Computational learning techniques for intraday FX trading using popular technical indicators

Article *in* IEEE Transactions on Neural Networks · August 2001

DOI: 10.1109/72.935088 · Source: IEEE Xplore

4 authors, including:

Michael A. H. Dempster
University of Cambridge
335 PUBLICATIONS **4,521** CITATIONS

SEE PROFILE

Computational Learning Techniques for Intraday FX Trading Using Popular Technical Indicators

M. A. H. Dempster, Tom W. Payne, Yazann Romahi, and G. W. P. Thompson

***Abstract*—There is reliable evidence that technical analysis, as used by traders in the foreign exchange (FX) markets, has predictive value regarding future movements of foreign exchange prices. Although the use of artificial intelligence (AI)-based trading algorithms has been an active research area over the last decade, there have been relatively few applications to intraday foreign exchange—the trading frequency at which technical analysis is most commonly used. Previous academic studies have concentrated on testing popular trading rules in isolation or have used a genetic algorithm approach to construct new rules in an attempt to make positive out-of-sample profits after transaction costs. In this paper we consider strategies which use a collection of popular technical indicators as input and seek a profitable trading rule defined in terms of them. We consider two popular computational learning approaches, reinforcement learning and genetic programming (GP), and compare them to a pair of simpler methods: the exact solution of an appropriate Markov decision problem and a simple heuristic. We find that although all methods are able to generate significant in-sample and out-of-sample profits when transaction costs are zero, the genetic algorithm approach is superior for nonzero transaction costs, although none of the methods produce significant profits at realistic transaction costs. We also find that there is a substantial danger of overfitting if in-sample learning is not constrained.**

***Index Terms*—Computational learning, foreign exchange (FX), genetic algoriths (GA), linear programming, Markov chains, reinforcement learning, technical trading, trading systems.**

I. Introduction

SINCE the era of floating exchange rates began in the early 1970s, technical trading has become widespread in the *foreign exchange* (FX) markets. Academic investigation of technical trading however has largely limited itself to daily data. Although daily data is often used for currency overlay strategies within an asset-allocation framework, FX traders trading continuously throughout the day naturally use higher frequency data.

In this investigation, the relative performance of various optimization techniques in high-frequency (intraday) foreign exchange trading is examined. We compare the performance of a genetic algorithm (GA) and a reinforcement learning (RL) system to a simple linear program (LP) characterising a Markov decision process (MDP) and a heuristic.

In Section II, we give a brief literature review of preceding work in technical analysis. Sections III and IV then introduce the GA and RL methods. The stochastic optimization problem to be solved by all the compared methods is defined in Section V, while Sections VI–VIII describe in more detail how each approach can be applied to solve this optimization problem approximately. The computational experiments performed are outlined and their results given in Section IX. Section X concludes with a discussion of these results and suggests further avenues of research.

Reinforcement learning has to date received only limited attention in the financial literature and this paper demonstrates that RL methods show significant promise. The results also indicate that generalization and incorporation of constraints limiting the ability of the algorithms to overfit improves out-of-sample performance, as is demonstrated here by the genetic algorithm.

II. Technical Analysis

Technical analysis has a century-long history amongst investment professionals. However, academics have tended to regard it with a high degree of scepticism over the past few decades largely due to their belief in the efficient markets or random walk hypothesis. Proponents of technical analysis had until very recently never made serious attempts to test the predictability of the various techniques used and as a result the field has remained marginalized in the academic literature.

However, due to accumulating evidence that markets are less efficient than was originally believed (see, for example, [1]), there has been a recent resurgence of academic interest in the claims of technical analysis. Lo and MacKinlay [2], [3] have shown that past prices may be used to forecast future returns to some degree and thus reject the random walk hypothesis for United States stock indexes sampled weekly.

LeBaron [1] acknowledges the risk of bias in this research however. Since various rules are applied and only the successful ones are reported, he notes that it is not clear whether the returns achieved could have been attained by a trader who had to make the *choice* of rules in the first place. LeBaron argues that to avoid this bias it is best simply to look at rules that are both widely used and have been in use for a long period of time. Neely *et al.* [4] use a genetic programming based approach to avoid this bias and found out-of-sample net returns in the 1–7% per annum range in currency markets against the dollar during 1981 to 1995.

Although there has been a significant amount of work in technical analysis, most of this has been based on stock market data. However, since the early 1970s this approach to trading has been widely adopted by foreign currency traders [4]. A survey by Taylor and Allen [5] found that in intraday trading 90% of respondents reported the use of technical analysis, with 60% stating that they regarded such information as at least as important as economic fundamentals. Neely *et al.* [4] argue that this

Manuscript received October 16, 2000; revised February 27, 2001.

The authors are with the Centre for Financial Research, Judge institute of Management, University of Cambridge, Cambridge, U.K.

Publisher Item Identifier S 1045-9227(01)05018-4.

can be partly explained by the unsatisfactory performance of exchange rate models based on economic fundamentals. They cite Frankel and Rose [6] who state that ... no model based on such standard fundamentals like money supplies, real incomes, interest rates, and current-account balances will ever succeed in explaining or predicting a high percentage of the variation in the exchange rate, at least at short or medium-term frequencies.

A number of researchers have examined net returns due to various trading rules in the foreign exchange markets [7], [8]. The general conclusion is that trading rules are sometimes able to earn significant returns net of transaction costs and that this cannot be easily explained as compensation for bearing risk. Neely and Weller [9] note however that academic investigation of technical trading has not been consistent with the practice of technical analysis. As noted above, technical trading is most popular in the foreign exchange markets where the majority of intraday foreign exchange traders consider themselves technical traders. They trade throughout the day using high-frequency data but aim to end the day with a net open position of zero. This is in contrast to much of the academic literature which has tended to take much longer horizons into account and only consider daily closing prices.

Goodhart and O'Hara [10] provide a thorough survey of past work investigating the statistical properties of high-frequency trading data, which has tended to look only at narrow classes of rules. Goodhart and Curcio [11] examine the usefulness of resistance levels published by Reuters and also examine the performance of various filter rules identified by practitioners. Dempster and Jones [12], [13] examine profitability of the systematic application of the popular channel and head-and-shoulders patterns to intraday FX trading at various frequencies, including with an overlay of statistically derived filtering rules. In subsequent work [14], [15] upon which this paper expands, they apply a variety of technical trading rules to trade such data (see also Tan [16]) and also study a genetic program which trades combinations of these rules on the same data [17]. None of these studies report any evidence of *significant* profit opportunities, but by focussing on relatively narrow classes of rules their results do not necessarily exclude the possibility that a search over a broader class would reveal profitable strategies. Gencay *et al.* [18] in fact assert that simple trading models are able to earn significant returns after transaction costs in various foreign exchange markets using high frequency data.

III. Genetic Algorithms

In recent years, the application of artificial intelligence (AI) techniques to technical trading and finance has experienced significant growth. Neural networks have received the most attention in the past and have shown varying degrees of success. However recently there has been a shift in favor of user-transparent, nonblack box evolutionary methods like GAs and in particular genetic programming (GP). An increasing amount of attention in the last several years has been spent on these genetic approaches which have found financial applications in option pricing [19], [20] and as an optimization tool in technical trading applications [17], [14], [4].

Evolutionary learning encompasses sets of algorithms that are inspired by Darwinian evolution. GAs are population-based optimization algorithms first proposed by Holland [21]. They have since become an active research area within the artificial intelligence community and have been successfully applied to a broad range of hard problems. Their success is in part due to their several control parameters that allow them to be highly tuned to the specific problem at hand. GP is an extension proposed by Koza [22], whose original goal was to evolve computer programs.

Pictet *et al.* [23] employ a GA to optimize a class of exponentially weighted moving average rules, but run into serious overfitting and poor out-of-sample performance. They report 3.6% to 9.6% annual excess returns net of transaction costs, but as the models of Olsen and Associates are not publicly available their results are difficult to evaluate. Neely and Weller [9] report that for their GA approach, although strong evidence of predictability in the data is measured out-of-sample when transaction costs are set to zero, no evidence of profitable trading opportunities arise when transaction costs are applied and trading is restricted to times of high market activity.

IV. Reinforcement Learning

Reinforcement learning has so far found only a few financial applications. The reinforcement learning technique is strongly influenced by the theory of MDPs, which evolved from attempts to understand the problem of making sequences of decisions under uncertainty when each decision can depend on the previous decisions and their outcomes. The last decade has witnessed the merging of ideas from the reinforcement learning and control theory communities [24]. This has expanded the scope of dynamic programming and allowed the approximate solution of problems that were previously considered intractable.

Although reinforcement learning was developed independently of MDPs, the integration of these ideas with the theory of MDPs brought a new dimension to RL. Watkins [25] was instrumental in this advance by devising the method of Q-learning for estimating action-value functions. The nature of reinforcement learning makes it possible to approximate optimal policies in ways that put more effort into learning to make good decisions for frequently encountered situations at the expense of less effort for less frequently encountered situations [26]. This is a key property which distinguishes reinforcement learning from other approaches for approximate solution of MDP's.

As fundamental research in reinforcement learning advances, applications to finance have started to emerge. Moody *et al.* [27] examine a recurrent reinforcement learning algorithm that seeks to optimize an online estimate of the Sharpe ratio. They also compare the recurrent RL approach to that of Q-learning.

V. Applying Optimization Methods to Technical Trading

In this paper, following [15], [17], [14], we consider trading rules defined in terms of eight popular technical indicators used by intraday FX traders. They include both buy and sell signals based on simple trend-detecting techniques such as moving averages as well as more complex rules. The indicators we use are

the price channel breakout, adaptive moving average, relative strength index, stochastics, moving average convergence/divergence, moving average crossover, momentum oscillator, and commodity channel index. A complete algorithmic description of these indicators can be found in [15], [14].

To define the indicators, we first aggregate the raw tick data into (here) quarter-hourly intervals, and for each compute the *bar* data—the *open*, *close*, *high*, and *low* FX rates. Most of the indicators use only the closing price of each bar, so we will introduce the notation $\boldsymbol{F}_t$ to denote the *closing* GBP:USD FX *rate* (i.e., the dollar value of one pound) of bar t (here we use boldface to indicate random entities).

We define the market state s_t at time t as the binary string of length 16 giving the *buy* and *sell pounds* indications of the eight indicators, and define the state space $\mathcal{S} = \{0,1\}^{16}$ as the set of all possible market states. Here a 1 represents a trading recommendation for an individual indicator whose entry is otherwise 0. In effect, we have constructed from the available tick data a discrete-time data series: at time t (the end of the bar t interval) we see F_t, compute s_t and must choose whether or not to switch currencies based on the values of the indicators incorporated in s_t and which currency is currently held. We consider this time series to be a realization of a binary string valued stochastic process and make the required trading decisions by solving an appropriate stochastic optimization problem.

Formally, a trading strategy ϕ is a function $\phi : \mathcal{S} \times \{0,1\} \to \{0,1\}$, $(s,h) \mapsto \phi(s)$, for some current position h ($= 0$, dollars, or 1, pounds), telling us whether we should hold pounds ($\phi = 1$) or dollars ($\phi = 0$) over the next timestep. It should be noted that although our trading strategies ϕ are formally *Markovian* (feedback rules), some of our technical indicators require a number of periods of previous values of F to decide the corresponding 0-1 entries in s_t. The objective of the trading strategies ϕ used in this paper is to maximize the expected dollar return (after transaction costs) up to some horizon T:

$$\mathbb{E}\left\{\prod_{t=1}^{T-1}\left(\frac{\boldsymbol{F}_{t+1}}{\boldsymbol{F}_t}\right)^{\phi(\boldsymbol{s}_t)}(1-c)^{|\phi(\boldsymbol{s}_t)-\phi(\boldsymbol{s}_{t-1})|}\right\} \tag{1}$$

where $\mathbb{E}$ denotes expectation, c is the proportional transaction cost, and ϕ is chosen with the understanding that trading strategies start in dollars, observe s_1 and then have the opportunity to switch to pounds. Since we do not have an explicit probabilistic model for how FX rates evolve, we cannot perform the expectation calculation in (1), but instead adopt the familiar approach of dividing our data series into an in-sample region, over which we optimize the performance of a candidate trading strategy, and an out-of-sample region where the strategy is ultimately tested.

The different approaches utilized solve slightly different versions of the in-sample optimization problem. The simple heuristic and Markov Chain methods find a rule which takes as input a market state and outputs one of three possible actions: either "hold pounds," "hold dollars" (switching currencies if necessary) or "stay in the same currency."

The GA and RL approaches find a rule which takes as input the market state and the currency currently held, and chooses between two actions: either to stay in the same currency or switch. Thus the RL and GA method are given slightly more information (their current position) than the heuristic and MDP methods and we might thus expect them to perform better. The GA method also has an extra constraint restricting the complexity of the rules it can generate which is intended to stop overfitting of the in-sample data.

VI. Applying RL to the Technical Trading Problem

The ultimate goal of reinforcement learning based trading systems is to optimize some relevant measure of trading system performance such as profit, economic utility or risk-adjusted return. A standard RL framework has two central components; an agent and an environment. The agent is the learner and decision maker that interacts with the environment. The environment consists of a set of states and available actions for the agent in each state.

The agent is bound to the environment through perception and action. At a given time step t the agent receives input i, which is representative of some state $s_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of all possible states in the environment. As mentioned in the previous section, s_t is defined here as being a combination of the technical indicator buy and sell pounds decisions prepended to the current state of the agent (0 for holding dollars and 1 for pounds). The agent then selects an action $a_t \in \mathcal{A}$ where $\mathcal{A} := \{0,1\}$ telling it to hold pounds ($\phi = 1$) or dollars ($\phi = 0$) over the next timestep. This selection is determined by the agent's policy π ($:= \phi$, i.e., defined in our case as the trading strategy) which is a mapping from states to probabilities of selecting each of the possible actions.

For learning to occur while iteratively improving the trading strategy (policy) over multiple passes of the in-sample data, the agent needs a merit function that it seeks to improve. In RL, this is a function of expected return R which is the amount of return the agent expects to get in the future as a result of moving forward from the current state. At each learning episode for every time-step t the value of the last transition is communicated to the agent by an immediate reward in the form of a scalar reinforcement signal r_t. The expected return from a state is therefore defined as

$$\begin{aligned} R_t :=& r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-1} r_T \\ =& \sum_{k=0}^{T} \gamma^k r_{t+k+1} \end{aligned} \tag{2}$$

where γ is the *discount factor* and T is the final time step. Note that the parameter γ determines the "far-sightedness" of the agent. If $\gamma = 0$ then $R_t = r_{t+1}$ and the agent myopically tries to maximize reward only at the next time-step. Conversely, as $\gamma \to 1$ the agent must consider rewards over an increasing number of future time steps to the horizon. The goal of the agent is to learn over a large number of episodes a policy mapping of $\mathcal{S} \to A$ which maximizes R_t for all $t = 0, \cdots, T$ as the limit of the approximations obtained from the same states at the previous episode.

In our implementation, the agent is directly attempting to maximize (1). The reward signal is therefore equivalent to actual returns achieved from each state at the previous episode.

This implies that whenever the agent remains in the base currency, regardless of what happens to the FX rate, the agent is neither rewarded nor penalized.

Often RL problems have a simple goal in the form of a single state which when attained communicates a fixed reward and has the effect of delaying rewards from the current time period of each learning episode. Maes and Brookes [28] show that immediate rewards are most effective—when they are feasible. RL problems can in fact be formulated with separate state spaces and reinforcement rewards in order to leave less of a temporal gap between performance and rewards. In particular it has been shown that successive immediate rewards lead to effective learning. Matarič [29] demonstrates the effectiveness of multiple goals and progress estimators, for example, a reward function which provides instantaneously positive and negative rewards based upon "immediate measurable progress relative to specific goals."

It is for this reason that we chose to define the immediate reward function (2) rather than to communicate the cumulative reward only at the end of each trading episode.

In reinforcement learning the link between the agent and the environment in which learning occurs is the value function V^{π}. Its value for a given state is a measure of how "good" it is for an agent to be in that state as given by the total expected future reward from that state under policy π. Note that since the agent's policy π determines the choice of actions subsequent to a state, the value function evaluated at a state must depend on that policy. Moreover, for any two policies π and π' we say that π is preferred to π', written $\pi > \pi'$, if and only if $\forall s \in S$, $V^{\pi}(s) \geq V^{\pi'}(s)$. Under suitable technical conditions there will always be at least one policy that is at least as good as all other policies. Such a policy π^* is called an optimal policy and is the target of any learning agent within the RL paradigm. To all optimal policies is associated the optimal value function V^*, which can be defined in terms of a dynamic programming recursion as

$$V^*(s) = \max_a \mathbb{E}\{\boldsymbol{r}_{t+1} + \gamma V^*(\boldsymbol{s}_{t+1}) | \boldsymbol{s}_t = s\}. \tag{3}$$

Another way to characterize the value of a state s is to consider it in terms of the values of all the actions a that can be taken from that state assuming that an optimal policy π^* is followed subsequently. This value Q^* is referred to as the Q-value and is given by

$$Q^*(s,a) = \mathbb{E}\{\boldsymbol{r}_{t+1} + \gamma \max_{a'} Q^*(\boldsymbol{s}_{t+1}, a') | \boldsymbol{s}_t = s, \boldsymbol{a}_t = a\}. \tag{4}$$

The optimal value function expresses the obvious fact that the value of a state under an optimal policy must equal the expected return for the best action from that state, i.e.,

$$V^*(s) = \max_a Q^*(s,a).$$

The functions Q^* and V^* provide the basis for learning algorithms for MDPs.

Q-learning [25] was one of the most important breakthroughs in the reinforcement learning literature [26]. In this method, the learned action-value function Q directly approximates the optimal action-value function Q^* and dramatically simplifies the analysis of the algorithm to enable convergence proofs. As a bootstrapping approach, Q-learning estimates the Q-value function of the problem based on estimates at the previous learning episode. The Q-learning update is the backward recursion

$$Q(s_t, a_t) \leftarrow Q(s_{t_c}, a_{t_c}) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_{t_c}, a_{t_c})] \tag{5}$$

where the current state-action pair $(s_t, a_t) = (s_{t_c}, a_{t_c})$ from the previous learning episode. At each iteration (episode) of the learning algorithm, the action-value pairs associated with all the states are updated and over a large number of iterations their values converge to optimality for (4). We note that there are some parameters in (5): in particular, the *learning rate* α refers to the extent with which we update the current Q-factor based on future rewards, γ refers to how "far-sighted" the agent is and a final parameter of the algorithm is the policy followed in choosing the potential action at each time step. Q-learning has been proven to converge to the optimal policy regardless of the policy actually used in the training period [25]. We find that following a random policy while training yields the best results.

In order for the algorithm to converge, the learning rate α must be set to decrease over the course of learning episodes. Thus α has been initially set to 0.15 and converges downwards to 0.000 15 at a rate of $0.15/(1 + (E/10))$, where E is the episode (iteration) number which runs from 0 to 10 000. The parameter γ has been set to 0.9999 so that each state has full sight of future rewards in order to allow faster convergence to the optimal.

With this RL approach we might expect to be able to outperform all the other approaches on the in-sample data set. However on the out-of-sample data set, in particular at higher slippage values, we suspect that some form of generalization of the input space would lead to more successful performance.

VII. Applying the Genetic Algorithm

The approach chosen extends the genetic programming work initiated in [14] and [17]. It is based on the premise that practitioners typically base their decisions on a variety of technical signals, which process is formalized by a trading rule. Such a rule takes as input a number of technical indicators and generates a recommended position (long £, neutral, or long $). The agent applies the rule at each timestep and executes a trade if the rule recommends a different position to the current one.

Potential rules are constructed as binary trees in which the terminal nodes are one of our 16 indicators yielding a Boolean signal at each timestep and the nonterminal nodes are the Boolean operators `AND`, `OR`, and `XOR`. The rule is evaluated recursively. The value of a terminal node is the state of the associated indicator at the current time; and the value of a nonterminal node is the associated Boolean function applied to its two children. The overall value of the rule is the value of the root node. An overall rule value of one (true) is interpreted as a recommended long £ position and zero (false) is taken as a recommended neutral position. Rules are limited to a maximum depth of four (i.e., a maximum of 16 terminals) to limit complexity. An example rule is shown in Fig. 1. This

definition of a rule generalizes that used in [17], [14] which allows trees in the comb form of Fig. 1, but to depth 10.

The fitness score of such a rule ϕ is defined as the *total return* cumulated over the appropriate data period [*cf.* (1)], i.e.

$$\prod_{t=1}^{T-1}\left(\frac{F_{t+1}}{F_t}\right)^{\phi(s_t)}(1-c)^{|\phi(s_t)-\phi(s_{t-1})|}. \tag{6}$$

The genetic algorithm is used to search the space of all such rules and is tuned to favor rules that trade successfully (i.e., achieve high fitness scores) in the in-sample training period. An initial population of 250 rules is randomly generated and each rule is evaluated according to (6). A new population of rules is generated from this in which high scoring rules are preferred to low scoring rules. This bias means that the fitness scores of the new population should be greater than those of the old population. New rules are generated by two processes: crossover and mutation.

To use crossover two parent rules are selected from the current population. The selection process is biased toward fitter (better performing) rules: all rules in the current population are ranked in order of fitness score, and are chosen with a probability linearly proportional to their rank. A random subtree is chosen from each parent rule and these two subtrees are swapped between the parent rules to create two new rules, each of which inherits characteristics from both parents. This process is shown in Fig. 2. Potential subtrees to swap are checked to ensure that the resulting new rules would not exceed the specified maximum tree depth. Two new rules meeting this criterion are inserted into the new population. For mutation, a single rule is selected from the current population (again biased toward the better performers) and a random node is replaced with a random node of the same type (e.g., an AND might become an OR). The mutated rule is then inserted into the new population.

New rules are generated using 75% crossover and 25% mutation until a total of 250 new rules are generated, when the new population is evaluated for fitness scores. The average score of the new population should be greater than that of the old due to the favoritism shown to the better performing rules in the old population. This process is repeated 100 times (*generations*) and *the* best rule found during the entire run is selected as the *output* of the genetic algorithm.

The rules found by this process exhibit a number of desirable properties. First, with careful tuning of the GA they should perform well in-sample. Second, a rule can be understood by humans: it is clear what the rule does, even if it is not clear why it does it. Thirdly, this structure limits (but does not prevent) how much a rule can learn in detail (i.e., *overfit*) the training data set. It is to be expected that this enforced generalization will lead to better out-of-sample performance with a possible reduction in in-sample performance.

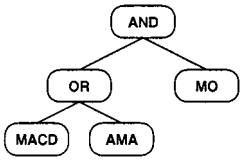

Fig. 1. An example rule.

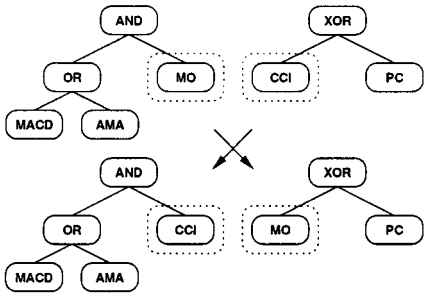

Fig. 2. Genetic algorithm crossover.

VIII. Markov Chain and Simple Heuristic

In addition to the RL and GA methods we will consider two alternative approaches. The first replaces the in-sample dataset with a Markov chain on a small set of market states and replaces the problem of maximizing the profit made over the in-sample period with that of maximizing a total expected discounted return assuming Markov dynamics. This approach is described in detail in Section VIII-A.

The second method is a simple *heuristic*: with each state we associate a number (which will be interpreted as the expected rise in the exchange FX rate over the next ℓ trading periods for some ℓ) and consider strategies which buy pounds if this number exceeds one threshold, and sell pounds when it falls below a second threshold. We then optimize over ℓ and the two thresholds to maximize the in-sample profit. More details on this method are given in Section VIII-B.

These two methods were used to benchmark the success of the true computational learning approaches in maximizing the in-sample profit. Neither are likely to attain the true optimum, but they should perform reasonably well. The heuristic was more successful at solving the in-sample problem with nonzero transaction costs than any of the other approaches, but it does not perform particularly well out-of-sample due to over-fitting.

A. A Markov-Chain Linear Programming Approximation

Recall that s_t denotes the market-state at time t (the values of the indicator recommendations on which the trading decision at time t is based). In this section we will let $\overline{\mathcal{S}}$ denote the set of all market states which occur in the in-sample dataset.

We will construct a controlled Markov chain on the set of pairs (s,h) where $s \in \overline{\mathcal{S}}$ denotes a market state present in the in-sample dataset $\overline{\mathcal{S}} = \cup_{1\le t\le T}\{s_t\}$ and $h \in \{0,1\}$ indicates whether or not pounds are currently held. The *controls* at each timestep are 0 or 1, indicating the currency we wish to hold over the next timestep as before.

Denoting by $N(s_1)$ the number of times state s_1 appears in the in-sample dataset (excluding the final data-point), and by $N(s_1,s_2)$ the number of times state s_1 is immediately followed by state s_2, we define a controlled Markov chain $\boldsymbol{X}_t$,

$t = 1, 2, \cdots$, by fixing $\boldsymbol{X}_1 := (s_1, 0)$ and choosing the probability of a transition from (s_1, h_1) to (s_2, h_2) using control k to be

$$P^{(k)}((s_1,h_1),(s_2,h_2)) := I(h_2 = k)\frac{N(s_1,s_2)}{N(s_1)}.$$

For $s \in \overline{\mathcal{S}}$, $h \in \{0,1\}$ we define an approximation to the expected dollar return over the next timestep given we are in state s and hold currency h as

$$R(s,h) := I(h=1)\frac{1}{N(s)}\sum_{1\le t<T:s_t=s}\log\left(\frac{F_{t+1}}{F_t}\right).$$

We are now in a position to replace the problem of maximizing the in-sample return with the problem

$$\max_{\pi}\mathbb{E}\sum_{t=1}^{\infty}\gamma^{t-1}[R(\boldsymbol{X}_t,\pi(\boldsymbol{X}_t)) + |\pi(\boldsymbol{X}_t)-\pi(\boldsymbol{X}_{t-1})|\log(1-c)] \quad (7)$$

where π is a feedback map from the state-space of the Markov chain to the set of controls $\{0,1\}$ and we treat the term $\pi(X_0)$ as zero (since we must start in dollars). The constant γ is a discount factor, chosen arbitrarily to be equal to 0.9999. This is an approximation to the objective (1).

Since the set $\overline{\mathcal{S}}$ is quite small, this problem can be solved exactly using the technique of linear programming. To see this, observe that the solution to (7) is characterized by the optimal value function $J^*(s,h)$ defined for all $s \in \overline{\mathcal{S}}, h \in \{0,1\}$ as the optimal value (1) when $X_1 = (s,h)$. The function $J^*(\cdot,\cdot)$ satisfies the dynamic programming recursion

$$J^*(s,h) = \max_{k\in\{0,1\}}\left(R(s,k)+|k-h|\log(1-c) + \gamma\sum_{s'\in\overline{\mathcal{S}},h'\in\{0,1\}}P^{(k)}((s,h),(s',h'))J^*(s',h')\right) \quad (8)$$

whose solution is unique under various conditions (it suffices that the Markov chain has a finite statespace and that $0 < \gamma < 1$, which is the case here). But a solution to the system above can be obtained by solving the linear program

$$\min_{J(\cdot,\cdot)}\sum_{s\in\overline{\mathcal{S}},h\in\{0,1\}}J(s,h)$$

subject to for all $s \in \overline{\mathcal{S}}$ and $h,k \in \{0,1\}$

$$J(s,h) \ge R(s,k)+|k-h|\log(1-c)+\gamma\sum_{s'\in\overline{\mathcal{S}},h'\in\{0,1\}}P^{(k)}((s,h),(s',h'))J(s',h'). \quad (9)$$

which must yield the required optimal value function J^*. The optimal action $\pi(s,h)$ in state (s,h) can be extracted from the optimal value function $J^*(\cdot,\cdot)$ by choosing any k maximizing the right-hand side of (8). The solution to the LP is found for each in-sample period using the CPLEX commercial LP solver.

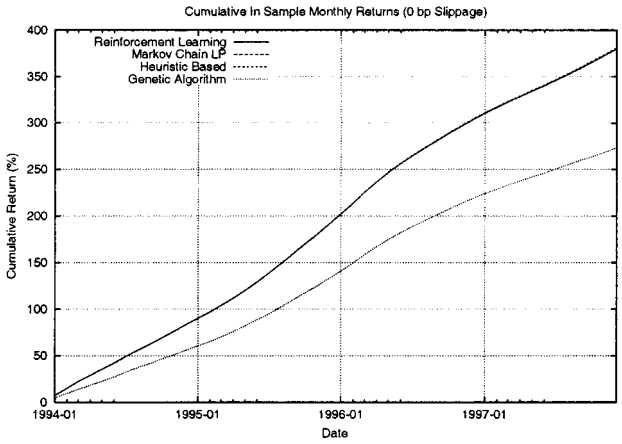

Fig. 3. Cumulative in-sample monthly returns at no slippage.

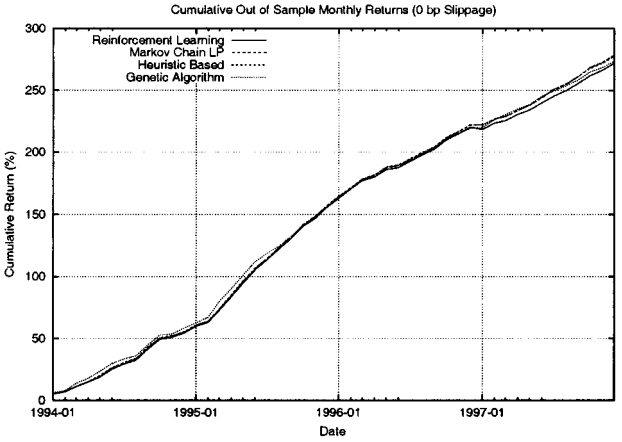

Fig. 4. Cumulative out-of-sample monthly returns at no slippage.

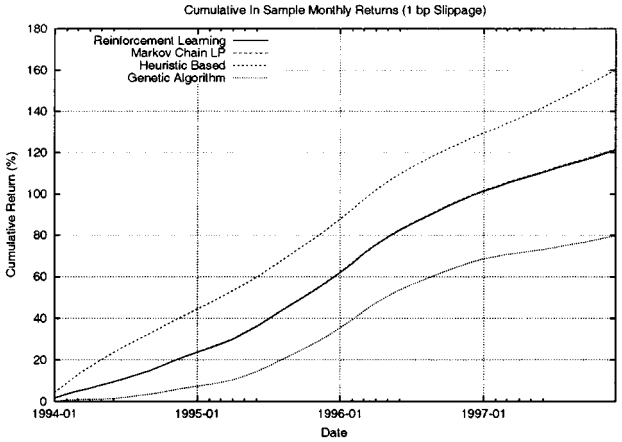

Fig. 5. Cumulative in-sample monthly returns at 1 bp slippage.

B. A Simple Heuristic

One objection to the method of the previous section is that when transaction costs are large the solution obtained to the problem in (7) above may perform badly over the actual in-sample period; worse even than the trivial strategy 'always hold dollars' with a return of 0. As an alternative, we consider a *heuristic* defined in terms of three parameters (b, x, ℓ):

$$buypounds \text{ when } \mathbb{E}(\log(\boldsymbol{F}_{1+\ell}/\boldsymbol{F}_1)|\boldsymbol{s}_1 = s) \quad > b \quad (10)$$

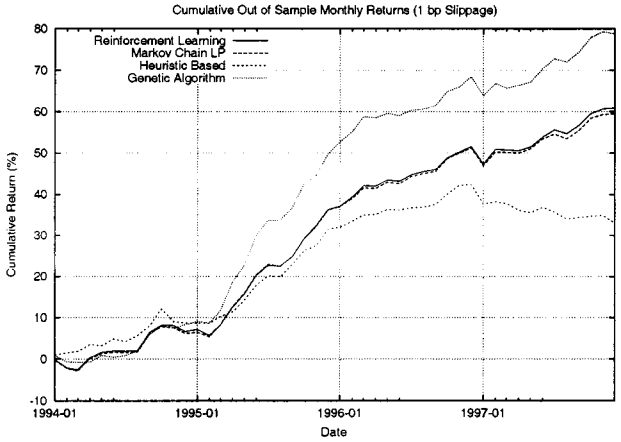

Fig. 6. Cumulative out-of-sample monthly returns at 1 bp slippage.

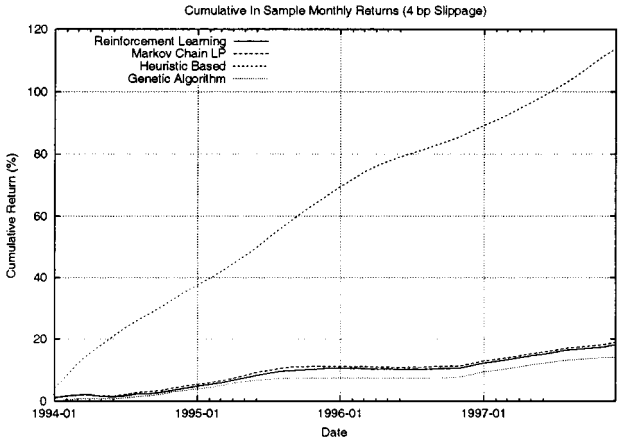

Fig. 7. Cumulative in-sample monthly returns at 4 bp slippage.

$$sellpounds \text{ when } \mathbb{E}(\log(\boldsymbol{F}_{1+\ell}/\boldsymbol{F}_1)|\boldsymbol{s}_1 = s) \quad < x. \quad (11)$$

The expected value in (10) and (11) is just the expected return available if we held pounds for the next ℓ days given that the current market state is s. Since we do not have a stochastic process model for FX rate movements, this expectation must also be estimated from the in-sample data (assuming the ergodic theorem holds) as

$$\mathbb{E}(\log(\boldsymbol{F}_{1+\ell}/\boldsymbol{F}_1)|s_1 = s) \approx \frac{1}{|Y|}\sum_{t\in Y}\log\left(\frac{F_{t+\ell}}{F_t}\right)$$

where Y is the set $\{t : 1 \le t \le T-\ell, s_t = s\}$.

For a several classes of stochastic process models for FX dynamics, the optimal strategy for both very low and very high transaction costs has the form of (10) and (11), making it a plausible heuristic in general.

The optimization of the three parameters of the heuristic is a nonconvex multiextremal problem and for each in-sample period is solved by a simple genetic algorithm.

IX. Numerical Experiments

The results reported below were obtained by applying the approaches described above to the GBP:USD exchange rate data from January 1994 until January 1998 using a moving window of one year for training (fitting) followed by one month out-of-sample testing.

The cumulative (without reinvestment) returns over the period are shown in odd-numbered Figs. 3–9 for selected in-sample (fitting) cases and evenly-numbered Figs. 4–10 for corresponding out-of-sample (testing) cases. The annualised

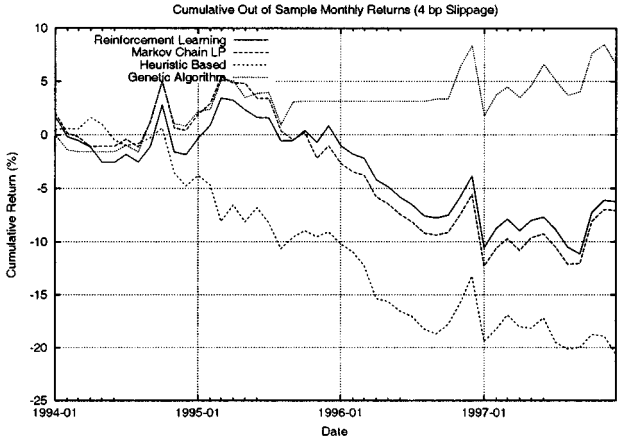

Fig. 8. Cumulative out-of-sample monthly returns at 4 bp slippage.

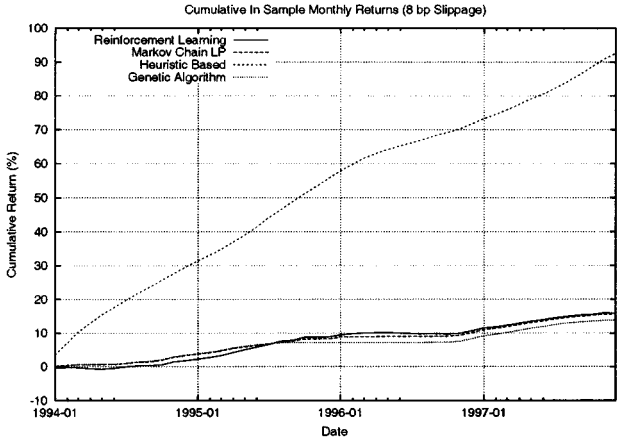

Fig. 9. Cumulative in-sample monthly returns at 8 bp slippage.

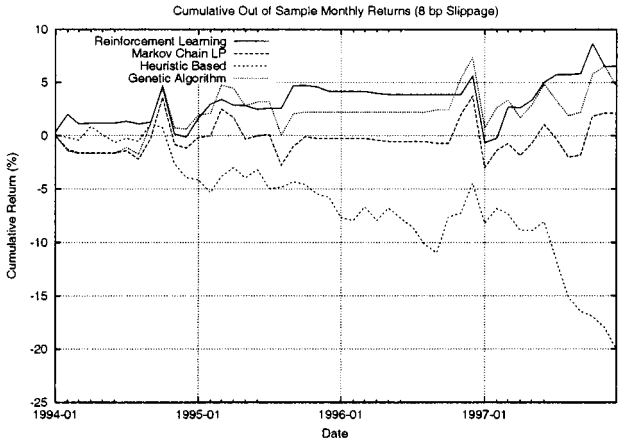

Fig. 10. Cumulative out-of-sample monthly returns at 8 bp slippage.

TABLE I
OUT-OF-SAMPLE AVERAGE ANNUAL RETURNS

Slippage	0 bp	1 bp	4 bp	8 bp	10 bp
RL	93.8	16.3	-1.55	1.64	1.45
GA	94.5	21.6	1.67	1.17	1.71
LP	96.8	15.9	-1.76	0.53	0.432
Heuristic	96.3	8.56	-5.03	-4.89	-5.63

average monthly returns are shown in Table I for the various approaches in the out-of-sample case.

In-sample fitting performance has been shown for completeness to demonstrate the learning ability of the various approaches. It is clear that on the in-sample data set, the simple heuristic approach consistently outperforms all the other methods except in the no slippage case, when all methods were able to fit the data to essentially the same degree. The out-of-sample test results demonstrate, however, that the heuristic approach was in fact significantly *overfitting* the data.

For the out-of-sample back-tests, we note that the genetic algorithm and reinforcement learning approaches tended to outperform the others at lower slippage values. In order to gain further insight into the overall best performing GA, a plot of how often it inferred rules using each indicator for each slippage value is shown in Fig. 11. Fig. 12 shows the frequencies with which the GA employed specific indicators over the entire four year data period, aggregated for all slippage values and into quarters, with considerable variability in the patterns evident. The GAs reduction in trading frequency with decreasing transaction costs is demonstrated dramatically in Fig. 13. Similar results apply to the other methods with the exception of the heuristic, whose high trading frequency at realistic transaction costs leads to its poor performance in out-of-sample back-tests. Data on the dealing frequency of all the different approaches is given in Table II in order to shed light on the risk profiles of the different methods. These results are discussed further in the final Section X.

In order to evaluate the relative risk-adjusted performance of the trading models further, we now consider several risk measures found in the financial literature. A risk-adjusted measure commonly used to evaluate portfolio models is the Sharpe ratio, defined as

$$\text{Sharpe Ratio} := \frac{\mu_{R_{\text{month}}}}{\sigma_{R_{\text{month}}}}. \tag{12}$$

The Sharpe ratio evaluations of our trading models, as shown in Table III demonstrate that on the dataset used we are able to gain significant risk-adjusted returns up to a slippage value of 1 bp. However the Sharpe ratio is numerically unstable for small variances of returns and cannot consider the clustering of profit and loss trades [13], [18]. Furthermore the Sharpe ratio penalizes strategies for upside volatility and its definition in terms of summary statistics means that a strategy can appear to be successful but in fact suffers from significant drawdown [6]. Maximum drawdown D_T over a certain period of length $T = t_1 - t_0$ is defined as

$$D_T := \max(R_{t_a} - R_{t_b} | t_0 \le t_a \le t_b \le t_1) \tag{13}$$

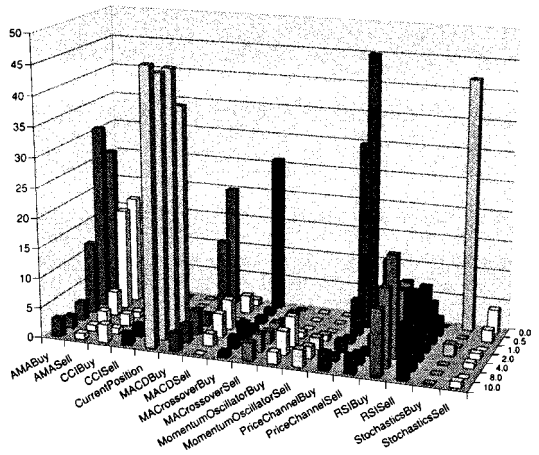

Fig. 11. Indicators used by the genetic algorithm by slippage.

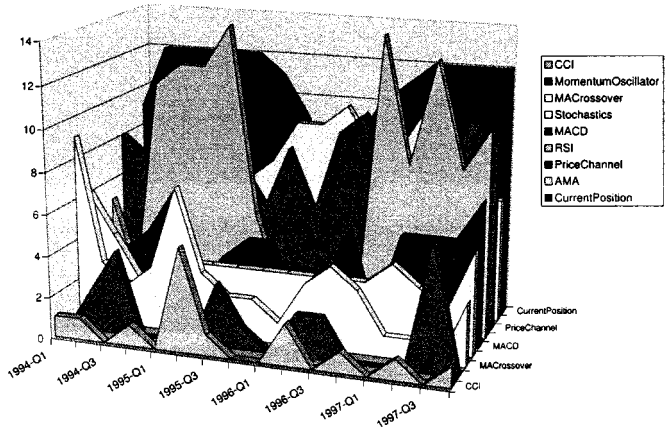

Fig. 12. Relative frequencies over time of indicators used by the genetic algorithm.

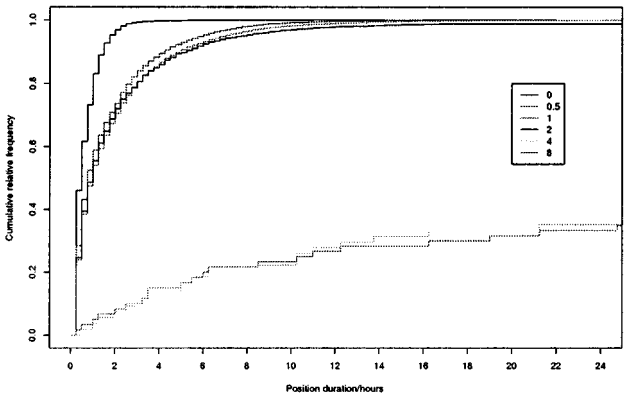

Fig. 13. Genetic algorithm position duration by slippage.

TABLE II
OUT-OF-SAMPLE AVERAGE MONTHLY DEALING FREQUENCY

Slippage	0 bp	1 bp	4 bp	8 bp	10 bp
RL	851	220	20.5	0.980	0.604
GA	759	254	0.688	0.688	0.667
LP	852	218	20.5	1.06	0.792
Heuristic	846	123	15.0	7.85	7.44

TABLE III
OUT-OF-SAMPLE SHARPE RATIOS

Slippage	0 bp	1 bp	4 bp	8 bp	10 bp
RL	2.10	0.682	-0.00711	0.885	0.0874
GA	2.18	0.732	0.0758	0.0531	0.785
LP	2.23	0.663	-0.0783	0.249	0.0202
Heuristic	2.22	0.397	-0.261	-0.279	-0.290

TABLE IV
OUT-OF-SAMPLE DRAWDOWNS

Slippage	0 bp	1 bp	4 bp	8 bp	10 bp
RL	1.38	1.71	2.25	1.29	0.932
GA	1.45	1.67	1.96	1.94	1.88
LP	1.36	1.72	2.28	1.91	1.89
Heuristic	1.36	1.90	2.03	1.91	2.16

TABLE V
OUT-OF-SAMPLE STIRLING RATIOS

Slippage	0 bp	1 bp	4 bp	8 bp	10 bp
RL	5.44	1.31	0.115	0.236	0.209
GA	5.08	1.72	0.246	0.225	0.239
LP	5.54	1.31	0.121	0.175	0.184
Heuristic	5.45	0.883	0.0375	-0.0152	-0.0457

where R_{t_a} and R_{t_b} are the total returns of the periods from t_0 to t_a and t_b as defined by (6) respectively. In our case T was defined as on a monthly basis and the mean over the out-of-sample back-test period is reported in Table IV. We therefore also quote the *Stirling ratio*, defined as

$$\text{Stirling Ratio} := \frac{R_{\text{month}}}{D_{\text{month}}} \tag{14}$$

which is the average monthly return divided by the maximum drawdown within that month. This value averaged over the 48 monthly back-test periods is reported in Table V.

The current RL implementation requires about eight minutes CPU time on a 650 MHz Athlon per single training optimization (episode). The GA is implemented in the interpreted language Scheme, but evaluation is parallelised over multiple similar CPUs. It also takes about eight minutes CPU time per optimization on a single machine. The Markov chain and heuristic approaches execute in four seconds and approximately four minutes, respectively.

X. DISCUSSION AND CONCLUSION

In this paper we have developed three trading strategies based on computational learning techniques and one simple heuristic based on trading thresholds over a fixed horizon. The strategies based on the genetic (programming) algorithm (GA) and reinforcement (Q-) learning train at 15-min intervals on the buy-sell signals from eight popular technical trading indicators—some of which require a number of previous observations—and current positions over a one year period of GBP:USD FX data, while the Markov chain strategy uses the *entire* set of training data to estimate the relative transition frequencies of the few hundred signal states that occur within a given year. Each of the four trading strategies is then evaluated out-of-sample at 15-min intervals on the next month of indicator signals and this back-testing process is then rolled forward a month and repeated for a total of 48 months.

It is evident that in-sample, all approaches were able to infer successful trading strategies and also notable that the genetic algorithm consistently underperforms the other methods in-sample. This is undoubtably due to the constraint imposed on the complexity of the rules which was specifically imposed to avoid overfitting. By contrast, the non-GA approaches—in particular the trading threshold heuristic—may end up exploiting noise in the in-sample data set. At zero-slippage (no costs of trading), however, we find that all approaches are able to infer similar strategies and perform similarly out-of-sample. There is evidence that the non-GA approaches do in fact overfit as the GA outperforms the other methods with nonzero transaction costs in the out-of-sample cases up to 8 bp slippage.

The fact that the techniques investigated here return positive results both in-sample and in out-of-sample back-tests implies that there is useful information in technical indicators that can be exploited. This is consistent with the tenets of technical analysis and contradictory to the efficient market hypothesis. Furthermore, the GAs relatively good out-of-sample performance demonstrates that using a combination of technical indicators leads to better performance than using the individual indicators themselves. In fact, Dempster and Jones [15], [14] demonstrate that with a few exceptions these indicators are largely unprofitable on the same data when considered in isolation. Figs. 11 and 12 demonstrate that some indicators also convey more information than others depending on the slippage value and the market state. We note that the relative strength index (buy/sell) indicators are not used at zero transaction costs but as the slippage is increased, the GA tends to favor them. Indicators such as price channel breakout, stochastics, and moving average crossover (buy) are very important at zero slippage, but the GA appears to disregard the information provided by them at higher slippage values. At zero slippage the GA is able to infer successful strategies without using the current position. However, at higher transaction costs, knowing the current position becomes very important. This lends credence to the argument that this extra position information tends to favor the RL and GA approaches, since the Markov chain approach and the heuristic did not have this information available to them.

The RL approach, the Markov chain and the heuristic all exploit the fact that of the 2^{16} market possible states only a few hundred actually occur in the in-sample period. This number is small enough that each state may be considered individually when deciding a strategy. However, there are two problems with such a rule when it is back-tested out-of-sample.

Firstly, we may encounter a state in the out-of-sample data which was not present in the in-sample data. In that case some arbitrary action must be made and both the RL and the Markov chain method choose to hold their current position. This may be a disadvantage if many new states are encountered and it also ignores the fact that some new states may be very similar to states which were present in the in-sample period. The genetic algorithm on the other hand generates a trading rule in the in-sample

training period whose structure tends to take the same actions in similar states.

Secondly, there is a severe danger that these approaches may learn too well (overfit) the specific in-sample data; in other words the in-sample problem they attempt to solve is *too specific*. Indeed the simple heuristic method demonstrates this quite clearly: it achieves excellent in-sample performance but is mediocre out-of-sample and terrible at realistic transaction costs. The limit on the complexity of the GA is an artificial constraint which reduces the opportunity for the GA to overfit while not prohibiting simple trading rules. This limit effectively forces the GA to work with *generalized* (classes of specific) states. Thus we hope to improve the current reinforcement learning approach by forcing state generalization and also by improving its convergence properties.

Another current avenue of research is to find constraints for the in-sample optimization problem which force state generalization (such as the rule-complexity constraint in the GA approach), but for which a heuristic similar to that of Section VIII-B can be applied.

A further goal is the exploration of generalization methods in the context of a broader RL approach. *Neuro-dynamic programming* (NDP) [24] attempts to combine neural networks with the central ideas of dynamic programming in order to address this goal. NDP's employ parametric representations of the value function (such as artificial neural networks) to overcome the curse of dimensionality. The free parameters are tuned using regression or stochastic approximation methods used in combination with classical dynamic programming methods. Further, Wilson's *X-classifier system* [30] attempts to merge ideas from reinforcement learning with those from the Classifier System community in order to incorporate generalization into a Q-Learning-like framework. We also intend to investigate this approach in the present context in the future.

ACKNOWLEDGMENT

The authors would like to thank Dr. C. Jones for his insightful comments, advice, and proofreading of the paper, and J. Scott for lending his expertise in Scheme.

REFERENCES

[1] B. LeBaron, "Technical trading rule profitability and foreign exchange intervention," *J. Int. Economics*, vol. 49, pp. 124–143, 1999.

[2] A. W. Lo and A. C. MacKinlay, "Stock market prices do not follow random walks: Evidence froma simple specification test," *Rev. Financial Studies*, vol. 1, pp. 41–66, 1988.

[3] ——, *A Nonrandom Walk Down Wall Street*. Princeton, NJ: Princeton Univ. Press, 1999.

[4] C. J. Neely, P. A. Weller, and R. Dittmar, "Is technical analysis in the foreign exchange market profitable? A genetic programming approach," *J. Financial Quantitative Anal.*, pp. 405–426, Dec. 1997.

[5] M. P. Taylor and H. Allen, "The use of technical analysis in the foreign exchange market," *J. Int. Money Finance*, vol. 11, pp. 304–314, June 1992.

[6] J. A. Frankel and A. K. Rose, "A survey of empirical research on nominal exchange rates," Nat. Bureau Economic Res., Working Paper No. 4865, Sept. 1994.

[7] R. Levich and L. Thomas, "The significance of technical trading rule profits in the foreign exchange market: A bootstrap approach," *J. Int. Money Finance*, vol. 12, pp. 451–474, Oct. 1993.

[8] C. R. Osler and K. Chang, "Head and shoulders: Not just a flaky pattern," Federal Reserve Bank of New York, Staff Papers no. 4, Aug. 1995.

[9] C. Neely and P. Weller, "Intraday technical trading in the foreign exchange market," Federal Reserve Bank of St. Louis, Working Paper 99-016A, 1999.

[10] C. Goodhart and M. O'Hara, "High frequency data in financial markets: Issues and applications," *J. Empirical Finance*, vol. 4, pp. 73–114, 1997.

[11] C. Goodhart and R. Curcio, When support/resistance levels are broken, can profits be made? Evidence from the foreign exchange market, London School of Economics, July 1992.

[12] M. A. H. Dempster and C. M. Jones, "Can channel pattern trading be successfully automated?," European J. Finance, 2001, to be published.

[13] ——, Can technical pattern trading be successfully automated? 2. Head and shoulders, Centre Financial Res., Judge Inst. Management, Univ. Cambridge, July 1999.

[14] C. M. Jones, "Automated Technical Foreign Exchange Trading with High Frequency Data," Ph.D. dissertation, Centre Financial Res., Judge Inst. Management, Univ. Cambridge, June 1999.

[15] M. A. H. Dempster and C. M. Jones, The profitability of intra-day FX trading using technical indicators, , Centre Financial Res., Judge Inst. Management, Univ. Cambridge, Dec. 2000.

[16] G. W. Tan, "Topics in foreign exchange trading systems," M.S. thesis, Centre Financial Res., Judge Inst. Management, Univ. Cambridge, June 1999.

[17] M. A. H. Dempster and C. M. Jones, "A real-time adaptive trading system using genetic programming," Quant. Finance, 2001, to be published.

[18] R. Gencay, G. Ballocchi, M. Dacorogna, R. Olsen, and O. Pictet, Real-time trading models and the statistical properties of foreign exchange rates, in Internal Document RAG.1988-12-01, Olsen & Associates, Zurich, Switzerland, 1998.

[19] S. Chen and W. Lee, "Option pricing with gas: A second report," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 1, 1997, pp. 21–25.

[20] N. K. Chidambaran, C. H. J. Lee, and J. R. Trigueros, "An adaptive evolutionary approach to option pricing via genetic programming," in *Genetic Programming 1998: Proc. 3rd Annu. Conf.*, J. R. Koza, W. Banzhaf, K. Chellapilla, K. Deb, M. Dorigo, D. B. Fogel, M. H. Garzon, D. E. Goldberg, H. Iba, and R. Riolo, Eds. Madison, WI, July 22–25, 1998, pp. 38–41.

[21] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.

[22] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[23] O. V. Pictet, M. M. Dacorogna, B. Chopard, M. Oudsaidene, R. Schirru, and M. Tomassini, "Using genetic algorithm for robust optimization in financial applications," *Neural Network World*, vol. 5, no. 4, pp. 573–587, 1995.

[24] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Waltham, MA: Athena Scientific, 1996.

[25] C. Watkins, "Learning from Delayed Reward," Ph.D. dissertation, Dept. Eng., Univ. Cambridge, 1989.

[26] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[27] J. Moody, L. Wu, Y. Liao, and M. Saffell, "Performance functions and reinforcement learning for trading systems and portfolios," *J. Forecasting*, vol. 17, pp. 441–470, 1998.

[28] P. Maes and R. Brooks, "Learning to coordinate behaviors," in *Proc. 8th Nat. Conf. Artificial Intell.*, July 29–Aug. 3, 1990, pp. 796–802.

[29] M. Mataric, "Learning in multi-robot systems," in *IJCAI-95 Workshop on Adaptation and Learning in Multiagent Systems*, 1995, pp. 32–37.

[30] S. Wilson, "Classifier fitness based on accuracy," *J. Evolutionary Comput.*, vol. 3, no. 2, pp. 149–175, 1995.

M. A. H. Dempster received the B.A. degree from Toronto University, Toronto, ON, Canada, the M.A. degree from Oxford University, Oxford, U.K., and the M.S. and Ph.D. degrees from Carnegie Mellon University, Pittsgurgh, PA, all in mathematics.

He is currently Professor of Management Studies (Finance and Management Science), Director of the Centre for Financial Research and Director of Research at the Judge Institute of Management Studies in the University of Cambridge. His present research interests include mathematical and computational finance and economics, optimization and nonlinear analysis, stochastic systems, algorithm analysis, decision support, and applications software and telecommunications systems modeling. He is author of more than 100 published research articles and reports and is author, editor or translator of eight books including *Mathematics of Derivative Securities* (Cambridge, U.K.: Cambridge Univ. Press, 1997).

Dr. Dempster is joint Editor-in-Chief of *Quantitative Finance* and Associate Editor of *Stochastics and Stochastic Reports*, *Journal of Economic Dynamics and Control*, *Computational Finance*, and *Computational Economics*.

Tom W. Payne received the M.A. degree in engineering and computer science from Clare College, Cambridge University, Cambridge, U.K., and the M.Sc. degree in artificial intelligence from Edinburgh University, Edinburgh, U.K. He is currently pursuing the Ph.D. degree at the Centre for Financial Research at the Judge Institute of Management Studies in the University of Cambridge.

His work focuses on the application on genetic algorithms to building successful automated trading systems.

Yazann Romahi received the B.Eng. degree in electronics and computer science and the M.Sc. degree in artificial intelligence from Edinburgh University, Edinburgh, U.K. He is pursuing the Ph.D. degree at the Centre for Financial Research at the Judge Institute of Management Studies in the University of Cambridge.

His Ph.D. work is involved in the investigation of FX Market Dynamics and optimal trading in the FX markets in the presence of transaction costs using artificial intelligence-based techniques.

G. W. P. Thompson received the M.A. and Ph.D. degrese in mathematics from the Queen's College, Cambridge University, Cambridge, U.K.

He is a Research Associate at the Centre for Financial Research at the Judge Institute of Management Studies in the University of Cambridge. His Ph.D. work investigated Gaussian models for interest-rates, numerical techniques for pricing complex options in Gaussian markets, and the optimal trading of securities in the presence of transaction costs. His current research interests also include optimization, particularly stochastic programming.