# Q-Learning and SARSA: Intelligent stochastic control approaches for financial trading

**Marco CORAZZA**

(corazza@unive.it)

Department of Economics – Ca' Foscari University of Venice

**EUROPEAN CENTER FOR LIVING TECHNOLOGY**

Venice, Italy – January 13, 2017

# Summary

# 0 – Introduction (1/8)

A (simple) definition

«A **trading system** is simply a group of specific rules, or parameters, that determine **entry** and **exit points** for a given equity.»

From: http://www.investopedia.com/

# 0 – Introduction (2/8)

## The starting question

Can machine learning methods be used to develop profitable trading systems?

Classical answer

Modern answer

# 0 – Introduction (3/8)

Classical answer

**Efficient Market Hypothesis (EMH)**

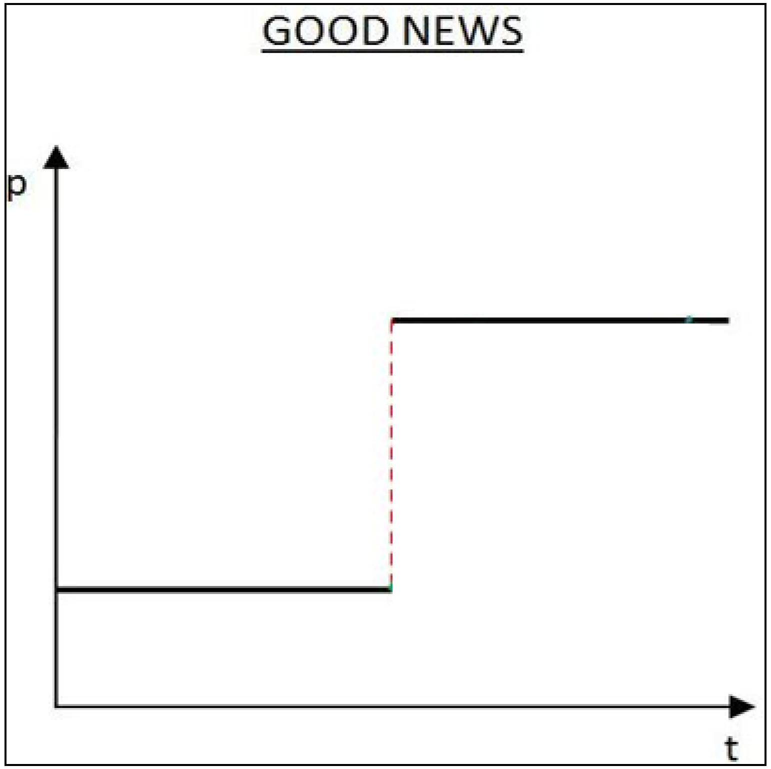
In financial markets, it is possible to "gain" only **randomly** and **rarely**.

In a liquid financial market, **asset prices fully reflect** all currently available **information**.

The EMH exists in various degrees:
**Weak** form (past prices and volumes);
**Semi-strong** form (public information);
**Strong** form (public and private information).

# 0 – Introduction (4/8)

Classical answer

# 0 – Introduction (5/8)

Classical answer

According to EMH, asset price movements can be described as

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

where

$t$ and $t$+1: consecutive **time instants**,

$P_\tau$: **asset price** at time $\tau$,

$E(\cdot)$: **expectation operator**,

$\tilde{P}_{t+1}$: **random variable** "asset price" at time $t$+1,

$\Omega_t$: **set of all available information** at time $t$,

$\tilde{\varepsilon}_{t+1}$: **error term** at time $t$+1, with $E(\tilde{\varepsilon}_{t+1}) = 0$.

# 0 – Introduction (6/8)

Modern answer

**Adaptive Market Hypothesis**

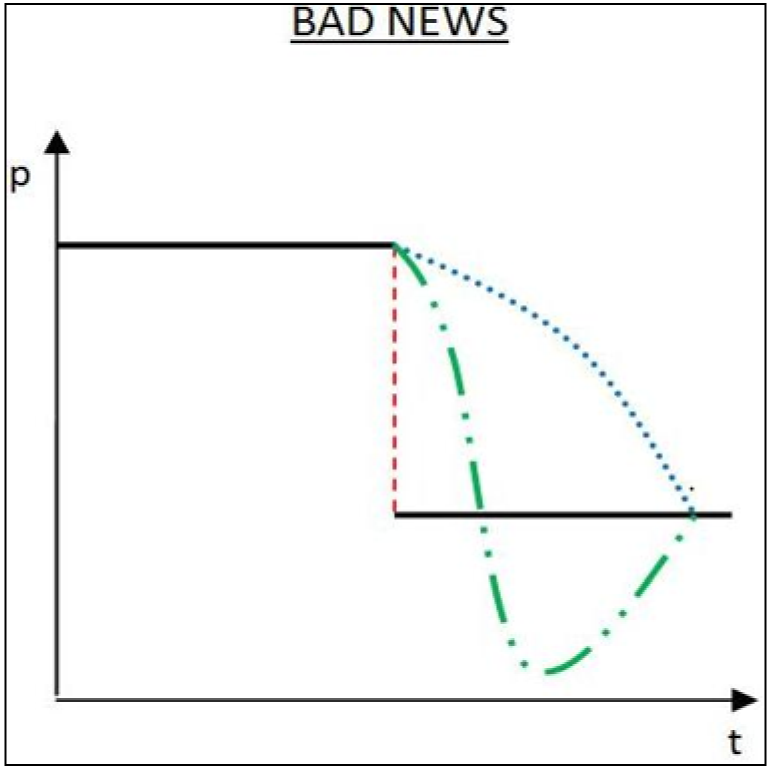
Financial markets are systems in which the agents **interact among them** in **complex ways** in order to **maximize** their **profits**.

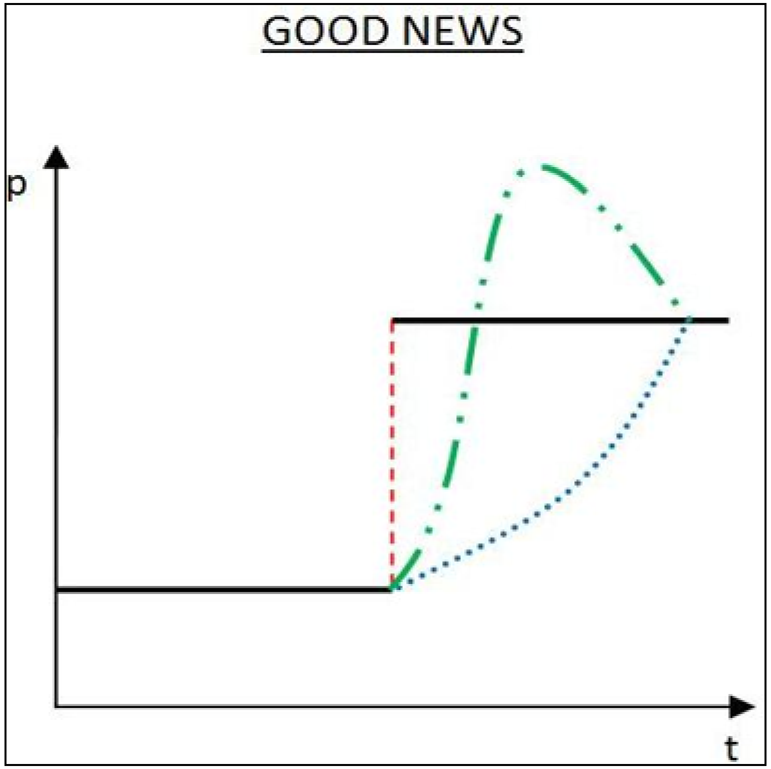
Agents react to the news in a **not fully rational ways**, making mistakes but **learning** and **adapting** their behaviors.

So, financial markets are generally **inefficient** and the traders can **more or less systematically** implement **profitable strategies**.

# 0 – Introduction (7/8)

Modern answer



# 0 – Introduction (8/8)

Summarizing

It is reasonable to assume **AMH** is valid.

Therefore, to develop profitable trading systems it is necessary to provide them with **problem-solving capabilities** similar to those of the superior living beings, that is to make them **"intelligent"**.

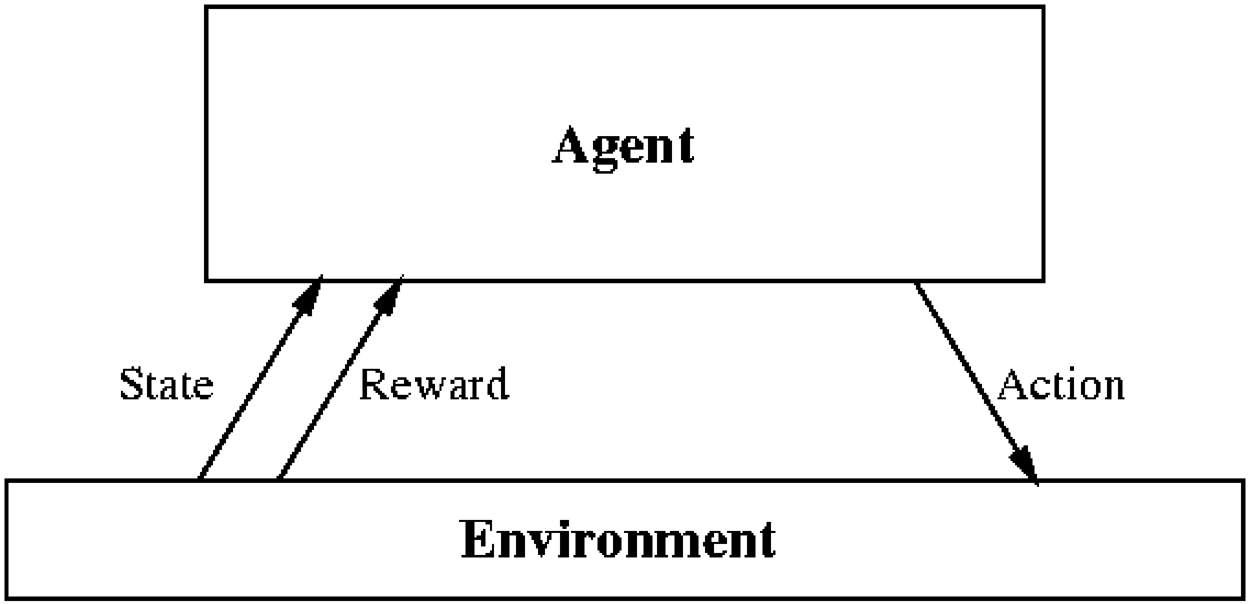
# 1 – Reinforcement Learning: A sketch (1/3)

**Machine learning** is a research field studying **methodologies** by which a computer is able to perform a finalized **action** given its **state** depending on an algorithmic learning based on **past** and **current data**.

Main machine learning methodologies

Supervised learning
Unsupervised learning
**Reinforcement learning**

# 1 – Reinforcement Learning: A sketch (2/3)




$$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} \ldots$$

# 1 – Reinforcement Learning: A sketch (3/3)

- *time* $t$: the system **detects the state of the environment**;

- *time* $t$: on this basis, the systems **takes an action**;

- *time* $t+1$: the environment returns a **negative** or **positive reward** coherently with the **exactness**, or less, of the action;

- *time* $t+1$: the agent modifies its **knowledge**, that is it learns, depending on whether the action is **incorrect** or **correct**.

# 1 – Reinforcement Learning: Formalization (1/5)

- **Discrete time framework**: $0, 1, \dots, t, t+1, \dots$;

- **State vector**: $\boldsymbol{s}_t \in \mathbb{R}^n$, it contains the relevant information;

- **Action**: $a_t \in A(\boldsymbol{s}_t) \subseteq \mathbb{R}$, where $A(\boldsymbol{s}_t)$ is the set of the possible actions according to the state $\boldsymbol{s}_t$;

- **Reward function**:

$$r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1});$$

# 1 – Reinforcement Learning: Formalization (2/5)

- **System's goal**: To maximize the expected value of
$$R(\boldsymbol{s}_t) = r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) + \gamma \cdot r(\boldsymbol{s}_{t+1}, a_{t+1}, \boldsymbol{s}_{t+2}) + \cdots +$$
$$+\gamma^2 \cdot r(\boldsymbol{s}_{t+2}, a_{t+2}, \boldsymbol{s}_{t+3}) + \cdots$$
where
$\gamma \in [0, 1[$ is a discount factor;

- **Reinforcement Learning's goal**:

To detect a **policy**, that is a map
$$\pi(\boldsymbol{s}_t) = a_t,$$
which associates to each state the action
that maximizes the expected value of $R(\boldsymbol{s}_t)$;

# 1 – Reinforcement Learning: Formalization (3/5)

- **Reformulated system's goal**: To maximize the expexted value of
$$R^{\pi}(\boldsymbol{s}_t) = r(\boldsymbol{s}_t, \pi(\boldsymbol{s}_t), \boldsymbol{s}_{t+1}) + \gamma \cdot r(\boldsymbol{s}_{t+1}, \pi(\boldsymbol{s}_{t+1}), \boldsymbol{s}_{t+2}) + \cdots +$$
$$+\gamma^2 \cdot r(\boldsymbol{s}_{t+2}, \pi(\boldsymbol{s}_{t+2}), \boldsymbol{s}_{t+3}) + \cdots;$$

- **Value functions**: For maximize the expexted value of $R(\boldsymbol{s}_t)$ or $R^{\pi}(\boldsymbol{s}_t)$, one introduce
$$V^{\pi}(\boldsymbol{s}) = \mathbf{E}(R^{\pi}(\boldsymbol{s}_t) | \boldsymbol{s}_t = \boldsymbol{s})$$
or
$$Q^{\pi}(\boldsymbol{s}, a) = \mathbf{E}(R^{\pi}(\boldsymbol{s}_t) | \boldsymbol{s}_t = \boldsymbol{s}, a_t = a),$$
where
$\boldsymbol{s}$ and $a$ are particular value of $\boldsymbol{s}_t$ and $a_t$ respectively;

# 1 – Reinforcement Learning: Formalization (4/5)

- **Bellman equation**: For the value equations, the following recursive relationship holds

$$V^{\pi}(\boldsymbol{s}) = \mathbf{E}[r(\boldsymbol{s}_t, \pi(\boldsymbol{s}_t), \boldsymbol{s}_{t+1}) + \gamma \cdot V^{\pi}(\boldsymbol{s}_{t+1}) | \boldsymbol{s}_{t+1} = \boldsymbol{s}]$$

from which the following **approximate iterative estimator**

$$\hat{V}^{\pi}_{k+1}(\boldsymbol{s}_t) = \mathbf{E}\left[r(\boldsymbol{s}_t, \pi(\boldsymbol{s}_t), \boldsymbol{s}_{t+1}) + \gamma \cdot \hat{V}^{\pi}_{k}(\boldsymbol{s}_{t+1})\right].$$

# 1 – Reinforcement Learning: Formalization (5/5)

- In order to improve the policy at each instant time, the following **approximate iterative procedure** is considered

$$\hat{V}_{k+1}^{\pi'}(\boldsymbol{s}_t) = \max_a \mathbf{E}\left[r(\boldsymbol{s}_t, a, \boldsymbol{s}_{t+1}) + \gamma \cdot \hat{V}_k^{\pi}(\boldsymbol{s}_{t+1})\right]$$

where

$\hat{V}_{k+1}^{\pi'}(\boldsymbol{s}_t)$ is the update estimate at time instant $t = k + 1$

and with

$$a_t = \begin{cases} \pi'(\boldsymbol{s}_t) \text{ with probability } 1 - \varepsilon \\ a \in A(\boldsymbol{s}_t) \text{ with probability } \varepsilon \end{cases}$$

where

$\varepsilon \in [0, 1]$

and

$\pi'(\boldsymbol{s}_t)$ is the action which maximizes $Q^{\pi}(\boldsymbol{s}, a)$.

# 2 – The Q-Learning and SARSA algorithms (1/4)

- In the **Q-Learning** framework, the **approximate iterative procedure** becomes

$$\hat{Q}_{k+1}(\boldsymbol{s}_t\,, a_t) = \hat{Q}_k\big(\boldsymbol{s}_{t,}\, a_t\big) + $$
$$+\frac{1}{k+1}\Big[r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) + \gamma \cdot \max_a \hat{Q}_k\big(\boldsymbol{s}_{t+1,}\, a_t\big) - \hat{Q}_k(\boldsymbol{s}_t\,, a_t)\Big];$$

- In the **SARSA** framework, the **approximate iterative procedure** becomes

$$\hat{Q}_{k+1}(\boldsymbol{s}_t\,, a_t) = \hat{Q}_k\big(\boldsymbol{s}_{t,}\, a_t\big) + $$
$$+\frac{1}{k+1}\Big[r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) + \gamma \cdot \hat{Q}_k\big(\boldsymbol{s}_{t+1,}\, a_t\big) - \hat{Q}_k(\boldsymbol{s}_t\,, a_t)\Big];$$

# 2 – The Q-Learning and SARSA algorithms (2/4)

- As the system takes into account **continuous** $\boldsymbol{s}_t$, $\hat{Q}_k(\boldsymbol{s}_t, a_t)$ may be approximate by

$$\hat{Q}_k(\boldsymbol{s}_t, a_t) = \hat{Q}_k(\boldsymbol{s}_t, a_t; \boldsymbol{\theta}_k)$$

where
$\boldsymbol{\theta}_k$ is a parameter vector, **randomly initialized**.

# 2 – The Q-Learning and SARSA algorithms (3/4)

- For obtaining the best estimation of $Q^{\pi}(\boldsymbol{s}_t, a_t)$, the following **linear functional** is considered

$$\hat{Q}^{\pi}(\boldsymbol{s}, \boldsymbol{a}; \boldsymbol{\theta}) = \boldsymbol{\theta}_0 + \sum_{i=1}^{n} \boldsymbol{\theta}_i \cdot \phi(s_i) + \sum_{i=n+1}^{n+m} \boldsymbol{\theta}_i \cdot \phi(a_i)$$

where
$\boldsymbol{s} = (s_1, \cdots, s_n)$ is the state vector,
$\boldsymbol{a} = (a_1, \cdots, a_m)$ is the action vector,
$\phi(\cdot)$ is a proper transformation function of states and actions.

# 2 – The Q-Learning and SARSA algorithms (4/4)

- Finally, one obtain the following **iterative updating rules**

$$\begin{cases} d_k = r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) + \gamma \cdot \max_a \hat{Q}^{\pi}\left(\boldsymbol{s}_{t+1,}\, a_t;\, \boldsymbol{\theta}_k\right) - \hat{Q}^{\pi}\left(\boldsymbol{s}_{t,}\, a_t;\, \boldsymbol{\theta}_k\right) \\ \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha_k \cdot d_k \cdot \nabla_{\boldsymbol{\theta}} \cdot \hat{Q}^{\pi}\left(\boldsymbol{s}_{t+1,}\, a_t;\, \boldsymbol{\theta}_k\right) \end{cases}$$

where

$\alpha_k \in\ ]0, 1]$ is the so called **learning rate**.

**Adaptive** stochastic optimal conrol problem.

# 3 – Operational implementation (1/6)

Specification of the **state vector** $\boldsymbol{s}_t$;

Specification of the **actions** $a_t$;

Specification of the **reward function** $r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$;

Specification of **transformation function** $\phi(\cdot)$;

Other.

# 3 – Operational implementation (2/6)

- Specification of the **state vector** $\boldsymbol{s}_t$:

$$\boldsymbol{s}_t = \left(e_{t-n}, e_{t-(n-1)}, \cdots, e_t, a_{t-1}\right),$$

where

$$e_t = \ln\left(\frac{p_t}{p_{t-1}}\right)$$

and

with

$n \in \{0, 4\}$;

- Specification of the **actions** $a_t$:

$$a_t = \begin{cases} -1 & \text{(short position)} \\ 0 & \text{(stay out position)}; \\ +1 & \text{(long position)} \end{cases}$$

# 3 – Operational implementation (3/6)

- Specification of the **reward function** $r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1})$, **Sharpe ratio**:

$$r(\boldsymbol{s}_t, a_t, \boldsymbol{s}_{t+1}) = \frac{\bar{g}_{t,L}}{\sqrt{\frac{\sum_{i=0}^{L-1}\left(g_{t-i} - \bar{g}_{t,L}\right)^2}{L-1}}},$$

where

$g_t = a_{t-1} \cdot e_t$

and

with

$L \in \{5, 22\}$;

- Specification of the **transaction costs**: 0,19%;

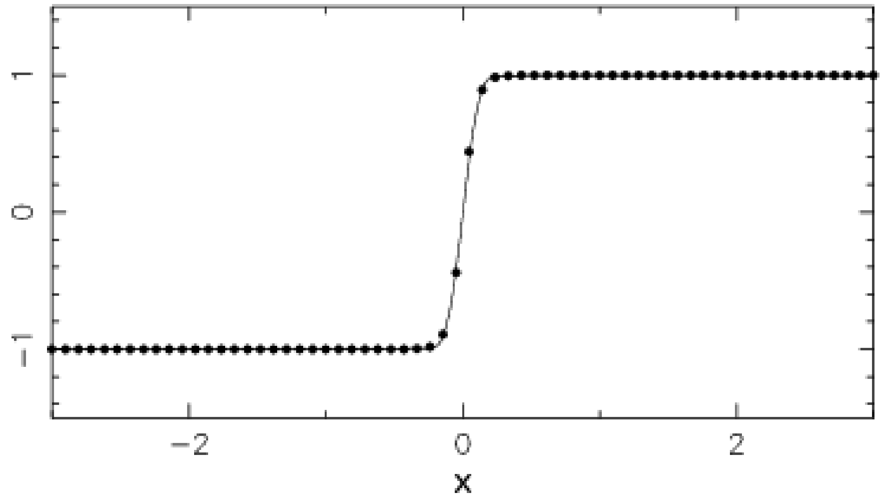
# 3 – Operational implementation (4/6)

- Specification of the **transformation** or **squashing function** $\phi(\cdot)$:

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

with

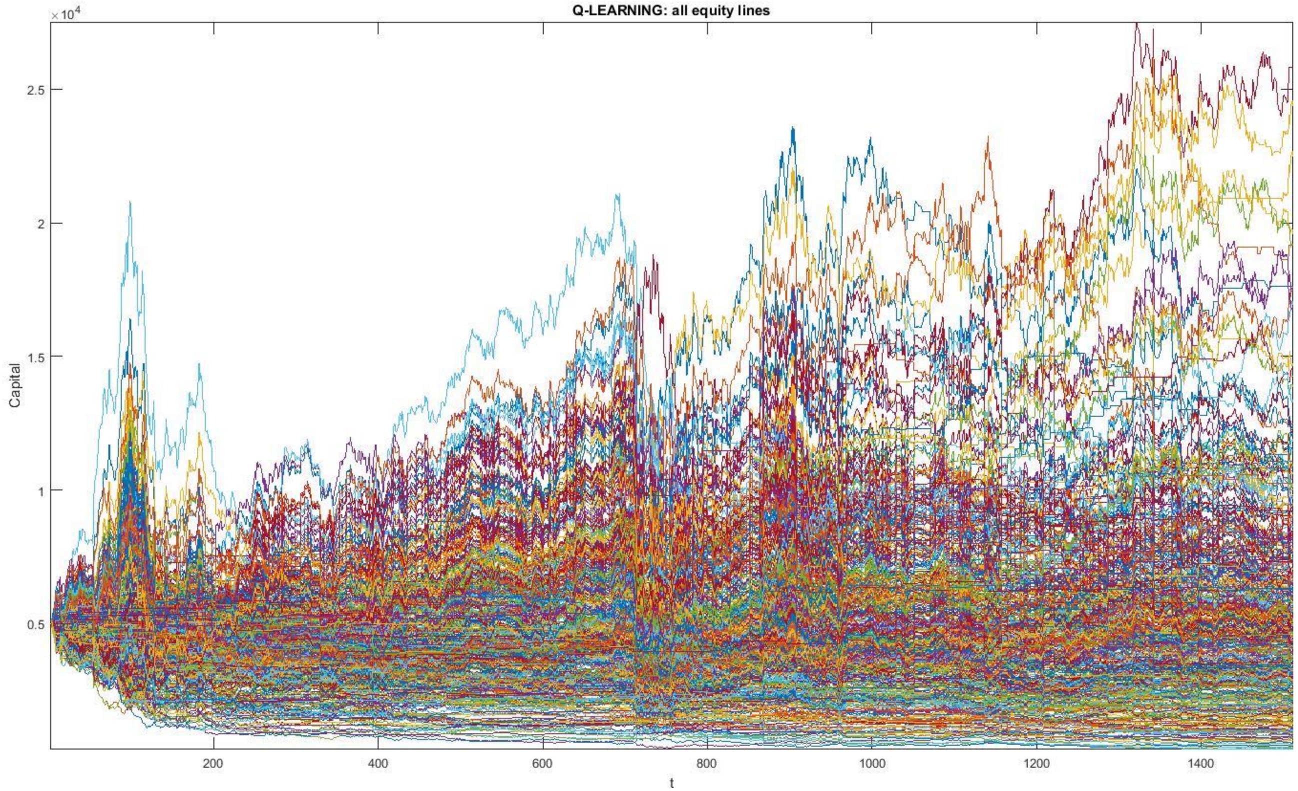
$a = 2, b = 1, c = 10^{15}$ and $d = -1$




- $\alpha = 5{,}0\%, \gamma = 95\%, \varepsilon \in \{2.5\%, 5.0\%, 10.0\%\}$;

# 3 – Operational implementation (5/6)

- Number of **iterations**: 500;
  (both Q-Learning and SARSA are stochastic algorithms)



# 3 – Operational implementation (6/6)

- Determination of the **action**:

$$a_t = \begin{cases} -1 \; if \; \bar{a}_t \in \left[-1, -\frac{1}{3}\right[ \\ 0 \; if \; \bar{a}_t \in \left[-\frac{1}{3}, \frac{1}{3}\right[ \\ +1 \; if \; \bar{a}_t \in \left[\frac{1}{3}, 1\right] \end{cases},$$

where

$$\bar{a}_t = \frac{1}{500} \sum_{i=1}^{500} a_{t,i}.$$

# 4 – Applications (1/9)

- **Daily closing prices**:

  From January 2, 1985 to September 30, 2014 (7518 data):

**Assicurazioni Generali**

**FIAT**

**Pirelli & C.**

**Saipem**

**Telecom Italia**

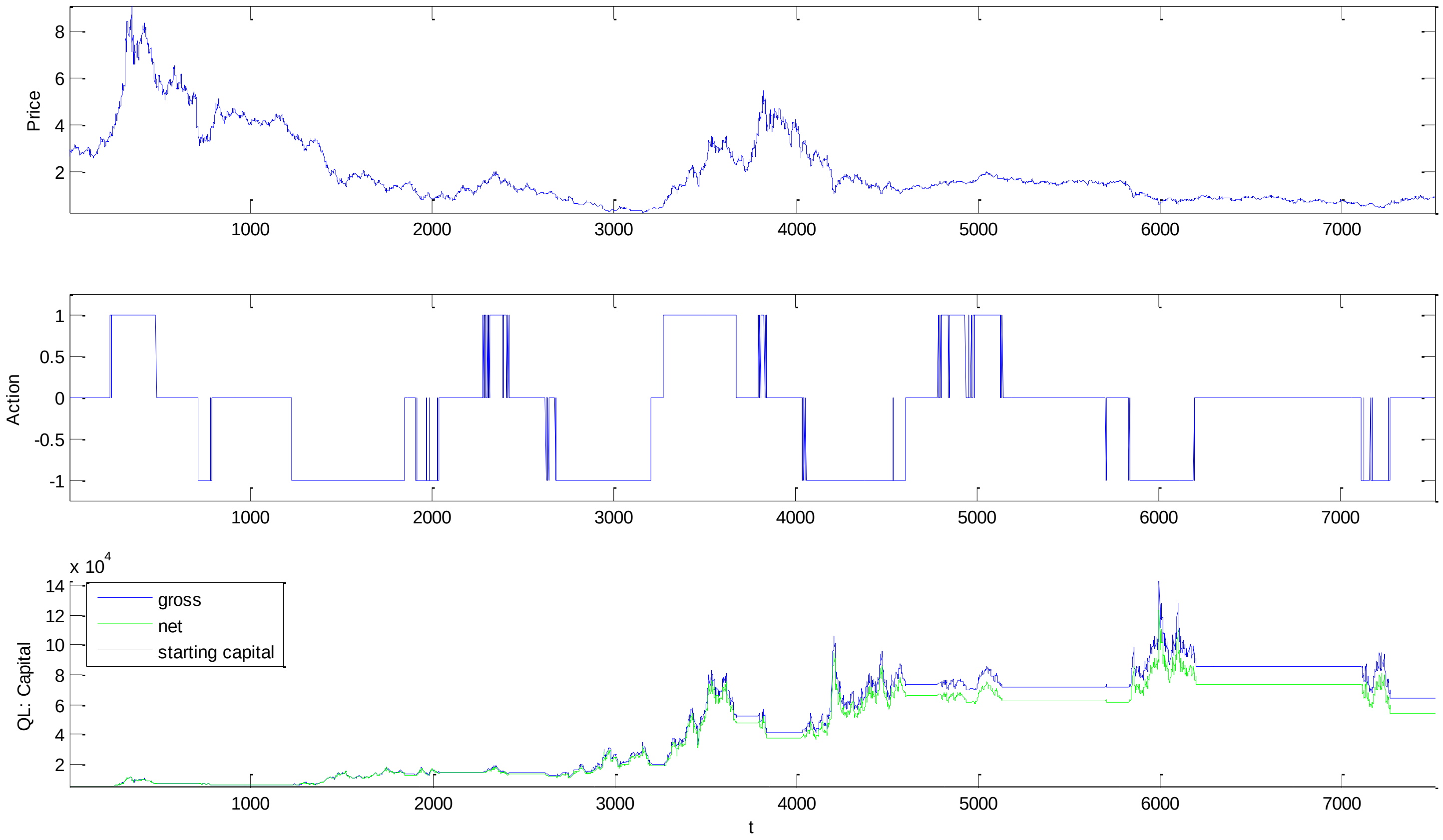
**UniCredit**

# 4 – Applications (2/9)

(**2** algorithms of **RL**) X
X (**2** values of $\boldsymbol{n}$) X
X (**2** values of $\boldsymbol{L}$) X
X (**3** values of $\boldsymbol{\varepsilon}$) =
= **24 configurations**.

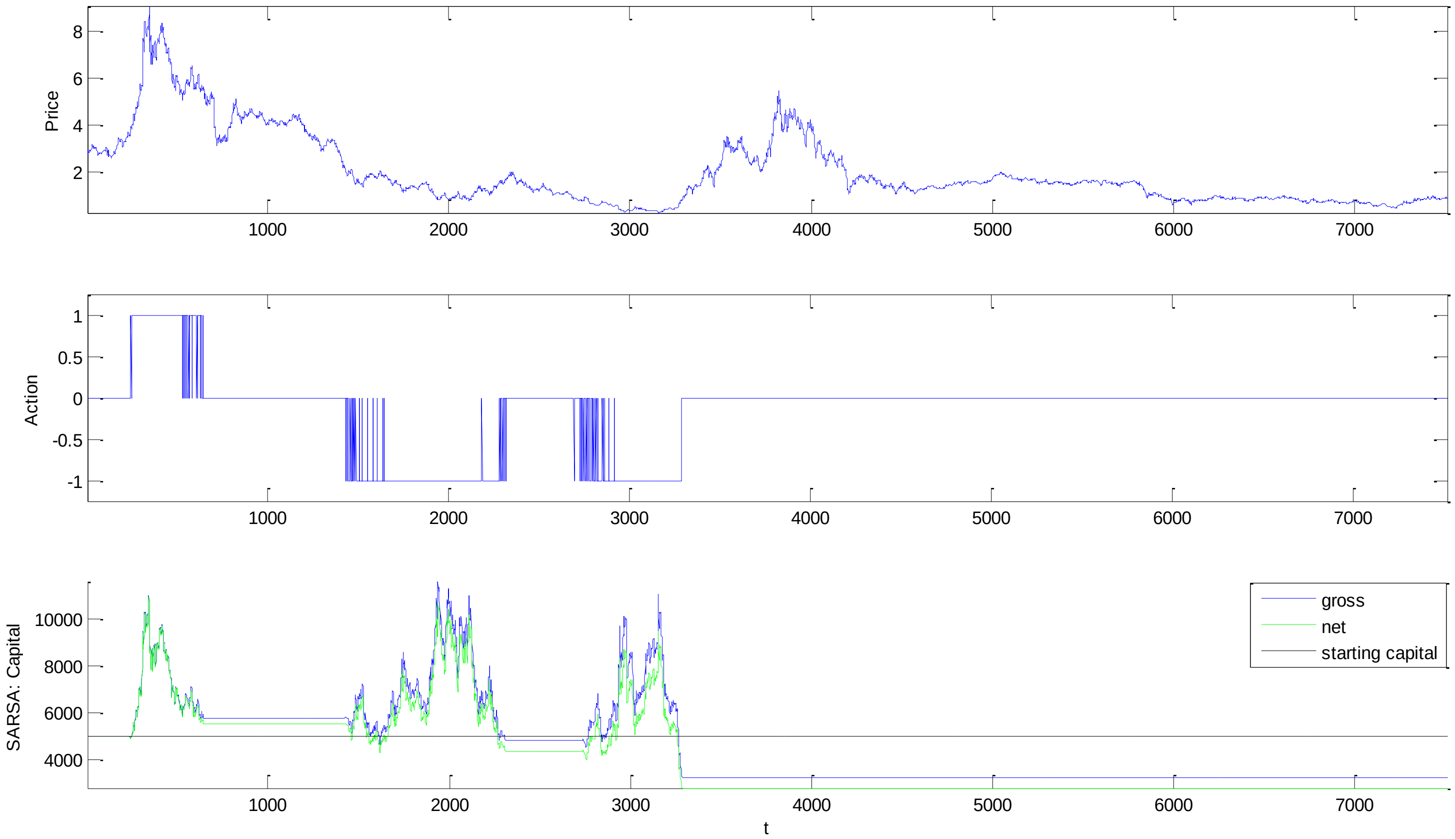
# 4 – Applications (3/9)

Telecom Italia – **Q-Learning**, $n = 5, L = 22, \varepsilon = 10.0\%$



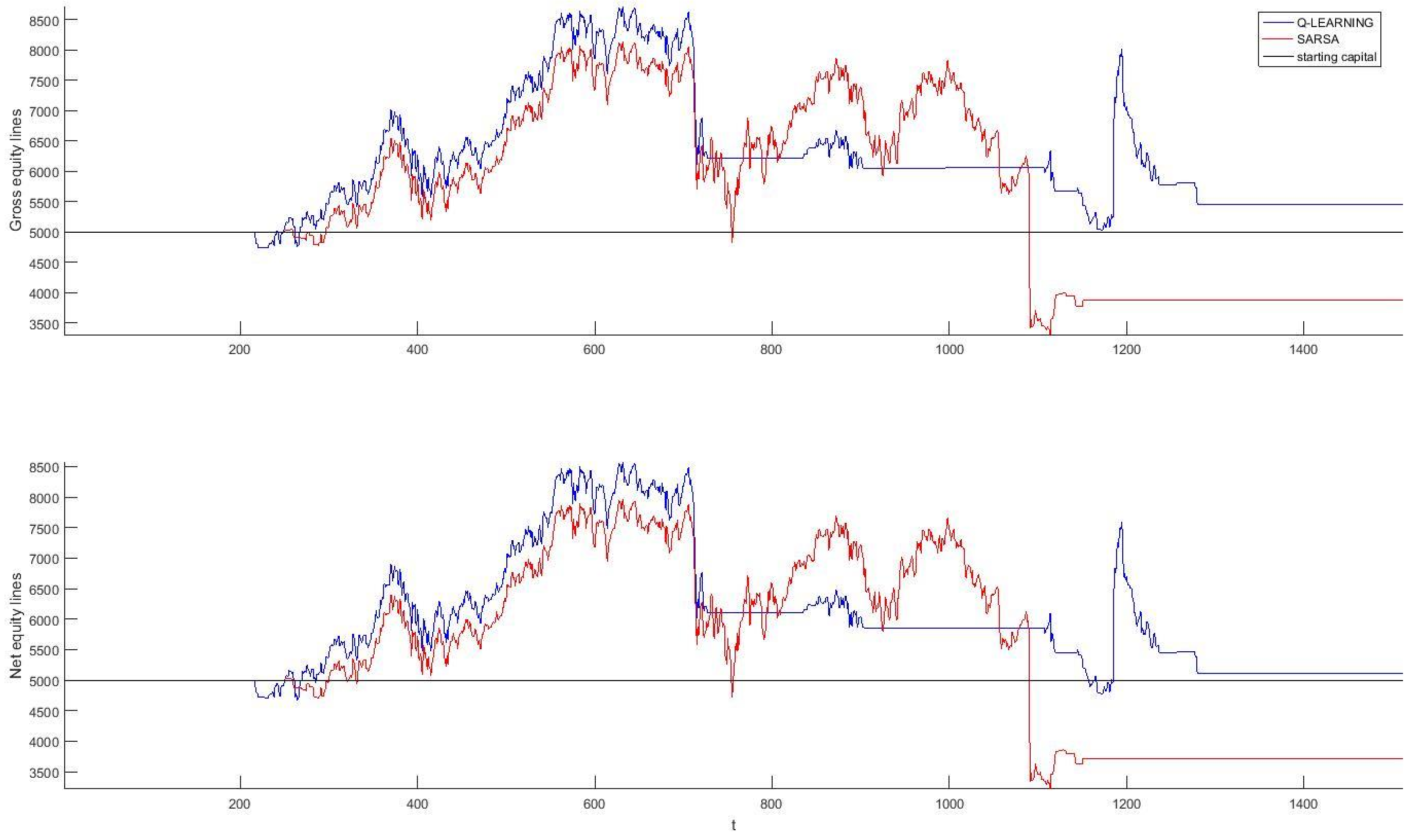
# 4 – Applications (4/9)

Telecom Italia – **SARSA**, $n = 5, L = 22, \varepsilon = 10.0\%$



# 4 – Applications (5/9)

## Pirelli & C. – **Q-Learning vs. SARSA**, $n = 5, L = 22, \varepsilon = 10.0\%$



# 4 – Applications (6/9)

**$g$[%]**: **Net average return per year**;

**%**: **Percentage of time such that $C(t) > C(1)$,**
**with $t = 2, \cdots, T = 7518$**;

**#**: **Average number of trade per year**;

**$g$[%]net>0**: **Percentage of configurations for which $g[\%] > 0$**;

**#>50%**: **Percentage of configurations such that $\% > 0$**
**in the 50% of the days**.

# 4 – Applications (7/9)

## Best case – Assicurazioni Generali

| | L | ε | N = 1 g[%] | N = 1 % | N = 1 # | N = 5 g[%] | N = 5 % | N = 5 # |
|---|---|---|---|---|---|---|---|---|
| *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 |

| | L | ε | N = 1 g[%] | N = 1 % | N = 1 # | N = 5 g[%] | N = 5 % | N = 5 # |
|---|---|---|---|---|---|---|---|---|
| *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 |

# 4 – Applications (8/9)

## Worst case – **Pirelli & C.**

| | L | ε | N = 1 g[%] | N = 1 % | N = 1 # | N = 5 g[%] | N = 5 % | N = 5 # |
|---|---|---|---|---|---|---|---|---|
| *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 |

| | L | ε | N = 1 g[%] | N = 1 % | N = 1 # | N = 5 g[%] | N = 5 % | N = 5 # |
|---|---|---|---|---|---|---|---|---|
| *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 |

# 4 – Applications (9/9)

| | *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 |

| | *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 |

# 5 – Some concluding remarks

- **Performances**
  - Both **Q-Learning** and **SARSA** achieve satisfactory results, although they use **simple states** and **not refined configurations**.

- **Log-returns as states**
  - Simple choice. (I am working on some new states)

- **Reward function**
  - The Sharpe ratio is too basic. (I am working on some new performance measures)

- **Estimation of $Q^{\pi}(s_t, a_t)$**
  - Experience new estimators besides the linear one.

# ***Thanks***
# ***for your attention!***

# References


- Bertoluzzo F. and Corazza M. (2014) "Reinforcement learning for automated financial trading: Basics and applications". In: Bassis S., Esposito A. and Morabito F.C. (Eds.): *Recent Advances of Neural Network Models and Applications*, Springer, 197-213.
- Lo A. (2004) "The Adaptive Markets Hypothesis: Market efficiency from an evolutionary perspective", *Journal of Portfolio Management,* 30, 15–29.
- Moody J., Saffel M. (2001) "Learning to trade via Direct Reinforcement", *IEEE Transactions on Neural Network*, 12, 875-889