10ISSN 1183-1057

SIMON FRASER UNIVERSITY

Department of Economics

Discussion Papers

07-24

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

Jasmina Arifovic and
Alexander Karaivanov

November 2007

SFU Economics

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

Jasmina Arifovic† Alexander Karaivanov‡

October, 2007

Abstract

We introduce learning in a principal-agent model of stochastic output sharing under moral hazard. Without knowing the agents' preferences and 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). Overall, 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) a discontinuity in the payoff space 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 learning algorithms we study while the third is the main contributor for the failure of the EWA and IEL models. Social learning (SEL), especially combined with selective replication, is much more successful in achieving convergence to the optimal contract than the canonical versions of individual learning from the literature. A modified version of the IEL algorithm using realized payoff evaluation performs better than the other individual learning models; however, it still falls short of the social learning's ability to converge to the optimal contract.

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 as well as participants at the Society of Economic Dynamics and Computing in Economics and Finance conferences for many 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

It is well known that the optimal contracts in principal-agent settings take complicated forms due to the intricate trade-off between provision of incentives and insurance. The optimal contract depends crucially on the principal's and agent's preferences, the properties of the production technology, as well as the stochastic properties of the income process[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 has perfect knowledge of objects that are realistically 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 properties of the output technology.

In this paper, we explicitly model the principal's *learning* process based only on observable information 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 contract in a standard principal-agent model of contracting under moral hazard (e.g. Hart and Holmstrom, 1987).

We use two alternative paradigms, *social* and *individual* learning, to describe the principal's learning process. Our 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, our individual learning paradigm can be viewed as an explicit micro-level modeling of "learning by doing" (e.g. Arrow, 1962; Stokey, 1988).

A large number of empirical studies in different research areas suggest that individuals and firms utilize in practice social and individual learning methods similar to those we study. 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 in their impact 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 as predicted by the Caplin and Leahy (1994) theoretical model. Singh, Youn and Tan (2006) find similar effects in the open source software industry. In development, 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 households' own and their neighbors' experience increases 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 evidence.[3] Finally, at the macro level, the seminal work of Romer (1986) and Lucas (1988) have emphasized the role of learning spillovers as an engine of economic growth.

[1] As an example, take a standard problem of optimal contracting under moral hazard. Applications corresponding to this setting 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] Zhang, Fan and Cai (2002) also present evidence for learning from others in technology adoption using household and GIS data from rural India.

[3] Related evidence exists in the business and management literature as well. For instance, Boyd and Bresser (2004) study the occurrence and performance effects 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 industry.

For simplicity, 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, etc.) while the agent supplies unobservable labor effort. Output is 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 for the agent.

We first describe the optimal contract that arises if the contracting parties are fully rational and *know* all the ingredients of the contracting problem and the environment i.e. the technology and preferences. Then, we build a model in which a principal with no prior knowledge of the environment has to *learn* what the optimal contract is. Agents remain fully rational.[4]

We implement the social learning paradigm (learning from others) using a model of *social evolutionary learning* (SEL) where players update their strategies based on imitating strategies of those players who have performed better in the past, and occasionally experiment with new strategies. The population of players thus learns jointly through experience that they share over time.[5]

For our implementation of the individual learning paradigm (learning by doing), we evaluate three classes of individual learning, namely *Reinforcement Learning*, RL (Roth and Erev, 1995, 1998), *Experience-Weighted Attraction Learning*, EWA (Camerer and Ho, 1999), and *Individual Evolutionary Learning*, IEL (Arifovic and Ledyard, 2004, 2007).[6] 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[7]. The feature shared by all the algorithms we study is that the frequency of representation of well performing strategies increases over time and the choice of a particular strategy to be used in a given period is probabilistic.

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 update the payoffs of all strategies in any given period based on calculations of '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 the entire strategy set in the algorithms' 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 the updating process, 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.

Our main result shows that SEL almost always converges to the theoretically optimal principal-agent contract. In contrast, individual learning algorithms based on evaluation of foregone payoffs (IEL and EWA) that have proven successful in a variety of Nash type environments completely fail to adapt in our setting. RL performs somewhat better than IEL and EWA since it only updates the payoffs of

[4] We are currently working on modeling an environment where both principal and agent are learning.

[5] In papers related to ours, evolutionary learning algorithms such as genetic algorithms, classifier systems, genetic programming, evolutionary programming, etc. have been widely used in numerous applications (see Arifovic, 2000 for a survey of applications in macroeconomics; LeBaron et al., 1999 for applications in finance; and Dawid, 1999 for a general overview). 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 matter for convergence. However, they do not analyze the relative performance of different learning algorithms or study the reasons for non-convergence.

[6] The first two have been used mainly in relation to experimental data from two-person games. The third has been implemented in public good and market environments with larger strategy spaces.

[7] For economic applications see Arifovic, 1994; Marimon, McGrattan, and Sargent, 1990; Vriend, 2000.

those strategies that were actually used. However, RL's overall learning performance is not satisfactory due to its disadvantage in handling large strategy spaces.

The intuition for the failure of EWA and IEL is that, when evaluating foregone payoffs of potential strategies that have not been tried, the principal assumes that agent's action will remain constant (as if playing Nash) while in fact the optimal contract involves an optimal response to the agent's *best response function* as in a Stackelberg game. The inability of individual learning to produce correct foregone payoffs for the principal's strategies precludes its convergence to the theoretically optimal contract[8]. In contrast, SEL involves evaluation of payoffs of the strategies that are actually played, thus circumventing the problem. As a result, SEL exhibits high rates of convergence to the optimal contract.

Two additional reasons specific to our principal-agent setting cause further difficulties in convergence to the optimal contract, independently of which learning algorithm is implemented. First, the presence of stochastic shocks makes learning difficult in any type of environment. Second, in our model, payoffs are a discontinuous function of the strategy space at the agent's participation constraint. 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 is in stark contrast to the findings reported in the existing literature.[9] However, as mentioned above, our principal-agent environment is different from the environments that have been studied so far, most importantly in its sequential rather than simultaneous game nature. To address this issue, we study a modified version of the IEL algorithm where only payoffs of strategies that are actually tried out are updated. At the same time we keep the basic evolutionary updating process that has enabled IEL to adapt well in other environments with large strategy space. The resulting IEL algorithm with realized payoffs (IELR) proved more successful in converging to the optimal contract than its canonical counterpart. However, its convergence rates still fall short of those achieved under social learning.

2 Contracting under Full Rationality

2.1 The Optimal Contract

This section describes the optimal contracting problem that we study. Consider an output sharing model, for instance, the standard moral hazard model of sharecropping (e.g. Stiglitz, 1974). Alternative applications include profit sharing under franchising, licensing, or author-publishing contracts. To fix ideas, we can interpret the principal as a landlord and the agent as a tenant working on the land. Output is given by $y(z) = z + \varepsilon$ where z is the effort employed by the agent and ε is a normally distributed random shock with mean 0 and variance σ^2. Effort is unobservable/ non-contractible. Output is publicly observable. Assume that the range of possible efforts is such that the landlord cannot infer from the output realization what effort level was employed. The principal is risk neutral while the agent is risk averse with utility from consumption $u(c)$ and a cost of effort $v(z)$. The agent's outside option is $\bar{u}$.

We restrict attention to linear compensation contracts as in Stiglitz (1974), 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 output share of the

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

[9] Various studies (see, for example, Camerer and Ho, 1999; Camerer, 2003; Arifovic and Ledyard, 2004, 2007) find that the performance of these models, when evaluated against evidence from experiments with human subjects, is superior to the performance of learning models where only actual strategy payoffs are taken into account.

agent and f is a fixed payment/rent. We are aware that, in general (e.g. see Holmstrom, 1979), the theoretically optimal compensation contract may be non-linear. There are two possible justifications for restricting our analysis to linear compensation schemes. The first is computational - given that this paper is about modeling *learning* about the best compensation contract, the linearity restriction transforms the problem into learning about two scalars, s and f, as opposed to learning about a general compensation function $c(y)$ which could make the problem both analytically intractable and much harder for agents to learn. The second justification is the observation that in reality output sharing agreements very often take the linear form, e.g. see Chao (1983) on sharecropping or Lafontaine (1992), Sen (1993) on franchising[10]. Clearly, our methods can be extended to more complicated schemes as long as the strategy space remains tractable.

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

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

subject to:

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

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

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 than 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 optimum.

Under our assumptions about the relationship between effort and output and the distribution of ε, it is easy to verify that the Monotone Likelihood Ratio Property (Rogerson, 1985) holds. Hence, we can replace the incentive compatibility constraint (1) with its first order condition:

$$sEu'(s(z+\varepsilon)) = v'(z) \tag{3}$$

and solve for the optimal contract (s^*, f^*) from (3) and (2).

2.2 A Computable Example

We use the following easily computable example in our numerical analysis of learning in the principal-agent model. Assume a mean-variance utility for the tenant: $Eu(c) \equiv E(c) - \frac{\gamma}{2}Var(c)$ and cost of effort $v(z) = \frac{1}{2}z^2$. The tenant's utility is thus:

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

The ICC, (3) implies that, given the offered share, s and fixed payment, f the agent will optimally choose effort

$$z^* = s$$

[10] There also exists a volume of literature trying to reconcile this observation with theory, e.g. see Arrow (1985) and Holmstrom and Milgrom (1987).

The principal's problem then becomes (substituting from the ICC and participation constraints):

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

The f.o.c. is:

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

or

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

The optimal fixed payment, f^*, is then recovered from the participation constraint:

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

Our main objective is to examine the behavior of the above model under learning. We assume that the stage contract is repeated over time. The main reason for this is numerical and in future work we plan to investigate learning in multi-period dynamic settings. In general, contract theory suggests (e.g. Townsend, 1982) that in a dynamic framework, intertemporal tie-ins would typically exist in the optimal contract. However, if standard learning algorithms cannot converge to the optimal static contract then we would expect them to be even less successful in the full dynamic setting.

We want to examine how hard or easy it is for boundedly rational players to learn what the optimal contract looks like. Specifically, as a first pass we assume that the principal is not endowed either with the ability to optimize or with the knowledge of the physical environment, i.e., she does not know what the agent's preferences and the exogenous stochastic shock properties are. In contrast, as explained in the introduction, agents are assumed to be able to find out their optimal effort choice, z^* given the messages about the share, s, and the rent, f, that they receive from the principals.

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

3 Learning about the Optimal Contract

We investigate two learning paradigms. The first is commonly known as *social learning* where boundedly rational 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 (i.e. the contracts they offer) accordingly. Our model of social learning is based on the evolutionary paradigm where the principals' success and survival is based on how successful their strategies, (s, f) are, and on occasional experimentation with new strategies.

The second learning paradigm we study is *individual learning* where principals learn only from their own experience. Specifically, each principal is endowed with a collection of different strategies that she updates over time based on her experience. As explained above, we examine the behavior of three baseline models of individual learning that share some common features: *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).

3.1 Common Structure of the Learning Algorithms

In each of the learning models that we consider all agents are identical and optimize each time period given the contract proposed by the principal. A strategy/message[11], m_t^i, that is in the strategy set M_t at time $t \in \{1, T_0\}$ consists of a share/rent pair, i.e. $m_t^i = \{s_t^i, f_t^i\}$. The strategy set M_t of fixed size N has elements m_t^i belonging to the strategy space, G, which is a two-dimensional grid of size $\#S \times \#F$ where S and F are linearly-spaced grids[12] for the share, s, and the rent, f. The coarseness of the S and F grids is given by the parameter d, which determines the number of points on the grid. Each point in G thus represents a strategy/contract consisting of a share-rent pair.

In case of social learning, the number of principals is equal to the number of strategies, N. Thus, each principal, $i \in \{1, N\}$ has strategy m_t^i that she uses at time t. However, in case of individual learning, M_t is a collection of strategies of size N that belong to a single principal. At each t, the principal chooses one of them that she uses as her actual strategy.

Each period t consists of $T_1 \geq 1$ 'interactions' between a fixed principal-agent pair where each interaction[13] corresponds to a separate output shock (ε draw). Every period, t, the principal announces a single message/contract, i.e. a share-rent pair $(s_t, f_t) \in M_t$. Given the announced contract, the agent provides the optimal level of effort which in our model is equal to the share that was offered, i.e. $z_t^* = s_t$. Output for each within-period interaction, y_s, $s = 1, ..T_1$, is then given by:

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

where $\epsilon_{s,t}$ is normally distributed with zero mean and variance σ^2. The principal thus collects profit $\pi_{s,t} = y_{s,t}(1 - s_t) + f_t$.

We assume that the principal uses the same strategy over T_1 interactions. During this time, the agent provides the same optimal effort level, z_t^* while output and profits vary since they depend on the realization of the stochastic shock, ε_s. This is why we use subscripts s to denote output for each individual interaction. At the end of T_1 interactions, the value of the average output produced during time period t is:

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

The average profit/payoff for period t is thus:

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

This average payoff represents the measure of performance of a particular strategy (s_t, f_t). In case that the proposed strategy does not satisfy the agent's participation constraint, the average payoff, $\bar{\pi}_t$ is set equal to zero.

[11] We will use the terms *message*, *strategy* or *contract* interchangeably.

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

[13] The function of these within-period interactions between a principal-agent pair is to allow the principal time to learn about the expected profits that can be generated from a given contract. We provide comparative statics with respect to T_1 at the end of the paper.

3.2 Social Learning

In the social learning framework, learning operates on the population scale. There are N principal-agent pairs, i.e. the number of strategies in M_t at any point of time is equal to the number of principals or, in other words, each principal is represented by a single strategy. The learning process proceeds following the general form described above. Below we describe the specifics related to our implementation of social evolutionary learning (SEL) as representing the idea of learning from others. Learning takes place between periods.

The first element of SEL algorithm is *replication* which allows for potentially better paying alternatives to replace worse ones. It is used to generate a population of N replicates of the strategies that were used in the population at period t. As our baseline operator, we use proportionate ("roulette wheel") replication. Specifically, a strategy m_t^i , $i \in \{1, ..N\}$, in the current strategy set has the following probability of obtaining a replicate:

$$Pr_t^i = exp(\lambda \bar{\pi}_t^i) / \sum_{j=1}^{N} exp(\lambda \bar{\pi}_t^i) \tag{4}$$

where λ is a parameter governing the relative fitness weights. In addition to *simple* proportionate replication, we also consider a *selective* proportionate replication. Under the selective proportionate replication, a new strategy replaces a strategy that was previously implemented only if it yields a higher average payoff. If this is not the case, the principal keeps her previously implemented strategy. More formally, the payoff of each strategy m_{t+1}^j, $j = \{1, \ldots, N\}$ is compared to the payoff of its corresponding strategy m_t^j, $j = \{1, \ldots, N\}$, i.e. the j^{th} member of the strategy collection at time t. The strategy at location j that has a higher payoff between the two, becomes the member of the set M_{t+1} at $t + 1$.

Alternatively, as a robustness check we also implement another, commonly used replication operator, *tournament selection*, in the following way. For $i = 1, \ldots, N$, m_{t+1}^i is chosen by picking two members of M_t randomly (with equal probability) with replacement. Let these be m_t^k and m_t^i. Then let

$$m_{t+1}^i = \left\{ \begin{array}{c} m_t^k \\ m_t^l \end{array} \right\} \text{ if } \left\{ \begin{array}{l} \pi(m_t^k) \geq \pi(m_t^l) \\ \pi(m_t^k) < \pi(m_t^l) \end{array} \right\} \tag{5}$$

After replication, *experimentation* takes place, each strategy/message m_{t+1}^i is subjected to experimentation with probability μ. If experimentation takes place, the existing strategy, m_{t+1}^i is replaced by a new message from G which is drawn from a square centered on m_{t+1}^i with sides of length $2r_m$.[14]

The updating process above describes 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. Those that yield above-average payoffs tend to be used by more principals in the following period. The experimentation incorporates innovations by principals, done either on purpose or by chance.

3.3 Individual Learning

The individual learning paradigm is based on an individual's learning and updating of strategies based only on her *own* experience.[15] In our setting this implies that each period the principal has a collection

[14] We also conducted simulations with *selective experimentation* whereby the new strategy is implemented only if it has a higher payoff than the one it is replacing. This procedure, however, did not have a significant impact on the performance of our learning algorithms, and thus we do not discuss this variant any further.

[15] Unlike SEL, here we have in mind only one principal.

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 results in the increase in the frequency of well performing strategies in the principal's collection. The choice of a particular strategy as the actual strategy that the principal uses in a given period is probabilistic, and the strategies' selection probabilities depend positively on their past performance.

All three individual learning models that we study, RL, EWA and IEL, have these common features but also differ in important ways. The main differences among the three algorithms are related to how the pool of strategies is determined and how it is updated over time. These differences turn out to play an important role in the results that we obtain. Next, we provide a detailed description of each of the individual learning algorithms that we study.

3.3.1 Reinforcement Learning

We follow the implementation of Roth and Erev's (1995) adjusted reinforcement learning (RL) algorithm. In contrast to the SEL model, the strategy set, M, is the whole strategy space, i.e. the full grid $G = S \times R$, i.e. all possible combinations of s and f. The number of strategies in M_t for each t is thus equal to the number of grid points, i.e. for the RL model, $N = \#S \times \#R$. A single principal chooses one of the strategies from the set to play each period. Each strategy in M is assigned a *propensity of choice*.[16] The propensity of choice of a given strategy is updated in a given period t based on the payoff it earned if it was used at t, and is otherwise left at its previous level. In our implementation, propensities of choice are given by their discounted payoffs.

Specifically, for each strategy m^j in M, let I_t^j denote an indicator value for the principal's strategy in time period t, where $I_t^j = 1$ if m^j is chosen in period t and $I_t^j = 0$ otherwise. Then, the discounted payoff of strategy j, at time t, R_t^j is given by:

$$R_t^j = \left[q \cdot R_{t-1}^j\right] + \left[I_t^j \cdot \bar{\pi}_t^j\right] \tag{6}$$

where $q \in [0, 1]$ is a time/memory discount factor parameter and $\bar{\pi}_t^j$ is the average payoff computed over T_1 interactions. At the beginning, each strategy's payoff, R_1^j is set equal to 0.

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

$$Pr_{t+1}^j = \frac{e^{\lambda R_t^j}}{\sum_{k=1}^{N} e^{\lambda R_t^k}}. \tag{7}$$

Once a strategy is selected, it undergoes experimentation with probability μ. In case that experimentation takes place, rather than announcing the initially selected strategy, e.g. $\tilde{m}$ to the agent, the principal announces a randomly drawn strategy from the square centered on $\tilde{m}$ with sides of length $2r_m$. As in the SEL model, the chosen strategy is then implemented for T_1 interactions during which the agent responds with the (same) optimal effort.

3.3.2 Experience Weighted Attraction Learning

Our second individual learning algorithm, EWA, is a generalization of the RL algorithm described above. We follow Camerer and Ho (1999) to describe the version of EWA that we implement here. The strategy space, M of size $N = \#S \times \#R$ is the same as that under RL, namely the complete grid G.

[16] In the initial period all strategies in G have equal probability of being selected.

In EWA, a strategy that was actually used, denoted by m_t^a, receives an actual evaluation of its performance, while all other strategies in M receive evaluation of their *foregone* (or hypothetical) performance. The hypothetical payoff for a strategy $m^j \in M$, $m^j \neq m_t^a$ in period t is:

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

where $\bar{y}_t(s_t^a)$ is the average output generated under strategy m_t^a in period t. In the performance evaluation process (see below) this foregone payoff is weighted by a discount factor $\delta \in (0, 1)$ 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 main variables that are updated after each round of experience: N_t, the number of "observation-equivalents" of past experience (called the *experience weight*); and A_t^j, the attraction of strategy m^j at the end of period t. Their initial values N_0^j and A_0^j can be interpreted as prior game experience and/or principal's predictions.

The experience weight, N_t, is updated according to

$$N_t = \rho N_{t-1} + 1 \tag{8}$$

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

$$A_t^j = \frac{\phi N_{t-1} A_{t-1}^j + [\delta + (1 - \delta) I_t^j]\bar{\pi}_t^j}{N_t}.$$

The parameter δ determines the extent to which hypothetical evaluations are used in computing attractions. If $\delta = 0$, then no hypotheticals are used, just as in the RL model. If $\delta = 1$, hypothetical evaluations are fully weighted. The parameter ϕ is another discount factor or decay rate, which depreciates the previous attraction and is similar to the parameter q in the RL model. In fact, if $\phi = q, \delta = 0, \rho = 0$, and $N_0 = 0$ then the EWA model is equivalent to the RL model. Finally, just as in the RL model, at $t + 1$ the principal selects each strategy $m^j \in M$ with probability:

$$Pr_{t+1}^j = \frac{e^{\lambda A_t^j}}{\sum_{k=1}^{N} e^{\lambda A_t^k}} \tag{9}$$

3.3.3 Individual Evolutionary Learning

Our third individual learning algorithm, IEL, shares some common features with RL and EWA. First, like in both the RL and EWA, the choice of the principal's strategy is probabilistic. Also, the selection probabilities are based on a strategy's 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 grid G but instead changes over time (as in SEL) in response to experience and occasionally, to pure random events (experimentation).

Specifically, as in the SEL algorithm, at time $t = 1$, a set of $N \geq 1$ strategies, M_1 is chosen from the grid G randomly. In contrast to SEL, these strategies do not pertain to N different principals but instead become the single principal's strategy collection at time $t = 1$. Over time, the principal always keeps N active strategies. Suppose that at the beginning of round t, the principal' 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, that is, it is implemented over T_1 interactions.

Similar to EWA learning, the payoffs of all other (inactive) strategies in the set M_t (but not in G) 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 a strategy $m_t^j \neq m_t^a$ in period t is:

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

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 (as in SEL). Replication reinforces strategies based on their actual and hypothetical payoffs and generates a population of N replicates of the strategies that were used at time period t. Again, as our baseline operator, we use proportionate ("roulette wheel") replication. Thus, each strategy has the following probability of obtaining a replicate:

$$Pr_t^j = exp(\lambda\bar{\pi}_t^j) / \sum_{j=1}^{N} exp(\lambda\bar{\pi}_t^j) \tag{10}$$

Again, for robustness check, we implement *selective* replication and tournament selection as well.

After replication, experimentation takes place. It works in the same way as in the SEL model. The application of replication and experimentation results in the next period's strategy set M_{t+1}. Finally, as in SEL, given M_{t+1}, the selection probabilities of the strategies in M_{t+1} are computed (using again equation 10).

3.3.4 Modified IEL Model with Realized Payoffs

Finally, we describe a modified model of individual evolutionary learning that we decided to study in light of the unsatisfactory convergence performance (see section 5) of the canonical IEL algorithm with foregone payoffs. The modified model differs from the standard IEL described above in that only the payoffs of those strategies that were actually played are updated. In this respect, the modified algorithm is the same as reinforcement learning. 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, i.e. we use replication to change the frequency with which different strategies are represented in the collection, and experimentation to draw 'new' strategies from our strategy space. Overall, the IELR model is thus a hybrid between the RL and the standard IEL model.

4 Computational Implementation of the Learning Algorithms

This section describes the computational procedures we followed to initialize and implement the learning algorithms in our setting. The next section contains the simulation results obtained from a wide set of parametrizations and numerous robustness checks.[17]

In order to obtain representative results we perform 7,350 different runs for each learning regime. These runs differ in the parameter values for γ and σ from the structural model and the random generator seed used to draw the initial pool of strategies, i.e. each run corresponds to a unique combination $(\gamma, \sigma, seed)$. The values for γ and σ that we use are exhibited in table 1 below. The agent's reservation utility is set to $\bar{u} = 0$.

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

The *strategy space,* G , from which strategies are chosen is composed of all (s_t, f_t) pairs belonging to the two-dimensional grid such that s_t belongs to an equally spaced linear grid on the interval $[0, 1]$ and f_t belongs to an equally spaced linear grid on the interval $[f_{\min}, f_{\max}] = [-0.05, 0.5]$. The strategy space dimensions 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 neighboring points.

In the SEL and IEL models, N messages 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.[18]

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]
strategy pool size (SEL/IEL), N	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[19]	1

In the results section next we also perform numerous robustness and comparative statics runs varying the above parameters.

5 Results

5.1 Benchmark Runs

We begin by reporting the results from our benchmark social and individual learning runs. In order to present our results, we define and examine the behavior of a number of different measures that reflect both qualitative and quantitative aspects of the learning dynamics. These measures include:

- 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

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

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

- the time paths of the fraction of simulated strategies/payoffs within a given distance from the optimum. Here, each point on the time path equals the average fraction over all of the strategies of the 7,350 runs. Two "distance" criteria are considered: 0 and 0.05
- strategy time paths generated by the different learning models for a sample run.

Table 2 characterizes the performance of our four benchmark learning algorithms. The table contains two alternative measures of performance, namely the percentages of last period ($t = 2,400$) strategies in the strategy pool and their payoffs (averaged over all 7,350 runs) that are within a given "distance" from the optimal contract, or its payoff (which is computed in percentage terms). The distance to the optimal payoff (rather than that in strategy space) is more relevant from an economic point of view given that the objective of the principal is to maximize profits rather than to get as close to the optimal contract as possible (although these two measures are clearly related).

It is evident that our SEL algorithm with baseline replication has a hard time learning what the optimal strategy is. With the 'exact convergence to the optimum criterion' is used, only 1.65% of all last period messages in the pool coincide with the optimal contract (s^*, f^*). If the performance criterion is relaxed to allow for 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 messages in the final pool over all runs ending within 10% of the optimal payoff.

However, our results indicate that our benchmark individual learning algorithms have no ability 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. The three individual learning algorithms show little improvement even under this relaxed performance criterion. Worth noting is a poor performance of the standard IEL algorithm that otherwise performs much better than RL and EWA in the environments with large strategy space.

Figures 1 and 2 complement table 2 by visualizing the algorithms' performance. In figure 1, which displays the histograms of the differences between simulated and optimal payoffs and strategies, we see that only the SEL model sometimes gets anywhere close to the optimal strategy and payoff. Figure 2 shows the time paths of the actually realized share, s, and fee, f, for a given sample run under each of the four learning regimes. The figures clearly illustrate the difficulties with convergence to the optimal contract in our principal-agent setting.

Discussion

There are three important factors that are responsible for the poor performance of our benchmark learning algorithms. First, and common to all of the algorithms, is the fact that in our problem, the payoffs are a discontinuous function of the message space at the participation constraint. Figure 3 which is drawn for a sample parameter configuration illustrates this point clearly. All strategies above the participation constraint, which is given by the parabola-shaped solid 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 if this constraint is violated. On the other hand, the optimal contract (s^*, f^*), denoted by a black diamond on the figure, lies *on* the participation constraint. Thus, small deviations away from the optimal contract that enter the zero-payoff area above the parabola lead to a large discontinuous drop in payoffs. This affects the performance of all the algorithms and slows or even prevents 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 quite steeply away from the participation constraint while they stay quite high near the constraint even for those (s, f) contracts 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, while even the smallest amount of experimentation can take a strategy "off the cliff", to the right, into the area of much lower payoffs, and to the left, into the area of zero payoffs.

A related issue with the strategy space affects the RL and EWA updating methods where all the points in the strategy grid G belong to the principal's strategy pool. In our benchmark simulations this number of points is quite high (over 5,000) which contributes additionally to the poor performance of the two algorithms. A decrease in the grid density improves their performance somewhat (see section 5.3).

A second, very important factor causes the poor performance of the IEL and EWA algorithms, namely the fact that they both rely on hypothetical (foregone) payoff evaluation as explained in section 3.3. That is, strategies in the pool that have not been played receive payoff updates together with the actual contract offered to the agent. The problem here is that the hypothetical payoffs are computed as the foregone profits that the principal would have obtained if they had played some alternative strategy $(\tilde{s}, \tilde{f})$. However, the effort, through the *currently* observed output realization, $\bar{y}$, that is used in this computation is the effort that is the optimal response for the contract that was actually played. This is the key to understanding the failure of the EWA and IEL algorithms based on the foregone payoff evaluation in our setting. The reason is that, if in fact the agent is offered a different $\tilde{s}$, she will change her behavior, and expected output will not be the original $\bar{y}$ anymore. Thus, all assigned hypothetical payoffs are incorrect which dooms any algorithm using the foregone payoffs' approach. Note that this is a general point that would apply to any economic model where the underlying game is sequential (Stackelberg) i.e. where one party moves first and then the other party reacts. In contrast, in simultaneous (Nash) games, evaluating hypothetical payoffs in the way assumed by the IEL algorithm is not subject to this problem since the equilibrium is defined by finding the best response *given* (i.e. holding fixed) the other party's choice.

The problem with the hypothetical payoffs is illustrated in the following IEL example. Suppose $\sigma^2 = 0$ and that the principal offers the contract (s_t, f_t) with both components strictly positive. Then she receives the signal $y_t = s_t > 0$. Among all the strategies in the current pool, the one that makes principal's profits, π_t, largest while satisfying the participation constraint is assigned the highest payoff. Let us look closer at what this strategy will 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, so π_t is a quadratic function with maximum achieved at a corner solution, i.e. $s = 0$ or $s = 1$. In particular, if $y_t > 1/2$, the strategy in the pool 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 $s_{t+1} < 1/2$, which is likely given the previous period's replication, then at $t + 1$ the strategies closer to (1, 0.5) are now favored.[20] 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.

A final factor explaining our benchmark results and common to all models of social and individual learning, is that our model features learning in a stochastic environment. It is well-known from experiments with various learning models[21] that convergence to equilibria in stochastic settings is often

[20] Note that this discussion assumes that the principal is able to learn the participation constraint.

[21] 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

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" does help improve convergence to the optimum, confirming this intuition.

5.2 Simulations with Modified Evolutionary Algorithms

We try to reduce negative effects of replication on our SEL and IEL performance by implementing its modified version: *selective* replication (described in section 3.) As mentioned earlier, 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 our benchmark IEL algorithm in order to deal with the problem related to the hypothetical payoff evaluation by adopting a method for evaluating strategies that is more similar to RL, i.e. where only strategies that were actually selected for play have their payoffs evaluated.[22] We call this model *IEL with realized payoffs* (IELR).

Note that the same modification of EWA would reduce it to a version of the RL algorithm whose performance has already been evaluated. 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 improves dramatically the performance of the SEL algorithm - now 73.5% (versus only 1.6% in the benchmark) of the strategies in the final pool over all 7,350 simulations coincide exactly with the optimal contract and virtually 100% of them come to within 5% of the maximum possible profit compared to only 37% in the baseline replication benchmark. We show 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 that a much larger fraction of the differences is now close to zero (compare with figure 1). Similar improvement can be seen in the time paths of fractions of strategies equal to or within 5% of the optimal strategies shown in 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 period 300.

Now let us look at the performance of the IELR algorithm. Table 3 as well as figures 4 and 5 report the improvement in performance under both baseline and selective replication relative to the baseline IEL model with hypothetical payoffs. We see that significant gains in performance result from both modifications (selective replication and evaluation of realized payoffs only), although ultimately the IELR's performance still remains worse than that of the SEL algorithm. Specifically, the IELR has respectively 30% and 59% of final pool strategies within 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 are fairly good results in terms of performance. Notice that selective replication alone

state' bias of the decision rules that are updated with an algorithm that combines elements of reinforcement learning and replicator dynamics.

[22] We keep the rest of the updating process (i.e. replication and experimentation) as in the standard IEL since it has proven successful with dealing with the large strategy space.

does not help the performance of the algorithm (line 2 in the table). These results are also illustrated in figure 6 where we display the strategy time paths for a sample run[23].

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 the IEL case as new mutants there (entering with a payoff of zero) can survive quite long in the pool without being played and hence without being updated. Once the experimentation rate decays to zero these strategies disappear from the 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 SEL algorithm (especially when allowing for a more sophisticated replication operator) are better able to learn the optimal contract in our environment. Individual evolutionary learning with realized payoff evaluation shows promise but still performs significantly worse than social evolutionary learning. One candidate reason for this under-performance seems the fact that, under SEL, all strategies in the strategy pool are evaluated in each period with their actual payoffs. In contrast, IELR updates only one strategy at a time (the actual contract offered to the agent) which seems to put the individual learning algorithm at disadvantage. In section 6 we provide more details and a formal analysis on this non-convergence issue showing that the SEL algorithm still performs better even controlling for total the number of strategies evaluated.

5.3 Robustness Runs

In this section we report the results from various additional simulations 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; varying the grid spacing, d for EWA and RL; using tournament selection in the replication process; and varying the timing of experimentation decay[24] All robustness runs were performed for the same set of 7,350 parametrizations as before. The results are displayed in table 4. Also, most of the robustness runs we report apply to the individual learning algorithm since SEL performs very well already in the benchmark once selective replication is allowed. The main findings are as follows:

1. Varying the strategy pool size, N

We find that increasing the pool 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 strategy pool, learning occurs faster as more strategies can be evaluated each period. The results for IEL with realized payoffs are quite different, however. Both increasing N to 100 and decreasing it to 10 generate slight drop-off in performance from the $N = 30$ benchmark with the fraction of strategies achieving the optimum going down from 32% to 27-28%. The reason for the difference with SEL results is that, with realized payoffs, the change in N does not affect the number of strategies' payoffs that are updated each period. On one hand, lower N can be potentially beneficial given that smaller number of strategies will have zero payoffs in the pool but on the other hand, it has a disadvantage of not allowing for enough diversity in the pool which is

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

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

especially important in the early stages of the learning process. In general, there will be some optimal pool size that maximizes the algorithm 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 SEL and IEL algorithms. In the SEL case the percentage of strategies achieving the optimal contract rises from 73.5 to 88.8, while in the IELR case the corresponding increase is even more significant (from 32% to 51.8%). Notice that the IELR with $T_1 = 100$ comes within 10% of the optimal payoff in 99% of all cases. (Increasing T_1 to 100, improves performance of IELR with $N = 10$ compared to the case with $N = 10$, and $T_1 = 10$.)

As expected from the pool size results, an increase in the length of the evaluation window, T_1 combined with a decrease in N to 10 in the IEL case 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 decay timing

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 from 5% to 2%, motivated by the idea that this will decrease the number of new strategies with zero payoffs in the strategy pool, leads to insignificant changes in the performance of the algorithm. We also experimented with reducing the value of the experimentation parameter, r_m. The motivation behind this exercise is that after the initial adjustment, the algorithm leads to a strategy pool 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 strategies that result in the optimal payoff, but a smaller increase (about 1%) for strategies within 5% of the optimal payoff. Finally, we studied the effect of moving forward the time period when experimentation starts decaying from $t = 2,000$ (which is our benchmark case) to $t = 500$. The effect is reduction in performance of about 25-30% relative to the benchmark values from table 3 as the principal has less time to learn about 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 algorithm 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. The reason for superior performance of selective replication is the following. 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. Other robustness checks - grid spacing, fitness weights

We also re-did the 7,350 runs under RL and EWA learning, increasing the grid spacing parameter, d from 0.01 to 0.1. This increase makes the grid on the strategy space G coarser. A coarser grid helps

the RL and EWA algorithms achieve better performance than in the benchmark but they are still far from being successful in converging to the optimal contract. Once again, the EWA algorithm performs worse than RL due to the problem with hypothetical payoffs' evaluation discussed above.

Finally, we experimented with increasing the parameter λ which governs the curvature in the mapping between average profits realized by the principals who are learning and the fitness of the strategies they are evaluating. The results from increasing λ from 1 to 3 show a slight deterioration in performance (e.g. the percentage of last period strategies coinciding with the optimum declines from 32% to 27%). Using the biased roulette wheel replication from (4), 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. The combination of these two effects explains the observed outcome.

5.4 Convergence Analysis

We report the results of our convergence analysis of the best performing social and individual learning models, i.e. 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 "hits" the optimum, i.e. 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 criterion. Once the learning process takes the strategy pool close to the optimum, it stays there forever (e.g. see the sample runs in figure 6). In contrast, as discussed in detail in section 6, the IELR algorithm does not exhibit similar behavior prior to a sufficient decrease in the rate of experimentation (i.e. holds only after period 2,000). That is, in the IELR simulations, we can often have instances when the optimal strategy shows up in the agent's strategy pool at some period only to be wiped out shortly after by an experimentation or replaced with another strategy with a "lucky" output draw (especially likely if T_1 is small). An example is presented in figure 7 where we show the fractions of strategies within a given distance, 0, 0.05, and 0.1 from the optimal strategy for a sample IELR run. In the panel that shows the fraction of strategies equal to the optimal, we can observe that, between periods 300 and 400, there is a substantial fraction (around 40%) of the strategies that are equal to the optimal one. Again, between periods 1,200 and 1,400, a substantial fraction of the current strategies in the pool (around 30% on average) coincide with the optimal strategy. However, shortly afterwards, these strategies disappear from the pool and the fraction of optimal strategies 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 at the optimum' criterion, SEL is three times faster, and up to twenty times faster according to our 'within 10% of the optimal payoff' criterion. The percentage of simulations under IELR 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.

We also analyze the convergence rates as a function of the parameters γ and σ from our underlying economic model. This is shown in figure 8, top panel for the SEL algorithm and bottom panel for the IELR algorithm. The figure depicts the fraction (out of the 70 random seed runs for fixed model parameters) of non-convergent (according to the above-defined criterion) simulations 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 it 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 γ makes convergence in payoffs relatively harder.

6 Discussion on Non-Convergence

The following discussion explains in more detail the reasons for the non-convergence of the IEL algorithm in a large fraction of runs observed in the results for both the benchmark and the modified algorithms. Our finding that the standard IEL model with foregone payoffs virtually never converges to the optimal strategy may seem surprising as this algorithm has been previously shown to converge fast in various 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. This way, the algorithm has the ability to make bad strategies disappear from the pool. Foregone payoffs also help in evaluating quickly strategies that are brought in via experimentation - only those that appear promising in terms of foregone payoffs are kept and replicated.

However, our principal-agent problem is different from the environments in which IEL has been studied so far. As we already pointed out, the main problem comes from the fact that in order to evaluate a foregone payoff of a strategy that was not actually used, the principal assumes that the observed agent's action will remain constant for different strategies played by the principal. However, different principal's strategies in fact result in different agent's optimal actions. Thus the whole collection of strategies is evaluated given the agent's action as a constant. Overall, principals who use this type of algorithm are not able to learn what the optimal contract is.

In addition, even with the inclusion of selective replication and "realized payoffs" evaluation of strategies, the IEL algorithm still experiences difficulties in learning the optimal contract compared to the SEL model. The main reasons are as follows. First, strategies that are generated randomly in the initial period are assigned zero payoffs. In addition, any new strategy generated during simulation via experimentation is also assigned zero payoff. 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 to be replaced by a strategy with a positive payoff, they may remain idle in the collection for a long time.

Direct observations of the actual learning process for numerous 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 participation constraint. When most of the strategies are on the "right" side of the constraint, the algorithm displays improvement in performance in terms of strategy payoffs and closeness to the optimal strategy. This is displayed in figures 9 and 10 which illustrate the evolution of the strategy pool, M_t, for the best performing versions of the two models, namely SEL with selective replication (figure 9) and IELR with selective replication (figure 10). The figures show the current strategies in the pool at various time periods in the strategy (s, f) space, as well as the participation constraint, for our sample run. Empty circles denote strategies with zero payoffs (that have not been played or that violate the participation constraint), filled circles denote strategies with positive payoffs, diamond denotes the currently played strategy, and the star denotes the optimal strategy. We see that the better performance of the social learning model is due to the fact that it weeds out the strategies that violate the participation constraint quite fast and then converges quickly and stays close to the optimum. In contrast, the IELR algorithm exhibits 'cycles', i.e. gets close to the optimum, but then the pool spreads out again. It is only once experimentation decays sufficiently that IELR converges to the currently

best strategy which 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 will converge to the optimal contract in any fixed number of periods. To see this more clearly, suppose that the optimal strategy was in the pool at some period. One way through which it can be replaced by some different strategy is via experimentation. A second possibility is that it can be replaced by some suboptimal strategy. This can take place in two ways: either (1) a suboptimal strategy was actually played and had higher payoff than the optimal strategy that was never played; or (2) because a suboptimal strategy was "lucky" and got a series of favorable random shocks to y which resulted in a payoff higher than the payoff that the optimal strategy earned last time it was played. In both of these cases, if the number of other replicates of the optimal strategy in the collection is small or zero, 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, r_m (scope) of the experimentation (or its rate, μ) is clearly diminishing to zero.

In general, given that under IELR not all strategies in the pool are evaluated each period (in fact only one is), previously 'lucky' strategies (i.e. 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. Furthermore, 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 would also contribute to a slow down or lack of convergence to the optimum.

The above discussion suggests that the SEL algorithm can focus faster on a smaller number of strategies than the IELR algorithm. This point is further illustrated in figure 11 which reports the frequency distribution of the strategies that were 'active' (i.e. evaluated and played) during a given simulation. The frequency of each strategy in G is computed by recording the number of times this strategy was 'active' and dividing it by the total number of evaluations that take place over the course of a given simulation. The upper panel of figure 11 shows the frequencies for the SEL algorithm, and the lower panel shows the frequencies for the IELR algorithm. Comparing the two 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 frequency distribution where a larger number of more dispersed strategies (including many on the "wrong" side of the participation constraint) were played and evaluated, with generally lower frequencies.

Does the SEL algorithm only perform better because N strategies (although not necessarily all different) are evaluated per period while only a single strategy is evaluated in IELR in each time period? In order to put IELR on an equal footing with SEL in terms of the number of strategies that have been evaluated, we compare the make-up of the strategy pools at time periods where the IELR number of strategy evaluations equals that of the SEL number of evaluations[25]. Figure 12 presents the result of this comparison. In order to obtain equal number of strategy evaluations, we allow 30 (=N) times as many periods for the IELR algorithm. Thus, the panels in the first row of figure 12 show the strategy pools at $t = 200$ for SEL, and at $t = 6,000$ for IELR. Similarly, the second row of panels exhibits the strategy pools for $t = 500$ for SEL, and $t = 15,000$ for IELR. Finally, the third row of panels shows the evolution of the fractions of strategies equal to the optimum and within .05 away from

[25] Actually, since the strategy pool under SEL will typically contain replicates of the same strategy later in the simulation, this exercise actually gives an advantage to the IELR model in terms of total number of evaluations.

the optimum for $t = 500$ for SEL, and $t = 15,000$ for IELR. It is clear that, even with equal number of strategy evaluations, the IELR strategy pool remains much more dispersed. In addition, looking at the evolution of the converging strategy fractions under IELR, we can see that, after some initial progress, these fractions stabilize around 0.1 for those strategies equal to the optimum, and around 0.25 for those strategies within 0.05 away from the optimum and stay at these levels until the end of a simulation (period 15,000). This result confirms our conclusion that the failure of the individual learning model is mainly due to the host of reasons described above and not to the different number of evaluations performed.

Finally, a feature of our economic model, or any principal-agent model for that matter, that might cause difficulties for the algorithm's smooth convergence is that strategies that are close to the participation constraint, but in terms of distance in (s, f) space further away from the optimal contract can have higher payoffs than strategies that are very close to the optimal strategy in strategy space but slightly off the participation constraint (see figure 3 which displays the theoretical iso-profit lines and the top ten strategies in terms of payoffs). This factor, taken together with the impact of replication, can also prolong the learning time or contribute to the failure of a particular simulation to converge to the optimal value.

In summary, all of the above effects make the task of learning the optimal strategy using the individual learning paradigm very difficult as demonstrated in our simulations.

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. The learning models that we look at 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 IEL with realized payoffs (IELR).

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

In order to try to overcome these difficulties with convergence to the optimal contract, we introduce modifications to our baseline IEL that improve its performance. Specifically, we modify the algorithm so that only the payoffs of those strategies that are actually tried out are updated. This updating of the payoffs is thus similar to the way updating is implemented in reinforcement learning. However, we keep important elements of the IEL updating, namely its replication and experimentation operators, that enable IEL to handle environments with large strategy space well. The resulting modified algorithm (IEL with realized payoffs) turns out to be much more successful than all other individual learning models that we studied. The implementation of selective replication further improves IELR's performance in terms of convergence to the optimal contract. Still, 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 operationalize a famous quote from Marshall's (1920) "Principles of Economics" related to the informal exchange of ideas and knowledge and diffusion of knowledge: "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".

How could we test this empirically? As a first step, one could 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) learn from their own experience only. In the second environment, subjects will be given a chance to interact with other 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, for example using the methods developed in Karaivanov and Townsend (2007). IEL agents learn from their own experience and 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 do on their own. Thus, how many other agents one is in contact with (e.g. data on social networks, friends, neighbors, etc.) should affect performance in addition to their own experience - both the strength of the relationships an agent has with others and their number should influence her outcome under SEL but not under IEL. Finally, we are also working on extending and applying the basic framework and methods developed here to other contractual settings such as adverse selection environments or the increasingly popular dynamic models of asymmetric information (e.g. as those used in the optimal taxation literature).

References

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

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

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

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

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

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

[7] 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.

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

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

[10] Caplin A. and J. Leahy (1994), "Business As Usual, Market Crashes and Wisdom After the Fact", *American Economic Review* 84(3), pp. 548–65.

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

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

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

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

[15] 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.

[16] 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.

[17] 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.

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

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

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

[21] Karaivanov, A. and R. Townsend (2007), "Enterprise Dynamics and Finance: Distinguishing Information Regimes", mimeo, University of Chicago.

[22] 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.

[23] LeBaron, B., W. Arthur and R. Palmer (1999), "Time Series Properties of an Artificial Stock Market", *Journal of Economic Dynamics and Control* 23, 1487-1516.

[24] 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.

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

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

[27] Marimon, R., E. McGrattan and T. Sargent (1990), "Money As a Medium of Exchange in an Economy with Artificially Intelligent Agents", *Journal of Economic Dynamics and Control*, 14(2), pp. 329-73.

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

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

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

[31] 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.

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

[33] 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.

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

[35] 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.

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

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

[38] 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.

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

[40] 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.

[41] 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

Model	% Last Period Strategies within x of the optimal contract			% Last Period Payoffs within x% of the optimal payoff		
	x=0	x=0.05	x=0.1	x=0	x=5	x=10
Social Learning	1.65	18.59	41.51	1.65	37.43	67.81
Individual Evolutionary Learning	0.00	0.00	0.00	0.00	0.00	0.00
Reinforcement (d = 0.01)	0.03	1.96	6.27	0.03	1.67	3.73
EWA (d = 0.01)	0.00	0.79	3.99	0.00	0.52	1.17

Table 3: Algorithm Performance - Modified

Model	% Last Period Strategies within x of the optimal contract			% 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 Social (SLR)	73.50	92.15	99.70	73.50	99.74	100.00
Modified IEL (SLR, hypothetical)	0.00	0.00	0.00	0.00	0.00	0.00
IEL with Realized Payoffs						
Individual (BR)	1.20	14.22	33.18	1.20	30.61	59.36
Individual (SLR)	32.03	62.19	82.12	32.03	80.96	91.80

Notes: BR - baseline replication; SLR - selective replication

The EWA parameters used are: δ=0.2; ρ=0.8; φ=0.8

Table 4: Algorithm Performance - Robustness Checks

Model	% Last Period Strategies within x of the optimal contract			% Last Period Payoffs within x% of the optimal contract		
	x=0	x=0.05	x=0.1	x=0	x=5	x=10
Social (SLR, N=100)	89.70	98.94	100.00	89.70	100.00	100.00
Social (SLR, T_1=100)	88.75	98.71	100.00	88.75	100.00	100.00
Social (SLR, early exper. decay)	48.57	64.04	84.60	48.57	97.54	99.77
Social (tournament)	68.90	90.12	99.13	68.90	99.25	99.99
Individual (SLR, N=100)	27.66	57.84	78.40	27.66	74.94	84.80
Individual (SLR, N=10)	26.88	56.97	78.69	26.88	77.63	89.06
Individual (SLR, T_1=100)	51.77	78.72	94.95	51.77	94.91	98.97
Individual (SLR, N=10, T_1=100)	42.22	71.77	91.44	42.22	92.48	98.34
Individual (SLR, exper. rate=0.02)	32.61	63.14	83.94	32.61	82.56	92.44
Individual (SLR, pert. radius decay)	37.26	67.28	88.18	37.26	85.17	94.50
Individual (SLR, early exper. decay)	18.94	44.93	67.54	18.94	77.99	90.95
Individual (tournament)	2.59	13.03	26.52	2.59	41.76	62.07
Individual (SLR, λ=3)	27.09	59.96	81.48	27.09	78.92	88.97
Reinforcement (d = 0.1)	15.03	15.03	25.59	15.03	21.16	30.33
EWA (d = 0.1)	1.31	1.31	4.63	1.31	3.06	3.59

Notes: BR - baseline replication; SLR - selective replication

EWA parameters used are: δ=0.2; ρ=0.8; φ=0.8

All individual runs use realized payoffs

Table 5: Convergence Analysis

Model	percent of simulations within x of optimum (90%, 90%)			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
Social (SLR)						
-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
Individual (SLR, realized)						
-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

Notes: SLR - selective replication
Max number of runs = 2400
(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

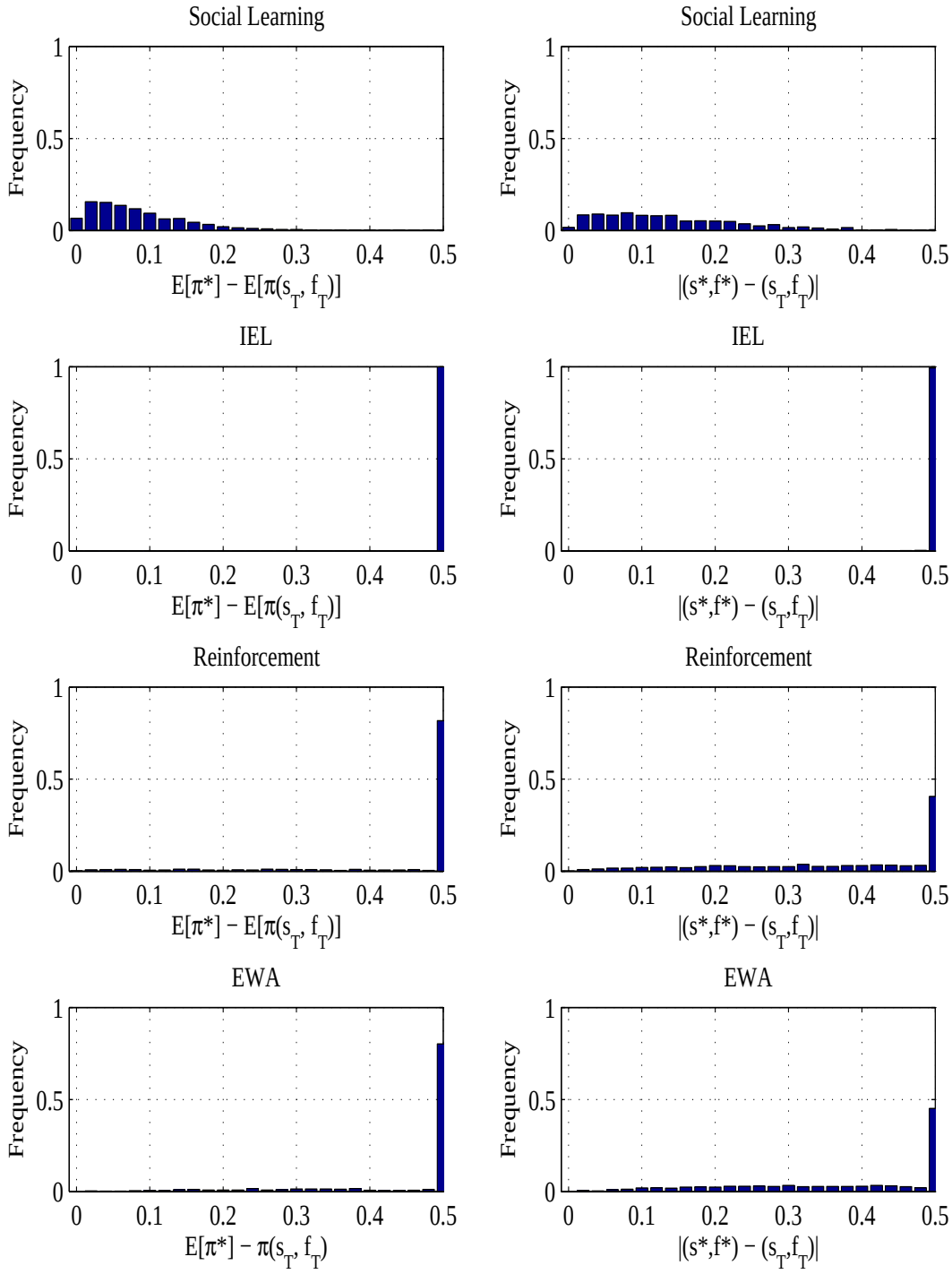

Figure 1: Differences between Simulated and Optimal Payoffs (first column) and Strategies (second column)

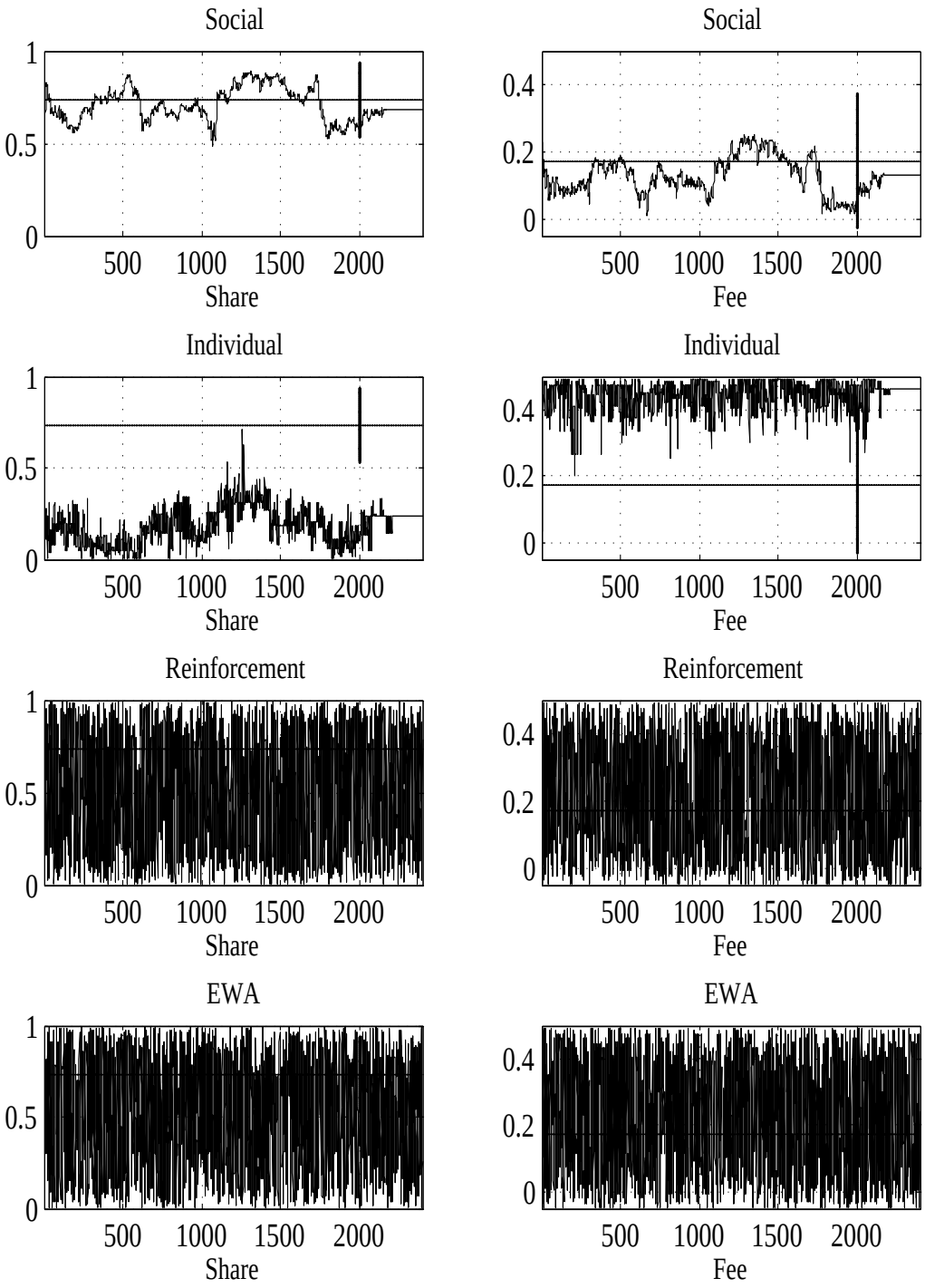

Figure 2: Message Time Paths for a Sample Run - Benchmark Case

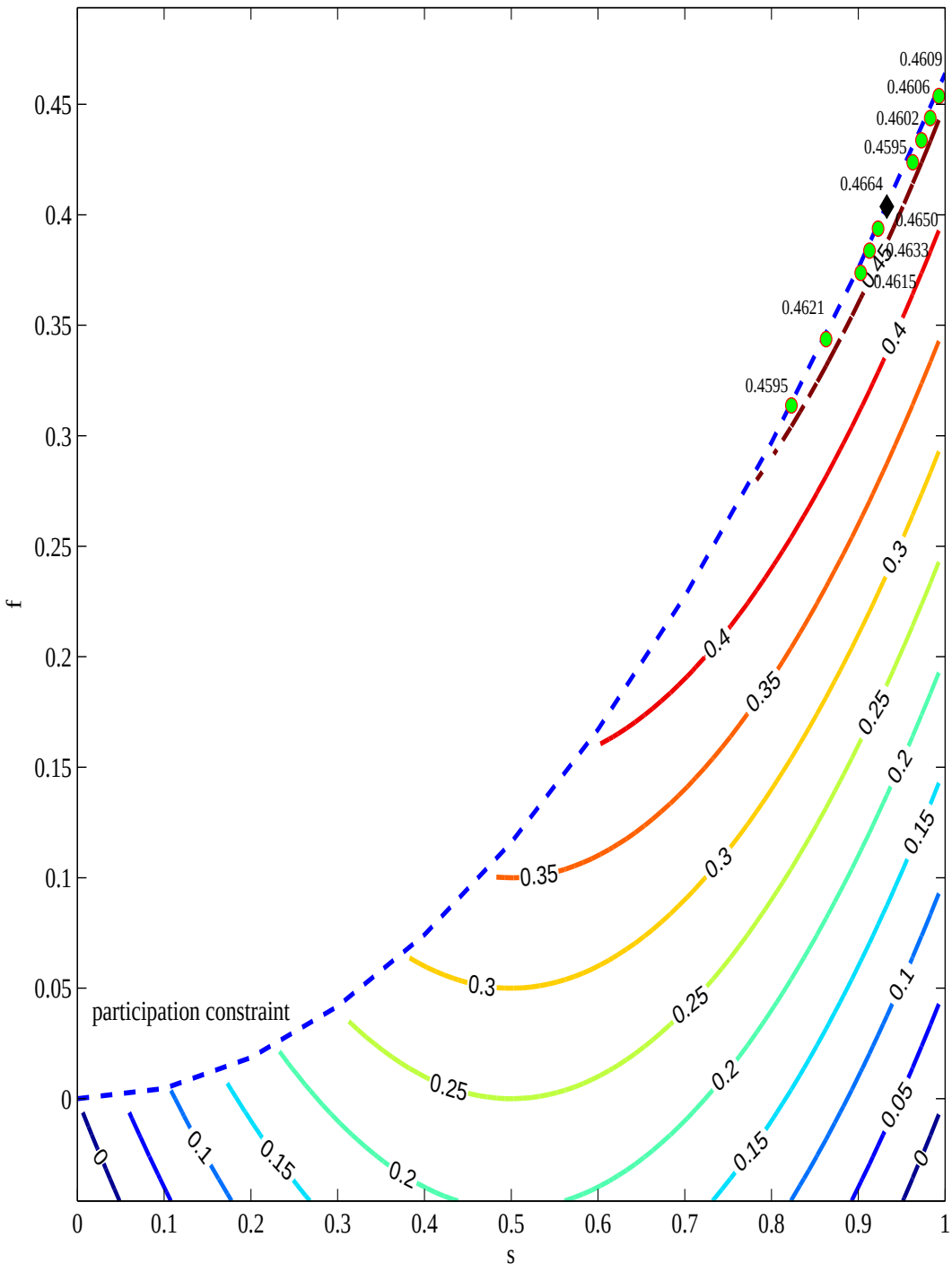

Figure 3: Iso-payoff Lines for a Typical Case

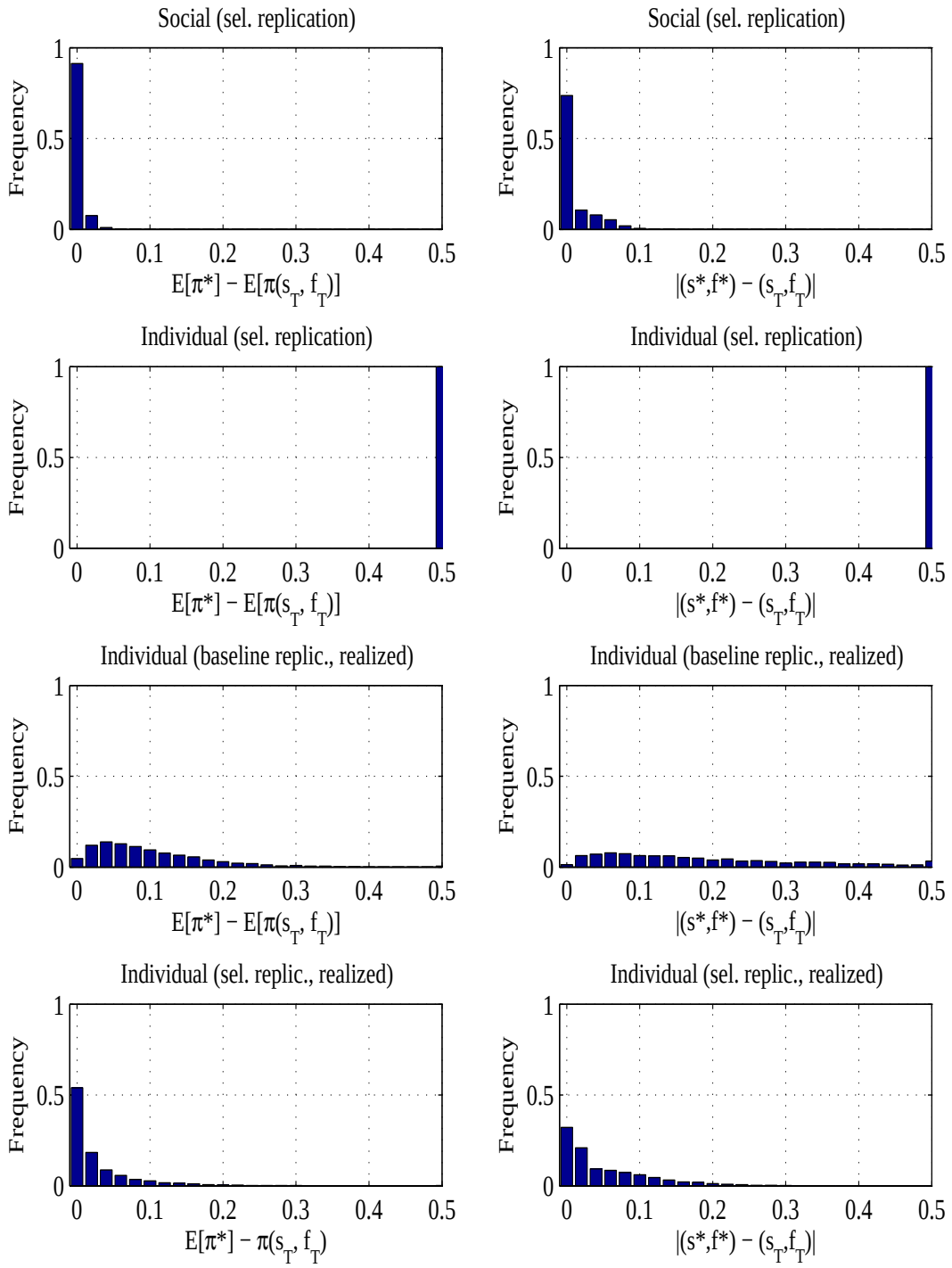

Figure 4: Differences between Simulated and Optimal Payoffs and Strategies, Last Period

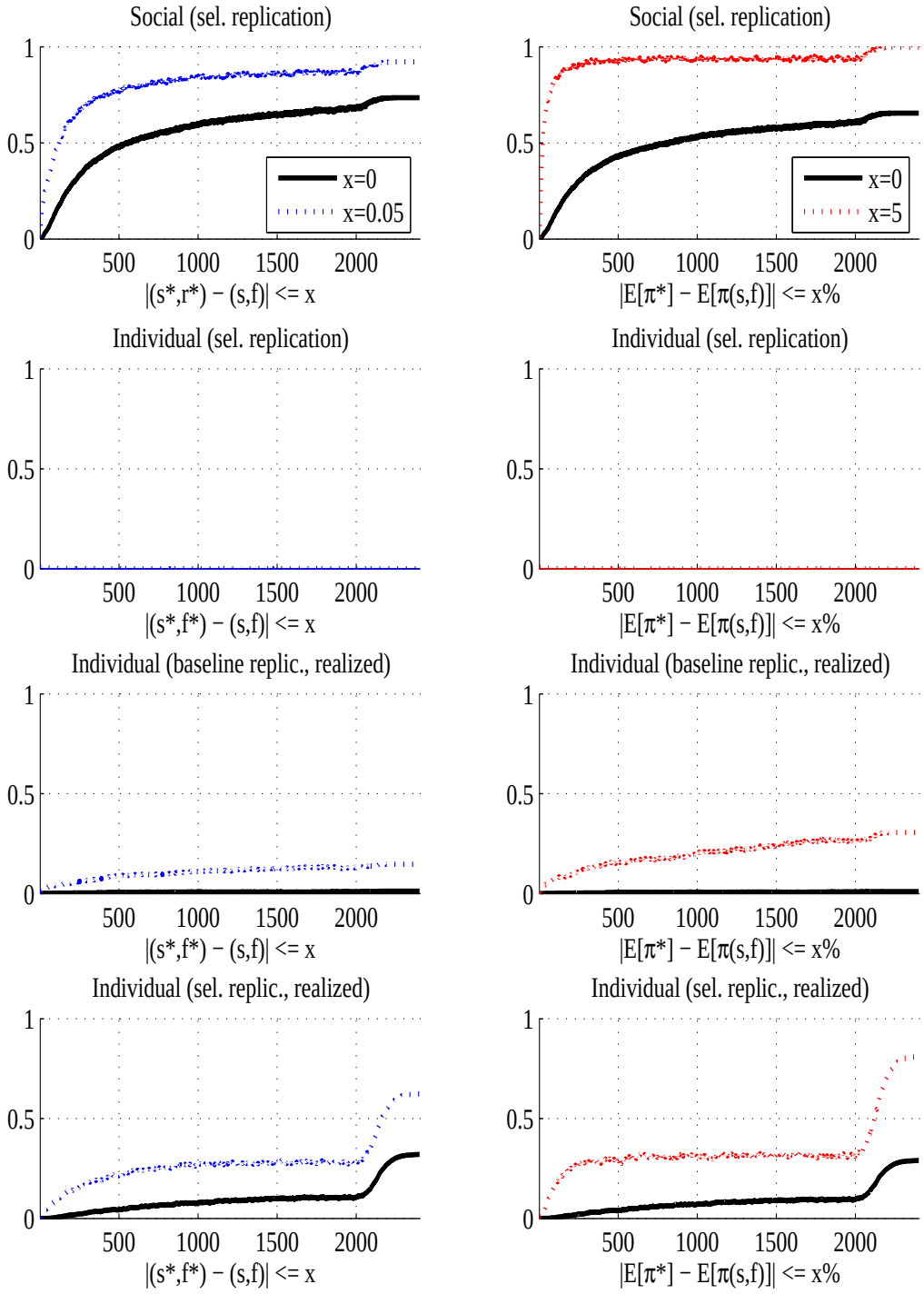

Figure 5: Time Paths of the Fraction of Simulated Strategies/Payoffs Equal to the Optimum (solid Line) and 5% of the Optimum (dotted Line)

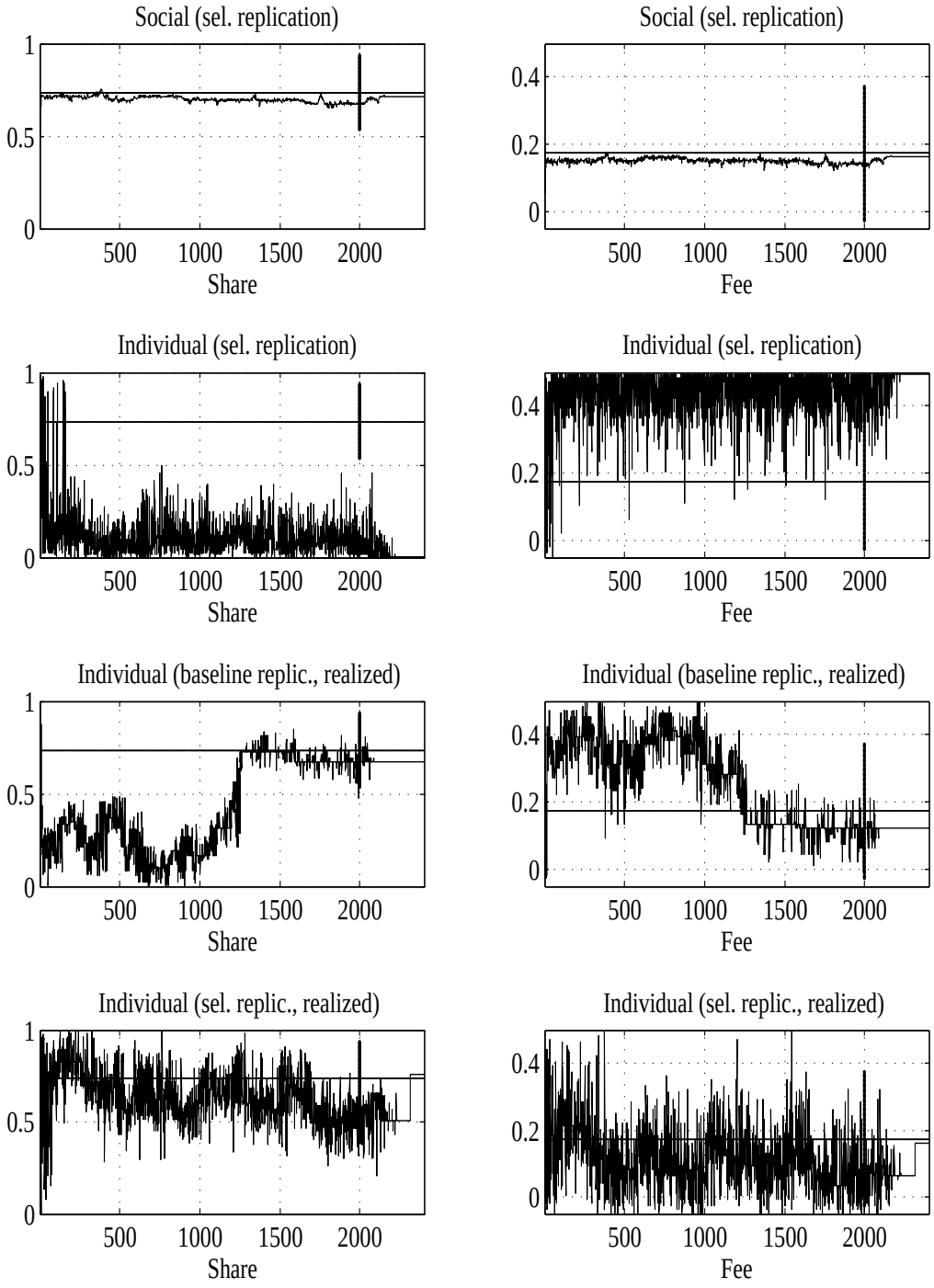

Figure 6: Strategies' Time Paths for Sample Run

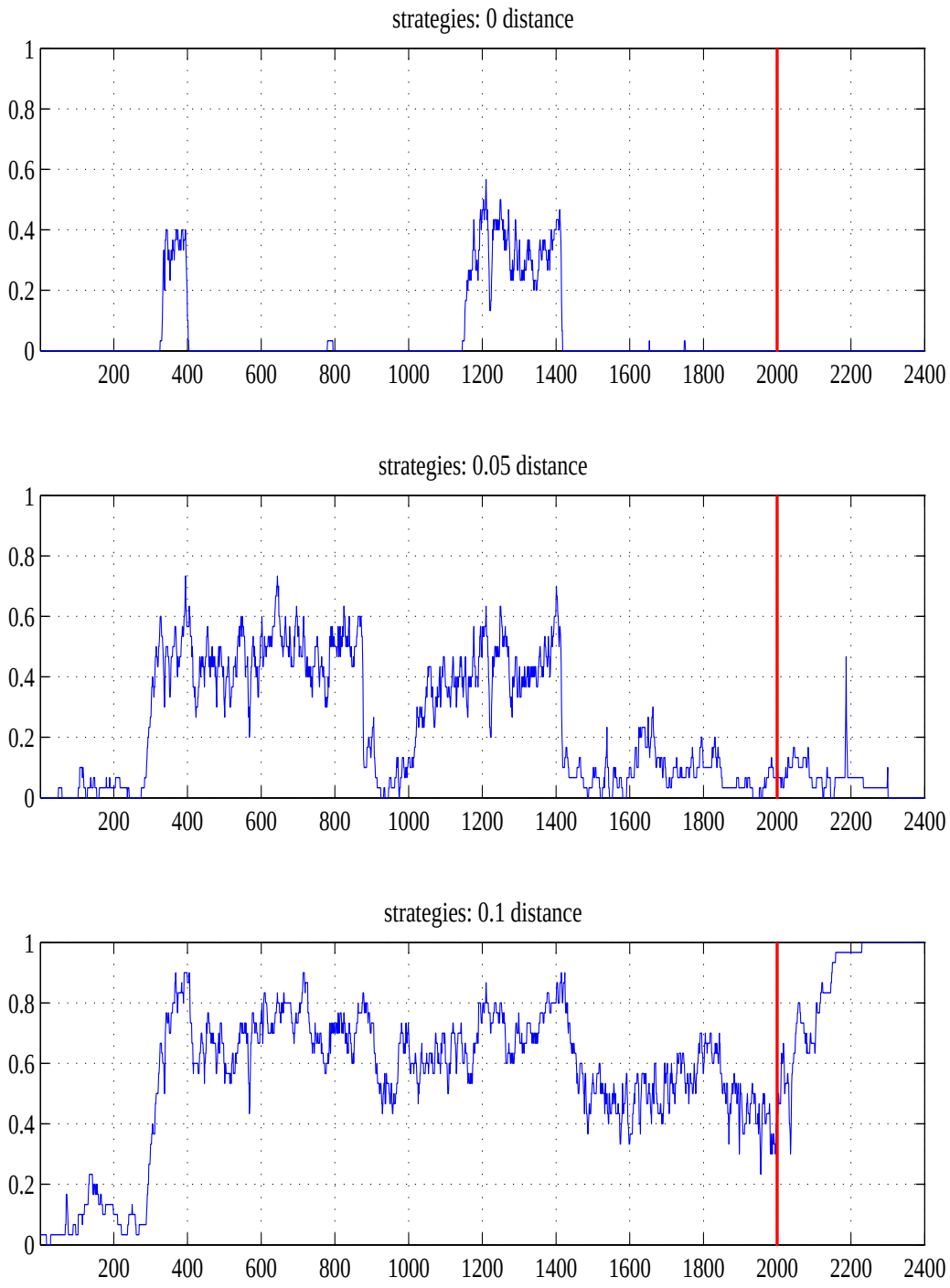

Figure 7: Sample Run for IEL with Realized Payoffs

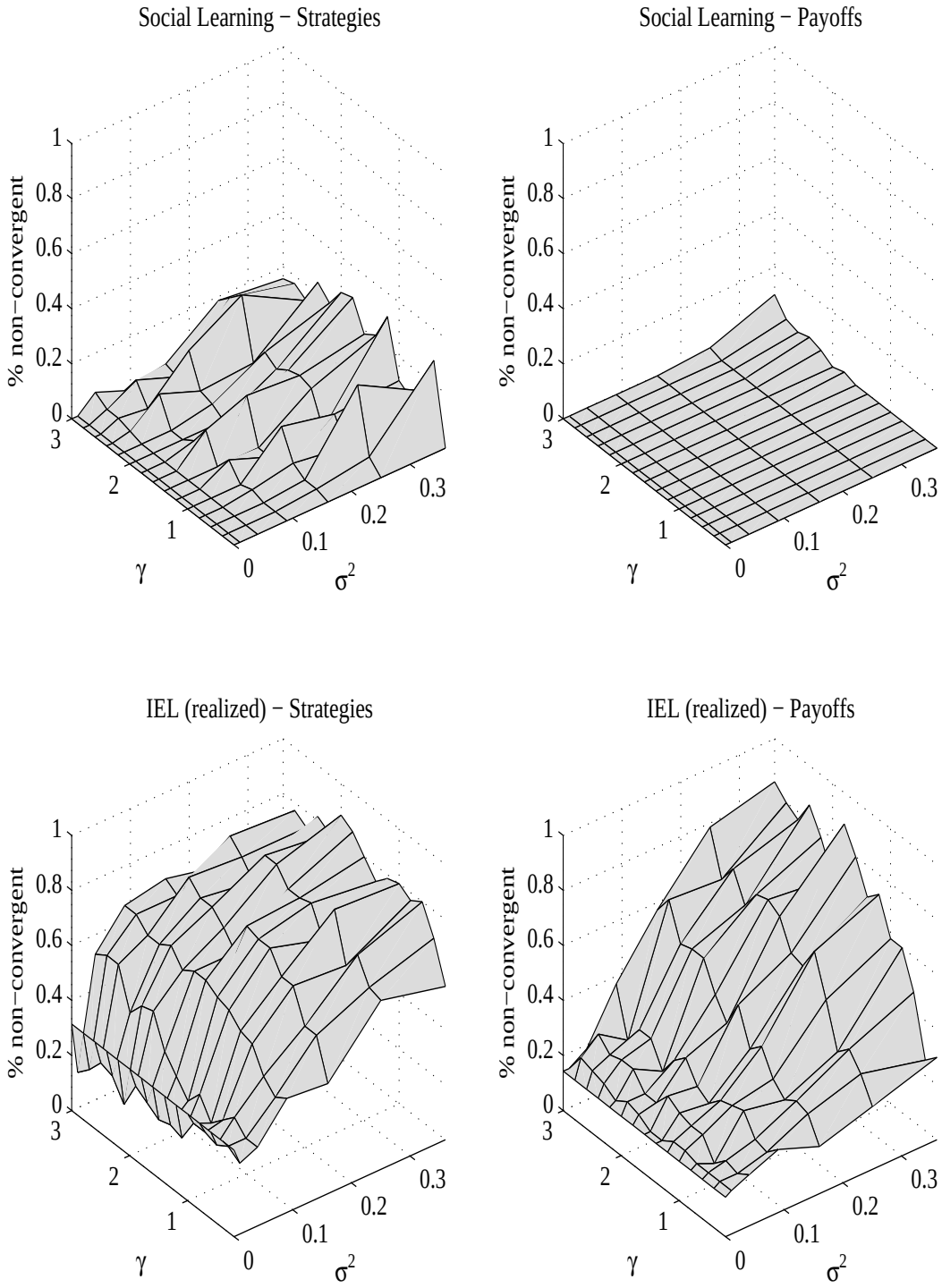

Figure 8: Fraction of Non-Convergent Simulations in Terms of Messages and Payoffs, Social and IEL with Realized Payoffs

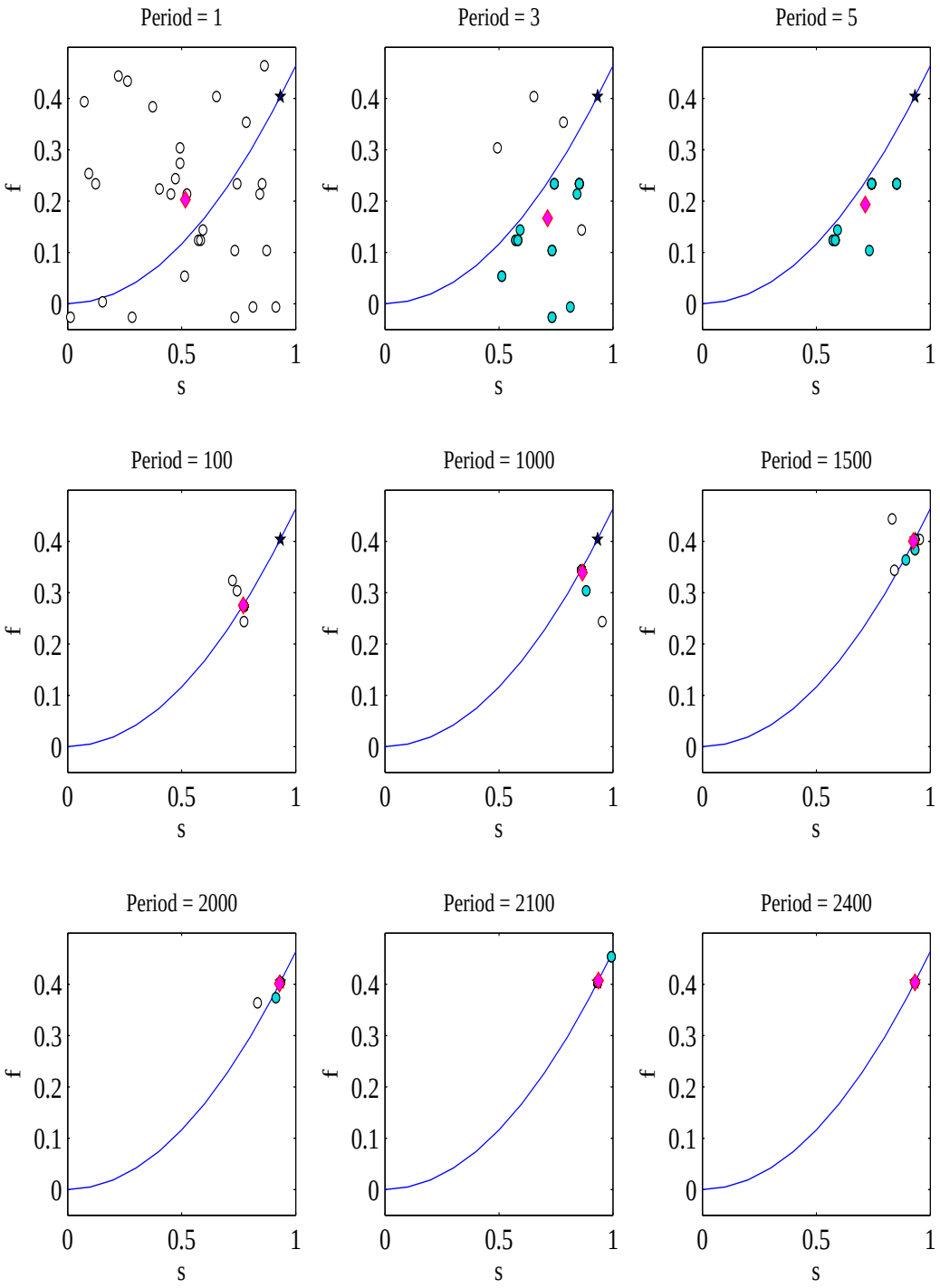

Figure 9: Evolution of a Typical Run - Social Learning

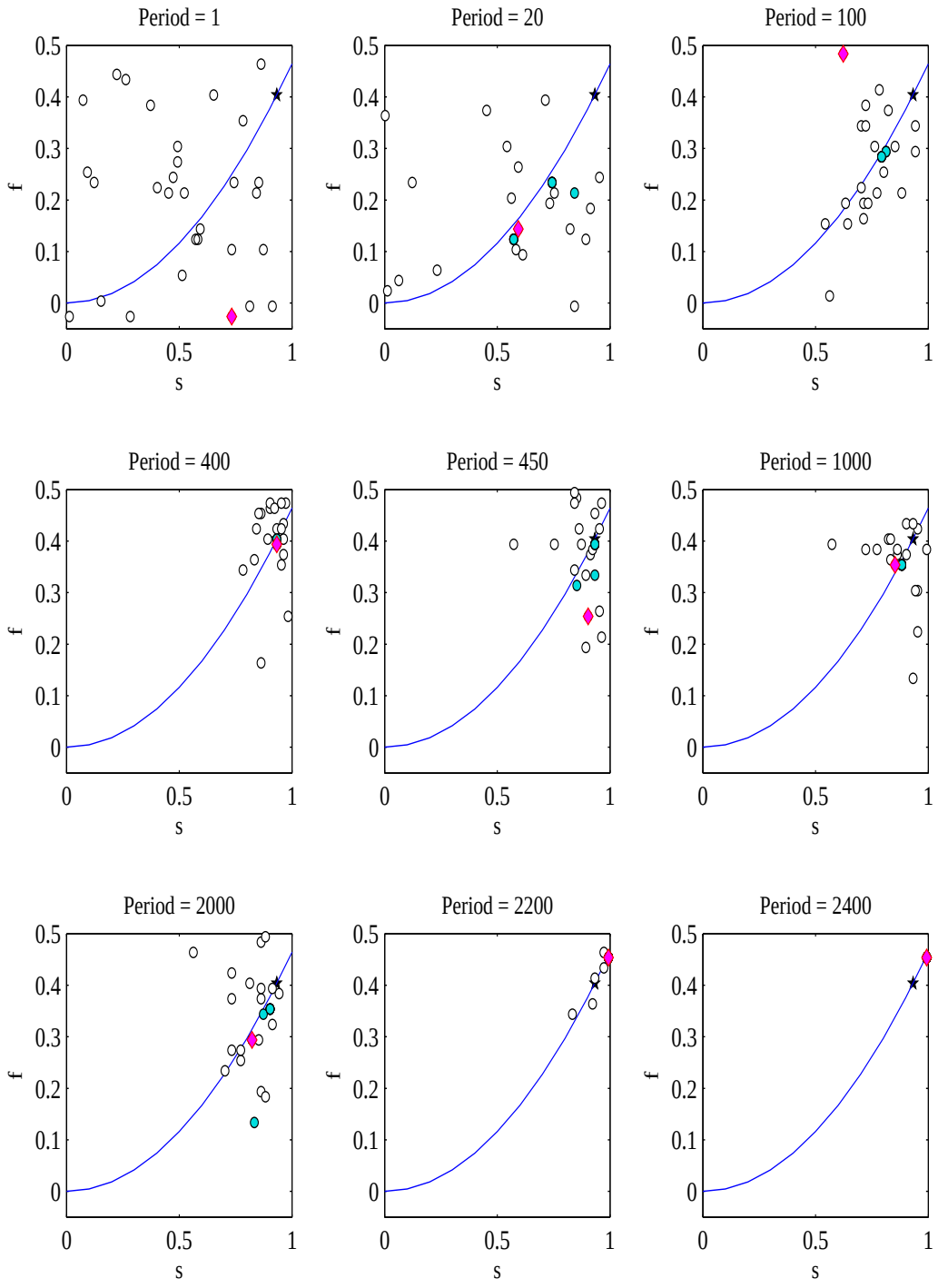

Figure 10: Evolution of a Typical Run - IEL with Realized Payoffs

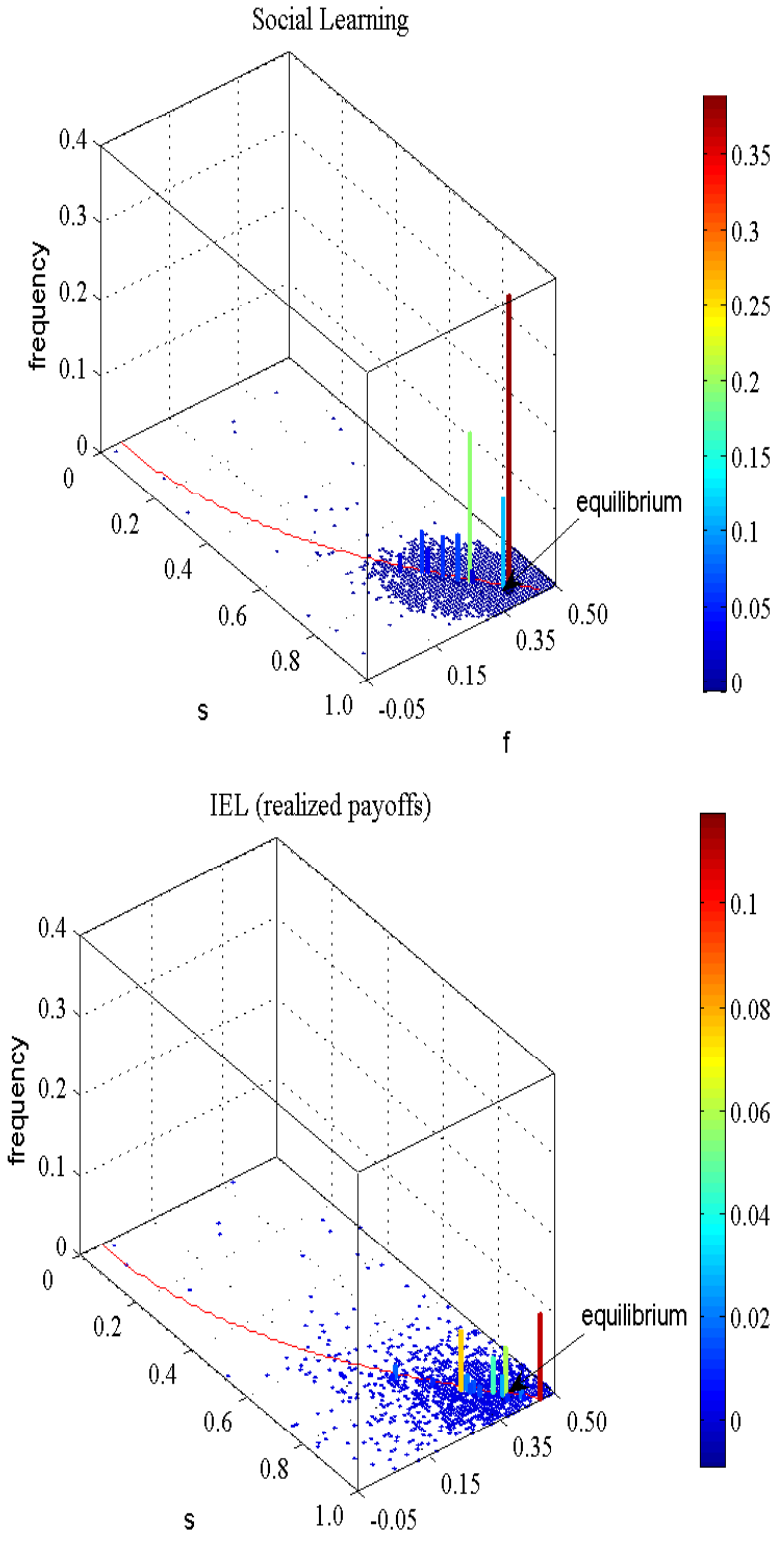

Figure 11: Frequency of Evaluations of Strategies During a Typical Run

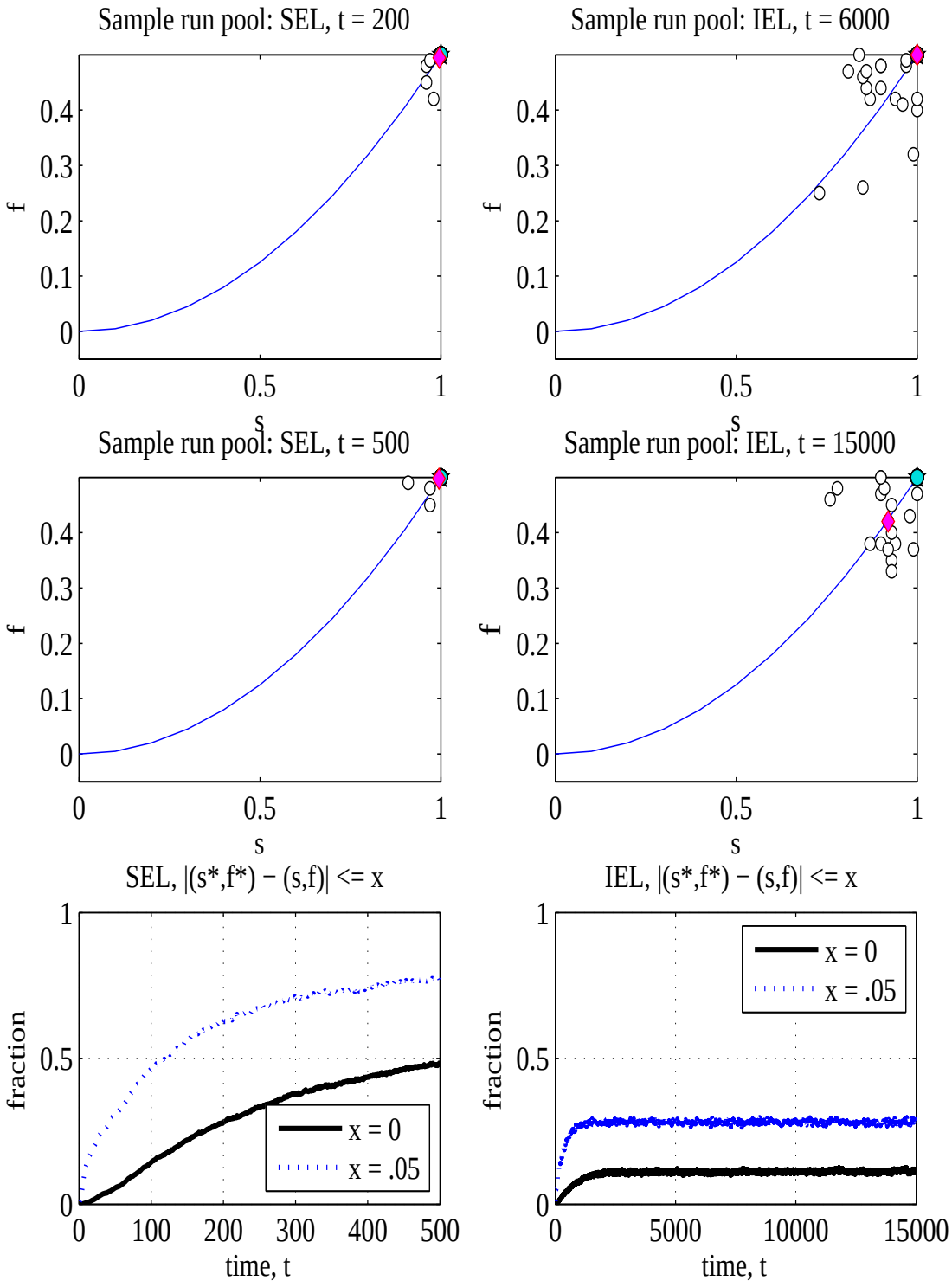

Figure 12: Equal Number of Evaluations