Learning by Doing vs. Learning from Others in a Principal-Agent Model*

Jasmina Arifovic† Alexander Karaivanov‡

October, 2009

Abstract

We introduce learning in a principal-agent model of stochastic output sharing under moral hazard. Without knowing the agents' preferences and the production technology, the principal tries to learn the optimal agency contract. We implement two learning paradigms – *social* (learning from others) and *individual* (learning by doing). We use a social evolutionary learning algorithm (SEL) to represent social learning. Within the individual learning paradigm, we investigate the performance of reinforcement learning (RL), experience-weighted attraction learning (EWA), and individual evolutionary learning (IEL). Our results show that learning in the principal-agent model is very difficult due to three main reasons: (1) the stochastic environment, (2) the discontinuity in the payoff space at the optimal contract caused by the binding participation constraint and (3) the incorrect evaluation of foregone payoffs in our sequential game principal-agent setting. The first two factors apply to all learning algorithms we study while the third is the main reason for EWA's and IEL's failures to adapt. We find that social learning, especially with a selective replication operator, is much more successful in adapting to the optimal contract than the canonical versions of individual learning from the literature. A modified version of the IEL algorithm using realized payoffs evaluation performs better than the other individual learning models; however, it still falls short of the social learning's performance.

Keywords: learning, principal-agent model, moral hazard

JEL Classifications: D83, C63, D86

*We thank Nick Kasimatis, Sultan Orazbayev and Sophie Wang for excellent research assistance. We also thank Geoff Dunbar, Ken Kasa and participants at the Society of Economic Dynamics and the Computing in Economics and Finance conferences for their useful comments and suggestions. Both authors acknowledge the support of the Social Sciences and Humanities Research Council of Canada.

†Department of Economics, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A 1S6, Canada; email: arifovic@sfu.ca

‡Department of Economics, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A 1S6, Canada; email: akaraiva@sfu.ca

1 Introduction

The optimal contracts in principal-agent models often take complicated forms, for example, due to the intricate trade-off between provision of insurance and incentives. Depending on the exact setting, the optimal contract depends crucially on both the principal's and agent's preferences, the properties of the production technology, and the stochastic properties of the income process. As an example, take a standard problem of optimal contracting under moral hazard (e.g. Hart and Holmstrom, 1987)[1]. The existing literature typically assumes that actions undertaken by the agent are unobservable or non-verifiable by the principal. However, at the same time, the principal is assumed to have perfect knowledge of objects that seem much harder or at least as hard to know or observe such as the agent's preferences, the agent's decision making process, or the functional form of the output technology.

In this paper, we address these issues by explicitly modeling the principal's *learning* process based only on observables such as output realizations. Our primary objective is to investigate whether this learning process leads to knowledge acquisition sufficient for convergence to the theoretically optimal principal-agent contract. To this end we analyze two alternative paradigms, *social* and *individual* learning, to describe the principal's learning process. The social learning paradigm represents a way of explicit, micro-level modeling of what is referred to in the literature as "learning spillovers", or "learning from others". At the same time, the individual learning paradigm can be viewed as an explicit, micro-level modeling of "learning by doing" (e.g. Arrow, 1962; Stokey, 1988).

Numerous empirical studies in widely diverse research areas suggest that individuals and firms utilize in practice social and individual learning methods resembling those we analyze. For example, in industrial organization, Thornton and Thompson (2001) use a dataset on shipbuilding during WWII to analyze learning across and within shipyards. They find that learning spillovers are significant and may have contributed more to increases in productivity than conventional learning by doing effects. Cunningham (2004) uses data from semiconductor plants and finds that firms which are installing significantly new technologies appear to be influenced by social learning. Singh, Youn and Tan (2006) find similar effects in the open source software industry. In the development literature, Foster and Rosenzweig (1996) use household panel data from India on the adoption and profitability of high-yield crop varieties to test the implications of learning by doing and learning from others. They find evidence that both households' own and their neighbors' experience increase profitability. Conley and Udry (2005) investigate the role of social learning in the diffusion of a new agricultural technology in Ghana[2]. They test whether farmers adjust their inputs to align with those of their neighbors who were successful in previous periods and present evidence that farmers do tend to adopt such successful practices. However, when they apply the same model to a crop with a known technology they find no such effect[3]. Last but not least, at the macro level, the seminal works of Romer (1986) and Lucas (1988) have emphasized the role of learning spillovers as an engine of economic growth.

Specifically, we adopt a repeated one-period contracting framework in an output-sharing model which can be thought of as optimal wage, sharecropping, or equity financing arrangement. An asset owner (the principal) contracts with an agent to produce jointly. The principal supplies the asset (e.g. a machine, land, know-how, etc.) while the agent supplies unobservable labor effort. Output is

[1] Applications that fit under this heading abound in the finance literature (credit under moral hazard), public finance (optimal taxation with hidden labor effort), development (sharecropping), macroeconomics (optimal social insurance), labor (optimal wage schedules), etc.

[2] See also Zhang et al. (2002) who present evidence for learning from others in technology adoption in India.

[3] Further evidence exists in the business / management literature. For instance, Boyd and Bresser (2004) study the occurrence and performance impact of different models of organizational learning in the U.S. retail industry and point out the importance of inter-organizational learning, while Ryu, Rao, Kim and Chaudhury (2005) document learning by doing and learning from others in the Internet data management sector.

stochastic and the probability of a given output realization depends on the agent's effort. The principal wants to design and implement an optimal compensation scheme for the agent which maximizes the principal's profits and satisfies participation and incentive compatibility constraints.

We first describe the optimal contract that arises if both contracting parties are fully rational and know all the ingredients of the contracting problem and environment. Then, we model and analyze the situation in which a principal (or, principals) with no prior knowledge of the environment has to *learn* what the optimal contract is.

We implement the social learning paradigm (learning from others) via a model of *social evolutionary learning*, SEL, in which players update their strategies based on imitating strategies of those players who have performed better in the past, and, occasionally, experimenting with new strategies. The population of players thus learns jointly through their experience that they share over time[4]. To implement the individual learning paradigm (learning by doing), we evaluate three algorithms that have been widely used in various game theoretic and applied settings: *reinforcement learning*, RL (Roth and Erev, 1995, 1998), *experience-weighted attraction learning*, EWA (Camerer and Ho, 1999)[5], and *individual evolutionary learning*, IEL (Arifovic and Ledyard, 2004, 2007). In contrast to social learning, individual learning is based on updating the entire collection of strategies that belong to an individual player, based on her *own* experience only.

Two features shared by all the algorithms are the increase in frequency of representation of well-performing strategies over time and the probabilistic choice of a particular strategy to be used in a given period. In terms of payoff evaluation, the difference between RL on one hand, and IEL and EWA, on the other, is that RL updates the payoff only of the strategy that was played in a given time period, and leaves the payoffs of the rest of the strategies unchanged. In contrast, EWA and IEL constantly update the payoffs of all strategies based on their 'foregone' payoffs. In terms of strategy representation, what distinguishes RL and EWA from IEL is that the implementation of RL and EWA requires representation of all possible strategies set in the algorithms' strategy collections, while IEL starts out with a collection (i.e. a subset) of strategies that are randomly drawn from the full set. Finally, in terms of payoff updating, RL and EWA use a procedure that is standard for a number of individual learning algorithms, i.e. the probabilities that strategies will be selected are updated based on their accumulated payoffs while IEL's updating is instead based on the evolutionary paradigm, i.e., consists of increases in frequency of the strategies that performed well in the past (based on evaluation of their foregone payoffs) and occasional experimentation with new strategies[6]

Our results show that social evolutionary learning almost always converges to the theoretically optimal principal-agent contract. In contrast, the individual learning algorithms based on evaluation of foregone payoffs (IEL and EWA) that have proven very successful in a variety of Nash environments completely fail to adapt in our setting. Reinforcement learning (RL) performs somewhat better than

[4]Our implementation of the evolutionary paradigm is based on genetic algorithms which have had numerous applications in economics – for example, Arifovic (1996), Arifovic (2000), Marks (1998), Dawid (1999), Lux and Schorstein (2005). Also, in organization theory, Rose and Willemain (1996) implement a genetic algorithm in a principal-agent environment where the principal's and agent's strategies are represented by finite automata. They find that the variance of output and agent's risk aversion affect adaptation. However, they do not analyze the relative performance of different learning algorithms nor study the reasons for non-convergence.

[5]For an excellent overview of applications of RL, EWA and other individual learning models see Camerer (2003) and Erev and Haruvy (2008).

[6]Arifovic (1994) implemented IEL in the context of the 'cobweb' model and showed it captured well price behavior in experiments with human subjects. More recently, Hommes and Lux (2008) use an individual genetic algorithm to match experimental data about expectations in cobweb settings. Our IEL implementation follows Arifovic and Ledyard (2004) who apply it to public good provision mechanisms and call market mechanisms to capture, in real time, behavior observed in experiments with human subjects. Overall, IEL has proven to be especially successful in adapting to environments with large strategy spaces (e.g., see Arifovic and Ledyard, 2004 for comparison with RL and EWA).

IEL and EWA since it only updates the payoffs of those strategies that were actually used. However, RL's overall learning performance is also unsatisfactory relative to SEL's due to RL's disadvantage in handling the large strategy space.

The intuition for the failure of EWA and IEL is that, when evaluating foregone payoffs of potential strategies that have not been tried out, the principal assumes that agent's action will remain constant (as if playing Nash), while in fact the optimal contract involves a reaction to the agent's best response function as in a Stackelberg game. The inability of these canonical individual learning models to produce correct foregone payoffs for the principal's strategies precludes their convergence to the theoretically optimal contract[7]. In contrast, social learning involves evaluation of payoffs of only those strategies that are actually played, thus avoiding this problem. As a result, SEL exhibits high rates of convergence to the optimal contract.

Two additional reasons, specific to the principal-agent setting, cause further difficulties with adapting to the optimal contract, independently of the learning algorithm used. First, as usual, the presence of stochastic shocks makes learning difficult in any type of environment. Second, in ours and any similar mechanism design model, the principal's payoffs are a discontinuous function of the strategy space at the agent's participation constraint. That is, profits are maximized at some positive value at a point on the constraint but any nearby contract which violates it yields zero payoff that in turn results in a large flat area in the payoff function. This creates problems for the successful adaptation of all learning algorithms since their performance is driven by the differences in payoffs that strategies receive over time.

The failure of individual learning where foregone payoffs are taken into account stands in stark contrast to the findings reported in the existing literature. However, our principal-agent environment is different from the environments studied so far, most importantly in its sequential as opposed to simultaneous game nature. To address this problem, we evaluate a modified version of the IEL algorithm where only payoffs of strategies that are actually tried out are updated while, at the same time, we keep the evolutionary updating mechanism that allows IEL to adapt well in environments with large strategy space as reported in the literature. The resulting 'IEL with realized payoffs' (IELR) algorithm does much better in adapting to the optimal contract than its canonical 'foregone payoffs' counterpart. Nevertheless, the IELR's convergence rates still fall short of those achieved with social learning, including when controlling for the total number of strategies being evaluated over a simulation run. [8]

2 Contracting with Full Rationality

Consider a standard moral hazard model of output/equity sharing, for example, Stiglitz (1974) on sharecropping in agriculture. Other applications include profit sharing under franchising, licensing, or author-publishing contracts. To fix ideas, interpret the principal as the owner of a productive asset

[7]Note that another commonly studied learning algorithm, *fictitious play* (Fudenberg and Levine, 1998) would suffer from the same problem in our setting, as it also uses foregone payoffs.

[8]Comparative studies of social and individual evolutionary learning have been done before in the context of the 'cobweb' model, e.g. Arifovic (1994), Vriend (2000), and Arifovic and Maschek (2006). Arifovic (1994) shows that both social and individual evolutionary learning converge to the Walrasian equilibrium. Her individual learning algorithm uses evaluation of hypothetical payoffs. On the other hand, Vriend (2000) demonstrates that while social learning converges to the Walrasian equilibrium, individual learning reaches the Cournot-Nash outcome. His individual learning algorithm uses evaluation of realized payoffs only. Arifovic and Maschek (2006) perform various robustness checks and identify the differences in the individual learning algorithm and the cobweb model parameter values that result in those different outcomes. A key difference between these studies and our paper is that our environment is not characterized by a Nash equilibrium solution.

(e.g. land) and the agent as a worker (e.g. a tenant working on the land). Output is $y(z) = z+\varepsilon$, where z is the effort employed by the agent and ε is a normally distributed random variable with mean 0 and variance σ^2. The agent's effort, z is unobservable/ non-contractible. Output, y is publicly observable. Assume that the technology is such that the principal cannot infer from the output realization what effort level was employed by the agent. The principal is risk neutral while the agent is risk averse with preferences $u(c, z)$ – increasing in consumption, decreasing in effort and strictly concave. The agent's outside option of not signing a contract with the principal is $\bar{u}$.

As in Stiglitz (1974) or Holmstrom and Milgrom (1991), restrict attention to linear compensation contracts, i.e., the principal receives $\pi(y) \equiv (1-s)y + f$ and the agent receives $c(y) \equiv sy - f$ where $s \in [0, 1]$ is the agent's output share and f is a fixed fee. We are aware that, in general (e.g. Holmstrom, 1979), the theoretically optimal compensation contract may be nonlinear (see also the discussion in Bolton and Dewatripont, 2005, section 4.3). There are several reasons for restricting our analysis to linear compensation schemes. The first reason is the observation that sharing or compensation principal-agent agreements in reality regularly take a linear form, e.g. see Chao (1983) on sharecropping; Lafontaine (1992) or Sen (1993) on franchising; Caves et al. (1983) on licensing; Masten and Snyder (1993) on equipment leasing, among others[9].

The second reason concerns complexity – given that our paper is about modeling learning about the best compensation contract, the linearity restriction simplifies the principal's problem to learning about two numbers, s and f, as opposed to learning about a general nonlinear function $c(y)$, possibly without an analytic closed form, which could make the setting much harder to adapt in[10]. Indeed, even with the restriction to linear contracts, we argued in the introduction that our environment is already quite hard to learn using the standard algorithms from the literature. Despite its relative simplicity, the linear contract is sufficiently flexible – its exact form depends on both the preferences and technology and it nests the (one-parameter) fixed wage, fixed rent, and fixed sharing rule contracts.

Because of all above reasons we assume that our learning principals offer linear contracts. Our methods can be extended to more complicated contracts to the extent that the strategy space remains tractable (e.g. if the non-linear scheme can be characterized with a small number of parameters).

2.1 The Optimal Contract

The optimal contract in the above setting can be found as the solution of a standard principal-agent mechanism design problem. The principal's objective is to maximize his expected profits subject to participation and incentive compatibility constraints for the agent:

$$\max_{s,f} \ (1-s)z + f$$

subject to:

$$z = \arg\max_{\hat{z}} \ Eu(sy(\hat{z}) - f, \hat{z}) \tag{1}$$

$$Eu(sy(z) - f, z) \geq \bar{u} \tag{2}$$

The first constraint is the incentive compatibility constraint (ICC) stating that the chosen effort must be optimal for the agent given the proposed compensation scheme (s, f). The second constraint is the participation constraint (PC) stating that the agent must obtain expected utility higher or equal to his outside option $\bar{u}$ in order to accept the contract. We assume that $\bar{u}$ is large enough so that the participation constraint is binding at the optimal contract (s^*, f^*).

[9] Linear contracts could be also required to prevent re-sale among agents.

[10] See Holmstrom (1979) or Bolton and Dewatripont (2005, ch. 4) for examples of nonlinear and even non-monotonic optimal contracts in the moral hazard problem and discussion on their general analytical non-tractability.

2.2 A Computable Example

We use the following easy-to-compute example in our numerical analysis of learning in the principal-agent model. Assume a mean-variance expected utility of consumption for the tenant, and quadratic cost of effort, $Eu(c,z) \equiv E(c) - \frac{\gamma}{2}Var(c) - \frac{1}{2}z^2$. With our production function specification mean-variance expected utility is equivalent to assuming exponential Bernouli utility of the form $u(c,z) = -e^{-\gamma(c-\frac{1}{2}z^2)}$ (e.g., see Bolton and Dewatripont 2005, pp. 137-9 for a derivation). Thus, our computable example is the so-called 'LEN' model (Linear contract, Exponential/CARA utility, and Normally distributed performance) widely used in the applied contract theory literature (e.g., see Holmstrom and Milgrom, 1991 on multi-tasking in firms; Dutta and Zhang, 2002 in accounting, etc.) [11]

The tenant's expected utility is:

$$U^T(z) = sz - f - \frac{\gamma}{2}\sigma^2 s^2 - \frac{1}{2}z^2.$$

For our preferences and production function, it is easy to verify that the 'first order approach' (Rogerson, 1985) is valid. Hence, we replace the incentive compatibility constraint (1) with its first order condition:

$$z^* = s$$

The principal's problem then becomes (substituting from the ICC and the participation constraint),

$$\max_s \ (1-s)s + \frac{1}{2}s^2(1-\gamma\sigma^2) - \bar{u},$$

with a first-order condition,

$$1 - 2s + s(1-\gamma\sigma^2) = 0,$$

which implies,

$$s^* = \frac{1}{1+\gamma\sigma^2}.$$

The optimal fixed fee, f^* is then found using the participation constraint:

$$f^* = \frac{1}{2}\frac{1-\gamma\sigma^2}{(1+\gamma\sigma^2)^2} - \bar{u}.$$

3 Learning about the Optimal Contract

Our main objective is to examine the behavior of our principal-agent model under learning. We assume that the stage game from section 2 is repeated over time. The main reason for this is computational complexity. Contract theory suggests (e.g. Townsend, 1982) that in a dynamic (as opposed to repeated) setting, intertemporal tie-ins and history dependence typically exist in the optimal contract. If standard learning algorithms cannot converge to the optimal static contract, then we expect them to be even less successful in adapting to the optimal dynamic contract[12].

[11] Holmstrom and Milgrom (1987) show further that the optimal incentive scheme in the LEN model is linear when y is interpreted as an aggregate performance measure resulting from the agent supplying effort in continuous time to control the drift of output.

[12] The optimal dynamic moral hazard contract requires that the principal keep track of the full history of output realizations and use it to determine the current transfers. This causes an exponential expansion of the dimensionality of the strategy space. While this could be resolved by introducing an extra state variable, 'promised utility' (e.g. Phelan and Townsend, 1991), the optimal contract must specify both (contingent) current consumption and promised future utility which significantly increases its complexity relative to the static case. We plan to investigate learning in such dynamic settings in future work.

We study how hard (or easy) it is for boundedly rational principals to learn what the optimal sharing contract looks like. Specifically, as a first pass, we assume that the principal is not endowed either with the ability to optimize or with knowledge of the physical environment, i.e., she does not know what the agent's preferences and the exogenous stochastic shock's properties are. In contrast, agents are assumed to be able to optimize[13] their effort choice, z^* given the share, s, and the fixed fee, f, that they are offered by the principals.

The learning proceeds as follows. Each period, (1) the principal offers a contract (s_t, f_t) belonging to some known set of feasible contracts[14], (2) the agent chooses effort, (3) output is realized, and (4) the principal's profit is computed. If the offered contract does not satisfy the agent's participation constraint, we assume that the principal gets a payoff of $\underline{\pi}$ for the current period (set to zero in the simulations). After profits are realized the principal updates her strategy set and chooses a new contract (s_{t+1}, f_{t+1}), etc.

We investigate two learning paradigms. The first is *individual learning* in which a player learns only from her own experience. Specifically, in our model each principal is endowed with a collection of different strategies that she updates over time based on her experience. We examine the behavior of three popular models of individual learning that share some common features but also differ in several important dimensions: 'Reinforcement Leaning' (RL) – Roth and Erev (1995); 'Experience-Weighted Attraction Learning', (EWA) – Camerer and Ho (1999); and 'Individual Evolutionary Learning' (IEL) – Arifovic and Ledyard (2004, 2007).

The second learning paradigm we study is *social learning* in which players can learn from each others' experience. In our setting, this translates into a learning model where principals are given an opportunity to observe the behavior of some of the other principals and update their strategies (the contracts offered) accordingly. Our preferred model of social learning is based on the evolutionary paradigm in which the principals' performance and survival are based on how successful their strategies, (s, f) are and on occasional experimentation with new strategies.

3.1 Common Structure of the Learning Algorithms

In each of the learning models we consider all agents are identical and optimize each time period given the contract proposed by the principal. A *strategy*[15], m_t, belonging to the *strategy set,* M_t at time $t \in \{1, T_0\}$ consists of a share/fee pair, i.e., $m_t = \{s_t, f_t\}$. The strategy set M_t of fixed size J has elements m_t^j, $j = 1, ..., J$ each of which belongs to the *strategy space*, G of all possible contracts that can be offered. We assume G is a two-dimensional grid of size $\#S \times \#F$ where S and F are linearly-spaced grids[16] for the share, s, and the fee, f. The coarseness of the S and F grids is governed by the parameter d, which determines the number of feasible contracts in G.

In the case of individual learning, the strategy set M_t is a collection of $J \geq 2$ strategies that belong to a single principal. At each t, the principal chooses one of these J strategies as the actual contract offered to the agent. In contrast, in the case of social learning, there is a population of $N \geq 2$ principals and each principal, $i \in \{1, ..., N\}$ has a single strategy m_t^i that she uses at time t. That is, in social learning an individual principal's strategy set, M_t^i is a singleton, $M_t^i \equiv m_t^i \in G$ and

[13] We provide a brief discussion on double-sided learning in the conclusions.

[14] The constraints on this contract feasibility set can be natural, e.g., the share s by definition must belong to the interval $[0, 1]$ or determined by the contractual environment, e.g., the bounds on f can come from limited liability or similar constraints. We assume these constraints are known to the principal.

[15] Hereafter, we use the terms *strategy* and *contract* interchangeably.

[16] In principle, we could use continuous sets for s and f in the implementation of the IEL and SEL algorithms. However, since we also evaluate the performance of the RL and EWA algorithms which can be implemented only using discretized strategy space, we chose a discrete G for consistency. We perform robustness checks with respect to the grid density.

the overall (population) strategy set, M_t is the collection of all N individual time-t strategies, i.e., $M_t = \cup_{i=1}^{n} M_t^i = \{m_t^1, ..., m_t^N\}$.

Each period consists of $T_1 \geq 1$ 'interactions' between a fixed principal-agent pair (one pair in case of individual learning, and N pairs in case of social learning) where each interaction[17] corresponds to a separate output shock (ε draw) but the principal uses the same strategy (s_t, f_t) over the T_1 interactions. Specifically, at any t, the principal announces a single contract, a share-fee pair, $(s_t, f_t) \in M_t$. Given the contract, the agent provides his optimal level of effort, $z_t^* = s_t$ while output and profits vary depending on the realization of the shock, ε_s. Formally, output for each within-period interaction, y_s, $s = 1, ..., T_1$ is:

$$y_{s,t} = s_t + \epsilon_{s,t}$$

where $\epsilon_{s,t}$ are i.i.d. normally distributed with mean zero and variance σ^2 while the principal's profit is:

$$\pi_{s,t} = y_{s,t}(1 - s_t) + f_t$$

At the end of the T_1 interactions, the value of the average output produced during time period t is

$$\bar{y}_t = s_t + \frac{1}{T_1} \sum_{s=1}^{T_1} \varepsilon_{s,t},$$

and the average payoff of the principal for period t is

$$\bar{\pi}_t = (1 - s_t)\bar{y}_t + f_t \tag{3}$$

The average payoff, $\bar{\pi}_t$ represents the measure of performance (fitness) of a particular strategy (s_t, f_t) that the principal uses at time period t. If the proposed strategy does not satisfy the agent's participation constraint, its average payoff, $\bar{\pi}_t$ is set to $\underline{\pi} = 0$.

3.2 Individual Learning

The individual learning paradigm is based on an individual's learning and updating of strategies based only on her *own* experience. In our setting this implies that each period the (single) principal has a collection of strategies that is used for her decision making process. Over time, as a result of accumulated information about the performance of individual strategies, the updating process results in an increase in the frequency of well-performing strategies in the principal's strategy set. The choice of a particular strategy as the actual strategy that the principal uses in a given period is probabilistic, and the selection probabilities depend positively on the strategies' past performance.

The three individual learning models that we study, RL, EWA, and IEL have this common feature but they also differ in important ways. The main differences among them are in how the strategy set, M_t is determined and updated over time. These differences play an important role for the results we obtain.

3.2.1 Reinforcement Learning

To model *Reinforcement Learning* (RL) we follow the implementation of Roth and Erev's (1995). The strategy set is the whole strategy space, $G = S \times R$, i.e., all possible combinations of s and f, and

[17] The function of these within-period interactions between a principal-agent pair is to give the principal some time to learn about the expected profits that can be generated from a given offered contract. We provide comparative statics with respect to T_1 in the robustness section 5.3.

is the same in all periods, i.e., $M_t = M$ for all t. The number of strategies, J in M for each t hence equals the total number of elements of G, i.e., in the RL model, $J = \#S \times \#R$. A single principal chooses one of these strategies to play each period. Each strategy in M is assigned a *propensity of choice* which is updated at time t based on the payoff this strategy earned if it was used at t and is otherwise left at its previous level. In our implementation, the propensities of choice are given by the strategies' discounted payoffs.

Specifically, for each strategy m_t^j in M, $j \in \{1, ..., J\}$, let I_t^j denote an indicator value for the principal's strategy in period t, where $I_t^j = 1$ if m_t^j is chosen/played in period t and $I_t^j = 0$ otherwise. Then, the *discounted payoff* of strategy j, at time t, R_t^j is defined as:

$$R_t^j = qR_{t-1}^j + I_t^j \bar{\pi}_t^j \tag{4}$$

where $q \in [0, 1]$ is a time/memory discount parameter and $\bar{\pi}_t^j$ is the average payoff of strategy j computed over T_1 interactions. The initial payoff, R_1^j of each strategy in the strategy space G is set to 0.

Strategies are selected to be played based on their propensities. Those with higher propensities have higher probability of being selected. Namely, at the end of each period t, the principal selects strategy $m^j \in M$, $j \in \{1, ..., J\}$, to be played at $t + 1$ with probability:

$$Prob_{t+1}^j = \frac{e^{\lambda R_t^j}}{\sum_{k=1}^J e^{\lambda R_t^k}}. \tag{5}$$

Once a strategy is selected, it undergoes experimentation with probability μ. In case that experimentation takes place, instead of the initially selected strategy, e.g. $\tilde{m}$, the principal uses a randomly drawn strategy from the square centered on $\tilde{m}$ with sides of length $2r_m$. The final chosen strategy is then implemented for T_1 interactions as explained above.

3.2.2 Experience-Weighted Attraction Learning

The second individual learning algorithm we evaluate, *Experience-Weighted Attraction Learning* (EWA), is a generalization of the RL algorithm. Specifically, we follow Camerer and Ho (1999) to model EWA learning. The (fixed) strategy set, M of size $J = \#S \times \#R$ is the same as that under RL, namely the complete strategy space G. That is, once again $M_t = M = G$ for all t – the principal's strategy set does not change over time.

In EWA, a strategy that was actually used, denoted by $m_t^a \equiv (s_t^a, f_t^a)$, receives an evaluation based on its actual performance from (3), while all other strategies in M receive evaluation based on their *foregone* (hypothetical) performance. The period-t foregone payoff, $\bar{\pi}_t^j$ for any strategy $m^j \in M$, $m^j \neq m_t^a$ is computed *taking as given* the optimal agent's effort response to m_t^a, the strategy actually used at t:

$$\bar{\pi}_t^j = (1 - s_t^j)\bar{y}_t(s_t^a) + f_t^j$$

where $\bar{y}_t(s_t^a)$ is the average output generated under the actually played strategy. In the performance evaluation process (see below) the foregone payoff is weighted by the discount parameter $\delta \in (0, 1)$ reflecting the fact that these strategies were not actually used.

At the end of each period, the so-called 'attractions' (corresponding to the propensities of choice in the RL model) of all strategies are updated. Specifically, in EWA there are two variables that are updated after each round of experience: W_t, the number of 'observation-equivalents' of past experience (called the *experience weight*); and A_t^j, the *attraction* of strategy $m^j \in M$ (whether played or not) at

the end of period t. Their initial values W_0 and A_0^j can be interpreted as prior game experience and/or principal's predictions.

The experience weight, W_t, is updated according to

$$W_t = \rho W_{t-1} + 1 \tag{6}$$

for any $t \geq 1$ and where ρ is a retrospective discount factor. The updated attraction of strategy m^j at time t is given by:

$$A_t^j = \frac{\phi W_{t-1} A_{t-1}^j + [\delta + (1-\delta) I_t^j] \bar{\pi}_t^j}{W_t} \text{ for all } j = 1, .., J$$

The parameter δ determines the extent to which hypothetical evaluations are used in computing the attractions, A_t^j. If $\delta = 0$, then no hypotheticals are used, just as in the RL model while if $\delta = 1$ hypothetical evaluations are weighted as much as actual payoffs. The parameter ϕ is another discount rate, which depreciates the previous attraction level similarly to the parameter q in RL. If $\phi = q$, $\delta = 0$, $\rho = 0$, and $W_0 = 0$ then the EWA model is exactly equivalent to the RL model. Finally, the principal selects strategy $m^j \in M$ to play at $t+1$ with probability:

$$Prob_{t+1}^j = \frac{e^{\lambda A_t^j}}{\sum_{k=1}^{N} e^{\lambda A_t^k}} \tag{7}$$

which is followed by experimentation in the same way as in the RL model.

3.2.3 Individual Evolutionary Learning

Our third individual learning algorithm, *Individual Evolutionary Learning* (IEL), shares some common features with both RL and EWA. First, like in RL and EWA, the choice of the principal's strategy is probabilistic. Second, the selection probabilities are based on the strategies' hypothetical (foregone) payoffs like in EWA learning. However, there is an important difference. In the IEL model, the set of active strategies is not the complete space G but instead is of smaller dimension and endogenously changes over time in response to experience and, occasionally, to pure random events (experimentation).

Specifically, at time $t = 1$, a set of $J \geq 2$ strategies[18], M_1 is randomly drawn from G. Over time, the principal always keeps J active strategies. Suppose that at the beginning of period t, the principal's collection of active strategies is $M_t \subset G$. One of these strategies, $m_t^a \in M_t$ is selected as the actual strategy to be played during t, i.e., it is implemented over T_1 interactions with the agent.

Similar to EWA learning, the payoffs of all other, inactive, strategies in the set M_t (but not those in the rest of the strategy space, $G \backslash M_t$) are updated as well. Their payoffs, averaged over the T_1 interactions, are computed by taking as given the optimal agent's effort response to m_t^a, strategy that was actually used at t. Denote this effort response by $z^*(m_t^a) = s_t^a$. Then, the *hypothetical payoff* for any strategy $m_t^j \in M_t$, $m_t^j \neq m_t^a$ in period t is:

$$\bar{\pi}_t^j = (1 - s_t^j) \bar{y}_t(s_t^a) + f_t^j \tag{8}$$

where $\bar{y}_t(s_t^a)$ is the average output generated under strategy m_t^a.

Once the hypothetical payoffs are computed, the updating of the principal's collection of strategies takes place applying *replication* and *experimentation*. The replication operator allows for potentially better paying alternatives to replace worse ones. It is used to generate a collection of J replicates

[18] Unlike in RL and EWA, J is typically chosen to be smaller than the number of strategies in G. IEL also works with a continuous strategy space.

of the strategies in the strategy set M_t. As our baseline operator, we use proportionate ('roulette wheel') replication. Specifically, each strategy m_t^j , $i \in \{1, ..., J\}$ in M_t has the following probability of obtaining a replicate to appear in the next period's strategy set:

$$Prob_t^j = \frac{e^{\lambda \ bar\pi_t^j}}{\sum_{j=1}^{J} e^{\lambda \bar{\pi}_t^j}} \tag{9}$$

where λ is a parameter governing the relative fitness weights.

We also consider *selective* proportionate replication. Under selective replication, the replicate strategy replaces the former strategy at location j in M_t only if the replicate yields a higher average hypothetical payoff. Formally, the payoff of each replicate strategy m_{t+1}^j, $j = \{1, \ldots, N\}$ is compared to the payoff of strategy, m_t^j, $j = \{1, \ldots, N\}$ – the j^{th} member of the strategy collection at t. The strategy that has a higher payoff between the two, becomes the member of M_{t+1} at $t + 1$.

As a robustness check, we also implement another commonly used replication operator, *tournament selection*, in the following way. For $j = 1, \ldots, J$, the location-j strategy in M_{t+1}, m_{t+1}^j is chosen by drawing (with replacement) two members of M_t randomly with equal probability. Formally, if the two drawn strategies be m_t^k and m_t^l we have,

$$m_{t+1}^j = \left\{ \begin{array}{c} m_t^k \\ m_t^l \end{array} \right\} \text{ if } \left\{ \begin{array}{l} \bar{\pi}(m_t^k) \geq \bar{\pi}(m_t^l) \\ \bar{\pi}(m_t^k) < \bar{\pi}(m_t^l) \end{array} \right\}. \tag{10}$$

After replication, *experimentation* takes place. That is, each strategy m_{t+1}^j is subjected to 'mutation' with probability μ. If experimentation takes place, the existing strategy, m_{t+1}^j is replaced by a new strategy from G drawn from a square centered on m_{t+1}^j with sides of length $2r_m$. Note that the IEL experimentation is different from the experimentation in RL. In RL, only the strategy actually selected for implementation could be experimented with. On the other hand, in IEL, *each* strategy in M_t can be changed by experimentation with probability μ.

3.2.4 Individual Evolutionary Learning with Realized Payoffs

We also propose a modified model of individual evolutionary learning that we decided to study in light of the unsatisfactory convergence performance (see section 5 for details) of the canonical IEL algorithm with foregone payoffs described above. This modified model differs from the standard IEL in that only the payoffs of strategies that were actually played are updated. In this respect, the modified algorithm is similar to RL. We call this algorithm *IEL with realized payoffs (IELR).* Apart from the elimination of hypothetical evaluations we keep all other features of the standard IEL model – an endogenous, time varying strategy set M_t; using replication to change the frequency with which different strategies are represented in it; and IEL experimentation. Overall, the proposed IELR model is a hybrid between RL and the standard IEL model.

3.3 Social Learning

The second major learning paradigm we study is social learning, or learning from others. In the social learning model, learning operates on the level of *population.* Unlike with individual learning, there is a 'large' number of principals, N who are given an opportunity to learn from each other over time. At

each time t, each principal i, $i \in \{1, ..., N\}$, has only a *single* strategy, i.e., $J = 1$ and $M_t^i = m_t^i \in G$. The *population strategy set* at time t, M_t consists of the N individual strategies. [19]

There are N principal-agent pairs. For each principal, the learning process proceeds following the general form in section 3.1. Below we describe the specifics related to our implementation of *Social Evolutionary Learning* (SEL) to represent the idea of learning from others.

The first element of SEL algorithm is *replication*. The mechanics of the process are basically the same as those in IEL. An important difference, however, is that in SEL replication operates on a population of strategies that belong to different principals, as opposed to operating on the single principal's strategy set. Hence, SEL replication can be interpreted as imitation of relatively successful strategies played by others, as opposed to replicating one's own strategies that have performed well in the past. Replication is used to generate a population of N replicates of the strategies that were used in the population at period t. We use proportionate ('roulette wheel') replication as our baseline operator. Specifically, any strategy m_t^i , $i \in \{1, ..N\}$, in the current strategy set has the following probability of obtaining a replicate to appear in the next period's population strategy set:

$$Prob_t^i = \frac{e^{\lambda \bar{\pi}_t^i}}{\sum_{j=1}^{N} e^{\lambda \bar{\pi}_t^i}} \tag{11}$$

where λ is a parameter governing the relative fitness weights. As in IEL, we also consider selective proportionate replication and tournament selection[20].

After the replication, *experimentation* takes place. Each replicate strategy m_{t+1}^i is subjected to 'mutation' with a probability μ. If mutation takes place, the existing strategy, m_{t+1}^i is replaced by a new strategy from G drawn randomly from a square centered on m_{t+1}^i with sides of length $2r_m$.

SEL models the interaction of a population of principals who learn collectively through gathering information about the behavior of others and through imitation of previously successful strategies. Strategies that yield above-average payoffs tend to be used by more principals in the following period. The experimentation stage incorporates innovations by principals, done either on purpose or by chance.

The SEL model shares a common feature with IELR in that only actually played strategies are used in the updating process. However, in IELR the single principal learning on her own, by definition can only evaluate one strategy per period, after which updating (replication and experimentation) takes place. In contrast, in SEL N strategies are simultaneously played each period by the population and their performance is used in the updating process. Due to the difference in the frequency of updating and information used, SEL is not equivalent to simply repeating IELR algorithm N times or, alternatively, to an N-player version of IELR. This important distinction plays a significant role how these two models adapt towards the optimal contract. We provide a detailed discussion in section 6.

4 Computational Implementation of the Learning Algorithms

In this section we describe the computational procedures we followed to initialize and implement the learning algorithms in our principal-agent model. The next section contains simulation results obtained for a large set of parametrizations and numerous robustness checks. [21]

[19] In our simulations, to compare between individual and social learning, we keep the number of strategies in the population set N under social learning equal to the number of strategies in the single principal's set J under individual learning.

[20] The selection criterion expression is the same as in (10) with i in place of j.

[21] The MATLAB codes for all simulations reported in this paper is available from the authors upon request.

To obtain representative results we perform 7,350 different runs for each learning regime. These runs differ in the parameter values for the risk aversion, γ, the output variance, σ from the principal-agent model, and the random generator seed used to draw the initial pool of strategies. That is, each run corresponds to a unique combination of $(\gamma, \sigma, seed)$. The values for γ and σ are exhibited in Table 1 below. The agent's reservation utility is set to $\bar{u} = 0$.

The strategy space, G, from which strategies are chosen is composed of all (s_t, f_t) pairs belonging to a two-dimensional grid such that s_t belongs to a uniformly spaced grid on the interval $[s_{\min}, s_{\max}] = [0, 1]$ and f_t belongs to a uniformly spaced grid on the interval $[f_{\min}, f_{\max}] = [-0.05, 0.5]$. The strategy space bounds were chosen to ensure that the optimal contract (s^*, f^*) is always inside G for each possible γ and σ we use. The strategy space G is discretized in both dimensions with distance, d, between any neighboring points.

In the SEL and IEL models, N strategies are randomly chosen from G at $t = 1$ and assigned an initial fitness (payoff) of zero. Under RL and EWA all possible strategies in G are initially assigned zero fitness. Each run continues for $T_0 = 2,400$ periods. At period $\hat{T} = 2,000$ the experimentation rate, μ (constant until then) is let to decay exponentially. [22]

The benchmark values for all parameters used in the computations are described in the table below:

Table 1 – Benchmark Parameter Values

Parameter	Values Used
risk aversion, γ	15 linearly spaced points on [0.2, 3]
output variance, σ	7 linearly spaced points on [0, 0.6]
random seeds	70 random integers on [1, 10,000]
SEL population strategy set size, N	30
IEL(R) individual strategy set size, J	30
run length, T_0	2,400
output draws per period, T_1	10
experimentation rate, μ	0.05
experimentation decay factor, χ	0.9998
experimentation radius, r_m	0.1
weighting factor, λ	1
grid density, d	0.01
EWA parameters, δ, ρ, ϕ	$\delta = 0.2$, $\rho = 0.8$, $\phi = 0.8$
RL discount parameter[23]	1

In the next section we also report results from numerous robustness and comparative statics runs varying the baseline parameters.

5 Results

5.1 Benchmark Simulations

We begin by reporting the results from our benchmark individual and social learning runs. Specifically, we define and examine a number of measures that reflect the qualitative and quantitative aspects of the learning dynamics. These measures are:

[22] We use the following formula: $\mu_t = \mu_{t-1} 0.998^{t-\hat{T}}$ where t is the current simulation period.

[23] We also tried a discount factor $q = .9$ but this value resulted in worse performance than the baseline.

- the frequency distribution over all simulations of the differences between simulated and optimal payoffs of all strategies in the final period
- the frequency distribution over all simulations of the differences (in Euclidean distance) between simulated and optimal strategies in the final period
- the time paths of the fraction of simulated strategies or payoffs within a given distance from the optimum. Each point on these time paths equals the average fraction over all strategies and over the 7,350 runs. Two distance criteria are considered: 0 and 0.05 (0 and 5% for the payoffs).
- the strategy time paths generated by the different learning models for a sample run.

Table 2 characterizes the performance of the four baseline learning algorithms (RL, EWA, IEL, and SEL). The table shows two alternative measures of performance, averaged over all 7,350 runs: (i) the percentage of last period ($t = 2,400$) strategies in the strategy set that are within 0 (i.e., achieve the optimal contract), 0.05 or 0.1 Euclidean distance from the optimal contract and (ii) the percentage of last period ($t = 2,400$) strategy payoffs within 0%, 5% or 10% from the optimal contract payoff.

Table 2 indicates that the benchmark individual learning algorithms are unable to adapt in the principal-agent environment. The performance of RL and EWA improves negligibly when the performance criteria are relaxed, with RL doing slightly better than EWA. All three individual learning algorithms show poor performance even under the most relaxed performance criterion (0.1 or 10% from the optimum). [24]

While performing better than all individual learning models, it is evident from table 2 that our SEL algorithm with baseline replication also has a hard time learning the optimal contract. When the exact convergence to the optimum (0%) criterion is used, only 1.65% of the last period strategies in the pool across all runs coincide with the optimal contract (s^*, f^*). When we relax the performance criterion to include convergence within 0.1 Euclidean distance from the optimal contract (or, alternatively, within 10% of the optimal payoff), the benchmark SEL algorithm shows better performance with 67.8% of all strategies in the final pool over all runs ending within 10% of the optimal payoff.

Figures 1 and 2 complement table 2 by visualizing the algorithms' performance. Figure 1 which displays the histograms of the differences between simulated and optimal payoffs and strategies shows that only the SEL model can sometimes get anywhere close to the optimal contract and payoff. Figure 2 shows the time paths of the actually offered share, s, and fee, f (averaged over the N strategies in SEL) for a given sample run under each learning regime. The figures illustrate clearly that all the learning algorithms we study have serious difficulties with convergence to the optimal contract in our principal-agent setting.

Discussion

There are three main factors responsible for the poor performance of our baseline learning algorithms. First, and common to all algorithms, is the fact that in our problem (and, generally, in any similar principal-agent problem), payoffs are discontinuous at the participation constraint. Figure 3 which is drawn for a sample parameter configuration illustrates this point. All strategies above the participation constraint given by the parabola-shaped dashed line defined by $f = \frac{s^2(1-\gamma\sigma^2)}{2}$ receive zero payoff since no contract materializes between the principal and the agent. The optimal contract (s^*, f^*), denoted by a black diamond in the figure, lies *on* the participation constraint. Thus, small

[24] The standard IEL algorithm that has been shown previously to perform better than RL and EWA in other environments with large strategy spaces (e.g., Arifovic and Ledyard, 2004).

deviations away from the optimal contract that enter the zero-payoff area above the participation constraint cause a large discontinuous drop in the payoff. This discontinuity affects the performance of all learning algorithms and can slow or even prevent convergence. In addition, as evident from figure 3 where we also plot the iso-payoff lines for a typical case, the principal's payoffs decrease steeply moving away from the participation constraint while they stay quite high near the constraint even for (s, f) pairs that are far away from the optimal strategy. Replication can thus result in increasing the number of instances of strategies that have relatively high payoffs but are far away from the optimal contract. At the same time, even the smallest amount of experimentation can take a strategy "off the cliff", to the right, into an area of much lower payoffs, and to the left, into the area of zero payoffs above the participation constraint.

A second, extremely important factor is directly responsible for the poor performance of the IEL and EWA algorithms. It is these algorithms' reliance on hypothetical (foregone) payoff evaluation. Specifically, strategies in the principal's strategy set that have not yet been played receive payoff updates together with the actual contract offered to the agent. The problem with this is that the hypothetical payoffs are computed as the foregone profits that the principal would have obtained if they had played some alternative strategy (s^a, f^a) – see (8). However, the actual observed output realization, $\bar{y}$, that is used in this calculation reflects the effort that is an optimal response to the contract that was actually played, not the hypothetical contract that was not played. This is key to understanding the failure of the EWA and IEL algorithms in our setting. The reason is that, if in fact the agent were offered a different s, she will change her behavior, and expected output will not be the observed $\bar{y}$ anymore. That is, all assigned hypothetical payoffs are incorrect, except by chance. This dooms any algorithm using foregone payoffs to update the strategies' fitnesses. Note that this is a general point that would apply to any economic model in which the underlying game is sequential (e.g., Stackelberg) i.e., in which one party moves first and then the other party reacts. In contrast, in simultaneous (Nash) games, evaluating and using hypothetical payoffs as in the IEL algorithm is not subject to this problem since the equilibrium is defined by finding the best response *given* (holding fixed) the other party's choice.

The problem with using hypothetical payoffs in learning sequential game equilibria is further illustrated in the following example in the context of IEL. Suppose, for simplicity, $\sigma^2 = 0$ and that the principal offers the contract (s_t, f_t) with $s_t > 0$ and $f_t > 0$. The principal then receives the signal $y_t = s_t > 0$. Among all strategies in her current set, M_t, the one that makes principal's profits, π_t, largest while still satisfying the participation constraint is assigned the highest payoff. Let us look closer at what this strategy would look like. At $\sigma^2 = 0$ the participation constraint is simply $f_t \leq \frac{s_t^2}{2}$, so we have $\pi_t \leq y_t(1 - s_t) + \frac{s_t^2}{2}$. The IEL hypothetical payoff evaluation scheme assumes (incorrectly) that the agent's behavior (i.e., z_t and therefore y_t) stays the same. Thus, π_t is a quadratic function in s with maximum achieved at a corner solution, $s = 0$ or $s = 1$. In particular, if $y_t > 1/2$, the strategy in M_t that is "closest" to (0, 0) receives the highest payoff, while if $y_t < 1/2$, the strategy in the pool closest to (1, 0.5) achieves the highest payoff. Suppose the former situation has occurred. Then, after replication, the pool at $t + 1$ will be biased towards contracts close to the point (0, 0) in G. If $s_{t+1} < 1/2$, (and so $y_{t+1} < 1/2$) which is likely given the time-t replication, then at $t + 1$ the strategies closer to (1, 0.5) are now favored.[25] This process cycles over time and which corner strategy survives the replication and experimentation is up to chance. Clearly, convergence to the optimum under these conditions can occur only under very special circumstances.

The third major factor explaining our benchmark results and common to all learning models, is that our setting features learning in a stochastic environment. It is well-known from experiments with

[25]For simplicity, this discussion assumes that the principal is able to learn the participation constraint.

different learning models[26] that convergence to equilibria in stochastic settings is often difficult and might depend on the parameters of both the learning algorithm and the underlying economic model. The main reason is that these algorithms require that the assessment of strategy payoffs (which drives the selection and reinforcement process) be quite accurate. In our benchmark runs, the principal observes $T_1 = 10$ output draws on which a strategy's payoff is based. In the robustness section 5.3 below we show that increasing this "evaluation window" helps improve convergence to the optimum, confirming this intuition.

One last factor applies specifically to the RL and EWA models in which all the points on the strategy grid G belong to the principal's strategy set. In our benchmark simulations this number of points is quite large (over 5,000) which contributes additionally to the poor performance of these two algorithms. A decrease in the grid density improves their performance somewhat (see section 5.3).

5.2 Simulations with Modified Evolutionary Algorithms

We try to reduce the potentially negative effects of simple roulette wheel (baseline) replication on the performance of IEL and SEL by replacing it with *selective* replication (described in section 3.) Variants of this type of selection are standard in the applications of social evolutionary learning. The basic idea is that a new strategy, selected via proportionate replication, replaces the existing strategy only if it has a higher payoff.

We also modify the benchmark IEL algorithm to deal with the hypothetical payoffs evaluation problem. Specifically, within IEL we adopt a method for evaluating strategy payoffs that is similar to RL, i.e., only strategies that were actually selected for play have their payoffs evaluated[27] – the Individual Evolutionary Learning with Realized Payoffs (IELR) model from section 3. With the IELR version of the algorithm, our objective is to examine whether good features of the evolutionary updating process, which have proven useful in handling large strategy spaces, combined with realized rather than hypothetical payoffs evaluation, facilitate individual evolutionary learning in the principal-agent environment.

The performance of the modified social and individual evolutionary learning algorithms is displayed in table 3. Selective replication alone does not change the IEL's poor performance. It however, improves dramatically the performance of the SEL model – now 73.5% (versus only 1.6% in the benchmark) of all 7,350 simulations converge exactly to the optimal contract and virtually 100% of them come to within 5% of the maximum possible profit, compared to only 37% with the baseline replication operator. We further illustrate this improvement in performance in figure 4 which displays the histograms of the differences between simulated and optimal payoffs and strategies for the modified algorithms. Note the much larger fraction of differences close to zero compared with figure 1. Similar improvement are seen in the time paths of fractions of strategies equal to or within 5% of the optima shown on figure 5. Note that the percentage of strategies coinciding with the optimum in the modified SEL (the top panels) increases fast over time with about 90% of them getting within 5% of the optimal profit by as early as period 300.

Next, we study the performance of the IELR algorithm. Table 3 and figures 4 and 5 report a significant improvement in performance under both baseline and selective replication relative to the

[26]For example, Lettau (1997) shows how agents who learn via genetic algorithms, in a social learning setting, hold too much risk as compared to the optimal portfolio of rational investors. Lettau and Uhlig (1999) demonstrate a 'good state' bias in decision rules updated with an algorithm that combines elements of reinforcement learning and replicator dynamics.

[27]Note that the same modification applied to EWA would reduce it to a version of the RL algorithm whose performance has already been evaluated so we refrain from this.

baseline IEL model with hypothetical payoffs. We see large gains in performance resulting from both modifications (using realized payoffs, and using selective replication, conditional on using realized payoffs). Ultimately, however, the IELR's performance remains worse than that of the SEL algorithm with the same replication method. Specifically, IELR has 30% and 59% of its time-T pool strategies within respectively 5% and 10% of the optimal profits compared to 0% under hypothetical payoffs. These numbers rise to 81% and 92% when selective replication is also applied. These results are further illustrated in figure 6 where we display the strategy time paths for a sample run[28].

Looking at figure 5, we see a jump in convergence around period 2,000 where the experimentation rate starts decaying. This happens because any diversity in the strategy pool disappears since experimentation is no longer possible. The "jump" is much larger in IEL as new mutants there (entering with a payoff of zero) can survive quite long in the strategy set without being played and hence without being updated. Once the experimentation rate decays to zero these strategies disappear from the strategy pool and thus the fraction of strategies in the pool equal or within some distance of the optimum increases.

Overall, as in the benchmark results, we find that principals using the social learning algorithm (especially when allowing for the more sophisticated selective replication operator) are better able to learn the optimal contract in our environment than individual learners. Individual evolutionary learning with realized payoff evaluation shows promise but still performs much worse than social evolutionary learning. A candidate reason for this under-performance seems that in SEL all N strategies in the population strategy set are evaluated each period based on their actual payoffs. In contrast, by definition, the single principal in IELR updates only one strategy at a time (the actual contract offered) which seems to put the individual learning algorithm at disadvantage as less strategies are evaluated per fixed number of periods. We come back to analyze this issue more formally in the discussion section 6 showing that SEL still performs better than IELR after controlling for the total number of evaluated strategies.

5.3 Robustness

In this section we report results from numerous additional simulations which we ran to investigate the robustness of the performance of the learning algorithms to various changes in the parameters. Specifically, we study the effect of increasing the strategy pool size, N; increasing the number of evaluation runs, T_1; varying the payoff weighting parameter, λ; varying the experimentation rate, μ; varying the scope of the experimentation governed by r_m; using tournament selection in the replication process; and varying the timing of experimentation decay[29] All robustness runs were performed for the same set of 7,350 parametrizations as in the baseline. The results are displayed in table 4. Most of the robustness runs we did apply to the individual learning algorithm since SEL performs very well already in the benchmark once selective replication is allowed. Our main findings are as follows:

1. Varying the strategy set size, N or J

We find that increasing the strategy set size, N to 100 (from 30 in the benchmark) improves convergence in the modified SEL algorithm – the percentage of strategies coinciding with the optimum rises from 73.5 to 89.7. The intuition is that, with more principals in the population learning occurs faster as more strategies can be evaluated each period. The results for IEL with realized payoffs are quite different, however. Both increasing the number of strategies, J to 100 and decreasing it to 10

[28]The same run (i.e. same γ, σ and seed) was used for all the learning models.

[29]Due to space constraints, we omit reporting a large number of additional robustness checks that we performed. The results are available upon request.

generate slight drop-off in performance from the $J = 30$ benchmark. The fraction of strategies over all runs achieving the optimum goes down from 32% to 27-28%. The reason for this difference between IEL and SEL is that, when using realized payoffs only, the change in J does not affect the actual number of strategy payoffs that are updated each period. On the one hand, a lower J can be potentially beneficial for the IELR principal since a smaller number of strategies in M_t will have zero payoffs but on the other hand, it has the disadvantage of not allowing enough diversity in the strategy set which is especially important in the early stages of the learning process. In general, the results suggest that there is some optimal pool size that maximizes the algorithm's performance.

2. Varying the number of within-period output evaluations, T_1

Table 4 shows that increasing the number of output realizations, T_1, that the principal observes and uses to compute her payoff from 10 to 100 improves the performance of both the IEL and SEL algorithms. In the IELR case the percentage of strategies achieving the optimal contract rises from 32% to 51.8%, while in the SEL case the corresponding increase is a bit less significant (from 73.5% to 88.8%). Note that the IELR with $T_1 = 100$ comes within 10% of the optimal payoff in 99% of all simulations. Similarly, increasing T_1 to 100, improves the performance of IELR with $J = 10$ compared to the run with $J = 10$, and $T_1 = 10$. An increase in the length of the evaluation window, T_1 to 100 combined with a decrease in the number of strategies, J to 10 in IELR results in better performance than the baseline IELR with selective replication but worse performance than increasing T_1 alone.

3. Varying the experimentation parameters, μ, r_m, and the decay timing, $\hat{T}$

In this robustness run we explore the sensitivity of the modified IEL algorithm to variations in the parameters governing the experimentation operator. Decreasing the experimentation rate, μ from 5% to 2%, motivated by the idea that this will decrease the number of new mutant strategies with zero payoffs in the strategy pool, causes insignificant change in the performance of the algorithm. We also experimented with reducing the value of the experimentation range parameter, r_m. The motivation behind this exercise is that after an initial adjustment, the IELR algorithm leads to a strategy set settled in the area around the optimal contract. At this point, shrinking the strategy space region within which experimentation occurs can be beneficial for convergence. Indeed, we find an increase of performance of about 5 percentage points for the runs converging to the optimal payoff, but a smaller increase (about 1%) for the simulations reaching within 5% of the optimal payoff. We also studied the effect of moving forward the time period when experimentation starts decaying from $\hat{T} = 2,000$ (in the benchmark case) to $\hat{T} = 500$. The effect is a reduction in performance of about 25-30% relative to the benchmark values from table 3 as the principal has less time to experiment when trying to learn the optimal contract.

4. Tournament selection

We also check the robustness of our findings to using tournament vs. proportionate selection, as our replication operator. As table 4 shows, replacing selective replication with tournament selection in the SEL model achieves very similar results in terms of performance – over all of the runs – 69% of the strategies in the final pool coincide with the optimum under tournament selection while the corresponding number is 73.5% under selective replication. The results for the fractions of simulated payoffs within 5% and 10% of the maximum are even closer with still over 99% of all strategies achieving payoffs within 5% of the optimum. However, note that tournament selection significantly outperforms the baseline replication (see table 2).

We also look at the effect of tournament selection on the IELR model. As in the SEL case, tournament selection achieves better performance than the baseline roulette wheel replication but, unlike in the SEL case, it performs much worse than our selective replication operator (e.g., the fraction

of runs converging to the optimal contract falls from 32% to 2.6%). The reason for superior performance of selective replication is that new mutants and unplayed strategies have zero payoffs. Such strategies are replicated less frequently with selective replication than with tournament selection. This results in more successful adaptation with selective replication.

5. The fitness weights, λ

We also experimented with increasing the value of the parameter λ which governs the curvature in the mapping between the average profits and the strategy fitness. The results from increasing λ from 1 to 3 show a slight deterioration in performance (the average percentage over all runs of last period strategies coinciding with the optimum declines from 32% to 27%). Using the biased roulette wheel replication from (9), a higher λ implies a higher probability of choosing a strategy with a high payoff. This may be beneficial later on when (or, if) we are close to the optimum but may lead to the strategies in the pool being "stuck" far away from the optimum in the early stages of the learning process. The combination of these two effects accounts for the observed outcome[30].

5.4 Convergence Analysis

We report the results of our convergence analysis of the best performing social and individual learning models, SEL and IELR with selective replication in table 5. We define the following criterion for convergence: we record the time period when the algorithm first reaches the optimum, that is, the first time when the optimal strategy is played. Then, we continue the simulation for the next 200 periods and report the frequencies on how often 90% of the strategies in the pool are within a given distance of the optimum (0, 0.05 or 0.1).

SEL performs well according to this convergence criterion. Once the learning dynamics take the strategy set close to the optimum, it remains there forever (e.g. see the sample runs in figure 6). In contrast, the IELR model does not exhibit similar behavior prior to a sufficient decrease in the experimentation rate (i.e., such 'settling down' of the strategy pool occurs only after period 2,000). In the IELR simulations we often have instances where the optimal strategy shows up in the strategy pool at some period only to be wiped out shortly after by experimentation or to be replaced with another strategy with a "lucky" output draw (this is especially likely when T_1 is small). An example of this is presented in figure 7 where we show the fractions of strategies within a given Euclidean distance (0, 0.05, or 0.1) from the optimal contract for a sample IELR run. Observe that in the panel that shows the fraction of strategies equal to the optimal (that is, the fraction of the J strategies equal to s^*, f^*) there is a substantial fraction (around 40%) of strategies that are equal to the optimal one between periods 300 and 400. Later on, between periods 1,200 and 1,400, again a substantial fraction of the current strategies in the strategy set (around 30%) coincide with the optimal strategy. However, in both cases, shortly afterwards these strategies disappear from the pool and the fraction of optimal strategies in M_t remains equal to zero until the end of the simulation.

The above reasoning explains the findings reported in table 5 and suggests that the modified SEL algorithm converges, in the sense defined above, much faster than the IELR algorithm. For example, if we use our 'exactly equal to the optimum' criterion, SEL is three times faster, and up to twenty times faster according to the 'within 10% of the optimal payoff' criterion. The percentage of IELR simulations converging exactly to the optimum measured by our "90% of strategies, 90% of time" criterion is only 21% compared to 74% in the SEL case.

[30] We also tested increasing the grid spacing parameter, d from 0.01 to 0.1 for RL and EWA. This makes the strategy space G coarser and thus reduces the strategy set size, J for these models. The coarser grid helps the RL and EWA algorithms achieve slightly better performance but they remain far from being successful in adapting to the optimal contract.

We also analyze the IELR and SEL convergence rates as a function of the structural parameters γ and σ in the underlying economic model. This is shown in figure 8. The figure depicts the fraction (out of the 70 random seed runs for any fixed parameter pair) of non-convergent simulations (according to our criterion above) for the different values of the risk aversion parameter, γ and the variance of output, σ that we use. Higher output variance, σ^2 clearly hampers convergence. The intuition is that for a fixed number of output observations (T_1) the principal has a harder time assigning a theoretically correct payoff to a strategy that was played, and thus "lucky" sub-optimal strategies can outperform the optimal one in case the latter obtains a bad sequence of output draws. The role of risk aversion, γ, on convergence is not so unambiguous but there is some evidence in the figures that higher risk aversion γ makes the convergence in payoffs relatively harder.

6 Discussion on Non-Convergence

In this section we discuss in more detail the reasons for the non-convergence of the individual evolutionary learning algorithm in a large fraction of runs for both the benchmark and the modified algorithms. At first glance our finding that the standard IEL model with foregone payoffs virtually never converges to the optimal strategy seems surprising as this algorithm has been previously shown to converge fast in numerous environments (e.g., Arifovic and Ledyard, 2004, 2007). In these environments, hypothetical payoff evaluations are actually very helpful in achieving fast convergence. Foregone payoffs play a useful role in the algorithm's ability to dismiss strategies that perform poorly. Foregone payoffs also help in quickly evaluating strategies that are brought in via experimentation – only strategies that are promising in terms of foregone payoffs are kept and replicated.

However, our principal-agent problem is different and, to some extent more complex than the environments in which IEL has been previously used. As we already pointed out, the main problem with IEL's adaptation comes from the sequential nature of our theoretical setting. According to the algorithm, in order to evaluate a foregone payoff of a strategy that was not actually used, the principal assumes that the agent's action would remain constant for any other strategy played. This is correct in a simultaneous-game (Nash) setting where each player is playing against a fixed strategy distribution. However, in our model – and in any similar principal-agent or sequential-play model for that matter – different principal's strategies in fact result in different optimal actions by the agent. Under IEL, the whole collection of strategies (actually played or hypothetical) is evaluated holding the agent's action constant. Clearly, principals who use such learning algorithm and thus ignore the direct incentive effects of their actions would not be able to learn the optimal contract.

Even with the modifications of selective replication and realized payoffs evaluation we found that the IEL algorithm still experiences difficulties in learning the optimal contract in comparison to SEL. We discuss the possible reasons next. To begin with, remember that the randomly generated strategies that comprise the initial period strategy set are assigned zero payoffs[31]. In addition, any new strategy generated during the simulation via experimentation is also assigned zero payoff. Strategies that violate the participation constraint are also assigned zero payoffs. Those strategies that satisfy the participation constraint can receive positive payoffs only once they have been played. However, strategies with zero payoffs will always have a positive probability of being replicated, and unless they are replaced by a strategy with a positive payoff, they may remain idle in the principal's strategy set for a long time.

Our direct visual observations of the learning process for many sample runs indicate that it takes a fairly long time for the IELR algorithm to eliminate most of the strategies that do not satisfy the

[31]The fact that the payoff is zero is not important per se. What is important is that all these strategies have the same payoff.

participation constraint. When most of the strategies are on the "correct" side of the constraint, the algorithm displays improvement in performance in terms of strategy payoffs and closeness to the optimal contract. This is displayed in figures 9 and 10 which illustrate and compare the evolution of the strategy set, M_t, for the best performing individual and social learning models, namely SEL with selective replication (figure 9) and IELR with selective replication (figure 10). The figures plot, in strategy (s, f) space, the current strategies in the set (the circles) at various time periods and the participation constraint, for the same sample run. The diamond denotes the currently played strategy for IELR, and the star denotes the optimal contract. We see that the better performance of the social learning model is due to the fact that it weeds out strategies that violate the participation constraint quite fast and then converges quickly and stays close to the optimum. In contrast, IELR exhibits 'cycles', i.e., the strategies get close to the optimum but then the set spreads out again. It is only once experimentation decays sufficiently that IELR converges to the currently best strategy in M_t which, however, is not necessarily the optimal one.

The main problem with IELR and the reason for the behavior exhibited in figure 10 is that, even if the optimal strategy appears in the pool at some t, it can, at any later point in time, disappear from it. This decreases the probability that the algorithm converges to the optimal contract in any fixed number of periods. To see this more clearly, suppose that the optimal strategy were present in the principal's strategy set at some period. One way through which it can be replaced by some different strategy is via experimentation. A second possibility is for it to be replaced by some suboptimal strategy. The latter can happen in two ways: (1) because a suboptimal strategy was actually played and received higher payoff than the optimal strategy that was never played (note this cannot happen in SEL as all strategies in M_t are played before updating); or (2) because a suboptimal strategy was "lucky" and got a series of favorable output draws which resulted in a payoff higher than the payoff that the optimal strategy earned the last time it was played (this is less likely to happen in SEL where multiple copies of the optimal strategy would be present in M_t). In both cases, if the number of other replicates of the optimal strategy in the collection is small or zero (much more likely for IELR than SEL), this can be detrimental. The chance of bringing the optimal strategy back into the strategy pool, especially towards the end of a simulation run when we are either decreasing the radius of the experimentation, r_m (or its rate, μ) is clearly diminishing to zero.

Similarly, given that under IELR not all strategies in the pool are evaluated each period (only a single one is), previously 'lucky' strategies (those that have obtained high payoffs because of good output draws) can persist in the pool while better strategies in terms of theoretically expected payoffs could be replaced. As the simulation is moving closer to the optimal contract in (s, f) space, because of experimentation, there might still be a number of points in the strategy pool outside of the participation constraint, but close to the optimal and near-optimal strategies in the strategy space. This also contributes to IELR's slow convergence or even the lack of convergence to the optimum.

The above discussion and evidence suggests that SEL focuses faster on a smaller number of strategies than the IELR. This point is further illustrated in figure 11 in which we report the frequency distribution of all the strategies that were 'active' (played) during a given simulation. For each strategy in G we plot the number of times this strategy was 'active' divided by the total number of strategy evaluations that took place over the course of the simulation. Comparing the IELR vs. SEL panels, we see that the SEL simulation results in higher frequencies of fewer strategies that are concentrated around the participation constraint and around the optimal strategy. At the same time, the IELR simulation generates a much more spread-out distribution in which a larger number of dispersed strategies including many on the 'wrong' side of the participation constraint were played.

Does the SEL algorithm perform better because N strategies (although not necessarily all different)

are evaluated per period while only a single strategy is evaluated in IELR each period? To put IELR on a more equal footing with SEL in terms of the total number of evaluated strategies, in figure 12 we compare the make-up of the strategy sets at time periods for which the number of strategy evaluations in IELR roughly equals that in SEL[32]. More specifically, we run IELR for 30 (=N) times as many periods as SEL (15,000 vs. 500) – for example, the top row of panels on figure 12 compares the strategy pool at $t = 200$ for SEL with $t = 6,000$ for IELR. Clearly, even with the same number of evaluations, the IELR strategy set remains much more dispersed. Looking at the fraction of converging strategy under IELR over time, we see that after some initial progress, these fractions stabilize around 0.1 for strategies equal to the optimum and around 0.25 for strategies within 0.05 away from the optimum and stay at these levels until the end of a simulation.

These results reveal that, since SEL is different from simply 'repeating' IELR N times or alternatively, an N-player version of IELR, the under-performance of individual learning in our setting is not simply due to the number of strategy evaluations that are performed. What causes the difference in performance then? Remember, IELR updates (via replication and experimentation) each time a strategy is played, while SEL updates once all N strategies in M_t are played. Thus SEL uses more information to update the strategy set and experimentation occurs less often. In IELR, as shown above, zero-payoff strategies stay in the pool for a relatively long periods of time as they are unlikely to be selected to play because of their low fitness. of their low fitnnesses they are unlikely to get played. In contrast, in SEL such strategies are played with certainty and weeded out quickly. One way to make IELR perform like SEL would be to 'force' the principal to postpone replication and experimentation and instead play all her J (=N) current strategies before updating. However, there is no reason for this to be optimal in general. Moreover, this major difference in SEL vs. IELR is not just a modeling assumption. Instead it is inherent to what these learning algorithms stand for – in SEL one, and everyone, learns from one's own and other people's experiences while in IELR one learns from one's own experience only. The physical time to updating is the same (every time period), however in IELR one (being alone) is able to play just a single strategy at a time, while in SEL N strategies are played.

7 Conclusions

We introduce learning in a principal-agent model and examine whether and what type of learning processes converge to the theoretically optimal agency contract. Solving for the optimal contract in such models is often computationally difficult and may require a fair amount of knowledge about the environment by the principal. The learning models that we analyze are social evolutionary learning (SEL) and three different models of individual learning: reinforcement learning (RL), experience weighted attraction learning (EWA), and individual evolutionary learning (IEL). In addition, we introduce and evaluate a modified version of IEL that we call individual evolutionary learning with realized payoffs (IELR).

Our results show that learning in the principal-agent environment is very difficult. This is due to three main reasons: (1) the stochastic environment, (2) the discontinuity in payoffs in a neighborhood of the optimal contract due to the participation constraint and (3) incorrect evaluation of foregone payoffs in the sequential game principal-agent setting. The first two factors apply to all the learning algorithms that we study, while the third one is the main reason for the failure of the EWA and IEL models in our setting. In terms of the performance, we show that SEL (especially with selective replication) is the most successful in achieving convergence to the optimal contract. In contrast, the

[32]In fact, since the strategy pool under SEL typically contains several replicates of the same strategy, this exercise gives an advantage to IELR in terms of total number of strategy evaluations.

canonical versions of the individual learning algorithms (IEL, RL and EWA) fail to converge.

To overcome the difficulties with convergence to the optimal contract for the individual learning algorithm and remedy the problem of incorrect evaluation of foregone payoffs, we modified the IEL algorithm so that only the payoffs of strategies that are actually tried out are updated. This updating rule is similar to that used in RL. However, we keep all other essential elements of the IEL updating, namely its replication and experimentation operators that enable IEL to handle environments with large strategy space much better than RL. The resulting hybrid algorithm (which we call IEL with realized payoffs, IELR) is much more successful than all other individual learning models we studied. The implementation of selective replication improves its performance further. Nevertheless, our main conclusion, based on numerous robustness checks and parametrizations, is that, in both the canonical algorithms and their modified versions, principals who are learning from each other can achieve much better results than principals who only learn on their own. Our results thus echo the famous quote from Marshall's (1920) *Principles of Economics*: "If one man starts new idea, it is taken up by others and combined with suggestions of their own; and thus becomes the source of new ideas".

More generally, our findings suggest that the relatively simple learning rules we evaluate in this paper, despite having proved successful in many normal form game settings (largely based on Nash play) or various macroeconomic settings, may not be well-suited for the sequential strategic problems of structuring incentives that arise in the principal-agent models. Our results suggest that these environments might require much more sophisticated principals in terms of the knowledge about the environment they operate in, for example, about the nature of the participation and incentive constraints, the nature of the stochastic process that the technology is subject to and degree to which agents' are risk averse. Correct implementation of this knowledge would also require greater computational ability than what our boundedly rational agents are endowed with.[33]

We chose a one-sided learning framework for its tractability and to isolate better the relative performance of the various learning algorithms. Naturally, ours is thus a partial characterization of the real-world principal-agent environments where both sides can continuously and simultaneously learn from each other. While this remains outside the scope of this paper, we conjecture that allowing for double-sided learning in our setting is likely to make the performance of the algorithms worse. Indeed, due to the one-shot structure of our principal-agent game with the agent playing last, there is no strategic advantage to the agent of being fully rational – she cannot 'manipulate' to her benefit the principal's choice of (s, f) and make the principal deviate from the optimal contract. Thus, modeling the agent as fully rational does not make learning harder for the principal. On the contrary, if the agents were also boundedly rational and learning about how much effort to supply, then for example, choosing an incorrect level of effort may lead the principal to pick 'wrong' contracts which is likely to slow the learning process for both parties. Finally, making the agents learn about their optimal effort, z would not make the problem of incorrect evaluation of hypothetical payoffs disappear. Remember, the nature of that problem lies in the principal's taking the agent's reaction to any other contract offered as fixed. We know that a fully rational agent varies his effort one-to-one with the share s and there is no reason that a learning agent would not vary his effort with different offered contracts as well. As long as the latter is the case, EWA or IEL would fail also with two-sided learning.

How could we test our results empirically? As a first step, we plan to use experiments with human subjects. We are developing an experimental design for two types of environments. In the first environment, subjects (who play the role of the principals in the model) will learn from their own experience only. In the second environment, subjects will be given a chance to interact with other

[33]Designing a learning model with just the 'right' amount of sophistication necessary for successful adaptation in such settings is on our future research agenda.

subjects and exchange ideas of what the best strategies are. Given sufficient experimental or other data, we can formally test and distinguish statistically between the social and individual learning models[34]. IEL agents learn from their own experience and therefore only the time they have spent on the task (plus maybe an idiosyncratic shock or fixed effect) should affect how well they perform. In contrast, SEL agents learn from others, in addition to what they can achieve on their own. Thus how many other agents one is in contact with (e.g. data on social networks, friends, neighbors, etc.) should affect their performance, in addition to their own experience. That is, both the strength of the relationships an agent has with others and their number should influence her outcome under SEL but not under IEL.

In future work we plan to apply the basic framework and methodology laid down here to other principal-agent settings. We have already studied an application to an adverse selection environment (Arifovic and Karaivanov, 2009) and intend to develop an application to a dynamic models of asymmetric information (e.g. as in the optimal taxation or optimal insurance literatures).

References

[1] Arifovic, J. (1994), "Genetic Algorithm and the Cobweb Model", *Journal of Economic Dynamics and Control*, 18, pp. 3-28.

[2] Arifovic, J. (1996), "The Behavior of the Exchange Rate in the Genetic Algorithm and Experimental Economies ", *Journal of Political Economy*, 104, 510-541.

[3] Arifovic, J. (2000), "Evolutionary Algorithms in Macroeconomic Modeling: A Survey ", *Macroeconomic Dynamics* 4, pp. 373-414.

[4] Arifovic, J. and A. Karaivanov (2009), "Social Learning in a Model of Adverse Selection", forthcoming in D. West, G. Dow, and A. Eckert (eds.), *Essays in Honour of B. Curtis Eaton*, University of Toronto Press

[5] Arifovic, J. and J. Ledyard (2004), "Scaling Up Learning Models in Public Good Games, *Journal of Public Economic Theory* 6, pp. 205-38.

[6] Arifovic, J. and J. Ledyard (2007) "Call Market Book Information and Efficiency ", *Journal of Economic Dynamics and Control*, 31, pp. 1971-2000.

[7] Arifovic J. and M. Maschek (2006) "Revisiting Individual Evolutionary Learning in the Cobweb Model. An Illustration of the Virtual Spite-Effect", *Computational Economics*, 28, pp. 333-354.

[8] Arrow, K. (1962) "The Economic Implications of Learning by Doing ", *Review of Economic Studies*, 29, pp. 155-73.

[9] Arrow, K. (1985), "Informational Structure of the Firm", *American Economic Review*, 75(2), pp. 303-07.

[10] Bolton, P. and M. Dewatripont (2005), *Contract Theory*, The MIT Press, Cambridge, MA.

[11] Boyd, J. and R. Bresser (2004), "Momentum, Imitation, and Learning: Evidence from and Effects on the U.S. Retail Industry", working paper, Free University Berlin.

[34] For example, we can use the maximum likelihood based model comparison approach of Karaivanov and Townsend (2009).

[12] Camerer, C. (2003), *Behavioral Game Theory: Experiments in Strategic Interaction*, Princeton University Press, Princeton, NJ.

[13] Camerer, C. and T. Ho (1999), "Experience Weighted Attraction Learning in Normal Form Games", *Econometrica* 67, pp. 167-88.

[14] Caves, R., Crookell, H., and Killing, J. (1983), "The Imperfect Market for Technology Licenses", *Oxford Bulletin of Economics and Statistics*, 45, pp. 249-67.

[15] Chao, K. (1983), "Tenure Systems in Traditional China", *Economic Development and Cultural Change* 31, pp. 295-314.

[16] Conley, T. and C. Udry (2005), "Learning About a New Technology: Pineapple in Ghana", working paper, University of Chicago.

[17] Cunningham, R. (2004), "Investment, Private Information, and Social Learning: A Case Study of the Semiconductor Industry", Bank of Canada Working Paper #2004-32.

[18] Dawid, H. (1999), *Adaptive Learning by Genetic Algorithms: Analytical Results and Applications to Economic Models*, 2nd revised and extended edition, Springer, Berlin.

[19] Dutta, S. and X. Zhang (2002), "Revenue Recognition in a Multiperiod Agency Setting", *Journal of Accounting Research* 40, pp. 67-83.

[20] Erev, I. and Haruvy, E. (2008), "Learning and the Economics of Small Decisions ", *Handbook of Experimental Economics*, vol. 2, J. Kagel and A. Roth (eds.), forthcoming.

[21] Erev, I. and A. Roth (1998), "Predicting How People Play Games: Reinforcement Learning in Experimental Games with Unique, Mixed Strategy Equilibria ", *American Economic Review* 88, pp. 848-81.

[22] Foster, A. and M. Rosenzweig (1996), "Learning by Doing and Learning from Others: Human Capital and Technical Change in Agriculture", *American Economic Review* 103(6), pp. 1176-1209.

[23] Fudenberg, D. and D. Levine (1998) "The Theory of Learning in Games", in *Series on Economic Learning and Social Evolution*, vol. 2, MIT Press, Cambridge, MA.

[24] Hart, O. and B. Holmstrom (1987), "The Theory of Contracts" in T. Bewley (ed.), *Advances in Economic Theory*, Cambridge University Press.

[25] Holmstrom, B. (1979), "Moral Hazard and Observability", *Bell Journal of Economics*, 10(1), pp. 74-91.

[26] Holmstrom, B. and P. Milgrom (1987), "Aggregation and Linearity in the Provision of Intertemporal Incentives", *Econometrica* 55, pp. 303-28.

[27] Holmstrom, B. and P. Milgrom (1991), "Multitask Principal-Agent Analyses: Incentive Contracts, Asset Ownership, and Job Design", *Journal of Law, Economics and Organisation*, 7, pp. 24-52.

[28] Hommes, C. and T. Lux (2008), "Individual Learning, Heterogeneity and Aggregate Behavior in Cobweb Experiments ", manuscript.

[29] Karaivanov, A. and R. Townsend (2009), "Enterprise Dynamics and Finance: Distinguishing Mechanism Design from Exogenously Incomplete Markets Models", working paper, University of Chicago.

[30] Lafontaine, E. (1992), "How and Why the Franchisors Do What They Do: A Survey Report", in P.J. Kaufmann, ed. *Franchising: Passport for Growth and World of Opportunity*, 6th Annual Proceedings of the Society of Franchising.

[31] Lettau, M. (1997), "Explaining the Facts with Adaptive Agents: The Case of Mutual Fund Flows", *Journal of Economic Dynamics and Control* 21, pp. 1117-47.

[32] Lettau, M. and H. Uhlig (1999), "Rules of Thumb versus Dynamic Programming", *American Economic Review* 89, pp. 148-74.

[33] Lucas, R. (1988), "On the Mechanics of Economic Development", *Journal of Monetary Economics*, 22, pp. 3-42.

[34] Lux, T., and S. Schornstein (2005), "Genetic Learning as an Explanation of Stylized Facts of Foreign Exchange Markets ", *International Journal of Mathematical Economics*, 41, 169-196.

[35] Marshall, A. (1920), *Principles of Economics*, 8th edition, McMillan, London.

[36] Marks, R.E. (1998), "Evolved Perception and Behavior in Oligopolies", *Journal of Economic Dynamics and Control*, 22, 1209-33.

[37] Masten, S. and E. Snyder (1993), "United States v. United Shoe Machinery Corporation: On the Merits," *Journal of Law and Economics*, 36, pp. 33-70.

[38] Phelan, C. and R. Townsend (1991), "Computing Multi-Period, Information-Constrained Equilibria", *Review of Economic Studies*, 58, pp. 853-881.

[39] Rogerson, W. (1985), "The First Order Approach to Principal-Agent Problems", *Econometrica* 53, pp. 1357-68.

[40] Romer, P. (1986) "Increasing Returns and Long-Run Growth", *Journal of Political Economy*, 94(5), pp. 1002-37.

[41] Roth, A. and I. Erev (1995), "Learning in Extensive Games: Experimental Data and Simple Dynamic Model in the Intermediate Term ", *Games and Economic Behavior* 8, pp. 164-212.

[42] Rose, D. and T. Willemain (1996), "The Principal-Agent Problem with Evolutionary Learning ", *Computational and Mathematical Organization Theory* 2, pp. 139-62.

[43] Ryu, S., H. Rao, Y. Kim and A. Chaudhury (2005), "Knowledge Acquisition via Three Learning Processes in Enterprise Information Portals: Learning-by-investment, Learning-by-doing and Learning-from-others ", *MIS Quarterly* 29(2), pp. 245-78.

[44] Sen, K. (1993), "The Use of Initial Fees and Royalties in Business-Format Franchising", *Managerial and Decision Economics*, Vol. 14, pp. 175-190.

[45] Singh, P., N. Youn and Y. Tan (2006), "Developer Learning Dynamics in Open Source Software Projects: A Hidden Markov Model Analysis", working paper, University of Washington.

[46] Stiglitz, J. (1974), "Incentives and Risk Sharing in Sharecropping", *Review of Economic Studies*, 41(2), pp. 219-55.

[47] Stokey, N. (1988), "Learning by Doing and the Introduction of New Goods ", *Journal of Political Economy*, 96, pp. 701-17.

[48] Thornton R. and P. Thompson (2001), "Learning from Experience and Learning from Others: An Exploration of Learning and Spillovers in Wartime Shipbuilding", *American Economic Review* 91(5), pp. 1350-68.

[49] Townsend, R. (1982), "Optimal Multi-Period Contracts and the Gain from Enduring Relationships under Private Information", *Journal of Political Economy* 61, pp. 166-86.

[50] Vriend, N. (2000), "An Illustration of the Essential Difference between Individual and Social Learning and its Consequences for Computational Analyses", *Journal of Economic Dynamics and Control* 24, pp. 1-19.

[51] Zhang, X., S. Fan and X. Cai (2002), "The Path of Technology Diffusion: Which Neighbors to Learn From?", *Contemporary Economic Policy* 20(4), pp. 470-78.

Table 2: Algorithm Performance - Benchmark [1,2]

Model	Percent of last period strategies within x of the optimal contract			Percent of last period payoffs within x% of the optimal contract		
	x=0	x=0.05	x=0.1	x=0	x=5	x=10
Individual Evolutionary Learning, IEL	0.00	0.00	0.00	0.00	0.00	0.00
Social Evolutionary Learning, SEL	1.65	18.59	41.51	1.65	37.43	67.81
Reinforcement Learning, RL (d = 0.01)	0.03	1.96	6.27	0.03	1.67	3.73
EWA Learning (d = 0.01)	0.00	0.79	3.99	0.00	0.52	1.17

Table 3: Algorithm Performance - Modified

Model	Percent of last period strategies within x of the optimal contract			Percent of last period payoffs within x% of the optimal contract		
	x=0	x=0.05	x=0.1	x=0	x=5	x=10
Selective Replication						
Modified IEL (SR, hypothetical payoffs)	0.00	0.00	0.00	0.00	0.00	0.00
Modified SEL (SR)	73.50	92.15	99.70	73.50	99.74	100.00
IEL with Realized Payoffs, IELR						
using baseline replication	1.20	14.22	33.18	1.20	30.61	59.36
using selective replication	32.03	62.19	82.12	32.03	80.96	91.80

Notes:

1. SR - selective replication
2. The EWA parameters used are: δ=0.2; ρ=0.8; φ=0.8

Table 4: Algorithm Performance - Robustness Checks [1,2]

Model	Percent of last period strategies within x of the optimal contract			Percent of last period payoffs within x% of the optimal contract		
	x=0	x=0.05	x=0.1	x=0	x=5	x=10
Individual (SR, N=100)	27.66	57.84	78.40	27.66	74.94	84.80
Individual (SR, N=10)	26.88	56.97	78.69	26.88	77.63	89.06
Individual (SR, T1=100)	51.77	78.72	94.95	51.77	94.91	98.97
Individual (SR, N=10, T1=100)	42.22	71.77	91.44	42.22	92.48	98.34
Individual (SR, experimentation rate=0.02)	32.61	63.14	83.94	32.61	82.56	92.44
Individual (SR, perturbation radius decay)	37.26	67.28	88.18	37.26	85.17	94.50
Individual (SR, early experimentation decay)	18.94	44.93	67.54	18.94	77.99	90.95
Individual (tournament selection)	2.59	13.03	26.52	2.59	41.76	62.07
Individual (SR, λ=3)	27.09	59.96	81.48	27.09	78.92	88.97
Social (SR, N=100)	89.70	98.94	100.00	89.70	100.00	100.00
Social (SR, T1=100)	88.75	98.71	100.00	88.75	100.00	100.00
Social (SR, early experimentation decay)	48.57	64.04	84.60	48.57	97.54	99.77
Social (tournament selection)	68.90	90.12	99.13	68.90	99.25	99.99

Notes:

1. SR - selective replication
2. All individual runs use realized payoffs only (IELR)

Table 5: Convergence Analysis [1,2]

Model	Percent of simulations within x of optimum (90%, 90%) [3]			Average time to first convergence within x of optimum (90%, 90%)		
	x=0	x=0.05	x=0.1	x=0	x=0.05	x=0.1
Individual (SR, realized payoffs)						
-strategies	20.59	48.29	78.53	2178.50	2170.20	2137.90
-payoffs	20.59	66.15	84.57	2178.40	2165.30	2149.20
Social (SR)						
-strategies	73.82	94.39	99.99	733.21	525.72	108.27
-payoffs	73.84	99.74	100.00	733.08	345.95	102.77

Notes:

1. SR - selective replication
2. Maximum number of runs = 2400
3. (90%, 90%) means 90% of runs were within x of the optimum for 90% of a 200 consecutive periods window
   the payoff numbers for x are in fractions of optimal payoff

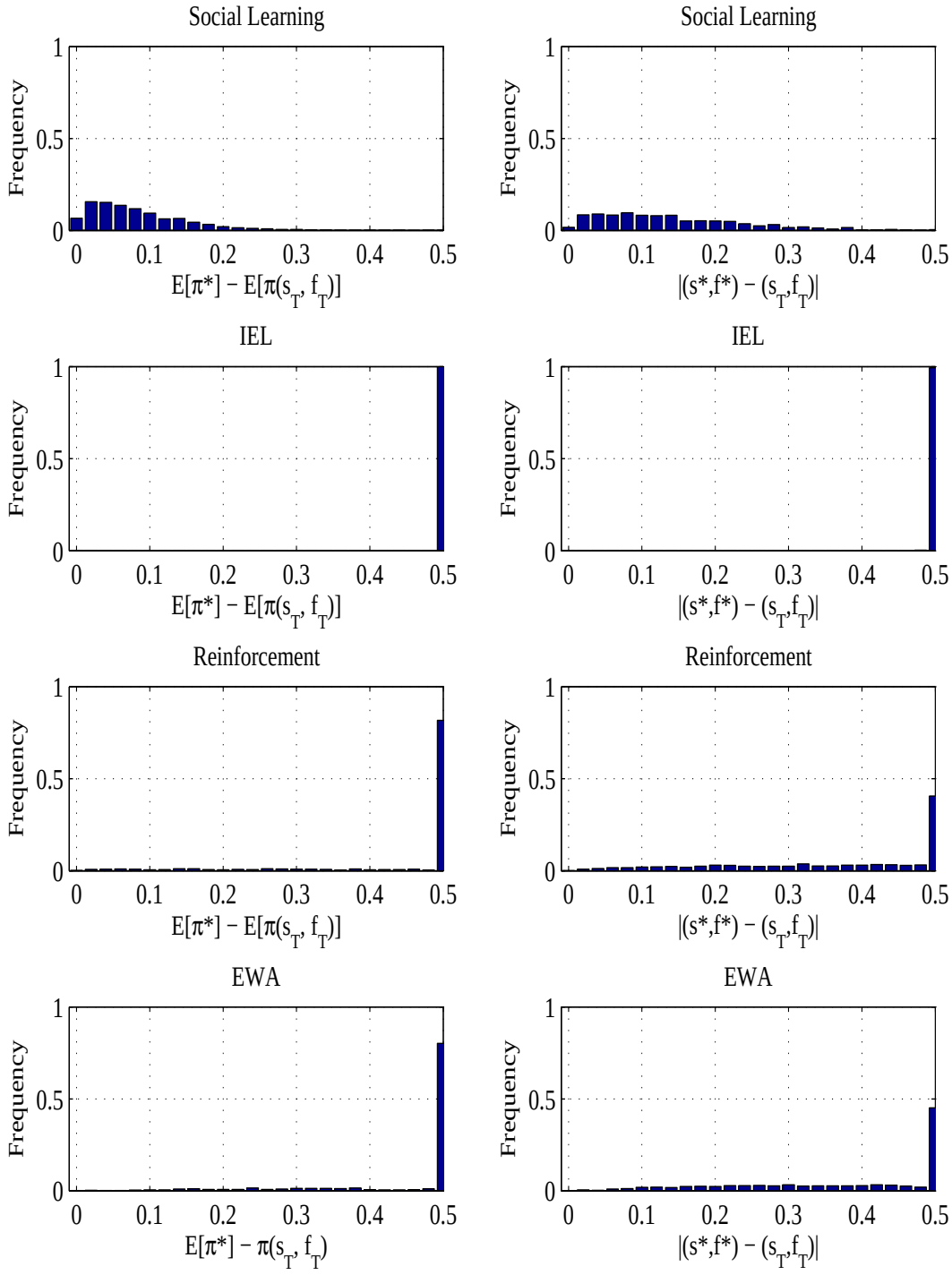

Figure 1: Differences between Simulated and Optimal Payoffs (first column) and Strategies (second column)

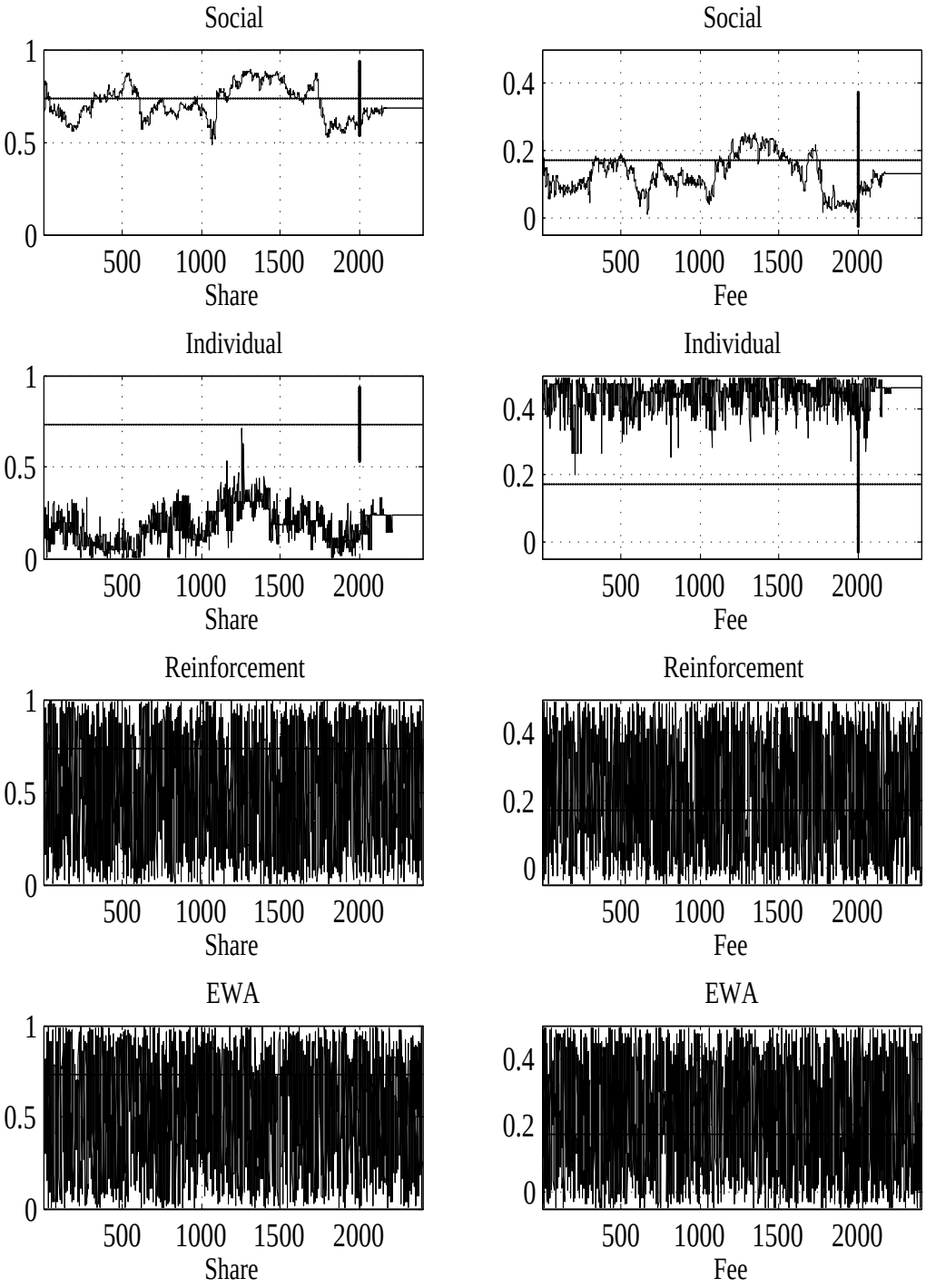

Figure 2: Message Time Paths for a Sample Run - Bemnchmark Case

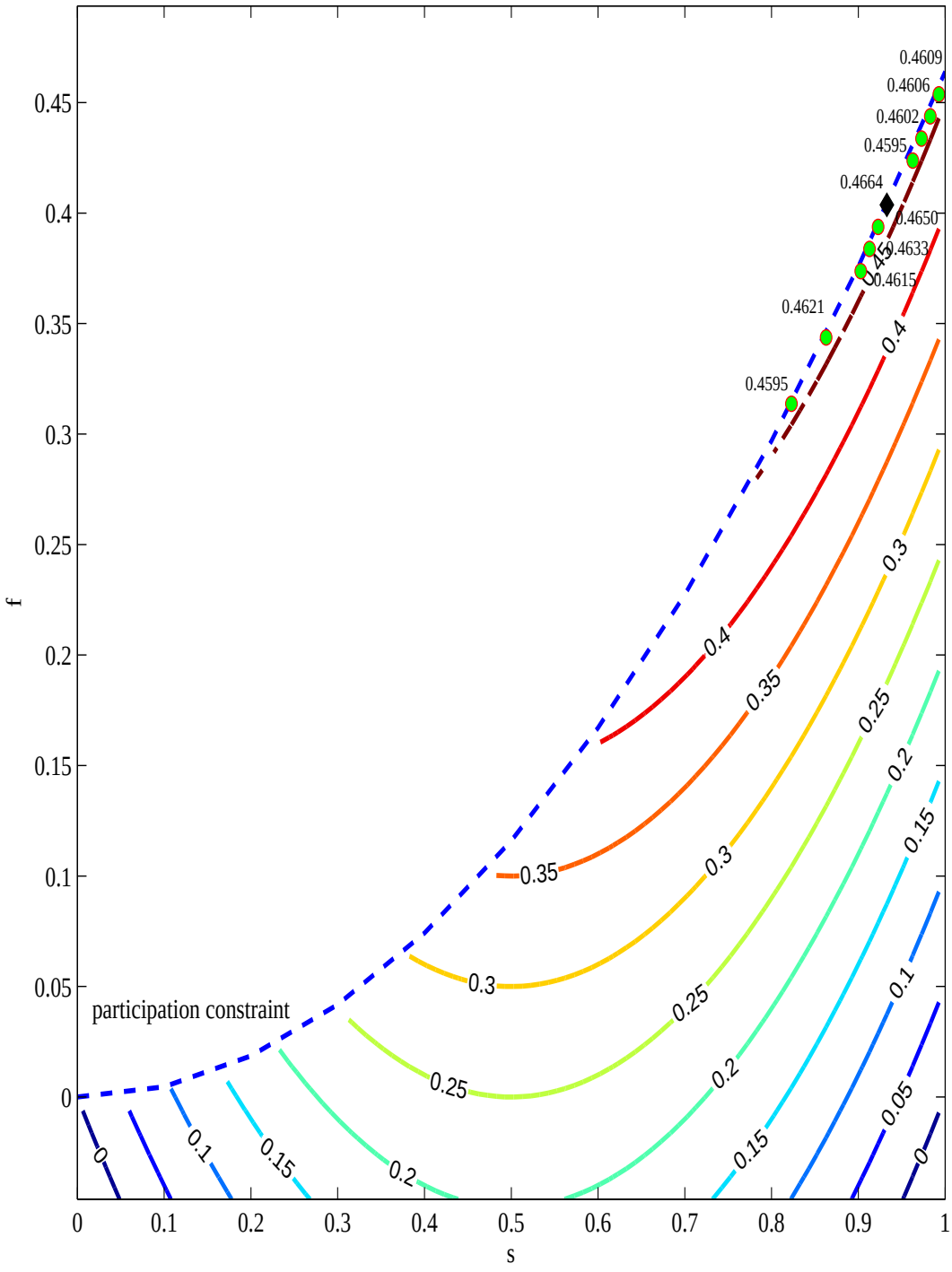

Figure 3: Iso-payoff Lines for a Typical Case

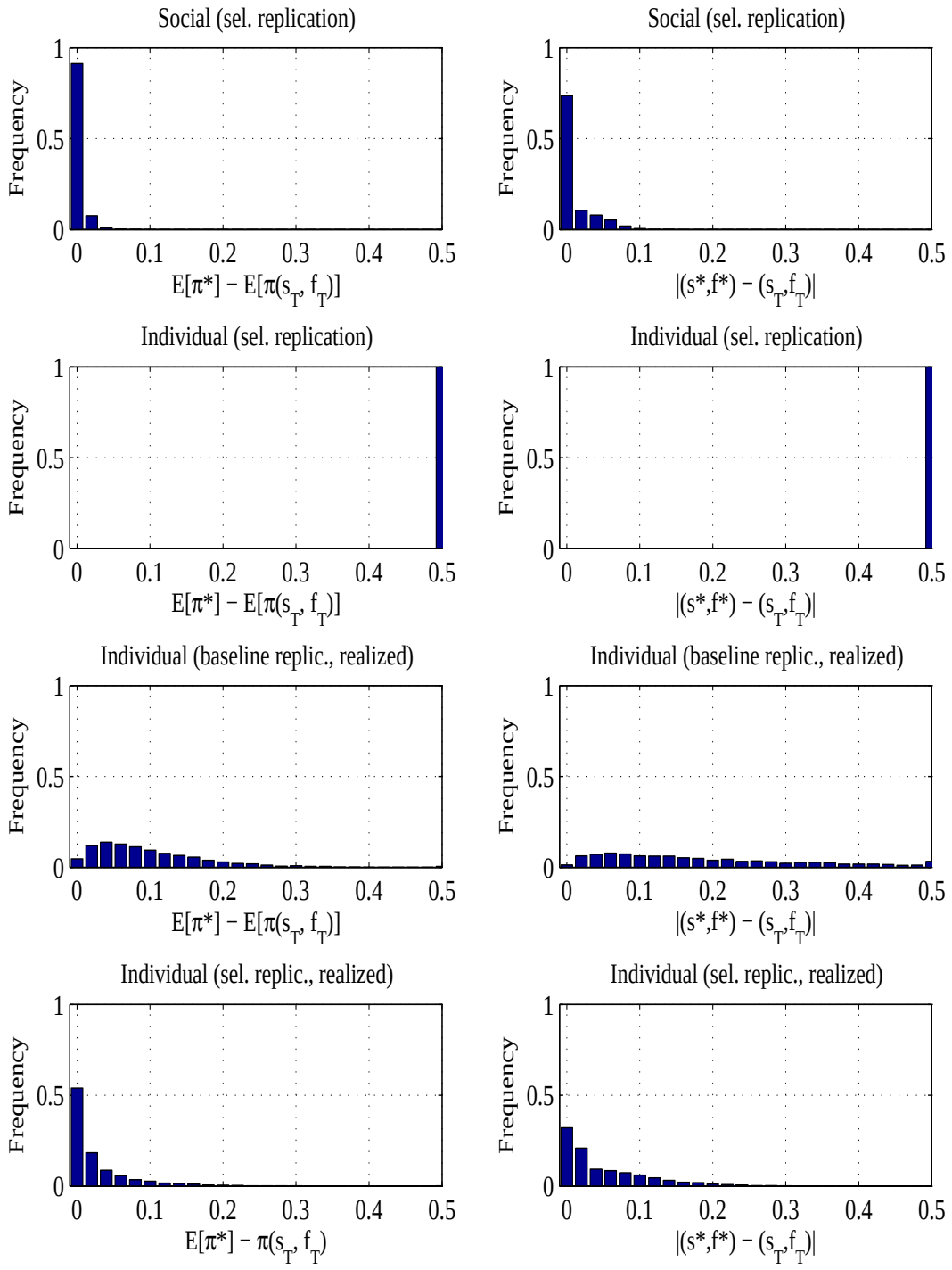

Figure 4: Differences between Simulated and Optimal Payoffs and Strategies, Last Period

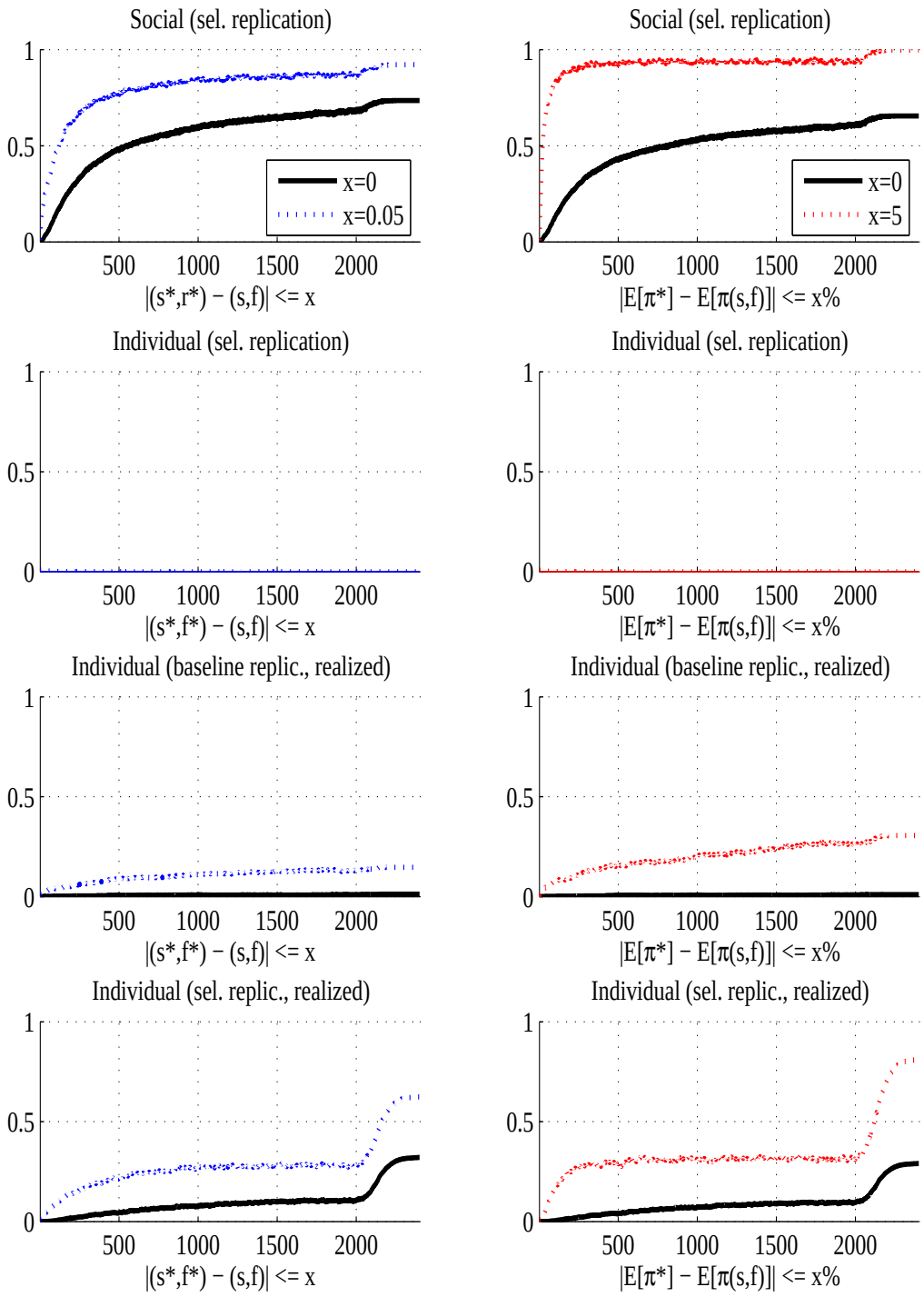

Figure 5: Time Paths of the Fraction of Simulated Strategies/Payoffs Equal to the Optimum (solid Line) and 5% of the Optimum (dotted Line)

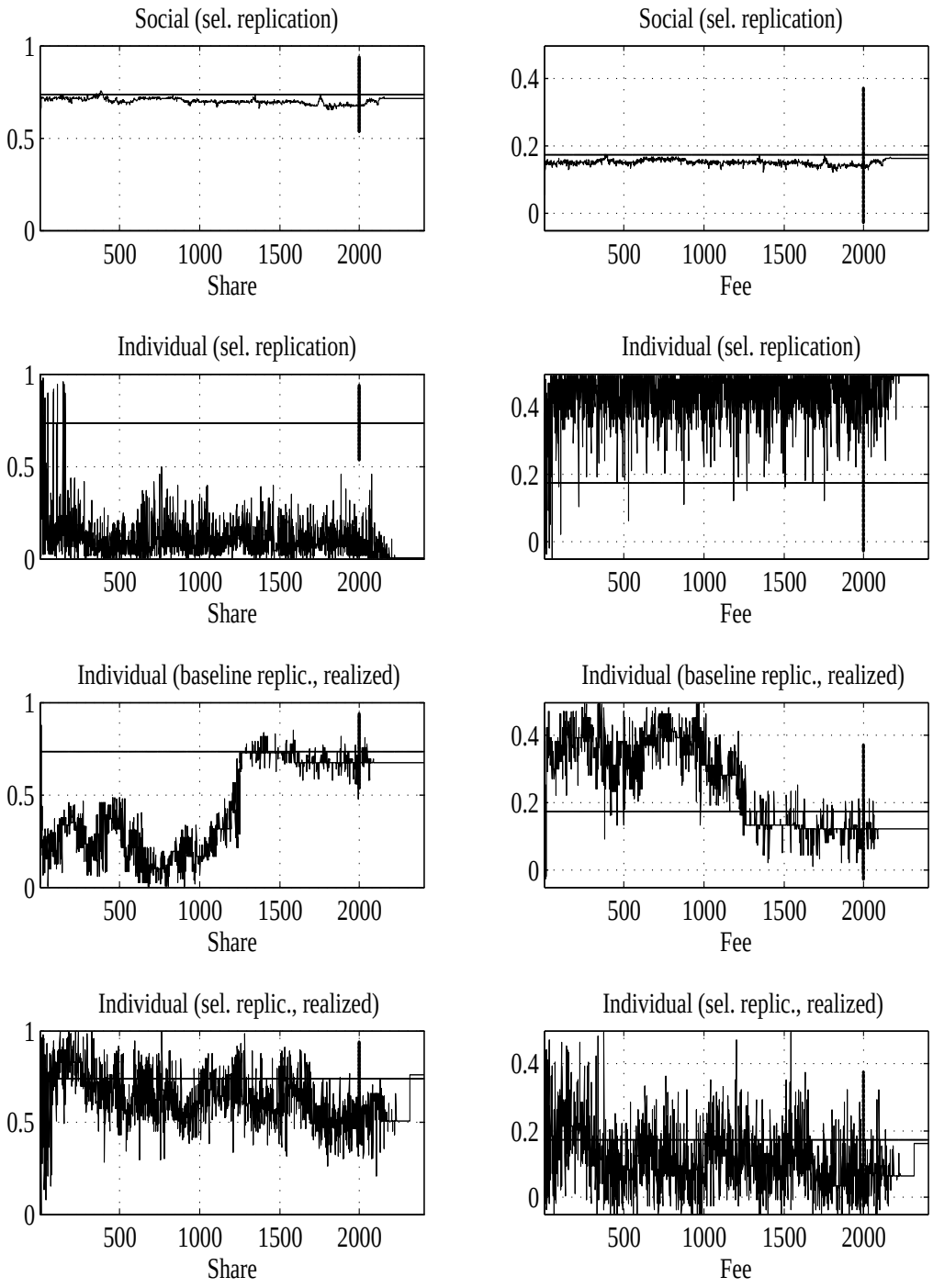

Figure 6: Strategies' Time Paths for Sample Run

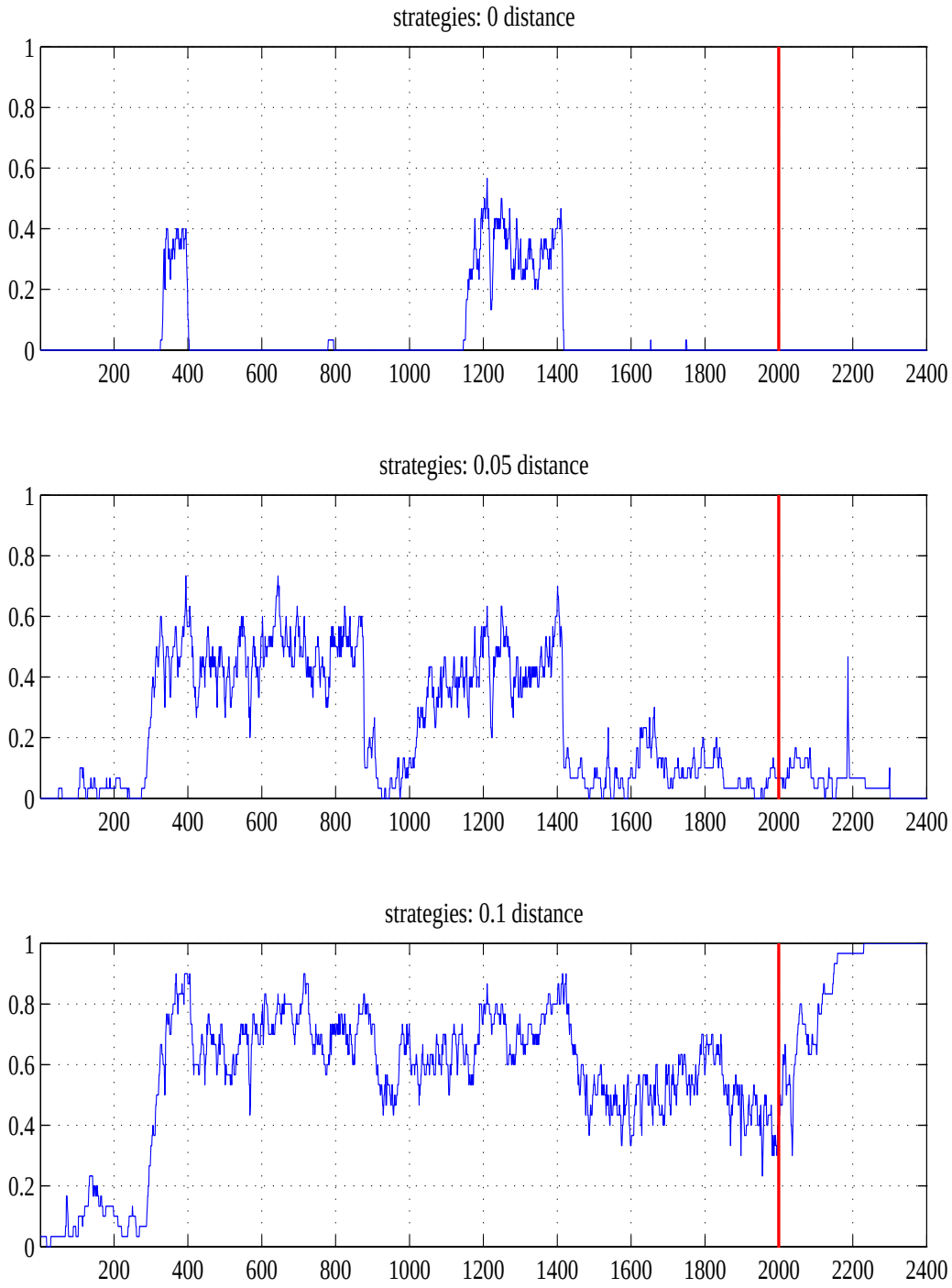

Figure 7: Sample Run for IEL with Realized Payoffs

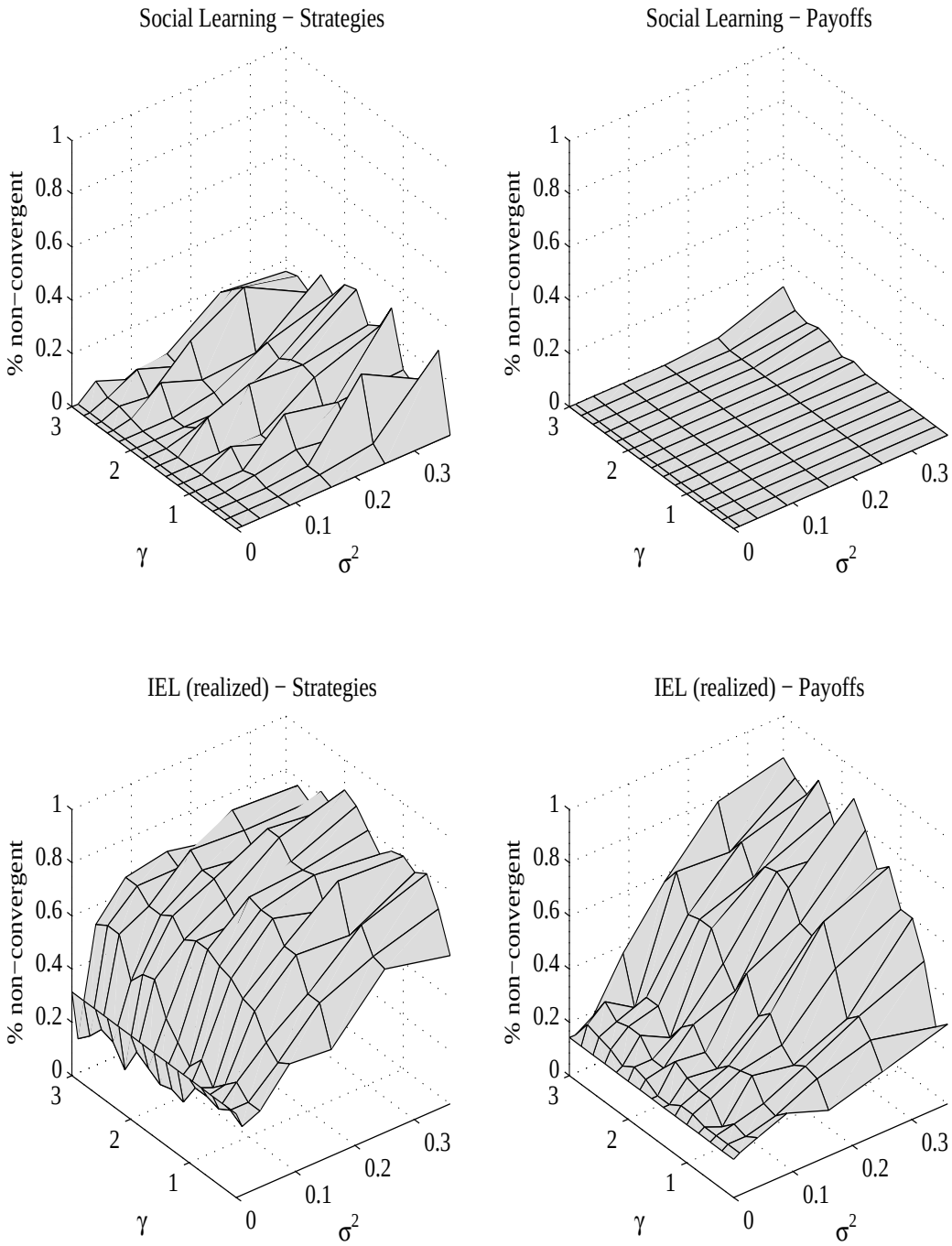

Figure 8: Fraction of Non-Convergent Simulations in Terms of Strategies and Payoffs, Social Learning and IEL with Realized Payoffs

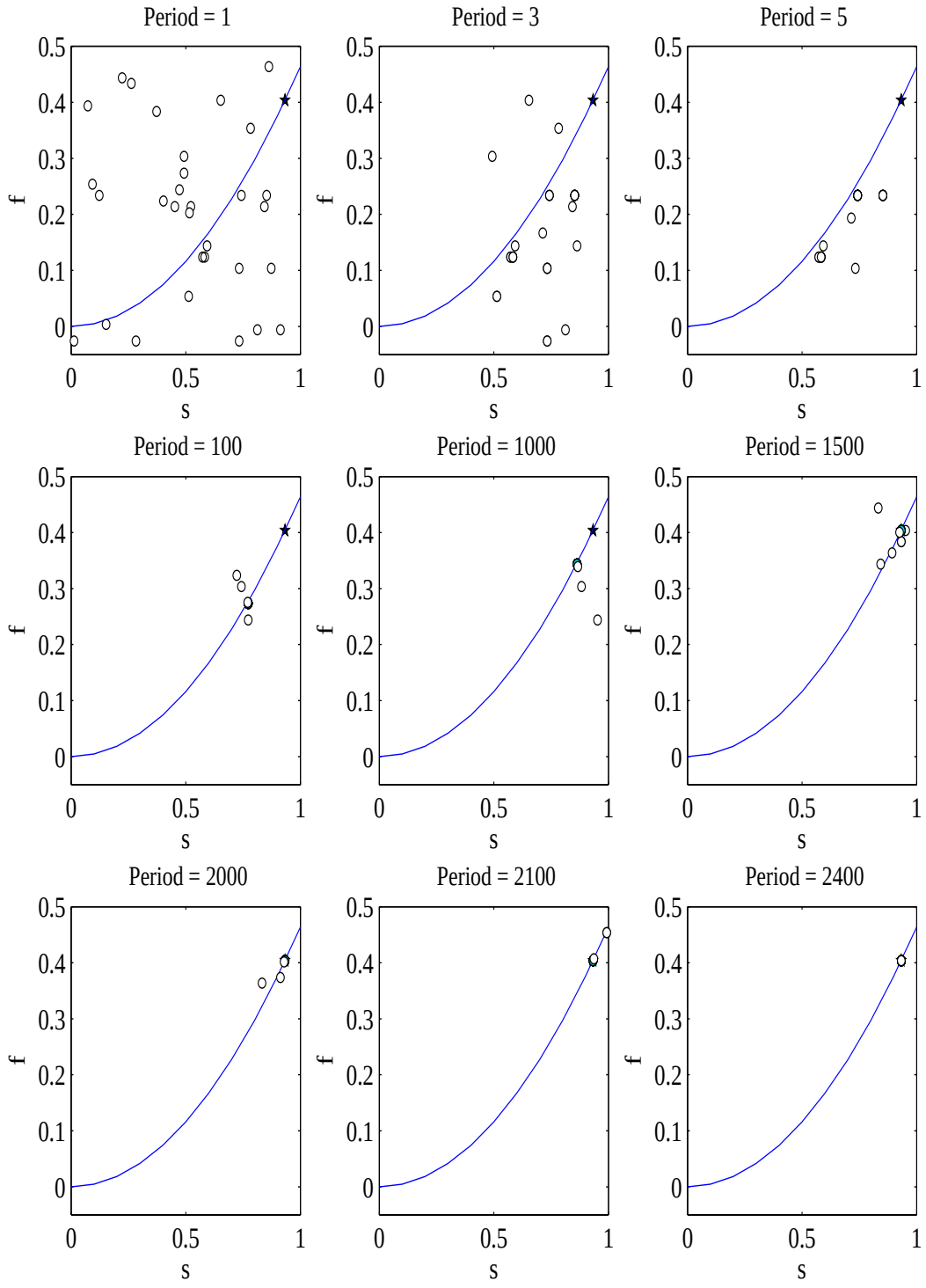

Figure 9: Evolution of a Typical Run - Social Learning

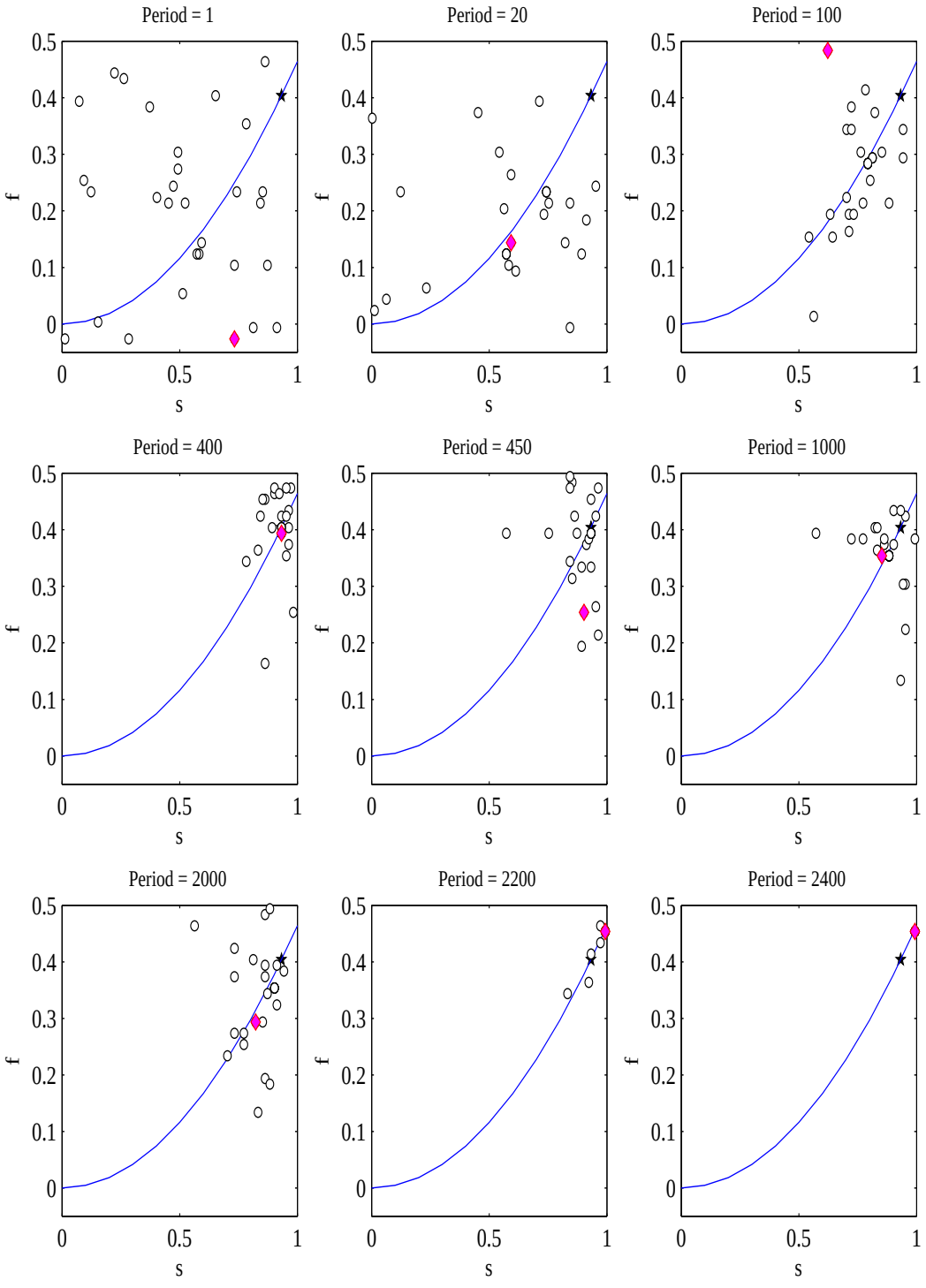

Figure 10: Evolution of a Typical Run - IEL with Realized Payoffs

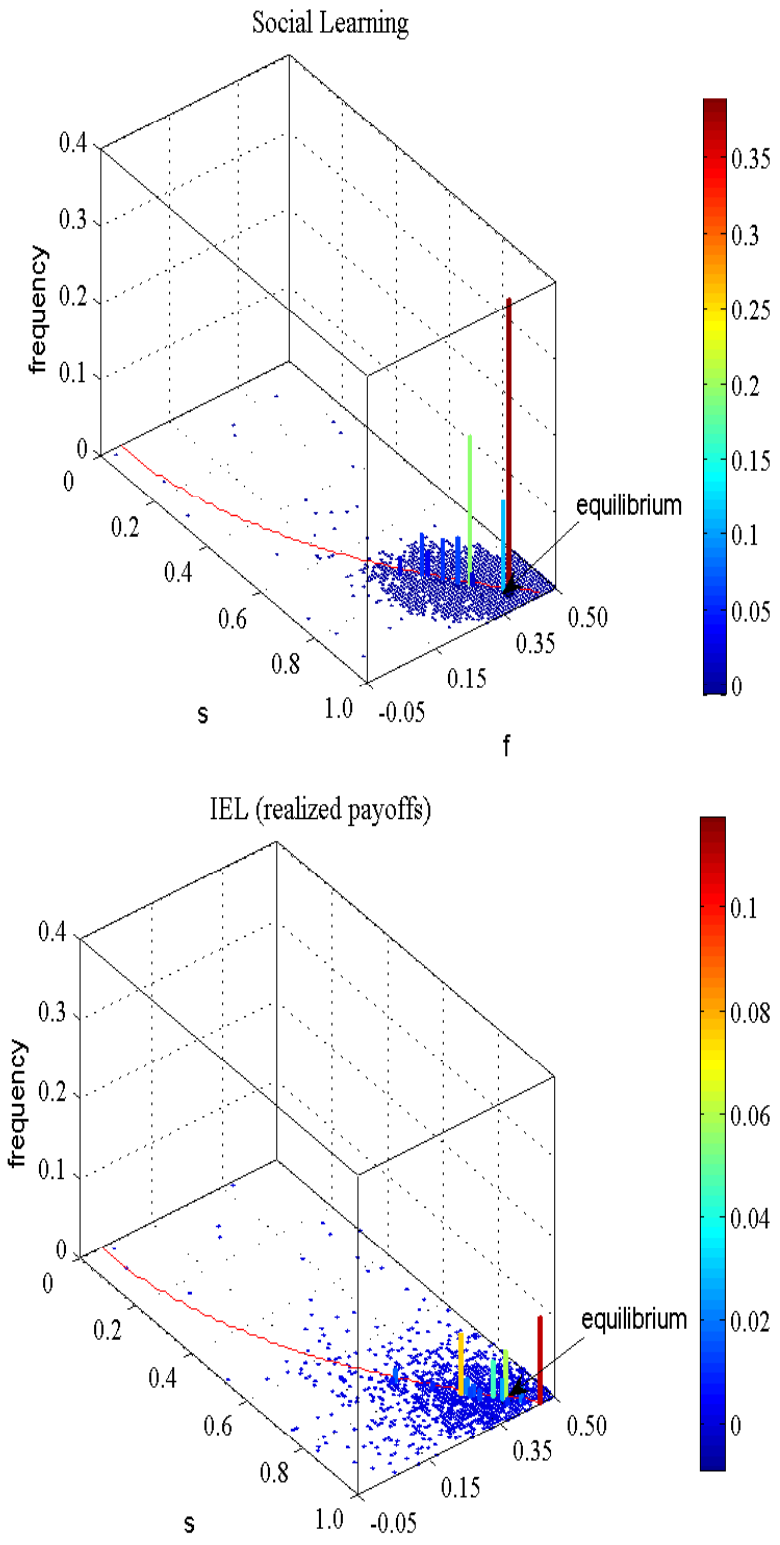

Figure 11: Frequency of Evaluations of Strategies During a Typical Run

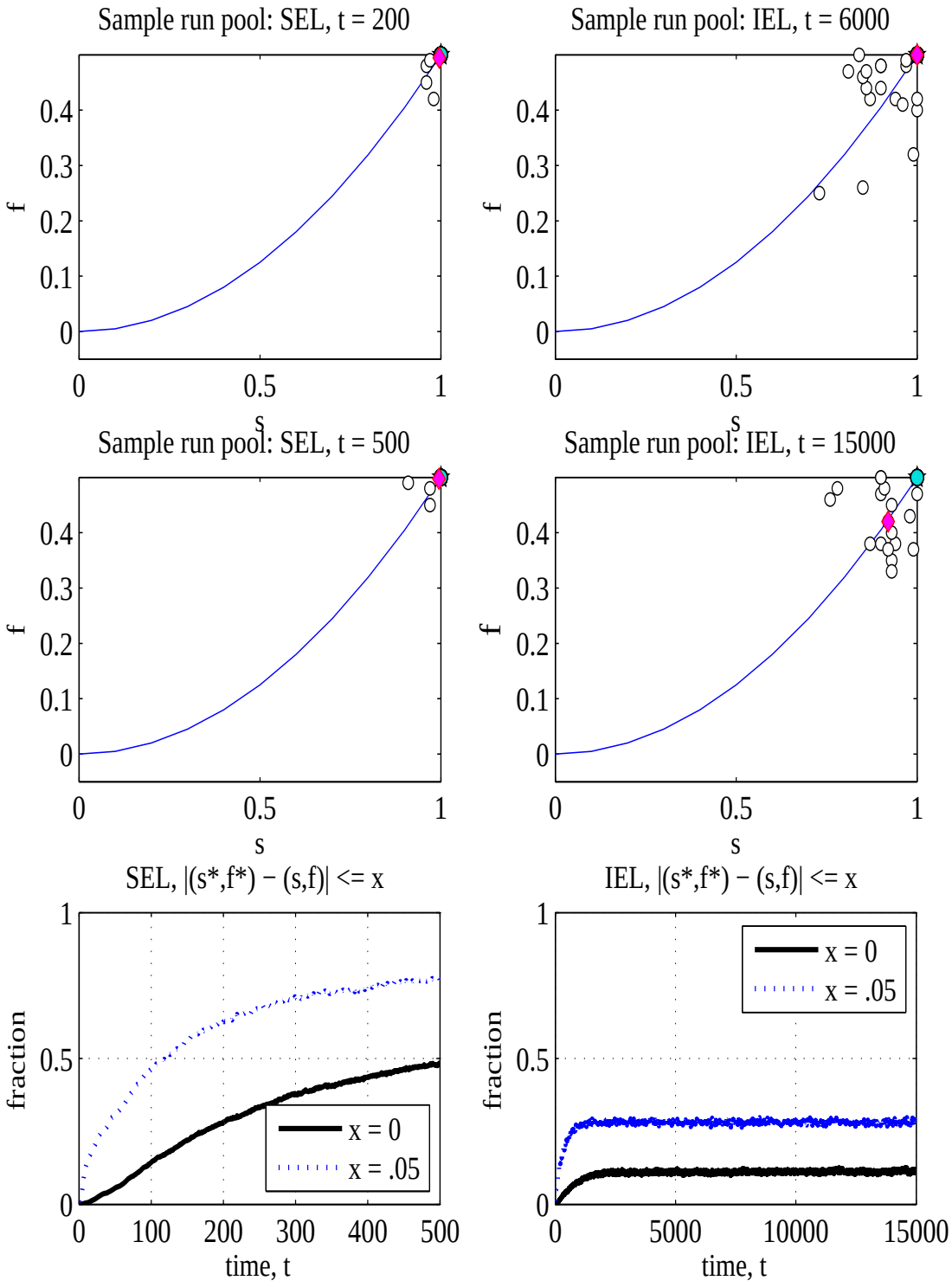

Figure 12: Equal Number of Evaluations