Monte Carlo simulations using artificial neural networks

Kudryavtsev Oleg

Russian Customs Academy, Rostov branch
Southern Federal University, koe@sfedu.ru

ICSM-10

Divnomorskoye, 04 June 2025

Outline

Outline

Pricing options under Lévy processes

Option valuation under Lévy processes has been dealt with by a host of researchers.
However, pricing path-dependent options in exponential Lévy models still remains a computational challenge.

New trends

There is an active interest in applying machine learning methods for dealing with problems in applied mathematics. A straightforward approach aimed at applying supervised learning algorithms based on market data faces certain criticism, since a serious theoretical justification for the adequacy of such models is usually not provided. However, ML methods can replace some routine elements of numerical methods.

Historical background

Neural networks in computational finance

A special class of machine learning methods known as ANN achieve notable results in almost any field of quantitative finance, including option pricing and model calibration.
In the state-of-the-art machine learning approaches to option pricing, artificial neural networks are used as function approximators

Hybrid numerical methods

Hybrid numerical methods that include elements of “traditional” methods of numerical mathematics and elements of ML is the most relevant direction in the development of computational finance.

The main goal

The purpose of this talk is to give probabilistic analogs of the universal approximation theorems and suggest the possibilities to developing hybrid Monte Carlo methods

Outline

Lévy processes: a short reminder

General definitions

A Lévy process is a stochastically continuous process with stationary independent increments (for general definitions, see e.g. Sato (1999)). A Lévy process can be completely specified by its characteristic exponent, ψ, definable from the equality $E[e^{i\xi X(t)}] = e^{-t\psi(\xi)}$.

Lévy-Khintchine formula

The characteristic exponent of the Lévy process is determined by the Lévy-Khintchine formula:

$$\psi(\xi) = \frac{\sigma^2}{2}\xi^2 - i\mu\xi + \int_{-\infty}^{+\infty} (1 - e^{i\xi x} + i\xi x 1_{|x|\leq 1}) F(dx),$$

where σ^2 is the variance of the Gaussian component, and the Lévy measure $F(dx)$ satisfies the condition $\int_{\mathbb{R}\setminus\{0\}} \min\{1, x^2\} F(dx) < +\infty$.

Examples of Lévy processes, $F(\mathrm{R}) < \infty$

Jump diffusion

$X_t = \gamma_0 t + \sigma W_t + \sum_{i=1}^{N_t} Y_i$, where W_t – Brownian motion, N_t – Poisson process with intensity λ, and Y_i – i.i.d of jumps.

Kou model

In the Kou model, the jumps Y_i in the process X_t have a double exponential distribution with the pdf:

$$p_{Y_i}(x) = (1-p)\Lambda_- e^{\Lambda_- x} 1_{\{x<0\}} + p\Lambda_+ e^{-\Lambda_+ x} 1_{\{x>0\}}.$$

where $\Lambda_- > 0$ – the intensity of negative jumps,
$\Lambda_+ > 1$ – the intensity of negative jumps,
$0 < p < 1$ – the probability of positive jumps.

Examples of Lévy processes, $F(\mathbb{R}) = \infty$

Tempered stable Lévy processes (TSL)

$$\psi(\xi) = -i\mu\xi + c_+\Gamma(-\nu_+)[\lambda_+^{\nu_+} - (\lambda_+ + i\xi)^{\nu_+}] + c_-\Gamma(-\nu_-)[(-\lambda_-)^{\nu_-} - (-\lambda_- - i\xi)^{\nu_-}],$$

where $\nu_+, \nu_- \in (1,2)$, $c_+, c_- > 0$, $\mu \in \mathbb{R}$, and $\lambda_- < -1 < 0 < \lambda_+$. If $c_- = c_+ = c$ and $\nu_- = \nu_+ = \nu$, then we obtain a KoBoL (CGMY) model.

$$\pi(x) = c_+ e^{\lambda_+ x}|x|^{-\nu_+ - 1}1_{\{x<0\}} + c_- e^{\lambda_- x}|x|^{-\nu_- - 1}1_{\{x>0\}}.$$

In the CGMY parametrization: $C = c$, $Y = \nu$, $G = \lambda_+$, $M = -\lambda_-$.

Computing CDF

One can express the cumulative distribution function F_X in terms of the Fourier integral

$$F_X(x) = \frac{e^{x\rho}}{\pi} \mathrm{Re} \int_0^\infty e^{-ix\xi} \frac{E[e^{i(\xi+i\rho)X}]}{\rho - i\xi} d\xi, x \in \mathrm{R}$$

Simulation: $X = F_X^{-1}(U)$

For an arbitrary $u \in (0,1)$, one can recover $F_X^{-1}(u)$ using for instance linear interpolation formulas:

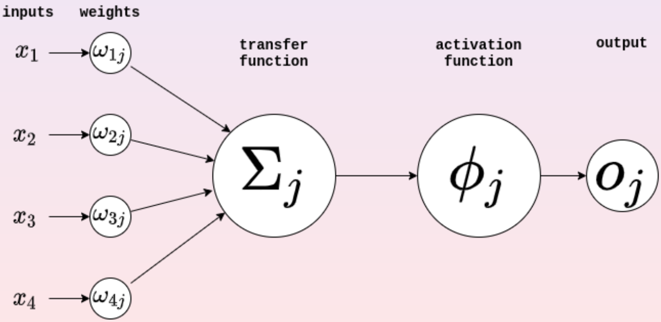
$$F_X^{-1}(u) = \begin{cases} x_0, & u < F_X(x_0), \\ x_k + \frac{(u - F_X(x_k))(x_{k+1} - x_k)}{F_X(x_{k+1}) - F_X(x_k)}, & F_X(x_k) \leq u < F_X(x_{k+1}), k < M, \\ x_M, & u \geq F_X(x_M). \end{cases}$$

Quadratic or cubic splines can also be used for the approximation

Outline

ANN

- ANN is a graph consisting of a large number of nodes (neurons) connected to each other through weighted connections (synapses)
- The weight of each neuron is adjusted as the network learns
- Weights are responsible for increasing or the influence of signals as they pass through neurons



Universal approximation theorems

Functions can be modeled with ANN due to the Cybenko's theorem. According to this theorem, a feedforward ANN with one hidden layer and the same-type sigmoidal activation functions can approximate any continuous function of multiple variables with any accuracy. This theorem belongs to the class of universal approximation theorems that establish the approximation capabilities of different neural networks.

Theorem

Let $s(x)$ *– be an arbitrary continuous sigmoidal function, and real numbers* a, b *are such that* $a < b$. *For a given* $\epsilon > 0$ *and a given* $F(x) \in C[a, b]$ *there is a sum of the form*

$$G(x) = \sum_{j=1}^{N} \omega_j s(c_j x + d_j), \quad \omega_j, c_j, d_j \in \mathrm{R},$$

such that $|G(x) - F(x)| < \epsilon$, *for all* $x \in [a, b]$.

Theorem

Let X be an arbitrary c.r.v. distributed on $[a, b]$, $a < b$. For a given $\epsilon > 0$ and a given c.r.v. Y there is a r.v. of the form

$$Z = \begin{cases} \alpha_1 Y_1 + \beta_1, & \text{with probability } p_1; \\ \dots \\ \alpha_j Y_j + \beta_j, & \text{with probability } p_j; \\ \dots \\ \alpha_N Y_N + \beta_N, & \text{with probability } p_N, \end{cases}$$

where $Y_j \overset{d}{\sim} Y$ are independent, $p_j > 0$, $\alpha_j > 0$, $\beta_j \in \mathrm{R}$, such that

$$\sum_{j=1}^{N} p_j = 1; \; |F_X(x) - F_Z(x)| < \epsilon, \text{ for all } x \in \mathrm{R},$$

$F_X(x)$ and $F_Z(x)$ are CDFs of X and Z, respectively.

Theorem

Let X be an arbitrary c.r.v. For a given $\epsilon > 0$ and a given c.r.v. Y there is a r.v. of the form

$$Z = \begin{cases} \alpha_1 Y_1 + \beta_1, & \text{with probability } p_1; \\ \dots \\ \alpha_j Y_j + \beta_j, & \text{with probability } p_j; \\ \dots \\ \alpha_N Y_N + \beta_N, & \text{with probability } p_N, \end{cases}$$

where $Y_j \overset{d}{\sim} Y$ are independent, $p_j > 0$, $\alpha_j > 0$, $\beta_j \in \mathrm{R}$, such that

$$\sum_{j=1}^{N} p_j = 1; \; |F_X(x) - F_Z(x)| < \epsilon, \text{ for all } x \in \mathrm{R},$$

$F_X(x)$ and $F_Z(x)$ are CDFs of X and Z, respectively.

Outline

Logistic regression

The standard logistic function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

is arguably the most well-known activation function for ANNs.

Another advantage of $\sigma(x)$ is that it is the cdf of the continuous random variable Y known as a standard logistic distribution. Hence, Y can be used successfully in our Theorems.

Since $\sigma^{-1}(u) = \ln \frac{u}{1-u}$, one can easily simulate Y from $\sigma^{-1}(U)$, where the random variable U is uniformly distributed in $(0, 1)$.

The key ideas

Let X_t be a Lévy process. We construct a monotonic feedforward ANN with N neurons in the single hidden layer and with the activation function $\sigma(x)$ that approximates $F_{X_T}(x)$ with a given accuracy ϵ.

From the architecture of the ANN we obtain a sum of the form

$$G(x) = \sum_{j=1}^{N} \omega_j \sigma(c_j x + d_j), \quad \omega_j > 0, c_j > 0, d_j \in \mathrm{R},$$

such that $|F_{X_T}(x) - G(x)| < \epsilon,$ for all $x \in \mathrm{R}$.

Consider the option price $V(x, T) = e^{-rT} E[H(x + X_T)]$, where $H(x)$ is a given payoff function, X_t – a Lévy process.

The key ideas

Let Y be a standard logistic distribution whose cdf is $\sigma(x)$. Then we have

$$\sigma(c_j x + d_j) = P(Y < c_j x + d_j) = P(\frac{1}{c_j}Y - \frac{d_j}{c_j} < x) = F_{\frac{1}{c_j}Y - \frac{d_j}{c_j}}(x)$$

Set $p_j = \omega_j$, $\alpha_j = 1/c_j$ and $\beta_j = -d_j/c_j$.

Define a sufficiently large number of simulations L. We approximate

$$V(x, T) = e^{-rT}\frac{1}{L}\sum_i H(x + z_i),$$

where z_i, $i = 1, ..., L$, are sample values from $\alpha_j Y + \beta_j$ with probability p_j, $j = 1, ..., N$. Thus, we do not need to run the entire ANN we constructed, but rather use its separate neurons to simulate $\alpha_j Y + \beta_j$.

The hybrid Monte Carlo algorithm

Algorithm 1 Evaluate $V(x, T) = e^{-rT} E[H(x + X_T)]$

Set $j \Leftarrow 1$, $i \Leftarrow 1$, $m \Leftarrow 0$, $V \Leftarrow 0$
while $j \leq N$ **do**
    $m \Leftarrow m + p_j \cdot L$
    while $i \leq m$ **do**
        Simulate y_i from $\sigma^{-1}(U)$
        $z_i \Leftarrow \alpha_j y_i + \beta_j$
        $i \Leftarrow i + 1$
        $V \Leftarrow V + H(x + z_i)$
    end while
    $j \Leftarrow j + 1$
end while
$V(x, T) = e^{-rT} \frac{V}{L}$

Approximation of $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-z^2/2} dz$

Build an ANN to approximate the values of $\Phi(x)$.

$$F(z) = \sum_{j=1}^{10} \omega_j s(c_j z + d_j), z \in [-3, 3], \omega_j > 0, c_j > 0, d_j \in \mathrm{R},$$

where $\sigma(x) = e^x/(1 + e^x)$.

The sum of the positive weights ω_j on the output layer is approximately 1, which allows them to be interpreted as the probabilities p_j associated with each component $\alpha_j Z_j + \beta_j$ in the mixture distribution Z, where $\alpha_j = 1/c_j$, $\beta_j = -d_j/c_j$, and Z_j is a standard logistic distribution with the CDF $\sigma(x)$.
Thus, the mixture constructed using our neural network approximates the standard normal distribution X

Approximation of $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-z^2/2} dz$

Table: ANN parameters for approximation $\Phi(x)$

c_j	d_j	ω_j
2.769603	4.218520	0.088979
3.526779	-1.294590	0.104438
3.362026	1.559401	0.096860
3.717573	3.697874	0.091598
3.957426	-1.881185	0.105242
4.287301	-0.057452	0.101482
3.302728	1.473509	0.097048
3.851278	-3.625712	0.107510
3.115624	1.080214	0.098064
2.745375	-4.008735	0.108418

Approximation of $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-z^2/2} dz$

The results of numerical experiments

- The mean square error on the test sample was 1.11×10^{-8}.
- For z in the range of $[-3.3]$, the statistical estimate of

$$P(|F(z) - \Phi(z)| < \epsilon), \epsilon = 0.000564$$

  is higher than 0.9993 with a confidence level of 99.9%.
- Testing on a grid with a step of 0.001 on the segment $[0, 3]$ showed that the ANN works faster on average than the standard function **norm.cdf()** from the SciPy library: 0.000036 sec versus 0.000070 sec

Conclusion

Hybrid Monte Carlo methods for pricing options under Lévy processes can be found in the following papers

- O. Kudryavtsev, N. Danilova, "Applications of Artificial Neural Networks to Simulating Lévy Processes" *Journal of Mathematical Sciences*, 2023, Vol. 271.
- O. E. Kudryavtsev, A. S. Grechko, and I. E. Mamedov, "Monte Carlo Method for Pricing Lookback Type Options in Lévy Models" // *Theory of Probability & Its Applications*, 2024, Vol. 69., Iss.2.

A hybrid Wiener-Hopf factorization method for pricing options under Lévy processes can be found in the following paper

- E. Alymova, O. Kudryavtsev, "Artificial Neural Networks and Wiener-Hopf Factorization" *IAENG International Journal of Computer Science*, 2024, Vol. 51, no. 8.