Morgan Keys
4/25/11
CSCS 530

# Modeling Social preference with simple games

## Abstract

The goal of my project is to create a population of agents whose individual behavior is an adaptation of a behavioral-economic model and examine how that behavior translates to large scale behavior. Based on a model by Charness and Rabin, I programmed agents to evaluate options based this model, and made conceptual assumptions where necessary. I compared single-run games (without updating) to experimental data to examine similarity. Results are mixed. For longer runs, reciprocity-updating appears to make some choices more universally accepted. I also looked at how theta-values change over time and found an interesting divergence in certain games. Other areas to extend this work are discussed.

## Introduction and Motivation

As a student specializing in Information Economics for Management (IEM) at the School of Information, my academic interest is in how economic, sociological, and psychological tools can be used to understand human interactions with information systems and online communities. We study game-theory, decision-science, and economics and apply theoretical models to understand the incentives which guide behavior and look for clues to influencing it.

As part of my IEM coursework, I recently took a class called User-Generated Content. The course examined online communities where the users themselves contribute the content which makes the site valuable. Major topics include public goods, exclusion and threshold mechanisms, crowd-sourcing, social identity, and social preference.

While studying social preference, I became interested in Charness and Rabin's "Understanding Social Preferences with Simple Tests". Their study described a mathematical model of decision-making which combined three competing theories. With it, they proposed a model for how an individual chooses among payoffs. Struck by the explanatory power of this model, I decided that I wanted to see how this model would actually play out if installed into agents.

The goal of my project is to create a population of agents whose individual behavior is an adaptation of Charness and Rabin's model and examine how that behavior translates to aggregate behavior. To extend this, I also want to see population tendencies change if the agents react to perceived treatment from others. Ideally, I would then also want to examine how different networks would affect the population's tendencies.

## Charness and Rabin's Social Preference Model

In their paper, they also described lab results from real life participants in multiple locations, using different game scenarios. They report results for 32 different game-types, averaging participant choices.

The behavior model put forth by Charness and Rabin [1] takes into account a sense of charity, "Behindness aversion", and a sense of reciprocity. These parameters come from three separate models of social preference.

The following is full utility function for a two-player game, as described by Charness and Rabin:

$$U_B(\pi_A,\pi_B) \equiv (\rho \cdot r + \sigma \cdot s + \theta \cdot q) \cdot \pi_A + (1 - \rho \cdot r - \sigma \cdot s - \theta \cdot q) \cdot \pi_B,$$

where

$r = 1$ if $\pi_B > \pi_A$, and $r = 0$ otherwise;
$s = 1$ if $\pi_B < \pi_A$, and $s = 0$ otherwise;
$q = -1$ if A has misbehaved, and $q = 0$ otherwise.

**Figure 1 From Charness and Rabin [1]**

Charity, which comes from the "social welfare" model, represents simply wanting others to do well. Essentially it is saying, "I like it (I gain utility) when I see others do well". Charity, $\rho$, is trigger by r which captures the scenario in which one is in a better position than their opponent.

"Behindness aversion" comes from the theory that people do not like to see a disparity of wealth between themselves and others. If one is behind, i.e. doing less well than, another person, one begins to dislike this. Even if one cares for another's well-being, too much of difference causes disutility. "Behindness aversion", $\sigma$, is triggered by s, which captures the scenario in which one is in a worse position than their opponent.

Reciprocity is modeled in a backward way. Reciprocity, $\theta$, is triggered by q. When q = -1, this is meant to represent a scenario in which one's opponent has misbehaved. Thus, misbehavior leads one to discount their other social preference parameters and act more like a strictly rational, self-interested agent.

Given this utility function, my goal was to evaluate options from the perspective of different agents, with different parameters, and then make choices in games based on those evaluations.

Charness and Rabin were also able to estimate population values for their parameters. The following table shows their estimates based on experimental data. Note that this table also illustrates the modularity of their model. By setting certain variables to 0 in different combinations, their model can be

used to capture different theories of behavior, including narrow self-interest and each of the three original social preference models.

TABLE VI
REGRESSION ESTIMATES FOR B BEHAVIOR ($N = 903$)

| Model | Restrictions | $\rho$ | $\sigma$ | $\theta$ | $\gamma$ | LL |
|---|---|---|---|---|---|---|
| Self-interest | $\rho = \sigma = \theta = 0$ | — | — | — | .004 | −593.4 |
| | | | | | (9.07) | |
| Single parameter—"altruism" | $\rho = \sigma, \theta = 0$ | .212 | .212 | — | .005 | −574.5 |
| | | (7.20) | (7.20) | | (8.65) | |
| Single parameter—"behindness aversion" | $\rho = \theta = 0$ | — | .118 | — | .004 | −591.5 |
| | | | (1.76) | | (8.53) | |
| Single parameter—"charity" | $\sigma = \theta = 0$ | .422 | — | — | .014 | −527.9 |
| | | (25.5) | | | (11.6) | |
| $\rho,\sigma$ model without reciprocity | $\theta = 0$ | .423 | −.014 | — | .014 | −527.7 |
| | | (25.5) | (−0.73) | | (11.6) | |
| "Reciprocal charity" | $\sigma = 0$ | .425 | — | −.089 | .015 | −523.7 |
| | | (27.9) | | (−2.98) | (11.3) | |
| $\rho,\sigma$ model with reciprocity | none | .424 | .023 | −.111 | .015 | −523.1 |
| | | (28.3) | (1.10) | (−3.11) | (11.6) | |

$t$-statistics are in parentheses. $\gamma$ is the precision parameter, and LL is the log-likelihood function. Games where A's entry is SWP-misbehavior are 1, 5, 11, 13, 22, 27, 28, 30, 31, and 32.

**Figure 2 From Charness and Rabin [1]**

Of the 32 games that Charness and Rabin tested, I decided to focus on a subset of these games. Twenty of the games that Charness and Rabin looked at were 2-person response games. Two-person response games involve an initial choice on the part of player A to opt-out or enter competition. If A opts out, the game is over and each player receives a payoff for that node. If A enters, player B has an opportunity to make a choice. The following figure diagrams the choices:

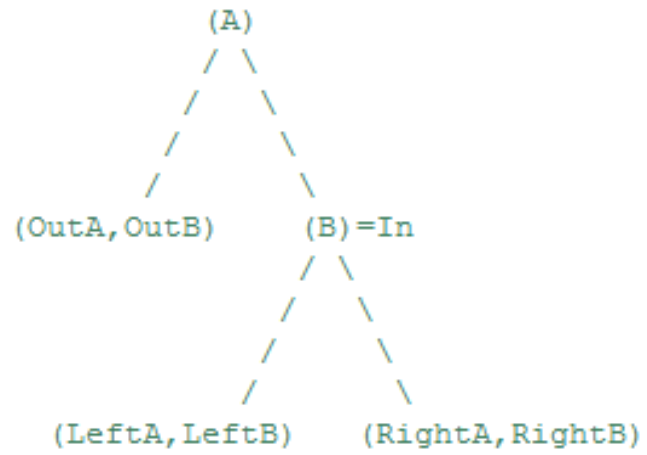



**Figure 3 A two-person response game**

Hence forth, “out” refers to A choosing to opt-out of competition, “in” refers to entering competition, “left” refers to B choosing the left hand choice in the diagram, and “right” refers to the right hand. The 20 games in Charness in Rabin present different dilemmas to participants and vary in severity—sometimes payoffs vary by hundreds, and sometimes very little.

| | | | | | |
|---|---|---|---|---|---|
| (1)Two person response games--B's payoff identical | | | | | |
| (4 games) | | | | | |
| 1-Barc7: | (750,0) | or | (400,400) | vs | (750,400) |
| 2-Barc5: | (550,550) | or | (400,400) | vs | (750,400) |
| 3-Berk28: | (100,1000) | or | (75,125) | vs | (125,125) |
| 4-Berk32: | (450,900) | or | (200,400) | vs | (400,400) |
| (2)Two-person response games--B's sacrifice helps A | | | | | |
| (11 games) | | | | | |
| 5-Barc3: | (725,0) | or | (400,400) | vs | (750,375) |
| 6-Barc4: | (800,0) | or | (400,400) | vs | (750,375) |
| 7-Berk21: | (750,0) | or | (400,400) | vs | (750,375) |
| 8-Barc6: | (750,100) | or | (300,600) | vs | (700,500) |
| 9-Barc9: | (450,0) | or | (350,450) | vs | (450,350) |
| 10-Berk25: | (450,0) | or | (350,450) | vs | (450,350) |
| 11-Berk19: | (700,200) | or | (200,700) | vs | (600,600) |
| 12-Berk14: | (800,0) | or | (0,800) | vs | (400,400) |
| 13-Barc1: | (550,550) | or | (400,400) | vs | (750,375) |
| 14-Berk13: | (550,550) | or | (400,400) | vs | (750,375) |
| 15-Berk18: | (0,800) | or | (0,800) | vs | (400,400) |
| (3)Two-person response games--B's sacrifice hurts A | | | | | |
| (5 games) | | | | | |
| 16-Barc11: | (375,1000) | or | (400,400) | vs | (350,350) |
| 17-Berk22: | (375,1000) | or | (400,400) | vs | (250,350) |
| 18-Berk27: | (500,500) | or | (800,200) | vs | (0,0) |
| 19-Berk31: | (750,750) | or | (800,200) | vs | (0,0) |
| 20-Berk30: | (400,1200) | or | (400,200) | vs | (0,0) |

**Figure 4 Game-types used in Charness and Rabin [1]**

In order to make for a more dynamic, more interesting model, I decided to add a learning element to these response games. That is, depending on how other players treat them, players will update their social preferences. For example, if another player treats them poorly, the will reduce their tendency for reciprocity. If they are treated well, they will increase their tendency for reciprocity.

## Methods

The following sections describe how my model builds and pairs agents and how it records game data.

### Command line parameters

When run from the command line, my model has two primary modes, and as many as 10 arguments. If no arguments are given, the model will default to a single-round with 1000 agents and 500 games. The default game type is Game #3: (100,1000) or (75,125) vs (125,125).

The first possible parameter is a string, either "settypes" or "normal". The string "settypes" allows the user to explicitly determine the distribution of agent parameters based on the types (theoretical models) as seen in Figure 2. The full parameter map for "settypes" is (Run_Mode="Settypes"):

[Run_Mode Number_of_rounds Game_Type

[#type1 #type2 #type3 #type4 #type5 #type6 #type7]]

When Run_Mode is set to "normal", the model allows users to specify the population by Normal distributions. The full parameter map for "default" is:

[Run_Mode Number_of_Agents Number_of_rounds Game_Type

[rhoMean rhoSD sigmaMean sigmaSD thetaMean thetaSD]]

### Creating agents

Depending on the Run_Mode parameter, the model will generate agents according to a specified distribution. In "settypes" mode, the model will create precisely the number of agents specified for each agent type. Agent parameters will be determined according to the regression estimates found in Charness and Rabin (see Figure 2). For example,

"java -jar model.jar settypes 1 10 0 0 0 0 0 100".

This would play one round with 10 purely self-interested agents and 100 full-model agents.

In "normal" mode, the model will create the specified number of agents with parameters randomly chosen from distributions defined by the given parameters. For example,

"java -jar Model.jar default 1000 1 7 .424 .1 .023 .1 .111 .1".

This would play 1000 agents for 1 round, in Game_Type 7 with a mean $\rho$-value of .424 and standard-deviation of .1, and so on.

Agents are created with the Agent class. Agent objects record an ID number and the three social preference parameters. Agent class methods include the computePref() method, which calculates the

agent's utility based on the full social preference model. The Agent class also has a tossUp() method, which returns a uniform result to help the agent choose when evaluations come back identical, and an update() method which adjusts the agent's social preference parameters based on inputs.

### Creating games and pairing agents

Games are created using a Game class. The game class defines the logic and procedure of a game. The Game class takes two agents, a round-number, and a game-type as inputs. The game-type is used to access a simple class called GameType, which returns a series of integers which represents a payoff scheme from one of the 20 response games used by Charness and Rabin [1].

Agents are paired two-by-two from an ArrayList of Agents. However, the list is shuffled using Repast utilities at every step, effectively making the pairings random.

### Decision making

For each game, a Game object calls computePref() for both users and predetermines the value of all payoffs assuming no misbehavior. The Game object then checks to see which options have the highest value for A. If A opts out, the Game compares B's preferences to determine whether or not A has misbehaved. Essentially the Game recomputes options from the point of view of B. If B would have chosen differently, that is considered misbehavior.

Lastly, the Game computes B's decision, and then evaluates misbehavior the same way as before from the perspective of A. If misbehavior is detected, A is updated.

If ambiguity is detected at any point, the player's tossUp() method is used to get a random choice. In the case of misbehavior, if it is ambiguous whether or not the opponent has misbehaved, the Game defaults to assume no misbehavior.

### Agent updating

By convention, Agent parameters are bounded by 0 and 1, inclusive. If an agent's parameter meets or exceeds these thresholds, they default to that extreme value for the update. When an agent's update method is called, they randomly increase or decrease their reciprocity parameter by a value between 0 and .1.

### Output files

Using a Report class, the model tracks game history and agent parameters to create 3 different output files. Each file is time-stamped and includes the game-type associated that agents played.

<u>All rounds</u>

This is history of each round; displays how many agents chose what options for each round.

<u>Agent history</u>

This is a history of each agent's reciprocity parameter. For each round, the file has a record of how the agent's parameter changed.

Toss-ups

This a list of every time a Game had to call the tossUp() method in order to handle an ambiguous situation. It lists the context and result of each toss-up situation.

### Console output

The Model also outputs a summary to the console. For each run it gives a decimal ratio of Ins, Outs, Lefts, and Rights. This can be compared directly with the tables from Charness and Rabin.

## How to run the model

I have uploaded the model as an executable JAR file. Along with this, I have included batch (.bat) files which will run scenarios of interest. Executing the JAR file directly will run the model under default conditions.

The model can be run from the command line with a command of the form:

> "java –jar [file.jar] [parameters]"

Each run of the program should produce a simple console summary and 3 text files.

## Results

Due to the exploratory nature of my model, there are many different ways to tune parameters and examine results. I have focused on two aspects that I thought showed interesting results.

### Game-sweeps and interaction plots

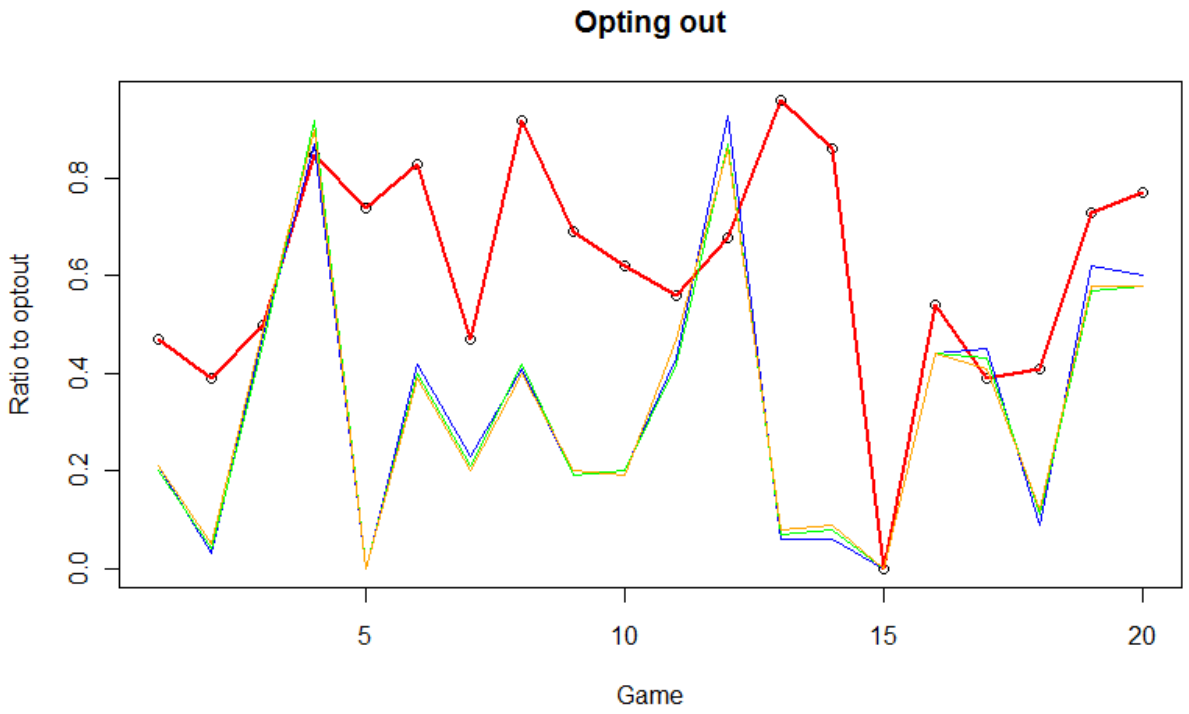
For my first round of analyses, I decided to sweep through every possible game-type, vary the number of rounds that agents played, and compare those results to the observed results of Charness and Rabin. For simplicity, I always ran agents with the

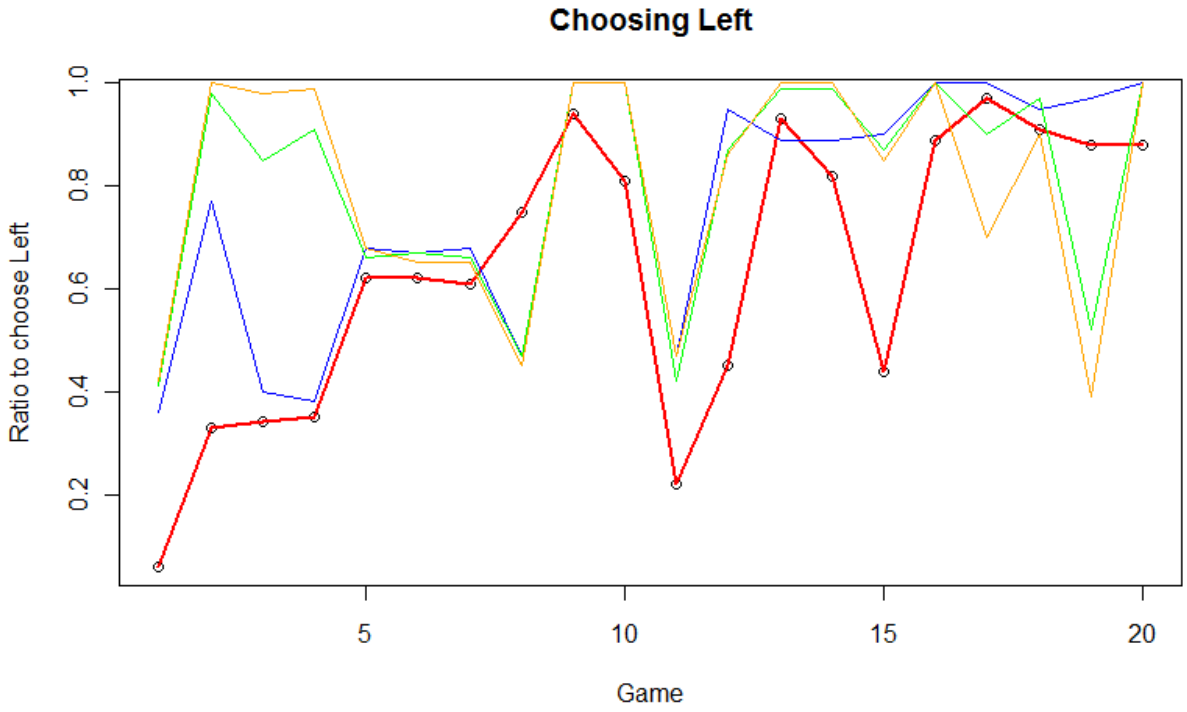
I ran sweeps for a single round, 100 rounds, and 100 rounds. Each sweep generated 1000 agents, playing 500 games. Recall that, since agents only update their parameters after a game, and they only play 1 game per round, a single-round run effectively show how the models run without any reactivity to other agents—that is, static parameters. Thus, this run is the closest conceptually to Charness and Rabin's experimental setup. Sweeps with 100 and 100 rounds should illustrate the effects that agent reactivity has on final outcomes.

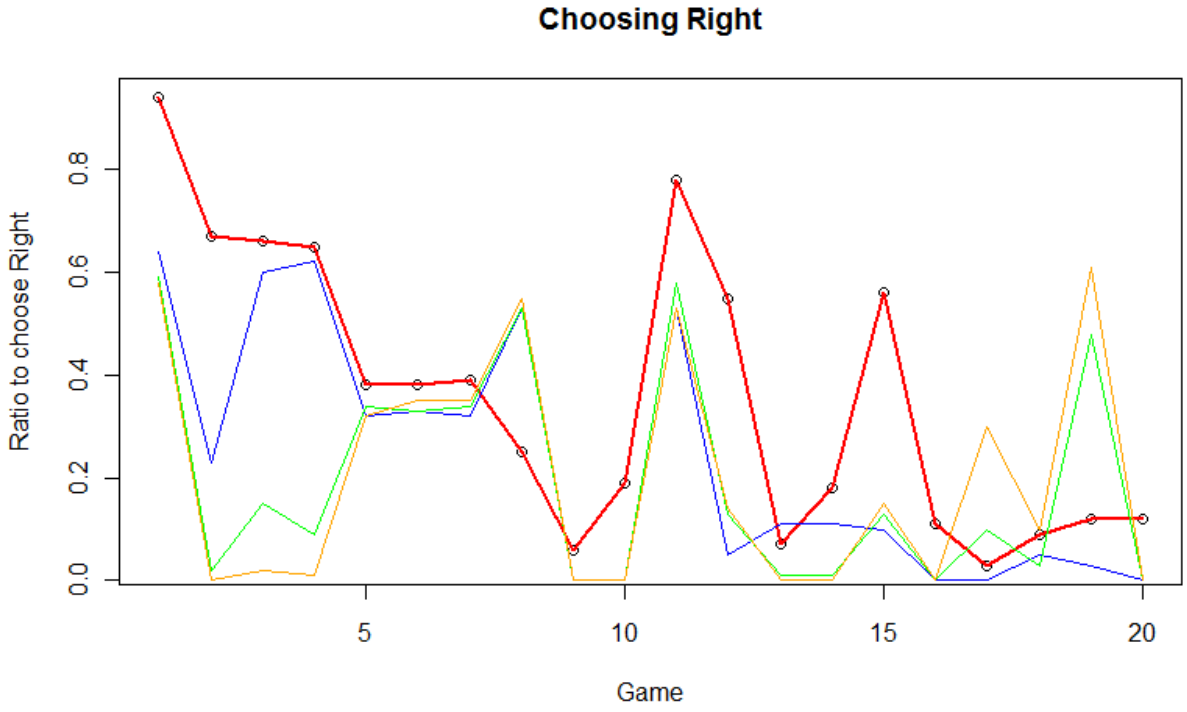
Below I have combined results into a single table. Results are given as a decimal ratio.

| | Sweep games, single round | | | | Sweep games, 100 rounds | | | | Sweep games, 1000 rounds | | | | Charness and Rabin observed results | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Game | Out | In | Left | Right | Out | In | Left | Right | Out | In | Left | Right | Out | In | Left | Right |
| 1 | 0.21 | 0.79 | 0.36 | 0.64 | 0.2 | 0.8 | 0.41 | 0.59 | 0.21 | 0.79 | 0.42 | 0.58 | 0.47 | 0.53 | 0.06 | 0.94 |
| 2 | 0.03 | 0.97 | 0.77 | 0.23 | 0.04 | 0.96 | 0.98 | 0.02 | 0.05 | 0.95 | 1 | 0 | 0.39 | 0.61 | 0.33 | 0.67 |
| 3 | 0.48 | 0.52 | 0.4 | 0.6 | 0.46 | 0.54 | 0.85 | 0.15 | 0.49 | 0.51 | 0.98 | 0.02 | 0.5 | 0.5 | 0.34 | 0.66 |
| 4 | 0.87 | 0.13 | 0.38 | 0.62 | 0.92 | 0.08 | 0.91 | 0.09 | 0.9 | 0.1 | 0.99 | 0.01 | 0.85 | 0.15 | 0.35 | 0.65 |
| 5 | 0 | 1 | 0.68 | 0.32 | 0 | 1 | 0.66 | 0.34 | 0 | 1 | 0.68 | 0.32 | 0.74 | 0.26 | 0.62 | 0.38 |
| 6 | 0.42 | 0.58 | 0.67 | 0.33 | 0.4 | 0.6 | 0.67 | 0.33 | 0.39 | 0.61 | 0.65 | 0.35 | 0.83 | 0.17 | 0.62 | 0.38 |
| 7 | 0.23 | 0.77 | 0.68 | 0.32 | 0.21 | 0.79 | 0.66 | 0.34 | 0.2 | 0.8 | 0.65 | 0.35 | 0.47 | 0.53 | 0.61 | 0.39 |
| 8 | 0.41 | 0.59 | 0.47 | 0.53 | 0.42 | 0.58 | 0.47 | 0.53 | 0.4 | 0.6 | 0.45 | 0.55 | 0.92 | 0.08 | 0.75 | 0.25 |
| 9 | 0.19 | 0.81 | 1 | 0 | 0.19 | 0.81 | 1 | 0 | 0.2 | 0.8 | 1 | 0 | 0.69 | 0.31 | 0.94 | 0.06 |
| 10 | 0.2 | 0.8 | 1 | 0 | 0.2 | 0.8 | 1 | 0 | 0.19 | 0.81 | 1 | 0 | 0.62 | 0.38 | 0.81 | 0.19 |
| 11 | 0.43 | 0.57 | 0.47 | 0.53 | 0.42 | 0.58 | 0.42 | 0.58 | 0.47 | 0.53 | 0.47 | 0.53 | 0.56 | 0.44 | 0.22 | 0.78 |
| 12 | 0.93 | 0.07 | 0.95 | 0.05 | 0.87 | 0.13 | 0.87 | 0.13 | 0.86 | 0.14 | 0.86 | 0.14 | 0.68 | 0.32 | 0.45 | 0.55 |
| 13 | 0.06 | 0.94 | 0.89 | 0.11 | 0.07 | 0.93 | 0.99 | 0.01 | 0.08 | 0.92 | 1 | 0 | 0.96 | 0.04 | 0.93 | 0.07 |
| 14 | 0.06 | 0.94 | 0.89 | 0.11 | 0.08 | 0.92 | 0.99 | 0.01 | 0.09 | 0.91 | 1 | 0 | 0.86 | 0.14 | 0.82 | 0.18 |
| 15 | 0 | 1 | 0.9 | 0.1 | 0 | 1 | 0.87 | 0.13 | 0 | 1 | 0.85 | 0.15 | 0 | 1 | 0.44 | 0.56 |
| 16 | 0.44 | 0.56 | 1 | 0 | 0.44 | 0.56 | 1 | 0 | 0.44 | 0.56 | 1 | 0 | 0.54 | 0.46 | 0.89 | 0.11 |
| 17 | 0.45 | 0.55 | 1 | 0 | 0.43 | 0.57 | 0.9 | 0.1 | 0.41 | 0.59 | 0.7 | 0.3 | 0.39 | 0.61 | 0.97 | 0.03 |
| 18 | 0.09 | 0.91 | 0.95 | 0.05 | 0.11 | 0.89 | 0.97 | 0.03 | 0.12 | 0.88 | 0.9 | 0.1 | 0.41 | 0.59 | 0.91 | 0.09 |
| 19 | 0.62 | 0.38 | 0.97 | 0.03 | 0.57 | 0.43 | 0.52 | 0.48 | 0.58 | 0.42 | 0.39 | 0.61 | 0.73 | 0.27 | 0.88 | 0.12 |
| 20 | 0.6 | 0.4 | 1 | 0 | 0.58 | 0.42 | 1 | 0 | 0.58 | 0.42 | 1 | 0 | 0.77 | 0.23 | 0.88 | 0.12 |

I have also made interaction plots, which illustrate where my model deviates from experimental data. For the following graphs, the red lines indicate Charness and Rabin's data, blue line indicate the single-round run, green lines indicate the 100-round run, and orange lines indicate the 1000-round run. Nodes represent specific data-points.




Choosing Left
Ratio to choose Left
0.2
0.4
0.6
0.8
1.0
5
10
15
20
Game

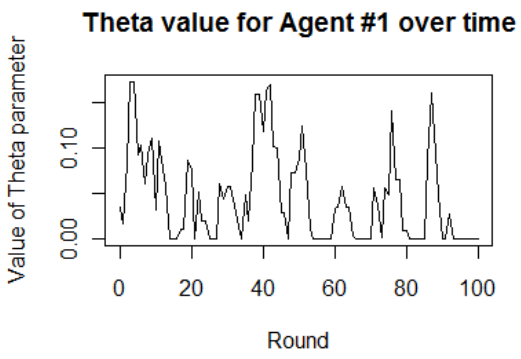

Choosing Right
Ratio to choose Right
0.0
0.2
0.4
0.6
0.8
5
10
15
20
Game

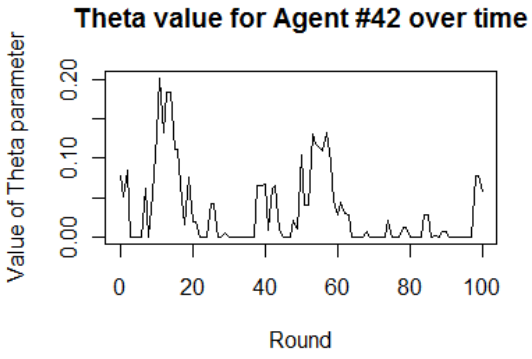
We see here, that while there are interesting areas of agreement, the model yields quite different results, showing significant interaction among data-sets. Particular areas of apparent overlap include Games 5-7 and game 18.

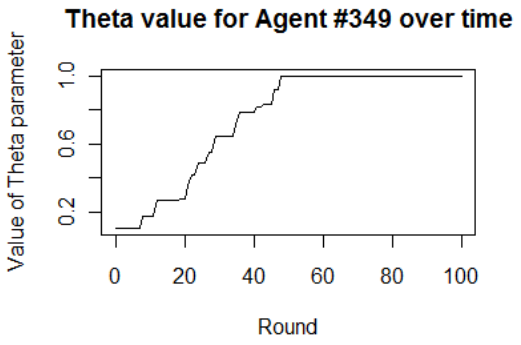
Also interesting is how the reciprocity changes affect the graph. The longer that we run the model, the more opportunity there is for theta to change. While see no difference in rate of opting-out (this is because player A always assumes good behavior at first, and hence reciprocity will never come into play), we do see cases where outcomes become more extreme in the second stage, particularly in games 4 and 18. Some options appear dominated as well, with near universal agreement after time in games 2, 4, 9, 10, 13, and 14. Note that games 9 and 10 have 0 as a possible payoff for Player A—zero-payoffs have a strong affect on outcomes.

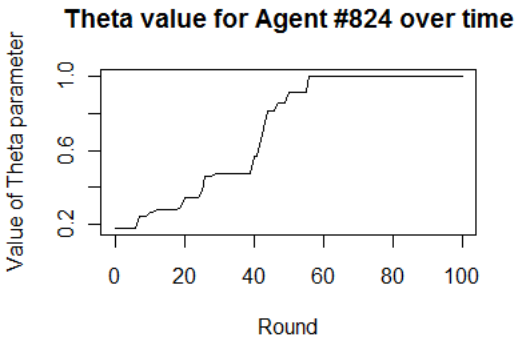
### Game-sweeps and evolution of Theta parameters

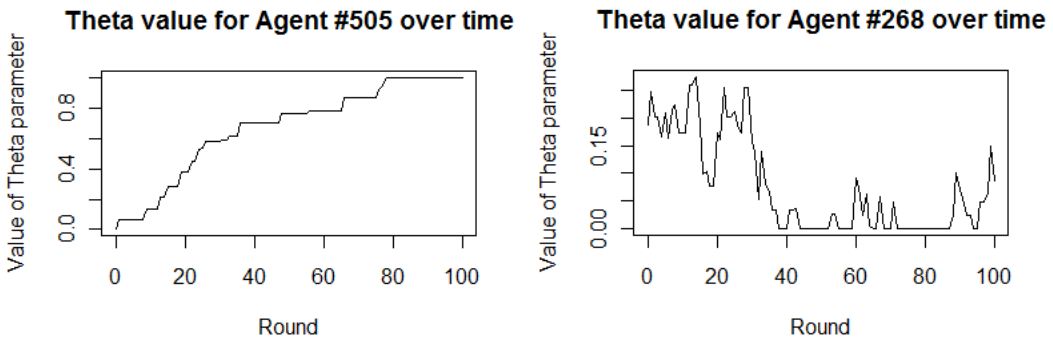
I thought would also be interesting to see how agent's theta values change over time. Below are the theta-value histories for 6 randomly chosen agents during the 100 round run of game 19. The most fascinating thing to me is how agents seem to have diverged into different types. Some agents have fairly random variation, with small spikes. However, there are others who's reciprocity value increases monotonically. It is important to note that reciprocity works inversely here—an increasing theta value means that the agent punishes wrong-doers with greater severity.



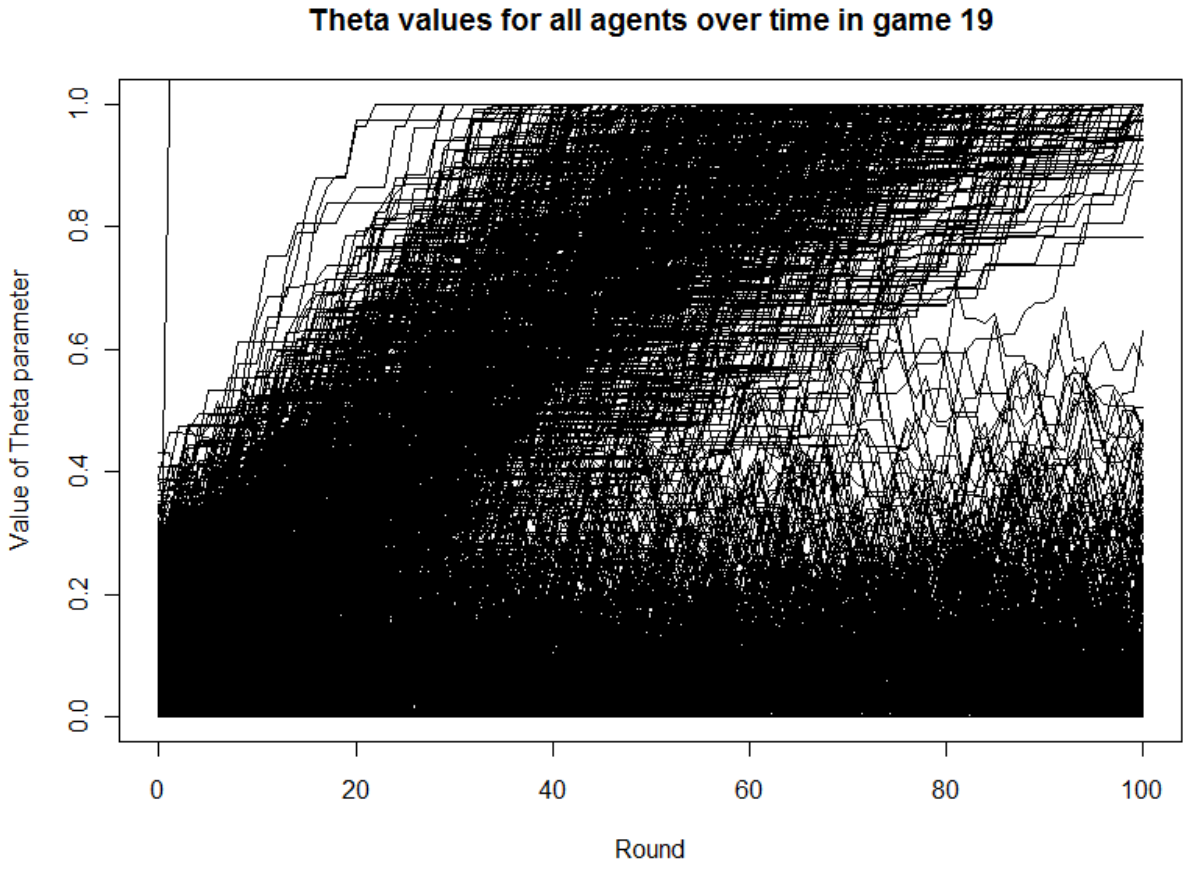



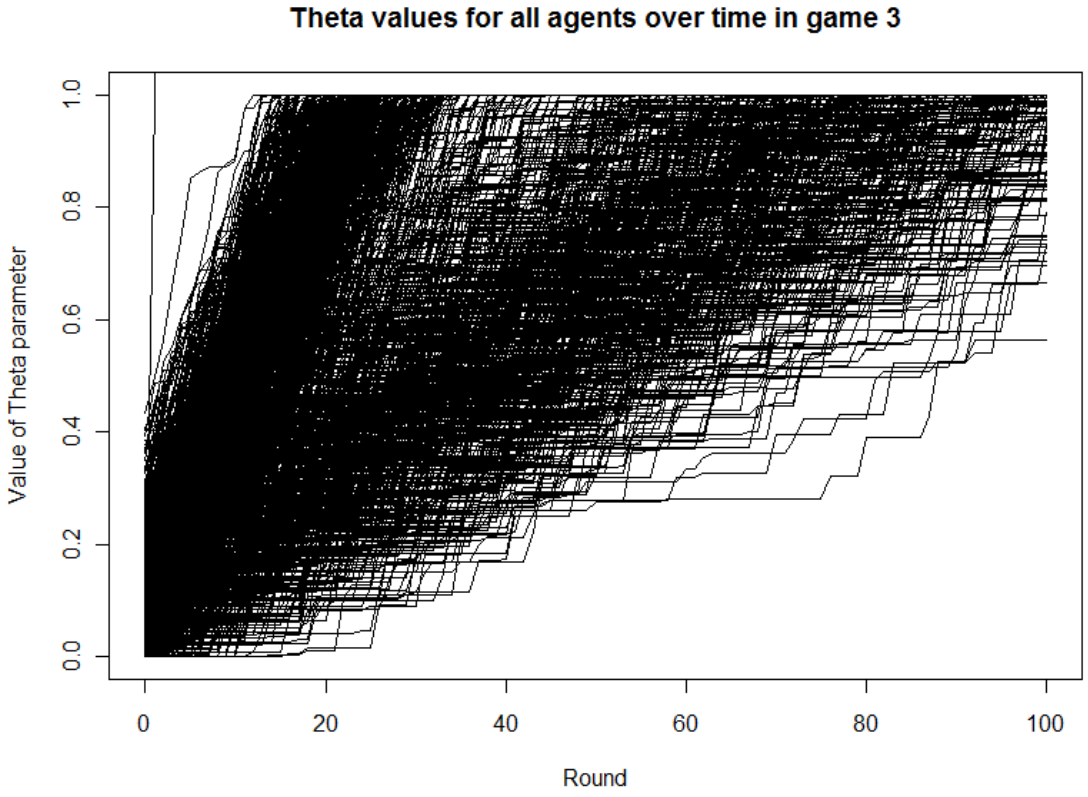



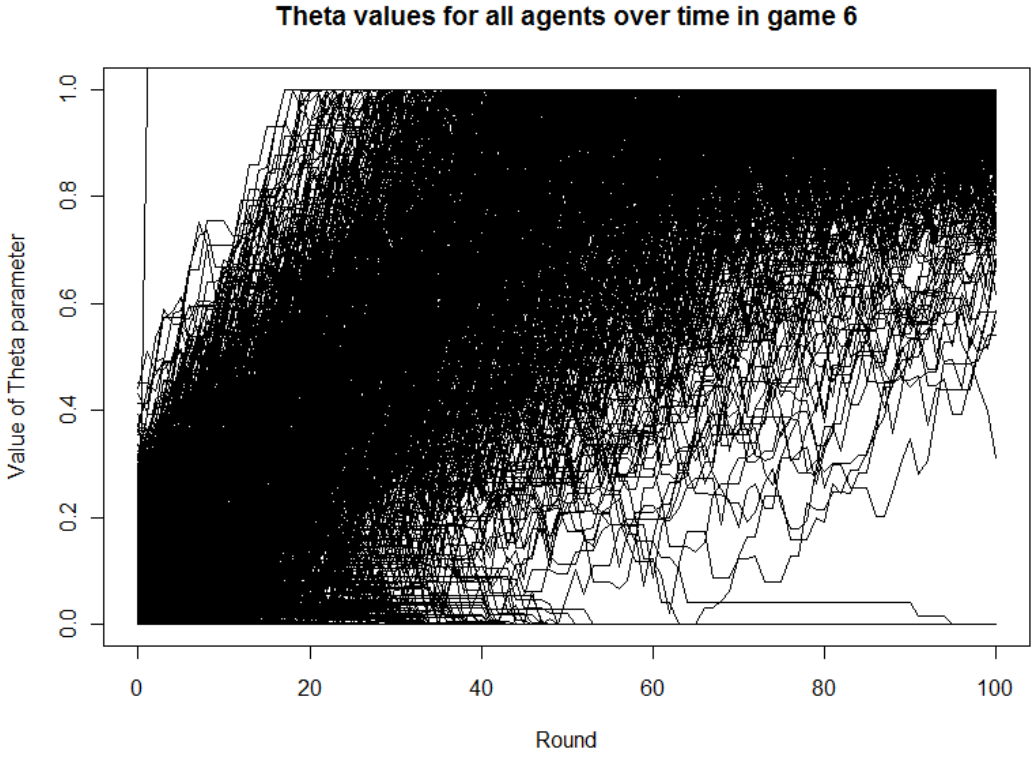



To illustrate this agent-divergence further, below is a graph of all agents for game 19:




For comparison, the same graphs for games 3 and 6:


Theta values for all agents over time in game 3
Value of Theta parameter
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
Round


Theta values for all agents over time in game 6
Value of Theta parameter
0.0
0.2
0.4
0.6
0.8
1.0
0
20
40
60
80
100
Round

We see divergence again in game 3 but to a lesser degree. There is not apparent divergence in game 6.

## Discussion/Conclusion

### Interaction graphs

I had hoped to produce a better fit to experimental data, however, considering the liberal and sometimes arbitrary assumptions I had to make in order to install Charness and Rabin's into and agent-based model, it is perhaps interesting that there is agreement at all. It might be interesting in further work to compare how games in which our models agree compare to models in which we diverge. For example, it seemed to me that zero-value payoffs could have sometimes resulted in edge-cases that yielded stronger reactions than a human would. Finding areas where game-types play a significant role could shed light on areas for adaptation in Charness in Rabin's model.

I was also disappointed that the reciprocity-reactions did not produce as interesting results as I had hoped. I think this is largely due to the fact that reciprocity, the only parameter that I updated, only came into play in certain cases. If A's incentive are aligned with B's, reciprocity may never be used, and agents will remain static. Perhaps this is actually conceptually correct—I lack the depth in understanding of human behavior to rightly assess this—but is a possible area for improvement. For example, maybe other social preference parameters should change as well? And maybe parameters should change as a function of payoff received, as oppose to by random increment?

### Other aspects to examine

I wanted to try different distributions of parameters to see if I could find different fits for the experimental data. While Charness and Rabin found regressions across their entire participant population, it seems possible that these could be a poor characterization of actual population behavior. For example, it is obvious that these value are not universal—otherwise all agents would make exactly the same decisions. For the purposes of this model, I assumed normal distributions, and arbitrarily chose standard deviations. Even then, there is the possibility of multimodal populations. Obviously, there is a lot of room to better fit this model to experimental data.

Due to a lack of time and ability to handle massive amounts of data, I did not pursue this further. Ideally, better fitting to data could be a very interesting and informative pursuit. Examining heterogeneous populations is the biggest advantage that agent-based modeling would have over charness and Rabin's model.

I had also wanted to examine how different distributions of types (as defined through the "settypes" mode) would compare with experimental data and my previous runs. Unfortunately, as of this writing, the "settypes" mode does not work due to a bug. I have been unable to correct this.

Lastly, a huge area that would have been interesting to example would have been how network effects changed choice tendencies. As is, the model essentially places agents in a fully-connected graph. It also

be interesting to see how populations would change with a grid network, networks with a high-degree of triads, or other topologies.

Works Cited

[1] G. Charness and M. Rabin, "Understanding Social Preferences with Simple Tests," *Quarterly Journal of Economics*, vol. 117, Aug. 2002, pp. 817-869.