UNIVERSITY OF
Southampton
School of Social Sciences

Economics Division
University of Southampton
Southampton SO17 1BJ, UK

Discussion Papers in
Economics and Econometrics

Title Coevolution Of Finite Automata With Errors

By Christos A. Ioannou

No. 1019

ISSN 0966-4246

Coevolution Of Finite Automata With Errors

Christos A. Ioannou [*†]

October 26, 2011

Abstract

Errors are common in strategic situations. We use a genetic algorithm to simulate the evolution of error-prone finite automata in the repeated Prisoner's Dilemma game. In particular, the automata are subjected to implementation and perception errors. The computational experiments assess whether and how the distribution of outcomes and structures in the population changes with different levels of errors. We find that the complexity of the automata is decreasing in the probability of errors. Furthermore, the prevailing structures tend to exhibit low reciprocal cooperation and low tolerance to defections as the probability of errors increases. In addition, by varying the error-level, the study identifies a threshold. At and above the threshold, the prevailing structures converge to the open-loop (history-independent) automaton Always-Defect. On the other hand, below the threshold, the prevailing structures are closed-loop (history-dependent) and diverse, which impedes any inferential projections on the superiority of a particular machine.

JEL Classification: C72, C80, C90

Keywords: Automata, Repeated Games, Genetic Algorithms, Local Polynomial Regression

[*]I am indebted to Aldo Rustichini and Ket Richter for their continuous support and invaluable discussions. I am grateful to John H. Miller for his comments. I would also like to thank Larry Blume, David Rahman, Itai Sher, Ichiro Obara, Snigdhansu Chatterjee, Ioannis Nompelis, Shi Qi, Maria Kyriacou, Kostas Biliouris, Panayotis Mertikopoulos and Scott Page for their suggestions. Finally, I would like to thank the seminar participants at the University of Vienna, University of Minnesota, Washington University and the Santa Fe Institute. Errors are mine.

[†]Mailing Address: Department of Economics, University College London, London, WC1E 6BT, United Kingdom. Email: christos.a.ioannou@ucl.ac.uk

1 Introduction

Agents, in the real world, are not hyper-rational but engage in actions that are constrained by the limitations of human nature and the surrounding environment. Due to these disturbances, the decision-makers may occasionally draw incorrect inferences about their peers' actions which may lead to significant complications quite fast. For example, on September 1, 1983, Korean Air Lines Flight 007 strayed into prohibited Soviet airspace around the time of a planned missile test. It was shot down by the Soviets, killing all 269 people aboard. The Americans and Soviets echoed their anger at each other in a short, but sharp, escalation of cold war tensions (Goldstein 1991).

In this paper, we use the genetic algorithm to simulate an evolving, error-prone population that plays the repeated Prisoner's Dilemma (PD) paradigm. The primary objective of the study is to use computational experiments that incorporate different levels of errors to assess whether and how the distribution of outcomes and strategies in the population changes. Our analysis utilizes a novel tool, local polynomial regression, which has recently gained a considerable amount of interest in social sciences. A secondary objective of the study is to identify and discuss behavioral patterns that fare well in the simulated environments.

According to the thought experiment, a group of agents is set to play the PD game. Each agent is required to submit a strategy that is implemented by a type of finite automaton called a *Moore machine* (Moore 1956). The machine specifies actions contingent upon the opponent's reported actions. The agents play the PD game against each other and against their twin in a round-robin structure. The machines of the agents are subjected to implementation and perception errors. *Implementation errors* are errors in the implementation of actions along the lines of Selten's trembling hand (Selten 1975). Moreover, imperfect monitoring takes the form of *perception errors*; that is, errors in the transmission of information. With the completion of all round-matches, the actual scores and machines of every agent become common knowledge. Based on this information, agents update their machines for the next generation.

Under the proposed framework, the incorporation of implementation and perception errors is sufficient to reduce cooperative outcomes. In particular, we find that the prevailing structures tend to exhibit low reciprocal cooperation and low tolerance to defections as the probability of errors increases. Furthermore, the *complexity* of the machines, defined as the number of

accessible states, is decreasing in the probability of errors. In addition, by varying the error-level, the study identifies a threshold. At and above the threshold, the prevailing structures converge to the open-loop (history-independent) automaton Always-Defect.[1] On the other hand, below the threshold, the prevailing structures are closed-loop (history-dependent) and diverse, which impedes any inferential projections on the superiority of a particular machine.

The study aims to elicit an understanding of the patterns of reasoning of agent-based behaviors in the presence of errors. Conventional game theory rests on the foundations that agents are hyper-rational with full ability to select the most-preferred action. Yet, the latter is rarely justified as an empirically realistic assumption. Rather, it is usually defended on methodological grounds as the appropriate theoretical framework to analyze behavior. On the contrary, the incorporation of errors in the proposed context is a viable alternative and consequently, one that merits further investigation. Having said this, we do hope, more than our specific findings and interpretations, that this work will help move computational research away from the study of misleading, hyper-rational agents and towards the study of agents with human-like characteristics and qualities.

The rest of the paper is organized as follows. In Section 2, we review the related literature while in Section 3, the concept of a Moore machine as the carrier of an adaptive agent's strategy is presented. In Section 4, we explain the methodology. In Section 5, we focus on the results of the evolutionary process while elaborating on broad characteristics of the evolutionary machines. In Section 6, we discuss the properties of the machines which are then compared to the results of previous studies. Finally, in the Conclusion, we offer direction for future research.

2 Literature Review

The study is related to several strands of literature. First, it is related to the large literature on optimization routines. The genetic algorithm (Holland 1975) is one of many search techniques developed for solving hard combinatorial optimization problems in large search spaces. Other

[1] Oscar Volij (2002) provides a theoretical framework which confirms that the only evolutionary stable strategy is Always-Defect when agents' preferences are lexicographic; first, according to the limit of the means criterion, and second, according to the complexity of the automaton.

optimization techniques include: Simulated Annealing (Kirkpatrick, Gelatt and Vecche 1983), Tabu Search (Glover and Laguna 1993), Stochastic Hill Climbing and Compset Algorithm (Hamo and Markovitch 2005). Axelrod (1987) was the first to model the evolutionary process of the repeated PD game with a genetic algorithm. Nevertheless, his study was restricted by his use of strategies whose actions were contingent to the action profiles of (only) the last three periods, and by his use of a fixed environment composed of (only) eight strategies. On the other hand, Miller (1996) circumvented these restrictions by the use of a variable environment where strategies co-evolved as the strategic population changed. In addition, Miller used automata to enable the definition of many theoretically important strategies (for example, strategies relying on counting or triggers) that could not be defined under the framework of Axelrod. Miller's approach is used in this formulation as well. Using finite automata as the carriers of agents' strategies was first suggested by Aumann (1981) for the study of decision-making with bounded rationality. The first application originated in the work of Neyman (1985) who investigated a finitely-repeated game model in which the pure strategies available to the agents were those which could be generated by machines utilizing no more than a certain number of states.

Additionally, several researchers have studied the effect of complexity on the set of equilibria in repeated games with finite automata. Abreu and Rubinstein (1988) for one, showed that if agents' preferences are increasing in repeated-game payoffs and decreasing in the complexity of the strategies employed, then the set of Nash-equilibrium payoffs that can occur is dramatically reduced from the folk-theorem result (Fudenberg and Maskin 1986). Yet, the authors indicate that a wide variety of payoffs remains consistent with equilibrium behavior in the presence of complexity costs, including the strategy Always-Defect. Binmore and Samuelson (1992) in their seminal work, motivated by the non-existence of an evolutionary stable strategy in the PD game (see Boyd and Lorberbaum 1987) propose a modified evolutionary stable strategies' solution concept. Under this solution concept, the strategy Always-Defect ceases to persist in equilibrium. Foster and Young (1990) on the other hand, develop the concept of stochastic stability which requires a population to be immune to persistent random mutations. Stochastic stability has been successfully applied by Kandori, Mailath and Rob (1993) in the analysis of symmetric 2×2 games, by Vega-Redondo (1997) in the analysis of competition among firms, and by Ben-Shoham, Serrano and Volij (2004) in the analysis of a housing problem. Yet, Bergin and Lipman (1996) point out a weakness of the concept of stochastic stability; the

actions selected out of the recurrent classes depend on the rate of mutation. It is noteworthy that Abreu and Rubinstein (1988), just like the other papers mentioned, define the complexity of a strategy as the size of the minimal automaton implementing it. On the other hand, Banks and Sundaram (1990) argue that the traditional number-of-states measure of complexity of an automaton neglects some essential features such as informational requirements at a state. They propose instead, a criterion of complexity which takes into account both the size (number of states) and transitional structure of a machine. Under this proposition, they prove that the resulting Nash equilibria of the machine-game are now trivial: the machines recommend actions in every period that are invariably stage-game Nash equilibria.

Finally, the current work builds on the computational simulations' literature. Robert Axelrod pioneered this area, with the computational tournaments he conducted to determine the best strategy in the repeated PD game. The results were a clear victory for Tit-For-Tat (TFT); a strategy that starts off by cooperating and then imitates the most recent action of the opponent. Bendor, Kramer and Stout (1991) have been, to our knowledge, the first to conduct a computer tournament with random shocks. In their study, the authors re-evaluate the performance of reciprocating strategies such as TFT and identify alternative strategies that sustain cooperation in an environment with random shocks. The computer tournament is constructed in a manner similar to the tournaments of Axelrod. The winning strategy in their tournament is Nice-And-Forgiving (NAF) which differs in many ways from TFT.[2] First, NAF is nice in the sense that it cooperates as long as the frequency of cooperation of the opponent is above some threshold. Second, NAF is forgiving in the sense that although NAF retaliates if the opponent's cooperation falls below the threshold level of cooperation, it reverts to full cooperation before its opponent does, as long as certain minimal levels of cooperation are met by the opponent.

3 Finite Automata

A finite automaton is a mathematical model of a system with discrete inputs and outputs. The system can be in any one of a finite number of internal configurations or "states." The state of the system summarizes the information concerning past inputs that is needed to determine the

[2]TFT's performance was mediocre, placing eight out of thirteen strategies.

behavior of the system on subsequent inputs. The specific type of finite automaton used here is a Moore machine (Moore 1956). Let I denote the set of agents, A^i denote the set of i's actions, A denote the cartesian product of the action spaces written as $A \equiv \overset{I}{\underset{i=1}{\times}} A^i$, and $g^i : A \to \Re$ denote the real-valued utility function of i. Thus, a *Moore machine* for an adaptive agent i in a repeated game of $G = (I, \{A^i\}_{i \in I}, \{g^i\}_{i \in I})$ is a four-tuple $(Q^i, q_0^i, f^i, \tau^i)$ where

- Q^i is a finite set of internal states,
- q_0^i is specified to be the initial state,
- $f^i : Q^i \to A^i$ is an output function that assigns an action to every state, and
- $\tau^i : Q^i \times A^{-i} \to Q^i$ is the transition function that assigns a state to every two-tuple of state and other agent's action.[3]

In the first period, the state is q_0^i and the machine chooses the action $f^i(q_0^i)$. If a^{-i} is the action chosen by the other agent in the first period, then the state of agent i's machine changes to $\tau^i(q_0^i, a^{-i})$, and in the second period agent i chooses the action dictated by f^i in that state. Then, the state changes again according to the transition function given the other agent's action. Thus, whenever the machine is in some state q, it chooses the action $f^i(q)$ while the transition function τ^i specifies the machine's transition from q (to a state) in response to the action taken by the other agent.

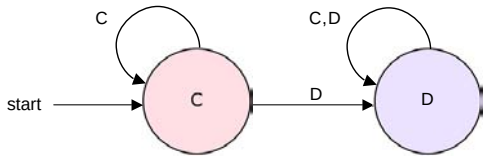

Figure 1: Grim-Trigger machine

[3]It is pertinent to note that the transition function depends only on the present state and the other agent's action. This formalization fits the natural description of a strategy as agent i's plan of action in all possible circumstances that are consistent with agent i's plans. On the other hand, the notion of a game-theoretic strategy for agent i requires the specification of an action for every possible history, including those that are inconsistent with agent i's plan of action. To formulate the game-theoretic strategy, one would have to construct the transition function such that $\tau^i : Q^i \times A \to Q^i$ instead of $\tau^i : Q^i \times A^{-i} \to Q^i$.

$Q^i = \{q_C, q_D\}$
$q_0^i = q_C$
$f^i(q_C) = C$ and $f^i(q_D) = D$
$\tau^i(q, a^{-i}) = \{^{q_C \quad (q,a^{-i})=(q_C,C)}_{q_D \quad otherwise}$

For example the machine (Q^i, q_0^i, f^i, τ^i) in Figure 1, carries out the Grim-Trigger strategy in the context of the PD game. Thus, the strategy chooses C so long as both agents have chosen C in every period in the past, and chooses D otherwise. In the transition diagram, the vertices denote the internal states of the machine and the arcs labeled with the action of the other agent, indicate the transition to the states.

The study considers errors in the implementation of actions and errors in the perception of actions. Implementation and perception errors when considered in isolation lead to quite different results. For instance, the machine Contrite-Tit-For-Tat in the repeated PD game is proof against errors in implementation but not against errors in perception. The machine acts in principle as Tit-For-Tat, but enters a "contrite" state if it erroneously implements a defection rather than a cooperation. Consequently, the machine accepts the opponent's retaliation and cooperates for the next two periods but leaves the contrite state soon after. On the other hand, if the machine Contrite-Tit-For-Tat mistakenly perceives that the opponent defected, will respond with a defection without switching to the contrite state and will not meekly accept any subsequent retaliation. It is therefore important to formally define implementation and perception errors in the context of Moore machines.

Definition 1 *The machine of agent i in the PD game commits an implementation error with probability ϵ, when for any given state q, the machine's output function returns the action $f^i(q)$ with probability $1-\epsilon$ and draws another action "$f^i(q)$" where $f^i(q) \neq$ "$f^i(q)$"* otherwise.[4]

That is, an implementation error level of ϵ indicates that with probability ϵ the course of action dictated by the particular state of the machine will be altered. For example, a

[4]A more general definition would postulate that *the machine of agent i commits an implementation error with probability ϵ, when for any given state q, the machine's output function returns the action $f^i(q)$ with probability $1-\epsilon$ and draws another action $a^i \in A^i \setminus f^i(q)$ randomly and uniformly otherwise.* However, since the action space in the PD game consists of only two actions, the former definition suffices.

cooperation dictated by the particular state will be implemented erroneously as a defection with probability ϵ. On the other hand, perception errors are defined as follows.

Definition 2 *The machine of agent i in the PD game commits a perception error with probability* δ, *when for any given opponent's action* a^{-i}, *the machine inputs the opponent's action* a^{-i} *into the transition function with probability* $1-\delta$ *and inputs the opponent's action* "a^{-i}" *into the transition function* where $a^{-i} \neq$ "a^{-i}" otherwise.

Thus, a perception error level of δ indicates that with probability δ an opponent's action is reported incorrectly, while with probability $1-\delta$ the opponent's action is perfectly transmitted. Furthermore, the study considers machines that hold no more than eight internal states. The choice to keep the upper bound on the number of internal states at eight is a considered decision. First, such a bound is reasonable given complexity considerations. As Rubinstein (1986) indicates, agents seek to devise behavioral patterns which do not need to be constantly reassessed and which economize on the number of states needed to operate effectively in a given strategic environment. A more complex plan of action is more likely to break down, is more difficult to learn, and may require more time to be executed. In fact, a number of studies (some with subjects in the laboratory) have been suggestive of the effectiveness of simple strategies over more complex ones in a wide range of environments (Axelrod 1984; Rust, Miller and Palmer 1994; Selten, Mitzkewitz and Uhlich 1997). Second, the choice of eight internal states allows for a sufficient variety of machines to emerge that incorporate a diverse array of characteristics.[5]

4 Methodology

The mechanics of the genetic algorithm involve copying strings and altering states through the operators of selection and mutation. Each generation starts with a given population called the *parent population*. A new population of the same size is then constructed called the *offspring population*. In our formulation, the algorithm operates with a population of machines. Initially, a population of thirty machines is chosen at random. Then, each machine is tested against the

[5]To test the robustness of the results with respect to the size selection, computational experiments with machines that held 16 states were conducted. The computations performed, confirm that the results are robust.

	Cooperate	Defect
Cooperate	3,3	0,5
Defect	5,0	1,1

Table 1: Prisoner's Dilemma Matrix

environment (which is composed of the other machines and its twin). The game-play occurs for 200 periods per match. Each machine, thus aggregates a raw score based on the payoffs illustrated in Table 1. The offspring population is constructed from the parent population, by selecting the machines that aggregated the top twenty scores. In addition, ten new structures are created via a process of selection and mutation. The process requires the draw of ten pairs of machines from the parent population (with the probabilities biased by their scores) and the selection of the better performer from each pair. Then, these ten machines undergo a process of mutation.[6] Mutation occurs when an element at a random location on the selected string changes value. Each element on the string is subjected to a 4% independent chance of mutation, which implies an expectation of 1 element-mutation per string.[7] The population is iterated for 500 generations. The adaptive plan is summarized in Figures 2 and 3.[8]

[6]Bergin and Lipman (1996) indicate that mutation rates intent to represent either experimentation or computational errors. In our context, mutation rates represent experimentation. In addition, we assume that the mutation rate is constant across strings, across agents and over time.

[7]A higher mutation rate increases the variation in the population; as a result, in such an environment the machine Always-Defect is hugely favored. The computational simulations performed with higher rates of mutation (8% and 12% independent chance of mutation per element which imply an expectation of 2 and 3 elements-mutation per string, respectively) confirm this claim. On the other hand, the choice of a 4% independent chance of mutation is a conservative one that allows forms of innovation to appear in the structure of the machines while controlling for the variation in the population.

[8]For a more detailed discussion on genetic algorithms refer to the book by Goldberg (1989). No previous knowledge of genetic algorithms is required to understand the description of this genetic algorithm.

```
Specify error-level
Fix max-periods = 200

Create initial population: 30 agents (seed randomly)
Initiate round-robin tournament

For t = 1 to 500 do

    For all agent-pairs do
        For p = 1 to max-periods do
            Award utils to each agent based on the PD matrix
        End loop

        Output performance score
    End loop

    Apply subroutine for the offspring-population-creation
    Store agent results

End loop
```

Figure 2: Pseudocode of the Main Program

```
Sort agents based on performance score

Copy top 20 agents to offspring-population

Select 10 agent-pairs via probabilities biased by performance scores

For each of 10 pairs do
    Create new agent as a copy of the winner of the pair's match
    Mutate new agent by switching one element at random

End loop
```

Figure 3: Subroutine of the Offspring-Population-Creation

5 Results

In order to assess how the distribution of outcomes and structures in the population changes with different levels of errors, four computational experiments were conducted. In particular, the machines are subjected to a constant independent chance of implementation and perception errors of 4%, 2%, 1% and 0%, respectively. In the absence of a theoretical background on the functional form of the model, non-parametric methods are used to carry out the estimation. More specifically, local polynomial regression was used to fit the dependent variables over the course of the evolution.[9] The dependent variables were found to exhibit non-stationary behavior which is attributed to the selection dynamics of the genetic algorithm. All the smoothed curves presented next, utilize first-ordered polynomial functions which are found in the literature to be quite effective in balancing the bias-variance trade-off. In addition, the Epanechnikov-Kernel weighting function was used. The bandwidth was chosen via the (data-based) rule-of thumb bandwidth. The latter was preferred, in the absence of any periodical patterns on the way agents behave across generations.

5.1 Evolution Of Payoffs

In Figure 4, local polynomial regression is used to fit the average payoffs[10] over the course of the evolution. The payoffs are found to exhibit non-stationarity which can be attributed to the dependence of the generational selection. In the early generations, the agents tend to use machines that defect continuously. The reason is that at the start of the evolution, the machines are generated at random. In such an environment, the best strategy is to always defect. With the lapse of a few generations though, machines in the less error-prone treatments achieve consistent cooperation which allows the payoffs to move higher. The paired-differences tests establish that the means of the treatments are statistically different at a 99% level of significance. Thus, the incorporation of implementation and perception errors is sufficient to alter the evolution of cooperative outcomes.

[9] A discussion on local polynomial regression can be found in the Appendix.

[10] The average payoffs of a given generation t are calculated as the sample average of the payoffs of the thirty members of the population, and over the thirty simulations conducted for each treatment.

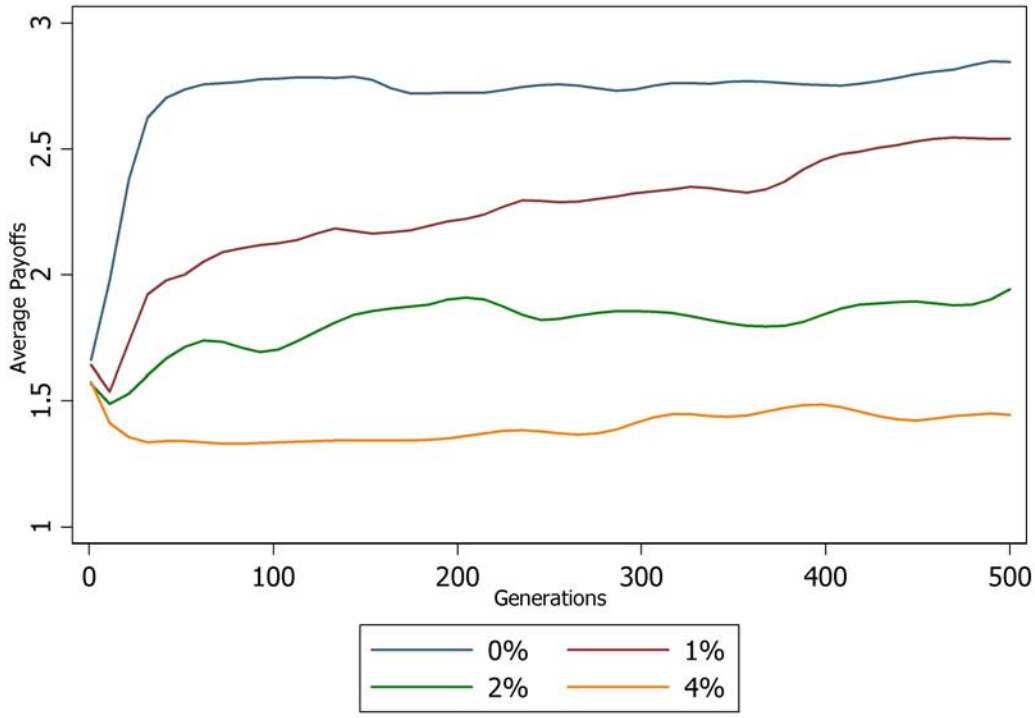

Figure 4: Average Payoffs

5.2 Prevailing Machines

The incorporation of different levels of errors in the analysis provides a richer framework to address the impact of errors on the structure of the prevailing machines. The clear winner in the 4% and 2% treatments was the machine Always-Defect. Always-Defect was the winner in 22 out of the 30 simulations run in the 4% treatment, and in 19 out of the 30 simulations run in the 2% treatment. The machine Always-Defect is presented in Figure 5. Always-Defect is an open-loop machine; that is, the actions taken at any time-period do not depend on the actions of the opponent.

On the other hand, the structures that prevailed in the 1% and 0% treatments were diverse. This result halts any possible attempt to discern a particular behavioral pattern that fares well in these specific treatments. Yet, it is noteworthy that unlike the open-loop machine Always-Defect, the diverse array of machines that prevailed in the 1% and 0% treatments were all closed-loop (history-dependent). Thus, the effect of different error-levels on the structure

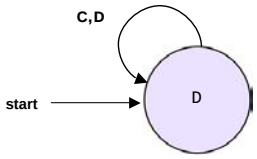

Figure 5: Always-Defect

of the machines points towards the existence of a threshold error-level at 2%. At and above the threshold, the prevailing structures converge to the open-loop automaton Always-Defect, whereas below the threshold, the prevailing structures are closed-loop and diverse.

5.3 Automaton Characteristics

Here, we ascertain broad characteristics of the structures that work "reasonably well." To that extent, it is necessary to use some summary measures of machine-behavior. The first summary measure is the size of the machine, which is measured by its number of accessible states. A *state is accessible* if, given the machine's starting state, there is some possible combination of the opponent's possible actions that will result in a transition in that state. Therefore, even though all machines are defined on eight states, some of these states may never be reached. Other summary measures that shed more light to the machine-composition are the cooperation-reciprocity and the defection-reciprocity. The *cooperation-reciprocity* is the proportion of accessible states that respond to an observed cooperation by the opponent with a cooperation. On the other hand, the *defection-reciprocity* is the proportion of accessible states that respond to an observed defection by the opponent with a defection. Consequently, in this subsection, we identify characteristics that are important to the survival of the machines in the simulated environments. A *characteristic* is a property of the machines whose presence can be objectively determined by an examination of the corresponding graph. Noticeably, the characteristics listed are also indicators of strategic ideas underlying the machines.[11]

In Figure 6, local polynomial regression is used to fit the average number of accessible states per generation of the four treatments. Over the course of the evolution the average number of

[11] All the statistical tables based on the tests performed can be provided by the author upon request.

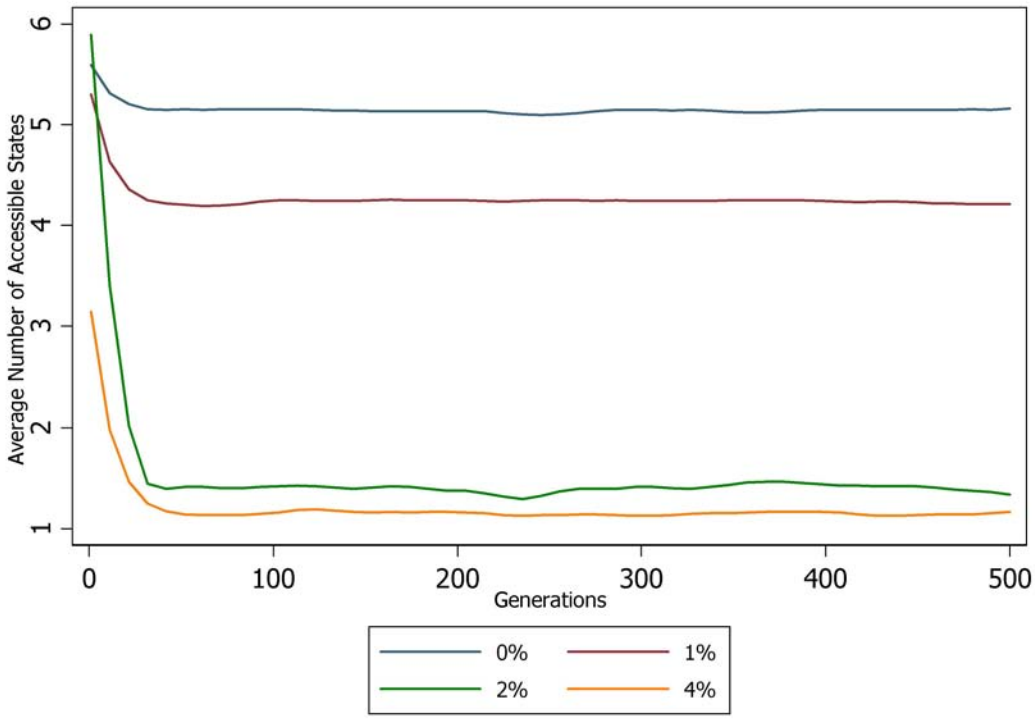

Figure 6: Average Number of Accessible States

accessible states is consistently less in the more error-prone environments. This suggests that under more error-prone treatments, the average number of accessible states drops. Thus, if we assume that the average number of accessible states is a good measure of machine-complexity, then a possible conclusion is that strategic simplification is advantageous in the presence of errors. The first descriptive characteristic of the machines is summarized below.

CHARACTERISTIC 1: *The average number of accessible states of the machines is decreasing in the probability of errors.*

The average cooperation-reciprocity of the four treatments is fit into a local polynomial regression in Figure 7. As the likelihood of errors increases, the machines tend to reduce their cooperation-reciprocity. This observation also accounts for the low payoffs of Figure 4. An alternative perspective to consider is the proportion of accessible states that respond to an observed cooperation by the opponent with defection; this proportion is given by the expression $1 - \textit{cooperation-reciprocity}$. The latter expression can be used as a proxy for the degree of

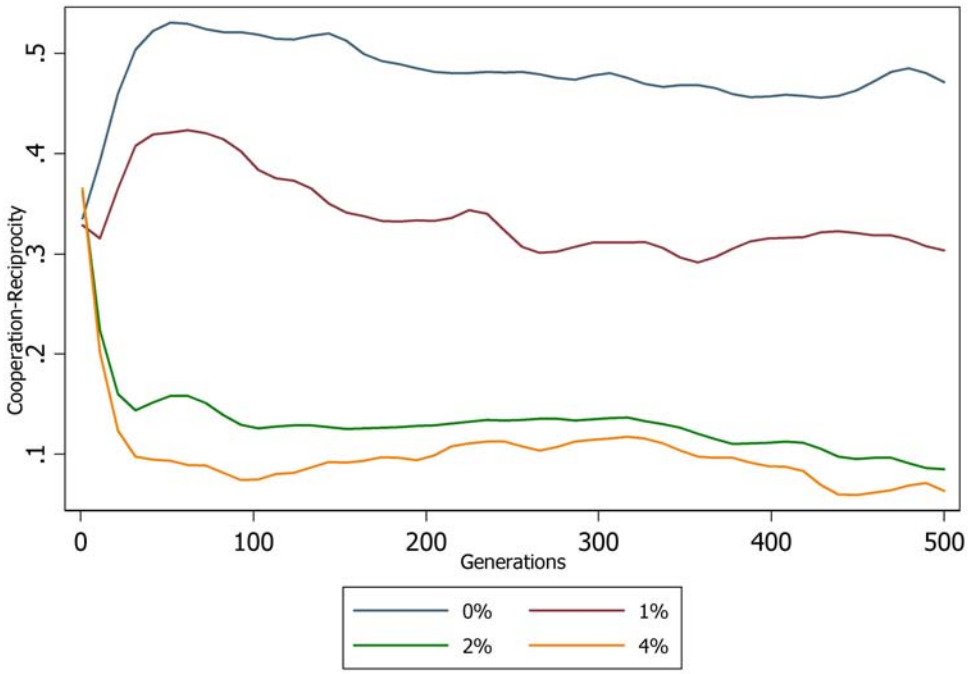

Figure 7: Cooperation-Reciprocity

sneakiness of the machines. Thus, machines with relatively low cooperation-reciprocity are also relatively more sneaky, in the sense that these machines incorporate states that shoot for the "temptation" payoff more often. Such exploitation can be camouflaged in the presence of errors, but not in their absence. The second descriptive characteristic is summarized next.

CHARACTERISTIC 2: *The cooperation-reciprocity of the machines is decreasing in the probability of errors or, alternatively, the degree of sneakiness is increasing in the probability of errors.*

The defection-reciprocity of the four treatments is fit into a local polynomial regression in Figure 8. The pattern deduced is that defection is not tolerated in the error-prone environments. It is noteworthy that machines in general, are more likely to reciprocate a defection by the opponent with a defection than to cooperate after the opponent cooperates. On the other hand, the proportion of accessible states that respond to an observed defection by the opponent with cooperation is $1 - defection\text{-}reciprocity$. This expression signifies the degree of forgiveness of the machines. For example, a sizable $1 - defection\text{-}reciprocity$ indicates that the machines on average, incorporate quite a few states that respond to observed defections by the opponent

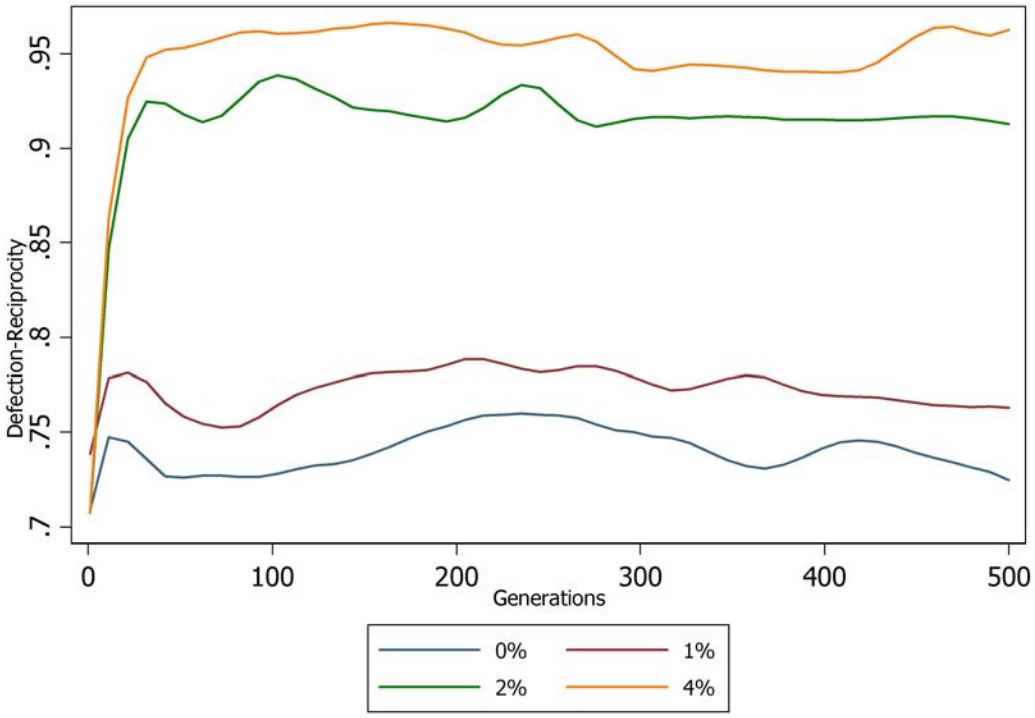

Figure 8: Defection-Reciprocity

with cooperation. Yet, the computational experiments of Figure 8 indicate otherwise. The third descriptive characteristic is summarized below.

CHARACTERISTIC 3: *The defection-reciprocity of the machines is increasing in the probability of errors or, alternatively, the degree of forgiveness is decreasing in the probability of errors.*

6 Discussion

TFT was the winner in the tournaments with error-free strategies of Robert Axelrod (1984). The performance of TFT lead Axelrod to identify some basic attributes that were necessary for the emergence and survival of cooperation. These were: (i) an avoidance of unnecessary conflict by cooperating as long as the other agent does, (ii) provocation in the face of an uncalled for defection by the other, (iii) forgiveness after responding to a provocation, and (iv)

clarity of behavior so that the other agent can adapt to your pattern of action. On the other hand, Bendor, Kramer and Stout (1991) incorporated in their computer tournament random shocks. The winning strategy in that tournament was Nice-And-Forgiving (NAF) which differed in many ways from TFT. First, NAF was nice in the sense that it cooperated as long as the frequency of cooperation of the opponent was above some threshold. Second, NAF was forgiving (generous) in the sense that although NAF would retaliate if the opponent's cooperation fell below the threshold level of cooperation, it would revert to full cooperation before its opponent did, as long as certain minimal levels of cooperation were met by the opponent.

Yet, the success of NAF is not a robust result but is limited to the particular ecology. As Bendor, Kramer and Stout (1991) note, the generosity of NAF creates a risk: other strategies may exploit NAF's willingness to give more than it receives. In other words, NAF can be suckered by a nasty strategy that is disinterested in joint gains.[12] As Axelrod and Dion (1988) aptly note, "in the presence of large amounts of errors, there is a trade-off: unnecessary conflict can be avoided by generosity, but generosity invites exploitation." Thus, strategies that are unilaterally generous will not fare well in every ecology.

Another objection to NAF's generosity is the lack of generalizability of this finding. With the exception of social-welfare models, where agents are willing to exhibit generosity to increase social surplus, other models do not establish the presence of ad hoc generosity in the agents' decisions. In the difference-aversion models for example, agents may be particularly sensitive and reactive to any indication that the other party is doing better or coming out ahead. Furthermore in the reciprocity models, an agent's generosity is likely to be influenced by the other party's behavior. Many recent experimental studies have demonstrated that the willingness to be generous in games is sensitive not only to the choice set available to the agent contemplating an action, but also to the behavior of the other agent that generated that choice set (see for example, Ioannou, Qi and Rustichini 2011). Consequently, when an unfavorable outcome is attributed to the other party's greed, individuals will abandon cooperation. Finally in the competitive (self-interest) models, agents approach the game with a preference for outcomes

[12]Due to its generosity NAF lost in its pairwise play with every one of its opponents. In contrast to NAF's pattern, VIGILANT, the strategy that placed dead last in the tournament, beat every one of its partners in bilateral play. VIGILANT was a highly provocable and unforgiving strategy that retaliated sharply if it inferred that its partner was playing anything less than maximal cooperation.

that maximize the difference between the parties. Thus, agents with a competitive motive may be particularly unlikely to regard generosity as an attractive or viable strategy in a PD game.

In addition, laboratory research has identified several psychological factors that might diminish generosity among human strategists. First, the fear of exploitation or desire to avoid the "sucker's" payoff may make it difficult for individuals to risk generosity. Experimental studies of the PD game in particular, have found that agents will often cooperate until they have evidence or even the mere suspicion that the other party is taking advantage of them (see discussions in Dawes and Thaler 1988). The fear of exploitation may induce individuals to engage in a kind of defensive "stinginess" even though they recognize the merits of generosity. Furthermore, Pedro Dal Bo and Guillaume Frechette (2011) provide compelling experimental evidence to suggest that even in treatments where cooperation can be supported in equilibrium, the level of cooperation may remain at low levels even after significant experience is obtained. The authors conclude that, "these results cast doubt on the common assumption that agents will make the most of the opportunity to cooperate whenever it is possible to do so in equilibrium."

On the other hand, the summary measures of the present study point to a very different direction from that in Bendor, Kramer and Stout (1991). In the computational experiments, the cooperation-reciprocity of the machines is relatively low reflecting their readiness to exploit (to sneak on the opponent). The reason is that an attempt to acquire the "temptation" payoff can be camouflaged in the presence of errors but not in their absence. Furthermore, in sharp contrast to NAF, the machines that evolve here are relentless punishers of defections. In fact, the defection-reciprocity of the machines climbs to a proportion close to 1 in the environments with high levels of errors indicating their lack of forgiveness to defections.

Of importance is also the finding that the size of the automaton is decreasing in the probability of errors. Thus, it is safe to infer that in the presence of errors, strategic simplification is advantageous (if the number of accessible states is assumed to be a good measure of complexity). In the absence of errors, the behavior of well-informed agents responding with flexibility to every perturbation in the environment does not produce easily recognizable patterns, but rather is extremely difficult to predict. On the other hand, in the presence of errors, behavior is governed by mechanisms that restrict the flexibility to choose potential actions. These mechanisms simplify behavior to less complex patterns, which are easier for an observer to recognize and predict.

In real-life, there exist a number of examples where an agent's choice of a simple strategy (a rule of thumb) in the presence of errors, has proved quite successful. An interesting example is the publishing history on strategies to win at blackjack. Books in the 60s and 70s emphasized sophisticated card-counting and bet-variation methods (see especially Edward Thorpe's book, Beat the Dealer). However, while no one has challenged the mathematical validity of these earlier more complex methods, their actual use resulted in worse performance by most persons attempting to use them (which generated sizable unexpected profits to the casinos). As a result, later books have steadily evolved towards more rigidly-structured methods (for example, the books No Need to Count and Winning Casino Blackjack for the Non-Counter).

7 Conclusion

The study indicates, via an explicit evolutionary process simulated by the genetic algorithm, that the incorporation of implementation and perception errors is sufficient to alter the evolution of cooperative machines. In particular, we find that the prevailing structures tend to exhibit low reciprocal cooperation and low tolerance to defections as the probability of errors increases. Furthermore, the complexity of the machines is decreasing in the probability of errors, which suggests that strategic simplification is advantageous in the presence of errors. In addition, the study identifies a threshold error-level. Below the threshold, the prevailing structures are closed-loop and diverse, which impedes any inferential projections on the superiority of a particular machine. On the other hand, at and above the threshold, the prevailing structures converge to the open-loop machine Always-Defect. The latter finding makes it pertinent to seek ways to limit the level of errors in our strategic interactions to avoid suboptimal outcomes. Revisiting our example in the introduction, president Ronald Reagan, soon after the disaster ordered the U.S. military to make the developing Global Positioning System (GPS) available for civilian use, so that navigational errors like that of Korean Air Lines Flight 007 could be averted in the future.

A wide variety of potential extensions exist. In this study, it was assumed that all machines were subjected to the same likelihood of errors, generation by generation. Instead, the likelihood of errors could be contingent upon the number of states accessed by the particular machine. The

rationale is that a machine that contains only one state (say, Always Cooperate) is less likely to commit errors than one that is far more complex. Furthermore, it would be interesting to examine whether the results are robust to the symmetry of the payoffs. One of the basic features of the conventional Prisoner's Dilemma stage-game is the requirement that the values assigned to the game are the same for both agents. Not uncommon however, are social transactions where not only is each agent's outcome dependent upon the choices of the other, but also where the resources and therefore possible rewards of one agent exceed those of the other. A social interaction characterized by a disparity in resources and potentially larger rewards for one of the two participants would in all likelihood call into play questions of inequality. Thus, one could run two co-evolving populations with asymmetric payoffs to see how the asymmetry in payoffs affects cooperation under the simulated environments.

Appendix

The traditional nonlinear regression model fits

$$y_i = f(\beta, \mathbf{x_i'}) + \epsilon_i$$

where $\beta = (\beta_1, ..., \beta_p)$ is a vector of parameters to be estimated, and $\mathbf{x_i} = (x_1, ..., x_k)$ is a vector of predictors for the i^{th} of n observations. The errors ϵ_i are assumed to be normally and independently distributed with mean 0 and constant variance σ^2. On the other hand, the non-parametric regression model is written in a similar manner, but the function f is left unspecified.

$$y_i = f(\mathbf{x_i'}) + \epsilon_i$$

Whilst in conventional parametric modeling the deterministic component of the model is defined by a single parameter for every value of x, non-parametric regression does not impose such restrictions. Instead, within local regression, all assumptions regarding the functional form of the model are discarded. Thus, the value of the deterministic part changes with the value of x as the regression coefficient is estimated "locally" at a pre-specified *sliding window*. Once the *bandwidth* $h(x)$ is defined, the sliding window which is centred as some point x within which the regression takes place is defined as $[x - h(x),\ x + h(x)]$. Within this smoothing window centred at point x, the regression component is approximated with the following polynomial of order p:[13]

$$\mu(x) = \beta_0 + \beta_1(u - x) + \ldots + \beta_p(u - x)^p$$

The estimates of the parameter vector $[\beta_0, \ldots, \beta_p]$ are obtained from the minimization of the following objective function:

$$min \quad \sum_{i=1}^{n} w_i(x)(y_i - (\beta_0 + \beta_1(u - x) + \ldots + \beta_p(u - x)^p))^2$$

where the weights $w_i(x) = W(\frac{x_i - x}{h(x)}) \geq 0$ are derived from the weight function; the latter, assigns the largest weights to the observations closer to the central point x. The most commonly used weight function is the Epanenchnikov-Kernel weight function (Epanenchnikov 1969) which is

[13] β_0 is usually referred to as the level, and p denotes the order of the polynomial.

defined in the following manner:[14]

$$W(z) = \frac{3}{4}(1 - z^2) \; for \; |z| \leq 1 \;\; and \;\; 0 \; for \; |z| > 1.$$

[14]Other choices of Kernel weight functions such as Gaussian, Parzen and Cosine are made available in widely used statistical software.

References

[1] Abreu, D., and Rubinstein, A. (1988). "The Structure of Nash Equilibrium in Repeated Games with Finite Automata," *Economerrica* 56, 1259-1282.

[2] Aumann, R. (1981). "Survey of Repeated Games," In Essays in Game Theory and Mathematical Economics in Honor of Oskar Morgenstern, Mannheim, Bibliographisches Institut.

[3] Axelrod, R. (1980). "Effective Choice in the Prisoner's Dilemma," *Journal Of Conflict Resolution* 24, 3-25.

[4] Axelrod, R. (1980). "More Effective Choice in the Prisoner's Dilemma," *Journal Of Conflict Resolution* 24, 379-403.

[5] Axelrod, R. (1984). The Evolution of Cooperation, Basic Books: New York.

[6] Axelrod, R. (1987). "The Evolution of Strategies in the Iterated Prisoner's Dilemma," In Lawrence Davis, Los Altos, California, Morgan Kaufmann.

[7] Axelrod, R., and Dion, D. (1988). "The Further Evolution of Cooperation," *Science* 242, 1385-90.

[8] Axelrod, R. and Hamilton, W. D. (1981). "The Evolution of Cooperation," *Science* 211, 1390-1398.

[9] Banks, J. S., and Sundaram, R. K. (1990). "Repeated Games, Finite Automata, and Complexity," *Games and Economic Behavior* 2, 97-117.

[10] Bendor, J., Kramer, R. and Stout, S. (1991). "When in Doubt... Cooperation in a Noisy Prisoner's Dilemma," *Journal Of Conflict Resolution* 35, 691-719.

[11] Ben-Shoham, A., Serrano, R., and Volij O. (2004). "The Evolution of Exchange," *Journal of Economic Theory* 114, 310-28.

[12] Bergin, J., and Lipman, B. L. (1996). "Evolution with State-Dependent Mutations," *Econometrica* 64, 943-56.

[13] Binmore, K. G., and Samuelson, L. (1992). "Evolutionary Stability in Repeated Games Played by Finite Automata," *Journal of Economic Theory* 57, 278-305.

[14] Boyd, R., and Loberbaum, J. (1987). "No Pure Strategy is Evolutionary Stable in the Repeated Prisoner's Dilemma Game," *Nature* 327, 58-9.

[15] Cardoza, A. D. (1981). Winning Casino Blackjack for the Noncounter, Cardoza School of Blackjack: Santa Cruz.

[16] Dal Bo, P., and Frechette, G.R. (2011). "The Evolution of Cooperation in Infinitely Repeated Games: Experimental Evidence," *American Economic Review* 101, 411-29.

[17] Dawes, R. M., and Thaler, R.H. (1988). "Anomalies: Cooperation," *Journal of Economic Perspectives* 2, 187-97.

[18] Dubey, L. B. (1980). No Need to Count: A Practical Approach to Casino Blackjack, A.S. Barnes & Co: San Diego.

[19] Epanechnikov, V. A. (1969). "Non-parametric Estimates of a Multivariate Probability Density," *Theory of Probability and its Applications* 14, 153-158.

[20] Foster, D.P., and Young, H. P. (1990). "Stochastic Evolutionary Game Dynamics," *Theoretical Population Biology* 38, 219-32.

[21] Fudenberg, D., and Maskin, E. (1986). "The Folk Theorem in Repeated Games with Discounting and with Incomplete Information," *Econometrica* 54, 533-554.

[22] Glover F., and Laguna M. (1993). "Tabu Search," In Modern Heuristic Techniques for Combinatorial Problems, C.R. Reeves.

[23] Goldberg, D. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company.

[24] Goldstein, J. (1991). "Reciprocity in Superpower Relations: An Empirical Analysis," *International Studies Quarterly* 35, 195-209.

[25] Hamo, Y., and Markovitch, S. (2005). "The Compset Algorithm for Subset Selection," In Proceedings of The Nineteenth International Joint Conference for Artificial Intelligence, 728-733.

[26] Holland, J. (1975). Adaptation in Natural and Artificial Systems, MIT Press.

[27] Ioannou, C. A., Qi, S., and Rustichini, A. (2011). "Group Outcomes and Reciprocity," Working Paper, University College London.

[28] Kandori, M., Mailath, G., and Rob, R. (1993). "Learning, Mutation, and Long Run Equilibria in Games," *Econometrica* 61, 29-56.

[29] Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). "Optimization by Simulated Annealing," *Science* 220, 671680.

[30] Ledolter, J. (2008). "Smoothing Times Series with Local Polynomial Regression on Time," Communications in Statistics - Theory and Methods, Vol. 37, 959-971.

[31] Miller, J. (1996). "The Coevolution of Automata in the Repeated Prisoner's Dilemma," *Journal Of Economic Behavior & Organization* 29, 87-112.

[32] Moore, E. (1956). "Gedanken Experiments on Sequential Machines," *Annals of Mathematical Studies* 34, 129-53.

[33] Neyman, A. (1985). "Bounded Complexity Justifies Cooperation in the Finitely Repeated Prisoners Dilemma," *Economic Letters* 19, 227-229.

[34] Rubinstein, A. (1986). "Finite Automata Play the Repeated Prisoner's Dilemma," *Journal Of Economic Theory* 39, 83-96.

[35] Rust, J., Miller, J. H., and Palmer, R. (1994). "Characterizing Effective Trading Strategies," *Journal of Economic Dynamics and Control* 18, 61-96.

[36] Selten, R. (1975). "A Re-examination of the Perfectness Concept for Equilibrium Points in Extensive Games," *International Journal of Game Theory* 4, 25-55.

[37] Selten, R., Mitzkewitz, G., and Uhlich, R. (1997). "Duopoly Strategies Programmed By Experienced Traders," *Econometrica* 65, 517-555.

[38] Thorp, E. O. (1962). Beat the Dealer: A Winning Strategy for the Game of Twenty-One, Vintage Books: New York.

[39] Volij, O. (2002). "In Defense of Defect," *Games and Economic Behavior* 39, 309-21.

[40] Vega-Redondo, F. (1997). "The Evolution of Walrasian Behaviour," *Econometrica* 65, 375-84.

[41] Wu, J., and Axelrod, A. (1995). "How to Cope with Noise in the Iterated Prisoner's Dilemma," *Journal of Conflict Resolution* 39, 183-89.