Lecture 3: Mathematical Tools

(a) Random number generators and Monte-Carlo simulations

Normally Distributed Random Numbers

Assume you are given a function `rand` that returns a uniformly distributed random number, how would you use it to generate a normally distributed random number? A quick way is the *Box-Muller algorithm*:

1. Draw two independent standard uniform random numbers z_1 and z_2.
2. Compute

$$x = \sqrt{-2\ln z_1}\,\cos(2\pi z_2), \qquad y = \sqrt{-2\ln z_1}\,\sin(2\pi z_2).$$

Then, x and y are two independent random numbers that both are distributed as $\mathcal{N}(0,1)$. The proof is left as an exercise. It is simple to implement this method in MATLAB:

```
% boxmuller.m
% demonstrates Box-Muller algorithm to create
% normally distributed random numbers

n = 1e6; z1 = rand(1,n); z2 = rand(1,n);

r1 = sqrt(-2*log(z1)).*cos(2*pi*z2);
r2 = sqrt(-2*log(z1)).*sin(2*pi*z2);

dx = 0.1; x = -10:dx:10; % create x range

ps1 = hist(r1,x);        % create histogram, centers at x
ps1 = ps1/sum(ps1)*1/dx; % normalize

ps2 = hist(r2,x);        % create histogram, centers at x
ps2 = ps2/sum(ps2)*1/dx; % normalize

plot(x,ps1,x,ps2);       % plot
```

If you are surprised that x and y are actually independent, you can type `cov(r1,r2)` for a numerical check. In MATLAB, of course, we will directly use `randn` to create random numbers, but it is good to know how such numbers can be generated. If you are interested in random numbers (and you should be!), please read the corresponding chapter in the book *Numerical Recipes.* With normally distributed random numbers at hand, we can simulate sample paths of any stochastic differential equation of the form

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t \tag{1}$$

quite easily with the following algorithm:

1. Start with an initial condition $X_0 = a$
2. To advance a step in time dt, use the discretized version of (1), namely

$$X_{t+dt} \approx X_t + \mu(X_t, t)dt + r \cdot \sigma(X_t, t)\sqrt{dt} \tag{2}$$

   where $r \sim \mathcal{N}(0, 1)$.

Again, intuitively, one might think that it would be better to use a Bernoulli variable for r, but it turns out that using a normally distributed r is usually numerically more efficient (except in cases, where unbounded values of r lead to instabilities, of course).

Monte-Carlo simulation of the Black-Scholes model

As an example, consider the Black-Scholes model with no interest rates where, using the risk-neutral measure, the stock process S_t is given as

$$S_t = S_0 \mathrm{e}^{\sigma \tilde{W}_t - \sigma^2 t/2} \,. \tag{3}$$

As we know, the corresponding stochastic differential equation is given by

$$dS_t = \sigma S_t dW_t \tag{4}$$

and the price V of a European call struck at K is found by computing the expectation

$$V = \mathbb{E}_{\mathbb{Q}}((S_T - K)^+) \,. \tag{5}$$

Therefore, the numerical solutions of the stochastic differential equation (4) offer also a way to compute the option price: Imagine we are given N paths of the stock price, then we can compute for each path the realization

$$V^{(i)} = (S_T^{(i)} - K)^+ \tag{6}$$

and estimate V as the mean, hence

$$V \approx \frac{1}{N} \sum_i V^{(i)} = \frac{1}{N} \sum_i (S_T^{(i)} - K)^+ \,. \tag{7}$$

Such a numerical simulation is called a *Monte-Carlo* simulation: You roll the dice many times and compute the expectation of an observable by counting. The following MATLAB code shows how this can be done for the Black-Scholes model:

```
function [v, vAna] = euroMC(N)
% euroMC.m Monte-Carlo simulation for Black-Scholes
%          no interest rates

S0 = 100; K = 100; sig = 0.2; T = 0.5;

nSteps = 1000; dt = T/nSteps;

S = S0*ones(N,1); % set initial condition

for k=1:nSteps
   S = S + S*sig*sqrt(dt).*randn(N,1); % solve SDE
end

v = sum(max(S-K,0))/N; % numerical option price

d1 = (log(S0/K)+sig^2/2*T)/(sig*sqrt(T));
d2 = (log(S0/K)-sig^2/2*T)/(sig*sqrt(T));

Nd1 = 0.5*erfc(-d1/sqrt(2)); Nd2 = 0.5*erfc(-d2/sqrt(2));

vAna = S0*Nd1 - K*Nd2; % analytical option price

end
```

Playing with the parameter N, the number of sample paths, we see that even a fairly small number (say $N = 10000$) gives a good idea of the option price. However, to obtain an accurate answer, we need many more paths. This is due to the fact that, usually, Monte-Carlo simulations converge roughly

$\sim 1/\sqrt{N}$. And things can get worse: Imagine an option that is far out of the money (set in the code, e.g. $K = 200$). Then, again for fairly small values of N, (say again $N = 10000$), the numerical simulation will, most of the time, compute a zero price as there are no 'hits' in such a sample, given that it is very unlikely that a stock path reaches that value. Note: by 'unlikely' we mean, of course, in terms of the *risk-neutral* probability measure. However, there are ways to speed up such simulations by *biasing* the noise in a way that the extreme event will be more likely and, on the other hand, compensating for this bias when computing the expectation. This technique is called *importance sampling* and we will discuss it in a later lecture.

(b) Fourier transform and fast Fourier transform (FFT)

The Fourier Transform

Fourier transforms are used in a wide range of situations, in particular to compute derivatives. Recall that the Fourier transform $\hat{f}$ of a function f is defined as

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} f(x) \mathrm{e}^{-i\omega x}\, dx \tag{8}$$

and, knowing the Fourier transform $\hat{f}(\omega)$ we can reconstruct the original function $f(x)$ by its inverse transform

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \hat{f}(\omega) \mathrm{e}^{i\omega x}\, d\omega\,. \tag{9}$$

With the above convention, the differential operator ∂_x is diagonal in Fourier space and we have

$$\partial_x \to i\omega \tag{10}$$

A typical example is the solution of the heat equation, given for example by

$$u_t = \frac{1}{2} u_{xx}, \qquad u(t=0,x) = \delta(x)\,. \tag{11}$$

In Fourier space, the equation becomes simply

$$\hat{u}_t = -\frac{1}{2}\omega^2 \hat{u}, \qquad \hat{u}(t=0,\omega) = u_0(\omega) = \frac{1}{\sqrt{2\pi}} \tag{12}$$

This is a simple ordinary differential equation with the solution

$$\hat{u}(t,\omega) = \exp\left(-\frac{1}{2}\omega^2 t\right) \hat{u}_0(\omega) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\omega^2 t\right) \tag{13}$$

and we can find the solution to the original partial differential equation by solving the inverse integral

$$u(t,x) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \hat{u}(t,\omega) \mathrm{e}^{i\omega x}\, d\omega = \frac{1}{\sqrt{2\pi t}} \mathrm{e}^{-x^2/(2t)}\,. \tag{14}$$

You might have noticed that this is the probability density of Brownian motion. In the next section, we will establish a more general relationship between stochastic differential equations for a stochastic process X_t and the related partial differential equation governing the evolution of the probability density $p(x,t)$ of this stochastic process X_t.

The Fast Fourier Transform (FFT)

While many Fourier transforms can be carried out analytically (in particular using tools from complex analysis like the residue theorem), we can also compute Fourier transforms numerically. Intuitively, this means to look for an efficient approximation of the Fourier integral

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} f(x) \mathrm{e}^{-i\omega x}\, dx\,.$$

Assume that we have a function that is sampled at N points, hence we have the set $\{f_0, f_1, ..., f_{N-1}\}$ given by

$$x_j = j \cdot \Delta x, \qquad f_j = f(x_j) \qquad j = 0, ..., N-1\,. \tag{15}$$

We assume now that f is periodic on $[0, L]$, such that $f_N = f_0$ if we sampled f at the right boundary as well, which is due to periodicity unnecessary. Then, it is easy to see that this corresponds to a sampling in frequency space at frequencies

$$\omega_j = \frac{2\pi j}{L} = \frac{2\pi j}{N \cdot \Delta x}\,, \qquad j = 0, ..., N-1 \tag{16}$$

The Fourier integral over one period becomes

$$\begin{aligned} \int_0^L f(x) \mathrm{e}^{-i\omega_n x}\, dx &\approx \sum_{j=0}^{N-1} f_j \mathrm{e}^{-i\omega_n x_j} \Delta x \\ &= \sum_{j=0}^{N-1} f_j \mathrm{e}^{-2\pi i j n/N} \Delta x = F_n \Delta x \end{aligned}$$

where F_n is the *discrete* Fourier transform of the vector given by the samples of f as $(f_0, f_1, ..., f_{N-1})$, defined by

$$F_n = \sum_{j=0}^{N-1} f_j \mathrm{e}^{-2\pi ijn/N} \tag{17}$$

For the discrete transform, it is easy to see that we can obtain $(f_0, f_1, ..., f_{N-1})$ from $(F_0, F_1, ..., F_{N-1})$ by the appropriate inverse transform:

$$f_n = \frac{1}{N} \sum_{j=0}^{N-1} F_j \mathrm{e}^{2\pi ijn/N} \tag{18}$$

This is a simple consequence of the periodicity properties of the complex exponential function:

$$\begin{aligned} \frac{1}{N} \sum_{j=0}^{N-1} \left(\sum_{j'=0}^{N-1} f_{j'} \mathrm{e}^{-2\pi ij'j/N} \right) \mathrm{e}^{2\pi ijn/N} &= \frac{1}{N} \sum_{j'=0}^{N-1} \sum_{j=0}^{N-1} f_{j'} \mathrm{e}^{2\pi ij(n-j')/N} \\ &= \sum_{j'=0}^{N-1} f_{j'} \delta_{j'n} = f_n \end{aligned}$$

Note that, in practice one usually chooses frequencies

$$\omega_n = \frac{2\pi n}{L} = \frac{2\pi n}{N \cdot \Delta x}, \qquad j = -\frac{N}{2} + 1, -\frac{N}{2} + 2, ..., 0, ..., \frac{N}{2} \tag{19}$$

as it turns out to be very convenient to work with frequencies that are negative and positive.

So far, there is nothing exceptional about this transform - it appears to be simply a discretization of the continuous integral. Moreover, as intuitively it should, we need to sum up N terms to compute each frequency, hence for the computation of N frequencies we will need N^2 operations. Note that, in real applications, N can easily be large. In *computational fluid dynamics* (short *CFD*), a typical resolution is $2^{10} = 1024$ grid points for each dimension. For a cube, we have then $N = 1024^3$ and $N^2 = 1024^6$ operations to perform one single Fourier transform. The fantastic news is that, in fact, the above sums can be computed recursively bringing down the number of operations to the order $\mathcal{O}(N \log N)$. For the above example, this results in a different of eight orders of magnitude in the number of computational operations. This recursive algorithm is called *fast Fourier transform* (short *FFT*).

Computing derivatives using the FFT

We will not present details of the implementation (again, *Numerical Recipes* is a great source for more information about the FFT), but show how to use the FFT in MATLAB. A simple task is, for example, the computation of a derivative using FFT. Have a look at the following code:

```
% diffFFT Compute the derivative using fft

N = 128; % number of grid points

% Set up computational grid

xMin = -10; xMax = 10; xDomain = xMax-xMin;

x = linspace(xMin, xMax-xDomain/N, N);

% Fourier space discretization

dom = 2*pi/xDomain; om = [0:N/2,-N/2+1:-1]*dom;

f = exp(-x.^2);
fnumF = 1i*om.*fft(f);
fnum = real(ifft(fnumF));

fana = -2.*x.*exp(-x.^2);

% plot results

plot(x,fnum,x,fana,'o')
```

Also, you can see that we have computed the derivative with *spectral* accuracy: The difference between the analytical and the numerical values is of the order of machine accuracy. When using the FFT, one has to pay attention to the fact that, due to the recursive computation, the computed vector containing the elements of the Fourier transform is in a non-intuitive order: The vector starts at the zero-frequency mode, contains all the positive modes in inceasing order and then switches to the most negative frequency and continues again in increasing order. The one-line MATLAB version of the order of the frequency is

```
dom = 2*pi/xDomain; om = [0:N/2,-N/2+1:-1]*dom;
```

It is much more efficient to pre-order the frequencies and then to use them in the FFT-adapted order then to reorder them. A second caveat is that there are frequency phase factors that, sometimes, need to be taken into consideration. We will see this in the next section when using the FFT to compute a convolution.

Convolutions and FFT

In a second example, we show how to compute a convolution using FFT. Recall that the convolution $u \star v$ of two functions u and v is defined as

$$f(x) = (u \star v)(x) = \int u(\tilde{x})v(x - \tilde{x})\, d\tilde{x} \tag{20}$$

As you know probably from a previous class: The Fourier transform of the convolution is simply the product of the Fourier transforms (with a constant C that depends on the chosen normalization of the Fourier transform):

$$\hat{f}(\omega) = C\hat{u}(\omega)\hat{v}(\omega) \tag{21}$$

Therefore, it is convenient and fast to use the FFT in order to compute convolutions. Here is a code to see how this is done:

```
% convFFT Compute the convolution of two Gaussians using fft

% Set up computational grid

xMin = -10;  xMax = 10; xDomain = xMax-xMin;
x = linspace(xMin, xMax-xDomain/N, N); dx = x(2)-x(1);

% Fourier space discretization
dom = 2*pi/xDomain; om = [0:N/2,-N/2+1:-1]*dom;
phase = dx*exp(-1i*om.*xMin);

u = exp(-x.^2); v = exp(-2*x.^2);

fnumF = fft(u).*fft(v).*phase;
fnum = real(ifft(fnumF));

fana = sqrt(3*pi)/3*exp(-2/3*x.^2);

plot(x,fnum,x,fana,'o')
```

As a last example on how to use the Fourier transform, we show how to compute the solution of the heat equation with $\delta(x)$ as initial condition. Note that this example can be adapted easily to deal with any initial condition (as long as its Fourier transform exists and as long as we have sufficient decay to not have to worry too much about boundary conditions.

```
% diffFFT Solve heat equation using fft

N = 128; % number of grid points
t = 2.0; % evolution time

% Set up computational grid

xMin = -10; xMax = 10; xDomain = xMax-xMin;

dx = xDomain/N;

x = linspace(xMin, xMax-dx, N);

% Fourier space discretization

dom = 2*pi/xDomain; om = [0:N/2,-N/2+1:-1]*dom;

f = zeros(1,N); f(N/2+1) = 1/dx; % discretized delta
fnumF = fft(f).*exp(-om.^2*t/2); % propagate Fourier modes
fnum = real(ifft(fnumF));        % inverse transform

fana = 1/sqrt(2*pi*t)*exp(-x.^2/(2*t)); % analytical solution

% plot results

plot(x,fnum,x,fana,'o')
```

(c) The Fokker-Planck equation and path integrals

The Fokker Planck equation

In this section, we develop a description of the dynamics of a stochastic process in terms of its probability density. We have already seen that, for simple Brownian motion $X_t = W_t$, the associated probability density $p(t, x)$ is given by

$$p(t,x) = \frac{1}{\sqrt{2\pi t}} \mathrm{e}^{-x^2/(2t)} \tag{22}$$

and we also have observed that p is a solution to the heat equation given by

$$p_t = \frac{1}{2} p_{xx}, \qquad p(t=0,x) = \delta(x)\,. \tag{23}$$

This result can be explained and generalized in the following way. Consider again the simple case of additive noise and an SDE of the form

$$dX_t = f(X_t)dt + \sigma dW_t \tag{24}$$

In order to derive an evolution equation for the probability density p of the process X_t, we compute the time evolution of the expectation of an arbitrary function g of X_t, hence

$$\frac{d}{dt}\mathbb{E}(g(X_t)) = \int \frac{\partial}{\partial t}(p(t,x))g(x)dx \equiv \int p_t g(x)dx \tag{25}$$

Alternatively, we can also compute for a small Δt the expectation $\mathbb{E}(\Delta g(X_t))$ using Ito's lemma. Here, we compute first $\mathbb{E}(\Delta g(X_t))$ and then divide by Δt. Ito's lemma yields

$$\Delta g(X_t) = g_x\left(f(X_t)\Delta t + \sigma \Delta W\right) + \frac{1}{2} g_{xx}\sigma^2 \Delta t$$

In order to compute $\mathbb{E}(\Delta g(X_t))$, we first use $p(t, x)$ as probability density of X_t and then we take one more step Δt where we draw a random number $\pm\sqrt{\Delta t}$ with probability $1/2$ to account for the Brownian increment ΔW. Putting everything together, we obtain

$$\begin{aligned}
\mathbb{E}(\Delta g(X_t)) &= \frac{1}{2}\int p(t,x)\left(g_x\left(f(x)\Delta t + \sigma\sqrt{\Delta t}\right) + \frac{1}{2}g_{xx}\sigma^2\Delta t\right)dx \\
&+ \frac{1}{2}\int p(t,x)\left(g_x\left(f(x)\Delta t - \sigma\sqrt{\Delta t}\right) + \frac{1}{2}g_{xx}\sigma^2\Delta t\right)dx \\
&= \int p(t,x)\left(g_x\left(f(x)\Delta t\right) + \frac{1}{2}g_{xx}\sigma^2\Delta t\right)dx
\end{aligned}$$

Under mild assumptions on the function g to make sure that boundary terms vanish, we can now integrate by parts. Thus, we find that

$$\int \left(p_t + \partial_x(pf) - \frac{\sigma^2}{2} p_{xx} \right) g(x)\, dx = 0\,, \tag{26}$$

and, since g is arbitrary, we have derived a partial differential equation describing the evolution of the probability density $p(t, x)$, the so-called *Fokker-Planck equation* that corresponds to the original SDE.

$$\partial_t p(t, x) = -\partial_x(p(t, x)f(x)) + \frac{\sigma^2}{2} p_{xx} \tag{27}$$

Clearly, for $f = 0$ and $\sigma = 1$ we obtain the heat equation corresponding to the evolution of the probability density of Brownian motion.

Path Integrals

We have already seen that, for stochastic differential equations driven by Brownian motion, the corresponding Fokker-Planck equation describing the evolution of the probability density offers an alternative approach to fully describe the dynamics of the associated stochastic process. A third, and for physicists very appealing way, are path-integrals. The basic idea here is to start again from a discretization of the SDE and to consider a discretized time evolution up to a time T in N steps, where $\Delta t = T/N$ is assumed to be small (and we are interested in the limit $\epsilon \to 0$). In a numerical simulation, at each time step, we will draw a random number, and the corresponding vector of random numbers will characterize a Brownian path. The SDE maps this path onto a solution X_t of the SDE and we can try to describe this map in terms of the related probability densities. For simplicity, we consider for now a one-dimensional SDE with additive noise, hence

$$dX_t = f(X_t)dt + \sigma dW_t, \qquad \dot{X}_t = f(X_t) + \sigma \xi_t\,. \tag{28}$$

The form on the right is called *Langevin* form of the stochastic equation and is popular in the physics literature. The "derivative" of Brownian motion $\xi_t = dW_t/dt$ is called *white noise* and can be understood intuitively as a stationary stochastic process with variance $1/\Delta t$ when working with time intervals that are spaced Δt apart. While it might seem from a mathematical perspective troublesome that this variance tends to infinity for $\Delta t \to 0$, this is actually not a problem as all, since the Langevin form is really just a different way of writing the original SDE and its discretization at the n-th time step

$$\frac{X_{t+\delta t} - X_t}{\Delta t} = f(X_t) + \sigma \xi_n = f(X_t) + \sigma \cdot r_n \frac{1}{\sqrt{\Delta t}}\,, \qquad r_n \sim N(0,1) \quad (29)$$

is of course entirely equivalent to the discretization of the original SDE as one seems after multiplying the equation with Δt. It is simply a matter of taste which notation to use.
To develop a path integral formalism to describe the probability density of a path $X_1, ..., X_N$, we recall that, in general, for a N-dimensional vector of Gaussian variables $(\chi_1, ..., \chi_N)$, the joint probability density is given by

$$P((\chi_1, ..., \chi_N)) = \frac{1}{(2\pi)^{N/2}} \frac{1}{\sqrt{|A|}} \exp\left(-\frac{1}{2} \sum_{n,m=1}^{N} \chi_n A^{-1}_{nm} \chi_m\right)\,, \qquad (30)$$

where $\mathbb{E}(\chi_n \chi_m) = A_{nm}$ is the covariance matrix. In our case, the vector $(\xi_1, ..., \xi_N)$ consists of independent (hence uncorrelated) random numbers, and we obtain the much simpler formula describing the joint probability density of the vector $(\xi_1, ..., \xi_N)$ with $\mathbb{E}(\xi_n \xi_m) = 1/\Delta t \delta_{nm}$

$$P((\xi_1, ..., \xi_N)) = \left(\frac{\Delta t}{2\pi}\right)^{N/2} \exp\left(-\frac{1}{2}\sum_{n=1}^{N} \xi_n^2 \Delta t\right)\,. \qquad (31)$$

While we have not done much (so far!), we have reformulated and rewritten our previous results an a very particular form. In the above expression, a term

$$\sum_{n=1}^{N} \xi_n^2 \Delta t$$

and, again from a physics point of view, it is very tempting to try to go to a continuous limit of the form

$$\sum_{n=1}^{N} \xi_n^2 \Delta t \; \rightarrow \int_0^T \xi_t^2 dt$$

This defines then, in the continuum limit, the probability density of a Brownian path (without normalization factors) as

$$\mathcal{P}(\xi_t) = \exp\left(-\frac{1}{2}\int_0^T \xi_t^2 dt\right) \qquad (32)$$

where $\Delta t \to 0$ such that $N\Delta t = T$ is fixed. It is fine to feel, at this stage, slightly uncomfortable with this construction, in particular as we have, implicitly, introduced infinitely many integration variables. Think, for instance, about the normalization of the probability density P in the finite-dimensional world. Clearly, we have

$$\begin{aligned} 1 &= \int_{\mathbb{R}^n} P((\xi_1, ..., \xi_N))\, d\xi_1, ..., d\xi_N \\ &= \int_{\mathbb{R}^n} \left(\frac{\Delta t}{2\pi}\right)^{N/2} \exp\left(-\frac{1}{2}\sum_{n=1}^{N} \xi_n^2 \Delta t\right) d\xi_1, ..., d\xi_N \,. \end{aligned}$$

How would we even *write* the continuum limit of this formula (even if we are not worried about its mathematical meaning)? This is how it is done: We need to consider the set $\mathcal{C}([0,T])$ of all possible Brownian paths over the time interval $[0,T]$ and write therefore

$$1 = \int_{\mathcal{C}([0,T])} \mathcal{P}(\xi_t)\mathcal{D}\xi_t = \int_{\mathcal{C}([0,T])} \exp\left(-\frac{1}{2}\int_0^T \xi_t^2 dt\right) \mathcal{D}\xi_t \tag{33}$$

where all the normalization factors (and the infinitely many differentials) got "absorbed" in the symbol $\mathcal{D}\xi_t$. Starting from this expression, we can now easily find a description of the path integral for a general SDE. Consider again the discretization at the n-th time step

$$\frac{X_n - X_{n-1}}{\Delta t} = f(X_t) + \sigma \xi_n$$

All we need to do is to solve this equation ξ_n and put the result into the joint probability density and take the continuum limit. Hence look at

$$\xi_n = \frac{1}{\sigma}\left(\frac{X_n - X_{n-1}}{\Delta t} - f(X_t)\right) \tag{34}$$

and, therefore at

$$\sum_{n=1}^{N} \xi_n^2 \Delta t = \sum_{n=1}^{N} \frac{1}{\sigma^2}\left(\frac{X_n - X_{n-1}}{\Delta t} - f(X_t)\right)^2 \Delta t \to \frac{1}{\sigma^2}\int_0^T (\dot{X}_t - f(X_t))^2\, dt$$

As mentioned before, this corresponds to a change of variables, mapping the Brownian path $(\xi_1, ...\xi_N)$ to a solution of the SDE given by $(X_1, ..., X_N)$. Multi-dimensional calculus tells us that we need to compute also the associated Jacobian $J = |\partial \xi_n / \partial X_m|$. Fortunately, in the above discretization (also called *pre-point discretization*), it turns out that J is a constant that can, again, be 'absorbed' in the correct normalization. This is simply to see by looking at (34) and computing the corresponding partial derivatives:

$$\frac{\partial \xi_n}{\partial X_n} = \frac{1}{\sigma}, \qquad \frac{\partial \xi_n}{\partial X_m} = 0 \text{ for } m > n \tag{35}$$

The latter equation simply reflects that, due to the pre-point discretization, ξ_n does not depend on X_m if $m > n$. Therefore, we find simply $J = 1/\sigma^N$. Note that, for other discretizations, the Jacobian J is not necessarily that simple. If we choose to discretize the SDE in the *Stratonovich* interpretation, we need to evaluate f at the midpoint, hence compute $f((X_n + X_{n-1})/2)$ which will complicate things.
Usually, we use path integrals to compute transition probabilities. For example, consider a stochastic process X_t starting at the value $X_0 = a$ at time $t = 0$ and we are interested in the probability of $X_T = x$ at time T. Let's denote the corresponding probability density by $p(x, T|a, 0)$. Then, in the path integral formulation, we need to take all paths $\mathcal{C}([x, T|a, 0])$ into consideration and we can write

$$p(x, T|a, 0) = \int_{\mathcal{C}([x,T|a,0])} \exp\left(-\frac{1}{2\sigma^2}\int_0^T (\dot{X}_t - f(X_t))^2 dt\right) \mathcal{D}X_t \tag{36}$$

How do we compute path integrals? The honest answer is that, most of the time, we actually don't compute them - we approximate their value. In many situations arising in physics, we are interested in the limit of *small noise*. Physically this is often a very complicated situation: If noise is strong, it will dominate and 'wipe' out most deterministic structures. If noise is small, deterministic effects will govern most of the dynamics, but the influence of small noise is often highly nontrivial: For example, in metastable systems, noise can make the transition between stable fix points possible that, in the purely deterministic case, are entirely forbidden. While many mathematical tools in the realm of classic perturbation either fail or are difficult to apply, the path integral formulation is very suited to describe the behavior for $\sigma \to 0$: From the representation (36) it is clear that, in the limit of small noise, the path with the smallest value of the *action* S defined by

$$S = \frac{1}{2}\int_0^T (\dot{X}_t - f(X_t))^2 dt \tag{37}$$

will dominate the probability density and, if we are lucky, we might even obtain a decent approximation by *only* taking this path into account. If we call this minimum action path x_c (for 'classical' trajectory), we find as an approximation of the transition probability density $p(x,T|a,0)$

$$p(x,T|a,0) \approx \phi(t)\mathrm{e}^{-\frac{1}{2\sigma^2}\int_0^T L(x_{ct},\dot{x}_{ct})dt}, \qquad L(x,\dot{x}) = \frac{1}{2}(\dot{x}-f(x))^2 \tag{38}$$

where we also defined the *Lagrangian* L under the integral of the action functional and the function ϕ can be found by the normalization constraint that

$$\int p(x,T|a,0)dx = 1 \tag{39}$$

In the next section, we will show how to carry out these computations in a concrete case using the example of the Ornstein-Uhlenbeck process.

The Ornstein-Uhlenbeck Process

While the SDE $dX_t = dW_t,\ \ X_0 = 0$ describes Brownian motion, introducing a very simple (linear) drift of the for $f(X_t) = -kX_t,\ \ k > 0$ changes the stochastic dynamics tremendously. The SDE

$$dX_t = -kX_t\,dt + \sigma dW_t, \qquad X_0 = a \tag{40}$$

defines the *Ornstein-Uhlenbeck* process, a stochastic process that presents as one of the most prominent examples in the fields of stochastic differential equations. The associated Fokker-Planck equation is given by

$$\partial p = k(xp)_x + \frac{\sigma^2}{2}p_{xx}, \qquad p(t=0,x) = \delta(x-a) \tag{41}$$

and, while the exact solution of this equation using Fourier transform is possible, it is slightly tedious. However, we can see immediately that the Fokker-Planck equation allows for a solution that is stationary and a simple calculation shows that this stationary solution is Gaussian. Concerning the path integral, we see that the associated Lagrangian is

$$L(x,\dot{x}) = \frac{1}{2}(\dot{x}+kx)^2 \tag{42}$$

If we want to find the minimum action path, we need to solve the Euler-Lagrange equations subject to appropriate boundary conditions:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} = 0, \qquad \ddot{x} = k^2 x \tag{43}$$

Let us, for example compute the transition probability to go from $x = 0$ at $t = 0$ to the point x at time t. Parametrizing the classical trajectory x_c with s, we find that we need to solve for $x_c = x_c(s)$ the equation

$$\ddot{x}_c = k^2 x_c, \qquad x_c(0) = 0,\ x_c(t) = x\,. \tag{44}$$

After a few lines of calculation, we find that

$$x_c(s) = \frac{\mathrm{e}^{ks} - \mathrm{e}^{-ks}}{\mathrm{e}^{kt} - \mathrm{e}^{-kt}} \cdot x \tag{45}$$

and the semi-classical approximation leads to

$$p(x, t|0, 0) = \phi(t) \exp\left(\frac{-kx^2}{(1 - \mathrm{e}^{-2kt})\sigma^2}\right)\,, \qquad \phi(t) = \sqrt{\frac{k}{\pi\sigma^2(1 - \mathrm{e}^{-2kt})}} \tag{46}$$

It is also possible, to obtain the exact solution from the SDE directly: Starting again at

$$dX_t + kX_t dt = \sigma dW_t, \qquad X_0 = a$$

we see that we can use an integrating factor to write

$$d\left(\mathrm{e}^{kt} X_t\right) = \sigma \mathrm{e}^{kt}\, dW_t \tag{47}$$

with the obvious solution

$$X_t = a\mathrm{e}^{-kt} + \int_0^t \mathrm{e}^{-k(t-s)} dW_s\,. \tag{48}$$

From here we see that X_t is a superposition of Gaussian variables, hence Gaussian itself and we only need to find the mean and the variance of X_t in order to completely characterize its dynamics. The mean is trivial:

$$\mathbb{E}(X_t) = \mu_t = a\mathrm{e}^{-kt} \tag{49}$$

and, for the variance, we need to compute

$$V(X_t) = \mathbb{E}((X_t - \mu_t)(X_t - \mu_t))\,. \tag{50}$$

For this calculation, we can use the explicit solution given in (48). In addition we note that $\mathbb{E}(dW_s dW_{\tilde{s}}) = 0$ if $s \neq \tilde{s}$ and $\mathbb{E}(dW_s dW_{\tilde{s}}) = ds$ if $s = \tilde{s}$:

$$\begin{aligned} V(X_t) &= \mathbb{E}\left(\int_0^t \int_0^t \mathrm{e}^{-k(t-s)} \mathrm{e}^{-k(t-\tilde{s})} dW_s dW_{\tilde{s}}\right) \\ &= \int_0^t \mathrm{e}^{-2k(t-s)} ds = \frac{1 - \mathrm{e}^{-2kt}}{2k} \end{aligned}$$

which is exactly the same result as in the path integral calculation.

With a little more work, we can also work with the Fokker-Planck equation and use Fourier transform. Although this calculation seems to confirm only the previous results, it is nevertheless useful in order to learn how to apply Fourier transform in a slightly more complicated setting. The Fokker-Planck equation for the Ornstein-Uhlenbeck process reads

$$p_t = \partial_x(kxp) + \frac{\sigma^2}{2} p_{xx}, \qquad p(t=0,x) = \delta(x-a)\,. \tag{51}$$

Let's consider again the particular case of $a = 0$. In order to solve this Fokker-Planck equation in Fourier domain, we need to transform the term $\partial_x(xp)$. Integration by parts yields

$$\begin{aligned} \frac{1}{\sqrt{2\pi}} \int \mathrm{e}^{-i\omega x} \partial_x(xp) dx &= i\omega \frac{1}{\sqrt{2\pi}} \int \mathrm{e}^{-i\omega x} xp\, dx \\ &= i\omega \frac{1}{\sqrt{2\pi}} \frac{-1}{i} \partial_\omega \int \mathrm{e}^{-i\omega x} p\, dx = -\omega \partial_\omega \hat{p} \end{aligned}$$

Therefore, we obtain the following partial differential equation for $\hat{p}$

$$\hat{p}_t + k\omega \partial_\omega \hat{p} = -\frac{\sigma^2}{2} \omega^2 \hat{p}, \qquad \hat{p}(t=0,\omega) = \frac{1}{\sqrt{2\pi}}\,. \tag{52}$$

Since this is a first-order partial differential equation, we can use the *method of characteristics* to find the solution. The characteristic curves $\omega = \omega(t)$ are defined by considering

$$\frac{d}{dt}\hat{p}(t,\omega(t)) = \hat{p}_t + \dot{\omega}\partial_\omega p = \hat{p}_t + k\omega \partial_\omega \hat{p} \tag{53}$$

and setting $\dot{\omega} = k\omega$. The solutions are given by $\omega(t) = \mathrm{e}^{kt}\omega(0)$ or $\omega(0) = \mathrm{e}^{-kt}\omega(t)$. Along these curves, we have

$$\frac{d}{dt}\hat{p}(t,\omega(t)) = \frac{d\hat{p}}{dt} = -\frac{\sigma^2}{2}\omega(t)^2 \hat{p} = -\frac{\sigma^2}{2} \mathrm{e}^{2kt} \omega(0)^2 \hat{p} \tag{54}$$

This is a simple ordinary differential equation with the solution

$$\begin{aligned} \hat{p}(t,\omega(t)) &= \hat{p}(t=0,\omega) \exp\left(-\frac{\sigma^2}{2} \int_0^t \mathrm{e}^{2kt} \omega(0)^2 d\tilde{t}\right) \hat{p}(t=0,\omega) \\ &= \exp\left(-\frac{\sigma^2}{2} \omega(0)^2 \frac{\mathrm{e}^{2kt}-1}{2k}\right) \hat{p}(t=0,\omega) \\ &= \exp\left(-\frac{\sigma^2}{2} \omega(t)^2 \frac{1-\mathrm{e}^{-2kt}}{2k}\right) \hat{p}(t=0,\omega) \end{aligned}$$

Putting all solutions on the characteristic curves together, we have therefore

$$\hat{p}(t,\omega) = \exp\left(-\frac{\sigma^2}{2}\omega^2 \frac{1-\mathrm{e}^{-2kt}}{2k}\right)\hat{p}(t=0,\omega) \tag{55}$$

and the inverse Fourier transform yields the same result as the path integral and the solution of the SDE.

Monte-Carlo simulations revisited: Importance Sampling

In the following we are discussing an example of how the path integral approach can be useful to speed up the convergence of Monte-Carlo simulations. For simplicity, in this section, we will consider the case of an interest rate that is zero ($r = 0$). Then, using the risk-free measure $\mathbb{Q}$ and the $\mathbb{Q}$-Brownian motion W_t, the stock process is written as

$$S_t = S_0 \, \mathrm{e}^{\sigma W_t - \sigma^2 t/2} \, .$$

Consider now the problem of pricing a European option that is far out of the money, so for instance $K \ll S_0$ (with T and σ taking reasonable values). Clearly, the option price

$$V = \mathbb{E}_{\mathbb{Q}}((S_T - K)^+)$$

will be small, and we can compute its value exactly using the Black-Scholes formula. If we are trying to use Monte-Carlo simulations, let's say an ensemble of N paths, and to estimate V using the mean as described before, namely

$$\hat{V} = \frac{1}{N}\sum_i (S^{(i)} - K)^+$$

we encounter the problem that, for most of the sample paths, we will obtain $(S_T^{(i)} - K)^+ = 0$ and we expect to need a large number of paths in order to arrive at a decent approximation of V. What if we, in order to speed up the simulations, introduce *artificially* a drift γt via

$$W_t = \tilde{W}_t + \gamma t \tag{56}$$

in order to bias the noise in a way that makes the rare event $(S_T^{(i)} - K)^+ > 0$ more likely. Note, that we could also try to introduce a more complicated drift $\gamma(t)$ in $dW_t = d\tilde{W} + \gamma(t)dt$. Intuitively, we would like to consider sample paths that are close to the minimizer ϕ of the Lagrangian discussed in the previous section for our SDE under consideration, with the boundary conditions $\phi(0) = S_0$ and $\phi(T) = K$. How can we find this minimizer? The corresponding SDE for S_t is written as

$$dS_t = \sigma S_t dW_t \tag{57}$$

and we need to minimize the functional

$$\int_0^T \frac{1}{2\sigma^2} \frac{\dot{\phi}^2}{\phi^2} dt, \qquad \phi(0) = S_0, \qquad \phi(T) = K$$

It is easy to see that the solutions of the Euler-Lagrange equations are exponentials of the form

$$\phi(t) = S_0 \, \mathrm{e}^{\alpha t} \tag{58}$$

and the boundary conditions yield $\alpha = \ln(K/S_0)/(\sigma T)$. Therefore, a good first choice as an artificial drift γ is to set

$$\gamma = \alpha = \frac{1}{\sigma T} \ln \left(\frac{K}{S_0} \right) . \tag{59}$$

The corresponding new SDE is then simply

$$dS_t = \sigma S_t dW_t + \gamma S_t dt \, . \tag{60}$$

How do we now compensate for this drift when estimation the value V of the option? All we did was adding a drift, corresponding to a change of measure. Therefore, we can use the Cameron-Martin-Girsanov theorem and the likelihood ratio provided by the Radon-Nikodym derivative in the theorem in order to compensate for the drift:

$$\mathbb{E}_{\mathbb{Q}}(X) = \mathbb{E}_{\tilde{\mathbb{Q}}} \left(\frac{d\mathbb{Q}}{d\tilde{\mathbb{Q}}} X \right), \qquad \frac{d\mathbb{Q}}{d\tilde{\mathbb{Q}}} = \mathrm{e}^{-\gamma \tilde{W}_t - \gamma^2 t/2} \tag{61}$$

The following code implements this idea: While solving the SDE, we keep track of the corresponding realization of the Brownian motion for each path such that we can use this information at the end to use the Radon-Nikodym derivative as a factor when we compute the mean over all sample paths.

```
function [v,vAna] = euroOptionIS(N,gam)

% Set parameters

Sini = 42;
K = 80;
sig = 0.2;
T = 0.5;

nSteps = 1000;

dt = T/nSteps;

S = Sini*ones(N,1);
W = zeros(N,1); % keep track of Brownian motion

for k=1:nSteps
   r = randn(N,1);
   S = S + S*sig*sqrt(dt).*r+sig*gam*S*dt; % with IS drift
   W = W + sqrt(dt)*r;
end

% compute expectation with Girsanov factor
v = sum(exp(-0.5*gam^2*T)*exp(-gam*W).*max(S-K,0))/N;

d1 = (log(Sini/K)+sig^2/2*T)/(sig*sqrt(T));
d2 = (log(Sini/K)-sig^2/2*T)/(sig*sqrt(T));

vAna = Sini*normcdf(d1) - K*normcdf(d2);

end


function v = normcdf(x)
v=0.5*erfc(-x/sqrt(2));
end
```

Exercises

1. (10 points) Give a proof of the Box-Muller algorithm.
2. (10 points) Assume that, for a probability density p, the cumulative distribution function F defined by

$$F(x) = \int_{-\infty}^{x} p(\tilde{x})\, d\tilde{x}$$

is invertible and its inverse function is F^{-1}. Then, we can generate random numbers with a density p by doing the following:

   (a) Draw random number r, uniformly distributed in $[0, 1]$.
   (b) Compute $z = F^{-1}(r)$.

Prove that this statement is true. Then formulate an algorithm to generate random numbers with a Weibull distribution with the parameters $\lambda > 0$ and $k > 0$ given by

$$p(x) = \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \mathrm{e}^{-(x/\lambda)^k}$$

for $0 \leq x$.

3. (10 points) Prove the Chebyshev inequality: For $\epsilon > 0$ and $\mu = \mathbb{E}(X)$, we have

$$P(|X - \mu|) \geq \epsilon) \leq \frac{\mathrm{Var}(X)}{\epsilon^2}\,.$$

Use this equation to show that we expect Monte-Carlo simulations to converge with $\sim 1/\sqrt{N}$.

4. (10 points) Show that, for

$$P((\chi_1, ..., \chi_N)) = \frac{1}{(2\pi)^{N/2}} \frac{1}{\sqrt{|A|}} \exp\left(-\frac{1}{2}\sum_{n,m=1}^{N} \chi_n A^{-1}_{nm} \chi_m\right), \quad (62)$$

we have indeed $\mathbb{E}(\chi_n \chi_m) = A_{nm}$.

5. (10 points) Carry out the path integral calculations for the Ornstein-Uhlenbeck process.