Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Economics and Finance 3 (2012) 68 – 77

Procedia Economics and Finance

www.elsevier.com/locate/procedia

Emerging Markets Queries in Finance and Business

Testing different Reinforcement Learning configurations for financial trading: Introduction and applications

Francesco Bertoluzzo[a], Marco Corazza[a, b, *]

[a] *Ca' Foscari University of Venice – Department of Economics, Cannaregio n. 873, Venezia 30121, Italy*
[b] *Advanced School of Economics, Cannaregio n. 873, Venezia 30121, Italy*

Abstract

The construction of automatic Financial Trading Systems (FTSs) is a subject of research of high interest for both academic environment and financial one due to the potential promises by self-learning methodologies and by the increasing power of actual computers. In this paper we consider Reinforcement Learning (RL) type algorithms, that is algorithms that optimize their behavior in relation to the responses they get from the environment in which they operate, without the need for a supervisor. In particular, first we introduce the essential aspects of RL which are of interest for our purposes, then we present some original automatic FTSs based on differently configured RL algorithms and apply such FTSs to artificial and real time series of daily financial asset prices.

Keywords: Financial Trading System; Reinforcement Learning; Stochastic control; *Q*-learning algorithm; Kernel-based Reinforcement Learning.

1. Introduction

Many resources have been recently spent among professionals as well as in the academic world in searching for automatic *Financial Trading System*s (FTSs) that can substitute for experience, capacity, and intuition human operators in the choice of investments in financial markets.

In the specialized literature several different approaches have been considered to address such a problem: Statistical and econometric models using, among others, macroeconomic indicators; Simpler and flexible models based on the Technical Analysis indicators; Models based on machine learning that make extensive use of computing capacity.

In this paper we consider a class of self-adaptive algorithms for dealing with the above problem in order to obtain profitable and fully automatic FTSs. In particular, the methodology we consider is known as *Reinforcement Learning* (RL) also known as Neuro-Dynamic Programming. In our approach, investment decision making is viewed as a stochastic control problem and the investment strategies are discovered directly interacting with the market. So, the need to build forecasting models for future prices or returns is eliminated.

* Corresponding author. Tel.: +39-(0)41-234-6921; Fax: +39-(0)41-234-7444.
E-mail address: corazza@unive.it

doi:10.1016/S2212-5671(12)00122-0

In literature there are many contributions in this research field. Among the ones which are of interest for us we recall Moody, Wu, Liao, and Saffel, 1998; Moody and Saffel, 2001; Gold, 2003. Generally, they show that the proposed trading strategies perform better than the ones based on other learning methodologies when market frictions are considered.

With respect to the prominent literature, in this paper we will do the following:

- We will develop and we will apply FTSs based on two different RL approaches, namely the *Temporal Difference* one (see subsection 2.3) and the *Kernel-based Reinforcement Learning* (see subsection 2.4);
- Beyond to consider the usual buy and sell signals, we will take into account also a third signal: The stay-out-from-the-market one;
- Instead of using the differential Sharpe ratio as performance indicator, we will utilize the classical Sharpe ratio computed on the last $L \in \mathbb{N}$ trading days.

The remainder of the paper is organized as follows: In section 2 we will introduce the essential aspects of RL which are of interest for our purposes; In section 3 we will present our RL-based FTSs and we will provide the results of their applications to artificial and real time series; In section 4 we will give some concluding remarks.

2. Reinforcement Learning

Learning by interacting directly with the environment without the need of a supervisor is likely the more immediate idea about the nature of learning. The consequences of the actions of the learner (agent) lead the agent himself to choose what are the actions that allow to obtain the desired results and what are the ones to avoid. RL formalizes this kind of learning by maximizing a numerical reward Barto and Sutton, 1996. The agent has to discover which actions yield the most reward by trying them. RL is different from Supervised Learning (SL) in which the agent learns from examples provided by an external supervisor. SL is an important kind of learning, but in interactive problems it is often impractical to obtain examples of desired behavior that are representative of the situation in which the agent has to act. In uncharted and unknown environments (like, for instance, financial markets) the agent must to be able to learn only from its own past experience.

To formalize these first ideas, let us consider a system observed at discrete time steps in which the state at time step t, $s_t \in \mathscr{S}$, summarizes all information concerning the system available to the agent. In the RL framework it is assumed that the system satisfies the Markov property, that is that the probability of transition from the actual state s_t to the next one s_{t+1} depends only on the current state s_t. On the basis of s_t, the agent selects an action $a_t \in \mathscr{A}(s_t)$, where $\mathscr{A}(s_t)$ is the set of all possible actions the agent can take given the state s_t. At time step $t+1$ the agent receives a reward, $r(s_t, a_t, s_{t+1}) \in \mathbb{R}$, as consequence of his actions a_t and of the new state s_{t+1} in which he finds himself. The reward is a numerical representation of the satisfaction of the agent. Generally, the agent wish to maximize the expected value of some global return, $R(s_t)$, which is defined as function of the actual reward and of the future discounted ones. Without specifying the chosen action as argument, this function can be written as:

$$R(s_t) = r(s_t, a_t, s_{t+1}) + \gamma r(s_{t+1}, a_{t+1}, s_{t+2}) + \gamma^2 r(s_{t+2}, a_{t+2}, s_{t+3}) + \ldots,$$

where $\gamma \in (0,1)$ is the discount factor.

In RL a policy $\pi(s_t) = a_t$ is a mapping from states to actions defining the choice of action a_t given state s_t. In order to maximize the expected $R(s_t)$, RL searches for a suitable policy. Considering also the policy $\pi(\cdot)$ we can write the global return as:

$$R^{\pi}(s_t) = r(s_t, \pi(s_t), s_{t+1}) + \gamma r(s_{t+1}, \pi(s_{t+1}), s_{t+2}) + \gamma^2 r(s_{t+2}, \pi(s_{t+2}), s_{t+3}) + \ldots.$$

2.1. Value functions

RL approaches are generally based on estimating *value function*s. These functions of states (or state-action pairs) attribute a value to each state s_t (or state-action pairs) proportional to the rewards achievable in the future from the current state s_t (or state-action pairs). They evaluate how good is for the agent to be in a given state (or to perform a given action in a given state). The notion "how good" is defined in terms of R_t. In particular, the value of a state $s_t = s$

following policy $\pi(\cdot)$ is the expected sum of the current and the future discounted rewards when starting in state $s_t = s$ and thereafter following policy $\pi(\cdot)$, that is:

$$V^{\pi}(s) = \mathrm{E}\left[R^{\pi}(s_t)|s_t = s\right].$$

Similarly, the value of tacking action $a_t = a$ being in state $s_t = s$ under policy $\pi(\cdot)$ is the expected sum of the current and the future discounted rewards starting from state $s_t = s$, taking action $a_t = a$ and thereafter following policy $\pi(\cdot)$, that is:

$$Q^{\pi}(s,a) = \mathrm{E}\left[R^{\pi}(s_t)|s_t = s, a_t = a\right].$$

A fundamental property of value functions is that they satisfy particular recursive relationships. For any policy $\pi(\cdot)$ and any state $s_t = s$, the following consistency condition holds between the value of s_t and the value of any possible successor state s_{t+1}:

$$V^{\pi}(s) = \mathrm{E}\left[r(s_t, \pi(s_t), s_{t+1}) + \gamma V^{\pi}(s_{t+1})|s_t = s\right]. \tag{1}$$

Equation (1) is the *Bellman equation* for $V^{\pi}(s_t)$. One can prove that the value $V^{\pi}(s)$ is the unique solution to its Bellman equation.

2.2. Generalized policy iteration

Task of RL consists in finding an optimal policy, that is a policy which is better than or equal to all the other policies. The optimal policy identifies the values $V^*(s)$ and $Q^*(s,a)$ such that

$$V^*(s) = \max_{\pi} V^{\pi}(s) \text{ and } Q^*(s,a) = \max_{\pi} Q^{\pi}(s,a), \tag{2}$$

for all $s \in \mathscr{S}$ and for all $a \in \mathscr{A}(s)$. Since $V^*(s)$ is the value function for a policy, it satisfy the Bellman equation (1). Because it is also the optimal value function, $V^*(s)$'s Bellman condition can be written in a special form without reference to any specific policy. This form is the Bellman equation for $V^*(s)$, or the *Bellman optimality equation*, which expresses the fact that the value of a state under an optimal policy must equal the expected global return for the best action from the state itself, that is:

$$V^*(s) = \max_{a} Q^*(s,a) = \mathrm{E}\left[R^*(s_t)|s_t = s, a_t = a\right]. \tag{3}$$

With equivalent arguments, the Bellman optimality equation for $Q^*(s,a)$ is:

$$Q^*(s,a) = \mathrm{E}\left[r(s_t, a_t, s_{t+1}) + \gamma \max_{a'} Q^*(s_{t+1}, a')|s_t = s, a_t = a\right].$$

At this point, it is possible to iteratively calculate the value function for a state (or for a state-action pair). Let $V_0^{\pi}(s_t)$ for all $s_t = s \in \mathscr{S}$ be an arbitrarily initialization of the state value function. Each successive approximation is obtained by using the Bellman equation as an update rule:

$$\widehat{V}_{k+1}^{\pi}(s_t) = \mathrm{E}\left[r(s_t, a_t, s_{t+1}) + \gamma \widehat{V}_{k}^{\pi}(s_{t+1})\right] \tag{4}$$

for all $s_t = s \in \mathscr{S}$. If the expectation $\widehat{V}_{k+1}^{\pi}(s_t)$ exists, then $\lim_{k \to +\infty} \widehat{V}_{k}^{\pi}(s) = V^{\pi}(s)$.

The reason for computing the value functions for a policy is to find better policies in order to increase the expected value of the global returns. This process is called *policy improvement*.

As described, the policy improvement process requires the evaluation of the previous policy. This evaluation can be made by (4), which is itself an iterative process that converges in the limit. Fortunately, there is no need to wait for the exact convergence, in fact one can stop the policy evaluation iteration in several ways without losing the convergence (Barto and Sutton (1996)). An important special case is when policy evaluation is stopped after just one step. One can

combine the evaluation process and the improvement one by stopping the policy evaluation process at each step and then by improving the policy itself. This mixed algorithm is called *generalized policy iteration*. It can be written as:

$$\widehat{V}^{\pi}_{k+1}(s_t) = \max_a \mathrm{E}\left[r(s_t,a,s_{t+1}) + \gamma \widehat{V}^{\pi}_{k}(s_{t+1})\right], \tag{5}$$

where $\widehat{V}^{\pi}_{k+1}(s_t)$ is the update estimate with the improved policy at step $k+1$ with respect to the old estimate and the old policy at step k.

In order to improve the policy we chosen an approach, among the ones presented in literature, which may produce increasing of the global return in the long run. Following such an approach, the choice of the action at each time step t is given by:

$$a_t = \begin{cases} \pi'(s_t) & \text{with probability } 1-\varepsilon \\ a \in \mathscr{A}(s_t) & \text{with probability } \varepsilon, \end{cases}$$

where $\varepsilon \in (0,1)$ and $\pi'(s_t)$ is the candidate action which maximizes $Q^{\pi}(s,a)$.

In the next subsections we will introduce the two different methods we will use in our FTSs for calculating the expected value (5). Note that we can not take into account methods like the *Dynamic Programming*-based ones and the *Monte Carlo*-based ones. In fact:

- The former needs a model to calculate the real probabilities of transition from a state to another one, whereas in the financial trading such a model is generally not known or not available;
- The latter, in order to improve the policy, needs to wait for until the end of all the trades, whereas a FTS trades an indefinite number of times.

2.3. Temporal Difference methods and Q-Learning algorithm

In this subsection we present a class of policy evaluation algorithms known as *Temporal Difference method*s (TDms) which update step by step the estimate of $V(s_t)$. First of all one puts in evidence that it is possible to write $\widehat{V}_{k+1}(s_t)$ in the following recursive way:

$$\widehat{V}_{k+1}(s_t) = \frac{1}{k+1}\sum_{j=1}^{k+1} R_j(s_t) = \frac{1}{k+1}\left[R_{k+1}(s_t) + \sum_{j=1}^{k} R_j(s_t)\right] = \cdots = \widehat{V}_k(s_t) + \alpha_k\left[R_{k+1}(s_t) - \widehat{V}_k(s_t)\right],$$

where $\alpha_k = 1/(k+1)$. The TDms can update the estimate $\widehat{V}_{k+1}(s_t)$ as soon as the quantity

$$d_k = R_{k+1}(s_t) - \widehat{V}_k(s_t) = r(s_t,s_{t+1}) + \gamma \widehat{V}_k(s_{t+1}) - \widehat{V}_k(s_t),$$

becomes available. Therefore the above recursive relationship can be rewritten as:

$$\widehat{V}_{k+1}(s_t) = \widehat{V}_k(s_t) + \alpha_k\left[r(s_t,s_{t+1}) + \gamma \widehat{V}_k(s_{t+1}) - \widehat{V}_k(s_t)\right].$$

With regard to the convergence of the TDms, one can prove that, if $\sum_{k=1}^{+\infty}\alpha_k = +\infty$ and $\sum_{k=1}^{+\infty}\alpha_k^2 < +\infty$, then $\lim_{k\to+\infty}\Pr\left\{\left|\widehat{V}_k(s_t) - V(s_t)\right| < \varepsilon\right\} = 1$, for any $\varepsilon > 0$ (Bertsekas and Tsitsiklis (1996)).

The TDms are "naturally" developed in an incremental amd on-line fashion which make them particularly appealing for the building of FTSs. In literature there exist several different TDms, the most widespread of whom is the Q-Learning algorithm (QLa). The QLa is an *off*-policy control method, where off indicates that two different policies are used in the policy improvement process: A first one is used to estimate the value functions, another is used to control the improvement process. One can prove that the so-obtained state-action value function is given by:

$$\widehat{Q}_{k+1}(s_t,a_t) = \widehat{Q}_k(s_t,a_t) + \alpha_k\left[r_{t+1} + \gamma \max_a \widehat{Q}_k(s_{t+1},a_{t+1}) - \widehat{Q}_k(s_t,a_t)\right]. \tag{6}$$

Up to now we have assumed that the states are discrete variables which, even more, assume a limited number of values. But generally a system like a financial market is characterized by continuous states which, of course, assume an infinite number of values. In this case one can prove that the value function at step k, $\widehat{V}_k(s_t)$, can now be approximated by a parameterized functional form with parameter vector θ_k. It involves that the associated value function $\widehat{V}_k(s_t) = \widehat{V}_k(s_t;\theta_k)$ totally depends on θ_k, which varies step by step.

In order to estimate the optimal parameter vector, θ^*, which minimizes the "distance" between the unknown $V^\pi(s_t)$ and its estimate $\widehat{V}^\pi(s_t;\theta_k)$, in most learning approaches the minimization of the mean square error is used, that is:

$$\min_{\theta_k} \sum_s \left[V^\pi(s) - \widehat{V}^\pi(s;\theta_k) \right]^2.$$

The convergence of θ_k to θ^* is proven for approximators characterized by simple functional fors like linear ones, and it is also possible for particular complex functional forms like the ones involved like the so-called artificial neural networks(Bertsekas and Tsitsiklis (1996)). Among the linear functional forms, in building our FTSs we use the following one:

$$\widehat{V}^\pi(s;\theta) = \sum_{i=1}^{n} \theta_i \phi_i(s_i) = \theta' \phi(s), \tag{7}$$

where n is the number of states and $\phi_i(\cdot)$ is a suitable transformation of the state. One can prove that, under mild assumptions, the update rules to use for estimating the state-action value function in the case of continuous states become:

$$d_k = r(s_t, a_t, s_{t+1}) + \gamma \max_a \widehat{Q}(s_{t+1}, a; \theta_k) - \widehat{Q}^\pi(s_t, a_t; \theta_k) \text{ and } \theta_{k+1} = \theta_k + \alpha d_k \nabla_{\theta_k} \widehat{Q}^\pi(s_t, a_t; \theta_k). \tag{8}$$

2.4. Kernel-based Reinforcement Learning

Another method for approximating $Q^\pi(s,a)$, alternative to the QLa and also usable in case of continuous states, is the nonparametric regression-based one known as *Kernel-based Reinforcement Learning* (KbRL) (Ormonet (2002), Smart and Kaelbling (2000)). Given a kernel $K(\cdot)$ and defined $q_t = (s_t \; a_t)$, the KbRL estimates $Q(s,a)$ as follows:

$$\widehat{Q}_k(q_t) = \sum_{i=1}^{t-1} p_i(q_t) \widehat{Q}_k(q_i),$$

where $p_i(q_t) = \frac{K\left(\frac{q_t - q_i}{h}\right)}{\sum_{i=1}^{t-1} K\left(\frac{q_t - q_i}{h}\right)}$. Note that the choice of the kernel $K(\cdot)$ is not crucial, whereas the choice of the bandwidth h has to be done carefully (Bosq (1996)).

In this new reference frame, the approximation relationship for $\widehat{Q}_{k+1}(q_t)$ is now given by:

$$\widehat{Q}_{k+1}(q_i) = \widehat{Q}_k(q_i) + p_i(q_t) \left[\widehat{Q}_{k+1}(q_t) - \widehat{Q}_k(q_i) \right], \tag{9}$$

wirh $i = 1, 2, \ldots, t-1$.

This approach is interesting because constitutes a kind of minimization method without derivatives (Brent (1996)). Further, it is also usable in non-stationary contexts as the update relationship (9) is based on the current values of state-action pairs.

3. The Reinforcement Learning-based Financial Trading Systems

In this section we use the QLa and the KbRL for developing daily FTSs. First one has to identify the quantities which specify the states, the possible actions of the FTS, and the reward function.

With regards to the states, as at present we are mainly interested in testing the applicability of the considered RL methods to the development of FTSs, we simply use as states the last five percentage returns of the asset to trade, like

in some of the cited literature. For this same reason we do not consider the transaction costs and other frictions. So, given the current price of the asset, p_t, the state of the system at the time step t is given by the vector

$$s_t = (e_{t-4},\, e_{t-3},\, e_{t-2},\, e_{t-1},\, e_t)\,,$$

where $e_\tau = \frac{p_\tau - p_{\tau-1}}{p_{\tau-1}}$. Concerning the possible action of the FTS, we utilize the three following actions:

$$a_t = \begin{cases} -1 & \text{(sell signal)} \\ 0 & \text{(stay-out-from-the-market signal)}, \\ 1 & \text{(buy signal)} \end{cases}$$

in which the stay-out-from-the-market implies the closing of whichever previously open position (if any). Note that in most of the specialized literature, like for instance in Moody and Saffel (2001), only the sell signal and the buy one are considered. Finally, with reference to the reward function, following Moody and Saffel (2001) we take into account the well known Sharpe ratio. In particular, as we wish an indicator that reacts enough quickly to the consequences of the actions of the FTS, we consider the Sharpe ratio calculated only in the last $L = 5$ trading days (a stock market week) and in the last $L = 22$ trading days (a stock market month), that is:

$$r_t = \frac{\mathrm{E}_L\left[g_{t-1}\right]}{\sqrt{\mathrm{Var}_L\left[g_{t.1}\right]}},$$

where $\mathrm{E}_L(\cdot)$ and $\mathrm{Var}_L(\cdot)$ are, respectively, the sample mean operator and the sample variance one, and $g_t = a_{t-1}e_t$ is the gained/lost percentage return obtained at time step t as a consequence of the action taken by the FTS at time step $t-1$.

Now let us pass to the two RL-based approaches we consider: The QLa and the KbRL. With respect the QLa, the kind of linear approximator of the state-action value function we choose is:

$$Q(s_t, a_t; \theta_k) = \theta_{k,0} + \sum_{n=1}^{5} \theta_{k,n} \arctan(s_{t,n}) + \theta_{k,6} \arctan(a_t),$$

in which $\arctan(\cdot)$ plays the role of transformer of the state. Then, the ε-greedy function we follow is:

$$a_t = \begin{cases} \arg\max_{a_t} Q(s_t, a_t; \theta_k) & \text{with probability } 0.95, \\ u & \text{with probability } 0.05 \end{cases}$$

where $u \sim \mathscr{U}_d(-1, 1)$. Concerning the KbRL, we follow Bosq (1996) and Ormonet (2002). Note that, to the best of our knowledge, the use of the arctangent transformation in the QLa and of the KbRL for building FTSs is new.

Summarizing, we consider two different RL-based approaches (QLa and KbRL) and two different values of L (5 and 22), for a total of four configurations.

We apply the above specified RL-based FTSs to two different time series of daily prices: An artificial one and a real one. With reference to the artificial time series, as in Moody and Saffel (2001) we generate log price series as random walks with autoregressive trend processes. The used model is:

$$p_t = \exp\left\{\frac{z_t}{\max z - \min z}\right\},$$

where $z_t = z_{t-1} + \beta_{t-1} + 3a_t$, in which $\beta_t = 0.9\beta_{t-1} + b_t$, $a_t \sim \mathscr{N}(0,1)$, and $b_t \sim \mathscr{N}(0,1)$. The length of the so-generated series is $T = 5000$. This artificial price series shows features which are often present in real financial price series. In particular, it is trending on short time scales and has a high level of noise. As far as the real time series regards, we utilize the closing prices of Banca Intesa (of the Italian stock market), from January 1, 1973 to September 21, 2006. The length of this series is $T = 5400$.

At this point we can present the results of the applications of the various configurations. In all the experimentations we set $\alpha = 0.8$ and $\gamma = 0.7$.

In figure 1 we graphically report the results of the application of the QLa-based FTS to the artificial price series, with $L = 5$. In particular: The first panel shows the price series; The second panel shows the actions taken by the FTS

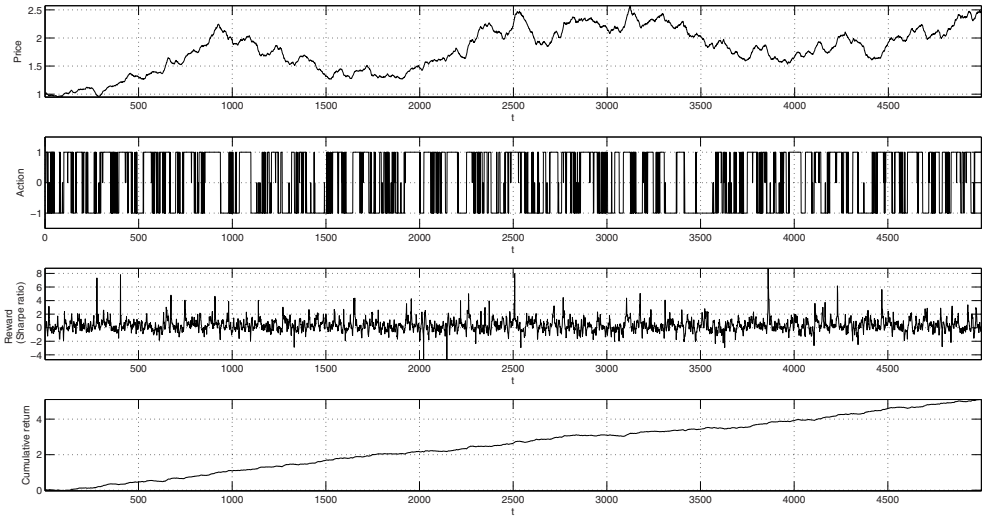

Fig. 1. Results of the QLa-based FTS applied to the artificial price series, with $L = 5$. Final cumulative returns: 510.86%.

at each time step, that is its investment strategy; The third panel shows the rewards, that is the Sharpe ratios, at each time step; The fourth panel shows the cumulative return one should obtain by investing the same monetary amount at each time step. At the end of the trading period, $t = T$, the cumulative return is 510.86%. In figure 2 we graphically report the results of the application of the QLa-based FTS to the real price series, with $L = 5$. At the end of the trading period, $t = T$, the cumulative return is 271.42%.

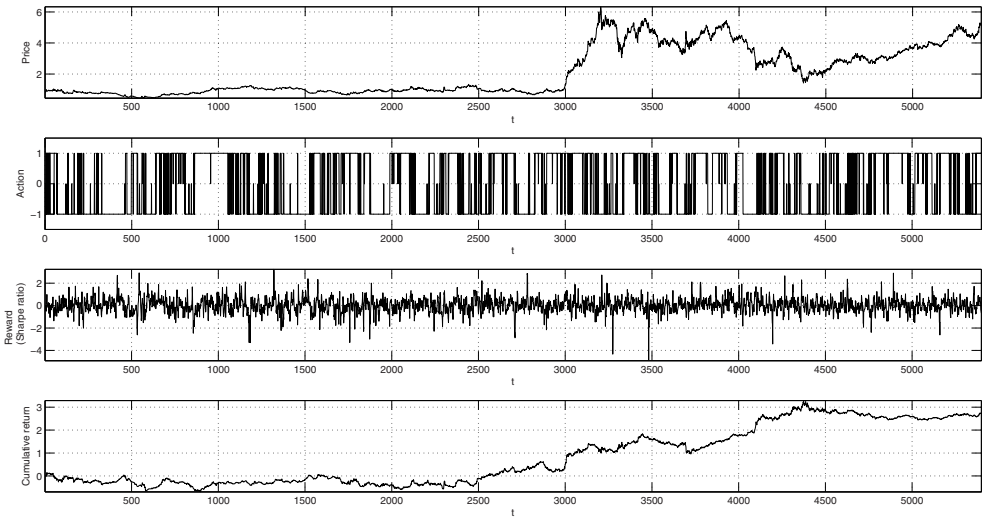

Fig. 2. Results of the QLa-based FTS applied to the real price series, with $L = 5$. Final cumulative returns: 271.42%.

It is very important to note that at the beginning of the trading period, $t = 0$, the vector of the parameters used in the linear approximator, θ_k, is randomly initialized. Because of it, by repeating some times this latter application we observe a certain variability in the final cumulative return: -158.52%, 174.91%, -26.38% …. This shows that the influence of the random initialization heavily spreads overall the trading period instead to soften as time step increases. In figure 3 we graphically report the results associated to the first of such repetition. To check the effects

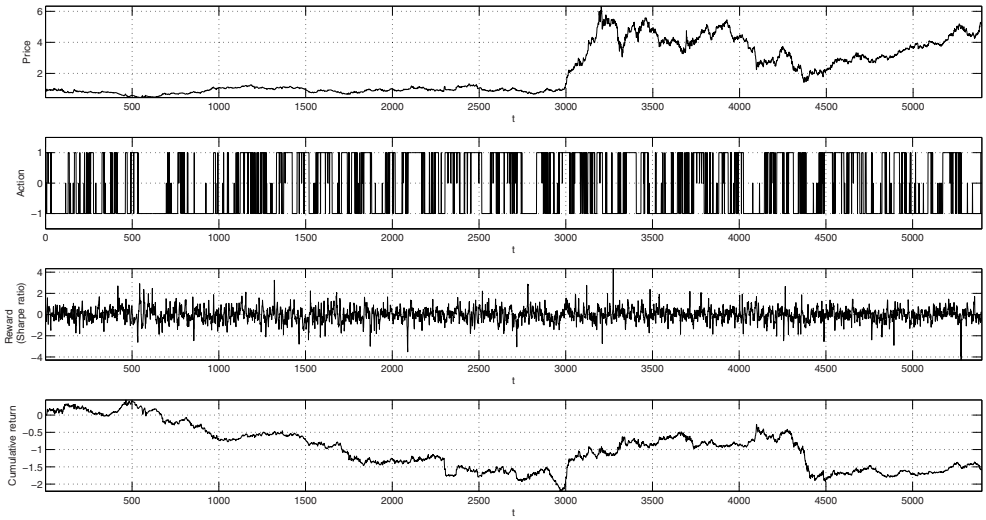

Fig. 3. Results of the QLa-based FTS applied to the real price series, with $L = 5$ (repetition). Final cumulative returns: -158.52%.

of this random initialization, we repeated 1000 times the application of each of the investigated configurations. When the application was to the artificial time series, the series has been taken the same in all the repetition. In table 1 we report some statistics concerning the final cumulative returns.

With reference to the KbRL, we obtain results similar to the one related to the QLa, although a bit less performing. As exemplification, in figure 4 we graphically report the results of the application of the KbRL-based FTS to the real price series, with $L = 22$. At the end of the trading period, $t = T$, the cumulative return is 309.86%. Note that for the KbRL there is not parameter vector to randomly initialize, but at the beginning of the trading period, $t = 0$, it is necessary to randomly initialize a suitable set of state-action pairs (Ormonet (2002)). So, also in this approach one puts the question of the variability of the results. As for the QLa, for the KbRL too we repeated 1000 times the application of each of the investigated configurations (see table 1 for some statistics about the final cumulative returns).

Approach	L	Statistics	Artificial time series	Real time series
QLa	5	μ	472.02%	40.00%
		σ	32.79%	139.09%
		Confidence interval	[407.76%, 536.29%]	[−226.73%, 306.73%]
KebRL	5	μ	435.42%	−7.99%
		σ	41.13%	131.26%
		Confidence interval	[354.81%, 516.02%]	[−265.26%, 249.28%]
QLa	22	μ	337.68%	92.92%
		σ	40.80%	149.28%
		Confidence interval	[257.71%, 417.64%]	[−199.66%, 385.50%]
KbRL	22	μ	216.61%	13.20%
		σ	49.98%	153.34%
		Confidence interval	[118.64%, 314.58%]	[−287.34%, 313.74%]

Table 1. Statistics about the final cumulative returns.

The main facts detectable from table 1 are the following ones:

- Given the values of the means, most of the investigated configurations appears to be profitable. Further, the QLa

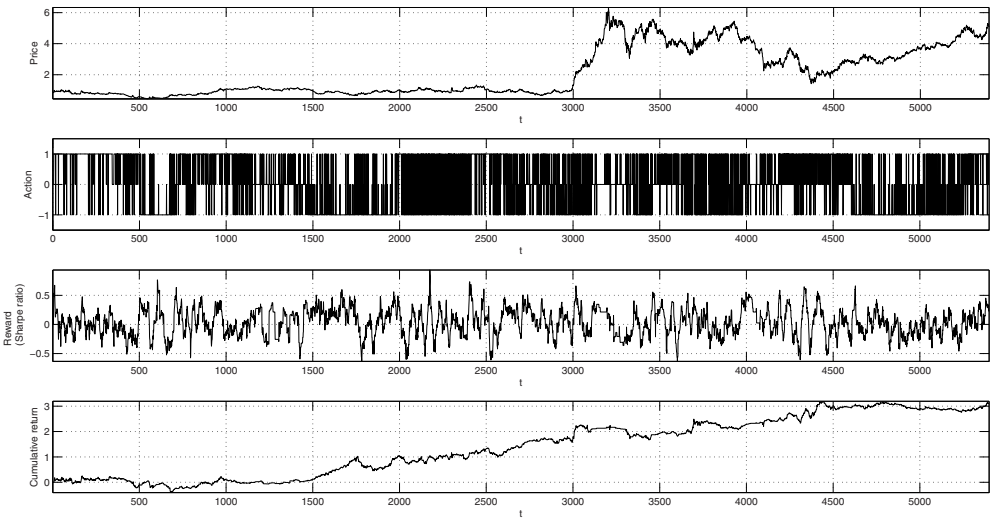

Fig. 4. Results of the KbRL FTS applied to the real price series, with $L = 22$. Final cumulative returns: 309.86%.

approach generally seems more performing than the KbRL one;

- Given the values of the standard deviations, the results of all the considered configurations are characterized by a certain level of variability. In particular, such values emphasize that the question of the influence of the random initialization on the results is mainly true for the real financial time series;
- Given the results, it appears that the value of L has a significant impact on the performances of the FTSs.

4. Some concluding remarks

In this paper we have developed and applied some original automatic FTSs based on differently configured RL algorithms. Here we have presented the results coming out from the current phase of our research on this topic. Of course, many questions have again to be explored. In particular:

- The choice of the last five percentage returns as states is a naive choice. Now we are beginning to work to specify some new indicators to use as states (in the first experimentations they have provided interesting results);
- As known, the Sharpe ratio as performance measure suffers several limits. Currently, as reward function we are considering alternative and more realistic performance measures;
- The management of the learning rate, α, we have used here is appropriate for stationary systems. But generally financial markets are non-stationary. Because of that, we are beginning to work to develop methods for the dynamic management of the learning rate in non-stationary contexts;
- In order to deepen the valuation about the capabilities of our FTSs, we wish to apply them to more and more financial price series coming from different markets.

Acknowledgements

The authors wish to thank the Department of Economics of the Ca' Foscari University of Venice for the support received within the research project *Machine Learning adattativo per la gestione dinamica di portafogli finanziari* [*Adaptative Machine Learning for the dynamic management of financial portfolios*]).

References

Barto, A. G. and Sutton, R. (1996). Reinforcement Learning: An introduction. *Adaptive Computation and Machine Learning*.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic programming*. Belmont: Athena Scientific.

Bosq, D. (1996). *Algorithms for minimization without derivatives*. Nonparametric statistics for stochastic processes. Estimation and prediction.Lecture notes in statistics. New York: Springer-Verlag.

Brent, R. P. (1996). *Algorithms for minimization without derivatives*. Englewood Cliffs: Prentice-Hall.

Gold, C. (2003). FX trading via recurrent Reinforcement Learning. *Proc of IEEE Intl Conf on Computat Intel in Financial Engin*, **1**(1), 363–370.

Moody, J. and Saffel, M. (2001). Learning to trade via direct reinforcement. *IEEE Trans on Neural Net*, **4**(8), 75–89.

Moody, J., Wu, L., Liao, Y., and Saffel, M. (1998). Performance functions and Reinforcement Learning for trading systems and portfolios. *J Forecast*, **17**(1), 441–470.

Ormonet, D. (2002). Kernel-based Reinforcement Learning. *Mach Learn*, **49**(1), 61–78.

Smart, W. D. and Kaelbling, L. P. (2000). Practical Reinforcement Learning in continuous spaces. *Proc. 17th Intl Conf on Mach Learn*, **9**(1), 3–10.