Ornstein–Uhlenbeck Model Applied to VIX

Marc Aliaga

January 4, 2026

Project Overview: Modeling Mean-Reverting Processes in Finance

Video Explanation

1 Introduction

We consider the following stochastic differential equation (SDE):

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

2 Definitions

Where:

- X_t is the value of the stochastic process at time t
- μ is the long-term mean
- θ is the speed of mean reversion
- σ is the volatility
- dW_t is an increment of a Wiener process

This SDE defines an **Ornstein–Uhlenbeck process**, commonly used to model mean-reverting phenomena.

3 Objective

The objective is to derive and calibrate the explicit solution of the Ornstein–Uhlenbeck process in order to project future values of a mean-reverting financial variable.

4 Application to Finance

This model is particularly suited for instruments such as the **VIX**, which exhibits strong mean reversion.

- μ represents the historical average VIX level

- θ controls the speed of reversion after shocks
- σ measures uncertainty in volatility movements

Applications include:

- Risk management
- Derivatives pricing
- Volatility trading strategies

5 Data Collection

The following code retrieves historical VIX data from Yahoo Finance using the `yfinance` library. The sample starts in 2012 and extends to the most recent available trading day. Only closing prices are retained for the analysis.

```
import yfinance as yf
import datetime

today = datetime.date.today()
yesterday = today - datetime.timedelta(days=1)
start_date = "2012-01-01"

vix_data_full = yf.download(
    "^VIX",
    start=start_date,
    end=yesterday + datetime.timedelta(days=1),
    progress=False
)

vix_data = vix_data_full["Close"]

print(vix_data.tail(3))
```

6 Estimation of the Long-Term Mean

The long-term mean parameter μ is estimated using the empirical average of the historical observations.
We compute the sample mean of the VIX time series and visualize it alongside the observed data to verify the mean-reverting behavior.

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

x_values = vix_data.values
mu = np.mean(x_values)

plt.figure(figsize=(10,6))
plt.plot(vix_data.index, x_values)
plt.axhline(mu, color="red")
```

```
plt.grid()
plt.show()
```

7 Maximum Likelihood Estimation

To estimate the remaining parameters, we adopt a Maximum Likelihood Strategy (MLS). The log-likelihood of the discretized Ornstein–Uhlenbeck process is given by:

$$\log L(\theta, \mu, \sigma) = \sum_{i=1}^{n} \log f(X_i \mid m_i, s_i^2) \tag{2}$$

7.1 Negative Log-Likelihood Function

The following function computes the negative log-likelihood by evaluating the exact transition density of the Ornstein–Uhlenbeck process. For each observation, the conditional mean and variance are calculated analytically.

```
from scipy.stats import norm

def neg_log_likelihood(params, x, dt):
    theta, sigma = params
    ll = 0.0
    for i in range(1, len(x)):
        m = mu + (x[i-1] - mu) * np.exp(-theta * dt)
        s2 = (sigma**2)/(2*theta)*(1 - np.exp(-2*theta*dt))
        ll += -norm.logpdf(x[i], loc=m, scale=np.sqrt(s2))
    return ll
```

7.2 MLE Optimization

The parameters θ and σ are obtained by minimizing the negative log-likelihood function. Positivity constraints are imposed to ensure numerical stability and theoretical validity.

```
from scipy.optimize import minimize

def mle_estimation(x):
    dt = 1
    init_params = [0.5, np.std(x)]
    res = minimize(
        neg_log_likelihood,
        init_params,
        args=(x, dt),
        bounds=[(1e-5, None), (1e-5, None)]
    )
    return res.x
```

8 Monte Carlo Simulation

After estimating the model parameters, we generate multiple future trajectories of the VIX using the exact solution of the Ornstein–Uhlenbeck process. All paths share the same parameters but

differ due to random innovations.

```
def simulate_ou_paths(X0, mu, theta, sigma, dt, T, n_paths):
    n_steps = int(T / dt)
    paths = np.zeros((n_steps + 1, n_paths))
    paths[0] = X0

    for t in range(1, n_steps + 1):
        mean = paths[t-1]*np.exp(-theta*dt) + mu*(1-np.exp(-theta*dt))
        var = (sigma**2)/(2*theta)*(1-np.exp(-2*theta*dt))
        paths[t] = np.random.normal(mean, np.sqrt(var), n_paths)

    return paths
```

9 Visualization

The simulated paths are plotted together with their cross-sectional mean and the realized future VIX values. This visualization illustrates the predictive behavior of the model and highlights the mean-reversion mechanism.

```
plt.figure(figsize=(10,6))
plt.plot(paths, color="gray", alpha=0.7)
plt.plot(np.mean(paths, axis=1), color="gold", lw=2)
plt.axhline(X0, color="red", linestyle="--")
plt.plot(future_x, color="blue", lw=3)
plt.grid()
plt.show()
```

Conclusion

The Ornstein–Uhlenbeck process provides a mathematically robust and empirically effective framework for modeling mean-reverting financial instruments such as the VIX. Through maximum likelihood estimation and Monte Carlo simulation, the model offers insight into the dynamics and future behavior of market volatility.

Video: Finance and SDE Connection