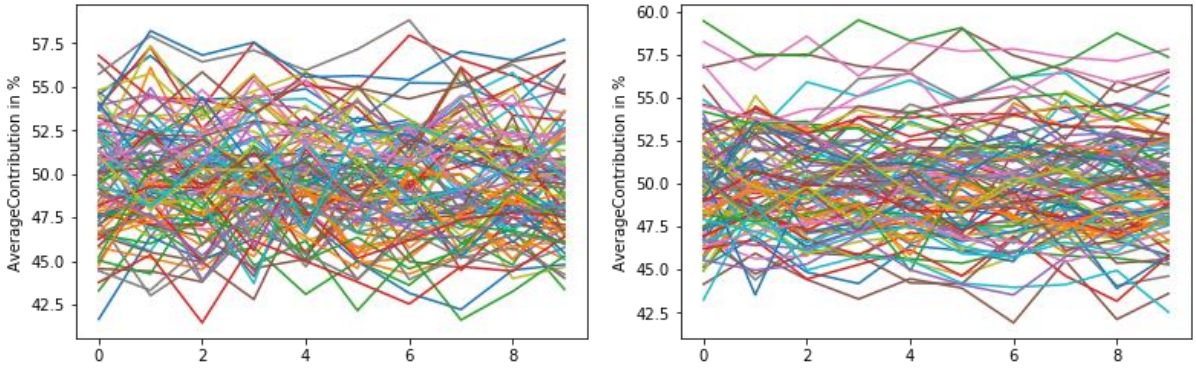
Based on the differences mentioned above, the model does not produce similar results to Isaac and Walker's experimental data and therefore has no predictive power for the real-life data of Isaac and Walker. The second run of the model will now try to improve the model concerning the differences and the predictive power.

6.2.2 The second execution of the model

During the fitting of the model parameters, various combinations were executed. Despite the effort, no improvement concerning differences mentioned above could be achieved. The average contribution for all experiment remained around 50% for the course of the game. Graphs 26-29 show the results for various parameter values which lead to very similar results for the 100/0.3 The three other experiments resulted in very similar outcomes with also no decline in average contribution and therefore no learning behavior observable. Since the results for Isaac and Walker's experimental data remain so bad even after the second execution of the model. The additional differences mentioned above remain. Conclusevly, the results of the model are nowhere close to the real-life experimental data of Isaac and Walker. The model therefore fails to predict Isaac and Walker's experiments completely.



Graph 26 & 27: Results of a monte-Carlo simulation with 100 iterations for the experiment 100/0.3 with $\mu = 0$, $\varepsilon = 0$, $\phi = 0$ and IP = 10 (left) and $\mu = 0.0001$, $\varepsilon = 0.05$, $\phi = 0$ and IP = 100 (right)



Graph 28 & 29: Results of a monte-Carlo simulation with 100 iterations for the experiment 100/0.3 with $\mu = 0.002$, $\varepsilon = 0$, $\phi = 0.05$ and IP = 10 (left) and $\mu = 0.007$, $\varepsilon = 0.05$, $\phi = 0.05$ and IP = 10

VII. Final results and concluding remarks

7.1 Why does learning slow down with higher public good multipliers?

During the simulation for the experiments of Weimann et al. and Isaac and Walker, it was observed, that the learning behavior in the model slows down significantly with higher public good multipliers. This is one of the main reasons why the model cannot describe the experimental data with high public good multipliers although it does a good job of describing low multiplier experiments. Humans always show a decline in average contribution over the game, this effect could not be simulated to a satisfactory level. Additionally, for very high public goods multiplier, no learning behavior at all was observed for the model which is puzzling at the least.[4] While the full explanation for this effect is exceeds the purpose of this paper, it is at least worth mentioning a thesis for the occurrence of this effect.

The first suggestion for this observation would probably be to say that the 10 rounds of the public goods games are not sufficient to overserving any learning behavior, but for longer periods, learning behavior should be observable. Interestingly this is not the case either. Graph 30 shows the simulation of a 100/0.3 game with 50 money units and with parameters $\mu = 0.007$, $\varepsilon = 0.05$ and $\phi = 0.05$ and initial Propensities of 10 (The same values as for graph 4) for 100 rounds. Even for such a longer period of time, no learning behavior was observed. Contrary to that, graph 31 shows the simulation of a 100/0.02 game for the exact same parameters. The first suggestion is therefore false. A more promising explanation for these observations is based on the effect that the public good multiplier has on the reinforcements of the players. For small public good multipliers, the reinforcements can differ greatly. Contrary to that, for very high multipliers, the reinforcements are often very similar. This effect is best described for the first round of the game, since the contribution is always normally distributed around 50% due to the even propensity stacks. In a game with 100 players and 50 money units the payoff for the first round is defined as:

4. *While the average contribution decline slows down for high public good multipliers as well, it does not disappear in the real-life experiments.*

N*Average contribution*Public good multiplier /N.

The following calculation can be made for contributing nothing or everything to the public good for different MPCRs:

***For MPCR = 0.*02**, contributing nothing results in (99*25+0)*2(MPCR*N)/100 - 0 = 49.5 while contributing everything results in (99*25+50)*2/100 -50 = 0.5.

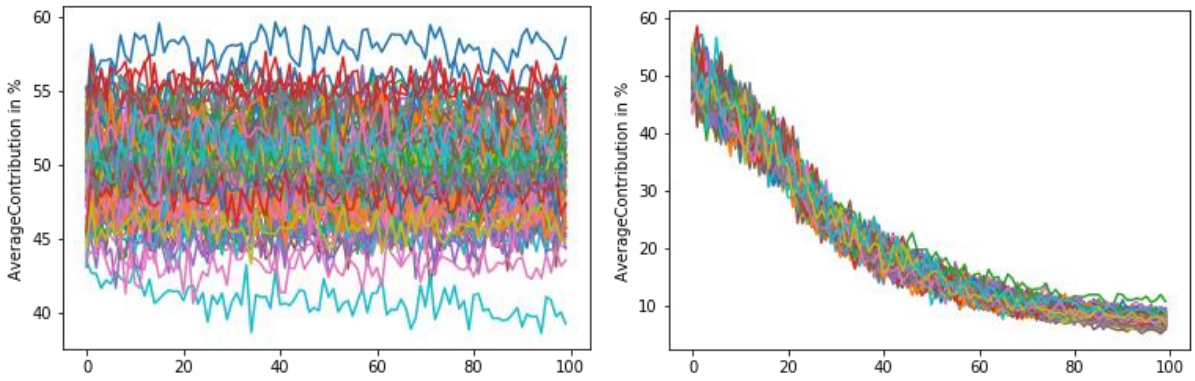
***For MPCR = 0.*3**, contributing nothing results in (99*25+0)*30(MPCR*N)/100 - 0 = 742.5 while contributing everything results in (99*25+50)*30/100 -50 = 707.5.

While the relative difference for a MPCR of 0.02 is striking, it is much lower for a MPCR of 0.3 which is a possible explanation for the slow down of the learning behavior. But how can one explain the complete absent of learning behavior for games with a high public good multiplier? Since the reinforcements for high multiplier games is always very positive and very close together, plays that were chosen at the beginning of the game will be chosen again with a relatively high probability no matter the play since all plays lead to a very high reward. This play than gets reinforced repeatedly since it gets chosen more often. This means no matter the initial play, the reinforcement learner keeps playing his initial play with some few exceptions which leads to no learning behavior.

When looking at the plays of a specific player in the 100/0.3 game described above and his propensity stack, this thesis seems to be supported. Picture 1 and 2 show the plays of a single player and his propensity stack after round 1 and 100. An objection that might be made based on these stats is, that

the initial Propensities are way to low compared to the reinforcement which. With higher initial Propensities this effect can be negated. This however has proven itself as incorrect. Two effect occur. With very high initial propensities, the difference between the plays gets relatively lower compared to the stack which negatively influences the learning process. Additionally, high initial propensities only lead to not one play being reinforced repeatedly, but multiple plays. This feature however seems only to lead to curves with higher fluctuations which still include no learning behavior.

While this there is some evidence in support of the thesis that the multiplier of the public good negatively influences the learning behavior of the models through the reinforcements for the players, additional research is needed to finally understand the correlation between multiplier of the public good and learning behavior.



Graph 30 & 31: Results of a monte-Carlo simulation with 100 iterations for the experiment 100/0.3 (left) and 100/0.02(right) with equal parameters

```
[   9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
    9.5         9.5         9.5         9.5         9.5         9.5
  26.626125 660.29275    26.626125]
```

Picture 1: Propensity Stack for a single player after the first round

```
[6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 6.23213602e-02
 6.23213602e-02 6.23213602e-02 6.23213602e-02 9.45050975e+00
 5.96928308e+02 9.18591707e+03 2.27219829e+03]
```

Picture 2: Propensity Stack for a single player after 100 rounds

7.2 Conclusion

After the model was executed for different datasets of real-life public goods games experiments, it can be concluded that the reinforcement learning model can reliably describe public goods experiments with a low public good multiplier. For high public goods multiplier however, the average contribution levels of the model are too different from the values of the real-life experiments in most of the cases. Additionally, the average free riding behavior of the model does not resemble the free-riding behavior of humans. Interestingly, there seems to be a connection between the decline of average contribution levels and therefore learning behavior and public goods multiplier. This connection needs however further investigation. While the graphic analysis that was done in this paper gives strong evidence for this final result, additional research that uses statistical methods for the comparison is needed to further support the results of this paper. Considering that this paper might seem to be rather divided concerning reinforcement learning and real-life public goods games, there is one factor, that should not be neglected. As Weimann et al. states: “Many of real–world public–goods are characterized by a marginal per capita return (MPCR) close to zero and have to be provided by large groups.” (Weimman et al.,2014, abstract). Examples like environmental protection or water supply are classic problems support such a claim. Concerning this, the reinforcement learning model tested in this paper might despite its restricted applicability be used to describe a number of real-life public goods games.

VIII. Sources

Börgers, Tilman, and Rajiv Sarin. "Learning through reinforcement and replicator dynamics." *Journal of Economic Theory* 77.1 (1997): 1-14.

Chaudhuri, Ananish. "Sustaining cooperation in laboratory public goods experiments: a selective survey of the literature." *Experimental Economics* 14.1 (2011): 47-83.

Cheung, Yin-Wong, and Daniel Friedman. "Individual learning in normal form games: Some laboratory results." *Games and Economic Behavior* 19.1 (1997): 46-76.

Dong, Yali, Boyu Zhang, and Yi Tao. "The dynamics of human behavior in the public goods game with institutional incentives." *Scientific reports* 6 (2016): 28809.

Durlauf, Steven N., and Lawrence E. Blume. "Learning and Evolution in Games: Belief Learning. " *Game Theory*. Palgrave Macmillan, London, 2010. 191-198.

Friedman, Daniel. "On economic applications of evolutionary game theory." *Journal of Evolutionary Economics* 8.1 (1998): 15-43.

Fudenberg, Drew, et al. *The theory of learning in games*. Vol. 2. MIT press, 1998.

Hauert, Christoph, et al. "Replicator dynamics for optional public good games." *Journal of Theoretical Biology* 218.2 (2002): 187-194.

Isaac, R. Mark, and James M. Walker. "Group size effects in public goods provision: The voluntary contributions mechanism." *The Quarterly Journal of Economics* 103.1 (1988): 179-199.

Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

Nyarko, Yaw, and Andrew Schotter. "An experimental study of belief learning using elicited beliefs." *Econometrica* 70.3 (2002): 971-1005.

Osborne, Martin J. *An introduction to game theory*. Vol. 3. No. 3. New York: Oxford university press, 2004.

Roth, Alvin E., and Ido Erev. "Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term." *Games and economic behavior* 8.1 (1995): 164-212.

Rouse, Margaret. “What Is Reinforcement Learning? - Definition from WhatIs.com.” *SearchEnterpriseAI*, searchenterpriseai.techtarget.com/definition/reinforcement-learning. Retrieved at 23.08.2018

Weibull, Jörgen W. *Evolutionary game theory*. MIT press, 1997.

Weimann, Joachim, et al. *Public-good experiments with large groups*. Univ., Fak. für Wirtschaftswiss., 2012.

IX. Appendix

How to open and use the code file for this paper:

1. Download the reinforcement learning model implementation from
https://github.com/Yondijr/Reinforcement-learning

2. Download and Install the Python Distribution “Anaconda” from
https://www.anaconda.com/download/

2. While installing click on “Add Anaconda to my PATH environment variable” to bypass errors.

3. Anaconda installs “Jupyter Notebook”, the program to open the code file, and all dependencies that were used.

4. Open “Jupyter Notebook”. It will show your files on your computer.

5. Now simply open the file “ReinforcementLearningModel_GideonStein.ipynb”.

6. You can now execute and edit the code file.