Simulating Events of Unknown Probabilities via Reverse Time Martingales

Krzysztof Łatuszyński
University of Warwick

Ioannis Kosmidis
University of Warwick

Omiros Papaspiliopoulos
Universitat Pompeu Fabra

Gareth O. Roberts
University of Warwick

June 25, 2009

Abstract

Assume that one aims to simulate an event of unknown probability $s \in (0,1)$ which is uniquely determined, however only its approximations can be obtained using a finite computational effort. Such settings are often encountered in statistical simulations. We consider two specific examples. First, the exact simulation of non-linear diffusions ([3]). Second, the celebrated Bernoulli factory problem ([10], [16], [13], [12], [9], and also [1] and [8]) of generating an $f(p)-$coin given a sequence $X_1, X_2, ...$ of independent tosses of a $p-$coin (with known f and unknown p). We describe a general framework and provide algorithms where this kind of problems can be fitted and solved. The algorithms are straightforward to implement and thus allow for effective simulation of desired events of probability s. In the case of diffusions, we obtain the algorithm of [3] as a specific instance of the generic framework developed here. In the case of the Bernoulli factory, our work offers a statistical understanding of the Nacu-Peres algorithm for $f(p) = \min\{2p, 1-2\varepsilon\}$ (which is central to the general question, c.f. [13]) and allows for its immediate implementation that avoids algorithmic difficulties of the original version. In the general case we link our results to existence and construction of unbiased estimators. In particular we show how to construct unbiased estimators given sequences of under- and overestimating reverse time super- and submartingales.

1 Introduction

Assume that one aims to simulate an event of unknown probability $s \in (0,1)$ which is uniquely determined, however only its approximations can be obtained using a finite computational effort. Typical examples include consistent estimation of s or a series expansion for s, we discuss it in the sequel. Such settings are often encountered in statistical simulations (see e.g. [7], [6], [5], [3] [10], [13]). A celebrated example of this kind is the Bernoulli factory problem which motivated our work. It can be stated as follows. Let $p \in \mathcal{P} \subseteq [0,1]$ be unknown and let $f : \mathcal{P} \to [0,1]$. How to generate Y, a single coin toss of an $s = f(p)-$coin, given a sequence $X_1, X_2, ...$ of independent tosses of a $p-$coin? For the historical context of this question and a range of theoretical results see [16], [10], [13],

[12] and [9]. In particular [10] provide necessary and sufficient conditions for f, under which an algorithm generating an $f(p)-$coin exists. Nacu and Peres in [13] suggest a constructive algorithm for simulating $f(p) = \min\{2p, 1-2\varepsilon\}$ which is central to solving the problem for general f and allows for generating an $f(p)-$coin for a large class of functions (e.g. real analytic, see [13] and Section 3 for details). The algorithm is based on polynomial envelopes of f. To run the algorithm one has to construct sets of $\{0,1\}$ strings of appropriate cardinality based on coefficients of the polynomial envelopes. Unfortunately its naive implementation requires dealing with sets of exponential size (we encountered e.g. $2^{2^{26}}$) and thus is not very practical. Hence the authors provide a simple approximate algorithm for generating $\min\{2p,1\}-$coins. An ongoing research in Markov chain Monte Carlo and rejection sampling indicates that the Bernoulli factory problem is not only of theoretical interest (c.f. also [1], [8] and [2] Chapter 16). However using approximate algorithms in these applications perturbs simulations in a way difficult to quantify.

In the present article we develop a framework for simulating events of unknown probabilities (Section 2). Our approach is based on random sequences, say L_n and U_n under- and overestimating s that are monotone in expectations (i.e. $\mathbb{E}\, L_n \nearrow s$ and $\mathbb{E}\, U_n \searrow s$) and are reverse time super- and submartingales respectively. From L_n and U_n we construct $\tilde{L}_n$ and $\tilde{U}_n$ that are monotone almost surely and have the same expectations (c.f. Theorem 2.5, Algorithm 4). Given $\tilde{L}_n$ and $\tilde{U}_n$ we sample events of probability s using a single $U(0,1)$ random variable. This result generalizes over classical constructions for simulation of events of unknown probabilities using deterministic sequences ([7]). We link these results to existence and construction of unbiased estimators. In particular one can use the algorithms of Section 2 to obtain unbiased sequential estimators of a parameter of interest that is not necessarily in $[0,1]$.

We illustrate our results with examples. First, in Section 3, we offer a statistical understanding of the Nacu-Peres algorithm that uses upper and lower polynomial envelopes for f to simulate an $f(p)-$coin (e.g. for $f(p) = \min\{2p, 1-2\varepsilon\}$). We identify coefficients of the lower and upper polynomial envelopes as random variables of desired properties and implement the algorithm using a single $U(0,1)$ auxiliary random variable. We do not need to identify subsets of $\{0,1\}$ strings and thus avoid algorithmic difficulties of the original version. Second, in Section 4 we obtain the Exact Algorithm for diffusions introduced in [3] as an application of the generic Algorithm 3 of Section 2.

2 Simulation of Events with Unknown Probabilities

Throughout the paper we assume that we can easily generate uniformly distributed iid random variables $G_0, G_1, ... \sim U(0,1)$ which will typically serve as a source of randomness for algorithms. Generic variables, not necessairy distributed as $U(0,1)$ introducing additional randomness to algorithms will be denoted as $R_0, R_1, ...$ Thus to simulate an $s-$coin C_s we just let $C_s := \mathbb{I}\{G_0 \leq s\}$. We will be concerned with settings where s is not known explicitly.

The following simple observation will turn out very useful.

Lemma 2.1. *Sampling events of probability $s \in [0,1]$ is equivalent to constructing an unbiased estimator of s taking values in $[0,1]$ with probability 1.*

Proof. Let $\hat{S}$, s.t. $\mathbb{E}\hat{S} = s$ and $\mathbb{P}(\hat{S} \in [0,1]) = 1$ be the estimator. Then draw $G_0 \sim U(0,1)$, obtain $\hat{S}$ and define a coin C_s as

$$C_s := \mathbb{I}\{G_0 \leq \hat{S}\}.$$

Clearly

$$\mathbb{P}(C_s = 1) = \mathbb{E}\,\mathbb{I}(G_0 \leq \hat{S}) = \mathbb{E}\left(\mathbb{E}\left(\mathbb{I}(G_0 \leq \hat{s}) \mid \hat{S} = \hat{s}\right)\right) = \mathbb{E}\hat{S} = s.$$

The converse is straightforward since an $s-$coin is an unbiased estimator of s with values in $[0,1]$. □

Thus given $\hat{S} \in [0,1]$, an unbiased estimator of s, we can sample events of probability s by the following algorithm that uses two sources of randomness $G_0 \sim U(0,1)$ and $\hat{S} = \hat{S}(R_0)$. In a sense $\hat{S}(R_0)$ carries information about s and G_0 is an auxiliary random variable. In the proof above we implicitely assumed that G_0 and R_0 are independent. We shall drop $R_0, R_1, ...$ from notation unless we focus on them.

Algorithm 1.
1. simulate $G_0 \sim U(0,1)$;
2. obtain $\hat{S}$;
3. if $G_0 \leq \hat{S}$ set $C_s := 1$, otherwise set $C_s := 0$;
4. output C_s.

Next assume that $l_1, l_2, ...$ and $u_1, u_2, ...$ are sequences of lower and upper bounds for s converging to s. This setting is well known ([7]) and appears in a variety of situations, usually as an element of more complex simulation procedures, see e.g. [5], [15]. Here we use the following algorithm for simulating an $s-$coin.

Algorithm 2.
1. simulate $G_0 \sim U(0,1)$; set $n = 1$;
2. compute l_n and u_n;
3. if $G_0 \leq l_n$ set $C_s := 1$;
4. if $G_0 > u_n$ set $C_s := 0$;
5. if $l_n < G_0 \leq u_n$ set $n := n + 1$ and GOTO 2;
6. output C_s.

The algorithm stops with probability 1 since l_n and u_n converge to s from below and from above. Precisely, the algorithm needs $N > n$ iterations to stop with probability $\inf_{k \leq n} u_k - \sup_{k \leq n} l_k$. Because we can always obtain monotone bounds by setting $u_n := \inf_{k \leq n} u_k$ and $l_n := \sup_{k \leq n} l_k$, we assume that l_n is an increasing sequence and u_n is a decreasing sequence.

The next step is to combine the above ideas and work with randomized bounds, i.e. in a setting where we have estimators L_n and U_n of the upper and lower bounds l_n and u_n. The estimators shall live on the same probability space

and have the following properties.

$$\mathbb{P}(L_n \le U_n) = 1 \qquad \text{for every} \quad n = 1, 2, ... \tag{1}$$

$$\mathbb{P}(L_n \in [0,1]) = 1 \quad \text{and} \quad \mathbb{P}(U_n \in [0,1]) = 1 \quad \text{for every} \quad n = 1, 2, ... \tag{2}$$

$$\mathbb{E}\, L_n = l_n \nearrow s \quad \text{and} \quad \mathbb{E}\, U_n = u_n \searrow s \tag{3}$$

$$\mathbb{P}(L_{n-1} \le L_n) = 1 \quad \text{and} \quad \mathbb{P}(U_{n-1} \ge U_n) = 1 \tag{4}$$

Note that we do not assume that $L_n \le s$ or $U_n \ge s$. Also condition (4) implies monotonicity of expectations in (3). Let

$$\mathcal{F}_0 = \{\emptyset, \Omega\}, \qquad \mathcal{F}_n = \sigma\{L_n, U_n\}, \qquad \mathcal{F}_{k,n} = \sigma\{\mathcal{F}_k, \mathcal{F}_{k+1}, ...\mathcal{F}_n\} \quad \text{for } k \le n.$$

Consider the following algorithm.

Algorithm 3.
1. simulate $G_0 \sim U(0,1)$; set $n = 1$;
2. obtain L_n and U_n given $\mathcal{F}_{0,n-1}$,
3. if $G_0 \le L_n$ set $C_s := 1$;
4. if $G_0 > U_n$ set $C_s := 0$;
5. if $L_n < G_0 \le U_n$ set $n := n + 1$ and GOTO 2;
6. output C_s.

Lemma 2.2. *Assume (1), (2), (3) and (4). Then Algorithm 3 outputs a valid $s-$coin. Moreover the probability that it needs $N > n$ iterations equals $u_n - l_n$.*

Proof. Probability that Algorithm 3 needs more then n iterations equals $\mathbb{E}(U_n - L_n) = l_n - u_n \to 0$ as $n \to \infty$. And since $0 \le U_n - L_n$ is a decreasing sequence a.s., we also have $U_n - L_n \to 0$ a.s. So there exists a random variable $\hat{S}$, such that for almost every realization of sequences $\{L_n(\omega)\}_{n\ge1}$ and $\{U_n(\omega)\}_{n\ge1}$ we have $L_n(\omega) \nearrow \hat{S}(\omega)$ and $U_n(\omega) \searrow \hat{S}(\omega)$. By (2) we have $\hat{S} \in [0,1]$ a.s. Thus for a fixed ω the algorithm outputs an $\hat{S}(\omega)-$coin a.s. Clearly $\mathbb{E}\, L_n \le \mathbb{E}\, \hat{S} \le \mathbb{E}\, U_n$ and hence $\mathbb{E}\, \hat{S} = s$. □

Remark 2.3. The random variable $\hat{S}$ constructed in the proof can be viewed as the unbiased estimator of s mentioned earlier with sequences L_n and U_n being its lower and upper random approximations.

Remark 2.4. For Algorithm 3 assumption (2) can be relaxed to

$$\mathbb{P}(L_n \in (-\infty, 1]) = 1 \quad \text{and} \quad \mathbb{P}(U_n \in [0, \infty)) = 1 \quad \text{for every} \quad n = 1, 2, ... \tag{5}$$

The final step is to weaken condition (4) and let L_n be a reverse time supermartingale and U_n a reverse time submartingale with respect to $\mathcal{F}_{n,\infty}$. Precisely, assume that for every $n = 1, 2, ...$ we have

$$\mathbb{E}\,(L_{n-1} \mid \mathcal{F}_{n,\infty}) \;=\; \mathbb{E}\,(L_{n-1} \mid \mathcal{F}_n) \;\le\; L_n \;\text{ a.s.} \quad \text{and} \tag{6}$$

$$\mathbb{E}\,(U_{n-1} \mid \mathcal{F}_{n,\infty}) \;=\; \mathbb{E}\,(U_{n-1} \mid \mathcal{F}_n) \;\ge\; U_n \;\text{ a.s.} \tag{7}$$

Consider the following algorithm, that uses auxiliary random sequences $\tilde{L}_n$ and $\tilde{U}_n$ constructed online.

Algorithm 4.
1. simulate $G_0 \sim U(0,1)$; set $n = 1$; set $L_0 \equiv \tilde{L}_0 \equiv 0$ and $U_0 \equiv \tilde{U}_0 \equiv 1$
2. obtain L_n and U_n given $\mathcal{F}_{0,n-1}$,
3. compute $L_n^* = \mathbb{E}\left(L_{n-1} \mid \mathcal{F}_n\right)$ and $U_n^* = \mathbb{E}\left(U_{n-1} \mid \mathcal{F}_n\right)$.
4. compute

$$\tilde{L}_n = \tilde{L}_{n-1} + \frac{L_n - L_n^*}{U_n^* - L_n^*}\left(\tilde{U}_{n-1} - \tilde{L}_{n-1}\right) \tag{8}$$

$$\tilde{U}_n = \tilde{U}_{n-1} - \frac{U_n^* - U_n}{U_n^* - L_n^*}\left(\tilde{U}_{n-1} - \tilde{L}_{n-1}\right) \tag{9}$$

5. if $G_0 \le \tilde{L}_n$ set $C_s := 1$;
6. if $G_0 > \tilde{U}_n$ set $C_s := 0$;
7. if $\tilde{L}_n < G_0 \le \tilde{U}_n$ set $n := n + 1$ and GOTO 2;
8. output C_s.

Theorem 2.5. *Assume (1), (2), (3), (6) and (7). Then Algorithm 4 outputs a valid $s-$coin. Moreover the probability that it needs $N > n$ iterations equals $u_n - l_n$.*

Proof. We show that $\tilde{L}$ and $\tilde{U}$ satisfy (1), (2), (3) and (4) and hence Algorithm 4 is valid due to Lemma 2.2.

Conditions (1), (2) and (4) are straightforward due to construction of $\tilde{L}$ and $\tilde{U}$ and (6), (7).

To prove (3) we show that the construction in step 4 of Algorithm 4 preserves expectation, i.e.

$$\mathbb{E}\,\tilde{L}_n = \mathbb{E}\,L_n = l_n \quad \text{and} \quad \mathbb{E}\,\tilde{U}_n = \mathbb{E}\,U_n = u_n. \tag{10}$$

It is straightforward to check that (10) holds for $n = 1, 2$. Moreover note that $\tilde{U}_0 - \tilde{L}_0 = 1$ a.s., $U_1^* - L_1^* = 1$ a.s. and from (8) and (9) we have

$$\begin{aligned} \tilde{U}_n - \tilde{L}_n &= \left(\tilde{U}_{n-1} - \tilde{L}_{n-1}\right)\frac{U_n - L_n}{U_n^* - L_n^*} \qquad \text{and hence} \\ \tilde{L}_n &= \tilde{L}_{n-1} + \frac{L_n - L_n^*}{U_n^* - L_n^*}\frac{U_{n-1} - L_{n-1}}{U_{n-1}^* - L_{n-1}^*}\cdots\frac{U_2 - L_2}{U_2^* - L_2^*}\left(U_1 - L_1\right). \end{aligned} \tag{11}$$

Now we compute $\mathbb{E}\,\tilde{L}_n$ by induction, conditioning (11) subsequently on $\mathcal{F}_{2,\infty}, \ldots, \mathcal{F}_{n,\infty}$ and using (6) and (7). Calculation of $\mathbb{E}\,\tilde{U}_n$ is identical.

$$\begin{aligned} \mathbb{E}\,\tilde{L}_n &= \mathbb{E}\,\tilde{L}_{n-1} + \mathbb{E}\left(\mathbb{E}\left(\frac{L_n - L_n^*}{U_n^* - L_n^*}\frac{U_{n-1} - L_{n-1}}{U_{n-1}^* - L_{n-1}^*}\cdots\frac{U_2 - L_2}{U_2^* - L_2^*}\left(U_1 - L_1\right)\Big|\mathcal{F}_{2,\infty}\right)\right) \\ &= \mathbb{E}\,L_{n-1} + \mathbb{E}\left(\frac{L_n - L_n^*}{U_n^* - L_n^*}\frac{U_{n-1} - L_{n-1}}{U_{n-1}^* - L_{n-1}^*}\cdots\frac{U_2 - L_2}{U_2^* - L_2^*}\mathbb{E}\left(U_1 - L_1\big|\mathcal{F}_{2,\infty}\right)\right) \\ &= \mathbb{E}\,L_{n-1} + \mathbb{E}\left(\frac{L_n - L_n^*}{U_n^* - L_n^*}\frac{U_{n-1} - L_{n-1}}{U_{n-1}^* - L_{n-1}^*}\cdots\frac{U_3 - L_3}{U_3^* - L_3^*}\left(U_2 - L_2\right)\right) = \cdots \\ &= \mathbb{E}\,L_{n-1} + \mathbb{E}\left(L_n - L_n^*\right) = \mathbb{E}\left(\mathbb{E}\left(L_{n-1} + L_n - L_n^*\big|\mathcal{F}_{n,\infty}\right)\right) = \mathbb{E}\,L_n. \end{aligned}$$

□

Remark 2.6. All of the discussed algorithms are valid if n takes values along an increasing sequence $n \nearrow \infty$.

Now let us link once again the algorithmic development of this Section with construction of unbiased estimators. Lemma 2.1 together with Theorem 2.5 result in the following construction of sequential unbiased estimators based on under- and overestimating reverse time super- and submartingale sequences. The estimators are sequential in the sense that the amount of input needed to produce them is random.

Theorem 2.7. *Suppose that for an unknown value of interest $s \in \mathbf{R}$, there exist a constant $M < \infty$ and random sequences L_n and U_n s.t.*

$$\begin{array}{rcl} \mathbb{P}(L_n \le U_n) = 1 & & \text{for every} \quad n = 1, 2, \ldots \\ \mathbb{P}(L_n \in [-M, M]) = 1 & \text{and} & \mathbb{P}(U_n \in [-M, M]) = 1 \quad \text{for every} \quad n = 1, 2, \ldots \\ \mathbb{E}\, L_n = l_n \nearrow s & \text{and} & \mathbb{E}\, U_n = u_n \searrow s \\ \mathbb{E}\,(L_{n-1} \mid \mathcal{F}_{n,\infty}) & = & \mathbb{E}\,(L_{n-1} \mid \mathcal{F}_n) \le L_n \quad a.s. \quad \text{and} \\ \mathbb{E}\,(U_{n-1} \mid \mathcal{F}_{n,\infty}) & = & \mathbb{E}\,(U_{n-1} \mid \mathcal{F}_n) \ge U_n \quad a.s. \end{array}$$

Then one can construct an unbiased estimator of s.

Proof. After rescaling, one can use Algorithm 4 to sample events of probability $(M+s)/2M$, which gives an unbiased estimator of $(M+s)/2M$ and consequently of s. □

3 Application to the Bernoulli Factory Problem

Based on Section 2, we provide here a practical version of the Nacu-Peres algorithm for simulating an $f(p)-$coin form a sequence of $p-$coins, where $f(p) = \min\{2p, 1 - 2\varepsilon\}$. This is central to the general version of the Bernoulli factory problem, as [13] develops a calculus for collapsing simulation of a real analytic function, say g, to simulation of $f(p) = \min\{2p, 1 - 2\varepsilon\}$. Briefly, one takes a series expansion of g and uses a composition of appropriate techniques (e.g. for simulating a sum or a difference of simulable functions). We refer to the original paper for details.

In particular we prove Proposition 3.1, a general result, which is a minor modification of Proposition 3 in [13]. However its proof, different from the original one, links polynomial envelopes of f with the framework of Section 2 by identifying terms. It results in an immediate application of Algorithm 4.

Proposition 3.1. *An algorithm that simulates a function f on $\mathcal{P} \subseteq (0, 1)$ exists if and only if for all $n \ge 1$ there exist polynomials $g_n(p)$ and $h_n(p)$ of the form*

$$g_n(p) = \sum_{k=0}^{n} \binom{n}{k} a(n,k) p^k (1-p)^{n-k} \quad \text{and} \quad h_n(p) = \sum_{k=0}^{n} \binom{n}{k} b(n,k) p^k (1-p)^{n-k},$$

s.t.

(i) $0 \le a(n,k) \le b(n,k) \le 1$,

(ii) $\lim_{n\to\infty} g_n(p) = f(p) = \lim_{n\to\infty} h_n(p)$,

(iii) For all $m < n$, their coefficients satisfy

$$a(n,k) \ge \sum_{i=0}^{k} \frac{\binom{n-m}{k-i}\binom{m}{i}}{\binom{n}{k}} a(m,i), \qquad b(n,k) \le \sum_{i=0}^{k} \frac{\binom{n-m}{k-i}\binom{m}{i}}{\binom{n}{k}} b(m,i). \quad (12)$$

Proof. We skip the implication *algorithm* $\Rightarrow$ *polynomials,* as it has been shown in [13], and focus on proving *polynomials* $\Rightarrow$ *algorithm* using framework of Section 2. Let $X_1, X_2, \dots$ be a sequence of independent tosses of a $p-$coin. Define random sequences $\{L_n, U_n\}_{n\geq 1}$ as follows: if $\sum_{i=1}^{n} X_i = k$, then let $L_n = a(n,k)$ and $U_n = b(n,k)$. In the rest of the proof we check that (1), (2), (3), (6) and (7) hold for $\{L_n, U_n\}_{n\geq 1}$ with $s = f(p)$. Thus executing Algorithm 4 with $\{L_n, U_n\}_{n\geq 1}$ yields a valid $f(p)-$coin.

Clearly (1) and (2) hold due to (i). For (3) note that $\mathbb{E}\, L_n = g_n(p) \nearrow f(p)$ and $\mathbb{E}\, U_n = h_n(p) \searrow f(p)$. To obtain (6) and (7) define the sequence of random variables H_n to be the number of heads in $\{X_1, \dots, X_n\}$, i.e. $H_n = \sum_{i=1}^{n} X_i$ and let $\mathcal{G}_n = \sigma(H_n)$. Thus $L_n = a(n, H_n)$ and $U_n = b(n, H_n)$, hence $\mathcal{F}_n \subseteq \mathcal{G}_n$ and it is enough to check that $\mathbb{E}(L_m|\mathcal{G}_n) \leq L_n$ and $\mathbb{E}(U_m|\mathcal{G}_n) \geq U_n$ for $m < n$. The distribution of H_m given H_n is hypergeometric and

$$\mathbb{E}(L_m|\mathcal{G}_n) = \mathbb{E}(a(m, H_m)|H_n) = \sum_{i=0}^{H_n} \frac{\binom{n-m}{H_n-i}\binom{m}{i}}{\binom{n}{H_n}} a(m,i) \leq a(n, H_n) = L_n.$$

Clearly the distribution of H_m given H_m is the same as the distribution of H_m given $\{H_n, H_{n+1}, \dots\}$. The argument for U_n is identical. $\square$

Remark 3.2. In contrast to [13], throughout this section we simulate f in the weak sense, i.e. we use $U(0,1)$ as an auxiliary random variable. This is the natural approach in applications and also this is equivalent to strong simulability if $\mathcal{P} \subseteq (0,1)$, c.f. [10].

Section 3 of [13] provides explicit formulas for polynomial envelopes of $f(p) = \min\{2p, 1-2\varepsilon\}$ that satisfy conditions of Proposition 3.1, precisely $a(n,k)$ and $b(n,k)$ satisfy (ii) and (iii) and one can easily compute $n_0 = n_0(\varepsilon)$ s.t. for $n \geq n_0$ condition (i) also holds, which is enough for the algorithm (however n_0 is substantial, e.g. $n_0(\varepsilon) = 32768$ for $\varepsilon = 0,1$ and it increases as ε decreases). By Theorem 2.5 the probability that Algorithm 4 needs $N > n$ inputs equals $h_n(p) - g_n(p)$. The polynomials provided in [13] satisfy $h_n(p) - g_n(p) \leq C\rho^n$ for $p \in [0, 1/2 - 4\varepsilon]$ guaranteeing fast convergence, and $h_n(p) - g_n(p) \leq Dn^{-1/2}$ elsewhere. Using similar techniques one can establish polynomial envelopes s.t. $h_n(p) - g_n(p) \leq C\rho^n$ for $p \in [0,1] \setminus (1/2-(2+c)\varepsilon, 1/2-(2-c)\varepsilon)$. We do not pursue this here, however in applications it will be often essential to obtain polynomial approximations tailored for a specific problem and with desired properties. Moreover, we note that despite the fact that the techniques developed in [13] for simulating a real analytic g exhibit exponentially decaying tails, they are often not practical. Nesting k times the algorithm for $f(p) = \min\{2p, 1-2\varepsilon\}$ is very inefficient. One needs at least $n_0(\varepsilon)^k$ of original $p-$coins for a single output.

To give a more complete view of the Bernoulli factory problem in the framework of Section 2 we show, as a corollary from Lemma 2.1, a result originally established in [10] and also provided in [13], namely that generating $\min\{2p, 1\}-$coins from $p-$coins is not possible.

Corollary 3.3. *An algorithm that simulates $f(p) = 2p$ for $p \in \mathcal{P} = (0, 1/2)$ does not exists.*

Proof. We show that there does not exists an unbiased estimator of $2p$ for $p \in (0, 1/2)$ that takes values in $[0,1]$ and we conclude the corollary from Lemma 2.1.

Let S be such an estimator and let $X_1, X_2, \dots$ be a sequence of $p-$coins. We allow S to be sequential and use an auxiliary random variable independent of the $p-$coins. So

$$S = S\big(\{X_1, X_2, \dots, \}, T, R_0\big) = S\big(\{X_1, X_2, \dots, X_T\}, R_0\big),$$

where T is a stopping time with respect to $\sigma\{\mathcal{F}_{1,n}, \mathcal{G}\}$, where $\{\mathcal{F}_{1,n}\}_{n\geq 1}$ is the filtration generated by $X_1, X_2, \dots$ and $\mathcal{G}$ is a $\sigma-$algebra independent of $\mathcal{F}_{1,n}$ and generated by R_0. Clearly the joint distribution of $\{\{X_1, X_2, \dots\}, T, R_0\}$ depends on p. We denote it by $\mathbb{P}_p$ and let $\mathbb{P}_{p|t}$ be the projection of $\mathbb{P}_p$ on $\{X_1, X_2, \dots, X_t\}$. Now fix $p = 1/4$. Since $2p = 1/2$ we have $\delta := \mathbb{P}_{1/4}(S \leq 1/2) > 0$. Moreover there exists such an t_0 that

$$\mathbb{P}_{1/4}(S \leq 1/2;\ T \leq t_0) \geq \delta/2.$$

Note that $\mathbb{P}_{p|t_0}$ is absolutely continuous with respect to $\mathbb{P}_{1/4|t_0}$ for all $p \in [1/4, 1/2)$ and

$$\inf_{p\in[1/4,1/2)} \inf_{A\subseteq\{0,1\}^{t_0}} \frac{\mathbb{P}_{p|t_0}(A)}{\mathbb{P}_{1/4|t_0}(A)} \geq 2^{-t_0},$$

and consequently for every $p \in [1/4, 1/2)$ we have

$$\mathbb{P}_p(S \leq 1/2) \geq \mathbb{P}_p(S \leq 1/2;\ T \leq t_0) \geq 2^{-t_0}\mathbb{P}_{1/4}(S \leq 1/2;\ T \leq t_0) \geq \delta 2^{-(t_0+1)}.$$

Now let $p \to 1/2$. This combined with $S \in [0, 1]$ contradicts unbiasedness. □

4 Application to the exact simulation of diffusions

In this Section we derive the Exact Algorithm for diffusions introduced by [3] as a specific application of Algorithm 3 of Section 2. Before proceeding, let us clarify that exact simulation algorithms for diffusions have been considerably generalized, compared to the case we consider here, in [4, 5] based on constructions of the Brownian motion and involving auxiliary Poisson processes.

The problem can be described as follows. We are interested in simulating X_T which is the solution at time $T > 0$ of the following Stochastic Differential Equation (SDE):

$$\mathrm{d}X_t = \alpha(X_t)\,\mathrm{d}t + \mathrm{d}W_t, \quad X_0 = x \in \mathbf{R},\ t \in [0, T] \tag{13}$$

driven by the Brownian motion $\{W_t\,;\, 0 \leq t \leq T\}$, where the drift function α is assumed to satisfy the regularity conditions that guarantee the existence of a weakly unique, global solution of (13), see ch.4 of [11]. Let $\Omega \equiv C([0, T], \mathbf{R})$ be the set of continuous mappings from $[0, T]$ to $\mathbf{R}$ and ω be a typical element of Ω. Consider the co-ordinate mappings $B_t : \Omega \mapsto \mathbf{R}$, $t \in [0, T]$, such that for any t, $B_t(\omega) = \omega(t)$ and the cylinder σ-algebra $\mathcal{C} = \sigma(\{B_t\,;\, 0 \leq t \leq T\})$. We denote by $W^x = \{W^x_t\,;\, 0 \leq t \leq T\}$ the Brownian motion started at $x \in \mathbf{R}$, and by $W^{x,u} = \{W^{x,u}_t\,;\, 0 \leq t \leq T\}$ the Brownian motion started at x and finishing at $u \in \mathbf{R}$ at time T; the latter is known as the Brownian bridge. We make the following assumptions for α:

1. The drift function α is differentiable.

2. The function $h(u) = \exp\{A(u) - (u-x)^2/2T\}$, $u \in \mathbf{R}$, for $A(u) = \int_0^u \alpha(y)\mathrm{d}y$, is integrable.

3. The function $(\alpha^2 + \alpha')/2$ is bounded below by $\ell > -\infty$, and above by $r + \ell < \infty$.

Then, let us define

$$\phi(u) = \frac{1}{r}[(\alpha^2 + \alpha')/2 - \ell] \in [0,1]\,, \tag{14}$$

$\mathbb{Q}$ be the probability measure induced by the solution X of (13) on $(\Omega, \mathcal{C})$, $\mathbb{W}$ the corresponding probability measure for W^x, and $\mathbb{Z}$ be the probability measure defined as the following simple change of measure from $\mathbb{W}$: $\mathrm{d}\mathbb{W}/\mathrm{d}\mathbb{Z}(\omega) \propto \exp\{-A(B_T)\}$. Note that a stochastic process distributed according to $\mathbb{Z}$ has similar dynamics to the Brownian motion, with the exception of the distribution of the marginal distribution at time T which is biased according to A. Hence, we refer to this process as the biased Brownian motion. In particular, the biased Brownian motion conditional on its value at time T has the same law as the corresponding Brownian bridge.

The final steps of the mathematical developement entail resorting to the Girsanov transformation of measures (see for instance ch.8 of [14]) to obtain $\mathrm{d}\mathbb{Q}/\mathrm{d}\mathbb{W}$; applying an integration-by-parts (possible by means of Assumption 1) to eliminate the stochastic integral involved in the Radon-Nikodym derivative; and using the definition of $\mathbb{Z}$ to obtain that

$$\frac{\mathrm{d}\mathbb{Q}}{\mathrm{d}\mathbb{Z}}(\omega) \propto \exp\left\{-rT\int_0^T T^{-1}\phi(B_t)\mathrm{d}t\right\} \leq 1 \qquad \mathbb{Z}-\text{a.s.} \tag{15}$$

The details of this argument can be found in [3, 5]. By a standard rejection sampling principle, it follows that a path ω generated according to $\mathbb{Z}$ and accepted with probability (15) it yields a draw from $\mathbb{Q}$. Hence, the following algorithm yields an exact sample from the solution of (13) at time T:

1. simulate $u \sim h$
2. generate a C_s coin where $s := e^{-rTJ}$, and $J := \int_0^T T^{-1}\phi(W_t^{x,u})\mathrm{d}t$;
3. If $C_s = 1$ output u and STOP;
4. If $C_s = 0$ GOTO 1.

Exploiting the Markov property, we can assume from now on that $rT < 1$. If T is such that $rT > 1$, then we can divise sub-intervals of length δ such that $r\delta < 1$ and apply the algorithm sequentially.

Clearly, the challenging part of the algorithm is Step 2, since exact computation of J is impossible due to the integration over a Brownian bridge path. On the other hand, it is easy to generate J-coins: $C_J = \mathbb{I}(\psi < \phi(W_\chi^{x,u}))$, where $\psi \sim U(0,1)$ and $\chi \sim U(0,T)$ independent of the Brownian bridge $W^{x,u}$ and of each other. Therefore, we deal with another instance of the problem studied in this article: given p-coins how to generate $f(p)$-coins, where here f is the exponential function. One possibility would be to follow [13] to first reduce this problem to one of simulating $\min\{2p, 1-2\varepsilon\}$-coins, and simulate the latter as in Section 3 using Algorithm 4.

However, a much more efficient algorithm exists which exploits the properties of the exponential function and the mechanism which generates the J-coins. We start by expanding the exponential function into a power series and constructing bounding sequences:

$$\begin{aligned} e^{-rTJ} &= \sum_{i=0}^{\infty}(rT)^i(-J)^i/i! \\ u_n &= \sum_{i=0}^{2n}(rT)^i(-J)^i/i!\,, \qquad \text{for } n=0,1,\ldots \\ l_n &= \sum_{i=0}^{2n+1}(rT)^i(-J)^i/i!\,, \qquad \text{for } n=0,1,\ldots \end{aligned}$$

Although the bounding sequences l_n and u_n are intractable, unbiased estimators of them can be devised. It is easy to check directly that letting $U_0 = 1$, and given $v \sim U(0,1)$ and a sequence of independent $\psi_j \sim U(0,1), \chi_j \sim U(0,T)$, the following iteratively constructed stochastic bounds

$$\begin{aligned} L_n &= U_n - \mathbb{I}\left(v < \frac{(rT)^{2n+1}}{(2n+1)!}\right)\prod_{i=1}^{2n+1}\mathbb{I}(\psi_i < \phi(W^{x,u}_{\chi_i}))\,, \quad n \geq 0 \\ U_n &= L_{n-1} + \mathbb{I}\left(v < \frac{(rT)^{2n}}{(2n)!}\right)\prod_{i=1}^{2n}\mathbb{I}(\psi_i < \phi(W^{x,u}_{\chi_i}))\,, \quad n \geq 1 \end{aligned}$$

satisfy the requirements of Lemma 2.2 and Algorithm 3 can be directly used to simulate an event of probability e^{rtJ}.

Acknowledgements

We would like to thank Şerban Nacu and Yuval Peres for helpful comments. The first author would also like to thank Wojciech Niemiro for a helpful discussion. The third author would like to acknowledge financial support by the Spanish government through a "Ramon y Cajal" fellowship and the grant MTM2008-06660 and the Berlin Mathematical School for hosting him as a visiting Professor while preparing this manuscript.

References

[1] S. Assmussen, P. W. Glynn and H. Thorisson. Stationarity Detection in the Initial Transient Problem. *ACM Transactions on Modelling and Computer Simulation,* 2(2):130-157, 1992.

[2] S. Asmussen and P. W. Glynn *Stochastic simulation: algorithms and analysis.* Springer, New York, 2007.

[3] A. Beskos, G.O. Roberts. Exact Simulation of Diffusions. *Ann. Appl. Probab.* 15(4): 2422–2444, 2005.

[4] A. Beskos, O. Papaspiliopoulos, G.O. Roberts. Retrospective Exact Simulation of Diffusion Sample Paths with Applications. *Bernoulli* 12: 1077–1098, 2006.

[5] A. Beskos, O. Papaspiliopoulos, G.O. Roberts. A factorisation of diffusion measure and finite sample path constructions. *Methodol. Comput. Appl. Probab.* 10(1): 85–104, 2008.

[6] A. Beskos, O. Papaspiliopoulos, G.O. Roberts, and P. Fearnhead. Exact and computationally effcient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* 68(3):333-382, 2006.

[7] L. Devroye. *Nonumniform Random Variable Generation.* Springer-Verlag, New York, 1986.

[8] S.G. Henderson and P. W. Glynn. Nonexistance of a class of variate generation schemes. *Operations Research Letters,* 31: 83–89, 2003.

[9] O. Holtz, F. Nazarov, and Y. Peres. New coins from old, smoothly. *eprint arXiv:* 0808.1936, 2008.

[10] M.S. Keane and G.L. OBrien. A Bernoulli factory. *ACM Transactions on Modeling and Computer Simulation (TOMACS),* 4(2):213-219, 1994.

[11] P.E. Kloeden and E.Platen. *Numerical Solution of Stochastic Differential Equations*, Springer-Verlag, 1995.

[12] E. Mossel and Y. Peres. New coins from old: computing with unknown bias. *Combinatorica,* 25(6):707-724, 2005.

[13] S. Nacu and Y. Peres. Fast simulation of new coins from old. *Annals of Applied Probability,* 15(1):93–115, 2005.

[14] , B.K. Øksendal. *Stochastic Differential Equations: An Introduction With Applications*, Springer-Verlag, 1998.

[15] O. Papaspiliopoulos, G.O. Roberts. Retrospective Markov chain Monte Carlo for Dirichlet process hierarchical models. *Biometrika,* 95:169–186, 2008.

[16] Y. Peres. Iterating von Neumann's procedure for extracting random bits. *Annals of Statistics,* 20(1): 590–597, 1992.