Project report: Exact Simulations

A computational introduction to stochastic differential equations

David Widmann

December 16, 2022

1 Motivation

The Wright-Fisher diffusion is a stochastic differential equation (SDE) model for the dynamcis of the frequency $X(t) \in [0, 1]$ of a genetic mutation in a population at times t. The SDE is of the form

$$\begin{aligned} \mathrm{d}X(t) &= \beta(X(t))\,\mathrm{d}t + \sqrt{X(t)(1 - X(t))}\,\mathrm{d}W(t) \qquad (t \in [0, T]), \\ X(0) &= X_0 \in [0, 1], \end{aligned}$$

where drift coefficient $\beta\colon [0, 1] \to \mathbb{R}$ describes different forces such as mutation and selection. The model is widely used in population genetics (see Kaj and Mugal 2016; Jenkins and Spanò 2017; Müller, Kaj, and Mugal 2022 and references therein).

One possible choice of the drift function is

$$\beta(x) := \gamma x(1 - x)$$

(Kaj and Mugal 2016), where parameter γ determines whether the diffusion process is neutral ($\gamma = 0$) or there is positive ($\gamma > 0$) or negative ($\gamma < 0$) selection of the mutant allele. Another example of the drift function in the Wright-Fisher diffusion is

$$\beta(x) := \frac{1}{2}[\theta_1(1 - x) - \theta_2 x] + \sigma x(1 - x)[x + h(1 - 2x)].$$

(Jenkins and Spanò 2017) where parameters θ_1 and θ_2 describe the mutation of two alleles and parameters σ and h model the fitness differences resulting from different number of copies of these alleles.

Simulation of the Wright-Fisher diffusion is important for statistical analysis and inference in population genetics (see e.g. Kaj and Mugal 2016; Müller, Kaj, and Mugal 2022). For instance, the so-called allele frequency spectrum is of particular interest in population genetics and can be approximated by simulations, as shown in Figure 1.

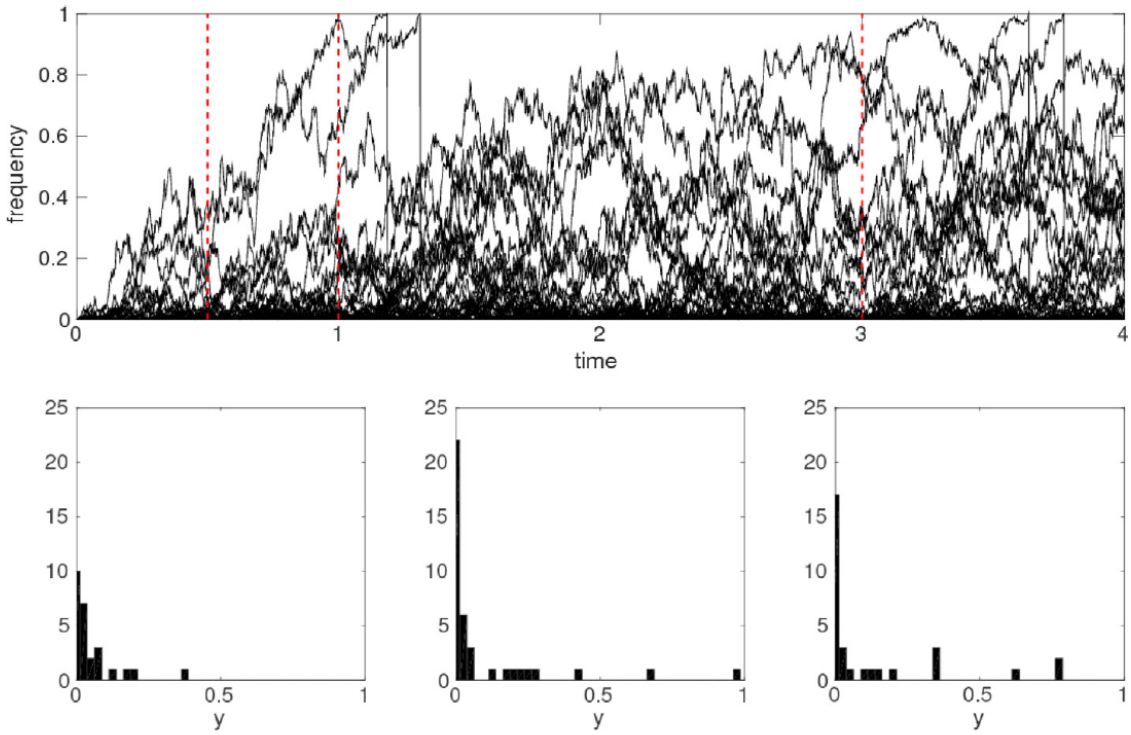


Figure 1: Simulations of a Wright-Fisher diffusion and allele frequency spectra at times $t = 0.5, 1, 3$ (Kaj and Mugal 2016).

Unfortunately, simulation of the Wright-Fisher diffusion can be challenging. As indicated in Figure 2 and Figure 3, common algorithms such as the Euler-Maruyama algorithm or the Milstein method can lead to samples that are outside of the domain of $X(t)$ (smaller than 0 or greater than 1). The diffusion coefficient cannot be evaluated for such values, and hence it is impossible to continue sampling the trajectory.

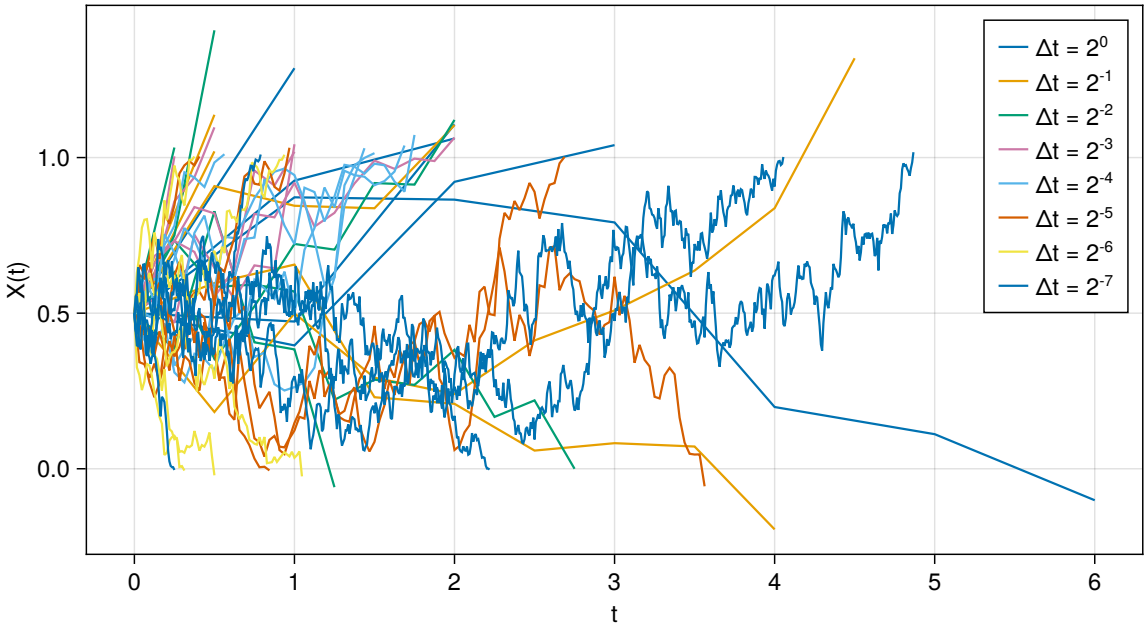


Figure 2: Simulations of the Wright-Fisher diffusion ($X_0 = 0.5$, $\beta \equiv 0$) with the Euler-Maruyama algorithm, using different time steps Δt.

Recently, Jenkins and Spanò (2017) presented an algorithm for simulating a large class of Wright-Fisher diffusions exactly. Unfortunately, their algorithm is restricted to drift coefficients of the form

$$\beta(x) = \frac{1}{2}[\theta_1(1-x) - \theta_2 x] + x(1-x)\eta(x)$$

where $\theta_1, \theta_2 > 0$ and $\eta \in C^1((0,1), \mathbb{R})$ (Jenkins and Spanò 2017, Proposition 6), and hence in particular the drift $\beta(x) = \gamma x(1-x)$ studied by Kaj and Mugal (2016) is not supported.

2 Project

In this course project we aim to study (parts of) the literature of exact simulations of SDEs. This includes main references such as the paper by Beskos and Roberts (2005) and its generalization in follow-up papers (Beskos, Papaspiliopoulos, and Roberts 2006, 2008). We also plan

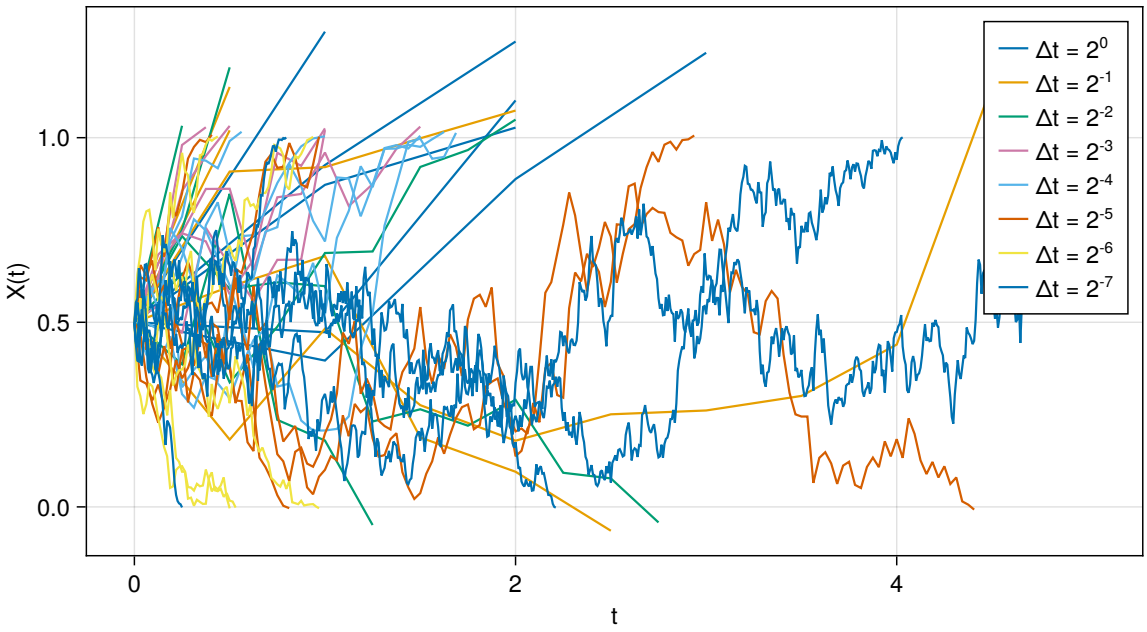


Figure 3: Simulations of the Wright-Fisher diffusion ($X_0 = 0.5$, $\beta \equiv 0$) with the Milstein method, using different time steps Δt.

to implement important and relevant algorithms in the Julia programming language (Bezanson et al. 2017), both to understand existing methods better and to be able to re-use them in future projects.

A future goal could be to release the implementations as part of a Julia software package, possibly integrated with the SciML/DifferentialEquations ecosystem. Maybe at some point we could also extend the exact algorithm by Jenkins and Spanò (2017) to the case $\theta_1 = \theta_2 = 0$.

3 Exact simulations

Many numerical methods for simulating stochastic differential equations (SDEs) such as the Euler-Maruyama method are not exact. These methods have additional approximation errors beyond unavoidable floating point errors due to the finite arithmetic of computers. That is, numerical simulations of the SDE at a fixed set of time points are *not exact* draws from the finite-dimensional distribution of the solution of the SDE.

We illustrate the approximation errors of the non-exact algorithms with an example by Nan

Chen and Zhengyu Huang (2013). We consider the Ornstein-Uhlenbeck process

$$\begin{aligned} \mathrm{d}X(t) &= -\theta X(t)\,\mathrm{d}t + \sigma\,\mathrm{d}W(t), \\ X(0) &= X_0, \end{aligned}$$

with parameters $\theta > 0$ and $\sigma > 0$. Our goal is to estimate $\mathbb{E}\,X(T)^2$ at a given time point $T > 0$.

Our first approach is based on simulating $X(T)$ with the Euler-Maruyama algorithm, using an equally spaced time grid of n steps $\Delta t = T/n$. The Euler-Maruyama method creates samples X_n of $X(T)$ by sampling

$$X_k \sim \mathcal{N}((1 - \theta\Delta t)X_{k-1}, \sigma^2\Delta t)$$

sequentially for $k = 1, \ldots, n$. We generate N independent simulations $X_n^{(1)}, \ldots, X_n^{(N)}$ by repeating the sampling procedure N times. This approach yields the Monte-Carlo estimate

$$\frac{1}{N}\sum_{i=1}^{N}\left(X_n^{(i)}\right)^2$$

of the desired quantity $\mathbb{E}\,X(T)^2$.

In contrast to most other SDEs, for the Ornstein-Uhlenbeck process actually the distribution of $X(T)$ is available in closed form. It is given by

$$p(X(T) = x \,|\, X(0) = X_0) = \mathcal{N}\Big(x; X_0\exp(-\theta T), \sigma^2(\exp(-2\theta T) - 1)/(2\theta)\Big).$$

A comparison of the samples obtained with the Euler-Maruyama method shows empirically that they are *not* distributed according to the analytical distribution of $X(T)$, and hence *not exact*.

Based on the exact distribution of $X(T)$, we obtain that the desired quantity $\mathbb{E}\,X(T)^2$ is

$$\mathbb{E}\,X(T)^2 = X_0^2\exp(-2\theta T) + \sigma^2\frac{\exp(-2\theta T) - 1}{2\theta}.$$

In Table 1 we compare the Monte-Carlo estimates for different number of time steps n and number of samples N. The root mean square error (RMSE) is evaluated with 100 trials. It is a common measure for Monte-Carlo estimators as it considers both bias and variance of the estimator. In this example, the bias is caused by the discretization of the Euler-Maruyama method and hence non-exact simulation of $X(T)$.

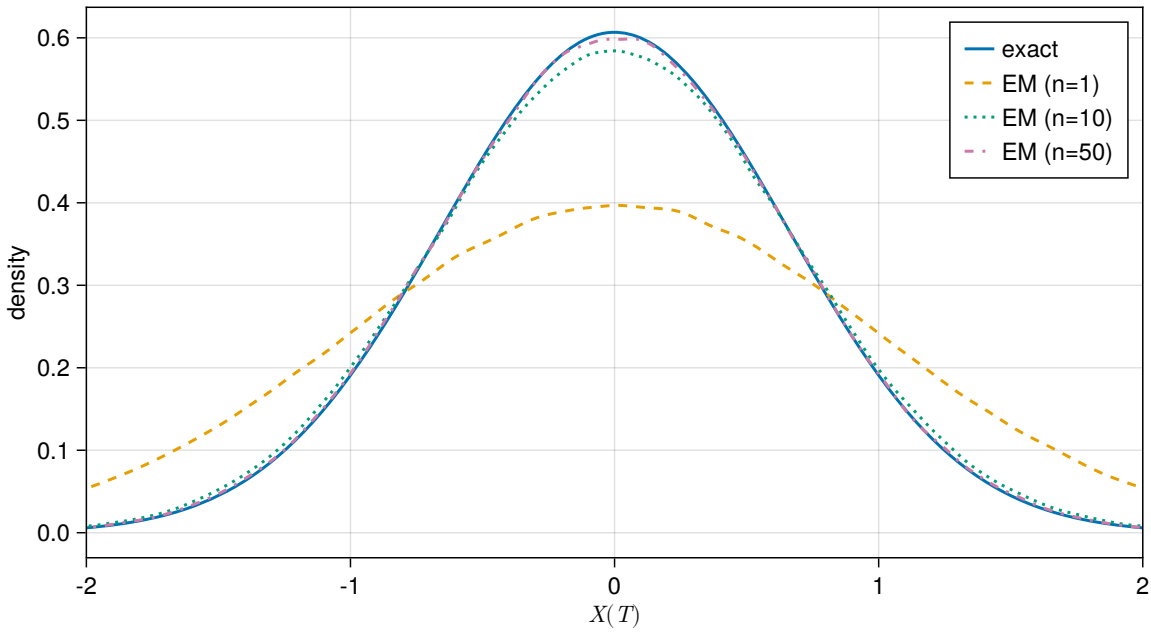


Figure 4: The true density of $X(T)$ and empirical probability densities of 10^6 samples generated with the Euler-Maruyama method ($X_0 = 0$, $T = 1$, and $\theta = \sigma = 1$). The Euler method was computed with $n = 1$, $n = 10$, and $n = 50$ time steps.

Table 1: Monte-Carlo estimates of $\mathbb{E}\, X(T)^2$ and corresponding root mean square error (RMSE).

Samples	Steps	Estimate	RMSE
400	16	0.435758	0.03571
1600	32	0.437566	0.01785
6400	64	0.449855	0.009242
25600	128	0.435983	0.004019

Our results are consistent with the findings reported by Nan Chen and Zhengyu Huang (2013).

4 Beskos-Roberts algorithm

Unfortunately, for many SDEs the transition density is not available in closed form. For instance, for the Wright-Fisher diffusion generally the transition density cannot be given explicitly and even for simple cases such as $\beta \equiv 0$ only a infinite series expansion is known. If we want to perform exact simulation in such cases, therefore we have to resort to exact algorithms that do not require explicit formulas of the transition density.

The most simple (and most restrictive) exact algorithm is the "Exact Algorithm 1" by Beskos, Papaspiliopoulos, and Roberts (2006) which is "simpler and even more efficient" than the one initially presented by Beskos and Roberts (2005). With the exact algorithm one can simulate SDEs of the form

$$\begin{aligned} \mathrm{d}X(t) &= \alpha(X(t))\,\mathrm{d}t + \mathrm{d}W(t) \qquad (t \in [0,T]), \\ X(0) &= 0. \end{aligned}$$

The exact algorithm is a rejection sampling algorithm with sample paths of a biased Brownian motion as proposals. In contrast to a standard Brownian motion, the biased Brownian motion $\widetilde{W}$ is defined to have a specific—typically non-Gaussian—density

$$h(x) \propto \exp\Big(A(x) - x^2/(2T)\Big)$$

at the final time point T where $A(x) := \int_0^x \alpha(u)\,\mathrm{d}u$ $(x \in \mathbb{R})$.

If the drift coefficient satisfies Novikov's condition

$$\mathbb{E} \exp\left(\frac{1}{2}\int_0^T \alpha^2(W(t))\,\mathrm{d}t\right) < \infty,$$

then we can use Girsanov's theorem to compute the acceptance probability of a proposal $\omega \in C([0,T],\mathbb{R})$ from the biased Brownian motion as the Radon-Nikodym derivative of the

measure P_X of the process $X(t)$ of interest with respect to the measure $P_{\widetilde{W}}$ of the biased Brownian motion:

$$\frac{\mathrm{d}}{\mathrm{d}P_{\widetilde{W}}} P_X(\omega) \propto \exp\left(-\frac{1}{2} \int_0^T \alpha^2(\omega(t)) + \alpha'(\omega(t)) \,\mathrm{d}t \right)$$

If $\alpha^2 + \alpha'$ is bounded from below, i.e., if there exists a constant $k_1 \in \mathbb{R}$ such that $\alpha^2(x) + \alpha'(x) \geq k_1$ for all x, then we can define a non-negative function $\phi \geq 0$ by

$$\phi(x) := \frac{1}{2}\Big(\alpha^2(x) + \alpha'(x) - k_1\Big)$$

and rewrite the Radon-Nikodym derivative as

$$\frac{\mathrm{d}}{\mathrm{d}P_{\widetilde{W}}} P_X(\omega) \propto \exp\left(-\int_0^T \phi(\omega(t)) \,\mathrm{d}t \right) \leq 1 \qquad P_{\widetilde{W}}\text{-almost surely.}$$

If we would be able to generate full continuous paths ω of the biased Brownian motion and could evaluate the integral

$$H(\omega) := \int_0^T \phi(\omega(t)) \,\mathrm{d}t,$$

then rejection sampling of $X(t)$ would be straightforward:

1. Sample a complete path ω from the biased Brownian motion.
2. Sample a number x from the exponential distribution Exp(1) with rate 1.
3. If $H(\omega) < x$, accept ω, otherwise go back to step 1.

In practice, however, both sampling of a complete path ω in step 1 and evaluating $H(\omega)$ in step 3 are impossible. Fortunately, one can overcome this issue and work with finite skeletons of ω. The key idea is captured by Beskos, Papaspiliopoulos, and Roberts (2006, Theorem 1) which we restate here.

Theorem 4.1. *Let ω be an element of $C([0,T],\mathbb{R})$ and $M(\omega)$ be an upper bound for the mapping $t \mapsto \phi(\omega(t))$ $(t \in [0,T])$. If Φ is a homogeneous Poisson process of unit intensity on $[0,T] \times [0, M(\omega)]$ and N is the number of points of Φ below the graph $\{(t, \phi(\omega(t))) \colon t \in [0,T]\}$, then*

$$\mathbb{P}[N = 0 \,|\, \omega] = \exp\left(-\int_0^T \phi(\omega(t)) \,\mathrm{d}t \right).$$

Theorem 4.1 shows that if we have a complete path ω of the biased Brownian motion, then actually we can accept/reject the proposal with the desired probability without computing $H(\omega)$. Moreover, we only have to know the values of ω at a finite number of time points.

Assume now that there exists some upper bound $M \in \mathbb{R}$ of ϕ. In this simple case, Theorem 4.1 leads to the following exact algorithm by Beskos, Papaspiliopoulos, and Roberts (2006), which is "simpler and even more efficient" (Beskos, Papaspiliopoulos, and Roberts 2006) than the one originally presented by Beskos and Roberts (2005). The main idea is to carry out the rejection sampling in a retrospective way by first sampling from the Poisson process and then sampling a skeleton of ω only at the time points required for computing N.

1. Generate a sample $\{(t_i, u_i)\}$ of the homogeneous Poisson process of unit intensity on $[0, T] \times [0, M]$.
2. Simulate a skeleton of ω from the biased Brownian motion at time instances $\{t_i\}$.
3. Compute the number $N = |\{i \colon u_i < \phi(\omega(t_i))\}|$.
4. If $N = 0$, accept the current skeleton of ω, otherwise go back to step 1.

To recap, we made the following assumptions:

- The drift coefficient $\alpha \colon \mathbb{R} \to \mathbb{R}$ satisfies "regularity properties that guarantee the existence of a global, weakly unique solution" (Beskos and Roberts 2005).
- The drift coefficient α is everywhere differentiable.
- There exists
$$c := \int_{\mathbb{R}} \exp\left(A(u) - u^2/(2T)\right) \mathrm{d}u < \infty$$
where $A(u) := \int_0^u \alpha(y)\, \mathrm{d}y$ $(u \in \mathbb{R})$.
- There exist constants $k_1, k_2 \in \mathbb{R}$ such that for all $u \in \mathbb{R}$
$$k_1 \leq \frac{1}{2}\alpha^2(u) + \frac{1}{2}\alpha'(u) \leq k_2.$$

The last two conditions are trivially satisfied if α and α' are lower- and upper-bounded, i.e., if there exist constants $k_1', k_2' \in \mathbb{R}$ such that for all $u \in \mathbb{R}$ it holds $k_1' \leq \alpha(u), \alpha'(u) \leq k_2'$.

4.1 Non-zero initial conditions

The assumption $X(0) = 0$ is not restrictive: When dealing with general initial conditions $X(0) = X_0 \in \mathbb{R}$, one can consider the process $Y(t) := X(t) - X_0$ instead. By Itô's lemma, we obtain
$$\begin{aligned} \mathrm{d}Y(t) &= \alpha(Y(t))\, \mathrm{d}t + \mathrm{d}W(t) \qquad (t \in [0, T]), \\ Y(0) &= 0. \end{aligned}$$
Thus we can simulate $Y(t)$ with the exact algorithm and afterwards linearly transform the simulated samples to samples of $X(t) = Y(t) + X_0$.

Alternatively, one can use the density

$$h(x) \propto \exp\Big(A(x) - (x - X_0)^2/(2 * T)\Big)$$

at the final time point T and fill in samples at additional times using Brownian bridges.

4.2 Non-unit diffusion

The exact algorithm can be generalized to SDEs with non-unit diffusion coefficients. The Lamperti transform (see Nan Chen and Zhengyu Huang 2013; Särkkä and Solin 2019 and the references therein) can be used to transform an SDE of the form

$$\begin{aligned} \mathrm{d}X(t) &= \mu(X(t))\,\mathrm{d}t + \sigma(X(t))\,\mathrm{d}W(t) \qquad (t \in [0, T]), \\ X(0) &= X_0, \end{aligned}$$

to an SDE with unit diffusion. Consider the function

$$h(x) := \int_{X_0}^{x} \frac{1}{\sigma(u)}\,\mathrm{d}u$$

for x in the support of $X(t)$. If h is well-defined for all x (e.g., if there exists a constant $c > 0$ such that $\sigma(x) > c$ for all x (Nan Chen and Zhengyu Huang 2013)), then we can define the transformed process $Y(t) := h(X(t))$. Furthermore, if σ is differentiable and the inverse h^{-1} exists (e.g., if h is strictly increasing, e.g., due to $\sigma > 0$), then by Itô's lemma $Y(t)$ satisfies the SDE

$$\begin{aligned} \mathrm{d}Y(t) &= \left[\frac{\mu(h^{-1}(Y(t)))}{\sigma(h^{-1}(Y(t)))} - \frac{1}{2}\sigma'(h^{-1}(Y(t)))\right]\mathrm{d}t + \mathrm{d}W(t) \qquad (t \in [0, T]), \\ Y(0) &= 0. \end{aligned}$$

This shows that again, under the given assumptions, we can simulate the transformed process $Y(t)$ with the exact algorithm and afterwards transform the samples of $Y(t)$ back to samples of $X(t) = h^{-1}(Y(t))$.

4.3 Implementation

For implementing the exact algorithm we note the following facts:

- One can sample points $\{z_i\}$ from a homogeneous Poisson proces with intensity λ on some domain D by a two-step procedure:

    1. Sample the number n of points from a Poisson distribution with rate $\lambda\,\mathrm{vol}(D)$.
    2. Sample n points $\{z_1, \ldots, z_n\}$ independently from the uniform distribution on D.

- One can sample a skeleton ω of the biased Brownian motion at time points $\{t_i\}$ in the following way:

  1. Fix $\omega(0) = 0$.
  2. Sample $\omega(T)$ from distribution h (e.g., with rejection sampling).
  3. For $i = 1, 2, ...,$ sequentially (but not necessarily in chronological order) sample $\omega(t_i)$ from a Brownian bridge, i.e., sample $\omega(t_i)$ according to

  $$\omega(t_i) \sim \mathcal{N}\left(\omega(t_-) + \frac{t_i - t_-}{t_+ - t_-}(\omega(t_+) - \omega(t_-)), \frac{(t_+ - t)(t - t_-)}{t_+ - t_-}\right)$$

  where $t_- = \max\left\{\{0\} \cup \{t_j\}_{j=1}^{i-1}\right\}$ and $t_+ = \min\left\{\{T\} \cup \{t_j\}_{j=1}^{i-1}\right\}$.

- To improve computational efficiency, one can perform the simulation of $\omega(t_i)$ (step 2) and the check of $u_i < \phi(\omega(t_i))$ in parallel. As soon as $u_i < \phi(\omega(t_i))$ for some i, it is clear that $N > 0$ and hence the skeleton will be rejected. Thus one can stop sampling the skeleton and go back to step 1 immediately. In principle, one could include $\{(t_i, u_i)\}$ (step 1) in the parallelization. But then time points t_i are not sampled in chronological order, and hence finding the surrounding existing time points t_- and t_+ and inserting t_i into the collection of time points becomes more expensive.

- If a skeleton of ω is accepted, one can fill in samples of values $\omega(t)$ at additional time points by continue sampling from the biased Brownian motion (i.e., by simulating from the Brownian bridge as explained above).

The following function is a Julia implementation of the Beskos-Roberts exact algorithm for bounded ϕ.

```
function beskos_roberts_exact1(
    x₀::Real,
    tspan::Tuple{<:Real,<:Real},
    h::H,
    ϕ::F,
    M::Real;
    rng::Random.AbstractRNG=Random.default_rng(),
    maxiters::Int=1_000,
) where {H,F}
    # Extract start and end times
    tstart, tend = map(float, promote(tspan...))
    T = tend - tstart

    # Create Poisson distribution
    pois = Poisson(T * M)

    # For all iterations
```

```
accepted = false
local ts, ωs
for _ in 1:maxiters
    # Sample number of time points of the skeleton
    n = rand(rng, pois)

    # Start sampling biased Brownian motion
    ts = Vector{typeof(tstart)}(undef, n + 2)
    ts[begin] = tstart
    ts[end] = tend
    d = Uniform(tstart, tend)
    sort!(rand!(rng, d, @view(ts[2:(n + 1)])))

    ωT = h(rng)
    ωs = Vector{typeof(ωT)}(undef, n + 2)
    ωs[begin] = x₀
    ωs[end] = ωT

    # For all intermediate time points
    t₋ = tstart
    ω₋ = ωs[1]
    t₊ = tend
    ω₊ = ωT
    N = 0
    for i in 2:(n + 1)
        # Sample ω(tᵢ)
        tᵢ = ts[i]
        c = (tᵢ - t₋) / T
        μ = ω₋ + c * (ω₊ - ω₋)
        σ = sqrt((t₊ - tᵢ) * c)
        ωs[i] = ωᵢ = rand(rng, Normal(μ, σ))

        # Check if the skeleton is rejected
        if M * rand(rng) < ϕ(ωᵢ)
            N += 1
            break
        end

        # Update iterates
        t₋ = tᵢ
        ω₋ = ωᵢ
    end
```

```
        # Check if the skeleton is accepted
        accepted = (N == 0)
        if accepted
            break
        end
    end

    if !accepted
        error("maximum number of iterations exceeded: ", maxiters, " proposals rejected")
    end

    return ts, ws
end
```

4.4 Example: Sine diffusion

As Beskos and Roberts (2005) and Särkkä and Solin (2019), we simulate the sine diffusion

$$\begin{aligned} \mathrm{d}X(t) &= \sin\left(X(t)\right)\mathrm{d}t + \mathrm{d}W(t) \qquad (t \in [0,T]), \\ X(0) &= X_0, \end{aligned}$$

with the exact algorithm. This SDE satisfies all assumptions of the algorithm. In particular, we have

$$-\frac{1}{2} \leq \frac{1}{2}(\alpha^2(x) + \alpha'(x)) = \frac{1}{2}(\sin^2 x + \cos x) \leq \frac{5}{8}$$

for all $x \in \mathbb{R}$. Thus we can choose

$$\phi(x) := \frac{1}{2}(\sin^2 x + \cos x + 1).$$

Hence ϕ is upper bounded by $M := 1/2 + 5/8 = 9/8$. Moreover, for all $x \in \mathbb{R}$, we have $A(x) = \int_0^x \alpha(u)\,\mathrm{d}u = 1 - \cos x$ and therefore

$$h(x) \propto \exp\left(A(x) - \frac{(x - X_0)^2}{2T}\right) \propto \exp\left(-\cos x - \frac{(x - X_0)^2}{2T}\right) =: \tilde{h}(x).$$

We can sample from h using rejection sampling with a Gaussian proposal $\mathcal{N}(X_0, T)$, based on the observation that the quotient of the unnormalized probability densities $\tilde{h}(x)$ and $\widetilde{\mathcal{N}}(x; X_0, T) := \sqrt{2\pi T}\mathcal{N}(x; X_0, T)$ satisfies

$$\frac{\tilde{h}(x)}{\widetilde{\mathcal{N}}(x; X_0, T)} = \exp\left(-\cos x\right) \leq e$$

for all $x \in \mathbb{R}$:

1. Draw a proposal x from the normal distribution $\mathcal{N}(X_0, T)$.
2. Sample a number y from the exponential distribution Exp(1) with rate 1.
3. If $\cos x + 1 < y$, accept x, otherwise go back to step 1.

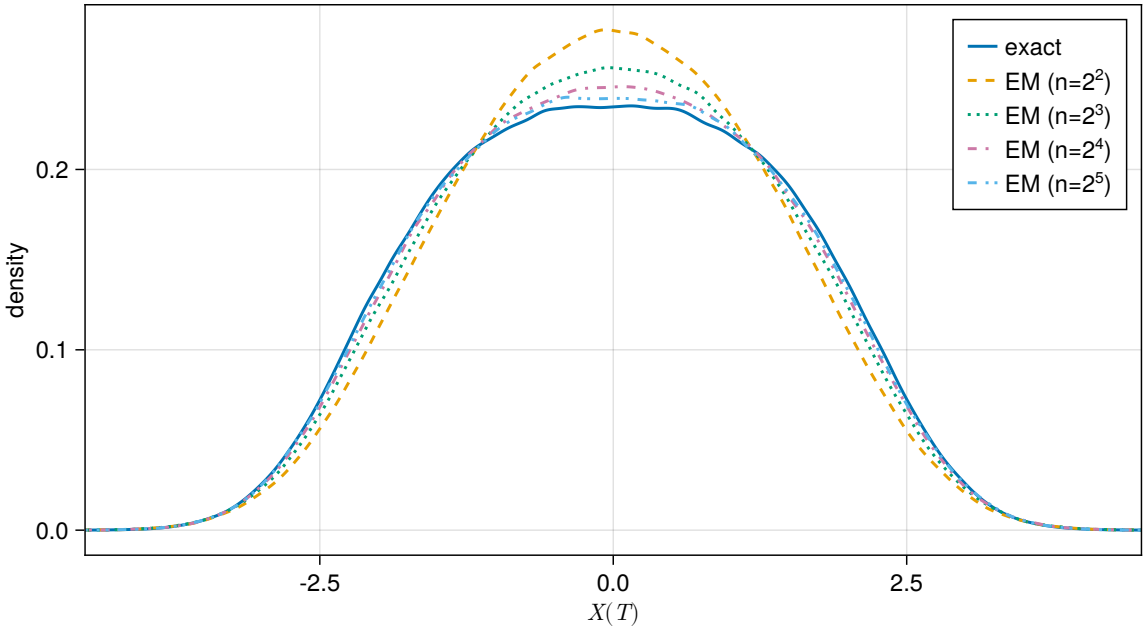
Table 2: Computation time (in seconds) of the exact algorithm and the Euler-Maruyama method for simulating 10^6 samples of $X(T)$ ($X_0 = 0$ and $T = 1$). The Euler method was computed with $n = 2^2$, $n = 2^3$, $n = 2^4$, $n = 2^5$, $n = 2^6$, and $n = 2^7$ time steps. The last column shows the p-value of the Kolmogorov-Smirnov test of the null hypothesis that the exact samples and the simulations with Euler-Maruyama are drawn from the same distribution.

Exact	Euler	Time steps	KS test
0.86	0.065	2^2	0.0
0.86	0.17	2^3	1.5e-111
0.86	0.3	2^4	1.4e-31
0.86	0.55	2^5	8.8e-12
0.86	1.2	2^6	0.0033
0.86	2.2	2^7	0.49

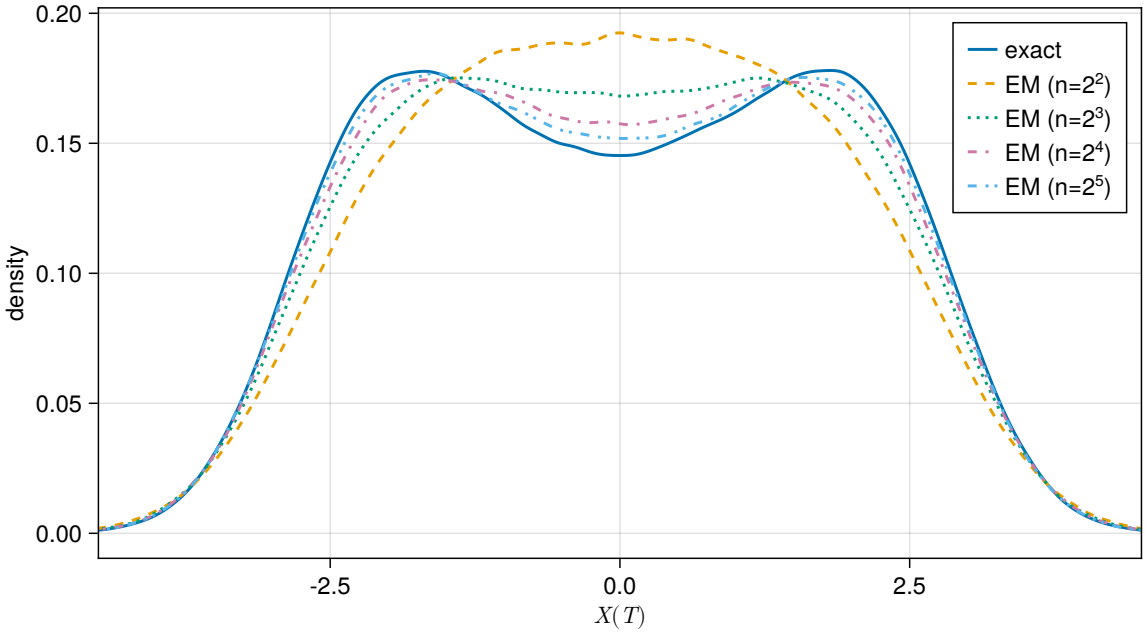
We generate samples of $X(T)$ with the exact algorithm and compare their distribution with samples of $X(T)$ using the Euler-Maruyama method with different time steps. The estimated densities of $X(T)$ for $T = 1$ and $T = 1.5$ are shown in Figure 5. The results are consistent with the visualizations by Beskos and Roberts (2005) and Särkkä and Solin (2019). In Table 2 we show the computation time of the different exact and approximate algorithms for simulating 10^6 samples of $X(1)$. We also perform Kolmogorov-Smirnoff tests of the null hypothesis that exact and approximate samples follow the same distribution. As Beskos and Roberts (2005), we see that in this example the exact algorithm is more efficient than the Euler-Maruyama method.

5 Conclusion

In this project we have learnt about algorithms for exact simulation of SDEs. Exact simulation remove all sources for approximation errors apart from floating-point issues, and hence one avoids having to rely on results that only hold in the limit as the time steps approach zero. This is illustrated by the example of the Ornstein-Uhlenbeck process in the report. We read papers by Beskos and Roberts (2005), Beskos, Papaspiliopoulos, and Roberts (2006), Beskos, Papaspiliopoulos, and Roberts (2008), Nan Chen and Zhengyu Huang (2013), and Jenkins and Spanò (2017). The simple "Exact Algorithm 1" by Beskos, Papaspiliopoulos, and Roberts (2006) is explained and implemented in this report. In the case of the sine diffusion exact simulation was very efficient and outperformed the Euler-Maruyama method.



(a) $T = 1$



(b) $T = 1.5$

Figure 5: The empirical densities of 10^6 samples of $X(T)$ generated with the exact algorithm and the Euler-Maruyama method ($X_0 = 0$ and different T). The Euler method was computed with $n = 2^2$, $n = 2^3$, $n = 2^4$, and $n = 2^5$ time steps.

6 References

Beskos, Alexandros, Omiros Papaspiliopoulos, and Gareth O. Roberts. 2006. "Retrospective Exact Simulation of Diffusion Sample Paths with Applications." *Bernoulli* 12 (6): 1077–98. https://doi.org/10.3150/bj/1165269151.

———. 2008. "A Factorisation of Diffusion Measure and Finite Sample Path Constructions." *Methodology and Computing in Applied Probability* 10 (1): 85–104. https://doi.org/10.1007/s11009-007-9060-4.

Beskos, Alexandros, and Gareth O. Roberts. 2005. "Exact Simulation of Diffusions." *The Annals of Applied Probability* 15 (4): 2422–44. https://doi.org/10.1214/105051605000000485.

Bezanson, Jeff, Alan Edelman, Stefan Karpinski, and Viral B Shah. 2017. "Julia: A Fresh Approach to Numerical Computing." *SIAM Review* 59 (1): 65–98. https://doi.org/10.1137/141000671.

Jenkins, Paul A., and Dario Spanò. 2017. "Exact Simulation of the Wright–Fisher Diffusion." *The Annals of Applied Probability* 27 (3): 1478–1509. https://doi.org/10.1214/16-AAP1236.

Kaj, Ingemar, and Carina F. Mugal. 2016. "The Non-Equilibrium Allele Frequency Spectrum in a Poisson Random Field Framework." *Theoretical Population Biology* 111 (October): 51–64. https://doi.org/10.1016/j.tpb.2016.06.003.

Müller, Rebekka, Ingemar Kaj, and Carina F. Mugal. 2022. "A Nearly Neutral Model of Molecular Signatures of Natural Selection After Change in Population Size." Edited by Adam Eyre-Walker. *Genome Biology and Evolution* 14 (5). https://doi.org/10.1093/gbe/evac058.

Nan Chen, and Zhengyu Huang. 2013. "Localization and Exact Simulation of Brownian Motion-Driven Stochastic Differential Equations." *Mathematics of Operations Research* 38 (3): 591–616. https://doi.org/10.1287/moor.2013.0585.

Särkkä, Simo, and Arno Solin. 2019. *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge: Cambridge University Press. https://doi.org/10.1017/9781108186735.