Journal of Mathematics and Modeling in Finance (JMMF)
Vol. 3, No. 1, Winter & Spring 2023
Research paper

Deep learning for option pricing under Heston and Bates models

Ali Bolfake[1], **Seyed Nourollah Mousavi**[2], **Sima Mashayekhi**[3]

[1] Department of Mathematics, Faculty of Sciences, Arak University, Arak, Iran
ali.bolfake@gmail.com

[2] Department of Mathematics, Faculty of Sciences, Arak University, Arak, Iran
n-mousavi@araku.ac.ir

[3] Department of Mathematics, Faculty of Sciences, Arak University, Arak, Iran
s-mashayekhi@araku.ac.ir

Abstract:
This paper proposes a new approach to pricing European options using deep learning techniques under the Heston and Bates models of random fluctuations. The deep learning network is trained with eight input hyper-parameters and three hidden layers, and evaluated using mean squared error, correlation coefficient, coefficient of determination, and computation time. The generation of data was accomplished through the use of Monte Carlo simulation, employing variance reduction techniques. The results demonstrate that deep learning is an accurate and efficient tool for option pricing, particularly under challenging pricing models like Heston and Bates, which lack a closed-form solution. These findings highlight the potential of deep learning as a valuable tool for option pricing in financial markets.

Keywords: Option pricing, Heston model, Bates model, Deep learning, Monte Carlo simulation, Variance reduction technique.
JEL Classifications: 91G20, 91G60.

1 Introduction

The Black-Scholes model, introduced in 1973 by Fischer Black and Myron Scholes [3], has become a widely adopted standard method for pricing options in the financial industry. However, despite the success of the Black-Scholes model, some of its assumptions do not accurately reflect the realities of financial markets [4,22]. For example, empirical analyses of key market indices such as the S&P500 or Nasdaq Composite demonstrate that prices do not conform to the Gaussian distribution assumed by the Black-Scholes model [5, 25]. Furthermore, the assumption of a constant parameter for implied volatility in the Black-Scholes model is not supported by empirical evidence [9].

[2]Corresponding author
Received: 05/04/2023 Accepted: 24/07/2023
https://doi.org/10.22054/JMMF.2023.73263.1085

These findings highlight the need for alternative option pricing models that can better account for the complexities and dynamics of financial markets [2]. In particular, the discontinuous behavior of stock prices is not sufficiently captured by the Brownian motion assumed in the Black-Scholes model. To overcome this issue, several models have been developed, including the stochastic volatility model of Heston [13], and its generalization for incorporating jumps in stock prices, known as the Bates model [2].

The Heston model is a popular choice for modeling the dynamics of stock prices with stochastic volatility. In this model, the volatility of the stock price is assumed to follow a separate stochastic process from the stock price itself, which allows for greater flexibility in capturing the volatility smile observed in the market [13]. The Bates model, on the other hand, incorporates jumps in stock prices as well as stochastic volatility. This model has been shown to provide a better fit to market data than the Black-Scholes model, especially for options with short maturities [2].

The Heston model and the Bates model are both examples of stochastic volatility models, which have become increasingly popular in recent years due to their ability to capture the volatility smile observed in the market.

Machine learning techniques have been increasingly explored for financial engineering and option pricing in recent years due to their potential to capture complex patterns in market data and adapt to changing market conditions. These models have shown promise in improving the accuracy of option pricing and have the potential to outperform traditional models such as the Black-Scholes model [7, 15, 16, 24].

One popular approach in machine learning for option pricing is the use of neural networks. Neural networks can be used to approximate the payoff function of an option based on the underlying asset price and other relevant features [7, 26]. In addition, deep learning techniques, such as convolutional neural networks and recurrent neural networks, have also been applied to option pricing [7, 18].

Another approach in machine learning for option pricing is the use of reinforcement learning. Reinforcement learning is a type of machine learning that involves training an agent to make decisions based on the rewards or penalties it receives for its actions. In option pricing, reinforcement learning can be used to learn an optimal trading strategy based on market data and other relevant features [17].

Despite the potential benefits of machine learning techniques for option pricing, there are also challenges associated with their use. These models can be computationally intensive and require large amounts of training data. In addition, the interpretability of these models can be a concern, as it can be difficult to understand how they arrive at their predictions.

In this article, we address the classic problem of option pricing with two models of random fluctuations, Heston and Bates, using deep learning. For this purpose, we design an architecture for deep learning. Due to the need for large amounts of data for learning models, we will use Monte Carlo methods along with variance reduction techniques to generate data optimally. We will demonstrate that using variance reduction techniques leads to generating data with better accuracy for the learning model and, as a result, better accuracy in option pricing. By reviewing and comparing the results, we will show that deep learning is a powerful and reliable competitor to existing methods.

The structure of this article is as follows: In section 2, the pricing of European options and the Heston and Bates fluctuation models, along with their characteristic function to determine the actual value of the option price, are summarized in section 2. In section 3, the concepts of Monte Carlo methods, Monte Carlo methods with antithetic variate, and Monte Carlo methods with control variate are explained. In section 4, some concepts and computational methods of machine learning and artificial neural networks are briefly reviewed, and a brief explanation of the Tensorflow software library for deep learning is provided. In section 5, the computational concepts and methods presented in sections 2, 3, and 4 are used for a deep learning model to predict the pricing of European options under random fluctuation models, and the results are reported in general. Finally, in section 6, we will interpret the main findings and draw conclusions.

2 Pricing of Options and Random Fluctuation Models

Financial derivatives are financial instruments traded in financial markets. As the name suggests, derivative instruments are derived from other underlying financial instruments, and the cash flows of a derivative depend on the prices of the underlying instruments [14].

Given the risks in financial markets, there is a great demand for predicting the future behavior of securities. Derivative instruments respond to this need and contain information for estimating the behavior of a security in the future. There are three general categories of derivatives, namely forward contracts, futures contracts, and options. Options are derivatives that can be applied to any underlying asset, including other derivative instruments. An investor may be more interested in the profit that can be obtained by entering into an option contract than actually owning the asset on which the option is based, as is the case with futures or forwards contracts. Options are divided into two categories, independent of path and dependent on path. The most common path-independent options are European options, and these options come in two types: call options and put options, which we use call options in this article.

Consider an underlying asset S. The price of a European call option on $S(t)$ is denoted by $C(t)$ and gives the holder the right to buy the underlying asset at a strike price K at a future time $T > t$. At $t = T$, when the option expires, the value of the option $C(t)$ is clearly defined by [14]:

$$C(T) = \max\{S(T) - K, 0\}. \tag{1}$$

We consider two models of random fluctuations: the Heston model and the Black-

Scholes model. The next section provides a general overview of each model along with an analytical formula for the characteristic function that is efficient for pricing options.

2.1 The Heston stochastic volatility model

The Heston model [5], introduced in 1993, is a stochastic volatility model in which the dynamics of the stock price's risk-neutral volatility are governed by

$$\begin{aligned} dS_t &= (r-q)S_t dt + \sqrt{\nu_t} S_t dW_t^{(1)}, \\ d\nu_t &= k(\theta - \nu_t)dt + \xi\sqrt{\nu_t} dW_t^{(2)}, \\ \operatorname{Cov}(dW_t^{(1)}, dW_t^{(2)}) &= \rho dt \end{aligned} \tag{2}$$

where r, q, θ, k, ξ, and ρ are constants, $W_t^{(1)}$ and $W_t^{(2)}$ are standard Brownian motions with correlation ρ, and ν_t denotes the instantaneous variance of the stock price, which follows a mean-reverting square-root process. The covariance between $W_t^{(1)}$ and $W_t^{(2)}$ is given by ρdt. The Heston model is widely used in quantitative finance for modeling and pricing derivatives on assets with stochastic volatility.

The Black-Scholes model assumes constant volatility, whereas in the Heston model, volatility follows a stochastic process known as the Cox-Ingersoll-Ross (CIR) process. This process, which reverts to a long-term mean, is controlled by the parameter k that determines the rate of reversion to the mean value η. High values of k make the process deterministic, quickly smoothing out any deviation from η. By including stochastic volatility, the Heston model can capture the observed empirical skewness in implied volatilities, making it a flexible and robust model. The parameter ρ represents the correlation between the level and volatility, while θ represents the volatility of volatility. Skewness of the return distribution is controlled by ρ, while θ affects kurtosis. The CIR process was first successfully used in interest rate modeling.

The characteristic function of the logarithmic price process in the Heston model, of which a derivative can be found in [9], is presented as follows:

$$\phi(u,t) = \mathbb{E}\left[e^{iu\log(S_t)} \middle| S_0, \sigma_0^2 \right] = e^{A(u,t)+B(u,t)+C(u,t)}, \tag{3}$$

where

$$
\begin{aligned}
A(u,t) &= iu\log(S_0) + iu(r-q)t, \\
B(u,t) &= \frac{\eta k}{\theta^2}(k-\rho\theta iu-d)t - 2\log\left(\frac{1-ge^{-dt}}{1-g}\right), \\
C(u,t) &= \frac{\sigma_0}{\theta^2}\frac{(k-\rho\theta iu-d)(1-e^{-dt})}{1-ge^{-dt}}, \\
d(u) &= \sqrt{(\rho\theta iu-k)^2+\theta^2\left(ui+u^2\right)}, \\
g(u) &= \frac{k-\rho\theta iu-d}{k-\rho\theta iu+d}.
\end{aligned}
$$

The Heston model incorporates stochastic volatility through the Cox-Ingersoll-Ross (CIR) process, which reverts to a long-term mean and is controlled by the parameter k that determines the rate of reversion to the mean value η. The correlation between the level and volatility is represented by the parameter ρ, while θ represents the volatility of volatility. The Heston model is able to capture the observed empirical skewness in implied volatilities, making it a flexible and robust model [9].

2.2 Monte Carlo Methods

Monte Carlo (MC) methods are commonly used to estimate the expected value of a random variable X from independent and identically distributed samples $X_1, \ldots, X_n$, where $\mu = E(X)$. The sample mean, $\hat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$, is used as an estimator for μ, and the law of large numbers guarantees that this estimator converges to the true value of μ as the sample size increases [10].

In more complex scenarios, X may be a function of other random variables, and the law of large numbers is replaced by the Kolmogorov's law of large numbers which states that the expected value of a function $f(X)$ can be estimated as the sample mean of $f(X_i)$ as n approaches infinity.

MC methods are commonly used in financial engineering for pricing derivatives, such as European options, under uncertain market conditions. By simulating the dynamics of asset prices based on a model's parameters and initial conditions, we can easily simulate the entire path of the stock prices up to the expiration date T. For European options, the value of the option at the expiration date is determined by the realized value of the underlying asset at that time, which can be calculated using the simulated paths and the payoff function of the option. The option price is then estimated as the discounted average of the payoffs across all simulated paths [10].

2.3 Variance Reduction Techniques

In the field of Monte Carlo simulation, reducing the variance of a random variable is a key factor in improving the efficiency and accuracy of the simulation. Suppose that $E[(X)^2] < \infty$, where X is a random variable, and the standard deviation of X is defined as $\sigma = \sqrt{E[(f(X) - E[f(X)])^2]}$. The Monte Carlo error can be represented by $\epsilon_n = E[f(X)] - \frac{1}{n}\sum_{i=1}^{n} f(X_i)$, where X_i is a sample of size n. It is shown that the mean square error of $E[\epsilon_n(f, X)] = \frac{\sigma}{\sqrt{n}}$ can be achieved. Thus, the error in Monte Carlo simulation can be reduced to $\sigma/\sqrt{n}$. However, it should be noted that controlling the error only provides information about the expected error, and does not provide any information about the error in the actual simulation paths. The error is proportional to the standard deviation of the random variable $f(X)$, so the efficiency of Monte Carlo simulations can be improved by reducing $\mathrm{Var}[f(X)]$ using variance reduction techniques.

Two main strategies for reducing variance are utilizing model features to adjust simulation outputs and reducing input fluctuations. In particular, we discuss Monte Carlo methods with antithetic variate and control variate, among other available techniques. In this paper, we utilize variance reduction techniques mentioned in references [21] for simulating random oscillation models using Monte Carlo methods.

3 Deep Learning

Deep learning is a powerful branch of machine learning that has gained significant success in various fields of artificial intelligence and machine learning. It models high-level abstract concepts by learning at different levels, which involves more than two layers of a model. To achieve this, deep learning uses artificial neural networks that have many hidden layers [11]. TensorFlow, a powerful open-source software library, is used to train neural networks with more than two layers. TensorFlow represents all calculations and samples in a machine learning algorithm, including mathematical operations, parameters, and update rules at a large scale [**?**].

Supervised learning is another approach to machine learning used to train a model. In this approach, the machine is trained with labeled training data, allowing it to make good decisions about previously unseen labeled data it may encounter in the future. This approach uses concepts such as loss function, cost function, gradient descent algorithm, backpropagation algorithm, activation functions (ReLU, SELU, and Softplus), and Adam algorithm to train the model [19, 23].

Traditional pricing models are based on an underlying process that reproduces the empirical relationship between observable option data and underlying stock data, while machine learning methods do not assume any underlying process. Regression, a technique for predicting the output variable based on the input variables, is used to interpret the effect of inputs on the output. Machine learning algorithms build a mathematical model based on sample data, known as ”training data”, which is then used to make predictions or decisions without being explicitly programmed to perform the task [8, 20].

Developing an efficient deep learning model involves critical elements such as the selection of layers and neurons, the utilization of activation functions, and the implementation of network optimization algorithms. Researchers conduct experiments to fine-tune these elements and determine the optimal configuration. The quantity of layers plays a pivotal role in capturing patterns while preventing overfitting. Striking the right balance is challenging because a scarcity of layers might disregard intricate patterns, while an excess of layers can lead to overfitting. Similarly, the number of neurons determines the model's capacity and expressiveness. Researchers experiment with various neuron arrangements to find a compromise between model complexity and generalization capability. Activation functions introduce nonlinearities, allowing the model to represent intricate relationships. Researchers explore different activation functions such as sigmoid, tanh, and ReLU to discover the most appropriate choices for each layer. Furthermore, network optimization algorithms like SGD, Adam, or RMSprop have a significant impact on model training. Researchers experiment with various optimization algorithms to achieve the best convergence and generalization performance.

Deep learning has been widely applied in the financial industry to model complex data patterns and extract useful information for investment and risk management [12]. It has shown remarkable results in areas such as stock price prediction, portfolio optimization, credit risk assessment, fraud detection, and algorithmic trading.

4 Numerical results in the application of deep learning in option pricing

In this section, we present a model for determining the price of European options under stochastic volatility models using Monte Carlo methods explained in Subsection 2.2 and high-precision deep learning networks. This trained model demonstrates that deep learning can be used to learn pricing models for European call options in various markets.

We use deep learning for pricing European options using simulated data to train the model. We generate eight sets of data using four methods, including the exact and explicit European formula under specified characteristic functions of stochastic volatility models explained in Subsection 2.1, Monte Carlo simulation for European option pricing, Monte Carlo simulation with antithetic variate for European option pricing, and Monte Carlo simulation with control variate for European option pricing explained in Subsection 2.3.

We use the parameter values required for the range of parameter values in Table 1 for eight price data sets. Then, we divide the generated data into two sets, training and testing data sets.

Table 1: Parameters and ranges

Parameter	Range
Stock price (S)	\$50 – \$700
Strike price(K)	\$10 – \$500
Maturity(T)	(1 - 3)years
Risk free rate(r)	1% – 3%
Volatility(V_0)	1% − 90%
Call price(C)	\$0 − \$400
volatility of volatility(θ)	0% – 90%
Revert rate(k)	0 − 10
Long-term volatility(η)	1% − 90%
Correlation(ρ)	$-0.9 - 0.9$
Random percentage jump(μ_J)	$-0.1 - 0.1$
Jump volatility(σ_J)	$0.1 - 0.2$
Annual jump frequency(λ_J)	$0.1 - 0.2$

With respect to the range of option parameters in Table 1, we generate a large amount of data randomly. Given the linear homogeneity assumption in [6] and the theory of European option pricing, it can be inferred that the option price C is linearly dependent on the stock price S and the strike price K, and therefore the data can be normalized by dividing the option and stock prices by the exercise price. Considering the normalization of European option prices, we normalize the parameters with respect to the parameter K and enter them into the deep learning model.

Now, with respect to the four methods in each stochastic model, we generate eight sets of data with sizes of 10,000 and 20,000, respectively. We then divide the data into training and test sets. We subsequently create a validation set from a portion of the test set. We first split the overall data into training (80%) and a set of tests (20%), then allocate 20% of the test data to the validation set.

Figure 1 illustrates the data generation process flowchart for pricing European options under the Heston and Bates models. The first flowchart demonstrates the data generation process for the explicit and precise formula for pricing European options under the characteristic function of the stochastic models. In this process, we first randomly generate the parameters and then use the obtained parameters in the precise formula for pricing European options, i.e., the European option price. The second flowchart depicts the data generation process for Monte Carlo methods for pricing European options using 1000 simulated numbers.

As we proceed with the discussion of the deep learning model, we must acknowl-

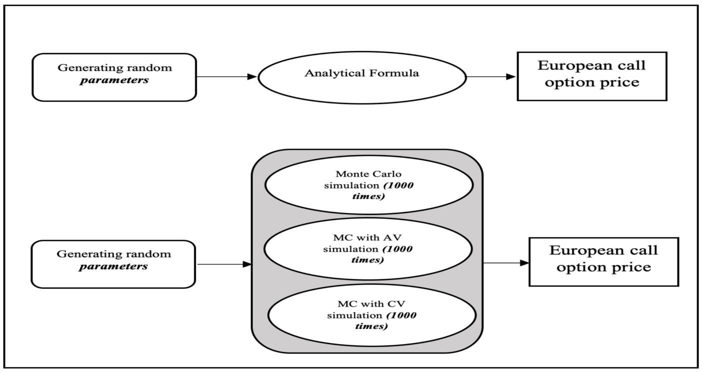

Figure 1: Flowchart of the process for generating data for pricing European call options under each mentioned model.

edge that many experiments were conducted in order to determine the optimal number of layers, the number of neurons in each layer, and the type of activation function in each layer, as well as the type of algorithm for network optimization.

According to Figure 2, the details of the deep learning network for option pricing are as follows: The network has 8 input hyperparameters, which are tuned by activating functions through 3 hidden layers, each consisting of 150, 100, and 20 neurons, respectively. ReLU activation function is chosen for the first layer, while SELU activation function is selected for the second and third layers. In the final output layer, which has a unique neuron, the exponential activation function is used to perform the computational operations. The softplus activation function is utilized because the option price cannot be a negative value. In this learning algorithm, we use the mean square error (MSE) cost function and the Adam optimization method for this deep learning network. This model is trained using the Tensorflow library.

Deep learning models can be better comprehended by using visual aids like flowcharts and diagrams. The *plot_model* function in the *keras.utils* library can generate such visual representations that showcase the structure, connections between layers, and flow of data. For even more improved clarity and aesthetics, the *visualizer* function in the *keras_visualizer* library enables customization of colors, shapes, and labels.

Using these functions enables researchers and practitioners to intuitively describe their deep learning models, making it easier to communicate their design and structure. These representations facilitate deeper understanding of deep learning models by providing a visual understanding of their architecture and flow.

Using the generated data from the previous section, we train a model for option pricing. This process is repeated 50 times. For each data group with a training set and a test set allocation, we use the data for training the deep learning model

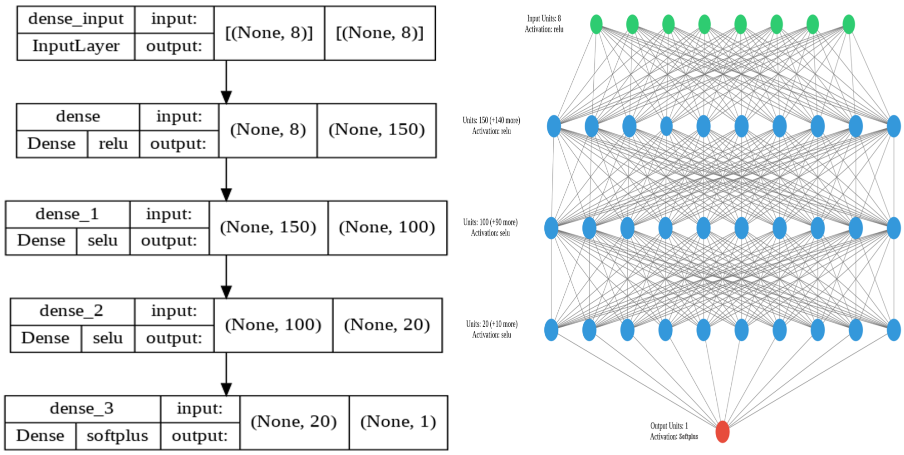

Figure 2: Flowchart and overall diagram of the deep learning model for predicting option pricing under stochastic volatility models.

50 times with 10 random initial states. Overall, each sample group is used for deep learning model training 500 times under different allocations and initial states. Each deep learning model provides 21201 parameter values as outputs, comprising of 20900 weight parameters and 301 bias parameters.

After 500 epochs of deep learning, we report three criterion for the effectiveness of deep learning: mean squared error (MSE), correlation coefficient ρ between the original and predicted data, and the R^2 value for the test set of each data group.

Tables 2 and 3 present the performance metrics of deep learning in pricing European call options using 8 price datasets divided into two groups of 10,000 and 20,000, respectively, based on the test set generated using 4 methods for pricing European call options. The results obtained from the generated data are generally as follows: mean squared error (MSE), correlation coefficient (ρ), and R^2 value in the test set are almost equal to 0, 1, and 1, respectively, indicating good performance of the model on the 8 price datasets. After demonstrating the strong performance of deep learning on different data sizes, we provide a visual representation of its effectiveness.

In the visualized results, we report 5 criteria for evaluating the efficacy of deep learning: the first column displays the predicted prices obtained through deep learning in comparison to the price data generated by the computer based on four methods for each model (Heston/Bates). The second column shows the distribution of absolute prediction errors to evaluate standardized option pricing errors using deep learning and price data generated by four methods in each model (Bates/Heston). The third column illustrates the learning curve of the model for mean squared error (MSE) on the training set and validation data set produced by four methods in each model (Heston/Bates). The fourth column depicts the distribution of squared error

Table 2: Numerical results for four option price datasets with 10,000 and 20,000 data points under the Heston model using three performance metrics

Methods	Sample size	MSE	ρ	R^2
The exact formulas	10000	6.17E-06	0.9983	0.9994
	20000	8.72E-06	0.9988	0.9994
MC	10000	3.44E-04	0.9959	0.9961
	20000	4.13E-04	0.9969	0.9973
MC with antithetic variate	10000	6.36E-03	0.9933	0.9928
	20000	1.67E-04	0.9946	0.9936
MC with control variate	10000	2.59E-04	0.9951	0.9949
	20000	3.17E-04	0.9966	0.9971

on the test set, where the y-axis indicates the probability of normal error. Column 5 is related to predicting option prices relative to actual option prices in the test data.

Figures 3 and 4 illustrate, in rows one through four, the data generation process using the closed-form formula based on the characteristic function, Monte Carlo simulation, Monte Carlo simulation with antithetic variate, and Monte Carlo simulation with control variate for each model (Heston/Bates), respectively.

Figure 3 shows the performance of deep learning in predicting European call option prices under the Heston model using four data generation methods and five evaluation metrics. The results are presented in the form of graphs, where the first

Table 3: Numerical results for four option price datasets with 10,000 and 20,000 data points under the Bates model using three performance metrics

Methods	Sample size	MSE	ρ	R^2
The exact formulas	10000	7.33E-06	0.9989	0.9996
	20000	9.01E-06	0.9994	0.9998
MC	10000	4.28E-04	0.9940	0.9957
	20000	8.67E-04	0.9963	0.9966
MC with antithetic variate	10000	3.82E-05	0.9975	0.9978
	20000	5.19E-05	0.9978	0.9988
MC with control variate	10000	4.91E-05	0.9975	0.9981
	20000	8.81E-05	0.9979	0.9990

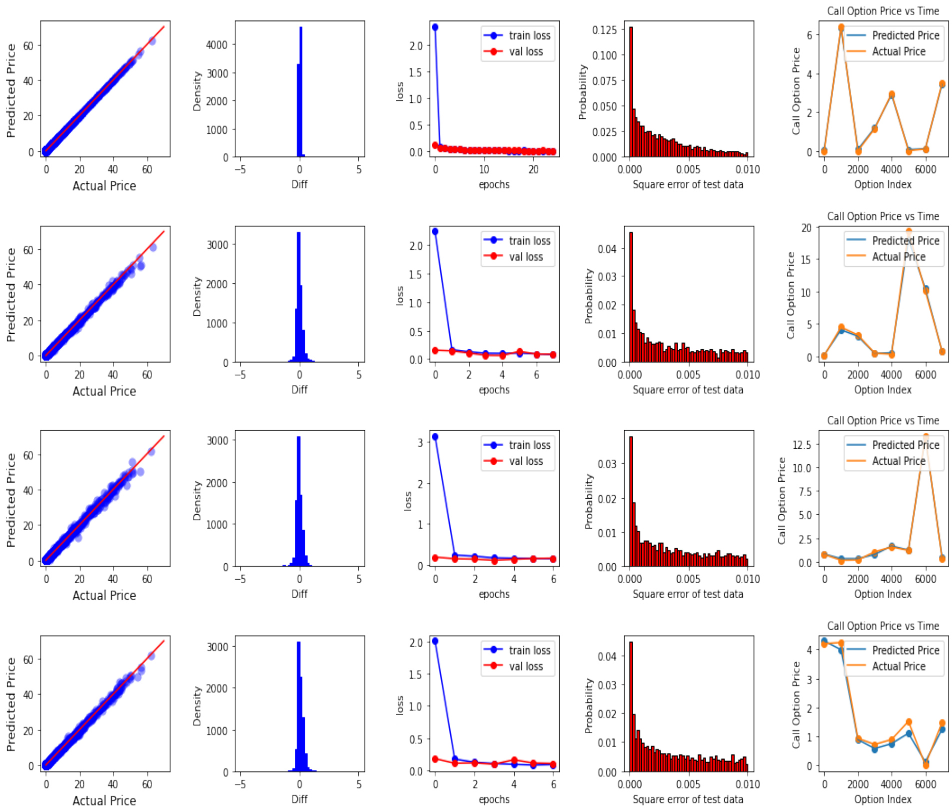

Figure 3: Flowchart of the process for generating data for pricing European call options under each mentioned model.

column shows that for all four methods, the predicted prices are very close to the actual prices. The second column shows that the pricing errors for the test data set are within the range of ± 0.02 for all methods under the Heston model. The third column shows that the deep learning curves for MSE on the training and validation data sets converge after about 6 epochs. The fourth column shows that the distribution of the squared error is mostly very small. The fifth column shows that the predicted path of option prices closely follows the actual path for all four methods under the Heston model.

Figure 4 displays the performance of deep learning models using five metrics for four different data production methods for European option pricing under the Bates model. The results are obtained using the plotted graphs as shown in the figure. For each of the four methods in the Bates model, which are used for the test set, almost all price pairs are very close to a thin line at a 45-degree angle in the first column. This indicates that the predicted prices are close to the actual option prices that are inputted into the deep learning model.

In the second column, it is observed that most pricing errors for the test set under each of the four methods in the Bates model are within the range of ± 0.02. In the third column, it can be seen that the deep learning curve for the mean squared error (MSE) for both the training and validation data sets under each of the four methods in the Bates model appears to converge after 12 epochs.

The fourth column displays the distribution of the squared error on the test set, where it is evident that most errors are very small. Finally, in the fifth column, the prediction path of the option prices is compared to the actual data in the test set, and it is observed that the predictions are very accurate for all four methods in the Bates model. These results demonstrate the effectiveness of the deep learning models for European option pricing under the Bates model using the four different data production methods.

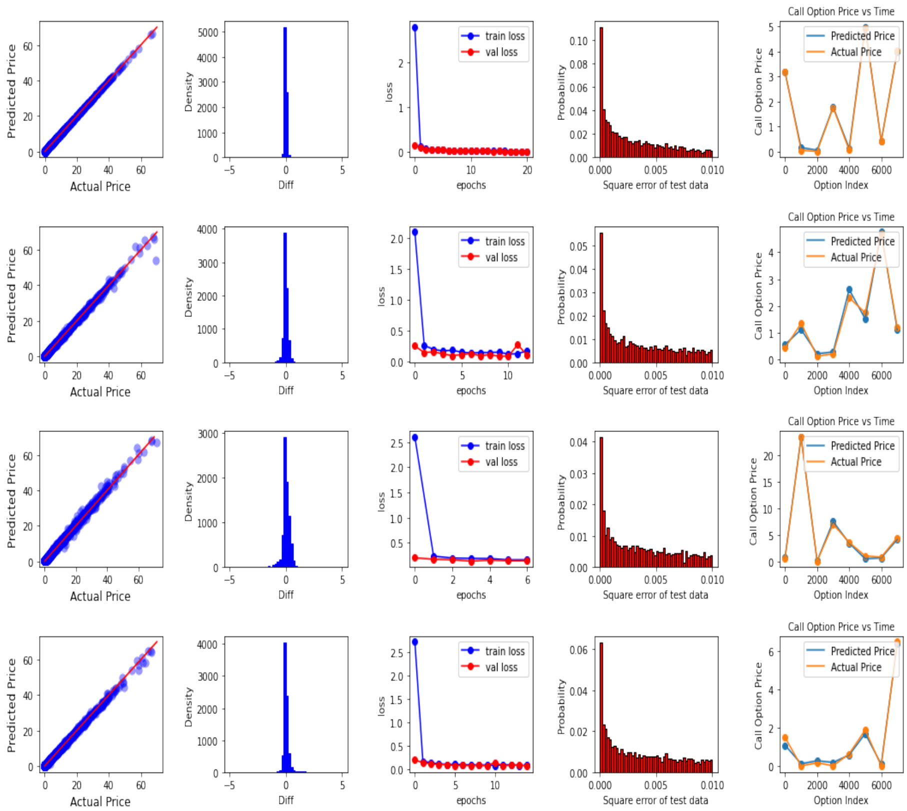

Figure 4: Flowchart of the process for generating data for pricing European call options under each mentioned model.

In summary, the deep learning model trained using four data generation methods has performed well in predicting European call option prices under the Heston and Bates models, as evidenced by the very small pricing errors and good convergence of the deep learning curves.

4.1 Numerical analysis efficiency

In this analysis, we discussed the application of deep learning in pricing European options. Finally, we examine the efficiency of the deep learning method. Table 4 compares the time to compute 1000 or 10000 European call options using five methods: the deep learning method presented by D.L. explained in Section 3, the characteristic function models for stochastic volatility discussed in Subsection 2.1, Monte Carlo simulation method, Monte Carlo simulation method with antithetic variate, and Monte Carlo simulation method with control variate models for stochastic volatility in Subsection 2.3.

Table 4: The computation time.

Number of option prices	Heston model				
	D.L	nExact formula	MC	MC with AV	MC with CV
1000	0.43	98.14	10^5	10^5	10^6
10000	0.66	139.22	10^7	10^7	10^8
	Bates model				
	D.L	Exact formula	MC	MC with AV	MC with CV
1000	0.56	101.00	10^5	10^5	10^6
10000	0.79	147.91	10^7	10^7	10^8

5 Conclusion

In this study, we investigated the pricing of European options under two models of random fluctuations, namely Heston and Bates, using a data-driven approach based on deep learning. To achieve this, we generated random pricing data using various methods such as analytical formula under the characteristic function of the stochastic models, standard Monte Carlo (MC), MC with antithetic variate, and MC with control variate, for a range of required parameters.

We designed a deep learning network with eight input hyperparameters and three hidden layers. The ReLU activation function is used for the first layer, while the SELU activation function is used for the second and third layers. The network employs the exponential activation function in the final output layer and the softplus activation function to prevent negative values. To train the network, we used the mean square error cost function and the Adam optimization method, and the

Tensorflow library was utilized for implementation.

In the accuracy analysis, three criteria, namely mean squared error (MSE), correlation coefficient (ρ) between the original and predicted data, and the coefficient of determination (R^2), were computed for various scenarios. Furthermore, the computation time of the considered methods was recorded. The results demonstrate that the deep learning method is reliable for pricing models of random fluctuations from both accuracy and computational time perspectives. Additionally, generating more accurate data using variance techniques leads to better accuracy in option pricing. The deep learning method significantly reduces computational time compared to competitive methods.

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., *Tensorflow: A system for large-scale machine learning*, In OSDI, 2016.

[2] D. S. Bates, *Jumps and stochastic volatility: Exchange rate processes implicit in deutsche mark options*, Review of financial studies, 9(1)(1996), pp. 69107.

[3] F. Black and M. Scholes, *The pricing of options and corporate liabilities*, Journal of political economy, 81(3)(1973), pp. 637654.

[4] R. Cont, *Empirical properties of asset returns: stylized facts and statistical issues*, Quantitative Finance, 1(2)(2001), pp. 223236.

[5] R. Cont, J. d. Fonseca, and V. Durrleman, *Stochastic models of implied volatility surfaces*, Economic Notes, 31(2)(2002), pp. 361377.

[6] R. Culkin and S. R. Das, *Machine learning in finance: the case of deep learning for option pricing*, Journal of Investment Management, 15(4)(2017), pp. 92100.

[7] M. F. Dixon, I. Halperin, and P. Bilokon, *Machine learning in Finance*, volume 1170, Springer, 2020.

[8] C. Dugas, Y. Bengio, F. Belisle, C. Nadeau, and R. Garcia, *Incorporating second-order functional knowledge for better option pricing*, Advances in neural information processing systems, 13(2001).

[9] J. Gatheral, *The volatility surface: A practitioners guide*, John Wiley & Sons, 2006.

[10] P. Glasserman, *Monte Carlo methods in financial engineering*, Springer, 2004.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[12] J. Heaton, N. G. Polson, and J. H. Witte, *Deep learning in finance*, arXiv preprint arXiv:1602.06561, 2016.

[13] S. L. Heston, *A closed-form solution for options with stochastic volatility with applications to bond and currency options*, Review of financial studies, 6(2)(1993), pp. :327343.

[14] J. C. Hull, *Options, Futures and Other Derivatives*, Pearson Education India, 2003.

[15] C.-F. Ivascu, *Option pricing using machine learning*, Expert Systems with Applications, 163(2021), pp. 113799.

[16] A. Jamnia, M. R. Sasouli, E. Heidouzahi, and M. Dahmarde Ghaleno, *Application of deep-learning-based models for prediction of stock price in the Iranian stock market*, Journal of Mathematics and Modeling in Finance, 2(1)(2022), pp. 151166.

[17] Z. Jiang, D. Xu, and J. Liang, *A deep reinforcement learning framework for the financial portfolio management problem*, arXiv preprint arXiv:1706.10059, 2017.

[18] A. Ke and A. Yang, *Option pricing with deep learning*, Department of Computer Science, Standford University, In CS230: Deep learning, 8 (2019), pp. 18.

[19] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, *Self-normalizing neural networks*, In Advances in Neural Information Processing Systems, (2017)pp. 971980.

[20] S. Liu, A. Borovykh, L. Grzelak, and C. Oosterlee, *A neural network-based framework for financial model calibration*, Journal of Mathematics in Industry 9(2019), pp. 1-28.

[21] R. LORD, R. KOEKKOEK, AND D. VAN DIJK, *A comparison of biased simulation schemes for stochastic volatility models*, Quantitative Finance, 10(2)(2010), pp. 177194.

[22] B. B. MANDELBROT, *The variation of certain speculative prices*, Springer New York, 1997.

[23] V. NAIR AND G. E. HINTON, *Rectified linear units improve restricted boltzmann machines*, In Proceedings of the 27th international conference on machine learning (ICML-10)(2010), pp. 807-814.

[24] K. PAKIZEH, A. MALEK, M. KARIMZADEH KHOSROSHAHI, AND H. HAMIDI RAZI, *Assessing machine learning performance in cryptocurrency market price prediction*, Journal of Mathematics and Modeling in Finance, 2(1)(2022), pp. 132.

[25] L. QIAN, J. ZHAO, AND Y. MA, *Option pricing based on GA-BP neural network*, Procedia Computer Science, 199(2022), pp.13401354.

[26] J. RUF AND W. WANG, *Neural networks for option pricing and hedging: a literature review*, arXiv preprint arXiv:1911.05620, 2019.

How to Cite: Ali Bolfake[1], Seyed Nourollah Mousavi[2], Sima Mashayekhi[3], *Deep learning for option pricing under Heston and Bates models*, Journal of Mathematics and Modeling in Finance (JMMF), Vol. 3, No. 1, Pages:67–82, (2023).