Journal of Machine Learning Research 23 (2022) 1-20 Submitted 4/20; Published 4/22

Exact simulation of diffusion first exit times: algorithm acceleration.

Samuel Herrmann Samuel.Herrmann@u-bourgogne.fr
Institut de Mathématiques de Bourgogne (IMB) - UMR 5584, CNRS,
Université de Bourgogne Franche-Comté
F-21000 Dijon, FRANCE

Cristina Zucca cristina.zucca@unito.it
Department of Mathematics 'G. Peano'
University of Torino, Via Carlo Alberto 10
10123 Turin, ITALY

Editor: Peter Auer

Abstract

In order to describe or estimate different quantities related to a specific random variable, it is of prime interest to numerically generate such a variate. In specific situations, the exact generation of random variables might be either momentarily unavailable or too expensive in terms of computation time. It therefore needs to be replaced by an approximation procedure. As was previously the case, the ambitious exact simulation of first exit times for diffusion processes was unreachable though it concerns many applications in different fields like mathematical finance, neuroscience or reliability. The usual way to describe first exit times was to use discretization schemes, that are of course approximation procedures. Recently, Herrmann and Zucca (Herrmann and Zucca, 2020) proposed a new algorithm, the so-called GDET-algorithm (General Diffusion Exit Time), which permits to simulate exactly the first exit time for one-dimensional diffusions. The only drawback of exact simulation methods using an acceptance-rejection sampling is their time consumption. In this paper the authors highlight an acceleration procedure for the GDET-algorithm based on a multi-armed bandit model. The efficiency of this acceleration is pointed out through numerical examples.

Keywords: Exit time, Brownian motion, diffusion processes, rejection sampling, exact simulation, multi-armed bandit, randomized algorithm.

Introduction

A precise description of the first time a given stochastic process exits from a domain is required in many mathematical applications: it can for instance be related to the evaluation of risk of default in mathematical finance or to the description of spike trains in neuroscience,... Unfortunately, in the diffusion framework (solutions of stochastic differential equations) a simple and explicit expression of the first exit time distribution is not attainable except in a few specific cases. It is therefore challenging to find out how to generate such variates. One way to overcome this issue is to introduce an algorithm based on an approximation procedure. Several studies are for instance based on a discretization scheme for the corresponding stochastic differential equation. Most of them are based on improvements of the

classical Euler scheme (see for instance (Broadie et al., 1997), (Gobet and Menozzi, 2010), (Gobet, 2000)) which essentially consists in reducing the error stem from the approximation procedure. Another way to deal with the distribution of first exit times consists in approximating their probability density functions and thus in approximating the solution of an integral equation (Sacerdote et al., 2014).

Apart from all these approximation procedures, Herrmann and Zucca (Herrmann and Zucca, 2020) proposed an exact simulation of diffusion first exit times based on an acceptance-rejection method. The method is directly linked to the Girsanov transformation, a crucial tool already used for the exact simulation of diffusion paths on a fixed time interval (Beskos et al., 2006; Beskos and Roberts, 2005) or for the simulation of first passage times (Herrmann and Zucca, 2019). It is impossible to reasonably compare the numerical methods listed so far since they are of very different types. On the one hand, approximation methods are fast but induce small errors to be controlled. On the other hand, exact method are rather time-consuming.

The aim of this paper is to improve and accelerate the algorithm presented in (Herrmann and Zucca, 2020) which permits to generate numerically the first exit time and exit location of a diffusion process from a given interval $[a, b]$. Let us consider the stochastic process $(X_t,\ t \geq 0)$, solution of the SDE:

$$dX_t = \mu(X_t)dt + \sigma(X_t)dB_t, \quad X_0 = x \in (a, b) \quad \text{with} \quad -\infty < a < b < \infty \tag{1}$$

where $(B_t,\ t \geq 0)$ stands for the standard one-dimensional Brownian motion, $\sigma \in \mathcal{C}^3([a; b])$ is a positive function on the whole interval $[a, b]$ and $\mu \in \mathcal{C}^2([a; b])$. In the particular case when the function σ is constant, we can relax the hypothesis on μ and we just take $\mu \in \mathcal{C}^1([a; b])$. These regularity conditions are required for the use of classical tools in stochastic calculus: the algorithm is essentially based on a suitable combination of Itô's lemma and Girsanov's transformation. We denote by $\tau_{a,b}$ the first time the diffusion exits from the interval $[a, b]$:

$$\tau_{a,b}(X) := \inf\{t > 0 :\ X_t \notin [a, b]\}. \tag{2}$$

Let $T > 0$ be a constant. We call BoxExit$(x, [a, b], T)$ the efficient algorithm which permits to simulate exactly the random vector $(\tau_{a,b}(X) \wedge T, X_{\tau_{a,b}(X)\wedge T})$, that is the first time the path of the diffusion process $(X_t,\ t \geq 0)$ exits from the time-space rectangle $[0, T] \times [a, b]$ and its associated location. A simple and unified version of this algorithm is presented in Section 1, Figure 2 (it corresponds to the algorithms DET and κ-DET introduced in (Herrmann and Zucca, 2020)).

Of course BoxExit is only a basic component for the exit problem from the interval $[a, b]$: the authors suggested in (Herrmann and Zucca, 2020) to use the Markov property of the time-homogeneous diffusion (1) in order to simulate $\tau_{a,b}(X)$. More precisely, the iteration procedure is initialized by $Z_0 = x$, the starting position of the diffusion. Then the sequence defined by

$$(\mathcal{T}_{n+1}, Z_{n+1}) \leftarrow \text{BoxExit}(Z_n, [a, b], T)$$

and stopped as soon as Z_n reaches either the value a or b permits to generate the couple $(\tau_{a,b}(X), X_{\tau_{a,b}(X)})$. The efficiency (time consumption) is just related to the unique parameter T since the size of the time-space rectangle associated to the basic component is $[0, T] \times [a, b]$.

The main idea of the acceleration procedure is to choose in an optimal way the box size related to the basic components. Instead of fixing the elementary box size equal to $[0,T]\times[a,b]$ ($[a,b]$ being the interval of the initial problem), we propose to cover the interval $]a,b[$ by a fixed number (denoted $N-1$ in Section 2) of slices of identical width: $]a,b[=\cup_{i=1}^{N-1} I_i$ and to successively use the basic components $\textsc{BoxExit}(\cdot, I_i, T)$ associated to the family of box sizes $([0,T]\times I_i)_{1\leq i\leq N-1}$ until the exit of the interval $]a,b[$ occurs. In other words, we introduce a random walk on small rectangles and stop it as soon as it reaches either a or b, see Figure 1. At first glance, such a procedure seems to slow down the exact simulation of the exit time since we introduce a new random walk and increase the number of appeals to basic components. But the observation reveals something surprising: for suitable choices of parameters N and T, the introduction of the random walk effectively speeds up the algorithm. It is less time-consuming for a diffusion process to exit from boxes of intermediate size compared to boxes of small or large size due to the acceptance-rejection method. This simple argument partly explains the over-performance of the modified algorithm. It is

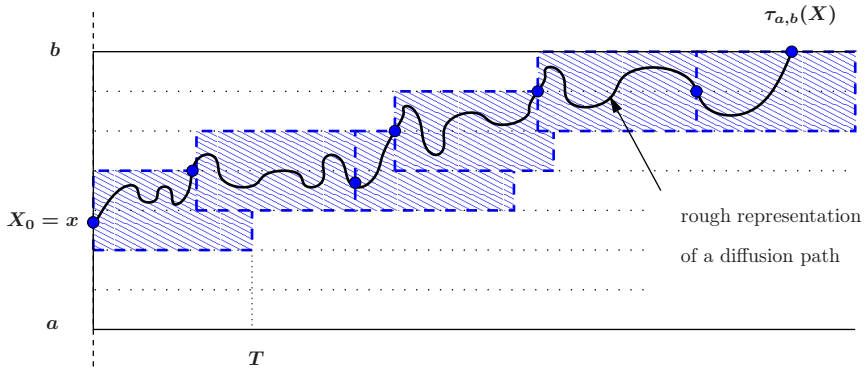


Figure 1: Diffusion path and associated random walk on rectangles

therefore challenging to find the optimal parameters T and N in order to obtain the most efficient algorithm. Instead of considering in detail all families of diffusion processes and determining the best choice of parameters on a case-by-case basis, we prefer to propose a randomized algorithmic approach. We find a reasonable value of T and choose N with a multi-armed bandit method (ϵ-greedy algorithm). Such general method can be applied to any diffusion process.

The material is organized as follows: in Section 1, we present a unified and simple version for the exact simulation of exit times, denoted by BoxExit. Section 2 concerns the introduction of the random walk on small rectangles of area $2T\times[a,b]/N$. A multi-armed bandit method is introduced in Section 3 for the optimal choice of the parameter N. Finally, in the last section we illustrate the efficiency of this new algorithm considering classical diffusion processes like the Ornstein-Uhlenbeck process or the Cox-Ingersoll-Ross model.

1. Exit problem from a rectangle

Let us first recall the algorithm introduced in (Herrmann and Zucca, 2020) (see Theorem 4.3) which permits to exactly simulate the first exit time from the rectangle $[0,T]\times[l,u]$ for the diffusion path $(X_t,\, t\geq 0)$. The algorithm essentially needs two basic elements:

1. the exact simulation of the exit time and location $(\mathcal{T}, B^x_{\mathcal{T}})$ from the interval $[l,u]$ for the Brownian motion starting in x, denoted by $(B^x_t,\, t\geq 0)$. Here $\mathcal{T}$ corresponds to a simplified notation for the first exit time $\tau_{l,u}(B^x)$ defined in (2). The generation of such a random vector is available (see Section 3 in (Herrmann and Zucca, 2020)) and will be denoted by $\textsc{ExitBm}(x,[l,u])$ in the sequel.

2. the generation of the Brownian position B^x_t given the specific event $\mathcal{T}>t$ which is denoted by $\textsc{CondBm}(x,[l,u],t)$ (see Section 2 in (Herrmann and Zucca, 2020)).

Both elements allow the construction of a general algorithm for the simulation of exit times. Before introducing the general procedure, we shall focus our attention onto a particular diffusion process which corresponds to the unique solution of a stochastic differential equation with unit diffusion coefficient:

$$dX_t=\mu_0(X_t)dt+dB_t,\quad X_0=x\in(a,b). \tag{3}$$

Here the drift term is assumed to satisfy $\mu_0\in\mathcal{C}^2([a;b])$. We define particular functions associated to equation (3) as:

$$\beta(x):=\exp\int_0^x\mu_0(y)\,dy\quad\text{and}\quad\gamma(x):=\frac{\mu_0^2(x)+\mu_0'(x)}{2}.$$

These functions play an important role in the simulation and do not depend on the considered interval $[l,u]$. Let us now complete these functions with different parameters depending on the interval $[l,u]$:

$$\beta^+:=\sup_{x\in[l,u]}\beta(x),\quad\gamma^-:=\inf_{x\in[l,u]}\gamma(x)\wedge 0,\quad\gamma^+:=\sup_{x\in[l,u]}\gamma(x),\quad\gamma^0:=\gamma^+-\gamma^-.$$

A unified statement of the exact simulation algorithms presented in (Herrmann and Zucca, 2020) is defined as follows:

Proposition 1 *The couple $(\tau_{l,u}(X)\wedge T, X_{\tau_{l,u}(X)\wedge T})$ which corresponds to the exit problem of the diffusion path (3) from the rectangle $[0,T]\times[l,u]$, has the same distribution than the outcome $(\mathcal{T},Z)$ of the algorithm $\textsc{BoxExit}(x,[l,u],T)$ for any $T>0$ (see the flowchart in Figure 2).*

It is worth noting that the random variables generated in the algorithm BoxExit (i.e. E, U, V, W) are independent (In Figure 2, U, V and W are represented by the same character $U_\bullet$ which corresponds to independent uniformly distributed variates).

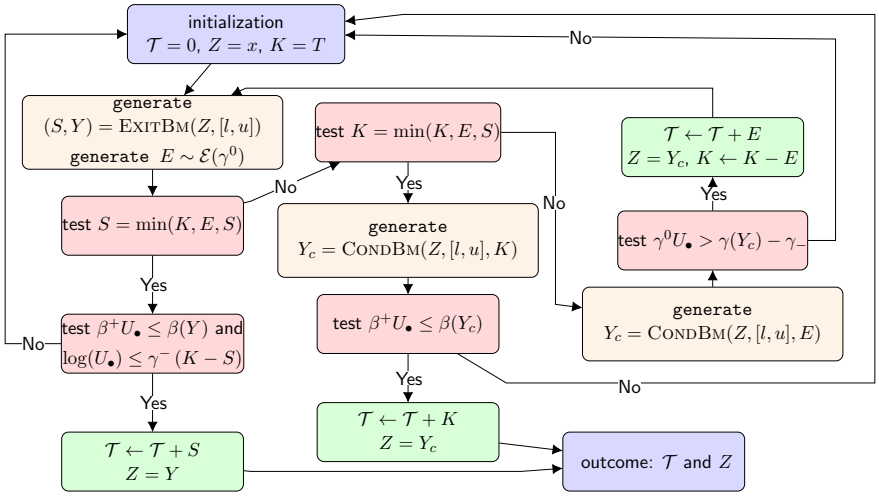


Figure 2: Flowchart of the algorithm $\text{BoxExit}(x, [l, u], T)$

Data: x (starting position), T, l and u (box size), $\gamma(\cdot)$ and $\beta(\cdot)$ (input functions).
Result: the random time $\mathcal{T}$ and the random location Z.

Initialization: $K = T$, $Z = x$, $\mathcal{T} = 0$, test $= 0$;
Computation of γ^-, γ^0, β^+ depending on the interval $[l, u]$;

while test $= 0$ **do**
  generate $E \sim \mathcal{E}(\gamma^0)$ and $U \sim V \sim W \sim \mathcal{U}([0, 1])$;
  generate $(S, Y) = \text{ExitBm}(Z, [l, u])$;
  if $S = \min(K, E, S)$ **then**
    if $\beta^+ U \leq \beta(Y)$ and $\log(W) \leq \gamma^-(K - S)$ **then**
      set test $= 1$, $Z \leftarrow Y$ and $\mathcal{T} \leftarrow \mathcal{T} + S$;
    else
      go to *initialization*;
    end
  else if $K = \min(K, E, S)$ **then**
    generate $Y_c = \text{CondBm}(Z, [l, u], K)$;
    if $\beta^+ U \leq \beta(Y_c)$ **then**
      set test $= 1$, $Z \leftarrow Y$ and $\mathcal{T} \leftarrow \mathcal{T} + K$;
    else
      go to *initialization*;
    end
  else
    generate $Y_c = \text{CondBm}(Z, [l, u], E)$;
    if $\gamma^0 V > \gamma(Y_c) - \gamma_-$ **then**
      $Z \leftarrow Y_c$, $\mathcal{T} \leftarrow \mathcal{T} + E$ and $K \leftarrow K - E$;
    else
      go to *initialization*;
    end
  end
end

Algorithm 1: $\text{BoxExit}(x, [l, u], T)$

Remark 2 *Under the assumption* $\gamma^- = 0$ *that is* $\inf_{x\in[l,u]} \gamma(x) \geq 0$*, it is allowed to choose* $T = \infty$ *in the algorithm* BoxExit$(x, [l, u], T)$*. It sould be noted that* BoxExit *with* $T < \infty$ *corresponds to the so-called* κ-DET *algorithm in (Herrmann and Zucca, 2020) whereas* BoxExit *with* $T = \infty$ *corresponds to the* DET *algorithm. Here we decided to unify the presentation for pedagogical reasons.*

Remark 3 *The Lamperti transform permits to generalize the study to equations with non-unitary diffusion coefficients as* (1)*. We simply present this well-known transformation. Let* $(X_t,\, t \geq 0)$ *be the unique solution to the SDE* (1) *and let us introduce*

$$\mathcal{S}(x) = \int_0^x \frac{du}{\sigma(u)}, \quad \forall x \in \mathbb{R}, \tag{4}$$

then Itô's lemma implies that $\widehat{X}_t := \mathcal{S}(X_t)$ *satisfies* (3) *with initial condition* $\widehat{X}_0 = \mathcal{S}(X_0)$ *and drift term*

$$\mu_0(x) := \frac{\mu(\mathcal{S}^{-1}(x))}{\sigma(\mathcal{S}^{-1}(x))} - \frac{1}{2}\sigma'(\mathcal{S}^{-1}(x)), \quad x \in \mathbb{R}.$$

The procedure to simulate the first exit time and location of a diffusion path $(X_t,\, t \geq 0)$ *defined by* (1) *from the rectangle* $[0, T] \times [l, u]$ *is therefore the following:*

1. *Simulate* $(\mathcal{T}, Z)$ *the first exit time and location of the diffusion* $(\widehat{X}_t,\, t \geq 0)$ *using the algorithm* BoxExit$(\mathcal{S}(x), [\mathcal{S}(l), \mathcal{S}(u)], T)$
2. *Compute* $\mathcal{S}^{-1}(Z)$*. Then* $(\mathcal{T}, \mathcal{S}^{-1}(Z))$ *corresponds to the first exit time and location of the diffusion* $(X_t,\, t \geq 0)$ *from the interval* $[l, u]$*.*

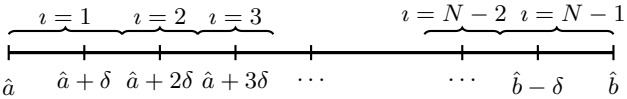
2. A random walk on rectangles

Using the exit problem of rectangles as the basic component, we can build a general algorithm that enables us to simulate exactly the first exit time of the diffusion process (1) from the interval $[a, b]$. Applying the Lamperti transformation already described in Remark 3, there is a one-to-one correspondence between the process $(X_t,\, t \geq 0)$ solution of (1) starting in $X_0 = x$ and $(\widehat{X}_t,\, t \geq 0)$ the solution of (3) starting in $\widehat{X}_0 = \mathcal{S}(x) = \hat{x}$ where $\mathcal{S}$ is defined by (4). Moreover, the interval $[a, b]$ is transformed into $[\hat{a}, \hat{b}] = [\mathcal{S}(a), \mathcal{S}(b)]$.

Let us now describe how to deal with the exit problem for $(\widehat{X}_t,\, t \geq 0)$ associated to the interval $[\hat{a}, \hat{b}]$. Let us first fix a parameter $T > 0$ and a number $N \geq 2$ (we shall comment on these choices later on). These parameters define the size of the typical boxes used in the algorithm illustrated by Figure 1: rectangles of area $2(\hat{b} - \hat{a})T/N$. The main idea is quite simple: the interval $[\hat{a}, \hat{b}]$ is split into N intervals of identical length δ, associated to the following space grid: $a_0 = \hat{a}$ and $a_{j+1} = a_j + \delta$ for $0 \leq j \leq N - 1$. Here $\delta = (\hat{b} - \hat{a})/N$. We define the index function:

$$\imath(x) = j \quad \text{if} \quad (x - \hat{a}) \in \left[\frac{\delta}{2} + (j-1)\delta, \frac{\delta}{2} + j\,\delta\right[, \tag{5}$$

otherwise either $\imath(x) = 1$ for $x \leq \hat{a} + \delta/2$ or $\imath(x) = N - 1$ for $x \geq \hat{b} - \delta/2$.



Each index value $\imath \in \{1, 2, \ldots, N-1\}$ is associate to an interval of length 2δ:

$$I_\imath =]\hat{a} + (\imath - 1)\delta, \hat{a} + (\imath + 1)\delta[. \tag{6}$$

We notice that the family of intervals $(I_\imath)_{1 \leq \imath \leq N-1}$ is a covering of the initial interval $]\hat{a}, \hat{b}[$. Moreover, for any $x \in]\hat{a}, \hat{b}[$, $x \in I_{\imath(x)}$.

A random walk corresponding to a skeleton of the diffusion path can be thus constructed (see Figure 1): $(T_0, Y_0) = (0, \hat{x})$ is the starting time and position of the diffusion process $(\widehat{X}_t,\, t \geq 0)$, solution of (3). The random sequence (T_{n+1}, Y_{n+1}) is defined recursively as follows: $T_{n+1} - T_n$ stands for the first exit time of the diffusion starting in Y_n from the rectangle $[0, T] \times I_{\imath(Y_n)}$ and Y_{n+1} corresponds to the associated exit location. Let us define

$$\mathcal{N} := \inf\{n \geq 0 :\ Y_n \notin]\hat{a}, \hat{b}[\}$$

then the combination of the Markov property and the Lamperti transform implies the following statement.

Proposition 4 *The diffusion first exit time and location* $(\tau_{a,b}(X), X_{\tau_{a,b}(X)})$ *has the same distribution as the stopped random walk* $(T_{\mathcal{N}}, \mathcal{S}^{-1}(Y_{\mathcal{N}}))$ *and consequently the same distribution as* $(\mathcal{T}, Z)$ *the outcome of the algorithm* DiffExit *given below in Algorithm 2.*

The algorithm DiffExit induced by this random walk is the following.

Data: x (starting position of the diffusion), T, N (box size), $\gamma(\cdot)$ and $\beta(\cdot)$ (input functions), $\mathcal{S}(\cdot)$ (Lamperti transform).
Result: the random time $\mathcal{T}$ and the random location Z.

initialization: $\mathcal{T} = 0$, $Z = \mathcal{S}(x)$, $\hat{a} = \mathcal{S}(a)$, $\hat{b} = \mathcal{S}(b)$;
while $Z \in]\hat{a}, \hat{b}[$ **do**
  $(S, Z) \leftarrow \text{BoxExit}(Z, I_{\imath(Z)}, T)$;
  $\mathcal{T} \leftarrow \mathcal{T} + S$;
end
$Z \leftarrow \mathcal{S}^{-1}(Z)$;

Algorithm 2: Diffusion Exit Problem DiffExit(T, N)

Of course, the efficiency of this exact simulation algorithm heavily depends on the parameters T and N which characterize the size of the typical boxes. If the box is large, then the algorithm BoxExit becomes time consuming since it is based on a rejection sampling. On the contrary, small boxes imply that the random walk on rectangles requires a lot of iterations in order to hit the boundaries of the interval $[a, b]$. There is therefore an intermediate box size which permits to observe simulations that take a reasonable computation time.

In order to illustrate this feature, let us introduce two particular examples:
Example 1: The diffusion process with unitary diffusion coefficient and with the following drift term: $\mu_0(x) = 2 + \sin(x)$. We consider the exit problem from the interval $[a, b] = [0, 7]$, the diffusion starting in $x = 3$. Figure 3 represents on the one hand the average number of boxes needed in order to observe the exit depending on the box size (we let N vary). On the other hand we also point out the computation time (in sec) needed to generate a sample of 10 000 diffusion exit times.

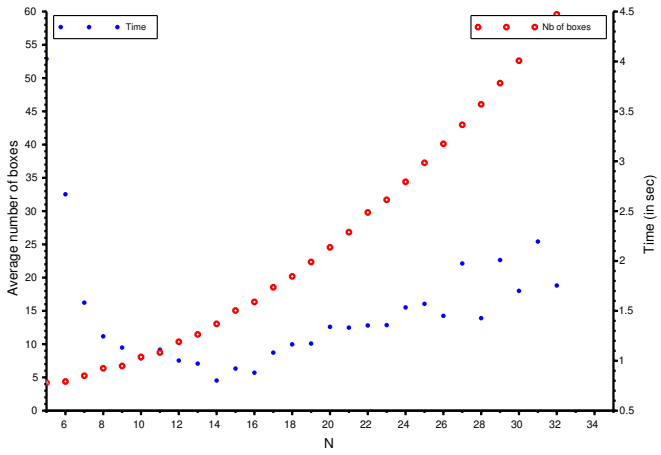


Figure 3: Average number of boxes used in the exit algorithm and total computation time (for the simulation of the whole sample) versus the box size parameter N for the diffusion process of Example 1. Exit problem from the interval $[a, b] = [0, 7]$ with the starting position $x = 3$. Each value is obtained with a sample of size 10 000 and $T = 1$.

Example 2: The Ornstein-Uhlenbeck process with unitary diffusion coefficient and drift term : $\mu_0(x) = -\lambda x$ with $\lambda > 0$. First we focus our attention to the exit problem from the interval $[a, b] = [0, 7]$ with the initial condition $x = 3$ and the parameter $\lambda = 1$, see Figure 4 (left).

We notice that the optimal box size corresponds to $N = 14$ when $T = 1$ is fixed. Such an optimal choice strongly depends on the interval $[a, b]$. Since the diffusion is mean-reverting, let us observe what happens when the interval $[a, b]$ contains 0. Figure 4 (right) illustrates that $N = 5$ is optimal for $[a, b] = [-2, 2]$ and $x = 0.5$. We also notice that the number of boxes used in such a particular situation is much larger than in the previous situation. It is therefore difficult to obtain a theoretical optimal value for the parameter N. That is why we aim to find an acceleration method for the simulation of exit times (Algorithm 2: DiffExit) using an algorithmic approach based on a multi-armed bandit.

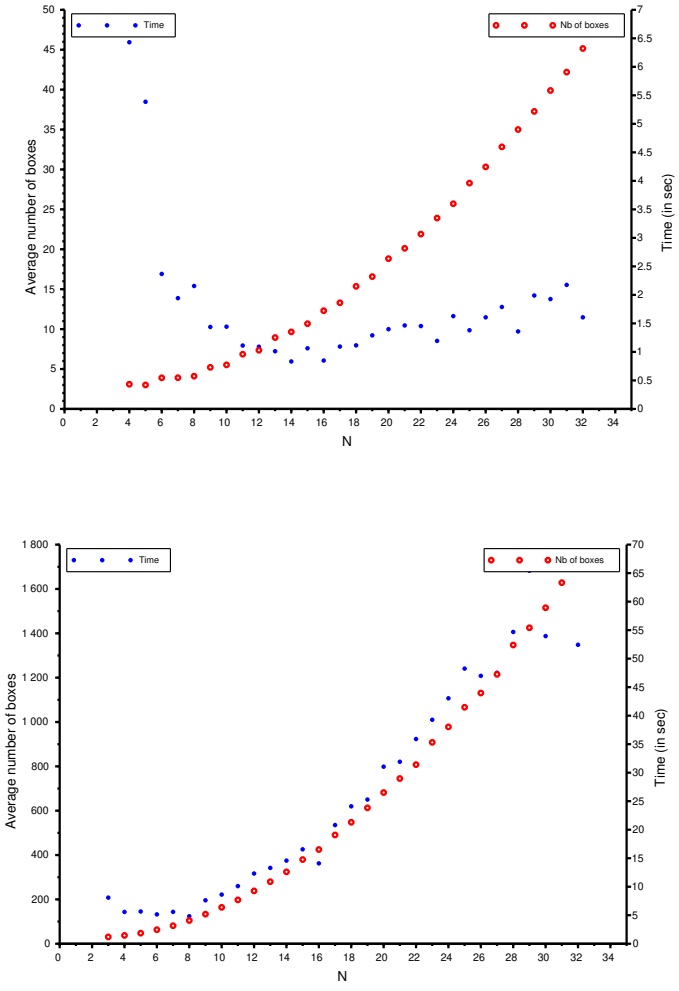


Figure 4: Average number of boxes and total computation time versus the box size parameter N for the Ornstein-Uhlenbeck process with parameter $\lambda = 1$. Exit problem from the interval $[a, b] = [0, 7]$ with the starting position $x = 3$ (l) and $[a, b] = [-2, 2]$ and $x = 0.5$ (right). Each value is obtained with a sample of size 10 000 and $T = 1$.

3. Algorithm acceleration: a multi-armed bandit approach

Let us now suggest an acceleration method for the algorithm DiffExit presented in the previous section and depending on both parameters N and T (size of the typical boxes). The procedure is the following: we first fix $T > 0$ and $N_0 \geq 2$. Then we introduce an algorithm used for the multi-armed bandit problem in order to choose an interesting value of N satisfying $N \leq N_0$ and reducing the time consumption of the algorithm DiffExit.

The multi-armed bandit corresponds to a famous problem where reinforcement learning plays a crucial role, theoretical and practical studies aim to find trade-offs between exploration and exploitation. The historical problem is quite simple and related to a slot machine with a finite number of levers. One is faced repeatedly to a choice between these actions and after each choice one receive a random numerical reward depending on the selected lever. The objective is to maximize the average cumulative reward of a series of actions (for instance, 10 000 successive selections) using a strategy based on an exploration-exploitation algorithm. The exploration consists in selecting several times any arm of the bandit in order to estimate the different mean rewards while the exploitation focuses on the choice of the arm whose estimated reward is maximal. We refer to the interesting textbooks (Slivkins, 2019) and (Sutton and Barto, 2018) for practical and theoretical results associated with this reinforcement learning framework. Several bandit algorithms permit to obtain theoretical bounds of the total expected regret which represents a simple performance measure in such a framework: ϵ-greedy, Boltzmann exploration, UCB (Upper Confidence Bounds), etc. Here we focus our attention on the *ϵ-greedy algorithm* which is rather intuitive, simple to implement and outperforms theoretically sound algorithms on most settings (Vermorel and Mohri, 2005). Note that we chose a frequentist approach and not a Bayesian one because we have no a priori information on the conditional probability distribution of the rewards. Moreover, we look for an algorithm that can easily be used in a general framework and does not depend on the characteristics of the diffusion process under review (Lattimore and Szepesvári, 2020)

In our particular situation, the multi-armed bandit corresponds to the already introduced algorithm DiffExit(T, N): each arm represents a value of $N \in \{2, 3, \dots, N_0\}$ which characterizes the space splitting used in the algorithm. The reward associated with each arm is the numerical time consumption of each exit time generation. It is of course random since the basic components of the algorithm use rejection sampling. Let us mention that the objective is here opposite: the aim is to minimize the cumulative reward. That means that each use of the algorithm DiffExit leads to an evaluation of the time spent. We shall therefore use a clock for determining the current time denoted by CurrentTime.

Let us present the application of ϵ-greedy algorithm in such a context. After the n-th use of the algorithm DiffExit(T, N), the empirical mean of the time consumption is denoted by $\bar{\mu}_n(N)$, for $2 \leq N \leq N_0$, and the number of times we already used the arm N until n is $m_n(N)$. In the ϵ-greedy algorithm, the choice of the parameter N evolves randomly as the number of simulations increases and depends on a fixed parameter ϵ. The probability to choose the arm N for the n-th simulation is defined by:

$$\pi_{n+1}(N) = (1-\epsilon)\mathbb{1}_{\{N = \underset{2 \leq j \leq N_0}{\arg\min}\, \bar{\mu}_n(j)\}} + \frac{\epsilon}{N_0 - 1}, \tag{7}$$

with the starting values $\pi_1(N) = \epsilon/(N_0 - 1)$ for all $N \in \{2, 3, \dots, N_0\}$. Such strategy for the random choice of the parameters permits to globally reduce the consumption time for a sequential use of the algorithm DiffExit. Of course the parameter ϵ characterizing the competition between exploration and exploitation has an influence on the acceleration strength and should depend on the sample size. Different studies even suggest to let ϵ depend on the number of actions $\epsilon := \epsilon(n)$ of the order $\epsilon(n) = n^{-1/3}((N_0 - 1)\log(n))^{1/3}$ (see, for

instance, Theorem 1.4 in (Slivkins, 2019)). Nevertheless experimental results emphasize that making the ϵ decrease does not significantly improve the performance of the multi-armed bandit strategy (Vermorel and Mohri, 2005). In the following we shall therefore only use the ϵ-greedy algorithm with fixed value $\epsilon \in (0, 1]$.

The modification of the DiffExit leads to the following algorithm.

Data: x (starting position), T, N_0, $\gamma(\cdot)$ and $\beta(\cdot)$ (input functions), M (size of the sample: number of simulations), ϵ (parameter in the ϵ-greedy algorithm).
Result: Sample of M simulations for the couple random time $\mathcal{T}$ and random location Z.
initialization:
$\pi(N) \leftarrow 1/(N_0 - 1)$, $\bar{\mu}(N) \leftarrow 0$ and $m(N) = 0$ for all $2 \leq N \leq N_0$;
for $j \leftarrow 1$ **to** M **do**
  choose randomly N w.r.t. the distribution $\pi(\cdot)$;
  $t \leftarrow$ CurrentTime;
  $(\mathcal{T}_j, Z_j) \leftarrow$ DiffExit(T, N);
  $t \leftarrow$ CurrentTime $- t$;
  $\bar{\mu}(N) \leftarrow (m(N)\bar{\mu}(N) + t)/(m(N) + 1)$;
  $m(N) \leftarrow m(N) + 1$;
  for $i \leftarrow 2$ **to** N_0 **do**
    $\pi(i) \leftarrow \epsilon/(N_0 - 1)$;
  end
  $\pi(\arg\min \bar{\mu}) \leftarrow \pi(\arg\min \bar{\mu}) + (1 - \epsilon)$;
end

Algorithm 3: BanditDiffExit(T, N_0)

This new algorithm called BanditDiffExit outperforms the exact algorithms introduced for the simulation of diffusion exit times in (Herrmann and Zucca, 2020) as it appears obvious in the numerical illustrations presented in Section 4.

4. Numerical illustration

4.1 First example

First we consider the first exit time from the interval $[a, b]$ for the diffusion:

$$dX_t = (2 + \sin(X_t))\, dt + dB_t, \quad t \geq 0, \quad X_0 = x. \tag{8}$$

In (Herrmann and Zucca, 2020), the DET-algorithm permits to generate the first exit time due to an acceptance rejection procedure (this algorithm corresponds to the already presented BoxExit$(x, [a, b], T)$ for the particular value $T = \infty$, we can observe that the condition described in Remark 2 is satisfied). Using a sample of first exit time generations we can estimate the average computation time. Here the data correspond to the first exit time from the interval $[0, 7]$ when starting in $x = 3$.

sample size	average time (ms)	confidence interval (95%)	
10 000	7.832	7.676	7.989

It is of prime interest to compare the computation time using BoxExit-algorithm with the computation time using the bandit algorithm presented in Section 3. Here we deal with a sample of 1 000 actions in the bandit algorithm, each run corresponds to the simulation of an exit time from the interval $[a, b] = [0, 7]$. Let us note that inbetween two consecutive runs, the bandit algorithm proceed to an optimisation computation corresponding to the choice of the box size. Therefore the sequence of the consumption times $\tau^{(1)}, \ldots, \tau^{(n)}$ do not represent i.i.d random variables (the confidence interval is therefore not available). We provide the performance of such an algorithm in Figure 5: the averaged computation time is strongly reduced. The figure represents the sequence : $(\frac{1}{n}\sum_{i=1}^{n}\tau^{(i)})_n$ for $10 \leq n \leq 1\,000$.

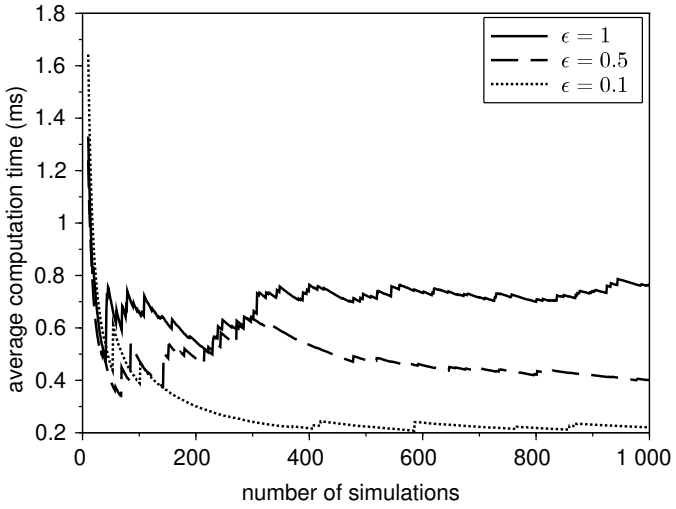


Figure 5: Empirical mean of the computation times (in ms) versus the number of simulations (10 to 1 000). Each computation time corresponds to the simulation of an exit from the interval $[a, b] = [0, 7]$ with starting value $x = 3$. We use the ϵ-greedy bandit algorithm with different values ϵ ($\epsilon = 1$ corresponds to a uniform choice of the parameter N in $\{2, \ldots, 21\}$). The elementary box size is $2(b-a)T/N$ with $T = 1$.

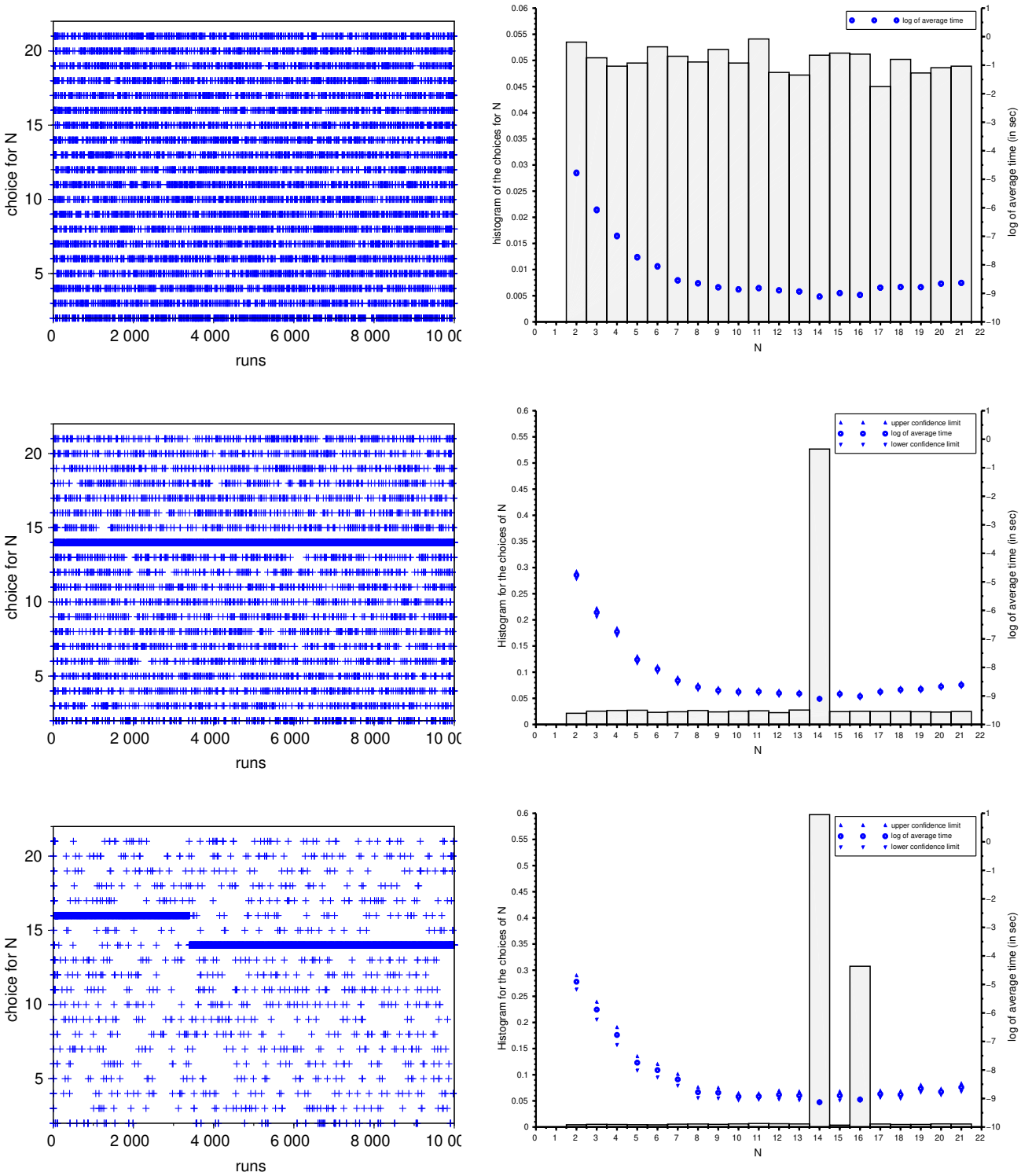


Figure 6: Choice of the box size (parameter N) versus the number of iterations (left) when the box size is chosen uniformly inbetween 2 and 21 accordingly to the ϵ-greedy algorithm with $\epsilon = 1$ (top), $\epsilon = 0.5$ (middle) or $\epsilon = 0.1$ (bottom) and histogram of the box size N for a sample of size 10 000. Here we consider the exit time of the interval $[a, b] = [0, 7]$ and starting point $x = 3$ and $T = 1$.

The multi-armed bandit approach permits to possibly change the box size used for the first exit time simulation by selecting the parameter N inbetween a set of given values (here $\{2, \dots, 21\}$, the arms of the bandit). The sequence of successive choices is randomized since the parameter ϵ which represents in some sense the level of noise (the proportion of exploration in the whole sequence of successive runs), belongs to $]0, 1]$. In other words, the particular choice $\epsilon = 1$ corresponds to a sequence of independent uniformly distributed choices whereas ϵ close to 0 corresponds to a sequence of mainly deterministic choices linked to the argmin of the previous rewards (here the rewards are the consumption times).

In Figure 6, we illustrate the behavior of the algorithm for three different values of ϵ. In each case, the selections of the parameter N throughout the sequence of iterations are represented by crosses in the figures (left). Once all first exit times have been simulated, the assessment is represented by both the frequencies of each value of N (histogram - right) and the corresponding average consumption time (with possibly its confidence interval). We can immediately observe the following.

- In the case $\epsilon = 1$, the choice of the parameter N at each step of the algorithm does not depend on the previously observed consumption times, $N = 14$ corresponding to its argmin is not privileged. In a sample of size 10 000, only 5.1% of the actions consist of the use of the optimal parameter $N = 14$. However, an acceleration is observed: the consumption time of the numerical generation using this bandit method equals 9.7% of the classical consumption time.

- In the second case studied ($\epsilon = 0.5$, middle), the particular choice $N = 14$ is rapidly privileged even if the relatively important level of noise implies a frequent visit of each proposed choice: $2, 3, \dots, 21$. The exploration is quite important in that case. As in the first case, for a generation of 10 000 exit times, 52% of the actions use the optimal parameter and the global consumption times equals 5.2% of the classical consumption time.

- Finally in the third case ($\epsilon = 0.1$), the experiment leads to the following observation: the bandit algorithm makes $N = 16$ its first choice but after a while (about 4 000 iterations) the noise permits to leave this local minimum and to choose the global one. For a sample of 10 000 exit times, 59.7% of the generations deal with the optimal choice $N = 14$ and the global time consumption equals 2.1% of the consumption time without acceleration.

So in order to reach a global minimum, it seems to be important not to choose the noise level ϵ too small. However we notice that the consumption times observed for both $N = 14$ and $N = 16$ are very close together, so the investigation of the argmin is not a crucial challenge. The acceleration algorithm using the multiarmed bandit approach is accurate: whatever the choice of the parameter ϵ is, the global time consumption is strongly reduced with respect to the initial method of generation. If the optimal choice of the parameter N is known in advance, it is possible to reach a strong acceleration just by using this optimal value: 0.15% of the classical time consumption. If the objective is to handle with huge samples and to describe the asymptotic behaviour of the expected regret, then more sophisticated bandit algorithms like UCB are preferred (Lattimore and Szepesvári, 2020).

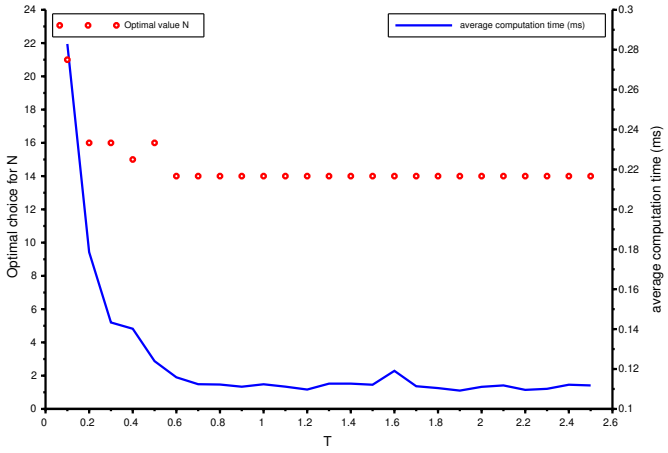


Figure 7: Optimal choice of the parameter N and average time consumption versus T. We recall that the box size is $T \times (b-a)/N$. Here N is chosen in the set $\{2, \ldots, 21\}$ accordingly to the ϵ-greedy algorithm with $\epsilon = 0.1$, and the average is computed using a sample of size 10 000. We consider the first exit time from the interval $[a, b] = [0, 7]$ and with starting value $x = 3$.

Of course the box size of the basic components is the essential lever for the efficiency of the first exit time simulation but it does not only depend on N. The area of the box is $2T \times (b-a)/N$ so that both T and N have to be correctly chosen. In Figure 6, the parameter T is fixed ($T = 1$) whereas N varies. Once the optimal choice of N is emphasized, it is possible to observe how it depends on T. Figure 7 illustrates that the consumption time of the algorithm does actually not precisely depend on T, provided that T is not too small ($T = 1$ is a reasonable choice).

4.2 Ornstein-Uhlenbeck processes

Let us now consider a diffusion process which does not verify the particular condition presented in Remark 2. We aim to illustrate the efficiency of the bandit algorithm with the Ornstein-Uhlenbeck processes. So we consider the stochastic process with unitary diffusion coefficient and drift term $\mu(x) = -\lambda x$. The aim is to simulate in some efficient way the first exit time of the interval $[a, b]$. Since the process is mean reverting, its behavior will depend on the location of 0, either in the interval $]a, b[$ either on the boundary or outside that interval. In order to present a complete illustration, we focus our attention on two different examples:

- Ex.1: interval $[a, b] = [0, 7]$, drift $\lambda = 1$ and starting position $x = 3$.
- Ex.2: interval $[a, b] = [-2, 2]$, drift $\lambda = 2$ and starting position $x = 0.5$.

In oder to simulate the first exit time from $[a, b]$ we aim to compare DiffExit with the multi-armed bandit approach BanditDiffExit. Let us just recall that DiffExit$(x, [a, b], T)$ is based on a sequential observation of the paths on the intervals $[nT, (n + 1)T]$, $n \geq 0$, till the exit happens. Here T is a parameter which influences the efficiency of the numerical procedure. For both cases under consideration, we observe that $T = 0.5$ is a reasonable choice as suggested by the following table. It presents the estimated computation times in *ms* for one exit time generation (estimation with a sample of 10 000 exit times).

	computation time in *ms* (Ex.1)			computation time in *ms* (Ex.2)		
T	average	confidence (95%)		average	confidence (95%)	
0.1	4.720	4.602	4.837	68.636	67.304	69.969
0.2	4.292	4.194	4.389	52.355	51.346	53.363
0.5	4.195	4.123	4.267	46.471	45.542	47.401
1	5.022	4.929	5.115	55.076	53.973	56.178
2	8.001	7.852	8.151	98.056	96.121	99.990
3	13.135	12.876	13.394	186.576	182.906	190.246

The consumption times associated with $T = 0.5$ become therefore our reference values which need to be compared to the times issued from the multi-armed bandit. Figure 8 illustrates the efficiency of our approach for both examples (Ex.1 and Ex.2) since these consumption times have been reduced especially for small noise intensity ϵ (we suggest to choose ϵ smaller than 0.5). This acceleration is less impressive when the origin 0 belongs to the interval $[a, b]$ (Figure 8, right). Let us also note that the curves of the average computation time associated with the parameters $\epsilon = 0.1$ and $\epsilon = 0.2$ intersect each other: if one needs a huge number of simulations, then one prefer $\epsilon = 0.1$ which permits to find the global minimum and to avoid to often visit the other values of N. If one needs rather an intermediate value of simulations (for instance, 1 000 exit times), then it is better to increase a little the noise in the multi-armed bandit ($\epsilon = 0.2$) in order to find quickly the optimal value N even if the algorithm frequently visits all the other values of N.

The parameter T was fixed so far in the study of the Ornstein-Uhlenbeck process and the attention was focused on the best choice of N. As already explained in Section 4.1, the BoxExit algorithm depend both on N and T. Figure 9 represents the dependence of the optimal choice of N and the average time consumption with respect to the parameter T. This illustration emphasizes that the efficiency is not strongly dependent with respect to T provided that T is neither too small nor too large. Even if the box size depends on both parameters T and N, it is therefore more clever to look after the best choice for N rather than the best choice for T. Moreover we prefer to avoid an application of the ϵ-greedy algorithm to the couple (T, N) (T would be discretized) trying to keep things simple.

4.3 Cox-Ingersoll-Ross Processes

In all the previous examples, the diffusions under observation have a unitary diffusion coefficient. In such situations, both DiffExit and BanditDiffExit can be applied directly without using Lamperti's transform (see Remark 3). In order to complete the numerical

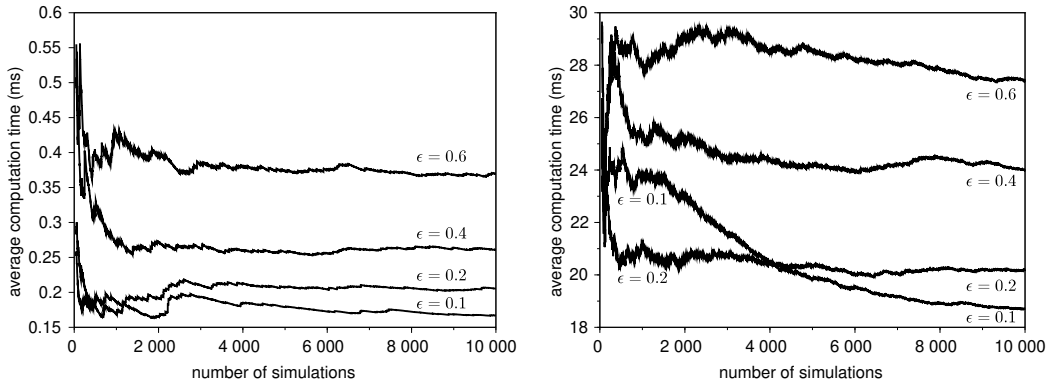


Figure 8: Empirical mean of the computation time (in ms) versus the number of simulations of exit times from the interval $[a, b]$ for both examples (Ex.1 left and Ex.2 right). We use different ϵ-greedy algorithms with $T = 0.5$ and N is chosen in the set $\{2, \ldots, 21\}$.

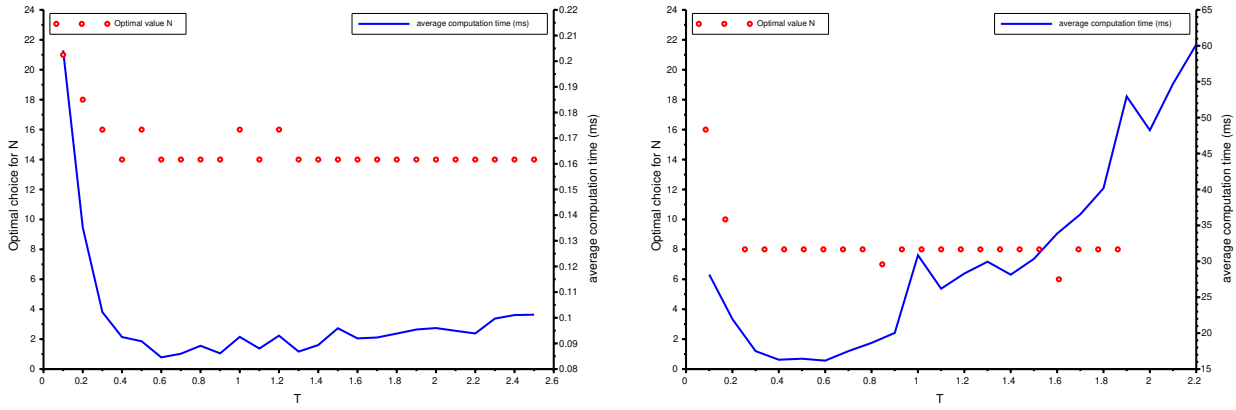


Figure 9: Optimal choice of the parameter N and average time consumption versus T for Ex.1 (left) and Ex.2 (right) . Here N is chosen in the set $\{2, \ldots, 21\}$ accordingly to the ϵ-greedy algorithm with $\epsilon = 0.1$, and the average is computed using a sample of size 10 000.

illustration, we introduce a third example linked to the so-called CIR model (Cox-Ingersoll-Ross) which is of prime importance in the mathematical finance framework, in particular for the modelization of interest rates. The CIR model is characterized by the following stochastic differential equation:

$$dX_t = k(\theta - X_t)\, dt + \sigma \sqrt{X_t}\, dB_t, \quad t \geq 0, \quad X_0 = x > 0. \tag{9}$$

Here k and θ are two parameters. Since the diffusion coefficient is not constant, we have to use the Lamperti transformation introduced in (4). So we define $\mathcal{S}(x) = \frac{2}{\sigma}\sqrt{x}$. Then $\widehat{X}_t := \mathcal{S}(X_t)$ is a diffusion process with unitary diffusion coefficient and drift term:

$$\mu_0(x) = \frac{\rho}{x} - \frac{kx}{2} \quad \text{where} \quad \rho := \frac{(4k\theta - \sigma^2)}{2\sigma^2}.$$

Let us assume that the parameters appearing in (9) satisfy the condition: $\rho > 0$. Consequently the CIR process starting from a positive initial point stays strictly positive (see, for instance, (Jeanblanc et al., 2009) in Section 6.3.1) and the function γ and β used in the algorithms have an explicit expression easy to handle with:

$$\gamma(x) = \frac{1}{2}\left(\left(\frac{\rho}{x} - \frac{kx}{2}\right)^2 - \frac{\rho}{x^2} - \frac{k}{2}\right), \quad \beta(x) = x^\rho e^{-kx^2/4}. \tag{10}$$

For numerical illustration, we deal with the exit problem from $[a, b] = [1, 6]$ for the CIR model starting in $x = 3$ with coefficients $k = 3$, $\theta = 7$ and $\sigma = 1$. As in the Ornstein-Uhlenbeck context, we can here use the BoxExit(x,[$\mathcal{S}(a), \mathcal{S}(b)$],T) algorithm in order to simulate both the first exit time and the exit location. This algorithm depends on a parameter T. We obtain the following average computation times for one first exit time generation:

T	average in ms	confidence (95%)	
0.1	12.206	11.981	12.431
0.2	11.939	11.718	12.161
0.5	11.901	11.671	12.130
1	12.272	12.036	12.509

We can observe that the parameter T has only a weak influence on the BoxExit efficiency provided T belongs to an interval of reasonable values (here inbetween 0.1 and 1). Let us now compare these computation times of the order of 12 ms per simulation to the BanditDiffExit algorithm one. In Figure 10 (left), we observe a significant time reduction as soon as ϵ (the parameter of the ϵ-greedy procedure) is sufficiently small, we reach a computation time near to 0.2 ms per simulation (for a sample size 10 000). Since the box size used in BanditDiffExit depends on both parameters N and T, we wonder if the optimal value of N strongly depends on T. As we can see in Figure 10 (right), it is not the case: there is neither large swings in the optimal choice of the value of N nor in the average consumption time associated with this optimal N.

Conclusion

The exact simulation procedure BoxExit$(x, [a, b], T)$ proposed in (Herrmann and Zucca, 2020) permits to generate the first exit time and exit location from an interval $[a, b]$ in the diffusion context. In this study, we emphasize a reinforcement learning method based on a multi-armed bandit which permits to accelerate the BoxExit algorithm in any case. As presented in Section 4, sometimes the consumption time reduction is very strong and sometimes sensible. The outstanding advantage of the algorithm BanditDiffExit is its universality: it does not depend on the the particular family of diffusion under consideration.

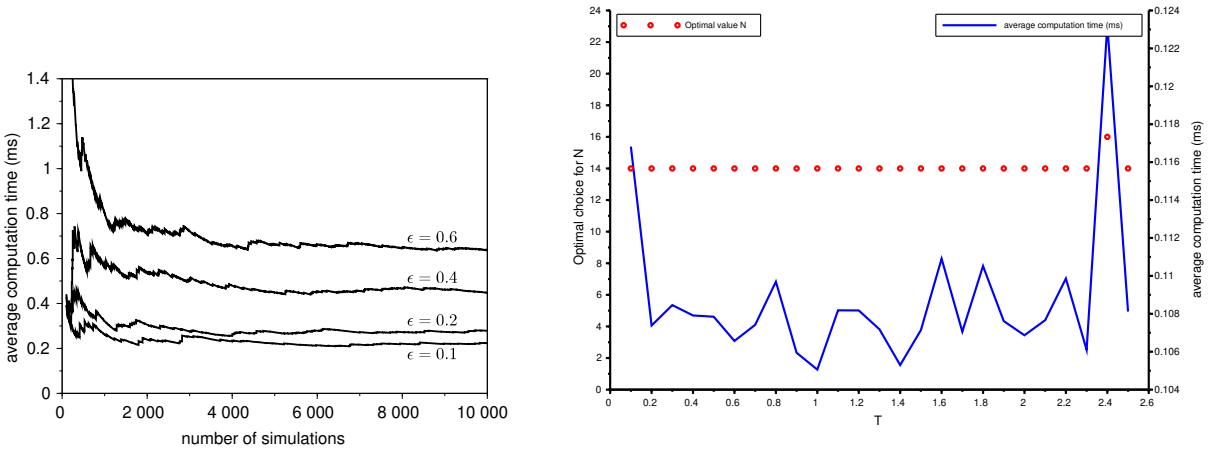


Figure 10: Empirical mean of the computation time (in *ms*) versus the number of simulations of exit times from the interval $[1, 6]$ with different ϵ-greedy algorithms and $T = 0.5$ (left). Optimal choice of the parameter N and average time consumption versus T for the CIR model (right). Here N is chosen in the set $\{2, \ldots, 21\}$ accordingly to the ϵ-greedy algorithm with $\epsilon = 0.1$, and the average is computed using a sample of size 10 000.

Let us also note that the authors have chosen the ϵ-greedy algorithm for the acceleration procedure since it is simple to explain and particularly efficient. Of course any other algorithm used in the classical multi-armed bandit problem can be tested for the acceleration of BoxExit.

All the numerical tests have been done on the same computer:
Intel Core i5, 1.6 GHz
The open-source code corresponding to the C++ programs used in Section 4 is available at https://github.com/SamHerr/Diff-FirstExitTime-Acceleration

References

A. Beskos and G.O. Roberts. Exact simulation of diffusions. *The Annals of Applied Probability*, 15(4):2422–2444, 2005.

A. Beskos, O. Papaspiliopoulos, and G.O. Roberts. Retrospective exact simulation of diffusion sample paths with applications. *Bernoulli*, 12(6):1077–1098, 2006. ISSN 1350-7265. doi: 10.3150/bj/1165269151. URL `http://dx.doi.org/10.3150/bj/1165269151`.

M. Broadie, P. Glasserman, and S. Kou. A continuity correction for discrete barrier options. *Math. Finance*, 7(4):325–349, 1997. ISSN 0960-1627. doi: 10.1111/1467-9965.00035. URL http://dx.doi.org/10.1111/1467-9965.00035.

E. Gobet. Weak approximation of killed diffusion using Euler schemes. *Stochastic Process. Appl.*, 87(2):167–197, 2000. ISSN 0304-4149. doi: 10.1016/S0304-4149(99)00109-X. URL http://dx.doi.org/10.1016/S0304-4149(99)00109-X.

E. Gobet and S. Menozzi. Stopped diffusion processes: boundary corrections and overshoot. *Stochastic Process. Appl.*, 120(2):130–162, 2010. ISSN 0304-4149. doi: 10.1016/j.spa.2009.09.014. URL http://dx.doi.org/10.1016/j.spa.2009.09.014.

S. Herrmann and C. Zucca. Exact simulation of the first-passage time of diffusions. *J. Sci. Comput.*, 79(3):1477–1504, 2019. ISSN 0885-7474. doi: 10.1007/s10915-018-00900-3. URL https://doi.org/10.1007/s10915-018-00900-3.

S. Herrmann and C. Zucca. Exact simulation of first exit times for one-dimensional diffusion processes. *ESAIM: M2AN*, 54(3):811–844, 2020. doi: 10.1051/m2an/2019077. URL https://doi.org/10.1051/m2an/2019077.

M. Jeanblanc, M. Yor, and M. Chesney. *Mathematical methods for financial markets*. Springer Finance. Springer-Verlag London, Ltd., London, 2009. ISBN 978-1-85233-376-8. doi: 10.1007/978-1-84628-737-4. URL https://doi.org/10.1007/978-1-84628-737-4.

T. Lattimore and C. Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

L. Sacerdote, O. Telve, and C. Zucca. Joint densities of first hitting times of a diffusion process through two time-dependent boundaries. *Adv. in Appl. Probab.*, 46(1):186–202, 2014. ISSN 0001-8678. doi: 10.1239/aap/1396360109. URL https://doi.org/10.1239/aap/1396360109.

A. Slivkins. Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning*, 12(1-2):1–286, 2019. ISSN 1935-8237. doi: 10.1561/2200000068. URL http://dx.doi.org/10.1561/2200000068.

R. S. Sutton and A. G. Barto. *Reinforcement learning: an introduction*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, second edition, 2018. ISBN 978-0-262-03924-6.

J. Vermorel and M. Mohri. Multi-armed bandit algorithms and empirical evaluation. In J. Gama, R. Camacho, P.B. Brazdil, A.M. Jorge, and L. Torgo, editors, *Machine Learning: ECML 2005*, pages 437–448, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31692-3.