Stochastic Differential Equation

Chiara Iannicelli

November 2023

1 Introduction

A stochastic differential equation (SDE) is a differential equation in which one or more of the terms is a stochastic process, resulting in a solution which is also a stochastic process. SDEs have many applications throughout pure mathematics and are used to model various behaviours of stochastic models such as stock prices, random growth models or physical systems that are subjected to thermal fluctuations.

SDEs have a random differential that is in the most basic case random white noise calculated as the derivative of a Brownian motion or more generally a semimartingale. However, other types of random behaviour are possible, such as jump processes like Lévy processes or semimartingales with jumps. Random differential equations are conjugate to stochastic differential equations. Stochastic differential equations can also be extended to differential manifolds.

2 Application

The notation used in probability theory (and in many applications of probability theory, for instance in signal processing with the filtering problem and in mathematical finance) is slightly different. It is also the notation used in publications on numerical methods for solving stochastic differential equations.

This notation makes the exotic nature of the random function of time η_m in the physics formulation more explicit. In strict mathematical terms, η_m cannot be chosen as an ordinary function, but only as a generalized function. The mathematical formulation treats this complication with less ambiguity than the physics formulation.

3 Formulas

A ordinary differential equation (ODE):

$$\frac{dx(t)}{dt} = f(t,x), \quad dx(t) = f(t,x)dt, \qquad (1)$$

with initial conditions $x(0) = x_0$ can be written in integral form

$$x(t) = x_0 + \int_0^t f(s, x(s))ds, \qquad (2)$$

where $x(t) = x(t, x_0, t_0)$ is the solution with initial conditions $x(t_0) = x_0$. An example is given as

$$\frac{dx(t)}{dt} = a(t)x(t), \quad x(0) = x_0. \qquad (3)$$

When we take the ODE (3) and assume that $a(t)$ is not a deterministic parameter but rather a stochastic parameter, we get a stochastic differential equation (SDE). The stochastic parameter $a(t)$ is given as

$$a(t) = f(t) + h(t)\xi(t), \qquad (4)$$

where $\xi(t)$ denotes a white noise process.
Thus, we obtain

$$\frac{dX(t)}{dt} = f(t)X(t) + h(t)X(t)\xi(t), \qquad (5)$$

When we write (5) in the differential form and use $dW(t) = \xi(t)dt$, where $dW(t)$ denotes differential form of the Brownian motion, we obtain:

$$dX(t) = f(t)X(t)dt + h(t)X(t)dW(t). \qquad (6)$$

In general an SDE is given as

$$dX(t.\omega) = f(t, X(t, \omega))dt + g(t, X(t, \omega))dW(t, \omega), \qquad (7)$$

where ω denotes that $X = X(t, \omega)$ is a random variable and possesses the initial condition $X(0, \omega) = X_0$ with probability one. As an example we have already encountered

$$dY(t, \omega) = \mu(t)dt + \sigma(t)dW(t, \omega)$$

Furthermore, $f(t, X(t, \omega) \in R$, $g(t, X(t, \omega)) \in R$, and $W(t, \omega) \in R$. Similar as in (2) we may write (7) as integral equation

$$X(t, \omega) = X_0 + \int_0^t f(s, X(s, \omega))ds + \int_0^t g(s, X(s, \omega))dW)s, \omega).$$

4 Simulation

Simulating a Stochastic Differential Equation (SDE) typically involves numerical methods like the Euler-Maruyama method. Let's simulate a simple example of an SDE using Python. Consider the geometric Brownian motion, which is commonly used to model stock prices. The SDE for geometric Brownian motion is given by:

$$dX_t = \mu X_t dt + \sigma X_t dW_t$$

where:

- X_t is the stock price at time t,
- μ is the drift coefficient,
- σ is the diffusion coefficient,
- dW_t is a Wiener process (Brownian motion),
- dt is the time step.

Follows a Python code using the Euler-Maruyama method to simulate the geometric Brownian motion:

```python
import numpy as np
import matplotlib.pyplot as plt

# Parameters
mu = 0.1        # Drift coefficient
sigma = 0.2     # Diffusion coefficient
initial_price = 50
dt = 0.01       # Time step
num_steps = 1000

# Function to simulate geometric Brownian motion using Euler-Maruyama method
def simulate_geometric_brownian_motion(mu, sigma, initial_price, dt, num_steps):
    t_values = np.arange(0, num_steps * dt, dt)
    X = np.zeros(num_steps)
    X[0] = initial_price

    for i in range(1, num_steps):
        dW = np.random.normal(loc=0, scale=np.sqrt(dt))
        dX = mu * X[i-1] * dt + sigma * X[i-1] * dW
        X[i] = X[i-1] + dX

    return t_values, X

```

```
# Simulate geometric Broumian motion
t_values, stock_prices = simulate_geometric_brownian_motion(mu,
↪  sigma, initial_price, dt, num_steps)

# Plot the results
plt.plot(t_values, stock_prices)
plt.title('Simulated Geometric Brownian Motion')
plt.xlabel('Time')
plt.ylabel('Stock Price')
plt.show()

```

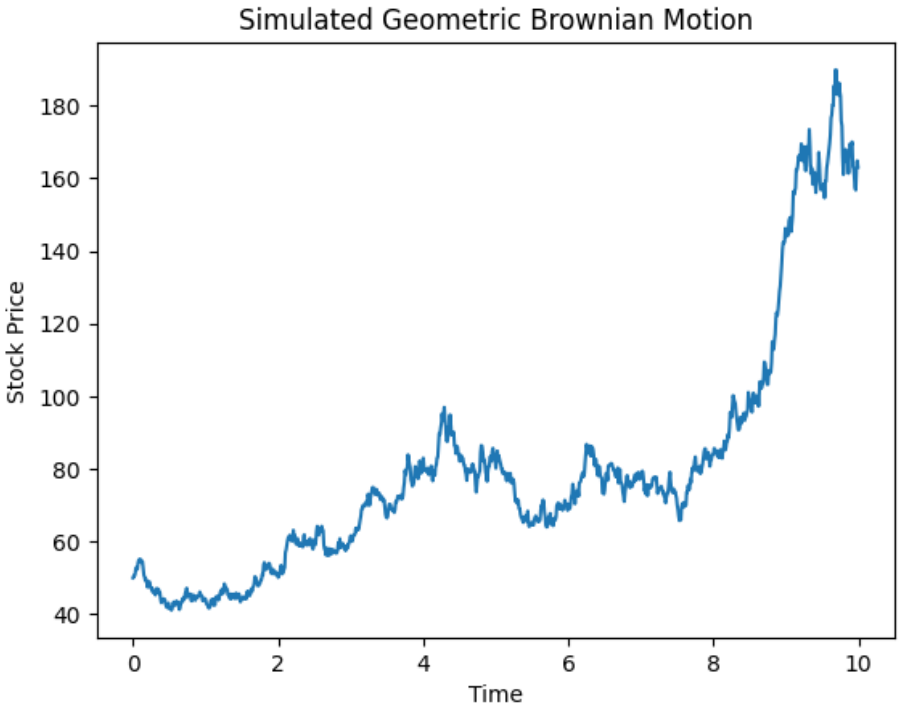

Figure 1: Brownian Motion simulation