Received: 21 February 2022 | Accepted: 20 December 2022

DOI: 10.1111/mafi.12378

MATHEMATICAL FINANCE An International Journal of Mathematics, Statistics and Financial Economics WILEY

ORIGINAL ARTICLE

Improving reinforcement learning algorithms: Towards optimal learning rate policies

Othmane Mounjid[1] | **Charles-Albert Lehalle**[2,3]

[1]CMAP, École Polytechnique, Palaiseau, France

[2]Abu Dhabi Investment Authority, Abu Dhabi, United Arab Emirates

[3]Imperial College London, London, UK

Correspondence
Charles-Albert Lehalle.
Email: c.lehalle@imperial.ac.uk

Abstract

This paper shows how to use results of statistical learning theory and stochastic algorithms to have a better understanding of the convergence of Reinforcement Learning (RL) once it is formulated as a fixed point problem. This can be used to propose improvement of RL learning rates. First, our analysis shows that the classical asymptotic convergence rate $O(1/\sqrt{N})$ is pessimistic and can be replaced by $O((\log(N)/N)^{\beta})$ with $\frac{1}{2} \leq \beta \leq 1$, and N the number of iterations. Second, we propose a dynamic optimal policy for the choice of the learning rate used in RL. We decompose our policy into two interacting levels: the inner and outer levels. In the inner level, we present the PASS algorithm (for "PAst Sign Search") which, based on a predefined sequence of learning rates, constructs a new sequence for which the error decreases faster. The convergence of PASS is proved and error bounds are established. In the outer level, we propose an optimal methodology for the selection of the predefined sequence. Third, we show empirically that our selection methodology of the learning rate outperforms significantly standard algorithms used in RL for the three following applications: the estimation of a drift, the optimal placement of limit orders, and the optimal execution of a large number of shares.

1 | INTRODUCTION

Reinforcement Learning (RL) has known a large success in solving control problems under a Markov setup either with full information (Silver et al., 2017) or partial information (Arulkumaran et al., 2019). However, when authors start applying this approach to financial markets, some additional difficulties show up. For example, while games such as Go or StarCraft can be repeated infinitely without any change of context and at no cost, financial markets are first nonstationary in the medium term, preventing the use of long windows of data, and then they host feedback loops that often prevent learning without paying any cost.

As a consequence, part of the literature focuses on using simulation environment (Ganesh et al., 2019; Guéant & Manziuk, 2019; Baldacci et al., 2022; Karpe et al., 2020; Vyetrenko et al., 2020; Amrouni et al., 2021; Ritter, 2017). It allows offline training as long as desired before using the learned control, but it relies on the accuracy of the simulator. Another part tries to fully exploit historical data by mixing real and simulated trajectories (Hendricks & Wilcox, 2014; Nevmyvaka et al., 2006; Cong et al., 2021; Leal et al., 2022). A third solution would consist of learning online. To do so, the optimal control needs to be determined as fast as possible. In all these cases, reaching the optimal control quickly enables either to reduce both the training time and cost or to decrease the amount of data needed in the training phase.

To learn the optimal strategy rapidly, this paper proposes an algorithm that can be viewed as an adaptation of the line search gradient technique (Hager & Zhang, 2005) for RL and provide theoretical results on its convergence. Furthermore, we give three financial applications: the estimation of drift, the optimal placement of limit orders, and the optimal execution of a large number of shares. The medium-term nonstationarity of the financial and economic context and the difficulty to model market dynamics make our results particularly suitable for application in financial markets.

We consider a discrete state space $\mathcal{Z} = \mathbb{N}$ or $\mathcal{Z} = \{1, \dots, d\}$ with $d \in \mathbb{N}^*$. We are interested in finding $q^* \in \mathcal{Q} \subset \mathbb{R}^{\mathcal{Z}}$ solution of

$$M(q, z) = \mathbb{E}[m(q, X(z), z)] = 0, \qquad \forall z \in \mathcal{Z}, \tag{1}$$

where $X(z) \in \mathcal{X}$ is a random variable with an unknown distribution, and m is a function from $\mathcal{Q} \times \mathcal{X} \times \mathcal{Z}$ to $\mathbb{R}$. Although the distribution of $X(z)$ is unspecified, we assume that we can observe some variables $(Z_n)_{n\geq 0}$ valued in $\mathcal{Z}$ and $(X_{n+1}(Z_n))_{n\geq 1}$ drawn from the same distribution of $X(Z_n)$. RL addresses this problem through the following iterative procedure:

$$q_{n+1}(Z_n) = q_n(Z_n) - \gamma_n(Z_n) m(q_n, X_{n+1}(Z_n), Z_n), \quad \forall n \geq 0, \tag{2}$$

where q_0 is a given initial condition, and each γ_n is a component-wise non-negative vector valued in $\mathbb{R}^{\mathcal{Z}}$. The connection between RL, problem (1), and Algorithm 4.2 is detailed in Section 2. It is possible to recover the classical SARSA, Q-learning, and double Q-learning algorithms used in RL by taking a specific expression for m and X_{n+1}. Note that Algorithm 4.2 is different from the standard Robbins–Monro (RM) algorithm used in stochastic approximation (SA)

$$q_{n+1} = q_n - \gamma_n \bar{m}(q_n, X_{n+1}), \tag{3}$$

TABLE 1 Asymptotic properties of some gradient methods. Note that d is the dimension of the state space $\mathcal{Z}$ and ϵ is a desired level of accuracy. Here ϵ corresponds to $1/n$. GD stands for Gradient Descent, SDG for Stochastic Gradient Descent, *Proximal* for Stochastic proximal gradient descent (Combettes & Pesquet, 2011; Schmidt et al., 2011), *Acc. prox.* for accelerated proximal stochastic gradient descent (Nitanda, 2014; Schmidt et al., 2011), SAGA for stochastic accelerated gradient approximation Defazio et al. (2014), and SVRG for stochastic variance reduced gradient (Johnson & Zhang, 2013).

		Iterations to achieve an ϵ precision		Time to reach an ϵ precision	
Algorithm	**Cost of one iteration**	**Convex**	**Strongly convex**	**Convex**	**Strongly convex**
GD	$O(d^2)$	$O(1/\epsilon)$	$O(1/\epsilon)$	$O(d^2/\epsilon)$	$O(d^2 \log(1/\epsilon))$
SGD	$O(d)$	$O(1/\epsilon^2)$	$O(1/\epsilon)$	$O(1/\epsilon^2)$	$O(1/\epsilon)$
Proximal	$O(d)$	$O(1/\epsilon)$	$O(\log(1/\epsilon))$	$O(1/\epsilon)$	$O(d \log(1/\epsilon))$
Acc. prox.	$O(d)$	$O(1/\sqrt{\epsilon})$	$O(\log(1/\epsilon))$	$O(d/\sqrt{\epsilon})$	$O(d \log(1/\epsilon))$
SAGA	$O(d)$	$O(1/\epsilon)$	$O(\log(1/\epsilon))$	$O(1/\epsilon)$	$O(d \log(1/\epsilon))$
SVRG	$O(d)$		$O(\log(1/\epsilon))$		$O(d \log(1/\epsilon))$

with $\bar{m} \in \mathbb{R}^{\mathcal{Z}}$ whose zth coordinate is defined such that $\bar{m}(q,x)(z) = m(q,x(z),z)$ for any $z \in \mathcal{Z}$ and $\gamma_n \geq 0$, mainly because, as it is frequent in RL, we do not observe the entire variable $(X_{n+1}(z))_{z\in\mathcal{Z}})$ but only its value according to the coordinate Z_n. Indeed, the way $(Z_n)_{n\geq 1}$ visits the set $\mathcal{Z}$ plays a key role in the convergence of Algorithm 4.2. RM algorithm was first introduced by Robbins and Monro (1951). After that, it was studied by many authors who prove the convergence of q_n towards q^*, see (Benveniste et al., 1987; Bertsekas & Tsitsiklis, 1996; Blumet al., 1954; Kushner & Clark, 1978). The asymptotic convergence rate has also been investigated in many papers, see Benveniste et al. (1987), Kushner and Yin (2003), Sacks (1958). They show that this speed is in general proportional to $1/\sqrt{N}$ with N the number of iterations (see Table 1).

In this work, we give a special focus to RL problems. Nowadays, RL covers a very wide collection of recipes to solve control problems in an exploration–exploitation context. This literature started in the seventies, see Watkins (1989); Werbos (1987), and became famous mainly with the seminal paper of Sutton, see Sutton et al. (1998). It largely relied on the recent advances in the control theory developed in the late 1950s, see Bellman and Dreyfus (1959). The key tool borrowed from this theory is the dynamic programming principle (DPP) satisfied by the value function. This principle enables us to solve control problems numerically when the environment is known and the dimension is not too large. To tackle the curse of dimensionality, recent papers, see Sirignano and Spiliopoulos (2018), use deep neural networks (DNN). For example, in Hutchinson et al. (1994), authors use DNN to derive optimal hedging strategies for finance derivatives and in Manziuk and Guéant (2019), they use a similar method to solve a high-dimensional optimal trading problem. To overcome the fact that the environment is unspecified, it is common to use an RM type algorithm, which estimates online quantities of interest. The combination of control theory and SA techniques gave birth to numerous papers on RL.

Our contributions are as follows.

- First, we conduct an error analysis to show that the classical asymptotic rate $O(1/\sqrt{N})$ is pessimistic and can be enhanced in many situations. For this, we borrow tools from the statistical learning theory and show how to use them in a RL setting to get a $O((\log(N)/N)^{\beta})$ asymptotic speed with $1/2 \leq \beta \leq 1$ and N the number of iterations.

- Second, we propose a dynamic policy for the choice of the step size $(\gamma_k)_{k\geq 0}$ used in Equation (2). Our policy is decomposed into two interacting levels: the inner and outer levels. In the inner level, we introduce the PASS algorithm, for "PAst Sign Search." This algorithm builds a new sequence $(\gamma^i_k)_{k\geq 0}$, using a predefined sequence $(\gamma^o_k)_{k\geq 0}$ and the sign variations of $m(q_n, X_{n+1}(Z_n), Z_n)$. The error of $(\gamma^i_k)_{k\geq 0}$ decreases faster than the one of $(\gamma^o_k)_{k\geq 0}$. In the outer level, we present an optimal dynamic policy for the choice of the predefined sequence $(\gamma^o_k)_{k\geq 0}$. These two levels are interacting in the sense that PASS influences the construction of $(\gamma^o_k)_{k\geq 0}$.
- Third, convergence of PASS algorithm is established and error bounds are provided.
- Finally, we show that our selection methodology provides better convergence results than standard RL algorithms in three numerical examples: the drift estimation, the optimal placement of limit orders, and the optimal execution of a large number of shares. When needed, the proofs of convergence of our numerical methods are given.

The structure of this paper goes as follows: Section 2 describes the relation between RL and Equation (1). Section 3 reformulates Equation (1) as an optimization problem and defines with accuracy the different sources of error. This enables us to derive the convergence speed of RL algorithms. Section 4 contains our adaptive learning rate algorithm. Finally, Section 5 provides numerical examples taken from the optimal trading literature: optimal placement of a limit order, and the optimization of the trading speed of a liquidation algorithm. Proofs and additional results are relegated to an appendix.

2 | REINFORCEMENT LEARNING

We detail in this section the relation between Equation (2) and value-based RL since we are interested in solving RL problems. RL aims at estimating the Q-function, which quantifies the value for the player to choose the action a when the system is at s. Let t be the current time, $U_t \in \mathcal{U}$ be a process defined on a filtered probability space $(\Omega, \mathcal{F}, \mathcal{F}_t, \mathbb{P})$, which represents the current state of the system, and $A_t \in \mathcal{A}$ the agent action at time t. We assume that the process (U_t, A_t) is Markov. The agent aims at maximizing

$$\mathbb{E}[\int_0^T \rho^s f(s, U_s, A_s)\, ds + \rho^T g(U_T)], \tag{4}$$

with g the terminal constraint, f the instantaneous reward, ρ a discount factor, and T the final time. Let us fix a time step $\Delta > 0$ and allow the agent to take actions only at times[1] $k\Delta$ with $k \in \mathbb{N}$. The Q-function is defined as follows:

$$Q(t,u,a) = \sup_A \mathbb{E}_A[\int_t^T \rho^{(s-t)} f(s, U_s, A_s)\, ds + \rho^{(T-t)} g(U_T) | U_t = u, A_t = a],$$

with $(t,u,a) \in \mathbb{R}_+ \times \mathcal{U} \times \mathcal{A}$, $A = \{A_t\,, t < T\}$ a possible control process for the agent. Note that the action of the agent depends on $s = (t,u)$ with t the current time and u the current state. We view the agent control A as a feedback process (i.e., adapted to the filtration $\mathcal{F}_t$). The Q-function

satisfies the classical DPP

$$Q(t,u,a) = \mathbb{E}[R_{t+\Delta} + \rho^{\Delta} \sup_{a' \in \mathcal{A}} Q(t+\Delta, U_{t+\Delta}, a') | U_t = u, A_t = a], \tag{5}$$

with $R_{t+\Delta} = \int_t^{t+\Delta} \rho^{(s-t)} f(s, U_s, A_s)\, ds$. Equation (5) shows that the optimal expected gain when the agent starts at s and chooses action a at time t is the sum of the next expected reward $R_{t+\Delta}$ plus the value of acting optimally starting from the new position $U_{t+\Delta}$ at time $t+\Delta$. By reformulating Equation (5), we obtain that Q solves the following equation:

$$\mathbb{E}[m(q, X(z), z)] = 0, \qquad \forall z = (t,u,a) \in \mathcal{Z} = [0,T] \times \mathcal{U} \times \mathcal{A}, \tag{6}$$

where

- $X(z) = (U^z_{t+\Delta}, R^z_{t+\Delta}) \in \mathcal{X} = \mathcal{U} \times \mathbb{R}$, U^z_s and R^z_s are, respectively, the conditional random variables U_s and R_s given the initial condition $(U_t, A_t) = (u,a)$ with $z = (t,u,a) \in \mathcal{Z}$.
- m is defined as follows:

$$m(q,x,z^1) = H(q,x,z^1) - q(z^1), \qquad H(q,x,z^1) = r + \rho^{\Delta} \sup_{a' \in \mathcal{A}} q(t^1 + \Delta, u, a'),$$

for any $x = (u,r) \in \mathcal{X}$, and $z^1 = (t^1, u^1, a^1) \in \mathcal{Z}$.

Thus, one can use Equation (1) to solve Equation (6).

Note that Equation (6) shows that one can study Q only on the time grid[2] $D_T = \{n\Delta,\ n \le T/\Delta\}$. Thus, we define A_k and U_k such that $A_k = A_{k\Delta}$ and $U_k = U_{k\Delta}$ for any $k \in \mathbb{N}$. The key variable to study is not the agent decision A_k but $Z_k = (k, U_k, A_k)$. Thus, the rest of the paper formulates the results in terms of Z_k only.

Actions of the agent.
It is important in practice to visit the space $D_T \times \mathcal{U} \times \mathcal{A}$ sufficiently enough. Thus, to learn Q, it is common to not choose the maximizing action[3], but to encourage exploration by visiting the states where the error is large.

Remark 2.1. In general, a solution q^* of Equation (1) does not necessarily solve an optimization problem in the form of Equation (4). However, when M can be written as the gradient of some given function f (i.e., $\nabla f = M$), q^* becomes a solution of a problem in the form of Equation (4).

3 | IMPROVEMENT OF THE ASYMPTOTIC CONVERGENCE RATE

In Benveniste et al. (2012, Part 2, Section 4), Kushner and Yin (2003, Section 10), and Kushner and Clark (1978, Section 7), the authors show a central limit theorem for the procedure (3), which ensures a convergence rate of $O(1/\sqrt{N})$ where N is the number of iterations. In this section, we extend such convergence rate to Algorithm 4.2 and aim at understanding how one can improve it. For this, we decompose our total error into two standard components: estimation and optimization errors.

3.1 | Error decomposition

In this section, the space $\mathcal{Z} = \{1, \dots, d\}$ is finite with $d \in \mathbb{N}^*$. In such a case, we view q, and $M(q)$ as vectors of $\mathbb{R}^{\mathcal{Z}}$. Moreover, the process $(Z_n)_{n\geq 1}$ is an homogeneous Markov chain. We consider the following assumption.

Assumption 3.1 (Existence of a solution). There exists a solution q^* of Equation (1).

Under Assumption 3.1, the function q^* is a solution to the minimization problem

$$\min_{q\in\mathcal{Q}} g(q), \tag{7}$$

where g can be selected as follows:

- If M can be written as the gradient of some function f, see Remark 2.1, one can take $g = f$.
- Otherwise, it is always possible to set $g(q) = \|M(q)\|$. For simplicity, we place ourselves in this case for the rest of the section.

In our context, we do not have direct access to the distribution of $X(z)$. Nevertheless, we assume that at time n, we keep a memory of a training sample of $n(z)$ independent variables $(X_i^z)_{i=1\cdots n(z)}$ drawn from the distribution $X(z)$ where $n(z)$ is the number of times the Markov chain Z_n visited z. We define q^n as a solution of

$$\min_{q\in\mathcal{Q}} g_n(q), \tag{8}$$

with $g_n(q) = \|M^n(q)\|$, and

$$M^n(q,z) = \mathbb{E}^n[m(q,X(z),z)] = \left(\sum_{j=1}^{n(z)} m(q,X_j(z),z)\right)/n(z),$$

the expected value under the empirical measure $\mu = (\sum_{j=1}^{n(z)} \delta_{X_j(z)})/n(z)$ (i.e., empirical risk). We finally define q_k^n as an approximate solution of the problem (5) returned by an optimization algorithm after k iterations. Thus, we can bound the error $g(q_k^n)$ by

$$0 \leq \mathbb{E}\left[\left(g(q_k^n) - g(q^*)\right)(z)\right] \leq \underbrace{\mathbb{E}[(g(q^n) - g(q^*))(z)]}_{\text{estimation error}} + \underbrace{\mathbb{E}\left[\left|g(q^n) - g(q_k^n)\right|(z)\right]}_{\text{optimization error}},$$

since q^* minimizes g.

3.2 | Convergence rate of the estimation error

3.2.1 | Slow convergence rate

We have the following result.

Proposition 3.2. *We assume that the Markov chain Z_n is irreducible. There exists $c_1 > 0$ such that*

$$\mathbb{E}[\sup_{q\in\mathcal{Q}} |g(q) - g_n(q)|] \leq c_1 \frac{1}{\sqrt{n}}, \quad \forall z \in \mathcal{Z}.$$

For sake of completeness, we give the proof of this result in Appendix A. Proposition 3.2 allows us to derive the following bound for the estimation error

$$\begin{aligned} \mathbb{E}[(g(q^n) - g(q^*))] &= \mathbb{E}[(g(q^n) - g_n(q^n))] + \underbrace{\mathbb{E}[(g_n(q^n) - g_n(q^*))]}_{\leq 0} + \mathbb{E}[(g_n(q^*) - g(q^*))] \\ &\leq 2\mathbb{E}\left[\sup_q |g(q) - g_n(q)|\right] \leq 2c_1 \frac{1}{\sqrt{n}}. \end{aligned} \tag{9}$$

This bound is known to be pessimistic.

3.2.2 | Fast convergence rate

We obtain the following fast statistical convergence rate.

Proposition 3.3. *Assume that the Markov chain Z_n is irreducible, and*

$$\mathbb{E}\left[\sup_{q\in\mathcal{Q}} |M(q,z) - M^n(q,z)|\, |n(z)\right] \leq c'\left(\frac{\log(\bar{n}(z))}{\bar{n}(z)}\right)^{\beta}, \tag{10}$$

with $\frac{1}{2} \leq \beta \leq 1$, $c' > 0$, and $\bar{n}(z) = n(z) \wedge 1$. Then, there exists $c_2 > 0$ such that

$$\mathbb{E}[\sup_{q\in\mathcal{Q}} |g(q^n) - g(q^*)|] \leq c_2\left(\frac{\log(n)}{n}\right)^{\beta}, \quad \forall z \in \mathcal{Z}.$$

The proof of this proposition is given in Appendix B. Since the conclusion of Proposition B relies on the condition (3.3), we give below two settings under which this condition is fulfilled.

Fast convergence rate for classification problems.
It is possible to establish Equation (10) for classification problems when

- The loss function g satisfies regularity conditions, of which the most important are: Lipschitz continuity and convexity, see Bartlett et al. (2006, Section 4).
- The data distribution satisfies some noise conditions, see for instance Tsybakov et al. (2004).
- The loss function has a bounded moment α with $\alpha > 1$, see Cortes et al. (2019, Section 4).

It is also possible to get rid of the $\log(n)$ factor in Equation (10), see the end of Section 5 in Bousquet et al. (2003).

Fast convergence rate for RL.
Since numerical examples focus on RL applications. We propose to derive Inequality (3.3) for RL problems. To do so, we adopt the same notations of Section 2.

Let t be the current time. We denote by $A = \{A_t, t < T\}$ the control process of the agent. Since we work under a Markov setting, A_t depends only on t, and the current state of the system U_t. Moreover, we assume that the agent can take at most a finite set of actions which means $N_A = |\mathcal{A}| < \infty$. For each A, we define q_A as follows:

$$q_A(t,u,a) = \mathbb{E}_A[\int_t^T \rho^{(s-t)} f(s, U_s, A_s)\, ds + \rho^{(T-t)} g(U_T) | U_t = u, A_t = a],$$

for any $(t,u,a) \in \mathbb{R}_+ \times \mathcal{U} \times \mathcal{A}$. By definition of q_A, we have

$$q_A(t,u,a) \leq \mathbb{E}[R_{t+\Delta} + \rho^{\Delta} \sup_{a' \in \mathcal{A}} q_A(t+\Delta, U_{t+\Delta}, a') | U_t = u, A_t = a], \tag{11}$$

where $R_{t+\Delta}$, and ρ^{Δ} are defined in Equation (5). Moreover, DPP ensures that the Q-function achieves equality in Equation (11). Thus, we can define the following loss:

$$g(A) = \sum_{z \in \mathcal{Z}} \mathbb{E}[m(q_A, X(z), z)], \tag{12}$$

for any control process A, with m, and X introduced in Equation (6). Note that this setting is very similar to the classification problem one

- The variable X plays the role of the input variable.
- The control process A can be assimilated to the function to learn. Since there are finite numbers of time steps and actions to perform, the strategy A should predict a finite set of options that can be interpreted as labels.

Thus, we can try to apply the same techniques and recover similar bounds. Such results are given in the proposition below.

Proposition 3.4. *Let $B > 0$. We define the class of controls $\mathbf{A}_B$ such that*

$$\mathbf{A}_B = \{A;\ |m(q_A, x, z)| \leq B,\ \forall (x,z) \in \mathcal{X} \times \mathcal{Z}\}.$$

- *Then, there exists a constant $C > 0$ such that with probability at least $1 - \delta$,*

$$M(q_A, z) - M^n(q_A, z) \leq C\left(\frac{\log(n(z))}{n(z)}\mathrm{Var}(A,z) + \frac{\log(1/\delta) + \log(\log(n(z)))}{n(z)}\right),$$

with $\mathrm{Var}(A,z)$ *the variance of the random variable* $m(q_A, X(z), z)$.

- *If in addition, there exists* $c > 0$, *and* $\beta \in [0,1]$ *such that*

$$\mathrm{Var}(A,z) \leq cM(q_A,z)^{\beta}, \quad \forall A \in \mathbf{A}_B,$$

then with probability at least $1-\delta$,

$$M(q_A,z) - M^n(q_A,z) \leq C\left(\left(\frac{\log(n(z))}{n(z)}\right)^{1/(2-\beta)} + \frac{\log(1/\delta) + \log(\log(n(z)))}{n(z)}\right).$$

The main steps of Proposition's 3.4 proof are given in Bousquet et al. (2003, Theorem 8). It is then standard to derive Equation (10) from Proposition 3.4.

3.3 | Convergence rate of the optimization error

We turn now to the optimization error. This means that the expected value in Equation (7) is replaced by the empirical risk, which is known. In such a case, one can use many algorithms to find q_n. We present in the table below the most important properties of some gradient methods.

3.4 | Conclusion

Following the formalism of Bottou and Bousquet (2008), we have decomposed our initial error into

- **Estimation error:** its convergence is $O(1/\sqrt{n})$ in pessimistic cases with n the number of iterations. In the other situations, the convergence is faster (i.e., $O((\log(n)/n)^{\beta})$) with $1/2 \leq \beta \leq 1$.
- **Optimization error:** the convergence is exponential under suitable conditions. In unfavo rable cases, the convergence rate is $O(1/n)$.

The comparison of these error sources shows that the estimation error is the dominant component. Thus, one can overcome the $O(1/\sqrt{n})$ asymptotic speed, in some situations, by improving the estimation error.

4 | OPTIMAL POLICY FOR THE LEARNING RATE γ

In this section, we take $\mathcal{Z} = \mathbb{N}$ and consider the following type of algorithms:

$$q_{n+1}(Z_n) = q_n(Z_n) - \gamma_n(Z_n)m(q_n, X_{n+1}(Z_n), Z_n), \qquad \forall n \in \mathbb{N}.$$

One can recover the classical SARSA, Q-learning, and double Q-learning algorithms used in RL by considering a specific expression for m and X_{n+1}. In such algorithms, the choice of γ_n is crucial. One can find in the literature (cf. Robbins & Monro, 1951) general conditions on γ_n needed for

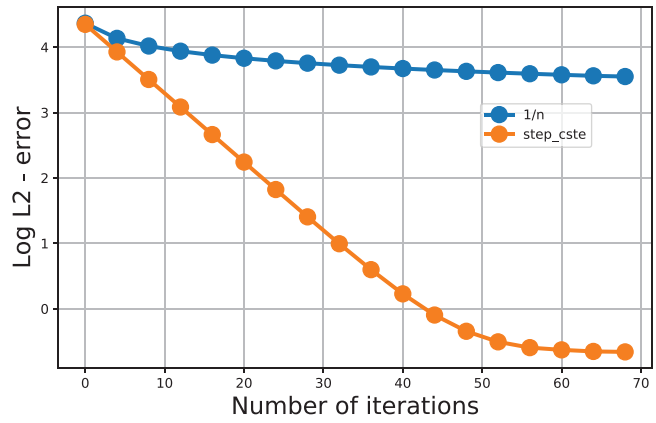

FIGURE 1 L^2-error for the estimation of the drift when γ_k is constant in orange and when $\gamma_k \propto \frac{1}{k}$ in blue. [Color figure can be viewed at wileyonlinelibrary.com]

convergence of Equation (2) such as

$$\sum_{k\geq 0} \gamma_k(z) = \infty, \quad a.s, \qquad \sum_{k\geq 0} \gamma_k^2(z) < \infty, \quad a.s, \quad \forall z \in \mathcal{Z}. \tag{13}$$

However, since the set of processes $(\gamma_n)_{n\geq 0}$ satisfying these conditions can be large, and even empty (when $(Z_n)_{n\geq 0}$ is not recurrent), many authors suggest to take γ_n proportional to $1/n^\alpha$. The exponent α may vary from 0 to 1 depending on the algorithm used, see (Gadat & Panloup, 2017; Moulines & Bach, 2011). Nonetheless, such a choice may be suboptimal. For example, Figure 1 shows that the blue curve is a way higher than the orange one. Here, the blue (resp. orange) curve shows how the logarithm of the error varies with n when $\gamma_n = \eta/n$ (resp. γ_n is constant). The constant η selected here ensures the fastest convergence for the blue curve.

In this paper, we propose to use a stochastic learning rate $(\gamma_k)_{k\geq 0}$; our learning policy is decomposed into two interacting levels: the inner and the outer level. In the inner level, we use the PASS algorithm, for "PAst Sign Search." This algorithm builds a new sequence $(\gamma_k^i)_{k\geq 0}$, based on a predefined sequence $(\gamma_k^o)_{k\geq 0}$ and the sign variations of $m(q_n, X_{n+1}(Z_n), Z_n)$, whose error decreases faster than the predefined one. In the outer level, we propose an optimal methodology for the selection of the predefined sequence $(\gamma_k^o)_{k\geq 0}$. These two levels are interacting in the sense that the PASS algorithm influences the construction of $(\gamma_k^o)_{k\geq 0}$.

4.1 | The inner level

4.1.1 | The algorithms

In this part, we introduce three algorithms. We start with our benchmark, which is the standard algorithm used in RL. Then, we present a second algorithm inspired from SAGA Defazio et al. (2014), which is a method used to accelerate the convergence of the stochastic gradient descent. Under suitable conditions, SAGA has an exponential convergence. Finally, we describe the PASS algorithm that modifies the learning rate $(\gamma_k)_{k\in\mathbb{N}}$ based on the sign variations of $m(q_n, X_{n+1}(Z_n), Z_n)$. The main idea is to increase γ_n as long as the sign of $m(q_n, X_{n+1}(Z_n), Z_n)$ remains unchanged. Then, we reinitialize or lower γ_n using a predefined sequence $(\gamma_k^o)_{k\in\mathbb{N}}$ when the sign of $m(q_n, X_{n+1}(Z_n), Z_n)$ switches. This algorithm can be seen as an adaptation of the line

search strategy, which determines the maximum distance to move along a given search direction. Actually, the line search method requires a complete knowledge of the cost function because it demands to evaluate several times $g(q_k + \gamma M(q^k)) - g(q_k)$ for different values of γ, with g being the loss and M representing a proxy of ∇g. However, our approach has neither access to g nor M. It can only compute $m(q_n, X_{n+1}(Z_n), Z_n)$ when the state $z = Z_k$ is visited. Moreover, to get a new observation, it needs to wait[4] for the next visit of the state $z = Z_k$. Nevertheless, it has instantaneous access to previously observed values. Thus, the main idea here is to use these past observations. Some theoretical properties of these algorithms are investigated in Section 4.1.3.

Algorithm 4.1 (RL). *We start with an arbitrary $q_0 \in \mathcal{Q}$ and define by induction q_k[5] as follows:*

$$q_{k+1}(Z_k) = q_k(Z_k) - \gamma_k(Z_k) m(q_k, X_{k+1}(Z_k), Z_k).$$

Algorithm 4.2 (SAGA). *We start with an arbitrary $q_0 \in \mathcal{Q}$, $M_0 = 0$[6] and define by induction q_k and M_k[5] as follows:*

$$\begin{aligned} q_{k+1}(Z_k) &= q_k(Z_k) - \gamma_k(Z_k)\left[m(q_k, X_{k+1}(Z_k), Z_k) - M_k[Z_k, i] + \frac{\left(\sum_{j=1}^{M} M_k[Z_k, j]\right)}{M} \right], \\ M_{k+1}[Z_k, i] &= m(q_k, X_{k+1}(Z_k), Z_k), \end{aligned}$$

with i picked from the distribution $p = (\sum_{i=1}^{M} \delta_i)/M$.

For the next algorithm, we give ourselves a predefined learning rate called $(\gamma_k)_{k\geq 0}$, a function $h : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ to increase the current learning rate, and another one $l : \mathbb{R}_+ \times \mathbb{R}_+ \to \mathbb{R}_+$ to lower it. The function h is used to accelerate the descent, while the function l goes back to a slower pace.

Algorithm 4.3 (PASS). *We start with an arbitrary q_0 and define by induction q_k and $\hat{\gamma}_k$*[5] *as follows:*

- *If $m(q_n, X_{n+1}(Z_n), Z_n) \times m(q_{r_1^n}, X_{r_1^n+1}(, Z_{r_1^n}), Z_{r_1^n}) \geq 0$, then do*

$$\begin{aligned} q_{n+1}(Z_n) &= q_n(Z_n) - h(\hat{\gamma}_n(Z_n), \gamma_n(Z_n)) m(q_n, X_{n+1}(Z_n), Z_n), \\ \hat{\gamma}_{n+1}(Z_n) &= h(\hat{\gamma}_n(Z_n), \gamma_n(Z_n)), \end{aligned}$$

  with r_1^n is the index of the last observation when the process Z visits the state Z_n.
- *Else, do*

$$\begin{aligned} q_{n+1}(Z_n) &= q_n(Z_n) - l(\hat{\gamma}_n(Z_n), \gamma_n(Z_n)) m(q_n, X_{n+1}(Z_n), Z_n), \\ \hat{\gamma}_{n+1}(Z_n) &= l(\hat{\gamma}_n(Z_n), \gamma_n(Z_n)). \end{aligned}$$

4.1.2 | Assumptions

In this section, we present the assumptions needed to study the convergence of Algorithms RL, SAGA, and PASS. We assume that Assumption 3.1 is in force. Hence, there exists q^*, a solution

of Equation (1). We write m^* for the vector $m^*(x,z) = m(q^*,x,z)$, $\forall (x,z) \in \mathcal{X} \times \mathcal{Z}$. Recall that $\mathbb{E}[m^*(X(z),z)] = 0$, $\forall z \in \mathcal{Z}$. Let us consider the following assumptions:

Assumption 4.4 (Pseudo strong convexity 2). There exists a constant $L > 0$ such that

$$(\mathbb{E}_k[m(q_k, X_{k+1}(Z_k), Z_k)])(q_k(Z_k) - q^*(Z_k)) \geq L(q_k(Z_k) - q^*(Z_k))^2,$$

with $\mathbb{E}_k[X] = \mathbb{E}[X|\mathcal{F}_k]$ for any random variable X.

Note that Assumption 4.4 is natural in the deterministic framework. For instance, if we take a strongly convex function f and call m its gradient (i.e., $m = \nabla f$). Then, m satisfies Assumption 4.4. Additionally, the pseudo-gradient property (PG) considered in Bertsekas and Tsitsiklis (1996, Section 4.2) is close to Assumption 4.4. However, Assumption 4.4 is slightly more general than PG since it involves only the component's norm $(q_k - q^*)(Z_k)$ instead of the vector's $(q_k - q^*)$. To get tighter approximations, we also introduce the quantity L_k as follows:

$$L_k = \begin{cases} \dfrac{\mathbb{E}_k[m(q_k, X_{k+1}(Z_k), Z_k)]}{q^k(Z_k) - q^*(Z_k)}, & \text{If } q^k(Z_k) - q^*(Z_k) \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Note that $L_k \geq 0$ under Assumption 4.4. It is also the biggest constant that satisfies Assumption 4.4 for a fixed k. In particular, this means that $L_k \geq L$.

Assumption 4.5 (Lipschitz continuity of m). There exists a positive constant $B > 0$ such that for any random variables X and X' valued in $\mathcal{X}$, we have

$$\mathbb{E}_k\left[\left(m(q_k, X, Z_k) - m^*(X', Z_k)\right)^2\right] \leq B\left\{1 + (q_k(Z_k) - q^*(Z_k))^2 + \mathbb{E}_k\left[\left(X - X'\right)^2\right]\right\},$$

with $\mathbb{E}_k[X] = \mathbb{E}[X|\mathcal{F}_k]$ for any random variable X.

Assumption 4.5 guarantees that m is Lipschitz. Authors in Bertsekas and Tsitsiklis (1996, Section 4.2) use a similar condition. To get better bounds, we introduce B_k such that

$$B_k = \frac{\mathbb{E}_k\left[\left(m(q_k, X, Z_k) - m^*(X', Z_k)\right)^2\right]}{1 + (q_k(Z_k) - q^*(Z_k))^2 + \mathbb{E}_k\left[(X - X')^2\right]}.$$

We have $B_k \leq B$ since B_k is the smallest constant satisfying Assumption 4.5 for a fixed k. We finally add an assumption on the learning $(\gamma_k)_{k\geq 0}$.

Assumption 4.6 (Learning rate explosion). For any $z \in \mathcal{Z}$, we have

$$\sum_{k\geq 1} \gamma_k(z) = \infty, \quad a.s.$$

When the process Z is Markov and $\gamma_k(z)$ bounded, Assumption 4.6 ensures that Z is recurrent. To see this, we first assume that $\gamma_k(z)$ is uniformly bounded without loss of generality. In such a case, there exists A such that $\gamma_k(z) \leq A$, for all $k \geq 1$. Thus, we get $\sum_{k\geq 1} \mathbb{E}[\gamma_k(z)\mathbf{1}_{Z_k=z}] \leq A \sum_{k\geq 1} \mathbb{P}[Z_k = z]$. Since the left-hand side of the previous inequality diverges under Assumption 4.6, we have

$$\sum_{k\geq 1} \mathbb{P}[Z_k = z] = \infty,$$

which proves that Z is recurrent.

4.1.3 | Main results

In this section, we compare Algorithms RL, SAGA, and PASS and prove the convergence of PASS. Let c be a positive constant and $k \in \mathbb{N}$. We define the error function as follows:

$$e^k(z) = \begin{cases} (q_k(z) - q^*(z))^2, & \text{for Algorithms RL and PASS,} \\ \dfrac{\sum_{j=1}^{M} \left(M^k[z,j] - m^*(z)\right)^2}{M} + c(q_k(z) - q^*(z))^2, & \text{for Algorithm SAGA,} \end{cases}$$

for all $z \in \mathcal{Z}$ and $j \in \{1, \dots, M\}$. We write E^k for the total error $E^k = \|e^k\|_\nu = \sum_{z\in\mathcal{Z}} e^k(z)\nu_z$ with $(\nu_z)_{z\in\mathcal{Z}}$ a non-negative sequence.[7] We also use the following notations:

$$p(x) = 2Lx - Bx^2, \qquad p_k(x) = 2L_k x - B_k x^2, \qquad \bar{\gamma}_k = \arg\sup_{l\in\mathbb{R}} p_k(l) = \frac{L_k}{B_k}, \qquad \forall x \in \mathbb{R}.$$

Proposition 4.7. *Let $z \in \mathcal{Z}$. Under Assumptions 3.1, 4.4, 4.5, and when there exists $r_1 \geq 1$ such that*

$$\gamma_2 \leq l(\gamma_1, \gamma_2), \qquad \text{and} \qquad h(\gamma_1, \gamma_2) \leq r_1\gamma_2, \qquad \forall(\gamma_1, \gamma_2) \in \mathbb{R}_+^2,$$

we have

$$\mathbf{1}_A \mathbb{E}_k[e^{k+1}(z)] \leq \mathbf{1}_A \left[\alpha_k e^k(z) + M_k\right], \tag{14}$$

with $A = \{Z_k = z\}$. The constants α_k and M_k vary from one algorithm to another as follows:

$$\alpha_k(z_1) = \begin{cases} [1 - p(\gamma_k(z))], & \text{for Algorithm RL,} \\ \max\left(1 - \left(2L\gamma_k(z) - 3B\gamma_k(z)^2\right) + \frac{B}{Mc}, 1 - \left(\frac{1}{M} - 6\gamma_k^2(z)c\right)\right), & \text{for Algorithm SAGA,} \\ \left\{1 - p_k\left(\underline{\gamma}_k\right) + d^1\gamma_k^2(z)\mathbf{1}_{c_k\geq 1}\right\}, & \text{for Algorithm PASS,} \end{cases} \tag{15}$$

and

$$M_k = \begin{cases} B\gamma_k^2(z)(4+3\upsilon_k), & \text{for Algorithm RL,} \\ 3B\gamma_k^2(z)(4+3\upsilon_k), & \text{for Algorithm SAGA,} \\ B(c_k\bar{\gamma}_k)^2(4+3\upsilon_k), & \text{for Algorithm PASS,} \end{cases} \tag{16}$$

with $c_k = \dfrac{\mathbb{E}_k[\hat{\gamma}_k(z)m(q^k, X_{k+1}(z), z)]}{\bar{\gamma}_k(z)\mathbb{E}_k[m(q^k, X_{k+1}(z), z)]}$, $\underline{\gamma}_k = c_k(z)\bar{\gamma}_k \vee \bar{\gamma}_k$, $d^1 = (r_1 - 1)^2 B_k$ *and* $\upsilon_k = \mathrm{Var}(Z_k)$.

The proof of Proposition 4.7 is given in Appendix C. Equation (14) reveals that the performance of Algorithms RL, SAGA, and PASS depends on the interaction between two competing terms:

- On the one hand, the slope α_k controls the decrease of the error from one step to the next.
- On the other hand, the quantity M_k gathers two sources of imprecision: the estimation and optimization errors. Both sources of imprecision have a variance term υ_n (because the distribution of Z is unknown), and a positive constant (coming from the noisy nature of observations).

There is a competition between these two terms: to decrease M_k, we need to send γ_k towards zero while the reduction of α_k requires a relatively small but still nonzero γ_k. Thus, γ_k should satisfy a trade-off in order to ensure the convergence of the algorithms. The RM conditions (4) are a way to address this trade-off. Now, in order to analyz e the properties of each algorithm, we compare for a fixed γ_k its respective values of α_k, and M_k in Table 2. For sake of clarity, we choose to present the variable $(1 - \alpha_k)$ instead of α_k in this table; note that a large value of $1 - \alpha_k$ means that α_k is small and thus induces a fast convergence.

Let $n \in \mathbb{N}^*$, $j \leq n$, $z_1 \in \mathcal{Z}$, and $\tau_{z_1} = \inf\{l > 0, Z_l = z_1\}$. We need to introduce the following notations: $a_j = \mathbb{P}[\tau_{z_1} \geq j | Z_0 = z_1]$, $b_j = \mu_j = \frac{\mathbb{E}_0[\alpha_j e^j(z_1)]}{\mathbb{E}_0[e^j(z_1)]}$, $r_j = 1 - \mu_j$, $\bar{\mu}_j^n = e^{-\sum_{j=n-j+1}^n r_j}$, $\bar{a}_j = a_j/r$, and $r = \sum_{j\geq 1} a_j$. Finally, we write $\bar{a}_n^{*\infty} = \lim_{m\to\infty} \bar{a}_n^{*m}$, and define the sequence $(\bar{a}_k^{*m})_{k\geq 1}$ recursively such that $\bar{a}_k^{*1} = \bar{a}_k$ and $\bar{a}_k^{*(m+1)} = \sum_{l=1}^k \bar{a}_{k+1-l}\bar{a}_l^{*m}$ for all $k \geq 1$ and $m \geq 1$. The result below holds only for Algorithms 4.1 and 4.3.

Theorem 4.8. *Let the Assumptions 3.1 and 4.4–4.6 be in force. Then, Algorithms 4.1 and 4.3 verify*

- *When* $\sum_{k\geq 0} \gamma_k^2(z) < \infty$ *for all* $z \in \mathbb{Z}$, *we have*

$$E^n \underset{n\to\infty}{\to} 0. \tag{17}$$

  in probability.
- *If in addition* $(Z_n)_{n\geq 1}$ *is an homogeneous Markov chain, and* $\mathbb{E}[E^1] < \infty$, *then*

$$\mathbb{E}[E^n] \underset{n\to\infty}{\to} 0. \tag{18}$$

TABLE 2 Comparison of the algorithms RL, SAGA, and PASS.

	$(1 - \alpha_k)$		M_k	
Algorithms	**Value**	**Comparison with Algo 1**	**Value**	**Comparison with Algo 1**
RL (Algo 1)	$2\,\gamma_k(z_1)\,L - \ B\,\gamma_k^2(z_1)$	—	$B\,\gamma_k^2(4+3\,\upsilon_k)$	—
SAGA (Algo 2)	$(2\gamma_k(z_1)L - 3B\gamma_k^2(z_1) - \frac{B}{Mc}) \vee (\frac{1}{M} - 6\gamma_k^2(z_1)c)$	Smaller	$3B\gamma_k^2(4+3\upsilon_k)$	Larger
PASS (Algo 3)	$2\underline{\gamma}_k L_k - B_k(\underline{\gamma}_k)^2 + d^1\gamma_k^2\mathbf{1}_{c_k\geq 1}$	Larger	$B(c_k\bar{\gamma}_k)^2(4+3\upsilon_k)$	Larger

- *Moreover, under the same condition, there exists a constant* $B' \geq 0$ *such that*

$$\mathbb{E}_0[e^n(z_1)] \leq B' \sum_{(l,j,i)\in(\mathbb{N}^*)^3, l+j+i=n} \bar{\epsilon}_j \bar{\mu}_l^j \bar{a}_i^{*\infty}, \tag{19}$$

with $\bar{\epsilon} = b_n \epsilon$, *and* $\epsilon_n = e^1(z_1)a_n(z_1) + \sum_{j=1}^{n-1} a_{n-j}\mathbb{E}[M_j]$.

Equations (17) and (18) ensure the convergence of the error E^n towards zero in both probability and L^2. Equation (19) gives an upper bound for the error. In particular, it shows how the terms $\bar{\epsilon}_n$, $\bar{\mu}_k^n$, and $\bar{a}_n^{*\infty}$ interact together to decrease of the error $\mathbb{E}_0[e^n(z_1)]$. We recall that $\bar{\epsilon}_n$ is a noise term that gathers the sources of imprecision, $\bar{\mu}$ represents the influence of the slope factor α, $\bar{a}_n^{*\infty}$ is related to the probability distribution of the process $(Z_k)_{k\geq 0}$. Note that the right-hand side of Equation (19) is not trivial and it converges towards 0 when there exists $L' \geq 0$ such that $\mathbb{E}_0[\alpha_j e^j(z_1)] \leq L'\mathbb{E}_0[e^j(z_1)]\mathbb{E}[\alpha_j]$ for all $j \geq 1$.

4.2 | The upper level

In practice, to apply PASS, we need an appropriate predefined sequence $(\gamma_k)_{k\in\mathbb{N}}$. It is possible to take γ_k proportional to $1/k^\alpha$ with $\alpha \in (0, 1]$ as proposed in Moulines and Bach (2011), Robbins and Monro (1951). However, in this section, we present an optimal dynamic policy for the choice of the learning rate $(\gamma_k)_{k\in\mathbb{N}}$. To do so, we assume that

$$e^{n+1}(z) = \mathbf{1}_A(\alpha_n e^n(z) + M_n + S_n) + \mathbf{1}_{A^c} e^n(z), \tag{20}$$

with $A = \{Z_n = z\}$, S_n does not depend on the learning rate and verifies $\mathbb{E}[\mathbf{1}_A S_n] \leq 0$. Equation (20) is consistent with Proposition 4.7. Moreover, we force e^n to stay below the upper bound $x_2 = \arg\sup_{x\in\mathbb{R}_+} g(x)$ with

$$g(x) = x - \frac{L^2 x^2}{2B(x + (2 + \upsilon))}, \forall x \in \mathbb{R}.$$

Such a constraint is not that restrictive since we know that the error e^n converges towards 0. Since α_n and M_n are both functions of the learning rate, the idea is to choose the learning rate γ_n such that

$$\gamma_n = \arg\min_\gamma(\alpha_n(\gamma)e^n(z) + M_n(\gamma)), \tag{21}$$

which gives

$$\gamma_n = \begin{cases} \dfrac{L}{B}\dfrac{e^n(z)}{e^n(z) + (2 + \upsilon_n)}, & \text{for Algorithm RL,} \\ \dfrac{L_n}{B_n}\dfrac{e^n(z)}{e^n(z) + d_1/B_n\mathbf{1}_{c_n\geq 1} + c_n^2(2 + \upsilon_n)}, & \text{for Algorithm PASS.} \end{cases}$$

The constants L, L_n, B, B_n, d_1, and c_n are defined in Proposition 4.7. Note that the value L/B is known to be a good choice for the learning rate. Thus, the proposed γ_n introduces a variation around this value that takes into account both the variance v_n, and an estimate of the past observed error e^n. The algorithm PASS adds a supplementary optimization layer since the global constants L, and B are replaced by the more local ones L_n, and B_n.

We write Γ for the set of processes $\tilde{\gamma} = (\tilde{\gamma}_n)_{n\geq 0}$ adapted to the filtration generated by the observed errors $(e^n)_{n\geq 0}$. We have the following result.

Proposition 4.9. *The sequence $\gamma = (\gamma_n)_{n\geq 0}$ defined in Equation (21) satisfies*

$$e^n_\gamma \leq e^n_{\tilde{\gamma}}, \quad a.s, \quad \forall n \in \mathbb{N}, \forall \tilde{\gamma} \in \Gamma.$$

We write e^n_γ to point out the dependence of the error e^n on the chosen control γ.

The proof of the above result is given in Appendix E. Proposition 4.9 shows that γ ensures that fastest convergence speed of the error and

$$\mathbb{E}[e^n_\gamma] = \inf_{\tilde{\gamma}\in\Gamma} \mathbb{E}[e^n_{\tilde{\gamma}}], \quad \forall n \in \mathbb{N}.$$

This guarantees its optimality.

We end this section with some practical considerations. Note that e^n is not known in practice because we do not have access to q^*. However, one can take the average value of $m(q_n, X_{n+1}(Z_n), Z_n)$ over the last $p \in \mathbb{N}^*$ visit times as a proxy of $e^n(Z_n)$. Moreover, the constants L and B are also unknown in practice. To tackle this issue, a first solution consists of starting with arbitrary values for B and L and generating a sequence of learning rates. If the error $m(q_n, X_{n+1}(Z_n), Z_n)$ increases, we take a larger value for B and a smaller one for L otherwise B and L values are kept unchanged. Finally, an alternative solution for the choice of the upper level learning rate consists of considering a piece-wise constant (PC) policy. To do so, one can track the average error of $m(q_n, X_{n+1}(Z_n), Z_n)$ over the last p visit times. If this average error does not decrease, the step size is divided by a factor α.

4.3 | Extension

The results of this section still hold when the descent sequence PASS is replaced by the vectorial version

- If $\langle \gamma_n m(q_n, X_{n+1}), m(q_{n-1}, X_n)\rangle \geq 0$, then do

$$\begin{aligned} q_{n+1} &= q_n - h(\hat{\gamma}_n, \gamma_n) m(q_n, X_{n+1}), \\ \hat{\gamma}_{n+1} &= h(\hat{\gamma}_n, \gamma_n), \end{aligned}$$

  with $m(q, X)$, and $h(\hat{\gamma}_n, \gamma_n)$, respectively, the vectors $m(q, X)(z) = m(q, X(z), z)$, and $h(\hat{\gamma}_n, \gamma_n)(z) = h(\hat{\gamma}_n(z), \gamma_n(z))$ for any $z \in \mathcal{Z}$, $q \in \mathcal{Q}$, and $X \in \mathcal{X}^{\mathcal{Z}}$.

- Else, do

$$\begin{aligned} q_{n+1} &= q_n - l(\hat{\gamma}_n, \gamma_n) m(q_n, X_{n+1}), \\ \hat{\gamma}_{n+1} &= l(\hat{\gamma}_n, \gamma_n), \end{aligned}$$

with $l(\hat{\gamma}_n, \gamma_n)$ the vector $l(\hat{\gamma}_n, \gamma_n)(z) = h(\hat{\gamma}_n(z), \gamma_n(z))$ for any $z \in \mathcal{Z}$.

When $\gamma_n(z) = 0$ if $z \neq Z_n$, we recover the standard PASS algorithm. Thus, the vectorial version is slightly more general and uses the scalar product between vectors instead of the product between two coordinates.

5 | SOME EXAMPLES

5.1 | Methodology

The code and numerical results presented in this section can be found in https://github.com/otM23/RL_adap_stepsize. Here, we compare four algorithms. The two first ones are two different versions of RL. In the first version, the learning rate $\gamma_k(z)$ is taken such that $\gamma_k = \eta / n_k(z)$ with $\eta > 0$ selected to provide the best convergence results and $n_k(z)$ the number of visits to the state z. In the second version, the step size follows the PC described at the end of Section D.2.2. The third algorithm is SAGA where the step size is derived from PC policy. Finally, we use the PASS algorithm presented in the previous sections with a predefined learning rate following the PC policy. We consider three numerical examples to compare the convergence speed of these algorithms: drift estimation, optimal placement of limit orders, and the optimal liquidation of shares.

5.2 | Drift estimation

Formulation of the problem.
We observe a process $(S_n)_{n \geq 0}$, which satisfies

$$S_{n+1} = S_n + f_{n+1} + W_n, \tag{22}$$

with W_n a centered noise with finite variance. We want to estimate the quantities f_i with $i \in \{1, \dots, n_{max}\}$. Using Equation (22) and $\mathbb{E}[W_t] = 0$, we get

$$\mathbb{E}[S_{i+1} - S_i - f_{i+1}] = 0, \qquad \forall i \in \{0, \dots, n_{max} - 1\}.$$

Thus, we can estimate f_i using stochastic iterative algorithms.

Numerical results.
Figure 2 shows the variation of the L^2-error when the number of iterations increases. We can see that the algorithm PASS outperforms standard SA algorithms. Moreover, other algorithms behave as expected: the standard RL decreases very slowly (but we know it will drive the asymptotic error to zero), the constant learning rate and SAGA provides better results than RL, while PASS seems to

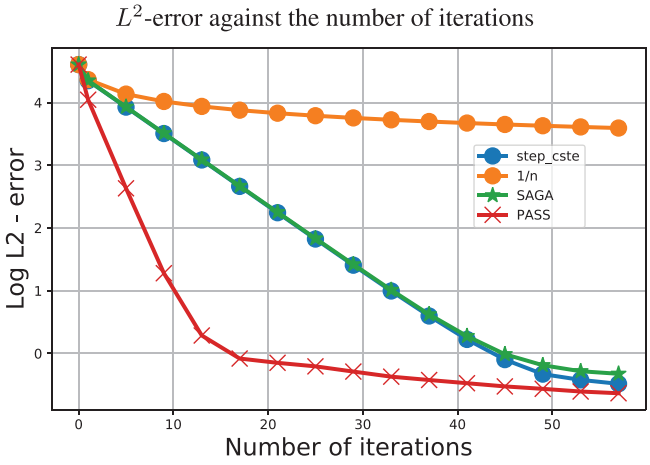

FIGURE 2 The L^2-error between f^k and f for different numerical methods averaged over 1000 simulated paths. [Color figure can be viewed at wileyonlinelibrary.com]

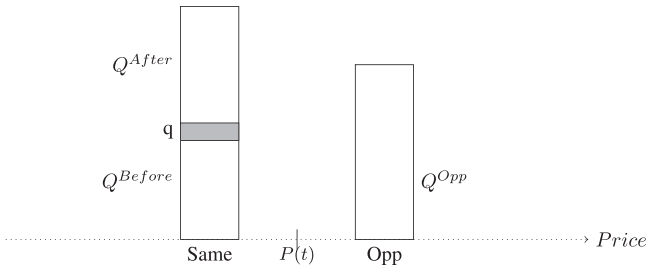

FIGURE 3 The state space of our limit order control problem.

have captured the best of the two worlds for this application: very fast acceleration at the beginning and the asymptotic error goes to zero.

5.3 | Optimal placement of a limit order

Formalization of the problem.
We consider an agent who aims at buying a unit quantity using limit orders, and market orders during the time interval $[0, T]$ (see Lehalle & Mounjid, 2017 for detailed explanations). In such a case, the agent wonders how to find the right balance between fast execution and avoiding trading costs associated with the bid–ask spread. The agent state at time t is modeled by $X_t = (Q^{Before}, Q^{After}, P)$ with Q^{Before} the number of shares placed before the agent's order, Q^{After} the queue size after the agent's order, and P_t the mid price, see Figure 3. The agents want to minimize the quantity

$$\mathbb{E}[F(X_{T\wedge T^{\text{exec}}\wedge\tau}) + \int_0^{T\wedge T^{\text{exec}}\wedge\tau} c\, ds],$$

where

- T is the final time horizon.
- $T^{\text{exec}} = \inf\{t \geq 0,\ P_t = 0\}$ is the first time when the limit order gets a transaction.
- τ is the first time when a market order is sent.

(a) Theoretical optimal control (b) step_cste optimal control (c) PASS optimal control

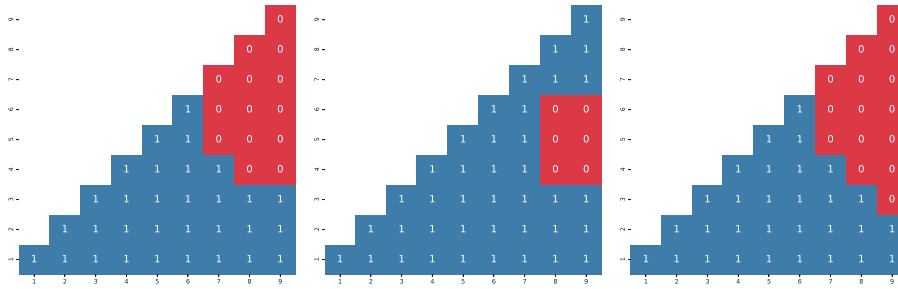

FIGURE 4 Comparison optimal control after 300 iteration for different methods: left is the optimal control, middle is RL with a step size derived from the upper level, and right is our optimal policy for the step size (i.e., upper level and inner level combined). [Color figure can be viewed at wileyonlinelibrary.com]

- $X = (Q^{Before}, Q^{After}, P)$ is the state of the order book.
- $F(u)$ is the price of the transaction (i.e., $F(u) = p + \psi$ when the agent crosses the spread and $F(u) = p$ otherwise).

We show in Section 2 that the Q-function is solution of Equation (6), see details in https://github.com/otM23/RL_adap_stepsize. Thus, we can use Algorithms RL, SAGA, and PASS to estimate it.

Numerical results.
Figure 4 shows three control maps: the x-axis reads the quantity on "same side" (i.e., $Q^{same} = Q^{Before} + Q^{After}$) and the y-axis reads the position of the limit order in the queue, that is, Q^{Before}. The color and numbers give the control associated to a pair (Q^{same}, Q^{Before}): 1 (blue) means "stay in the book," while 0 (red) means "cross the spread" to obtain a transaction. The panel (at the left) gives the reference optimal controls obtained with a finite difference scheme, the middle panel the optimal corresponds to the controls obtained for a RL algorithm where the step-size $(\gamma_k)_{k\geq 0}$ is derived from the upper level policy, and the right panel the optimal control obtained with our optimal policy (i.e., upper level and inner level combined). It shows that after a few iterations, our optimal policy already found the optimal controls. Figure 5 compares the log of the L^2 error, averaged over 100 trajectories, between the different algorithms. We see clearly that our methodology improves basic SA algorithms. Again, the other algorithms behave as expected: SAGA is better than a constant learning rate that is better than the standard RL (at the beginning, since we know that asymptotically RL will drive the error to zeros whereas a constant learning rate does not).

5.4 | Optimal execution

Formalization of the problem.
This is not the first work where RL is used to solve optimal trading problems. For example, authors in Nevmyvaka et al. (2006) apply RL techniques to solve optimal execution issues and in Manziuk and Guéant (2019), they use deep RL to solve a high-dimensional market making problem. However, we consider here a different application. An investor wants to buy a given quantity q_0 of a

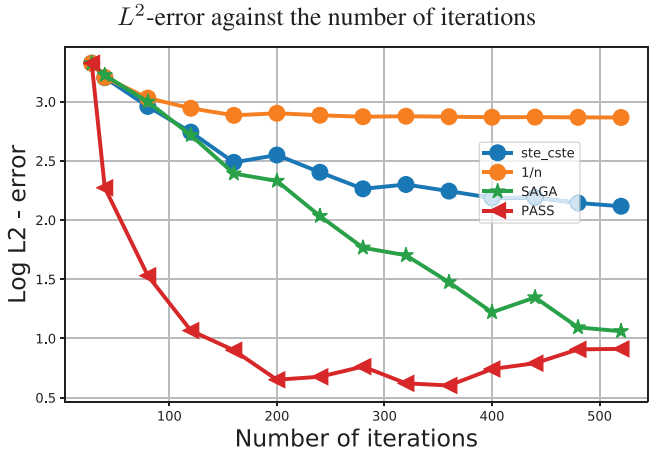

FIGURE 5 The log L^2-error against the number of iterations averaged over 1000 simulated paths. [Color figure can be viewed at wileyonlinelibrary.com]

tradable instrument (see Cartea et al. (2015) and Lehalle et al. (2018) for details about this setting). The price S_t of this instrument satisfies the following dynamic:

$$dS_t = \alpha dt + \sigma dB_t, \tag{23}$$

where $\alpha \in \mathbb{R}$ is the drift and σ is the price volatility. The state of the investor is described by two variables: its inventory Q_t and its wealth X_t at time t. The evolution of these two variables reads

$$\begin{cases} dQ_t = \nu_t dt, & Q_0 = q_0, \\ dW_t = -\nu_t(S_t + \kappa\nu_t)dt, & W_0 = 0, \end{cases} \tag{24}$$

with ν_t the trading speed of the agent and $\kappa > 0$. The term $\kappa\nu_t$ corresponds to the temporary price impact. The investor wants to maximize the following quantity:

$$W_T + Q_T(S_T - AQ_T) - \phi \int_t^T Q_s^2 \, ds,$$

it represents its final wealth X_T at time T, plus the value of liquidating its inventory minus a running quadratic cost. The value function V is defined such that

$$V(t,w,q,s) = \sup_{\nu} \mathbb{E}\left[W_T + Q_T(S_T - AQ_T) - \phi \int_t^T Q_s^2 \, ds | W_t = w, \, Q_t = q, \, S_t = s \right].$$

We remark that $v(t,w,q,s) = V(t,w,q,s) - w - qs$ verifies

$$v(t,w,q,s) = \sup_{\nu} \mathbb{E}\left[\underbrace{(W_T - W_t) + (Q_T S_T - Q_t S_t)}_{=M_T^t} - AQ_T^2 - \phi \int_t^T Q_s^2 \, ds | W_t = w, \, Q_t = q, \, S_t = s \right].$$

Using Equations (23) and (24), we can see that the variable M_T^t is independent of the initial values W_t and S_t. This means that v is a function of only two variables: the time t and the inventory q.

(a) Theoretical value function (b) step_ cste value function (c) PASS value function

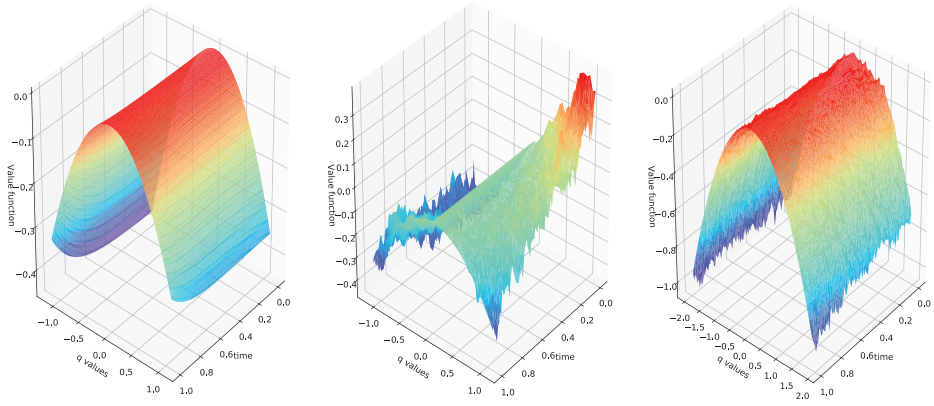

FIGURE 6 Comparison value function between methods. [Color figure can be viewed at wileyonlinelibrary.com]

The DPP ensures that v satisfies

$$v(t,q) = \sup_{\nu} \mathbb{E}\left[M^t_{t+\Delta} - \phi \int_t^{t+\Delta} Q_s^2 \, ds + v(t+\Delta, Q_{t+\Delta}) | Q_t = q \right]. \tag{25}$$

We fix a maximum inventory $\bar{q}$. Let $k = (k_T, k_q) \in (\mathbb{N}^*)^2$, $\Delta = T/k_T$, $D_T = \{t_i^{k_T};\ i \le k_T\}$, and $D_q = \{q_i^{k_q};\ i \le k_q\}$ with $t_i^{k_T} = i\Delta$ and $q_i^{k_q} = -\bar{q} + 2i\bar{q}/k_q$. To estimate v, we use the numerical scheme $(v_n^k)_{n\ge 1,\, k\in(\mathbb{N}^*)^2}$ defined below:

$$v^k_{n+1}(Z_n) = v^k_n(Z_n) + \gamma_n(Z_n)\left[\sup_{\nu \in A(Z_n)} \{ M^{\nu}_{n+1} - \phi \Delta Q_n^2 + v^k_n(Z^{\nu}_{n+1}) - v^k_n(Z_n) \} \right],$$

with $Z_n = (n\Delta, Q_{n\Delta})$ and $A(Z_n) \in D_q$ is the set of admissible actions[8]. When the final time T is reached (i.e., $n = k_T$), we pick a new initial inventory from the set D_q and start again its liquidation. At a first sight, it is not clear that v_n^k approximates v. However, we have the following result.

Proposition 5.1. *The sequence* $(v_n^k)_{n\ge 1, k\ge 1}$ *converges point-wise towards* v *on* $D_T \times D_q$ *when* $n \to \infty$ *and* $k \to \infty$.

The proof of Proposition 5.1 is given in supplementary manuscript.

Numerical results.
Figure 6 shows the value function v for different values of the elapsed time t and the remaining inventory Q_t. The panel (at the left) gives the reference value function. It is computed by following the same approach of Pierre and Charles-Albert (2016). The middle panel the value function obtained obtained after 120,000 iterations for RL algorithm where the step-size $(\gamma_k)_{k\ge 0}$ is derived from the upper level of our optimal policy, and the right panel the value function obtained with our optimal policy (i.e., upper level and inner level combined). It shows that our optimal strategy

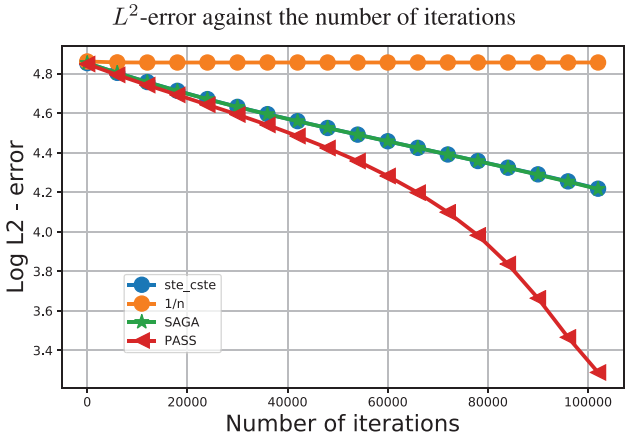

FIGURE 7 The log L^2-error against the number of iterations averaged over 1000 simulated paths. [Color figure can be viewed at wileyonlinelibrary.com]

leads to better performance results. We also plot, in Figure 7, a simulated path for the variations of the log L^2 error for different algorithms. Here again, we notice that our methodology improves the basic RL algorithm and that the ordering of other approaches is similar to the one of the "drift estimation" approximation (i.e., SAGA and the constant learning rate are very similar).

ACKNOWLEDGMENTS

None.

ORCID

Othmane Mounjid https://orcid.org/0000-0001-6480-8373
Charles-Albert Lehalle https://orcid.org/0000-0002-9978-9501

ENDNOTES

[1] We recall the following classical result: when Δ goes to zero, the value function and the optimal control of this problem converges towards the one where decisions are taken at any time.
[2] Here, we take $T = n^*\Delta$ with of $n^* \in \mathbb{N}^*$. Such an approximation is not restrictive.
[3] The maximizing action a^* for a state u is defined such that $a^* = \arg\max_{a\in\mathcal{A}} q(u, a)$.
[4] This waiting time may be very long depending on the dimension of the state space $\mathcal{Z}$ and the properties of the process $(Z_k)_{k\geq 0}$.
[5] Nonvisited coordinates are not modified. For example, we set $q_{k+1}(z) = q_k(z)$ for all $z \neq Z_k$.
[6] Here, M_0 is the zero function in the sense that $M_0[z, i] = 0$ for any $z \in \mathcal{Z}$ and $i \in \{1, \dots, M\}$.
[7] The sequence $(\nu_z)_{z\in\mathcal{Z}}$ is used to ensure that the error is bounded when needed.
[8] We do not allow controls that lead to states where the inventory exceeds $\bar{q}$.
[9] Note that the dependence of X^1 and X^2 on k is omitted since there is no possible confusion.

REFERENCES

Amrouni, S., Moulin, A., Vann, J., Vyetrenko, S., Balch, T., & Veloso, M. (2021). ABIDES-gym: Gym environments for multi-agent discrete event simulation and application to financial markets. In Proceedings of the Second ACM International Conference on AI in Finance, pp. 1–9.

Arulkumaran, K., Cully, A., & Togelius, J. (2019). AlphaStar: An evolutionary computation perspective. In *Proceedings of the genetic and evolutionary computation conference companion* (pp. 314–315).

Baldacci, B., Manziuk, I., Mastrolia, T., & Rosenbaum, M. (2022). Market Making and Incentives Design in the Presence of a Dark Pool: A Stackelberg Actor-Critic Approach. Operations Research.

Bartlett, P. L., Jordan, M. I., & McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, *101*(473), 138–156.

Bellman, R., & Dreyfus, S. (1959). Functional approximations and dynamic programming. *Mathematical Tables and Other Aids to Computation*, *13*, 247–251.

Benveniste, A., Métivier, M., & Priouret, P. (1987). Algorithmes adaptatifs et approximations stochastiques: théorie et applications à l'identification, au traitement du signal et à la reconnaissance des formes.

Benveniste, A., Métivier, M., & Priouret, P. (2012). *Adaptive algorithms and stochastic approximations* (Vol. 22). Springer Science & Business Media.

Bertsekas, D. P., & Tsitsiklis, J. N. (1996). *Neuro-dynamic programming* (Vol. 5). Athena Scientific Belmont.

Blum, J. R. (1954). Approximation methods which converge with probability one. *The Annals of Mathematical Statistics*, *25*(2), 382–386.

Bottou, L., & Bousquet, O. (2008). The tradeoffs of large scale learning. In *Advances in neural information processing systems* (pp. 161–168).

Bousquet, O., Boucheron, S., & Lugosi, G. (2003). Introduction to statistical learning theory. In *Advanced lectures on Machine Learning. Summer School on Machine Learning* (pp. 169–207). Springer.

Cardaliaguet, P., & Lehalle, C.-A. (2018). "Mean field game of controls and an application to trade crowding." *Mathematics and Financial Economics*, *12*, 335–363.

Cartea, A., Jaimungal, S., & Penalva, J. (2015). *Algorithmic and High-Frequency Trading (Mathematics, Finance and Risk)*. Cambridge University Press.

Combettes, P. L., & Pesquet, J.-C. (2011). Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering* (pp. 185–212). Springer.

Cong, L. W., Tang, K., Wang, J., & Zhang, Y. (2021). AlphaPortfolio: Direct construction through deep reinforcement learning and interpretable ai. *Available at SSRN 3554486*.

Cortes, C., Greenberg, S., & Mohri, M. (2019). Relative deviation learning bounds and generalization with unbounded loss functions. *Annals of Mathematics and Artificial Intelligence*, *85*(1), 45–70.

Defazio, A., Bach, F., & Lacoste-Julien, S. (2014). Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems* (pp. 1646–1654).

Gadat, S., & Panloup, F. (2017). Optimal non-asymptotic bound of the Ruppert-Polyak averaging without strong convexity. *arXiv preprint arXiv:1709.03342*.

Ganesh, S., Vadori, N., Xu, M., Zheng, H., Reddy, P., & Veloso, M. (2019). Reinforcement learning for market making in a multi-agent dealer market. *arXiv preprint arXiv:1911.05892*.

Guéant, O., & Manziuk, I. (2019). Deep reinforcement learning for market making in corporate bonds: Beating the curse of dimensionality. *Applied Mathematical Finance*, *26*(5), 387–452.

Hager, W. W., & Zhang, H. (2005). A new conjugate gradient method with guaranteed descent and an efficient line search. *SIAM Journal on Optimization*, *16*(1), 170–192.

Hendricks, D., & Wilcox, D. (2014). A reinforcement learning extension to the Almgren-Chriss framework for optimal trade execution. In *2014 IEEE conference on computational intelligence for financial engineering & economics (CIFEr)* (pp. 457–464). IEEE.

Hutchinson, J. M., Lo, A. W., & Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. *The Journal of Finance*, *49*(3), 851–889.

Johnson, R., & Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems* (pp. 315–323).

Karpe, M., Fang, J., Ma, Z., & Wang, C. (2020). Multi-agent reinforcement learning in a realistic limit order book market simulation. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1–7).

Kushner, H., & Yin, G. G. (2003). *Stochastic approximation and recursive algorithms and applications* (Vol. 35). Springer Science & Business Media.

Kushner, H. J., & Clark, D. S. (1978). *Stochastic approximation methods for constrained and unconstrained systems*. Springer.

Leal, L., Laurière, M., & Lehalle, C.-A. (2022). Learning a functional control for high-frequency finance. *Quantitative Finance*, *22*(11) 1973–1987.

Lehalle, C.-A., Laruelle, S., Burgot, R., Pelin, S., & Lasnier, M. (2018). *Market microstructure in practice*. World Scientific Publishing.

Lehalle, C.-A., & Mounjid, O. (2017). Limit order strategic placement with adverse selection risk and the role of latency. *Market Microstructure and Liquidity*, *3*, 1750009.

Manziuk, I., & Guéant, O. (2019). Deep reinforcement learning for market making in corporate bonds: Beating the curse of dimensionality.

Meyn, S. P., & Tweedie, R. L. (2012). *Markov chains and stochastic stability*. Springer Science & Business Media.

Moulines, E., & Bach, F. R. (2011). Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in neural information processing systems* (pp. 451–459).

Nevmyvaka, Y., Feng, Y., & Kearns, M. (2006). Reinforcement learning for optimized trade execution. In *Proceedings of the 23rd international conference on machine learning* (pp. 673–680).

Nitanda, A. (2014). Stochastic proximal gradient descent with acceleration techniques. In *Advances in neural information processing systems* (pp. 1574–1582).

Pierre, C., & Charles-Albert, L. (2016). Mean field game of controls and an application to trade crowding.

Ritter, G. (2017). Machine learning for trading. *Available at SSRN 3015609*.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The Annals of Mathematical Statistics*, *22*, 400–407.

Sacks, J. (1958). Asymptotic distribution of stochastic approximation procedures. *The Annals of Mathematical Statistics*, *29*(2), 373–405.

Schmidt, M., Roux, N. L., & Bach, F. R. (2011). Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems* (pp. 1458–1466).

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., & Hassabis, D. (2017). Mastering the game of go without human knowledge. *Nature*, *550*(7676), 354–359.

Sirignano, J., & Spiliopoulos, K. (2018). DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, *375*, 1339–1364.

Sutton, R. S., & Barto, A. G. (1998). *Introduction to reinforcement learning* (Vol. 135). MIT Press.

Tsybakov, A. B. (2004). Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, *32*(1), 135–166.

Vyetrenko, S., Byrd, D., Petosa, N., Mahfouz, M., Dervovic, D., Veloso, M., & Balch, T. (2020). Get real: Realism metrics for robust limit order book market simulations. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1–8).

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*.

Werbos, P. J. (1987). Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IEEE Transactions on Systems, Man, and Cybernetics*, *17*(1), 7–20.

How to cite this article: Mounjid, O., & Lehalle, C.-A. (2024). Improving reinforcement learning algorithms: Towards optimal learning rate policies. *Mathematical Finance*, *34,* 588–621. https://doi.org/10.1111/mafi.12378

APPENDIX A: PROOF OF PROPOSITION 3.2

Proof of Proposition 3.2. Let $z \in \mathcal{Z}$. Standard uniform convergence results ensure that

$$\mathbb{E}[\sup_q \left|M(q,z) - M^n(q,z)\right| \big| n(z)] \leq c \frac{1}{\sqrt{n(z) \wedge 1}}, \qquad a.s.$$

with $c > 0$ a positive constant. Since the Markov chain $(Z_n)_{n\geq 1}$ is irreducible and the set $\mathcal{Z}$ is finite, the sequence $(Z_n)_{n\geq 1}$ is positive recurrent and we have

$$\frac{n(z)}{n} = \frac{\sum_{k=1}^n \mathbf{1}_{Z_k=z}}{n} \underset{n\to\infty}{\to} \mathbb{P}_\mu[Z_n = z] > 0 = p(z), \quad a.s,$$

with μ the unique invariant distribution of $(Z_n)_{n\geq 1}$. Thus, we have

$$u_n(z) = \mathbb{E}\left[\sqrt{\frac{n}{n(z)\wedge 1}}\right] \underset{n\to\infty}{\to} \frac{1}{\sqrt{p(z)}} > 0.$$

This shows that $u_n(z)$ is bounded by a constant called $u_\infty(z)$ and ensures that

$$\mathbb{E}[\sup_q |M(q,z) - M^n(q,z)|(z)] \leq c_1(z)\frac{1}{\sqrt{n}}, \tag{A.1}$$

with $c_1(z) = cu_\infty(z)$. Since $\mathcal{Z}$ is finite, we close the proof by summing Inequality (A.1) over all the coordinates z. □

APPENDIX B: PROOF OF PROPOSITION 3.3

Proof of Proposition 3.3. Let $z \in \mathcal{Z}$. We follow the same approach used in the proof of Proposition 3.2 to get

$$v_n(z) = \mathbb{E}\left[\left(\frac{\log(\bar{n}(z))/\log(n)}{\bar{n}(z)/n}\right)^\beta\right] \leq \mathbb{E}\left[\left(\frac{1}{\bar{n}(z)/n}\right)^\beta\right] \underset{n\to\infty}{\to} \frac{1}{p(z)^\beta} > 0.$$

This shows that $v_n(z)$ is bounded by a constant $v_\infty(z)$ and ensures that

$$\mathbb{E}[\sup_q |M(q,z) - M^n(q,z)|(z)] \leq c_2(z)\left(\frac{\log(n)}{n}\right)^\beta,$$

with $c_2(z) = c'v_\infty(z)$. Using Equation (10), and the same manipulations used in the proof of Proposition 3.2 and Inequality (9), we complete the proof. □

APPENDIX C: PROOF OF PROPOSITION 4.7

Proof of Proposition 4.7. Let $k \geq 0$, A be the set $A = \{Z_k = z\}$, and $m(q_k) \in \mathbb{R}^{\mathcal{Z}}$ such that $m(q_k)(z') = m(q_k, X_{k+1}(z'), z')$ for any $z' \in \mathcal{Z}$. We split the proof into three cases. In each one of these steps, we prove Equation (14) for a given algorithm. □

Case (i):

In this step, we prove Equation (14) for Algorithm RL. Let us fix $z \in \mathcal{Z}$. For simplicity, we forget about the dependence of m, and q on z and write, respectively, $m(q_k)$, and q_k instead of $m(q_k)(z)$, and $q_k(z)$. We have

$$\begin{aligned}\mathbf{1}_A\mathbb{E}_k[(q_{k+1} - q^*)^2] &= \mathbf{1}_A\mathbb{E}_k[(q_k - \gamma_k m(q_k) - q^*)^2] \\ &= \mathbf{1}_A\Big\{(q_k - q^*)^2 \underbrace{-2\gamma_k\mathbb{E}_k[m(q_k)](q_k - q^*)}_{=(i)} + \gamma_k^2 \underbrace{\mathbb{E}_k[m(q_k)^2]}_{=(ii)}\Big\}.\end{aligned}$$

Using Assumption 4.4 and $\mathbb{E}_k[m^*] = 0$, we get $(i) \leq -2L\gamma_k|q_k - q^*|^2$. Since $\mathbb{E}_k[m^*] = 0$, Assumption 4.5 gives

$$\begin{aligned}(ii) &= \mathbb{E}_k\left[(m(q_k) - \mathbb{E}_k[m(q_k)])^2\right] + (\mathbb{E}_k[m(q_k) - m^*])^2 \\ &\leq \mathbb{E}_k\left[(m(q_k) - \mathbb{E}_k[m(q_k)])^2\right] + B(1 + (q_k - q^*)^2).\end{aligned}$$

We use now two independent copies of X_k, respectively, denoted by X^1 and X^2[9]. We also write $m(q_k)_X = m(q_k, X(z'), z')$ to emphasize the dependence of $m(q_k)$ on X. Using Jensen's inequality, and Assumption 4.5, we get

$$\begin{aligned}\mathbb{E}_k\left[(m(q_k)_{X^1} - \mathbb{E}_k[m(q_k)_{X^2}])^2\right] &= \mathbb{E}_k\left[(\mathbb{E}_k[m(q_k)_{X^1} - m(q_k)_{X^2}])^2\right] \\ &\underbrace{\leq}_{\text{Jensen's inequality}} \mathbb{E}_k\left[(m(q_k)_{X^1} - m(q_k)_{X^2})^2\right] \\ &\leq 3\Big(\mathbb{E}_k\left[(m(q_k)_{X^1} - m^*(q_k)_{X^1})^2\right] + \mathbb{E}_k\left[(m^*(q_k)_{X^2} - m^*(q_k)_{X^1})^2\right] \\ &\quad + \mathbb{E}_k\left[(m^*(q_k)_{X^2} - m^*(q_k)_{X^2})^2\right]\Big) \underbrace{\leq}_{\text{Assumption 4}} 3B(1 + \upsilon_k).\end{aligned}$$

Thus, we deduce that

$$\mathbf{1}_A\mathbb{E}_k[(q_{k+1} - q^*)^2(z)] \leq \mathbf{1}_A\Big\{(1 \underbrace{-2\gamma_k L + B\gamma_k^2}_{=-p(\gamma_k)})(q_k - q^*)^2(z) + \underbrace{\gamma_k^2(z)B(4 + 3\upsilon_k)}_{=M_k}\Big\},$$

which shows Equation (14) for Algorithm RL.

Case (ii):

Here we show Equation (14) for Algorithm SAGA. Let $z \in \mathcal{Z}$, and $\bar{M}^k(z) = (\sum_{j=1}^M M^k[z, j])/M$. We forget here about the dependence of m, q, and M on z and write, respectively, $m(q_k)$, q_k, and $M^k[j]$ instead of $m(q_k)(z)$, $q_k(z)$, and $M^k[z, j]$. Using $\mathbb{E}_k[m^*] = 0$ and $\mathbb{E}_k[M^k[i]] = \bar{M}^k$, we have

$$\begin{aligned}\mathbf{1}_A\mathbb{E}_k\left[(q_{k+1} - q^*)^2\right] &= \mathbf{1}_A\left\{(q_k - q^*)^2 + 2(\mathbb{E}_k[q_{k+1}] - q_k)(q_k - q^*) + \mathbb{E}_k[(q_{k+1} - q_k)^2]\right\} \\ &= \mathbf{1}_A\left\{(q_k - q^*)^2 - 2(\gamma_k\mathbb{E}_k[m(q_k) - m^*])(q_k - q^*)\right. \\ &\quad \left. + \gamma_k^2\mathbb{E}_k\left[\left(m(q_k) - M^k[i] + \bar{M}^k\right)^2\right]\right\} \\ &\underbrace{\leq}_{\text{Assumption 4.4}} \mathbf{1}_A\Big\{(1 - 2L\gamma_k)(q_k - q^*)^2 + \gamma_k^2 \underbrace{\mathbb{E}_k\left[\left(m(q^k) - M^k[i] + \bar{M}^k\right)^2\right]}_{=(1)}\Big\}.\end{aligned} \tag{C.2}$$

We first dominate the term (1). Since $\mathbb{E}_k[m^*](z) = 0$ and

$$\mathbb{E}_k\left[\left(M^k[i] - m^*\right)^2\right] = 1/M \sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right],$$

we have

$$\begin{aligned}
(1) \quad &= \mathbb{E}_k\left|(m(q_k) - \mathbb{E}_k[m^*]) - \left(M^k[i] - m^*\right) + \left(1/M \textstyle\sum_j (M^k[z,j] - m^*(z))\right)\right|^2 \\
&\leq 3\left[\mathbb{E}_k[m(q_k) - m^*]^2 + \mathbb{E}_k\left[\left(M^k[i] - m^*\right)^2\right] + \mathbb{E}_k\left[\left(1/M \textstyle\sum_j (M^k[j] - m^*\right)\right]^2\right] \\
&\underbrace{\leq}_{\text{Jensen's inequality}} \quad 3\left[\mathbb{E}_k\left[(m(q_k) - m^*)^2\right] + \mathbb{E}_k\left[\left(M^k[i] - m^*\right)^2\right] + 1/M \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right] \\
&= 3\left[\mathbb{E}_k\left[(m(q_k) - \mathbb{E}_k[m(q_k)])^2\right] + (\mathbb{E}_k[m(q_k) - m^*])^2 + 2/M \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right] \\
&\underbrace{\leq}_{\text{Assumption 4.5}} \quad 3\left[B\left(4 + 3\upsilon_k + (q_k - q^*)^2\right) + 2/M \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right] \\
&= 3B(4 + 3\upsilon_k) + 3B(q_k - q^*)^2 + 6/M \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right].
\end{aligned} \tag{C.3}$$

By combining Equations (C.2) and (C.3), we get

$$\mathbf{1}_A \mathbb{E}_k[|q_{k+1} - q^*|^2] \leq \mathbf{1}_A\left\{(1 - 2\gamma_k L + 3B\gamma_k^2)(q_k - q^*)^2 + 6/M \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right] + 3B\gamma_k^2(3\upsilon_k + 4)\right\}. \tag{C.4}$$

Moreover, we have

$$\begin{aligned}
\mathbf{1}_A 1/M \mathbb{E}_k\left[\textstyle\sum_{j=1}^M \left(M^{k+1}[j] - m^*\right)^2\right] &= \mathbf{1}_A\left\{\frac{1}{M}\mathbb{E}_k\left[(m(q_k) - m^*)^2\right]\right. \\
&\quad \left. +(1 - \tfrac{1}{M})\tfrac{1}{M} \textstyle\sum_{j=1}^M \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right\} \\
&\leq \mathbf{1}_A\left\{\frac{B}{M}\left(1 + (q_k - q^*)^2\right)\right. \\
&\quad \left. +(1 - \tfrac{1}{M})\tfrac{1}{M} \textstyle\sum_{j=1}^M \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right\}.
\end{aligned} \tag{C.5}$$

Thus, using Equations (C.4) and (C.5), we conclude

$$\begin{aligned}
\mathbf{1}_A \mathbb{E}_k[e^{k+1}] &\leq \mathbf{1}_A\left\{\underbrace{(1 - 2\gamma_k L + 3B\gamma_k^2 + \frac{B}{Mc})}_{=\alpha_1} c(q_k - q^*)^2\right. \\
&\quad \left. + \underbrace{(1 - \frac{1}{M} + 6\gamma_k^2 c)}_{\alpha_2} \tfrac{1}{M} \textstyle\sum_j \mathbb{E}_k\left[\left(M^k[j] - m^*\right)^2\right]\right\} + 3Bc\gamma_k^2(3\upsilon_k + 4) \\
&\leq \mathbf{1}_A \alpha e^k + 3cB\gamma_k^2(3\upsilon_k + 4),
\end{aligned}$$

with $\alpha = \max(\alpha_1, \alpha_2) \in [0, 1)$.

Case (iii):
In this final step, we show Equation (14) for Algorithm PASS. Here again, we forget about the dependence of the variables on z as in the previous steps. We have

$$\mathbf{1}_A\mathbb{E}_k[(q_{k+1}-q^*)^2] = \mathbf{1}_A\mathbb{E}_k[(q_k-\hat{\gamma}_k m(q_k)-q^*)^2]$$
$$= \mathbf{1}_A\Big\{(q_k-q^*)^2 \underbrace{-2\mathbb{E}_k[\hat{\gamma}_k m(q_k)](q_k-q^*)}_{=(i)} + \underbrace{\mathbb{E}_k[\hat{\gamma}_k^2(m(q_k))^2]}_{=(ii)}\Big\}.$$

For the term (i), using Assumption 4.4 and $\mathbb{E}_k[m^*]=0$, we have $(i)\le -2c_k\bar{\gamma}_k(q_k-q^*)^2$ with $c_k = \frac{\mathbb{E}_k[\hat{\gamma}_k m(q_k)]}{\bar{\gamma}_k\mathbb{E}_k[m(q_k)]}$. Using Assumption 4.5, and $\mathbb{E}_k[m^*]=0$, we get

$$\begin{aligned}(ii) = c_k^2\bar{\gamma}_k^2\mathbb{E}_k\left[m(q_k)^2\right] &= c_k^2\bar{\gamma}_k^2\Big(\mathbb{E}_k\Big[(m(q_k)-\mathbb{E}_k[m(q_k)])^2\Big] + (\mathbb{E}_k[m(q_k)-m^*])^2\Big)\\ &\le c_k^2\bar{\gamma}_k^2\big(B_k(4+3\upsilon_k)+B_k(1+(q_k-q^*)^2)\big).\end{aligned}$$

Thus, we deduce that

$$\mathbf{1}_A\mathbb{E}_k[(q_{k+1}-q^*)^2] \le \mathbf{1}_A\Big\{(1\underbrace{-2c_k\bar{\gamma}_k L_k + B_k(c_k\bar{\gamma}_k)^2}_{=-p_k(c_k\bar{\gamma}_k)})(q_k-q^*)^2 + \underbrace{c_k^2\bar{\gamma}_k^2 B_k(4+3\upsilon_k)}_{=M_k}\Big\}. \quad \text{(C.6)}$$

We write $\underline{\gamma}_k$ for the quantity $\underline{\gamma}_k = c_k\bar{\gamma}_k \wedge \bar{\gamma}_k$. Since $\hat{\gamma}_k \in [\gamma_k, r_1\gamma_k]$, we have $c_k\bar{\gamma}_k \in [\gamma_k, r_1\gamma_k]$. When $c_k\bar{\gamma}_k \in]\gamma_k, \bar{\gamma}_k]$, we have $p_k(\underline{\gamma}_k) = p_k(c_k\bar{\gamma}_k) > p_k(\gamma_k) \ge 0$. When $c_k\bar{\gamma}_k \in]\bar{\gamma}_k, r_1\gamma_k]$ (i.e., $c_k \ge 1$), we use the following canonical decomposition of the function p_k:

$$p_k(x) = B_k(x-\bar{\gamma}_k)^2 - \left(\frac{L_k}{B_k}-1\right),$$

and $\underline{\gamma}_k = \bar{\gamma}_k$ to get

$$|p_k(c_k\bar{\gamma}_k) - p_k(\underline{\gamma}_k)| = |p_k(c_k\bar{\gamma}_k) - p_k(\bar{\gamma}_k)| \le B_k(c_k\bar{\gamma}_k-\bar{\gamma}_k)^2 \le B_k(r_1\gamma_k-\gamma_k)^2 = \gamma_k^2 d^1,$$

with $d^1 = (r_1-1)^2 B_k$. Thus, using Equation (C.6), we conclude

$$\mathbf{1}_A\mathbb{E}_k[(q^{k+1}-q^*)^2] \le \mathbf{1}_A\Big\{(1-p_k(\underline{\gamma}_k)+\gamma_k^2 d^1\mathbf{1}_{c_k\ge 1})(q^k-q^*)^2 + \underbrace{c_k^2\bar{\gamma}_k^2(z)B_k(4+3\upsilon_k)}_{=M_k}\Big\}.$$

This completes the proof.

APPENDIX D: PROOF OF THEOREM 4.8

For simplicity, the proof is split into two parts.

D.1 | Proof of Equation (17)

Proof of Equation 4.8. Using Proposition 4.7, we get

$$\begin{aligned}\mathbb{E}_n[e^{n+1}] &\leq e^n + \mathbf{1}_A(-\mu_n e^n + M_n)\\ &\leq e^n - R_n + L_n,\end{aligned}$$

with $R_n = \mathbf{1}_A \mu_n e^n$, and $L_n = \mathbf{1}_A M_n$. Using the assumption $\sum_{n\geq 1} \gamma_n^2 < \infty$, and the expression of M_n, we obtain that $\sum_{n\geq 1} L_n < \infty$. We can then apply the supermartingale convergence theorem, to deduce that e^n converges towards a random variable with probability 1, and $\sum_{n\geq 1} R_n < \infty$. Since $\sum_{n\geq 1} \gamma_n^2 < \infty$, we know that γ_n converges towards 0. Thus, for n large enough, we have

$$\mu_n \geq 2L\gamma_n - B\gamma_n^2 \geq L\gamma_n.$$

Replacing R_n and μ_n by their expressions gives

$$R_n = \mathbf{1}_A \mu_n e^n \geq \mathbf{1}_A L\gamma_n e^n,$$

If there were n_1, and $\Delta > 0$, such that

$$e^n \geq \Delta, \qquad \forall n \geq n_1,$$

this would contradict the property $\sum_{n\geq 1} R_n < \infty$ since $\sum_{n\geq 1} \gamma_n = +\infty$. Thus, we deduce that e^n converges towards 0. □

D.2 | Proof of Inequalities (18) and (19)

Preparation for the proof of Inequality (8)

We introduce the following notations. Let $j \in \mathbb{N}^*$ and $(\mu_n, a_n, b_n)_{n\geq 1}$ be a sequence valued in $\mathbb{R}_+^3$. We write $(\mu, b)^j = (\mu_n^j, b_n^j)_{n\geq 1}$ for the delayed sequence $\mu_n^j = \mu_{j+n}$ and $b_n^j = b_{n+(j-1)}$ with $n \geq 1$. Additionally, we define recursively the sequence $(a_n^{\mu,b})_{n\geq 1}$ as follows:

$$a_1^{\mu,b} = 1, \qquad \text{and} \qquad a_{n+1}^{\mu,b} = \mu_{n+1} \sum_{l=1}^{n} a_{n+1-l} b_l a_l^{\mu,b}, \qquad \forall n \geq 1. \tag{D.7}$$

Lemma D.1. *By convention, an empty sum is equal to zero. Let $(\upsilon_n)_{n\geq 1}$ be the sequence defined as follows:*

$$\upsilon_n = \epsilon_n + \mu_n \left(\sum_{j=1}^{n-1} a_{n-j} b_j \upsilon_j \right), \qquad \forall n \geq 1,$$

where $(\epsilon_n)_{n\geq 1}$ is a sequence. Then, we have

$$v_n = \sum_{j=1}^{n} a_{n+1-j}^{(\mu,b)^j} \epsilon_j, \qquad \forall n \geq 1. \tag{D.8}$$

Lemma D.2. *Let $n \in \mathbb{N}$, $(a_n^{\mu,b})_{n\geq 0}$ be the sequence defined in Equation (D.7), $(\mu_n)_{n\geq 0}$ be a positive nondecreasing sequence, $r_n = 1 - \mu_n$, $\sum_n a_n \leq 1$, and $\sup_n b_n \leq 1$.*

- *When $\sum_{n\geq 0} r_n = +\infty$, we have*

$$a_n^{\mu,b} \underset{n\to\infty}{\to} 0. \tag{D.9}$$

- *There exists a non-negative constant B such that*

$$a_n^{\mu,b} \leq B \sum_{k=1}^{n-1} \bar{\mu}_k^n a_{n-k}^{*\infty}, \qquad \forall n \geq 2, \tag{D.10}$$

with $\bar{\mu}_k^n = e^{-\sum_{j=n-k+1}^{n} r_j}$ and $a_k^{*\infty} = \lim_{n\to\infty} a_k^{*n}$. The sequence $(a_k^{*n})_{k\geq 1}$ is defined recursively such that $a_k^{*1} = a_k$ and $a_k^{*(n+1)} = \sum_{l=1}^{k} a_{k+1-l} b_{n-k+l} a_l^{*n}$ for all $k \geq 1$.

Propagation of the error

We introduce the following notations. Let $n \in \mathbb{N}^*$, and $z^1 \in \mathcal{Z}$. We write $\tau_{z_1} = \inf\{l > 0,\ Z_l = z_1\}$, and $a_k = \mathbb{P}[\tau_{z_1} \geq k, |Z_0 = z_1]$. We have the following result.

Proposition D.3. *Let $z_1 \in \mathcal{Z}$, and $n \in \mathbb{N}^*$. Under Assumptions 3.1, 4.4, 4.5, and 4.6, we have*

$$\mathbb{E}_0[e^n(z_1)] \leq \epsilon_n + \sum_{j=1}^{n-1} a_{n-j} b_j \mathbb{E}_0[e^j(z_1)],$$

with $\epsilon_n = e^1(z_1)a_n(z_1) + \sum_{j=1}^{n-1} a_{n-j}\mathbb{E}[M_j]$ and $b_j = \dfrac{\mathbb{E}_0[\alpha_j\, e^j(z_1)]}{\mathbb{E}_0[e^j(z_1)]}$. The variables α_j and M_j are given by Equations (15) and (16).

Proof of Proposition D.3. Using the last-exit decomposition, see Section 8.2.1 in Meyn and Tweedie (2012), we have

$$\begin{aligned}
\mathbb{E}_0[e^n(z_1)] &= \mathbb{E}_0[e^n(z_1)\mathbf{1}_{\tau_{z_1}\geq n, A^n}] + \textstyle\sum_{j=1}^{n-1} \mathbb{E}_0[e^n(z_1)\mathbf{1}_{\{\tau_{z_1}\geq n-j, Z_j = z_1\}}] \\
&= \mathbb{E}_0[e^1(z_1)\mathbf{1}_{\tau_{z_1}\geq n}] + \textstyle\sum_{j=1}^{n-1} \mathbb{E}_0[e^{j+1}(z_1)\mathbf{1}_{\{\tau_{z_1}\geq n-j, Z_j = z_1\}}] \\
&\leq \mathbb{E}_0[e^1(z_1)\mathbf{1}_{\tau_{z_1}\geq n}] + \textstyle\sum_{j=1}^{n-1} \mathbb{P}[\tau_{z_1} \geq n-j | Z_j = z_1]\mathbb{E}_0[e^{j+1}(z_1)\mathbf{1}_{\{Z_j = z_1\}}] \\
&\underbrace{\leq}_{\text{Proposition 4.7}} \mathbb{E}_0[e^1(z_1)\mathbf{1}_{\tau_{z_1}\geq n, A^n}] + \textstyle\sum_{j=1}^{n-1} \mathbb{P}[\tau_{z_1} \geq n-j | Z_j = z_1]\mathbb{E}_0[(\alpha_j e^j(z_1) + M_j)] \\
&\leq \epsilon_n + \textstyle\sum_{j=1}^{n-1} a_{n-j} b_j \mathbb{E}_0[e^j(z_1)],
\end{aligned}$$

with ϵ_n, a_j, and b_j defined in Proposition D.3. The variables α_n and M_n are defined in Proposition 4.7. In the second equality, we use that $e^n(z_1)$ does not change as long as the state z_1 is not reached. This completes the proof. □

Proof of Inequalities (18) and (19)

Proof of Theorem 4.8. We split the proof in two steps. In Step (i), we show Equation (18) and then in Step (ii), we show Equation (19). □

Step (i):. In this part, we first prove Equation (18) when the space $\mathcal{Z}$ is finite. Then, we show how to extend Equation (18) to the general case.

Substep (i-1):. Let us demonstrate Equation (17) when the space $\mathcal{Z}$ is finite. We define υ_n by $\upsilon_n = b_n \mathbb{E}[e^n(z_1)]$, and $r = \sum_{j\geq 1} a_j \leq \mathbb{E}[\tau_{z_1}] < \infty$. Using Proposition D.3, the sequence υ_n verifies

$$\upsilon_n \leq \bar{\epsilon}_n + \mu_n \sum_{j=1}^{n-1} \bar{a}_{n-j} \upsilon_j(z_1),$$

with $\bar{\epsilon}_n(z_1) = b_n \epsilon_n$, $\mu_n = b_n$, and $\bar{a}_j = a_j / r$. Thus, using Lemma D.1, we get

$$\upsilon_n \leq \sum_{j=1}^{n} \bar{a}_{n+1-j}^{(\mu,b)^j} \bar{\epsilon}_j, \tag{D.11}$$

with $(\bar{a}_n^{(\mu,b)})_{n\geq 1}$ defined in Equation (D.7). We recall that $\epsilon_n = e^1 \bar{a}_n + \sum_{k=1}^{n-1} \bar{a}_{n-k} b_k \mathbb{E}[M_k]$.

Let us prove $\sum_{k\geq 1} \epsilon_k < \infty$. For this, we assimilate the sequence $\bar{\epsilon}$ to the measure $\mu = \sum_{k\geq 1} \bar{\epsilon}_k \delta_k$ with δ_k the Dirac measure at k and recall that $\epsilon_n = e^1 \bar{a}_n + \sum_{k=1}^{n-1} \bar{a}_{n-k} \mathbb{E}[M_k]$. Then, we introduce the variable $\tau_{z_1} = \inf\{l > 0, Z_l = z_1\}$ already defined in Section D.2.2. By definition of $\bar{a}_{n-k}$, we have $\sum_{n\geq 1} \bar{a}_n = 1 < \infty$. Given that $\sum_k \mathbb{E}[\gamma_k^2] < \infty$, and $\mathbb{E}[M_k] = O(\mathbb{E}[\gamma_k^2])$ for all the algorithms, we also get

$$\sum_k \mathbb{E}[M_k] < \infty. \tag{D.12}$$

We deduce from Equation (D.12) that

$$\mathbb{E}[\sum_{n\geq 1} \epsilon_k] \leq e^1 \mathbb{E}[\textstyle\sum_{n\geq 1} \bar{a}_n] + \sum_{k\geq 1} \mathbb{E}[M_k] \Big(\mathbb{E}[\textstyle\sum_{n\geq k} \bar{a}_{n-k}] \Big) \leq e^1 + \sum_{k\geq 1} \mathbb{E}[M_k] < \infty.$$

This ensures that $\sum_k \epsilon_k < \infty$ and shows that the measure μ has a finite mass.

Now, Lemma D.2 gives $a_n^{(\mu,b)^j} \underset{n\to\infty}{\to} 0$, for any $j \geq 1$. Thus, the dominated convergence theorem ensures that $\upsilon_n \underset{n\to\infty}{\to} 0$. Since α_n converges towards 1, b_n converges towards 1 as well, which means that $\mathbb{E}[e^n(z_1)] \underset{n\to\infty}{\to} 0$. Since the space $\mathcal{Z}$ is finite, we deduce that $\mathbb{E}[E^n] \underset{n\to\infty}{\to} 0$.

Substep(i-2):. In this second step, we show Equation (18) when the space $\mathcal{Z}$ is countable. Let $\epsilon > 0$. Since $\mathbb{E}[E^1] < \infty$, there exists $k_0 \in \mathbb{N}$ such that

$$\sum_{k \geq k_0} \mathbb{E}[e^1(z_k)]\nu_k < \frac{\epsilon}{2}.$$

We write A_{k_0} for the set $A_{k_0} = \{z_k,\ k \leq k_0\}$. Since A_{k_0} is finite, we use **substep(i-1)** to show the existence of $k_1 \in \mathbb{N}$ such that

$$\sum_{k \leq k_0} \mathbb{E}[e^{k_1}(z_k)]\nu_k < \frac{\epsilon}{2}, \quad \forall k \geq k_1, \forall z_k \in A_{k_0}.$$

We take now $k \geq k_1$. Using $(\mathbb{E}[e^l(z)])_{l \geq 1}$ is nonincreasing for any $z \in \mathcal{Z}$, we get $E^k = \sum_{k' \geq k_0} \mathbb{E}[e^k(z_{k'})]\nu_{k'} + \sum_{k' < k_0} \mathbb{E}[e^k(z_{k'})]\nu_{k'} \leq \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon$.

Step (ii):. In this step, we show Equation (19). By applying Lemma D.2, we obtain the existence of a constant B such that

$$\bar{a}_n^{(\mu,b)} \leq B \sum_{k=1}^{n-1} \bar{\mu}_k^n \bar{a}_{n-k}^{*\infty},$$

with $\bar{\mu}_k^n$ and $\bar{a}^{*\infty}$ defined in Equation (D.10). We can then use Equation (D.11), to get

$$v_n \leq B \sum_{j=1}^{n} \sum_{l=1}^{j} \bar{\mu}_k^n \bar{a}_{l-j}^{*\infty} \bar{\epsilon}_j.$$

Since α_n converges towards 1 almost surely, the variable b_n is bounded from below and thus there exists a constant B' such that

$$\mathbb{E}[e^n(z_1)] \leq B' \sum_{j=1}^{n} \sum_{l=1}^{j} \bar{\mu}_k^n \bar{a}_{l-j}^{*\infty} \bar{\epsilon}_j.$$

This completes the proof.

APPENDIX E: PROOF OF PROPOSITION 4.9

Proof of Proposition 4.9. For simplicity, we prove the result for Algorithm 4.1; however, the same argument holds for Algorithm 4.3. Let us prove by induction on $n \in \mathbb{N}$ that

$$e_\gamma^n \leq e_{\tilde{\gamma}}^n, \quad a.s, \quad \forall \tilde{\gamma} \in \Gamma. \tag{E.13}$$

For $n = 0$, Inequality (E.13) is directly satisfied since the initial error e^0 does not depend on the choice of the learning rate. Let $\tilde{\gamma} \in \Gamma$. We assume now that $e_\gamma^n \leq e_{\tilde{\gamma}}^n$, $a.s$. Using Equation (20) and the definition of $(\gamma_n)_{n \geq 0}$, we have

$$e_\gamma^{n+1} = \mathbf{1}_A\big(g(e_\gamma^n) + S_n\big) + \mathbf{1}_{A^c} e_\gamma^n, \tag{E.14}$$

with $g(x) = x - \dfrac{L^2x^2}{2B(x+(2+\upsilon_n))}$. The previous expression of g is obtained by minimizing the function $y :\to (1-2Ly+By^2)e_\gamma^n + B(2+\upsilon_n)\times y^2$. This means that

$$g(x) \le (1-2Ly+By^2)x + M_n y^2, \quad \forall y \in \mathbb{R}, \forall x \in \mathbb{R}_+. \tag{E.15}$$

A study of the function g shows that it is nondecreasing on the interval $[0, x_2[$ with $x_2 = \arg\sup_{x\in\mathbb{R}_+} g(x)$ and $g(x) \le x$ for any $x \in \mathbb{R}_+$. Thus, we have necessarily $\mathbf{1}_A e_{\tilde\gamma}^n \le \mathbf{1}_A g(e_{\tilde\gamma}^{n-1}) \le g(x_2) \le x_2$ for any $n \ge 1$, and $\tilde\gamma \in \Gamma$. Note that when $x_2 = +\infty$, we do not need to assume that $\mathbf{1}_A S_n \le 0$. Using that $e_\gamma^n \le e_{\tilde\gamma}^n$ a.s, the monotonicity of g on $[0, x_2[$, and Equation (E.15), we get

$$\begin{aligned} \mathbf{1}_A e_\gamma^{n+1} = \mathbf{1}_A g(e_\gamma^n) + \mathbf{1}_A S_n \quad &\le \mathbf{1}_A g(\mathbf{1}_A e_{\tilde\gamma}^n) + \mathbf{1}_A S_n \\ &\le (1-2L\tilde\gamma_n + B\tilde\gamma_n^2)\mathbf{1}_A e_{\tilde\gamma}^n + \mathbf{1}_A M_n \tilde\gamma_n^2 + \mathbf{1}_A S_n \\ &= \mathbf{1}_A e_{\tilde\gamma}^{n+1}, \quad a.s. \end{aligned} \tag{E.16}$$

We complete the proof by combining Equations (E.14), (E.16), and the induction assumption. □