Ca' Foscari
University
of Venice

Department
of Economics

Working Paper

Marco Corazza
and
Andrea Sangalli

Q-Learning and SARSA: a comparison between two intelligent stochastic control approaches for financial trading

ISSN: 1827-3580
No. 15/WP/2015

Working Papers
Department of Economics
Ca' Foscari University of Venice
No. 15/WP/2015
ISSN 1827-3580

Q-Learning and SARSA: a comparison between two intelligent stochastic control approaches for financial trading

Marco Corazza
Department of Economics
Ca' Foscari University of Venice

and

Andrea Sangalli

Abstract
The purpose of this paper is to solve a stochastic control problem consisting of optimizing the management of a trading system. Two model free machine learning algorithms based on Reinforcement Learning method are compared: the Q-Learning and the SARSA ones. Both these models optimize their behaviours in real time on the basis of the reactions they get from the environment in which operate. This idea is based on a new emerging theory about the market efficiency, the Adaptive Market Hypothesis. We apply the algorithms on single stock price time series using simple state variables. These algorithms operate selecting an action among three possible ones: buy, sell and stay out from the market. We perform several applications based on different parameter settings that are tested on an artificial daily stock prices time series and on different real ones from Italian stock market. Furthermore, performances are both gross and net of transaction costs.

Keywords: Financial trading system, Adaptive Market Hypothesis, model free machine learning, Reinforcement Learning, Q-Learning, SARSA, Italian stock market.

JEL Codes: C61, C63, G11.

Address for correspondence:
Marco Corazza
Department of Economics
Ca' Foscari University of Venice
Cannaregio 873, Fondamenta S.Giobbe
30121 Venezia – Italy
Phone: (++39) 041 2346921
Fax: (++39) 041 2347444
E-mail: corazza@unive.it

The Working Paper Series
is available only on line
(http://www.unive.it/nqcontent.cfm?a_id=86302)
For editorial correspondence, please contact:
wp.dse@unive.it

Department of Economics
Ca' Foscari University of Venice
Cannaregio 873, Fondamenta San Giobbe
30121 Venice Italy
Fax: ++39 041 2349210

Q-Learning and SARSA: a comparison between two intelligent stochastic control approaches for financial trading

Marco Corazza[1] and Andrea Sangalli[2]

[1] Department of Economics – Ca' Foscari University of Venice
Sestiere Cannaregio 873 – 30121 Venice, Italy
corazza@unive.it

[2] **andrea.sangalli.90@gmail.com**

Abstract. The purpose of this paper is to solve a stochastic control problem consisting of optimizing the management of a trading system. Two model free machine learning algorithms based on Reinforcement Learning method are compared: the Q-Learning and the SARSA ones. Both these models optimize their behaviours in real time on the basis of the reactions they get from the environment in which operate. This idea is based on a new emerging theory about the market efficiency, the Adaptive Market Hypothesis. We apply the algorithms on single stock price time series using simple state variables. These algorithms operate selecting an action among three possible ones: buy, sell and stay out from the market. We perform several applications based on different parameter settings that are tested on an artificial daily stock prices time series and on different real ones from Italian stock market. Furthermore, performances are both gross and net of transaction costs.

Keywords. Financial trading system, Adaptive Market Hypothesis, model free machine learning, Reinforcement Learning, Q-Learning, SARSA, Italian stock market.

1 Introduction

The more entrenched theory related to the financial markets, the *Efficient Market Hypothesis* (*EMH*), asserts that market prices instantly fully reflect all available information. According to this theory, the asset price movement can be described as a stochastic process where the price is conditioned by the coming of new information. The formalization is:

$$P_{t+1} = \mathbb{E}\left(\tilde{P}_{t+1} \middle| \Omega_t\right) + \tilde{\varepsilon}_{t+1}$$

where t and $t+1$ indicate two consecutive time instants, P_{t+1} is the financial asset price at time $t+1$, $\mathbb{E}(\cdot)$ is the expectation operator, $\tilde{P}_{t+1}$ is the random variable "financial asset price" at time $t+1$, Ω_t is the set of all available information at time t, and $\tilde{\varepsilon}_{t+1}$ is the financial asset prediction error random variable at time $t+1$, with $\mathbb{E}(\tilde{\varepsilon}_{t+1}) = 0$. This last condition implies it is not possible to systematically obtain gains exploiting adjustment prices movement errors. Therefore in an efficient market that fully reflects all available information, price changes are completely random and

unpredictable. This is due to many active financial agents that, working in a fully rational way, attempt to gain profits from their information. By doing so, they incorporate their information into market prices and so profit opportunities, that justify their trades, are quickly eliminated. This happens instantaneously, no profits can be garnered because such profits are already captured. So, unless in the exact moment and for pure chance an agent is taking a long or short position, it is impossible to catch systematic gains, and collection of information would be only an expensive activity that would constitute only a sunk cost. Then, if the *EMH* holds, there would be no reason to trade.

As an alternative, an inefficient market degree may push the investors to spend resources to gather and to trade information. Indeed, common feeling leads to think that markets are not perfectly efficient due to the economic agent behaviors. Human agents' decision can be effected by several qualities/weaknesses behaviour distortions, i.e. overconfidence, risk aversion, herding, miscalibration of probabilities, presumption, psychological account, overreaction, regret and assessment errors. Especially when agents project themselves into the future and therefore in uncertainty situations, they do not operate in a fully rational way. So sometimes, economic agents show ruinous predictive and financial attitudes, which probably would make inefficient markets, giving rise to the possibility to trade effectively, exactly because it is not true that as soon as new information arrives the prices adapt themselves instantly or correctly.

This social background suggests the possibility of an alternative theory based on a bio-inspired evolutionary approach of economic interactions. It argues that the environment in which the economic agents acts is in incessant evolution and financial operators interact with each other within a continuous changing structure.

So, it is possible reconcile the *EMH* with all previous behaviors leading to a new principle: the *Adaptive Market Hypothesis* (*AMH*). Its main assumptions are (see [5] and [4]): equity risk premium is not constant over the time, but it varies according to market trends and investors population; asset allocation can add value by exploiting market trends and by behavioral systematic changes; all investment products tend to experience high and low performance cycles; market efficiency and risk preferences vary continuously over time and from market to market. The natural selection laws determine the market evolution and the financial market participants to identify the successful trader typical profile. Unsuccessful traders are eventually eliminated from the population if they scored a certain level of losses. Therefore, the efficiency degree

of financial markets depends on factors like competitors number their adaptation ability and the available profit opportunities. Since the environment is constantly changing, it is necessary that agents possess the ability to adapt themselves to new situations by learning and by developing new profitable behaviors.

Generally, human agents are not able to achieve the optimal (expected) rationality considered in the *EMH*. Rather than make an effort in an optimization, they prefer achieving a certain satisfaction degree. In this way, individuals perform actions not necessarily optimal, but simply satisfactory, acting in a not analytic way but through trial and error and then by a natural selection-based approach.

Individuals make their own choices based on experience and they formulate their best guesses about may be optimal. In this way, agents develop some heuristic to solve various economic challenges, which, if they remain stable, allow them to adapt their behavior in order to achieve optimal solutions. However, if environment changes, the old behavior should adapt itself to new conditions. In such cases, it is normal to observe temporarily wrong, suboptimal and irrational actions. So, *AMH* may be view as an extension of the *EMH*, where agents act in their own interests making often mistakes, but they can learn and they can adapt their behaviors.

Therefore, due to this complexity of the financial markets, it is necessary to "create" an intelligent agent for performing profitable trading. In this paper, such an intelligent agent, i.e. the trading system, is implemented through the branch of artificial intelligence known as *Reinforcement Learning* (*RL*). The underlying philosophy of *RL* is: the agent learns what to do in a given situations directly interacting with the environment in which the agent is (see [4]). So, the trading system is created in order to be able to choose if to buy/to sell a given financial asset, or to stay out from the market (see [2], [1] and [7] – the Master's degree thesis of the second author –).

In particular, in this paper two methods belonging to *RL* are compared, the *Q-Learning algorithm* (*QLa*) and the *SARSA algorithm* (*SARSAa*), in order to test if *QLa* can perform better than *SARSAa* given the same parameter configurations. It is noteworthy that, due to the complexity of the process that drive the evolution of the stock prices, we have not an exact model of the environment. Nevertheless, the algorithms considered in this paper are suitable to deal with this problem as they are model free approaches.

The remainder of the paper is organized as follows. In the next section we recall some basics of *RL*, *QLa* and *SARSAa*. In Section 3 we describe the features of

the implemented algorithms and the stock prices time series (an artificial daily stock prices time series and seven real ones from the Italian stock market). In Section 4 we present the considered parameters settings. In Section 5 we describe how to make operative the algorithms. In Section 6 the obtained results are compared and explained. In Section 5 we conclude with some final remarks.

2 Basics of *Reinforcement Learning*

An agent developed on the basis of *RL* is constituted by four subelements: a policy, a reward function, a value function and, optionally, a model of the environment (see [4]).

A policy defines the agent's behavior at each time instant, i.e. it is a mapping from the set of perceived environmental states to the set of the actions to be taken when the agent finds itself in those states.

A reward function represents the agent's goal, i.e. it translates each perceived states-action pair into a single number, i.e. a reward indicates the intrinsic desirability of that state-action pair. It is the way to communicate to the agent about what is desirable (but not how reach it).

The agent's objective is the maximization of the sum of the total rewards in the long run, which is specified by a value function. It computes the value of a state-action pair as the total amount of rewards that the agent can expect to accumulate over the future, starting from the state in which it is. So, the agent selects the action based on valued judgments. Indeed, while the reward determines the immediate, intrinsic desirability of a state-action pair, the value indicates the long term desirability of a state-action pair considering the probable future state-action pairs and their rewards.

A model of the environment has the purpose to reproduce the behavior of the environment, i.e. the model which directs the agent in the next state and to the next reward on the basis of the current state-action pair. The model of the environment is not always available as, for example, in financial contexts, especially for financial trading purposes.

Therefore, in our algorithms, at each step of discrete time, $t = 0, 1, 2, 3, \dots, N$, the agent and the environment interact between them. The former receives from the latter a representation of it at time t, $s_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of available states, that summarizes all information concerning the environment. On basis of this, an action $a_t \in \mathcal{A}(s_t)$ is selected by the agent, where $\mathcal{A}(s_t)$ is the set of available actions in the

state s_t. In the next step, $t+1$, the agent is in a new state s_{t+1} and receives a reward $r_{t+1} \in \mathcal{R}$ as response to the action a_t.

Since in *RL* the agent's goal is the maximization of the value function, at time t the agent selects the action that maximizes the expected value of the total of rewards that it can obtain in the future from the state in which the agent is. The expected return, R_t, is generally defined as a function of the current reward and of the discounted future ones:

$$R_t = \sum_{k=0}^{+\infty} \gamma^k r_{t+k+1}$$

where $\gamma \in [0,1]$ is the discount rate. Note that "$+\infty$" can be replaced with a finite value T.

Therefore, the agent's target consists in learning a policy able to maximize the rewards in the long run interacting with the environment on the basis of own experience. To do this, in each step t, the agent, starting from the state s_t, has to calculate as follows the value of the action-value function $Q^\pi(s,a)$ for each possible action on the basis of the policy π:

$$Q^\pi(s,a) = \mathbb{E}_\pi\{R_t | s_t = s, a_t = a\}.$$

A fundamental property of the action-value function is the fulfillment of the following special recursive relationship, known as Bellman equation for $Q^\pi(s,a)$:

$$Q^\pi(s,a) = \mathbb{E}_\pi\{r_{t+1} + \gamma Q^\pi(s_{t+1}, a_{t+1}) | s_t = s, a_t = a\}.$$

This Bellman equation has a unique solution.

To learn a policy able to maximize the rewards in the long run, the agent has to select the action that fulfils the following relationship, called greedy policy:

$$Q^*(s,a) = \max_\pi Q^\pi(s,a).$$

Since $Q^*(s,a)$ is a value function under a policy π, it satisfies the Bellman equation, and it is called Bellman optimality equation. It expresses that the value of a state

following an optimal policy must be equal to the expected return for the best action selected in that state, that is $Q^*(s,a) = \max_{a \in \mathcal{A}(s)} Q^\pi(s,a)$.

It is possible to compute the action-value function $Q^\pi(s,a)$ through an iterative method. It is called policy evaluation and it starts with an arbitrary initialization of $Q_0^\pi(s,a)$. In each step of the computation, the action-value function is approximated using the Bellman equation for $Q^\pi(s,a)$ as an update rule:

$$Q_{k+1}^\pi(s,a) = \mathbb{E}_\pi\{r_{t+1} + \gamma Q_k^\pi(s_{t+1}, a'_{t+1}) | s_t = s, a_t = a\}.$$

Furthermore, in each state it is useful to verify if there exists another policy that, if followed, is able to perform better than the currently followed one. This process is known as policy improvement. The action which appears the best according to the highest $Q^\pi(s,a)$,is selected.

2.1 *Q-Learning* and *SARSA* algorithms

The two implemented algorithms, *QLa* and *SARSAa*, learn the value of the optimal state-action pairs, $Q^*(s,a)$, through the transitions from state-action pair to state-action pair. These methods are both online control algorithms. They are online algorithms because they perform the updates of the action-value function estimate at the end of each step without waiting the terminal one. So, in both methods the updated estimate of $Q^*(s,a)$ is already available to be used in the next state. They are control algorithms because they perform actions to reach their purpose, that is the estimate of the optimal state-action value function, $Q^*(s,a)$.

The difference among the two algorithms is about the evaluation of the policy, π.

QLa is an off-policy algorithm, i.e. the agent learns the value of the state-action pair independently from the performed action, because their updates are done regardless from the current action, but with respect to the action that maximizes the value of the next state-action pair.

QLa structure, for each single step:

$$Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q_k(s_{t+1}, a_{t+1}) - Q_k(s_t, a_t) \right]$$

where $\alpha \in (0,1]$ is the so-called learning rate. It determines the degree of the update.

SARSAa is an on-policy algorithm, i.e. the agent learns the value of the state-action pair on the basis of the performed action. In this way the current policy is evaluated, unlike *QLa* that performs one policy and evaluates another one.
SARSAa updating rule is the following:

$$Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha[r_{t+1} + \gamma Q_k(s_{t+1}, a_{t+1}) - Q_k(s_t, a_t)]$$

where the action a_{t+1} is the action carried out in the state s_{t+1} on the basis of the current policy.

3 Features of the algorithms

3.1 State descriptors

We are interested in checking the performance capabilities of the trading system starting from basic information. So, as environmental information we consider the last four logarithmic returns of the investigated financial asset and the last performed action. Note that the states taken into account are both continuous (the logarithmic returns) and discrete (the action). So at time t, the environmental state can be expressed by the following vector:

$$\vec{s}_t = [e_{t-N+1} \quad e_{t-N+2} \quad \cdots \quad e_t \quad a_{t-1}]$$

where $e_\tau = \ln(p_\tau / p_{\tau-1})$, in which p_τ is the stock price at time τ, and a_{t-1} is the last performed action.

Note that the tests are carried out using $N = 5$ states.

3.2 Actions

In each time t, the possible actions are the following (see [2] and [1]):

$$a_t = \begin{cases} -1: buy\ or\ stay\ long\ in\ the\ market \\ 0: stay\ out\ from\ the\_market \\ +1: sell\ or\ stay\ short\ in\ the\ market \end{cases}$$

Through the action a_t, a trade is possible only at the end of each step t and then it is evaluated in the next one by the received reward r_{t+1}.

The action selected in such a way is not always undertaken. In fact, in each time t and with probability $1-\varepsilon$, the action is randomly selected on the basis of the greedy policy. This method is called ε-greedy policy and it is expressed as follows:

$$a_t = \begin{cases} \arg\max_{a_t} Q(s_t, a_t) & with\ probability\ 1-\varepsilon \\ u_t & with\ probability\ \varepsilon \end{cases}$$

where $\varepsilon \in \{2.5\%, 5.0\%, 10.0\%\}$ and $u_t \sim \mathcal{A}_d\{buy, out, sell\}$.
Parameter ε decides the frequency of the exploratory actions, which are selected by a uniform distribution. Note that this may regress the current performance, because the agent may choose disadvantageous actions, but also the agent may find actions can achieve better results. However without such trials, it is impossible to find possible performance improvements especially if the environment is non-stationary.

3.3 Reward function

The reward is a numeric representation of the agent's satisfaction. Here, the Sharpe ratio is used as reward function similarly to what done in [2], [1] and [7].
Sharpe ratio formula is the following:

$$SR_{t+1} = \frac{\mathbb{E}_L(g_{t+1})}{\sqrt{\mathbb{V}ar_L(g_{t+1})}} \in \mathbb{R}$$

where SR_{t+1} is the Sharpe ratio at time $t+1$, $\mathbb{E}_L(\cdot)$ and $\mathbb{V}ar_L(\cdot)$ are the sample mean and the sample variance calculated on the obtained returns g_t of the last L days, respectively., with L = 5, 22.

3.4 Squashing function

The squashing function transforms the input state descriptors in order to obtain values that may increase the sensitivity of the process of calculation of the action-value function $Q(s_t, a_t)$. This is similar to a clustering process that gathers the very near continuous variables in clusters.

The squashing function we consider is the logistic function, that is:

$$\phi(x) = \frac{a}{1 + be^{-cx}} - d.$$

It is set with parameter values that make it similar to the hyperbolic tangent function which has been rarely (if not) used and checked in this context. In particular, the parameters values are $a = 2$, $b = 1$, $c = 10^{15}$ and $d = -1$.
It is noteworthy that $c = 10^{15}$ makes $\phi(x)$ comparable to a "step". This increases the sensitivity of the agent. Indeed, very close values will be distant each from the other after the transformation. In this way, the differentiation between the considered state descriptors is amplified.

3.5 Transaction costs

Percentual transaction costs are applied on each action that opens or closes a position. The amount of the transaction costs is half divided and it is computed both in the opening and in the closing of each position. In this way, the result on the performance does not change compared to the case where they were applied only to the opening positions, as in [3]. Therefore, the transaction cost, δ, is defined as follows:

$$\delta = \frac{transaction\ cost}{2}.$$

Its amount is equal to 0.19%. This value is based on the current transaction costs applied by several Italian brokerage companies.
The percentage transaction cost is applied in the following way:

$$g_{t+1} = g_{t+1}^{gross} - \delta$$

where g_{t+1}^{gross} is the return related to a given action a_t.

3.6 Action-value function

The action-value function $Q(s, a)$ is approximated by a parameterized functional form with a parameters vector $\vec{\theta}_k$. So, $Q(s, a)$ is now indicated with $Q(s, a; \vec{\theta}_k)$ (see [6]). Its functional form is the follows:

$$Q(s_t, a_t; \vec{\theta}_k) = \theta_{k,0} + \sum_{n=1}^{N-1} \theta_{k,n} \phi(s_{t,n}) + \phi_{k,N}\, \phi(a_{p,t})$$

where $a_{p,t}$ is the action to carry out in state s_t.

The parameter vector $\vec{\theta}_k$ may vary step by step in order to determine the optimal parameter vector $\vec{\theta}^*$ that minimizes the mean square error between the unknown optimal action-value function $Q^*(s,a)$ and its estimation $Q(s_t, a_t; \vec{\theta}_k)$, i.e.:

$$\min_{\vec{\theta}_k} \sum_{s,a} \left[Q^*(s,a) - Q(s_t, a_t; \vec{\theta}_k) \right]^2.$$

The updating rule of the parameters vector $\vec{\theta}_k$ is the gradient descent method, widely used in the approximation processes of functions to find minimizer vectors. This method is based on an iterative algorithm; the parameters vector $\vec{\theta}_k$ is adjusted by a small value in the direction that reduce the error. In particular:

- for *QLa*, the parameters vector updating rule is:

$$\vec{\theta}_{k+1} = \vec{\theta}_k + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}; \vec{\theta}_k) - Q(s_t, a_t; \vec{\theta}_k) \right] \nabla_{\vec{\theta}_k} Q(s_t, a_t; \vec{\theta}_k);$$

- for *SARSAa*, the parameters vector updating rule is:

$$\vec{\theta}_{k+1} = \vec{\theta}_k + \alpha [r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}; \vec{\theta}_k) - Q(s_t, a_t; \vec{\theta}_k)] \nabla_{\vec{\theta}_k} Q(s_t, a_t; \vec{\theta}_k).$$

3.7 Stock prices time series

Tests are carried out on six daily stock prices time series plus an artificial one. The real ones are downloaded by Bloomberg® database. They are Assicurazioni Generali S.p.A., Fiat S.p.A., Pirelli & C. S.p.A., Saipem S.p.A., Telecom Italia S.p.A. and Unicredit S.p.A., from January 2, 1985 to September 30, 2014, i.e. 7520 days, about 30 stock market years.

The artificial one is generated through a GARCH process. It is specified on the basis of the following form (see [2], [1] and the references therein):

$$p_\tau = \exp\left\{\frac{z_\tau}{\text{maxz} - \text{minz}}\right\}$$

where $z_\tau = z_{\tau-1} + \beta_{\tau-1} + 3a_\tau$, $\beta_\tau = 0.9\beta_{\tau-1} + b_t$ with $a_t \sim \mathcal{N}(0,1)$ and $b_t \sim \mathcal{N}(0,1)$.

This process shows classic features of stock prices time series, such as non-constant volatility, short term different trends and high level of noise. The length of this time series is equal to the length of the real ones.

4 Simulations

500 simulations are simultaneously carried out for each different setting.

In all tests and for both methods, we set $\alpha = 5\%$ and $\gamma = 95\%$. These values are those usually considered in the literature. Indeed, the former is sufficiently small to ensure the convergence on average to the true value $Q^*(s,a)$ and it is not too small to make the convergence too slow. The latter is a reasonable value both to make finite the expected reward.

For each simulation, the parameters vector $\vec{\theta}_k$ is initialized on the basis of an uniform probability distribution included in the range $[-1,1]$.

The starting capital is €5000.

Summarizing, we consider two values of N (1 and 5), one type of reward function (SR), two values of L (5 and 22) and three values of ε (2.5%, 5.0% and 10.0%) for a total of twelve parameters configurations.

5 Operational signals

Every configuration is simultaneously simulated 500 times because the final results are different due to the stochastic inizialization of the parameters vector $\vec{\theta}_k$. So, at each time step t, there are 500 actions that may be different. To make operational the algorithm, in order to obtain at each time step t only an action to perform, such an action may be obtained on the basis of the majority of the 500 actions in each step

(see [8]). To do this, firs the average action is calculated on the basis of the average of the actions of all the simulations in each respective step t, i.e.:

$$\bar{a}_t = \frac{\sum_{j=1}^{J} a_{t,j}}{J}$$

where $\bar{a}_t$ is the average action value at the step t, $a_{t,j}$ is the action signal of the j-th simulation, and $J = 500$ is the number of simulations.
Second, the average action, $\bar{a}_t$, is translated in an operational action signal through the following rule, similarly to what done in [8]:

- $if\ \bar{a}_t \in [-1, -0.\bar{3}) \Rightarrow sell\ or\ stay\_short\_in\_the\_market$
- $if\ \bar{a}_t \in [-0.\bar{3}, +0.\bar{3}] \Rightarrow stay\_out\_from\_the\_market$
- $if\ \bar{a}_t \in (+0.\bar{3}, +1] \Rightarrow buy\ or\ stay\_long\_in\_the\_market$

In this way, the general trend of all the generated equity lines by all the simulations is followed in mean.

6 Results

We present the results of the applications of several tests[1]. In Figure 1 and in Figure 2 we show as example the plot of an output of *QLa* and of an output of a *SARSAa* applied on Telecom Italia S.p.A. daily stock prices time series, respectively. Each figure is divided in three panels. The first one shows the daily stock prices time series; the second one shows the operational action signals during the trading period; the third panel shows both the gross equity line (blue line) and the net equity line (green line) one should obtain by investing at time $t = 0$ a starting capital $C_0 = €5000$. In this cases, the parameters configuration is: $N = 5$, $L = 22$ and $\varepsilon = 10.0\%$.

[1] The results are calculated by a parallelized code implemented in Matlab® environment.

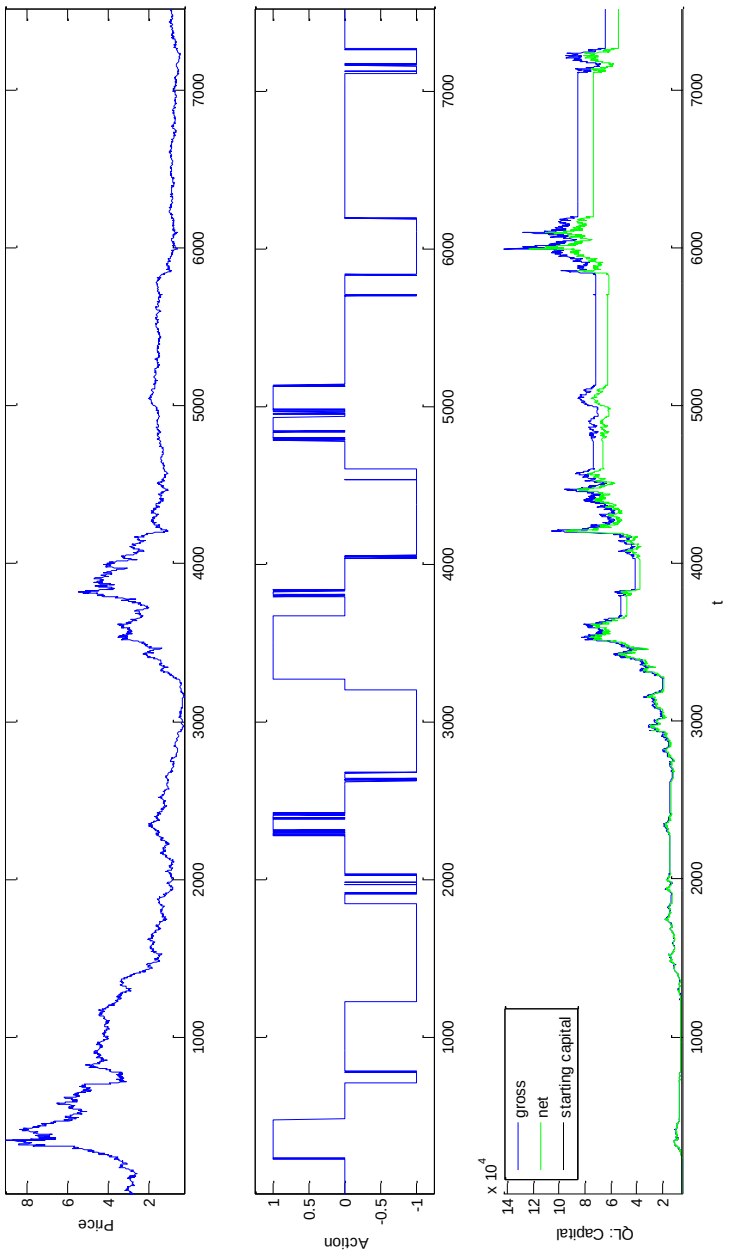

Figure 1. Results the *QLa* applied to Telecom Italia S.p.A. daily stock prices time series.

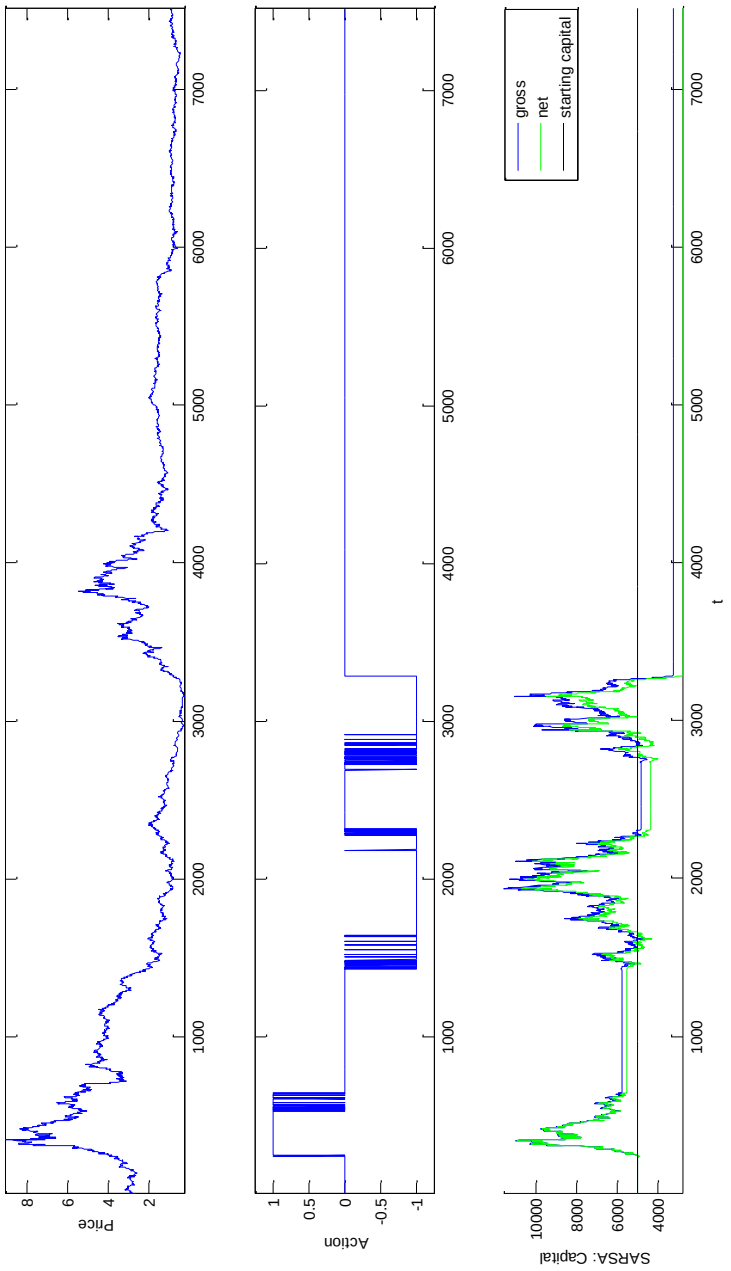

Figure 2. Results of *SARSAa* applied to Telecom Italia S.p.A. daily stock prices time series.

In the following tables we report some statistics concerning the results. The columns labeled "g[*%*]" shows the net average annual logarithmic return obtained during the trading period; the column labeled "*%*" shows the percentage of times in which the net equity line is greater than or equal to the starting capital during the trading period; the column labeled "#" shows the average annual number of transactions.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
QLa	**5**	**2.5%**	3.64	99.20	0.54	2.91	99.88	0.74
	5	**5.0%**	3.30	100.00	4.09	2.22	99.77	4.50
	5	**10.0%**	-1.97	54.04	10.19	1.62	99.33	7.78
	22	**2.5%**	-0.17	92.84	5.00	-0.57	82.33	4.09
	22	**5.0%**	2.89	99.88	5.13	1.54	100.00	5.47
	22	**10.0%**	3.38	99.84	7.81	3.71	100.00	6.21

Table 1. *Q-Learning*: Assicurazioni Generali S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
SARSAa	**5**	**2.5%**	3.80	99.27	0.67	3.31	99.88	0.60
	5	**5.0%**	3.88	100.00	0.34	3.40	99.65	0.47
	5	**10.0%**	2.92	99.96	1.48	2.07	99.32	2.01
	22	**2.5%**	-0.08	58.37	13.01	0.89	99.75	7.85
	22	**5.0%**	1.58	99.88	8.72	1.47	99.52	0.94
	22	**10.0%**	1.65	100.00	1.68	0.85	99.88	9.19

Table 2. *SARSA*: Assicurazioni Generali S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
QLa	**5**	**2.5%**	3.14	99.87	1.68	3.93	99.85	1.41
	5	**5.0%**	1.74	99.36	7.24	6.09	99.75	4.03
	5	**10.0%**	-2.00	46.82	9.69	1.86	80.64	10.80
	22	**2.5%**	-1.92	82.01	5.40	2.07	99.85	3.96
	22	**5.0%**	1.76	99.83	4.93	-0.37	89.34	7.41
	22	**10.0%**	3.46	100.00	8.82	0.36	72.64	10.84

Table 3. *Q-Learning*: Fiat S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	ε	g[%]	%	#	g[%]	%	#
SARSAa	5	2.5%	3.01	99.80	1.88	3.78	99.93	0.20
	5	5.0%	3.04	99.55	0.60	3.74	99.75	0.40
	5	10.0%	3.07	99.65	0.87	3.32	99.97	0.07
	22	2.5%	1.74	99.97	1.07	1.28	100.00	1.95
	22	5.0%	0.87	99.29	1.07	1.90	99.84	3.56
	22	10.0%	-2.18	76.52	27.53	-2.36	78.74	9.90

Table 4. *SARSA*: Fiat S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	ε	g[%]	%	#	g[%]	%	#
QLa	5	2.5%	1.78	99.91	1.01	-0.34	1.36	2.21
	5	5.0%	1.43	98.60	6.97	0.18	39.80	7.25
	5	10.0%	3.34	98.82	11.13	-0.36	15.81	11.61
	22	2.5%	-4.70	52.18	7.54	-3.84	20.55	2.85
	22	5.0%	-1.40	41.02	4.59	-4.51	20.22	7.62
	22	10.0%	3.81	98.16	8.32	0.62	69.34	7.31

Table 5. *Q-Learning*: Pirelli & C. S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	ε	g[%]	%	#	g[%]	%	#
SARSAa	5	2.5%	2.02	99.91	0.27	-0.34	1.36	1.34
	5	5.0%	1.31	99.04	0.94	-1.24	0.45	1.48
	5	10.0%	0.99	98.82	0.94	-0.77	0.37	2.01
	22	2.5%	1.43	98.76	0.54	0.76	99.79	0.34
	22	5.0%	1.24	98.83	0.27	0.38	98.90	0.47
	22	10.0%	-0.13	17.34	4.02	0.38	99.61	0.94

Table 6. *SARSA*: Pirelli & C. S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	ε	g[%]	%	#	g[%]	%	#
QLa	5	2.5%	-0.13	91.38	4.63	1.02	100.00	1.54
	5	5.0%	0.48	64.13	6.64	-2.34	54.54	5.23
	5	10.0%	-0.96	49.81	7.85	-3.29	20.50	7.31
	22	2.5%	3.18	99.99	1.14	3.58	97.04	2.01
	22	5.0%	0.11	58.54	7.58	0.80	52.84	4.43
	22	10.0%	3.75	67.78	8.92	3.77	96.53	8.19

Table 7. *Q-Learning*: Saipem S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
SARSAa	**5**	**2.5%**	1.33	97.61	1.61	1.51	100.00	0.13
	5	**5.0%**	1.29	95.10	1.34	1.39	94.97	0.54
	5	**10.0%**	0.53	93.08	1.61	1.13	99.93	1.34
	22	**2.5%**	1.01	99.95	0.67	0.85	99.37	0.47
	22	**5.0%**	-2.86	88.08	14.95	-3.05	22.98	6.88
	22	**10.0%**	-2.36	25.82	12.44	-1.96	28.49	17.68

Table 8. *SARSA*: Saipem S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
QLa	**5**	**2.5%**	0.85	99.87	2.95	1.09	99.79	0.20
	5	**5.0%**	5.48	99.40	6.64	-0.16	96.67	8.39
	5	**10.0%**	5.88	99.91	8.58	5.65	99.81	9.33
	22	**2.5%**	1.23	100.00	5.70	-4.10	52.70	6.51
	22	**5.0%**	5.80	99.95	5.50	8.14	99.89	2.48
	22	**10.0%**	6.96	99.35	7.91	8.34	99.89	5.84

Table 9. *Q-Learning*: Telecom Italia S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
SARSAa	**5**	**2.5%**	0.67	99.55	0.67	1.32	99.93	0.07
	5	**5.0%**	1.35	100.00	0.13	0.76	99.59	0.54
	5	**10.0%**	-2.21	40.85	14.95	0.41	100.00	0.67
	22	**2.5%**	-1.47	23.12	13.68	-0.21	9.33	0.81
	22	**5.0%**	-0.69	37.57	5.63	-1.44	28.00	5.64
	22	**10.0%**	-0.60	42.99	3.69	-1.91	34.23	5.43

Table 10. *SARSA*: Telecom Italia S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
QLa	**5**	**2.5%**	0.69	99.27	0.20	0.59	98.94	0.13
	5	**5.0%**	-2.62	21.31	9.52	-4.73	11.40	14.49
	5	**10.0%**	0.67	46.65	9.35	1.48	46.13	10.63
	22	**2.5%**	-1.05	91.63	3.15	0.56	74.80	3.42
	22	**5.0%**	1.18	56.96	7.28	1.00	44.17	7.35
	22	**10.0%**	6.56	99.32	7.28	5.42	84.52	6.41

Table 11. *Q-Learning*: Unicredit S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
SARSAa	**5**	**2.5%**	0.38	98.90	0.27	0.68	99.08	0.13
	5	**5.0%**	0.73	99.37	0.13	0.28	99.81	0.54
	5	**10.0%**	0.26	98.90	0.13	0.44	99.85	0.20
	22	**2.5%**	-2.91	46.08	1.07	-7.45	16.09	6.11
	22	**5.0%**	-2.31	60.14	2.62	-4.10	11.93	4.96
	22	**10.0%**	-1.03	21.53	4.16	-1.17	38.72	10.67

Table 12. *SARSA*: Unicredit S.p.A. daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
QLa	**5**	**2.5%**	1.81	99.89	1.07	1.70	100.00	1.88
	5	**5.0%**	1.96	100.00	6.37	1.61	100.00	5.57
	5	**10.0%**	7.69	100.00	6.04	6.39	100.00	8.19
	22	**2.5%**	2.01	100.00	0.27	1.56	100.00	2.75
	22	**5.0%**	1.18	100.00	11.20	2.03	100.00	7.18
	22	**10.0%**	3.67	100.00	8.72	3.21	100.00	7.72

Table 13. *Q-Learning*: Artificial daily stock prices time series.

			N = 1			N = 5		
	L	**ε**	**g[%]**	**%**	**#**	**g[%]**	**%**	**#**
SARSAa	**5**	**2.5%**	2.20	99.87	0.40	2.13	100.00	0.74
	5	**5.0%**	2.23	100.00	0.74	2.27	100.00	0.54
	5	**10.0%**	2.14	100.00	1.41	1.93	99.99	1.81
	22	**2.5%**	2.14	100.00	0.40	2.12	100.00	0.07
	22	**5.0%**	2.25	100.00	0.13	2.37	99.99	0.34
	22	**10.0%**	1.41	100.00	1.61	2.04	100.00	0.87

Table 14. *SARSA*: Artificial daily stock prices time series.

Observing the individual performances of each stock, it is possible to see that generally both the algorithms perform well. Indeed, at the end of the considered investment period, most configurations obtain a positive return on the starting capital. Furthermore, the equity lines are usually greater than or equal to the starting capital for most of the period. In particular, the equity lines for which $\% \geq 97\%$ are under the starting capital only in the initial time, when the algorithms were beginning to learn.

The following tables, Table 15 and Table 16, show a couple of statistics computed with respect to the value of each single parameter for *QLa* and *SARSAa*,

respectively. These statistics measure the percentages of settings that obtained a positive net average annual logarithmic return, $g[\%]\text{net} > 0$, and the percentages of settings in which the obtained equity line is greater than or equal to the starting capital for at least half of the trading period, $\% > 50\%$.

	QLa	
	g[%]net > 0 [%]	% > 50% [%]
All configurations	75.00	83.33
N = 1	76.19	88.10
N = 5	76.19	78.57
L = 5	73.81	76.19
L = 22	76.19	90.48
ε = 2.5%	67.86	92.86
ε = 5.0%	67.86	78.57
ε= 10.0%	82.14	78.57

Table 15. *Q-Learning* statistics.

In Table 15 there is a strong evidence that *QLa* generally works well, because the percentages of settings that have a positive annual logarithmic return, are 75.00%. Furthermore, the percentages of setting that have the invested capital greater than or equal to the starting capital at least for the fifty percent of the trading days are 83.33%.

It is possible to verify that the differences about the positive average annual logarithmic return computed with respect to N and L are little relevant. Indeed, as regards N, the percentages of configurations are the same, i.e. 76.19%, while as regards L, they are 73.81% for $L = 5$ and 76.19% for $L = 22$. Then, with respect to $\% > 50\%$, $N = 1$ performs better than $N = 5$ (88.10% versus 78.57%) and $L = 22$ performs better than $L = 5$ (90.48%versus 76.19%).

Concerning ε, for values equal to 2.5% and to 5.0%, the percentages related to $g[\%]\text{net} > 0$ are the same, i.e. 67.86%, while the percentage related to $g[\%]\text{net} > 0$ increases for $\varepsilon = 10.0\%$ to 82.14%. Instead, an increment of ε causes a decrement of the percentages related to $\% > 50\%$ (likely, an increment of the frequency of the exploratory action may obtain positive results, but with a consequent increment of the volatility).

	SARSAa	
	g[%]net > 0 [%]	% > 50% [%]
All configurations	71.43	77.38
N = 1	71.43	80.95
N = 5	71.43	73.81
L = 5	90.48	90.48
L = 22	52.38	64.29
ε = 2.5%	78.57	82.14
ε = 5.0%	75.00	82.14
ε= 10.0%	60.71	67.86

Table 16. *SARSA* statistics.

In Table 16 it is possible to see that also *SARSAa* generally works well. Indeed $g[\%] > 0 = 71.43\%$ and $\% > 50\% = 77.38\%$.

It is possible to see that there is no difference between the performances with respect to $N = 1$ and $N = 5$. Their percentages are very similar if not the same.

Instead, there is a strong evidence in favor of $L = 5$. The percentages related to $g[\%]\text{net} > 0$ are 90.48% both for *QLa* and for *SARSAa*. Then, for $L = 22$, it seems that *SARSAa* reacts faster to the information than *QLa*.

Concerning ε, an increment of the exploratory action gets worse the results. The percentages related to $g[\%] > 0$ decreases from 78.57% to 60.71%, and the percentages related to $\% > 50\%$ decrease from 82.14% to 67.86%. So for *SARSAa*, the greedy actions are favorite.

Now, comparing Table 15 with Table 16, it is possible to see that the results are enough similar. The greater differences are about the performances related to L and to ε. *QLa* performs better than SARSAa with $L = 22$ (76.19% versus 52.38% for $g[\%] > 0$ and 90.48% versus 64.29% for $\% > 50\%$), while *SARSAa* performs better than *QLa* with $L = 5$ (90.48% versus 73.81% for $g[\%] > 0$ and 90.48% versus 76.19% for $\% > 50\%$). It is possible to assert that *SARSAa* reacts faster than *QLa* to the new information.

An increment of the value of ε entails most *QLa* configurations with positive annual logarithmic return although with an increase of the volatility. On the contrary, a low value of ε entails most *SARSAa* configurations with $g[\%] > 0$. So, it is possible to see that *QLa* prefers carrying out explorative actions, while *SARSAa* prefers greedy ones.

In the following table there are the values of the average annual number of transactions for all the configurations and for each stock.

	QLa	***SARSAa***
Ass. Generali S.p.A.	5.13	3.91
Fiat S.p.A.	6.35	4.09
Pirelli S.p.A.	6.53	1.13
Saipem S.p.A.	5.46	4.97
Telecom Italia S.p.A.	5.84	4.33
Unicredit S.p.A.	6.60	2.58
Artificial time series	5.58	0.78
TOTAL	41.49	21.79

Table 17. Average annual numbers of transactions.

Notice that *QLa* carries out more actions than *SARSAa*; the former performs on average twice of actions compared to the latter. *SARSAa* works mainly in the first period as it is possible to see in the Figure 2, while in Figure 1 it is possible to see that *QLa* works until the end of the considered period, also if it stays out from the market for long periods. A low number of transactions is preferable because it reduces the total amount of transaction costs, but the long periods of "indecision", that compel the algorithm to stay out from the market, may be counter-productive for a financial trading system.

Finally, we compare the above results with the ones presented in [7] (the Master's degree thesis of the second author). The comparison is limited to the *QLa*'s performances. The difference consists in the fact that in [7] the last performed action is not taken into account as state descriptor. In the following tables we report the statistics obtained in [7].

Observing Tables 15, 16 and 18, it is possible to verify that the adding as state descriptor the last performed action added is efficacious. Indeed, the above percentages of settings that obtain a positive average annual logarithmic return and that have the invested capital greater than or equal to the starting one at least for the fifty percent of the trading days are higher than or close to those presented in [7].

	QLa thesis	
	g[%]net > 0 [%]	% > 50% [%]
All configurations	64.29	85.71
N = 1	64.29	85.71
N = 5	64.29	85.71
L = 5	54.76	80.95
L = 22	73.81	90.48
ε = 2.5%	50.00	67.86
ε = 5.0%	57.14	92.86
ε= 10.0%	85.71	96.88

Table 18. Q-Learning thesis statistics.

	QLa thesis
Ass. Generali S.p.A.	9.56
Fiat S.p.A.	9.33
Pirelli S.p.A.	9.69
Saipem S.p.A.	9.33
Telecom Italia S.p.A.	9.26
Unicredit S.p.A.	8.54
Artificial time series	7.75
TOTAL	63.46

Table 19. Average annual numbers of transactions.

Then, comparing Table 17 with Table 19, it is possible to see that the addition of the last performed action as state descriptor reduces the average annual number of transactions.

7 Some final remarks

In this paper, two methods based on *RL*, i.e. *QLa* and *SARSAa*, are developed, applied and compared in order to exploit the human biases to obtain gains in financial markets. Indeed, following *AMH*, it is possible to effectively trade, because the stock prices are not able to instantly and accurately adapt themselves as soon as a new information arrives. The considered algorithms are able to interact with the environment and to

learn by own experience in order to carry out the "correct" actions to manage an invested capital.

QLa and *SARSAa* obtain both positive results although simple state descriptors are used.

Only one type of reward function is used, the Sharpe ratio. We think that this reward function is not sufficiently effective due to the complexity of financial markets.

Anyway, *SARSAa* obtains better results than *QLa* calculating its rewards on few days unlike *QLa*. So, it seems reasonable to assert that *SARSA*a is more responsive than *QLa* on the new information.

Concerning ε, we note that *QLa* prefers a greater frequency of explorative action to obtain positive results, while *SARSAa* prefers carrying out greedy actions for most of trading time.

Finally, the addition of the last performed action as state descriptor reduces the average annual number of transactions, and consequently reduces also the total amount of annual transaction costs. But, on the other hand, if the algorithm exits from the financial market for too long time, it means that the financial trading system is uncertain and so it has to be improved.

References

[1] F. Bertoluzzo and M. Corazza: Reinforcement Learning for automated financial trading: Basics and applications. In: S. Bassis, A. Esposito and F.C. Morabito (Eds.): *Recent Advances of Neural Network Models and Applications*, Springer, 197-213, 2014

[2] F. Bertoluzzo and M. Corazza: Testing different Reinforcement Learning configurations for financial trading: Introduction and applications. *Procedia. Economics and Finance*, 3, 68-77, 2012

[3] C. Gold: FX Trading via Recurrent Reinforcement Learning. *IEEE International Conference on Computational Intelligence in Financial Engineering*, 2003.

[4] A.W. Lo: Reconciling efficient markets with behavioral finance: The Adaptive Markets Hypothesis. *The Journal of Investment Consulting*, 7, 21-44, 2005.

[5] A.W. Lo: The Adaptive Markets Hypothesis. *The Journal of Portfolio Management*, 30, 15-29, 2004.

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

[7] A. Sangalli: Q-Learning: an intelligent stochastic control approach for financial trading. *Thesis of Master's degree – Ca' Foscari University of Venice*, 2015.