DIVISION OF THE HUMANITIES AND SOCIAL SCIENCES
CALIFORNIA INSTITUTE OF TECHNOLOGY
PASADENA, CALIFORNIA 91125

DESIGNING EXPERIMENTS WITH COMPUTATIONAL TESTBEDS: EFFECTS OF CONVERGENCE SPEED IN COORDINATION GAMES

Noah Myung

Julian Romero

SOCIAL SCIENCE WORKING PAPER 1307

June 2009

Designing Experiments with Computational Testbeds: Effects of Convergence Speed in Coordination Games

Noah Myung Julian Romero

Abstract

Using a computational testbed, we theoretically predict and experimentally show that in the minimum effort coordination game, as the cost of effort increases, 1) the game converges to lower effort levels, 2) the convergence speed increases, and 3) the average payoff is not monotonically decreasing. In fact, the average profit is an U-shaped curve as a function of cost. Therefore, contrary to the intuition, one can obtain a higher average profit by increasing the cost of effort.

JEL classification numbers: C63, C73, C91, C92.

Key words: Coordination Game. Computational Testbed. Experiments.

Designing Experiments with Computational Testbeds: Effects of Convergence Speed in Coordination Games

Noah Myung* Julian Romero†

1 Introduction

This paper makes two contributions: First, we use a computational testbed to determine the experimental parameters. Testbeds are useful because they allow us to run many simulations over wide range of parameters very cheaply instead of experimenting with the parameters via pilot sessions. Second, we run the actual experiment using subjects in a laboratory setting to test the predictions made using our testbed. We make the following three predictions using the testbed which are indeed confirmed by subjects in experimental laboratory.

In a minimum effort coordination game, increasing the cost of effort causes:

1. The game converge to a lower minimum effort.
2. An increase in convergence speed to an equilibrium.
3. A non-monotonic change in average payoffs.

The intuition behind the results of the minimum effort coordination game is that there are both negative and positive effects on welfare as cost increases. We use the standard notion of welfare; total surplus or in our case, average payoff. The negative effects of higher cost are in two parts. First, lower payoffs are obtained from the same strategy profile for a higher cost in effort. In addition, the game also converges to a lower effort, which causes a lower payoff in general. The positive effects of higher costs is the faster rate of convergence to an equilibrium. Therefore, there is less wasted effort from agents searching for the equilibrium to converge. In sum, the average payoff increases

*Email: noah.myung@caltech.edu. Phone: 626-395-8772. Web: www.hss.caltech.edu/∼noah and at the Naval Postgraduate School's Graduate School of Business and Public Policy starting Fall 2009.

†Email: jnr@hss.caltech.edu. Phone: 626-395-3142. Web: www.hss.caltech.edu/∼jnr

if positive effects outweigh the negative effects, while the average payoff decreases if the negative effects outweigh the positive effects.

One reason why we implement computation testbed is the general difficulty in determining the specific parameters to use for the results stated above. One may guess and run many pilot sessions to guess the exact parameters but this can be a costly procedure. We propose that a computational testbed, which is often used in other areas of experimental science such as chemistry, offers an alternative solution to this problem.

1.1 Agenda

We first start with the theory section where we introduce the minimum effort coordination game, computational testbed, and our theoretical predictions. The details of the computational testbed and its algorithm are provided in the Appendix. Next, we proceed with testable hypotheses and our experimental design. We then provide the experimental results and concluding remarks.

2 Theory

2.1 Minimum Effort Coordination Game

Minimum effort coordination game, also known as a weakest-link game, takes the following form: Given N agents, every agent chooses an effort level $s_i \in \{1, 2, ..., M\}$, M finite, with payoff function

$$p_i = \alpha \left(\min_{j \in N} \{s_j\} \right) - c(s_i) + \delta \text{ where } \alpha > c > 0, \delta \in \mathbb{R} \text{ for all agents } i \in N \qquad (1)$$

Best response in this game is for agent i to match the lowest effort from everyone else:

$$s_i = \min_{j \in N \setminus i} \{s_j\}$$

Notice that the minimum effort coordination game is a game of strictly complementarity. In turn, it has multiple equilibria. For this particular class of game, we get pure strategy equilibria that are pareto ranked. The Nash Equilibria for this game are any strategy profile that satisfies the following condition: $\sigma = \{s_1, ..., s_N\}$ where $s_1 = s_2 = ... = s_N$. For example, everyone choosing $s_i = 3 \ \forall i \in N$ is a Nash Equilibrium. Among these M pure strategy equilibria, a strategy profile $\sigma = \{s_1, ..., s_N\} = \{M, ..., M\}$ is the payoff dominant equilibrium, while $\sigma = \{s_1, ..., s_N\} = \{1, ..., 1\}$ is the worst, but is a risk dominant equilibrium.

Please refer to Myung (2008 Working Paper) for a more detailed review and the experimental background of this particular game.

2.2 Computational Testbed

A computational testbed is a computer environment that allows us to run simulations in order to make predictions about human behavior. Though these testbeds will likely never be able to perfectly predict human behavior, they are still a useful tool for making these predictions. These testbeds allow us to run simulations of an experiment over a wide variety of parameters. Based on the simulations, we can develop behavioral hypotheses in these games, as well as select interesting parameters to be used in a laboratory experimental setting.

Others have developed computational testbeds in order to design experiments. Arifovic and Ledyard (2005 Working Paper) build computational agents to be used as a testbed for experiments on the Groves-Ledyard mechanism. In particular, the mechanism has one parameter that plays an important role in the speed of convergence. Arifovic and Ledyard make predictions about optimal values of this parameter with their computational testbed, and then confirm these predictions with experiments. Their learning algorithm is a combination of a genetic algorithm with some behavioral intuition. Their

computational agents are able to converge quickly, on average in 20 rounds. Their algorithm strongly favors convergence to a single point. Therefore in a game like battle of the sexes, their algorithm cannot support the commonly observed behavior where players learn to alternate meeting places. Our algorithm uses pattern recognition, and is therefore able to capture this behavior.

For our study, the algorithm determines which choice each agent makes in each period of a repeated game. This choice depends on the history of play as well as the agent's current state. After each agent made their choice, the choices and payoffs are revealed to all agents. The agents then update their history and current state, and make their choice for the following round.

Two main features of this algorithm are the pattern recognition scheme and the agent's states. The experiments of Sonsino and Sirota (2003) show that subjects are able to sustain patterns of Nash equilibria (alternate, not randomly mixed, between multiple equilibria). Even in 2-by-2 games, the probability of sustaining a pattern of Nash equilibria for n rounds by random choice decreases exponentially as n increases; yet subjects are still able to sustain these patterns. People's ability to sustain these patterns of equilibria provide evidence that they are in fact recognizing these patterns. Therefore, pattern recognition is a natural feature when modeling human behavior in repeated interactions. Our pattern recognition scheme is a modification of the k-nearest neighbor classification algorithm from machine learning (Dasarathy 1991). Patterns are recognized by first identifying the current play (the most recent choices in the history) and then finding previous plays that are similar to the current play. The prediction for next round is a weighted average of the outcomes of these similar plays. In each round, agents make their choice based on their current state, which are given by two parameters, γ and σ. The γ parameter represents an agent's current level of confidence. This is determined by how well that agent predicts what the other agents will do. The σ parameter represents the agent's satisfaction of the current play of the game. If the agent is not satisfied and wants to change what is happening in the game, then σ is close to 1. If the agent is satisfied with how the game is going then σ is close to 0. When all agents have high values of γ and low values of σ, then each agent's choice has low variance and each agent is satisfied with the predicted outcome of their choice, so the algorithm has converged.

Another important aspect in the algorithm is that agents are not able to calculate exact best-responses to their predictions. Instead, agents determine best responses by randomly sampling from the strategy space, and keeping the strategy that gives the highest payoff. This is important for two reasons. First, it allows for completely general payoff functions. Because the explicit best response function isn't required, the payoff functions need not be continuous nor differentiable. Also, it allows agents to have different levels of intelligence by changing the number of samples they take. For example, a very intelligent agent has a good grasp of the payoff function, and therefore is able to find the best response. This can be modeled by an agent who takes a large number of random samples to find the best response. Conversely, a very unintelligent agent is not able to

find the best response. This can be modeled as an agent that takes a very small number of samples to find the best response.

For a more detailed description of computational testbeds in economics, see Romero (2008 Working Paper). We have attached the algorithm and a detailed mathematical description in the Appendix.

3 Prediction

We run simulations using the algorithm on the minimum effort coordination games and develop testable experimental hypotheses. The benefit of using computational agents is that simulations are essentially costless, which allows us to run many trials for each parameter value.

Previous experiments on the minimum effort coordination game have focused on differences in cost and group size. The experiments have typically compared two different parameter values: a low and high cost or a small and large group (Goeree and Holt 2005). Experiments examining a large set of parameters are difficult due to constraints on the number of subjects in a given subject pool, as well as monetary costs for running large experiments. Simulations using the algorithm provide a testbed to simulate these experiments for many different parameter values. Unlike the binary comparisons, examining a larger set of parameters will give us a better understanding of the behavior which may have been overlooked in the past.

From the minimum effort coordination game defined in the previous section using equation 1, we run simulations with $\alpha = 1$, $\delta = 0$, $s_i \in [0, 1]$ for groups of four agents with 9 different costs, varying from $c = 0.1$ to $c = 0.9$. At each parameter value, we run 300 simulations lasting for 50 rounds.

Convergence Point: We find that higher costs lead to lower convergence points. Convergence points are the average play over the last 10 periods of the repeated game. The convergence points of these simulations are displayed in Figure 1. This is consistent with experimental results from minimum effort coordination games as shown in Goeree and Holt (2005).

Convergence Speed: We then examine the effect of different costs on speed of convergence.[1]

Based on the simulations, we find that the number of rounds required to converge increases with c. A plot of convergence as a function of c is displayed in Figure 2 (higher bars mean slower convergence). The intuition for increase in speed of convergence for higher cost is simple; it is more expensive for agents to search for different outcomes or experiment with different strategies.

Average Payoff: These convergence results have some interesting effects on the agent's payoffs. When agents do not all choose the same effort (i.e., best respond), the outcome is pareto inefficient. If all agents chose the minimum effort for a given strategy profile, then everyone's payoff would be weakly higher, with at least one receiving a strictly higher payoff. Since it is inefficient when all agents are not choosing the same effort, slow convergence may lead to lower average payoffs. The average payoff per agent for different costs is displayed in Figure 3. It is difficult to compare the welfare between

[1] We will use convergence in γ as a measure of convergence. See Appendix.

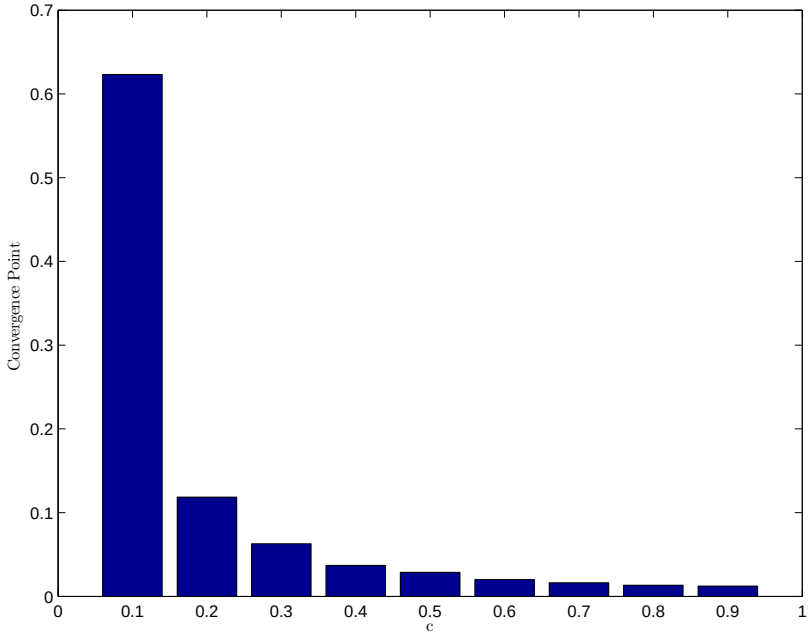


Figure 1: Convergence Points as a Function of Cost

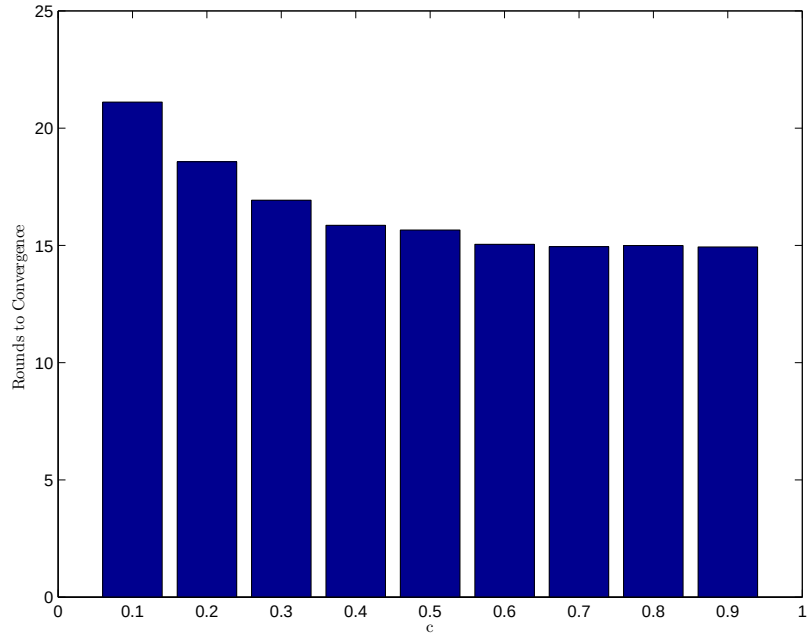


Figure 2: Convergence Speed as a Function of Cost
Higher bars indicate slower convergence

two experiments with different costs because they have different payoff functions. Even though welfare is difficult to compare, the payoff for any given strategy profile is lower when the cost of effort is higher. Intuition thus suggests that higher cost of effort should lead to lower average payoffs in the repeated game. However, we argue that higher cost can actually lead to higher payoffs. The increase in payoffs due to faster convergence outweighs the decrease in payoffs due to higher cost. Note that the difference in average payoff shrinks as number of rounds increases in Figure 3. This result is due to the fact that the positive welfare of faster convergence gets averaged out by the negative welfare of higher cost in effort as the game is played for more periods.

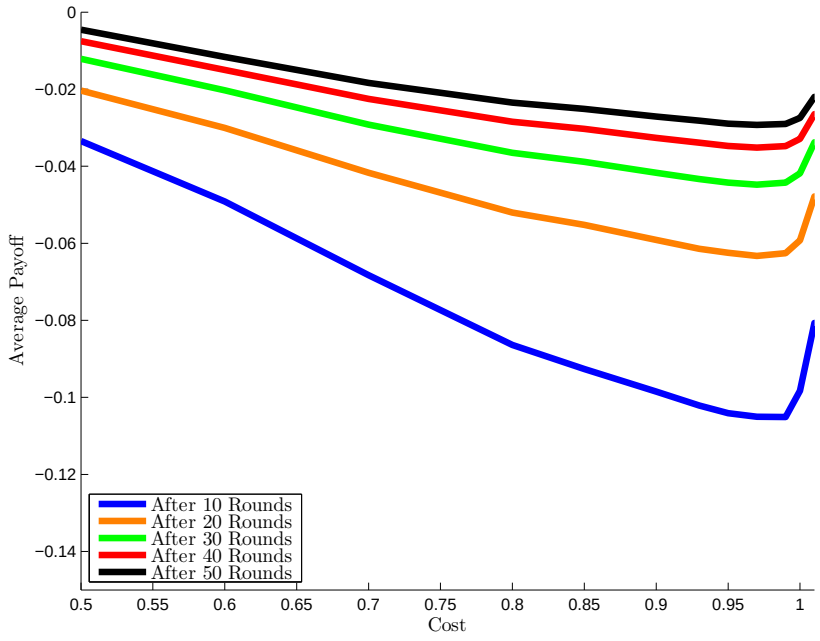


Figure 3: Average Payoffs for Different Costs in Minimum Effort Coordination Game as a Function of Number of Rounds

3.1 Hypotheses

We test the following three hypotheses that were generated by the computational testbed in the experimental laboratory:

Hypothesis 1. Convergence Point: The game will converge to a pareto dominated payoff as the cost of effort increases.

Hypothesis 2. Convergence Speed: The game will converge faster to an equilibrium as the cost of effort increases.

Hypothesis 3. Average Payoff: The average payoff does not monotonically decrease as the cost of effort increases.

4 Experimental Design

4.1 Overview

The experiments were conducted at the California Social Science Laboratory (CASSEL) located in the University of California, Los Angeles (UCLA). A total of 60 subjects participated in the experiments. The average performance-based payment was 20USD. All students were registered as subjects with CASSEL (signed a general consent form) and the experiment was approved by the local research ethics committee at both universities. These labs consist of over 30 working computers divided into a cubicles, which prevents students from viewing another student's screen.

The experiment was programmed and conducted with the experiment software z-Tree (Fischbacher 2007). The instructions were available both in print as well as on screen for the participants, and the experimenter explained the instruction in detail out-loud. Participants were also given a brief quiz after instruction to insure proper understanding of the game and the software. A copy of the instruction, as well as the payoff tables, are available on the author's website.

The subjects were randomly assigned to their roles in the experiment. Furthermore, no one participated in more than one experiment. The identity of the participants as well as their individual decisions were kept as private information. However, each groups knew their own minimum effort. Experiment used fictitious currency called francs. The participants were fully aware of the sequence, payoff structure, and the length of the experiment. All participants filled out a survey immediately after the experiment.

4.2 Details of the Experiment

A total of 20 subjects participated in each session. These 20 subjects were split into 5 groups of 4, and each group used a different cost parameter. The entire session was divided into 5 blocks, and each block was divided into 15 rounds. After each block, the subjects were randomly rematched (with replacement) to another group of 4 and were randomly reassigned another payoff parameter (with replacement). See Figure 4 for the time line.

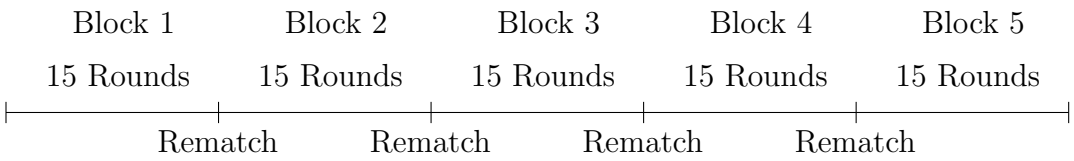


Figure 4: Timeline and Matching Structure for the Experiment

Subjects played a minimum effort coordination game per round. Their task was to choose an effort level,

$$s_i \in \{1, ..., 7\}$$

and their payments were determined by the following payoff function

$$p_i = 1000\left(\min_{j\in N}\{s_j\}\right) - c(s_i) + 5950$$

In each block, there were 5 groups each with a different payoff matrix based on

$$c \in \{50, 500, 900, 950, 990\}$$

The subjects were shown the payoff table displayed in Table 1, with the calculation already completed for the subjects. The group size, randomization, and the fact that everyone in the group were using the same payoff table were common knowledge. However, the group's own minimum effort was private information to the group and was not available to the outside members.

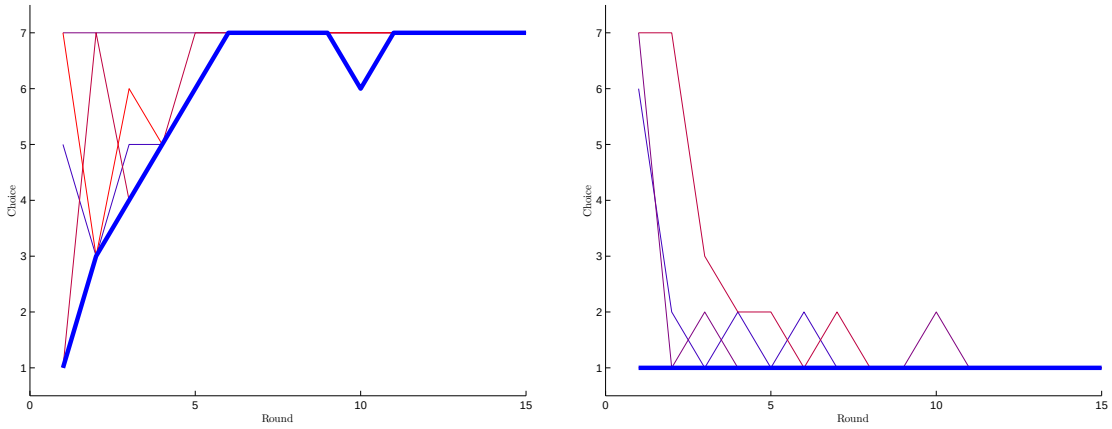
	Minimimum effort of all agents						
i**'s Effort**	7	6	5	4	3	2	1
7	$12950 - 7c$	$11950 - 7c$	$10950 - 7c$	$9950 - 7c$	$8950 - 7c$	$7950 - 7c$	$6950 - 7c$
6	–	$11950 - 6c$	$10950 - 6c$	$9950 - 6c$	$8950 - 6c$	$7950 - 6c$	$6950 - 6c$
5	–	–	$10950 - 5c$	$9950 - 5c$	$8950 - 5c$	$7950 - 5c$	$6950 - 5c$
4	–	–	–	$9950 - 4c$	$8950 - 4c$	$7950 - 4c$	$6950 - 4c$
3	–	–	–	–	$8950 - 3c$	$7950 - 3c$	$6950 - 3c$
2	–	–	–	–	–	$7950 - 2c$	$6950 - 2c$
1	–	–	–	–	–	–	$6950 - c$

Table 1: Sample Payoff Table that was used in the Experiment

Calculations were already filled in for the subjects

5 Experimental Results

Figure 5 illustrates sample results from one of the block of sessions. Figure 5 (a) is an example where there is a high level of coordination (converging to an effort level of 7) and Figure 5 (b) is an example where there is a low level of coordination (converging to an effort level of 1).



(*a*) Coordination to High Effort (*b*) Coordination to Low Effort

Figure 5: Sample Results From One of The Block of Session for Illustration Purpose
The thin lines represent individual choices and the thick line represents the group's minimum choice

5.1 Convergence Points

First, we test the hypothesis that higher costs will lead to lower convergence points and provide the results in Table 2, Table 3, and Figure 6. These results are taken from the average choice of the last 5 rounds and it supports the hypothesis that the average choice drops as the cost parameter increases. While the cost parameter between $c \in \{50, 500\}$ provides a high level of average choice around 4.5 to 5, the average choice drops significantly lower to 1 to 1.2 for cost parameter between $c \in \{900, 950, 990\}$. Although we do not get a significant difference between the means from $c = 900$ and $c = 950$, we do obtain significant differences in the right direction for the rest of the mean comparisons.

	$\mathbf{c = 50}$	$\mathbf{c = 500}$	$\mathbf{c = 900}$	$\mathbf{c = 950}$	$\mathbf{c = 990}$
Choice	4.8485	4.5000	1.2864	1.2606	1.1242
SE	0.0932	0.0975	0.0363	0.0391	0.0244

Table 2: Average Choice for Different Cost Parameters

	$\mu_{50} > \mu_{500}$	$\mu_{500} > \mu_{900}$	$\mu_{900} > \mu_{950}$	$\mu_{950} > \mu_{990}$
p-value	0.0126	0	0.1677	0.0023
t-value	2.2428	21.2006	-0.9642	2.8422

Table 3: Average Choice Comparison

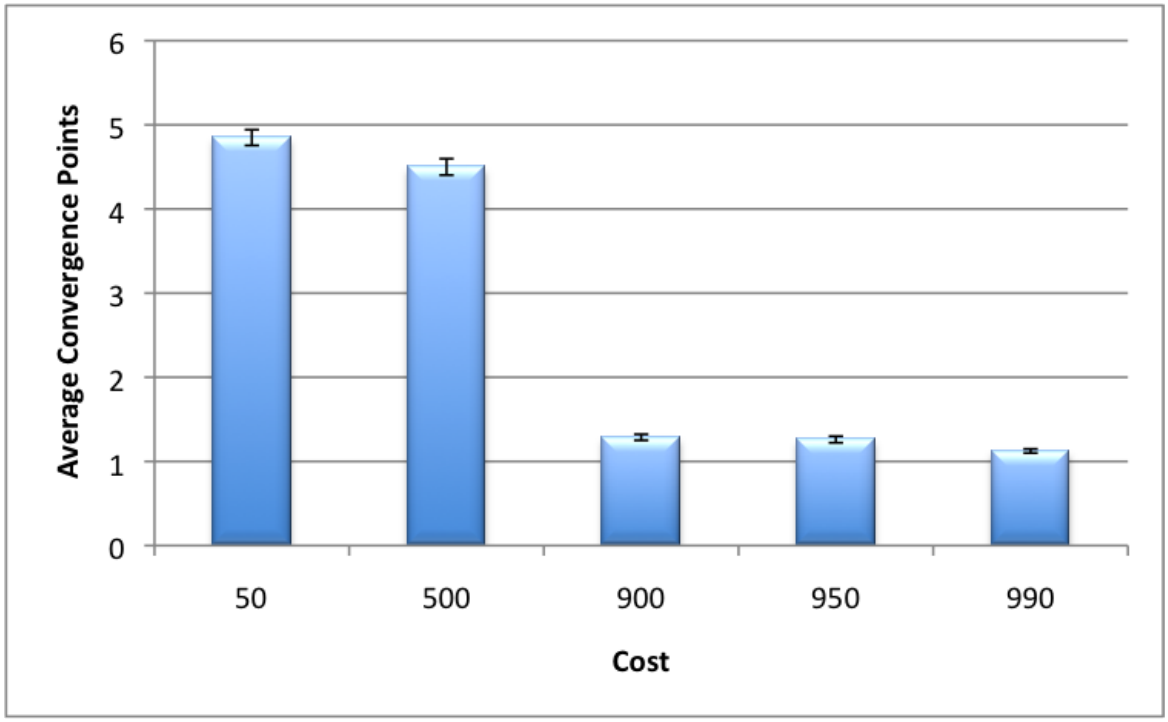


Figure 6: Average Convergence Points for Different Cost Parameters

5.2 Convergence Speed

Comparing convergence speed is bit trickier than comparing convergence points. Consider the following example in Figure 7. If one were to use a rule that the convergence occurs when there are no deviations (i.e., everyone is best responding), then there won't be any convergence until round 13 in the example. When studying experimental results with subjects from a laboratory, this may be too conservative of a criterion. Noisy choice in human behavior is often expected in experiments. Whether these noises are rational or not is another story. However, there are many different ways of modeling noisy choices, such as the Quantal Response Equilibrium (McKelvey and Palfrey 1995), the Level-K Model, and the Cognitive Hierarchy Model (Camerer, Ho, and Chong 2004), among others.

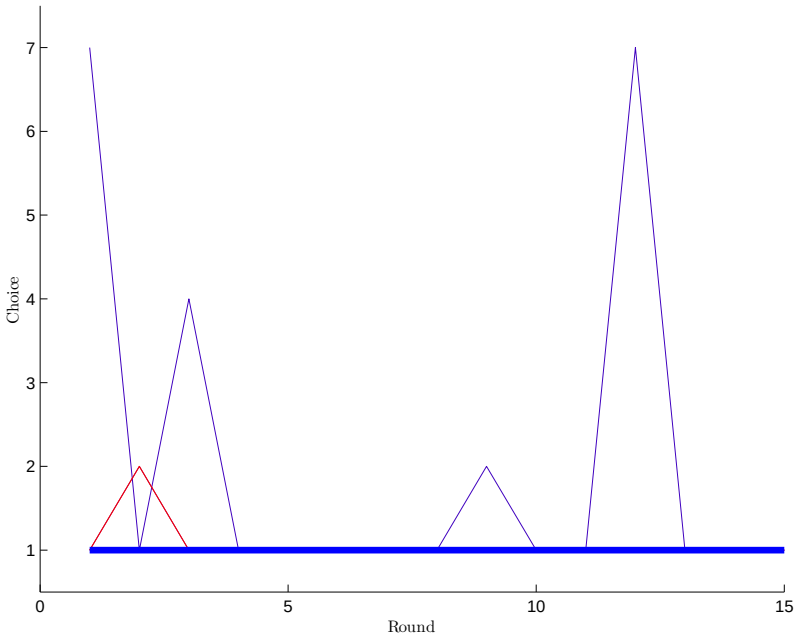


Figure 7: A Sample Result From a Block of Session
The thin lines represents individual choices and the tick line represents the group's minimum choice

Here, we provide two means of measuring convergence. First, we use a more quantitative measure of convergence called v-bounded condition. Then we introduce a more qualitative and intuitive measure of convergence called the similarness condition.

Definition: We say that the game has converged to a particular equilibrium at round t under *v-bounded condition* if the variance of number of strategies chosen is always less than v for every round starting from t. Specifically, $var_{t+m}(\sigma_1, ..., \sigma_n) \leq v$, $\forall m \geq 0$.

For example, if the strategy profile σ consists of $[3, 3, 3, 4]$, this will require that a variance parameter of $v \geq 0.25$ will be needed to consider this strategy profile as converged under the v-bounded condition. See Table 4 for other samples of strategy profile and its required variance parameter for v-bounded condition.

$\boldsymbol{\sigma}$	**Minimum** v
$[3, 3, 3, 4]$	.25
$[3, 3, 4, 4]$	.33
$[2, 3, 3, 4]$	.66
$[3, 3, 4, 4]$	.92
$[3, 3, 3, 5]$	1

Table 4: Samples of Strategy Profile and its Required v Parameter for v-bounded Condition

Using the v-bounded condition criterion for the notion of convergence, Figure 8 illustrates the average rounds it took for the game to converge.[2]Although convergence speed seems to be increasing as the cost parameter increases, differences are not statistically significant. Consider the following example from Figure 7 to illustrate why the v-bounded condition may not be a good criterion: We would require $v \geq 9$ in order to allow this particular example to be considered converged under v-bounded condition due to a large jump in choice of effort by one of the players in round 12. This does not take into account that the deviation is by one person for only one period. However, intuitively, one may think that this game has converged at round 4.

Therefore, we use a more intuitive and qualitative measure of convergence. We consider the number of different strategies being used from the strategy profile for a given round. We say the game has converged to a particular equilibrium if a high proportion of people use the same strategy. We define this as *similarness condition*. The added benefit of the similarness condition is that it does not unreasonably penalize cases where one person may deviate significantly away from the best response for just one period. By the same token, it also means that this measure treats the following two strategy profiles as equally converged: $[2, 2, 2, 3]$ and $[1, 1, 1, 7]$.

Figure 9 shows the frequency of different strategies played for various cost of effort. If the game is indeed converging faster under the similarness condition, we expect to see a higher frequency of blue and sky-blue, which indicates everyone playing the same strategy and three people playing the same strategy, respectively. As the cost of effort increases, we observe an increase in frequency of blue and sky-blue. This increase in frequency holds true for any given round. Furthermore, the frequency of blue and sky-blue also increases as the experiment proceeds (number of round increases). In other words, there are many different strategies being played in the initial round but subjects learn to best respond.

Using this similarness condition as a convergence criterion, we conclude that the game

[2]We drop the last round deviation because there may be end game effects.

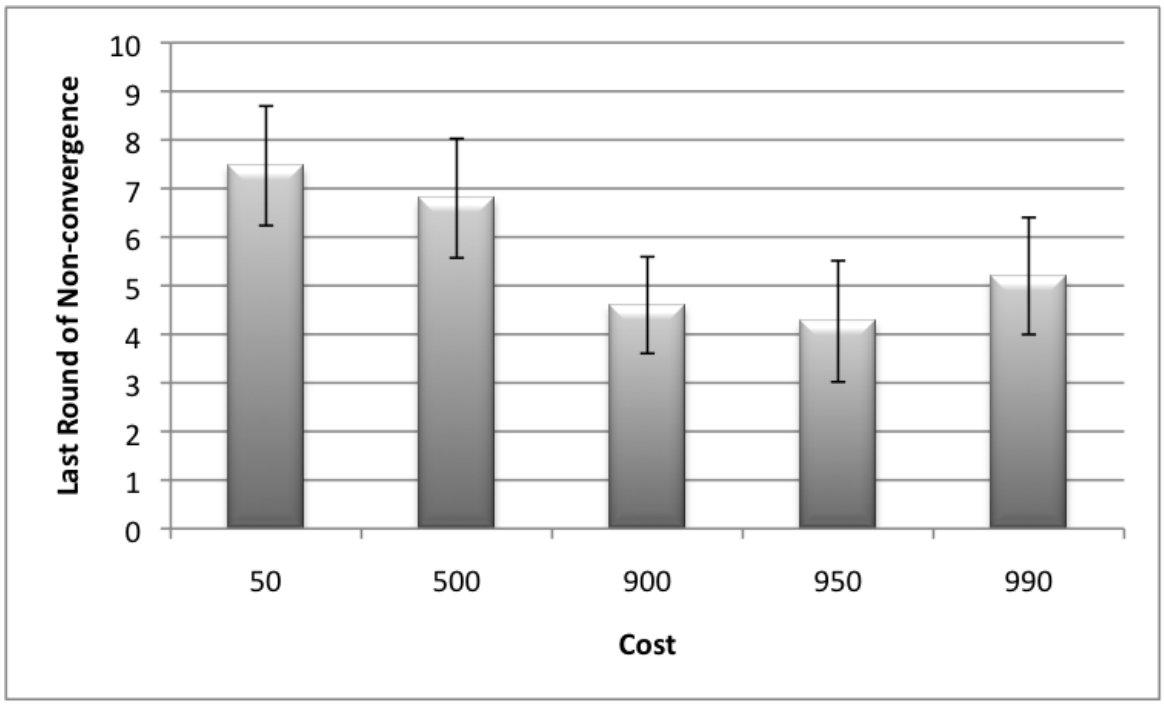
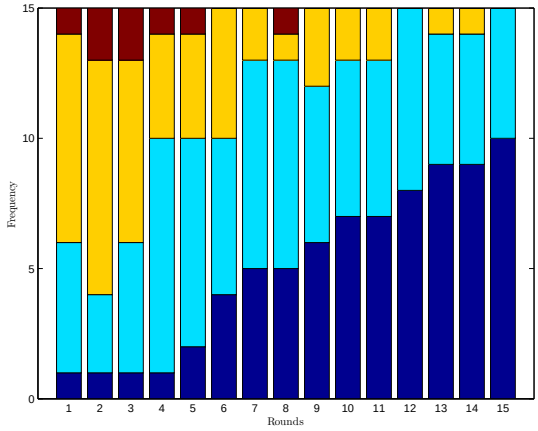
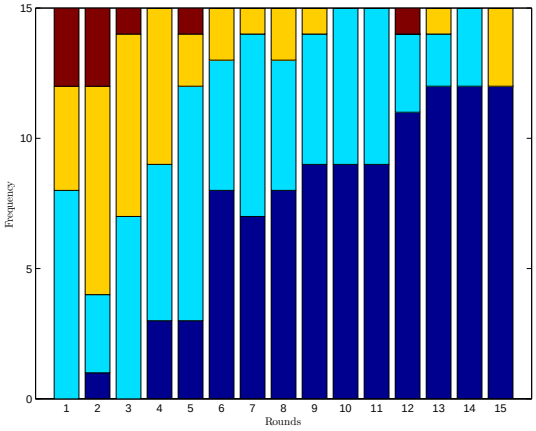


Figure 8: Number of Rounds Needed for Convergence for $v \leq 0.5$

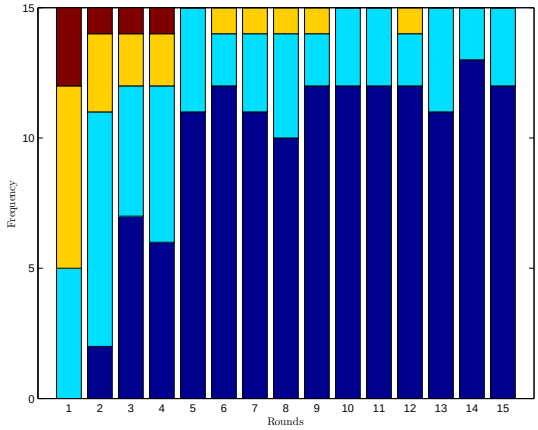
converges faster to a particular equilibrium as the cost of effort increases.



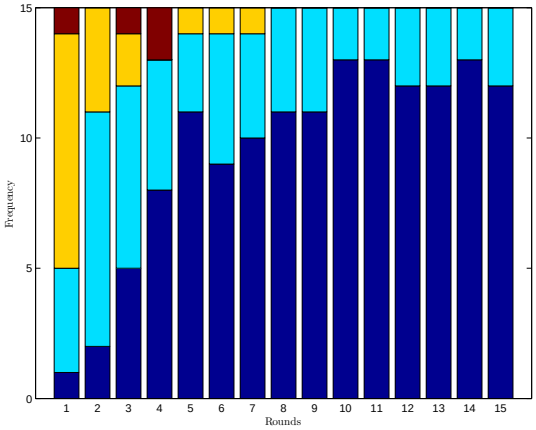
(a) $c = 50$



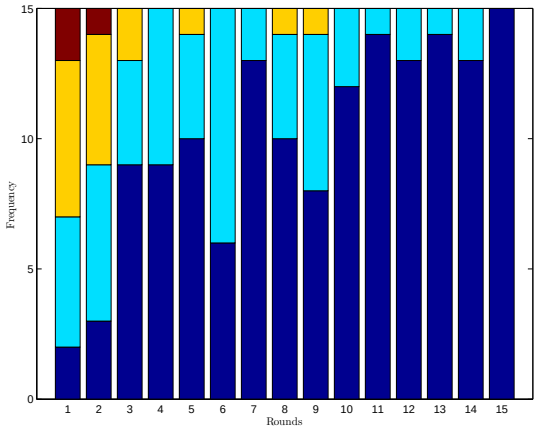
(b) $c = 500$



(c) $c = 900$



(d) $c = 950$



(e) $c = 990$

All same effort Two different efforts Three different efforts Four different efforts

Figure 9: Frequency of Different Strategies Played for Various Costs

5.3 Average Payoff

Finally, we analyze the behavior of the average payoff as the cost increases. Refer to Figure 10 and Table 5 and 6 to see the average payoff and their mean comparisons up to 4 rounds for each of the cost parameters from the experiment. What we observe, and is statistically significant, is that the average payoff does decrease from $\mu_{50} = 9088$ at $c = 50$ to $\mu_{950} = 4846$ at $c = 950$. However, as the simulation has predicted, the average payoff at $c = 990$ of $\mu_{990} = 5136$ is significantly higher than the average payoff at $c = 950$ of $\mu_{950} = 4846$ ($p < 0.05$). Although the average payoff of $\mu_{900} = 4968$ at $c = 900$ is higher than the average payoff of $\mu_{950} = 4846$ at $c = 950$, they are not statistically different.

Given that we observe a non-monotonicity in average profit as a function of cost of effort in the first 4 rounds, we test the significance after the entire block of the experiment (15 rounds). The result is displayed in Figure 11 and Table 7 and 8. Again, we observe a similar pattern to the results from the first 4 rounds. The average payoff of $\mu_{990} = 5650$ at $c = 990$ is significantly greater than the average payoff of $\mu_{950} = 5560$ at $c = 950$ ($p < 0.1$). Furthermore, the average payoff in this setting is the lowest at $c = 950$, which is also lower than the average payoff of $\mu_{900} = 5652$ at $c = 900$ ($p < 0.1$).

Another topic worth mentioning is that the difference between the average payoff when $c = 990$ and $c = 950$ diminishes as more rounds are played. This confirms the prediction made by the simulation in Figure 3. As more rounds are played, the positive welfare from the lower cost averages out the negative welfare from the wasted effort. For example, after 4 rounds, the difference in average payoff is $\mu_{990} - \mu_{950} = 288.9583$. But, after 15 rounds, the difference decreases to $\mu_{990} - \mu_{950} = 90.1889$. In other words, the non-monotonicity of average payoff is most salient at the initial phase of the game.

	c = 50	**c = 500**	**c = 900**	**c = 950**	**c = 990**
μ	9088	6527	4968	4846	5136
SE_μ	118.05	103.36	115.42	124.51	106.56

Table 5: Average Payoffs for Different Cost Parameters After 4 Rounds

	$\mu_{50} > \mu_{500}$	$\mu_{500} > \mu_{900}$	$\mu_{900} > \mu_{950}$	$\mu_{950} < \mu_{990}$
p-value	0	0	0.2366	0.0393
t-value	16.3234	10.05	0.7178	1.7632

Table 6: Average Payoffs Comparison After 4 Rounds

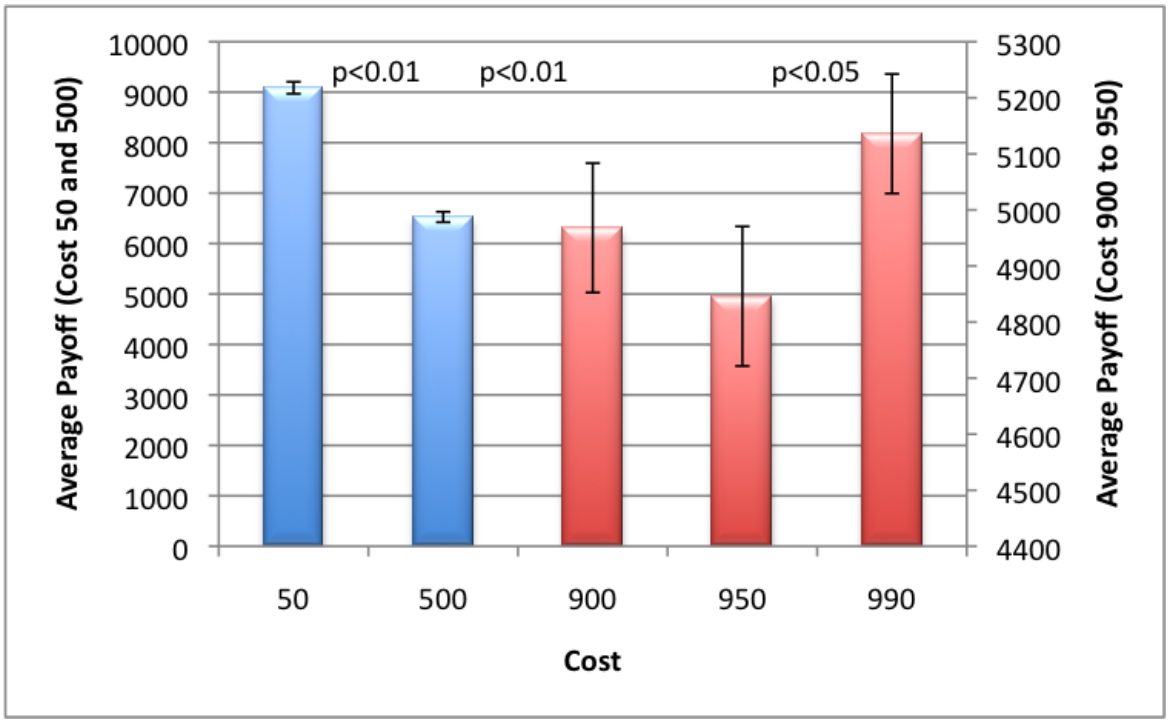


Figure 10: Average Payoff After 4 Rounds

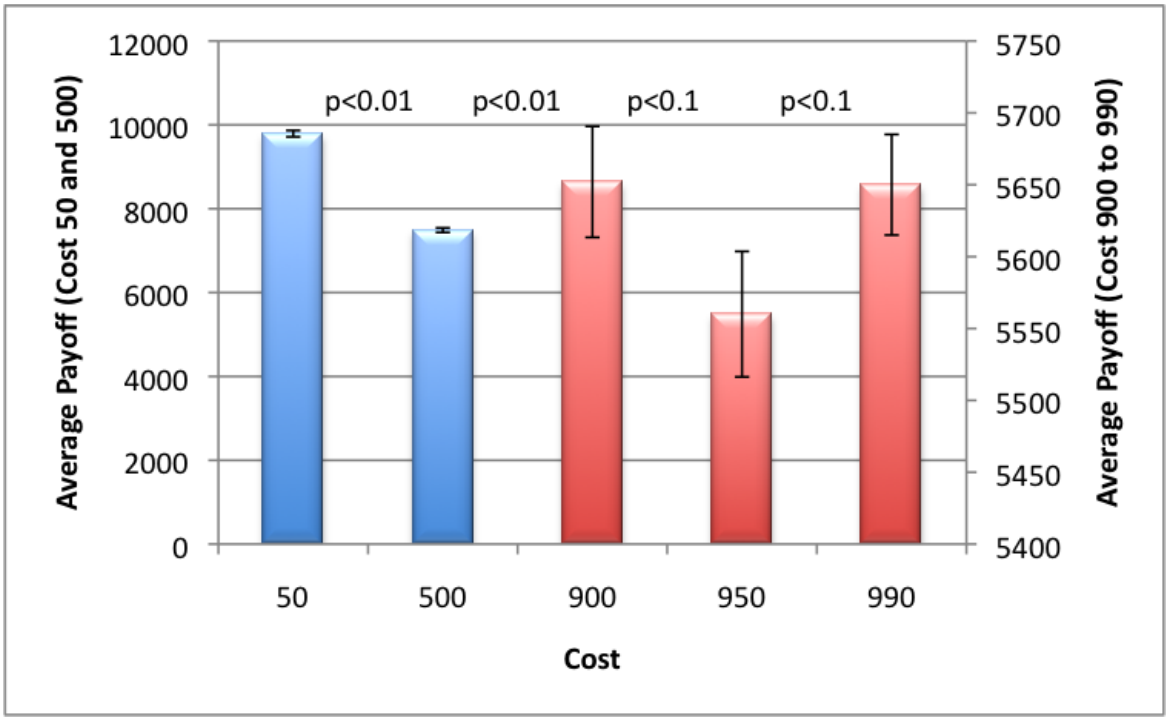


Figure 11: Average Payoff After 15 Rounds

	c = 50	**c = 500**	**c = 900**	**c = 950**	**c = 990**
μ	9791	7489	5652	5560	5650
SE_μ	76.23	53.75	38.61	43.66	35.02

Table 7: Average Payoffs for Different Cost Parameters After 15 Rounds

	$\mu_{50} > \mu_{500}$	$\mu_{500} > \mu_{900}$	$\mu_{900} > \mu_{950}$	$\mu_{950} < \mu_{990}$
p-value	0	0	0.0557	0.0536
t-value	24.6797	27.75	1.5928	1.6114

Table 8: Average Payoffs Comparison After 15 Rounds

6 Conclusion

We utilize a computational testbed to design a laboratory experiment to better understand the behavior of the minimum effort coordination game. Using the computational testbed, we are able to make predictions of interesting and un-intuitive behavioral features of the minimum effort coordination game. First, the game converges to a pareto dominated equilibrium as the cost of effort increases. Second, the game converges faster to an equilibrium as the cost of effort increases. Lastly, the average payoff does not monotonically decrease as cost of effort increases. Had we not used the testbed, the likelihood of running across these behavioral features would have been low and the cost of running multiple sessions to figure out the parameters would have been expensive.

Another important contribution from this research is to show that the testbed we have designed effectively predicts human behaviors in the minimum effort coordination game.

We focused primarily on the behavior of the minimum effort coordination game as a function of cost. However, our results also suggest predictions from changing the number of players in the game. These are testable hypotheses we encourage others to pursue. Furthermore, we have focused mainly on minimum effort coordination game but we are hopeful that our computational testbed would generalized to other class of coordination games such as battle of the sexes.

7 Appendix

The explanation of the algorithm is divided up into four parts: notation, preliminary initialization, round k action, and preparation for round $k+1$. For notational purposes, the superscript typically denotes the agent and the subscript denotes the round.

7.1 Notation

Each agent has a database of information that is used to help make their choice in each round. At the start of each round, each agent has two parameters in their database, the confidence parameter γ and the satisfaction parameter σ. These parameters for agent i in round k are denoted by γ_k^i and σ_k^i. The agents use these parameters to help make their choice. Agent i's choice in round k is represented by $\mathbf{x}_k(i)$. The choice of all agents in round k is given by $\mathbf{x}_k$, which yields payoffs $\pi_i(\mathbf{x}_k) = \pi_k^i$ for agent i.

After the agents make their choices, they update their database of information in preparation for the next round. Each agent makes a prediction about what the other agents will play in the following round. Let $\hat{\mathbf{x}}_k^i(j)$ be agent i's prediction for agent j's play in round k. The full prediction vector, $\hat{\mathbf{x}}_k^i$, consists of predictions for all of the other agents.

As the game progresses, each agent creates a quasi-best-response matrix. Agent i's quasi-best response matrix at round k is denoted by Q_k^i. This matrix helps the agent determine what they should choose after they have made their prediction. To do this, the agent groups similar strategy profiles together in the quasi-best-response matrix. The agent then determines which play is best against these similar strategy profiles by randomly sampling responses from the strategy space. In the future, when a similar strategy profile arises, the agent uses this quasi-best-response matrix to help remember what they did in the past. From this quasi-best-response matrix, the agent determines the quasi-best-response for their prediction for round k, which is denoted by ${x_k^i}^*$. More details about the quasi-best-response are given below in the description of the algorithm in the preparation for round $k+1$ section.

Each agent also keeps track of their best and worst outcomes. To do this, each agent randomly chooses J strategy profiles from the uniform distribution on the joint strategy space $S = [0,1]^N$. Next, they calculate the payoffs for each of these profiles, and save the strategy profiles which yield the highest and lowest payoffs, $\bar{\mathbf{x}}_k^i$ and $\underline{\mathbf{x}}_k^i$, respectively. These are referred to as the highest and lowest known choices for agent i in round k. The payoffs for these strategy profiles, $\bar{\pi}_k^i$ and $\underline{\pi}_k^i$, are referred to as the highest and lowest known payoffs for agent i at round k.

All of this information is stored in the agent's database, and is available when they are making their choice in round k.

7.2 Initialization

Many learning algorithms contain multiple initialization periods, where the agents choose randomly in the strategy space. Since the focus of this paper is not long run convergence, but rather short run behavior, the initialization period has to be short. Before the first choice is made, the agents randomly choose J strategy profiles to determine their initial highest and lowest known payoffs, $\bar{\mathbf{x}}_0^i$ and $\underline{\mathbf{x}}_0^i$, respectively. Each agent then makes the initial predictions about the other agents by randomly drawing a number from the uniform distribution on $[0,1]$, that is $\mathbf{x}_1^i(j) \sim U[0,1]$. Finally, each agent starts with the lowest possible confidence level, $\gamma_1^i = 10$. They also start with the highest satisfaction parameter, $\sigma_1^i = 1$, because they have no reason to try to change the outcome of the game yet. With these initial parameters, the algorithm is ready to run.

7.3 Round k

Entering round k, agent i has a database of information which is used to make a choice in round k. The choice in round k is a random number from a beta distribution with mean μ and variance ν^2. The mean of the distribution is a convex combination of the quasi-best-response, ${x^i}_k^*$, and the strategy which yields the highest known payoff for agent i at round k, $\bar{x}_k^i$. The weight on each term is determined by the current level of satisfaction. If the agent's satisfaction level is high ($\sigma_k^i = 1$) then they play the quasi-best-response for their prediction. If the agent is not satisfied ($\sigma_k^i < 1$), then they try to move the outcome towards the point which yields their highest known payoff. That is,

$$\mu = \sigma_i^k {x^i}_k^* + (1 - \sigma_i^k)\bar{x}_k^i$$

The variance of the distribution is inversely proportional to the current level of confidence[3]. The proportionality constant is ρ, so the variance is,

$$\nu^2 = \frac{1}{\rho \gamma_k^i}$$

As the confidence level of the agent increases, the choice distribution has lower variance, and therefore the choice is more accurate. When the agent is not confident about what the other agents will do, then his choice distribution has high variance, and his choice is not as accurate.

After all agents have made their choices as described above, the payoffs are calculated. The agents then learn the choices of the other agents as well as the payoffs of all agents. At this point, the agents begin their preparation for round $k+1$ by updating their database of information.

[3]It is not possible to have a distribution over a closed region, if the variance is high, and the the mean is sufficiently close to the endpoints. If this is the case, then it is corrected by using a modified beta distribution with mass point on the endpoint.

7.4 Preparation for Round $k+1$

The agents have a variety of tasks to perform in preparation for round $k+1$.

Update extremes As the game progresses the agents become more acquainted with the payoff function. To model this, each round the agents update their highest and lowest known payoffs by taking J random samples from the joint strategy space. For each random sample $\mathbf{z}_j$, the payoff vector is calculated. If the payoff for agent i from the sample is higher than the highest known payoff for agent i in round k, then the agent sets the highest known choice for round $k+1$ to $\bar{\mathbf{x}}^i_{k+1} = \mathbf{z}_j$ and the highest known payoff round $k+1$ to $\bar{\pi}^i_{k+1} = \pi_i(\mathbf{z}_j)$. If none of the payoffs from the J sample points are higher than the highest known payoff for agent i at round k, then the highest known choice and payoff from round k are carried over to round $k+1$, i.e., $\bar{\mathbf{x}}^i_{k+1} = \bar{\mathbf{x}}^i_k$ and $\bar{\pi}^i_{k+1} = \bar{\pi}^i_k$. The same update is performed for the lowest known play and payoff.

Prediction for round $k+1$ In order to make a choice in round $k+1$ it is useful for the agents to have some prediction about what their opponents are going to do in round $k+1$. The prediction scheme used by the agents is a modification of the nearest neighbor classification algorithm from machine learning. The goal of the prediction scheme is to make a prediction for $\mathbf{x}_{k+1}$. Since there are N agents, the agents' choices at round k are given by the vector $\mathbf{x}_k \in \mathbb{R}^N$. A pattern is vector combining one or more of these choice vectors. For example, a pattern of length 3 is $[\ \mathbf{x}_k \quad \mathbf{x}_{k+1} \quad \mathbf{x}_{k+2}\ ]$. The agents divide the history of choice into the current pattern, previous patterns, and outcomes. Each previous pattern has a corresponding outcome. The algorithm makes a prediction for the outcome of the current pattern. The agents determine which of the previous patterns are closest to the current pattern. Then the agents' prediction is a weighted sum of the outcomes of the closest patterns. The agents repeat this process for patterns of different lengths, n. After the agent has done this for all values of n, he compares them, and determines which pattern length provides the best prediction.

For example, consider a two-player game with the history of play after eight rounds,

$$(0,0), (1,1), (1,1), (0,0), (1,1), (1,1), (0,0), (1,1)$$

Let's examine the prediction by agent 1 of what agent 2 will play in the ninth round. First, agent 1 considers patterns of length 1. The current pattern is the most recent play, $(1,1)$. This has been played four previous times in rounds 2, 3, 5, and 6. These are the closest patterns. When these closest patterns have been played in the past, agent 2 has responded by playing 1, 0, 1, and 0 in the respective following rounds. These are the outcomes for the four closest patterns. This is not good, because agent 2 has played 0 half the time, and 1 half the time, so it is difficult to predict what agent 2 will play in the next round based on patterns of length 1.

Next, agent 1 looks at patterns of length 2. The current pattern in this case is the play in the previous 2 rounds, $(0,0),(1,1)$. This pattern has been played twice before in the past, in rounds 1-2 and 4-5. In response to this pattern, agent 2 has played 1 in both rounds 3 and 6. After patterns of length 2, agent 2 always chose 1. Therefore, patterns of length 2 are better for prediction that patterns of length 1.

More formally, at the end of the k^{th} round, each agent considers patterns of different lengths n. For each length, there are $k-n$ previous patterns of length nN each. The agent forms the previous patterns matrix $X \in \mathbb{R}^{k-n \times nN}$ and the output matrix $Y \in \mathbb{R}^{k-n \times N}$,

$$X = \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \\ \mathbf{x}_2 & \cdots & \mathbf{x}_{n+1} \\ \vdots & & \vdots \\ \mathbf{x}_{k-n} & \cdots & \mathbf{x}_{k-1} \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_2 \end{bmatrix} \text{ and } Y = \begin{bmatrix} \mathbf{x}_{n+1} \\ \mathbf{x}_{n+2} \\ \vdots \\ \mathbf{x}_k \end{bmatrix}$$

Each row of the previous patterns matrix is a single pattern, and these are denoted by X_m for $m = 1, \ldots, k-n$. The current prediction is the vector $c \in \mathbb{R}^{nN}$

$$c = \begin{bmatrix} \mathbf{x}_{k-n+1} & \cdots & \mathbf{x}_k \end{bmatrix}$$

Next, the agent finds the j rows of X which are closest to the current pattern c in terms of Euclidean distance. To do this, the agent forms the distance vector by finding the length between the current point and each of the previous points,

$$\mathbf{d} = \begin{bmatrix} \|X_1 - c\| \\ \|X_2 - c\| \\ \vdots \\ \|X_{k-n} - c\| \end{bmatrix}$$

Let J be the set of indices of the j smallest terms in the distance vector $\mathbf{d}$. That is $d_j \leq d_k$ for $j \in J$ and $k \neq J$. These indices correspond to the j rows of X which are closest to the current point c.

The agent now determines which pattern length gives the best prediction. As exhibited in the above example, the agent wants to choose the pattern length with the most similar outcomes. To determine the optimal pattern length for each n, the agent takes the outcome of the j closest points, and calculates the average of these points, $\bar{Y}$. Then the agent computes the variance of these j closest points,

$$V_n = \sum_{j \in J} \|Y_j - \bar{Y}\|$$

Now, the agent compares the variance for all considered pattern lengths and chooses the pattern length with the smallest variance. If there is a tie, then the agent chooses

the shorter pattern. Note that average variances are higher in higher dimensions. This is not corrected for, which gives an additional benefit to the shorter patterns, because shorter patterns are easier to recognize.

Once the agent has selected which pattern length to use, he forms a weighted average of the closest outcomes. The closer the pattern is to the current outcome, the higher the weight is. The patterns are weighted using a logistic function. The prediction for the next period is thus,

$$\hat{\mathbf{x}}^i_{k+1} = \frac{\sum_{j \in J} Y_j e^{\mathbf{d}(j)}}{\sum_{j \in J} e^{\mathbf{d}(j)}}$$

Therefore, if the distance to each of the j closest patterns is 0, then the prediction is just the average outcome from those j closest patterns. The agent makes their choice for period $k+1$ based on this prediction.

Quasi-Best-Response The quasi-best-response helps the agent determine the best response for his prediction for round $k+1$. To do this the agent updates the quasi-best-response matrix from the previous period, Q^i_k. Each row of the quasi-best-response matrix consists of three items: prediction about what the other agents will do, what agent i should do given that prediction, and the payoff given that strategy profile. More formally row m has the terms,

$$Q^i_k = [\ \mathbf{q}^m_{-i} \quad \mathbf{q}^m_i \quad \pi_i\left(\mathbf{q}^m_i, \mathbf{q}^m_{-i}\right)\ ]$$

Here, $\mathbf{q}^m_{-i}$ are the choices of the other agents, and $\mathbf{q}^m_i$ is the choice of agent i. Agent i updates Q^i_k as follows. First, agent i determines if the current prediction is similar to any of the entries already in the quasi-best-response matrix. To do this, agent i chooses a set, R, of random strategies. For each row of the quasi-best-response matrix, agent i calculates the payoff difference,

$$pd_m = \sum_{r \in R} \left| \pi_i\left(r, \mathbf{q}^m_{-i}\right) - \pi_i\left(r, \hat{\mathbf{x}}^i_{k+1}\right) \right|$$

Next, the agents find the minimum payoff distance, $pd^* = \min pd_m$. If the distance is small, i.e., $pd^* < \delta$, then the two strategies are similar, and therefore are combined in the quasi-best-response matrix. If $pd^* > \delta$, then the two strategies are not similar, so a new entry is created in the quasi-best-response matrix. Let the threshold δ be a fraction of the difference between the highest and lowest payoff,

$$\delta = \frac{\bar{\pi}^k_i - \underline{\pi}^k_i}{20}$$

If $pd^* < \delta$, then the agent updates the row of the quasi-best-response matrix corresponding to pd^*, call this row m^*. The agent takes the set of R strategies, and calculates the payoffs $\pi_i\left(r, \hat{\mathbf{x}}_i^{k+1}\right)$. Let r^* denote the strategy from R which maximizes $\pi_i\left(r, \hat{\mathbf{x}}_i^{k+1}\right)$. If this new strategy yields a higher payoff than the current quasi-best-response, i.e., $\pi_i\left(r^*, \hat{\mathbf{x}}_i^{k+1}\right) > \pi_i\left(\mathbf{q}_i^m, \mathbf{q}_{-i}^m\right)$, then the row m^* is updated to $\mathbf{q}_{-i}^{m^*} = \hat{\mathbf{x}}_i^{k+1}$ and $\mathbf{q}_i^{m^*} = r^*$.

If $pd^* > \delta$, then the agent creates a new row for the quasi-best-response matrix, call this row $M+1$. Again, the agent calculates the payoffs $\pi_i\left(r, \hat{\mathbf{x}}_i^{k+1}\right)$ for all $r \in R$, with r^* being the strategy which yields the maximum payoff. The agent then updates the quasi-best-response matrix by setting $\mathbf{q}_{-i}^{M+1} = \hat{\mathbf{x}}_i^{k+1}$ and $\mathbf{q}_i^{M+1} = r^*$.

Update γ The parameter γ measures the current level of confidence of the agent. When the agent makes accurate predictions, his confidence increases. In preparation for round $k+1$, the agent compares his prediction for round k that was made in round $k-1$, $\hat{\mathbf{x}}_k^i$, with the actual play from round k, $\mathbf{x}_k$. Based on this prediction and outcome, the agent updates his confidence as follows,

$$\gamma_{k+1} = \frac{\alpha_1}{\|\hat{\mathbf{x}}_k^i - \mathbf{x}_k\| + \alpha_2}\gamma_k$$

Therefore, if the Euclidean distance between the prediction and the actual outcome is less than $\alpha_1 - \alpha_2$, then the confidence increases. The maximum possible increase in confidence is α_1/α_2.

Update σ The parameter σ represents the agent's satisfaction at the current state of the game. If the agent is not satisfied with the current outcome, then he may try to induce the other agents to play something else in order to change the current outcome. If the agent's attempt to move is unsuccessful, then he will stop trying. For example, suppose two agents are coordinating at one of the equilibria repeatedly in the battle of the sexes game. Agent 1 is at her optimal equilibrium, and Agent 2 is at his least favored equilibrium. Agent 2 realizes that he can receive a higher payoff at the other equilibrium. Therefore he will try to induce agent 1 to start playing the other equilibrium. However, agent 1 may not change the way she is playing, even when agent 2 is starts playing something else. If agent 2 has tried for a long time with no success, he will give up, and start playing the original equilibrium. The entire process of trying to move and giving up is called a *moving session*.

Agent i will start with the highest satisfaction possible. The satisfaction will remain at the highest level until some event causes agent i to start a moving session. In order for the agent to become dissatisfied, he has to have a good idea of what the other agents are going to play. Therefore, agent i must have a confidence greater than γ_{MS} in order to start a moving session. Given that agent i has confidence greater than γ_{MS}, he will start a moving session in two situations. If agent i knows that all agents receive higher payoffs

at his highest known play, then he will try to move there because everyone will receive a higher payoff. Also if agent i's highest known payoff increases agent i's payoff by a large amount, and decreases the other agents' payoffs by only a small amount, then he will try to change the outcome. There are also some situations in which agent i will not start a moving session, even if his confidence is greater than γ_{MS}. If moving to agent i's highest known play will increase agent i's payoff by a small amount, but will decrease all other agents payoffs by a large amount, then agent i will not try to change the outcome. Also, if agent i has tried to move before unsuccessfully, then he will not try to move again until he has found a better strategy.

Once the moving session has started, agent i will try to induce the other agents to play his optimal strategy. If the play of the game is moving away from the play at the start of the moving session, and towards the highest play for agent i, then agent i will continue the moving session. If the play of the game does not move towards the highest play for agent i, then that round will be considered a *failure*. If the total number of failures become to high, then i will stop the moving session.

To more formally define this event that triggers a moving session, consider the term,

$$\Sigma_k^i = \frac{\pi_i\left(\bar{\mathbf{x}}_k^i\right) - \pi_i\left(\mathbf{x}_k\right)}{\frac{1}{N-1}\sum_{j\neq i}\pi_j\left(\bar{\mathbf{x}}_k^i\right) - \pi_j\left(\mathbf{x}_k\right)}$$

Σ_k^i will be referred to as the *relative gain* for agent i in round k. Agent i's payoff at the highest known play is always greater than his payoff at the current play, because the agent takes the current play into account when updating his highest known play. Therefore, switching from the current play $\mathbf{x}_k$ to agent i's highest known play $\hat{\mathbf{x}}_k^i$ will always increase agent i's payoff. So the numerator of Σ_k^i will always be weakly positive.

The agent will also keep track of the *maximum relative gain* for round k, $\bar{\Sigma}_k^i$, and the *minimum relative gain* for round k, $\underline{\Sigma}_k^i$. At the beginning of the game, agent i will start with maximum relative gain of $\bar{\Sigma}_0^i = 0$ and minimum relative gain of $\underline{\Sigma}_0^i = -1$. The agent will update these extreme relative gains with the current relative gain when the current relative gain is more extreme (higher than maximum or lower than minimum) and confidence is greater than γ_{MS}. The role of the extreme relative gains is to ensure that the agent does not continuously try to move to a point which the other agents refuse to move to.

Based on the current relative gain, the extremes relative gains, and the confidence, agent i will determine whether or not to start a moving session. When the denominator of Σ_k^i is positive, and hence $\Sigma_k^i > 0$, the other agents will benefit on average when switching from $\mathbf{x}_k$ to $\bar{\mathbf{x}}_k^i$. So, if $\Sigma_k^i > \bar{\Sigma}_k^i$ and $\gamma_k^i > \gamma_{MS}$, then the agent will start a moving session because all agents will have higher payoffs at $\bar{\mathbf{x}}_k^i$. When the denominator of Σ_k^i is negative, the other agents will get lower payoffs on average when switching from $\mathbf{x}_k$ to $\bar{\mathbf{x}}_k^i$. However, if Σ_k^i is very negative, then the average decrease of the other agents payoff will be small compared to the increase for agent i. So if $\Sigma_k^i < \underline{\Sigma}_k^i$ and $\gamma_k^i > \gamma_{MS}$

then the agent will also start a moving session. To summarize, agent i will try to move if $\Sigma_k^i \notin \left[\underline{\Sigma}_k^i, \bar{\Sigma}_k^i\right]$ and $\gamma_k^i > \gamma_{MS}$.

In the first round of the moving session, agent i will decrease from the full satisfaction level $\sigma = 1$ to the level $\sigma = \sigma_0 < 1$. Agent i will also set the number of failures to 0, $f = 0$. Agent i should not expect the other agents to respond to this move until they have seen the play in second round of the moving session and had a chance to respond to it in the third round of the moving session. So the agent will remain with satisfaction $\sigma = \sigma_0$ in the second round of the moving session, and this will not count as a failure. Starting in the third round, agent i's satisfaction and failures will depend on whether the other agents are responding to agent i's move. In particular, if the other agents are responding, and play is moving toward the highest known payoff, i.e.,

$$\|\mathbf{x}_k - \hat{\mathbf{x}}_k^i\| > \|\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1}^i\|$$

then the satisfaction will increase, $\sigma_{k+1} = \bar{\xi}\sigma_k$ and the number of failures will stay constant $f_{k+1} = f_k$ (for some $\bar{\xi} > 1$). Alternatively, if the other agents are not responding, so play is not moving toward agent i's highest known payoff, i.e.,

$$\|\mathbf{x}_k - \hat{\mathbf{x}}_k^i\| < \|\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1}^i\|$$

then the satisfaction will decrease, $\sigma_{k+1} = \underline{\xi}\sigma_k$ and the number of failures will increase by one, $f_{k+1} = f_k + 1$.

When the number of failures reaches the threshold $f_k = \bar{f}$, then the session ends because the other agents are not responding to the move. After the session ends, the amount of failures is set back to 0, and the satisfaction is set back to the highest level $\sigma = 1$.

References

Arifovic, J., and J. O. Ledyard (2005 Working Paper): "Computer Testbeds for Mechanism Design," *Caltech Working Paper*.

Camerer, C. F., T.-H. Ho, and J.-K. Chong (2004): "A Cognitive Hierarchy Model of Games," *The Quarterly Journal of Economics*, 119(3), 861–898.

Dasarathy, B. (1991): *Nearest neighbor (NN) norms: NN pattern classification techniques*. Los Alamitos, CA: IEEE Computer Society Press.

Fischbacher, U. (2007): "z-Tree: Zurich Toolbox for Readymade Economic Experiments," *Experimental Economics*, 10(2), 171–178.

Goeree, J. K., and C. A. Holt (2005): "An Experimental Study of Costly Coordination," *Games and Economic Behavior*, 51(2), 349–364.

McKelvey, R., and T. R. Palfrey (1995): "Quantal Response Equilibrium for Normal Form Games," *Games and Economic Behavior*, 10, 6–38.

Myung, N. (2008 Working Paper): "Improving Coordination and Cooperation Through Competition," *Caltech Working Paper*.

Romero, J. (2008 Working Paper): "Computational Testbeds for Coordination Games," *Caltech Working Paper*.

Sonsino, D., and J. Sirota (2003): "Strategic Pattern Recognition - Experimental Evidence," *Games and Economic Behavior*, 44, 390–411.