Digital Signal Processing 112 (2021) 103003

Contents lists available at ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Numerical solving of the generalized Black-Scholes differential equation using Laguerre neural network

Yinghao Chen [a], Hanyu Yu [b], Xiangyu Meng [a], Xiaoliang Xie [c], Muzhou Hou [a,*], Julien Chevallier [d,e]

[a] School of mathematics and statistics, Central South University, Changsha, 410083, China
[b] Business school, Central South University, Changsha, 410083, China
[c] School of Mathematics and Statistics, Hunan University of Technology and Business, Changsha, Hunan, 410205, China
[d] IPAG Business School (IPAG Lab), 184 boulevard Saint-Germain, Paris 75006, France
[e] University Paris 8 (LED), 2 rue de la Liberté, Saint-Denis 93526, France

ARTICLE INFO

Article history:
Available online 19 February 2021

Keywords:
Finance
Black-Scholes equation
Neural network
Numerical solution

ABSTRACT

Reasonable pricing of options in the financial derivatives market is crucial. For American options, or when volatility and interest rate are not constant, it is often difficult to obtain analytical solutions to the Black-Scholes (BS) equation. In this paper, the Laguerre neural network was proposed as a novel numerical algorithm with three layers of neurons for solving BS equations. The validity period and stock price are the input of the network, and the option price is the only output layer. Laguerre functions are used as the activation function of the neuron in the hidden layer. The BS equation and boundary conditions are set as penalty function, the training points are uniformly selected in the domain, and the improved extreme learning machine algorithm is used to optimize the network connection weights. Three experiments calculated the numerical solutions of BS equations for European options and generalized option pricing models. Compared with existing algorithms such as the finite element method and radial basis function neural network, the numerical solutions obtained by Laguerre neural network have higher accuracy and smaller errors, which illustrates the feasibility and superiority of the proposed method for solving BS equations.

1. Introduction

The key to modern financial research is how to price financial assets accurately. In 1952, Markowitz put forward the theory of portfolio selection. By treating the rate of return of security as a random variable, the mean value of this random variable is defined as the return, and the variance represents the risk of the security [33]. In 1973, Black and Scholes first proposed the Black-Scholes (BS) model for European options [6], and won the Nobel economic prize in 1997 [22]. BS model is promoted and widely used in pricing commodity futures and portfolios. In an ideal market, the price of European options can be expressed as the solution of the following partial differential equation as equation (1) [6].

$$\mathcal{V}_\tau + \frac{\sigma^2}{2}S^2\mathcal{V}_{SS} + (r-d)\,S\mathcal{V}_S - r\mathcal{V} = 0,\ (S,\tau)\in\Re\times(0,T)\,, \qquad (1)$$

where $\mathcal{V}$ represents the option price, which is related to the asset price S and the remaining time τ. r, d, and σ are constants representing the risk-free interest rate, dividend, and the annual volatility of asset prices, respectively. T means the expiration time. The asset price S is assumed to conform to geometric Brownian motion defined as equation (2):

$$dS = \mu S d\tau + \sigma S dz, \qquad (2)$$

where $\mu S d\tau$ shows the drift, and μ represents the mean growth rate. Meanwhile, $\sigma S dz$ shows the diffusion, and z is the process of Brownian motion on probability space.

Merton proposed a series of modification BS model based on the assumption that stock returns are not continuous, making the resulting model independent of investors' preference for stocks or expected returns [36–38]. Early research used arbitrage methods to estimate the price of call options when dividends were not paid based on the BS model [10]. Chan considered the Levy processes in option pricing. In this case, the noise has a random size jump [7]. Amster et al. studied the option pricing model with non-increasing linear transaction costs, and the approximate solution

* Corresponding author.
E-mail address: houmuzhou@sina.com (M. Hou).

https://doi.org/10.1016/j.dsp.2021.103003
1051-2004/© 2021 Elsevier Inc. All rights reserved.

was obtained by the upper and lower solution method [2]. By changing the execution strategy of options expiration (i.e., modifying the boundary conditions of the BS equation), American options [14,35,44] and Asian options [12,16,50] and other options have been studied. Tonze proved that the optimal exercise boundary is a decreasing function in American options with a stochastic volatility model [52]. Yavuz et al. changed the derivative of time dimension to fractional derivative, and carried out a series of studies [40,63–67]. We note that the classic BS model and its improvements are based on harsh market assumptions, such as constant risk-free interest rates, no dividends, and no transaction fees. These assumptions are usually difficult to meet in practical problems. In this work, we assume that the risk-free interest rate r, the dividend d, and the asset's volatility σ are related to the remaining time τ and the asset price S [23,48], and obtained a generalized BS model with the following form:

$$\mathcal{V}_\tau + \frac{\sigma^2(S,\tau)}{2} S^2 \mathcal{V}_{SS} + (r(S,\tau) - d(S,\tau)) S \mathcal{V}_S - r(S,\tau)\mathcal{V} = 0,$$
$$(S,\tau) \in \Re \times (0,T). \tag{3}$$

An obvious fact is that when interest rate $r(S,\ \tau)$, dividend $d(S,\ \tau)$ and volatility $\sigma(S,\ \tau)$ is constant, Equation (3) degenerates into the classic BS equation. We assume that dividend $d(S,\ \tau) > 0$. Denote the exercise price as K, the terminal value condition of European call options can be written as:

$$\mathcal{V}(S,T) = \max(S - K, 0).$$

Although European call options under the classic BS model can obtain analytical solutions, for American options and the generalized BS model, analytical expressions are difficult to calculate. Therefore, it is necessary to study a numerical solution method with high accuracy to calculate the approximate solution of the BS equation. Discretizing space domain and time domain by explicit finite difference format, Avellaneda Marco used the finite element method (FEM) to calculate the price of hedging derivatives when the transaction cost is non-zero, but this method is conditionally stable [32]. Alziary, Décamps, and Koehl discussed the approximate solutions of European options and Asian options with the finite difference method (FDM) and analyzed the difference and elasticity of European options and Asian options [1]. Forsyth used FEM to evaluate option prices under random fluctuations. Reducing the BS equation to the first-order hyperbolic Equation, the approximate solution of European options is obtained. However, experiments show that FEM has high requirements for parameter selection, the error is still as high as 35% in some cases, even after calibration [13]. Modeling stock options with discrete dividends. Company, R. et al. obtained the approximate solution of the modified Black-Scholes equation using the generalized Dirac delta function and Merlin transform [9]. Transforming the BS equation into a heat conduction equation with nonlinear convection terms In European call options. For the free boundary problem of American call options, the BS equation is transformed into a nonlinear nonlocal parabolic equation. Julia Ankudinova used FDM to calculate the approximate solution of the BS equation under different transaction cost assumptions [4]. Abraham et al. used an explicit nonstandard FDM to calculate the nonlinear BS model established under the assumption that the market is incomplete, there are transaction costs and insufficient market liquidity, but the proposed method has only first-order accuracy [5]. Using the implicit Euler method to discretize the time domain and the B-spline method to match the price domain, Mohmed obtained an unconditionally stable combination method, which can obtain an approximate solution of the BS equation with $\mathcal{O}(\Delta x^2 + \Delta\tau)$-order accuracy [24]. Godin used the Monte Carlo method in 2019 to simulate the vanilla option price based on the principle of risk neutrality under the institutional transformation model [15]. By using the high-order compact finite difference method (HOCFDM), Pradip Roul calculates the approximate solution of the generalized BS model with MATLAB, and the obtained approximate solution has second-order ($\mathcal{O}(\Delta x^4 + \Delta\tau^2)$) accuracy. Moreover, Pradip Roul proved the convergence of HOCFDM in [48]. Later, Pradip Roul designs a combination method based on HOCFDM and B-spline to calculate the time-fractional BS model used in European options [47].

As a machine learning algorithm with powerful learning ability, neural network (NN) is widely used in the research of natural science and social science [11,25,34,55,56]. In economics, neural network models are used to predict stock prices at the end of the 20th century [43], and this field is still widely studied until today [26,27,39,45,53,58,59]. Besides, the stock price trends prediction [21,57], stock index predictions [28,54], and futures price prediction [8,17] using neural network models have also been studied by some researchers. Given the powerful representation capabilities of neural network models, some researchers have proposed NN models for numerically solving differential equations [3,29–31,41,46,49,51,61,69]. Qu et al. used a cosine basis function neural network to calculate the fractional differential equation. An approximate solution is obtained by repeatedly training the model, but this method requires multiple training steps and costs more computing time [42]. Chebyshev neural networks are also introduced for numerically solving differential equations [31]. Hou et al. proposed the Legendre neural network for solving ordinary and partial differential equations with extreme learning machine (ELM) algorithm [60,62]. Chen et al. constructed a trigonometric exponential function neural network model to solve the ruin probability equation and obtained a numerical solution with higher accuracy [68]. As a kind of orthogonal polynomials associated with gamma distribution density function, Laguerre function has important applications in quantum mechanics and physics, but the related research is lack.

In this study, we proposed the Laguerre neural networks (LNN) for calculating the approximate solutions of the generalized Black-Scholes equation. Compared with FDM and FEM, LNN avoids discretization and improves solution efficiency. Moreover, LNN has stronger representation ability, which results in higher accuracy solutions than radial basis function (RBF) neural networks.

The rest of this paper is organized as follows: the ELM algorithm is briefly reviewed in Section 2. In Section 3, the mathematical model of the Laguerre neural network is given. Section 4 discusses the application of the Laguerre neural network for numerical solving the Black-Scholes equation. Three numerical examples are explored in Section 5. Finally, Section 6 briefly summarizes our study and gives a prospect.

2. Extreme learning machine algorithm

The single hidden layer feedforward neural network (SLFN) model is a widely used neural network structure. Because it contains only one hidden layer, SLFN is easier for training. Nevertheless, using the back-propagation algorithm to train the model still cannot avoid gradient explosion problems or dispersion and overfitting. Huang proposed a novel method for training SLFN without iteration, an extreme learning machine (ELM) algorithm [20]. Huang et al. perfected the theory and application of ELM in classification and regression tasks in [18,19]. The brief topology structure of an SLFN model is plotted in Fig. 1.

For a data set with M training samples $\{(x_i, y_i)\}_{i=1}^M \in R^{M\times(N+1)}$, y_i is the label of i-th sample and $x_i = \left(x_i^1, x_i^2, \cdots, x_i^N\right)$ represent N features. In light of Fig. 1, the output of a single hidden layer network with m hidden neurons can be calculated with the following mathematical expression:

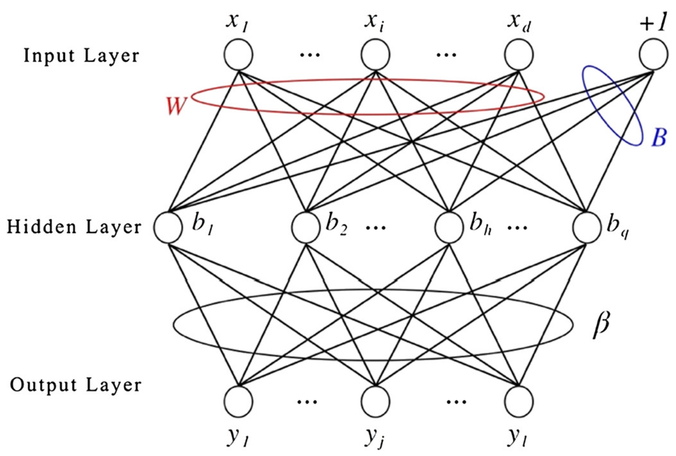


Fig. 1. The topology structure of a single hidden layer feedforward neural network (SLFN).

$$f_m(x) = \sum_{i=1}^{m} \beta_i G(a_i, b_i, x) = h(x)\beta,$$

where $G(a_i, b_i, x)$ is the activation function (generally a nonlinear function), which is used to improve the expression ability of the neural network. $h(x) = [G(a_1, b_1, x), G(a_2, b_2, x), \cdots, G(a_m, b_m, x)]$ is the projection vector of the input x on the m-dimensional feature space obtained by a cluster of activation functions. $\beta = [\beta_1, \beta_2, \cdots, \beta_m]^T$ is the connection weight that needs to be optimized. Whenever the output of the ELM neural network can approach the data set with zero error, Equation (5) is satisfied for each training sample, and can be written as:

$$f_m(x_j) = \sum_{i=1}^{m} \beta_i G(a_i, b_i, x_j) = h(x_j)\beta = y_j \quad j = 1, 2, \cdots M. \tag{4}$$

Equation (4) can be rewritten as a matrix form, that is:

$$H\beta = y, \tag{5}$$

where H in equation (5) is,

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_M) \end{bmatrix} = \begin{bmatrix} h_1(x_1) & \cdots & h_m(x_1) \\ \vdots & \ddots & \vdots \\ h_1(x_M) & \cdots & h_m(x_M) \end{bmatrix}, \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_M \end{bmatrix}.$$

For any input layer connection weight a_i and deviation b_i, instead of an iteration, ELM uses least squares instead of repeated iterations to optimize the hidden layer weight β that minimizes 2-norm of the output error, i.e.

$$\beta^* = arg\min_{\beta} \|H\beta - y\|^2 = H^\dagger y,$$

where, $H^\dagger = (H^T H)^{-1} H^T$ represents the Moore-Penrose generalized inverse of matrix H.

3. Laguerre neural network

In our study, Laguerre neural network is established as a special case of a single hidden layer feedforward neural network model. An LNN model usually has two input neurons, which are used to receive information about stock price and expiration time. When the option contains multiple stocks, the information can be processed separately by adding neurons in the input layer. The only output layer neuron is used to display the numerical solution of the BSM equation. The number of neurons in the hidden layer is given in advance before network training, and more hidden layer neurons can usually obtain a higher-precision approximate solution. The expansion of Laguerre polynomials is used as the activation function in hidden layer neurons, which provides powerful nonlinear fitting capabilities for the network. The following recurrence relation can obtain the Laguerre polynomial:

$$L_0(x) = 1,$$
$$L_1(x) = -x + 1,$$
$$L_{n+1}(x) = (2n + 1 - x) L_n(x) - n^2 L_{n-1}(x), \quad (n \geq 1).$$

As a solution to the Laguerre equation $xy_{xx} + (1 - x) y_x + \lambda y = 0$, the Laguerre polynomial has good orthogonal properties. It is pairwise orthogonal to the weight function e^{-x}, which simplifies the calculation process of the network.

$$\int_0^\infty e^{-x} L_n(x) L_m(x)\, dx = \begin{cases} 0, & m \neq n \\ (n!)^2, & m = n \end{cases}.$$

Consider an m dimensional input vector $x = (x_1, x_2, \cdots, x_m)$. Project it to the high-dimensional feature space through first $N - 1$ Laguerre polynomial, the enhanced coordinates can be written as:

$$\begin{pmatrix} L_0(x_1) & \cdots & L_{N-1}(x_1) \\ \vdots & \ddots & \vdots \\ L_0(x_m) & \cdots & L_{N-1}(x_m) \end{pmatrix}.$$

The linear combination coefficient $\alpha = (\alpha_0, \alpha_1, \cdots, \alpha_{N-1})$ can be obtained using the ELM algorithm, then the output of the LNN can be calculated as formula (6):

$$Network(x, \alpha) = \sum_{i=0}^{N-1} \alpha_i L_i(x). \tag{6}$$

The structure of LNN is shown in Fig. 2, where $m = 2$. In this study, this LNN model is applied for obtaining the approximate solution of Black-Scholes equations.

Remark 1 *(Convergence theorem).* Consider $u(z) \in C^\infty(0, 1)$ and denote $u_N(z) = \sum_{i=0}^{N-1} \alpha_i L_i(z)$. Let ε_N be the approximate error with N hidden neurons. Then $|\varepsilon_N(z)| = \mathcal{O}(2^{-N+1})$.

Proof. According to the Taylor expansion formula, we have:

$$\begin{aligned} u(z) &= \frac{u(z_0)}{0!}(z - z_0)^0 + \frac{u'(z_0)}{1!}(z - z_0)^1 + \frac{u''(z_0)}{2!}(z - z_0)^2 \\ &\quad + \cdots + \frac{u^k(z_0)}{k!}(z - z_0)^k + \cdots \\ &= \sum_{i=0}^{\infty} \frac{u^i(z_0)}{i!}(z - z_0)^i. \end{aligned} \tag{7}$$

Formula (7) is the sum of infinite series, which can be regarded as a combination of two parts:

$$\begin{aligned} u(z) &= \sum_{i=0}^{N-1} \frac{u^i(z_0)}{i!}(z - z_0)^i + \sum_{i=N-1}^{\infty} \frac{u^i(z_0)}{i!}(z - z_0)^i \\ &= \sum_{i=0}^{N-1} \frac{u^i(z_0)}{i!}(z - z_0)^i + \mathcal{O}\left((z - z_0)^{N-1}\right). \end{aligned} \tag{8}$$

On the other hand, $u_N(z) = \sum_{i=0}^{N-1} \alpha_i L_i(z)$, in this way, we can obtain:

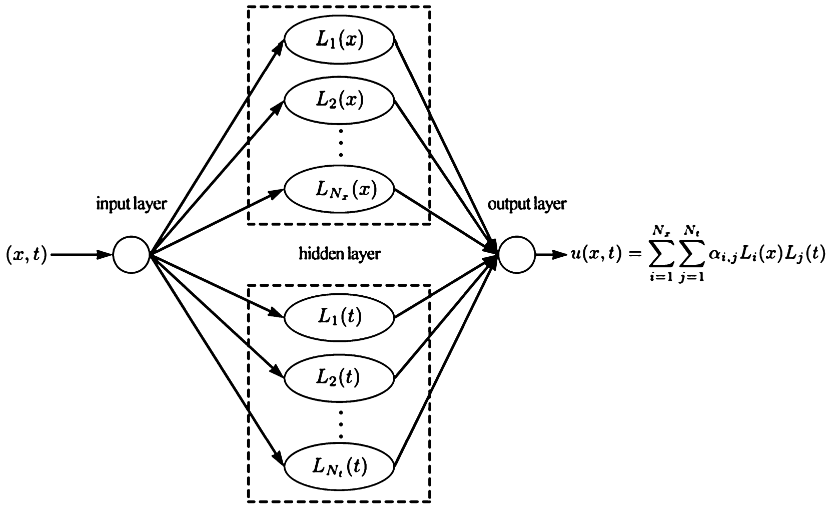


Fig. 2. The structure of a Laguerre neural network with two input neurons.

$$
\begin{aligned}
&|\varepsilon_N(z)| \\
&=|u(z)-u_N(z)| \\
&=\left|\sum_{i=0}^{\infty}\frac{u^i(z_0)}{i!}(z-z_0)^i-\sum_{i=0}^{N-1}\alpha_iL_i(z)\right| \\
&=\left|\sum_{i=0}^{N-1}\frac{u^i(z_0)}{i!}(z-z_0)^i+\sum_{i=N-1}^{\infty}\frac{u^i(z_0)}{i!}(z-z_0)^i-\sum_{i=0}^{N-1}\alpha_iL_i(z)\right| \\
&\le\sum_{i=0}^{N-1}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i-\alpha_iL_i(z)\right|+\sum_{i=N-1}^{\infty}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i\right|.
\end{aligned}
$$

Let $\alpha_i^*=arg\min_\alpha\left\|\sum_{i=0}^{N-1}\alpha_iL_i(z)-u(z)\right\|$, we get

$$
\sum_{i=0}^{N-1}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i-\alpha_i^*L_i(z)\right|\le\left|\frac{u^N(z_0)}{N!}(z-z_0)^N\right|. \tag{9}
$$

In addition,

$$
\sum_{i=N-1}^{\infty}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i\right|\le\left|\frac{u^N(z_0)}{N!}(z-z_0)^N\right|. \tag{10}
$$

Substituting inequality condition (9) and (10) into equation (8), that is

$$
\begin{aligned}
|\varepsilon_N(z)|\le&\sum_{i=0}^{N-1}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i-\alpha_iL_i(z)\right| \\
&+\sum_{i=N-1}^{\infty}\left|\frac{u^i(z_0)}{i!}(z-z_0)^i\right| \\
&\le 2\left|\frac{u^N(z_0)}{N!}(z-z_0)^N\right|\le 2^{-N+1}.
\end{aligned}
\tag{11}
$$

Hence $|\varepsilon_N(z)|=\mathcal{O}\left(2^{-N+1}\right)$.

Moreover, we can find from Equation (11) that for a vast N, $|\varepsilon_N(z)|\to 0$. This means LNN can approximate the exact solution almost without error and proves that LNN has good representation ability.

4. The LNN model for the BS equation

In this section, we will introduce the form and details of numerical solving generalized Black-Scholes differential equations by Laguerre neural network. First, consider the generalized BS equation in Equation (3), but we can easily find that it is degenerate and backward to time. Let $S=e^x$ and $\tau=T-t$, Equation (3) can be transformed into a non-degenerate and time-forward form, which can be written as follows:

$$
\begin{aligned}
\frac{\partial u(x,t)}{\partial t}=&A(x,t)\frac{\partial^2u(x,t)}{\partial x^2}+B(x,t)\frac{\partial u(x,t)}{\partial x}+C(x,t)u(x,t) \\
&+D(x,t),
\end{aligned}
\tag{12}
$$

$$(x,t)\in\Omega=\Omega_x\times\Omega_t=[x_{min},x_{max}]\times[0,T]$$

where,

$$
\begin{aligned}
A(x,t)&=\frac{\hat\sigma^2(x,t)}{2}, \\
B(x,t)&=r(S,\tau)-d(S,\tau)-\frac{\hat\sigma^2(x,t)}{2} \\
&=\hat r(x,t)-\hat d(x,t)-\frac{\hat\sigma^2(x,t)}{2}, \\
C(x,t)&=-r(S,\tau)=-\hat r(x,t).
\end{aligned}
$$

The boundary condition can be written as:

$$
\begin{aligned}
&u(x,0)=\psi(x),x\in[x_{min},x_{max}], \\
&u(x_{min},t)=\varphi_l(t),t\in[0,T], \\
&u(x_{max},t)=\varphi_r(t),t\in[0,T].
\end{aligned}
$$

$D(x,t)$ in Equation (12) is usually 0, but more generally, it can be written as a function of variables x and t. For numerical solving Equation (12), a trial solution can be established with the following form:

$$
\tilde u(x,t)=\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1}\alpha_{i,j}L_i(x)L_j(t), \tag{13}
$$

where, $L_i(x)$ is the i-th Laguerre polynomial, and $\alpha_{i,j}$ is the connection weights that need to be solved.

The trial solution $\tilde u(x,t)$ is infinitely differentiable, replaced $u(x,t)$ by $\tilde u(x,t)$ in (12), we have,

$$\frac{\partial \tilde{u}(x,t)}{\partial t} = \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} L_i(x) L_j'(t), \tag{14}$$

$$\frac{\partial u(x,t)}{\partial x} = \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} L_j'(x) L_i(t), \tag{15}$$

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} L_j''(x) L_i(t). \tag{16}$$

Substituting equations (13)-(16) into generalized BS equation (12), that is,

$$LHS = \frac{\partial u(x,t)}{\partial t} = \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} L_i(x) L_j'(t), \tag{17}$$

$$\begin{aligned} RHS &= A(x,t)\frac{\partial^2 u(x,t)}{\partial x^2} + B(x,t)\frac{\partial u(x,t)}{\partial x} + C(x,t)u(x,t) \\ &\quad + D(x,t) \\ &= \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j}\Big\{A(x,t)L_j''(x)L_i(t) + B(x,t)L_j'(x)L_i(t) \\ &\quad + C(x,t)L_i(x)L_j(t)\Big\} + D(x,t). \end{aligned} \tag{18}$$

Merging the similar items, equations (17) and (18) can be written as:

$$D(x,t) = \sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} HH(x,t), \tag{19}$$

where $H_{i,j}(x,t) = L_i(x)L_j'(t) - A(x,t)L_j''(x)L_i(t) - B(x,t)L_j'(x)L_i(t) - C(x,t)L_i(x)L_j(t)$.

Take a series of training points as:

$$\Omega_{train} = \left\{x_{min} = x_1 < x_2 < x_{m_x} = x_{max}\right\} \times \left\{0 = t_1 < t_2 < t_{m_t} = T\right\}.$$

A clear fact is that Equation (19) is considered satisfied at these training points, that is:

$$\sum_{i=0}^{N_x-1}\sum_{j=0}^{N_t-1} \alpha_{i,j} H_{i,j}(x_m, t_n) = D(x_m, t_n), \ (x_m, t_n) \in \Omega_{train}. \tag{20}$$

Equation (20) can be rewritten as a matrix form, we get

$$H\alpha = D,$$

where,

$$\begin{aligned}\alpha = \big(&\alpha_{0,0}, \alpha_{1,0}, \cdots \alpha_{N_x-1,0}, \alpha_{0,1}, \alpha_{1,1}, \cdots \alpha_{N_x-1,1}, \cdots, \\ &\alpha_{0,N_t-1}, \alpha_{1,N_t-1}, \cdots \alpha_{N_x-1,N_t-1}\big)^T,\end{aligned}$$

$$H = \begin{pmatrix} H_{0,0}(x_1,t_1) & H_{1,0}(x_1,t_1) & \dots & H_{N_x-1,0}(x_1,t_1) & \dots & H_{N_x-1,N_t-1}(x_1,t_1) \\ H_{0,0}(x_2,t_1) & H_{1,0}(x_2,t_1) & \dots & H_{N_x-1,0}(x_2,t_1) & \dots & H_{N_x-1,N_t-1}(x_2,t_1) \\ \vdots & \vdots & \ddots & s\vdots & \ddots & \vdots \\ H_{0,0}(x_{m_x},t_1) & H_{1,0}(x_{m_x},t_1) & \dots & H_{N_x-1,0}(x_{m_x},t_1) & \dots & H_{N_x-1,N_t-1}(x_{m_x},t_1) \\ H_{0,0}(x_1,t_2) & H_{1,0}(x_1,t_2) & \dots & H_{N_x-1,0}(x_1,t_2) & \dots & H_{N_x-1,N_t-1}(x_1,t_2) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ H_{0,0}(x_{m_x},t_{m_t}) & H_{1,0}(x_{m_x},t_{m_t-1}) & \dots & H_{N_x-1,0}(x_{m_x},t_{m_t}) & \dots & H_{N_x-1,N_t-1}(x_{m_x},t_{m_t}) \end{pmatrix}$$

$$D = \begin{pmatrix} D(x_1,t_1) & D(x_2,t_1) & \dots & D(x_1,t_2) & \dots & D(x_{m_x},t_{m_t}) \end{pmatrix}^T.$$

Moreover, the trial solution also need to meet the boundary conditions, that is:

$$\tilde{u}(x,0) = \psi(x), \quad x \in \left\{x_{min} = x_1 < x_2 < x_{m_x} = x_{max}\right\}, \tag{21}$$

$$\tilde{u}(x_1,t) = \varphi_l(t), \quad t \in \left\{0 = t_1 < t_2 < t_{m_t} = T\right\}, \tag{22}$$

$$\tilde{u}\left(x_{m_x},t\right) = \varphi_r(t), \quad t \in \left\{0 = t_1 < t_2 < t_{m_t} = T\right\}. \tag{23}$$

Expanding equations (21)-(23) and express it as a matrix from, we get:

$$H_B\alpha = D_B,$$

where,

$$H_B = \begin{pmatrix} L_{0,0}(x_1,0) & L_{1,0}(x_1,0) & \dots & L_{N_x-1,0}(x_1,0) & \dots & L_{N_x-1,N_t-1}(x_1,0) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ L_{0,0}(x_{m_x},0) & L_{1,0}(x_{m_x},0) & \dots & L_{N_x-1,0}(x_{m_x},0) & \dots & L_{N_x-1,N_t-1}(x_{m_x},0) \\ L_{0,0}(x_l,t_1) & L_{1,0}(x_l,t_1) & \dots & L_{N_x-1,0}(x_l,t_1) & \dots & L_{N_x-1,N_t-1}(x_l,t_1) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ L_{0,0}(x_l,t_{m_t}) & L_{1,0}(x_l,t_{m_t}) & \dots & L_{N_x-1,0}(x_l,t_{m_t}) & \dots & L_{N_x-1,N_t-1}(x_l,t_{m_t}) \\ L_{0,0}(x_r,t_1) & L_{1,0}(x_r,t_1) & \dots & L_{N_x-1,0}(x_r,t_1) & \dots & L_{N_x-1,N_t-1}(x_r,t_1) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ L_{0,0}(x_r,t_{m_t}) & L_{1,0}(x_r,t_{m_t}) & \dots & L_{N_x-1,0}(x_r,t_{m_t}) & \dots & L_{N_x-1,N_t-1}(x_r,t_{m_t}) \end{pmatrix},$$

$$D_B = \begin{pmatrix} \psi(x_1) & \dots & \psi(x_{m_x}) & \varphi_l(t_1) & \dots & \varphi_l(t_{m_t}) & \varphi_r(t_1) & \dots & \varphi_r(t_{m_t}) \end{pmatrix}^T,$$

and $L_{i,j}(x,t) = L_i(x)L_j(t)$.

When the trial solution satisfies the boundary conditions and the generalized BS equation simultaneously, the following linear system can be obtained.

$$\begin{pmatrix} H \\ H_B \end{pmatrix}\alpha = \begin{pmatrix} D \\ D_B \end{pmatrix}.$$

We use the extreme learning machine to calculate the optimal connection weight vector of the network, that is:

$$\alpha^* = \begin{pmatrix} H \\ H_B \end{pmatrix}^{\dagger} \begin{pmatrix} D \\ D_B \end{pmatrix}, \tag{24}$$

where $A^{\dagger} = \left(A^T A\right)^{-1} A^T$ is the least square error solution of system (24), and it exists and is unique.

5. Numerical results and analysis

In this section, three numerical experiments are presented to verify the efficiency of the proposed LNN model. All experiments are run on a late 2013 MacBook Pro laptop, and coding with Python 3.6.5. The first and second experiments are the classical Black-Scholes Equation of European call options. To further illustrate the advantages of the proposed LNN method, generalized BS equations are shown in Experiment 3. For facilitating the calculation, the connecting weights ϖ and their deviations b have set up as 1 and 0, respectively.

Test points are uniformly selected in the solution domain, and compare the difference between the real solution and the approximate solution at these test points. Mean square error (MSE) and maximum absolute error (MAE) are used to measure the accuracy and reliability of different methods. They can be calculated as:

$$MSE = \frac{1}{N_{test}} \sum_{l=1}^{s_{test}} \sum_{k=1}^{t_{test}} \left(y(s_l,t_k) - \tilde{y}(s_l,t_k)\right)^2,$$

$$MAE = \max_{l=1,2,\cdots,N_{test}} \left|y(s_l,t_k) - \tilde{y}(s_l,t_k)\right|,$$

where s_{test} and t_{test} represent the division of test points in the price domain and time domain, and $N_{test} = s_{test} \times t_{test}$ is the number of all test points. $y(x)$ is the close solution of the generalized Black-Scholes equation (if it exists), and $\tilde{y}$ means the approximate solution.

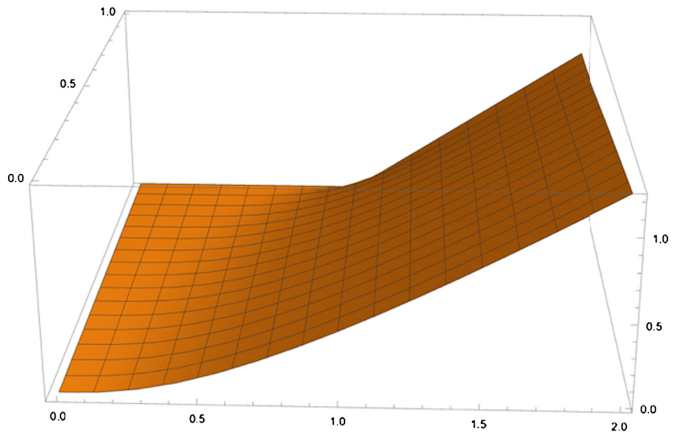


Fig. 3. The approximate solution of Example 1.

5.1. Numerical Experiment 1

This example illustrates the classical Black-Scholes Equation of European call options. We support the risk-free rate $r = 0.05$, the exercise price $K = 1$, the annual volatility $\sigma = 1$, and the stock price S fluctuates in $[0, 2]$. In this case, the BS equation can be expressed as:

$$\begin{cases} C_t + \frac{\sigma^2}{2} S^2 C_{SS} + rSC_S - rC = 0, (S,t) \in [0,2] \times [0,1] \\ C(S,1) = max(S-K,0) \\ C(0,t) = 0 \\ C(2,t) = 2 \times E(\dfrac{\ln(2) + \left(r + \frac{\sigma^2}{2}\right)(1-t)}{\sigma\sqrt{1-t}}) \\ \quad - 2e^{-r(1-t)} E(\dfrac{\ln(2) + \left(r - \frac{\sigma^2}{2}\right)(1-t)}{\sigma\sqrt{1-t}}) \end{cases} . \quad (25)$$

First let $x = e^S$ and $\tau = 1 - t$, then transformed equation into a non-degenerate and forward in time form, that is:

$$\begin{cases} C_\tau = \frac{\sigma^2}{2} C_x + rC_x - rC \\ C(x,0) = max(e^x - K, 0) \\ C(-\infty, \tau) = 0 \\ C(\ln 2, \tau) = 2 \times E(\frac{\ln(2) + \left(r + \frac{\sigma^2}{2}\right)\tau}{\sigma\sqrt{\tau}}) - 2e^{-r\tau} E(\frac{\ln(2) + \left(r - \frac{\sigma^2}{2}\right)\tau}{\sigma\sqrt{\tau}}) \end{cases}$$

The exact solution of equation (25) is given by:

$$C(S,t) = S \times E(d_1) - 2e^{-r(1-t)} E(d_2),$$

where $d_1 = \frac{\ln\left(\frac{S}{2}\right) + \left(r + \frac{\sigma^2}{2}\right)(1-t)}{\sigma\sqrt{1-t}}$, $d_2 = \frac{\ln\left(\frac{S}{2}\right) + \left(r - \frac{\sigma^2}{2}\right)(1-t)}{\sigma\sqrt{1-t}}$ and $E(z) = \frac{1}{2}\mathrm{Erf}\left(\frac{-z}{\sqrt{2}}\right)$ is the cumulative distribution function of the normal distribution.

Ten hidden neurons are used in each domain, and the first ten order Laguerre polynomial is used as basis functions. One hundred training points are equidistantly chosen for training the LNN model. To show that the numerical solution obtained is accurate, test points are selected each 0.1 step size. The approximate solution is described in Fig. 3, and the absolute error is shown in Fig. 4.

From Fig. 4, we can see that the numerical solution obtained by LNN has high accuracy, and the mean square error is only 2.30×10^{-8}. In most solution domains, LNN can obtain the approximate solution of the Equation more accurately, and the maximum absolute error occurs at $t = 0.8$. This is because the solution is not smooth enough near $t = T$. Table 1 compares the MAE and MSE of the approximate solution obtained by the LNN model, FEM, and FDM, which shows that the numerical solution by LNN has a smaller error than FEM and FDM. Moreover, Table 2 compares the errors of the numerical solutions obtained by LNN when more training points added and configuring different numbers of hidden layer neurons. From Fig. 3, we can easily find that the option price will increase with the increase of asset price S. In addition, the option with longer maturity has higher price. When the asset price S is greater than the exercise price K, the value of the option increases greatly.

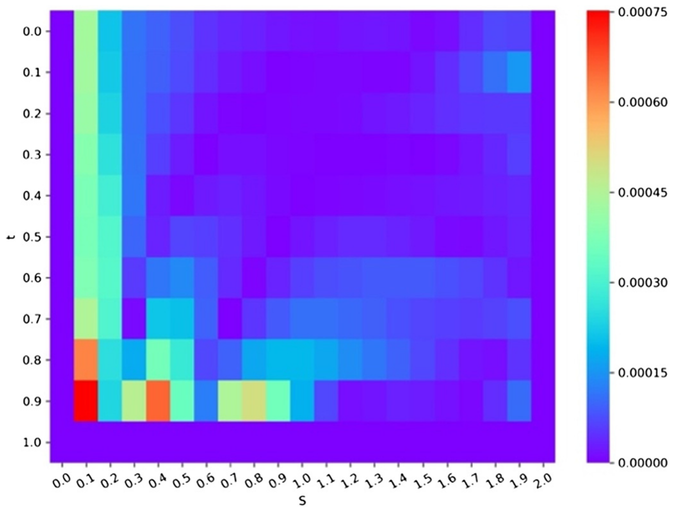


Fig. 4. The absolute error of Example 1. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 1
Comparison of the errors of Example 1 obtained by different methods.

Method	MSE	MAE
LNN $(N = 10)$	2.30×10^{-8}	7.53×10^{-4}
FEM $(N = 10)$	8.19×10^{-7}	9.70×10^{-3}
FDM $(N = 10)$	9.56×10^{-7}	2.85×10^{-2}
B-Spline $(N = 10)$	4.82×10^{-7}	1.59×10^{-2}

Table 2
Comparison of the errors of Example 1 with different numbers of training points and hidden neurons.

Method	MSE	MAE
LNN $(N = 10, M = 10)$	2.30×10^{-8}	7.53×10^{-4}
LNN $(N = 10, M = 20)$	$1.84. \times 10^{-8}$	5.18×10^{-4}
LNN $(N = 10, M = 40)$	1.07×10^{-8}	3.92×10^{-4}
LNN $(N = 15, M = 20)$	5.27×10^{-9}	7.16×10^{-5}
LNN $(N = 20, M = 20)$	3.64×10^{-9}	4.58×10^{-5}
LNN $(N = 40, M = 40)$	$3.89. \times 10^{-10}$	6.91×10^{-6}

5.2. Numerical Experiment 2

Consider the following Black Scholes differential equation for the European call option [48].

$$\frac{\partial y}{\partial t} + 0.08S^2 \frac{\partial^2 y}{\partial S^2} + 0.04 \frac{\partial y}{\partial S} - 0.06y = 0,$$
$$(S,t) \in (0,2) \times (0,1)$$

The boundary conditions are as follows:

$$\begin{cases} y(0,t) = 0 \\ y(2,t) = 2e^{-0.02(1-t)} - e^{-0.06(1-t)} \\ y(S,1) = max(S-1,0) \end{cases}$$

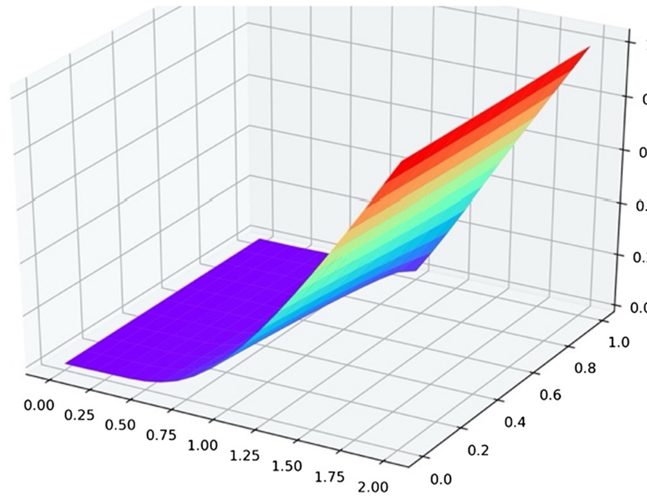


Fig. 5. The numerical solution obtained by LNN of Example 2.

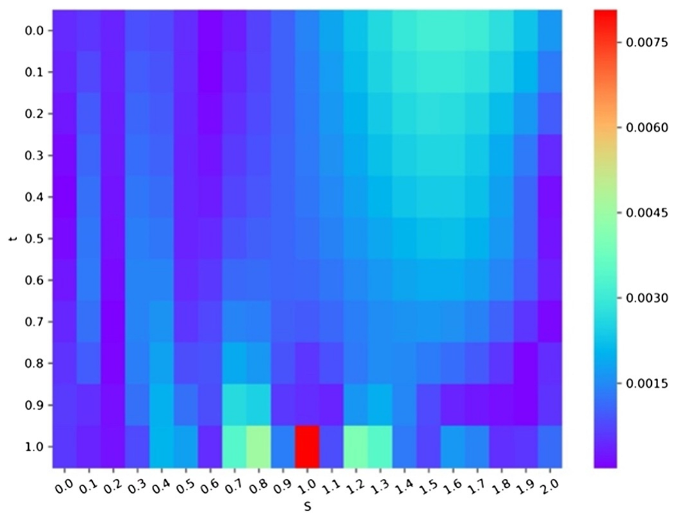


Fig. 6. The absolute error obtained by LNN of Example 2.

First let $x = e^S$ and $\tau = 1 - t$, then transformed equation into a non-degenerate and forward in time form, that is:

$$u_\tau(x,t) = 0.08u_{xx}(x,\tau) - 0.04u_x(x,\tau) - 0.06u(x,\tau), \tag{26}$$

and the boundary conditions can be rewritten as:

$$\begin{cases} u(-\infty,t) = 0 \\ u(2,\tau) = e^{2-0.02\tau} - e^{-0.06\tau} \\ u(x,0) = max\left(e^x - 1, 0\right) \end{cases}. \tag{27}$$

The exact solution of the BS problem (26), (27) can be calculated as:

$$u(x,\tau) = E(d_1)e^{x-0.02\tau} - E(d_2)e^{-0.06\tau},$$

where

$$d_1(x,\tau) = \frac{x + 0.12\tau}{0.4\sqrt{\tau}} \quad and \quad d_1(x,\tau) = d_1(x,\tau) - 0.4\sqrt{\tau}.$$

Setting ten hidden neurons in both x and τ domain, the numerical solution is plotted in Fig. 5. Simultaneously, the comparison of the MAE and MSE of the approximate solution obtained by the finite element method and the B-spline method with ten meshes is shown in Table 3. Although the LNN and B-spline method's maximum error is similar, the LNN method has a smaller MSE, which shows that the approximate solution obtained by LNN has higher accuracy in most regions. Fig. 6 analyzed the absolute error of the approximate solution obtained by LNN at the test points. When the asset price is higher than the exercise price, the option price is mainly affected by the asset price. On the other hand, when the asset price is lower than the exercise price, the maturity time has a greater impact on the option price. On the exercise date, the price of the option is $y(S,1) = max(S-1,0)$. The equation can only guarantee $\mathbb{C}_0$ continuity at point $(1,1)$, and the discontinuity of the first derivative makes it difficult for LNN to approximate accurately near this point. Except for the largest error at point $(1,1)$, the absolute error in most test points is less than 3×10^{-3}.

Table 3
Comparison of the errors of Example 2 obtained by different methods.

Method	MSE	MAE
LNN ($N = 10$)	2.54×10^{-6}	8.07×10^{-3}
FEM ($N = 10$)	3.67×10^{-5}	6.43×10^{-2}
B-spline ($N = 10$)	1.32×10^{-5}	9.52×10^{-3}
LNN ($N = 500$)	5.81×10^{-10}	9.31×10^{-5}
HOCFDM ($N = 500$)	9.26×10^{-8}	6.93×10^{-4}

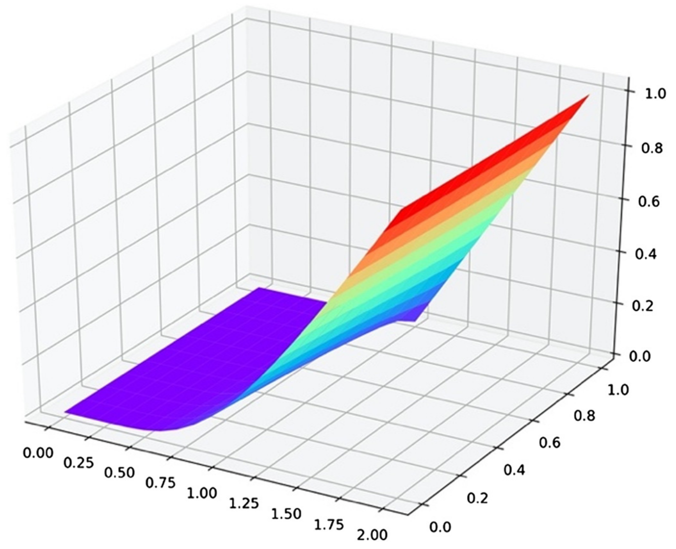


Fig. 7. The numerical solution obtained by LNN with 40 hidden neurons of Example 2.

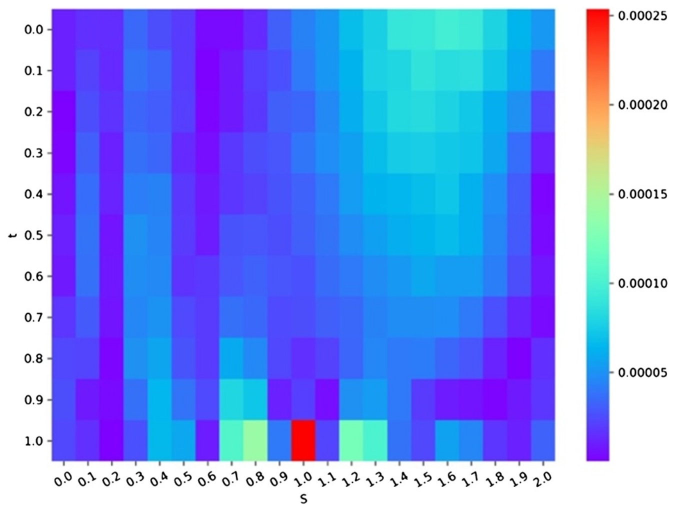


Fig. 8. The absolute error obtained by LNN with 40 hidden neurons of Example 2.

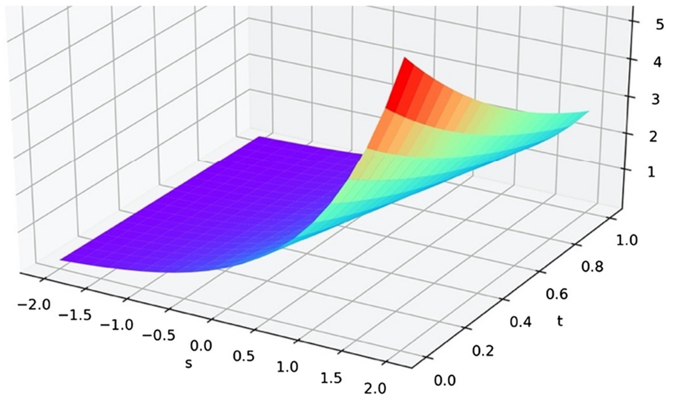


Fig. 9. The exact solution of Example 3.

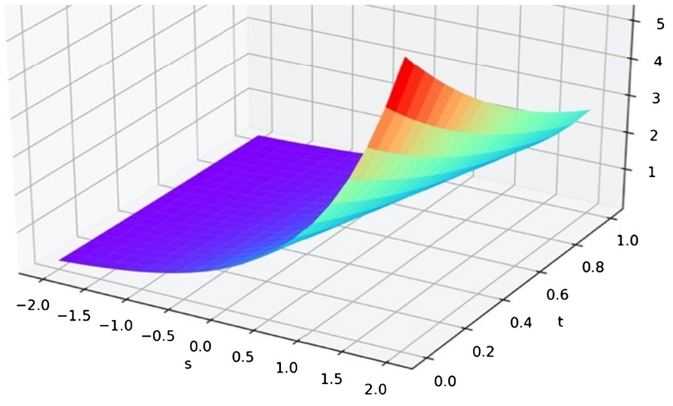


Fig. 10. The numerical solution obtained by LNN of Example 3.

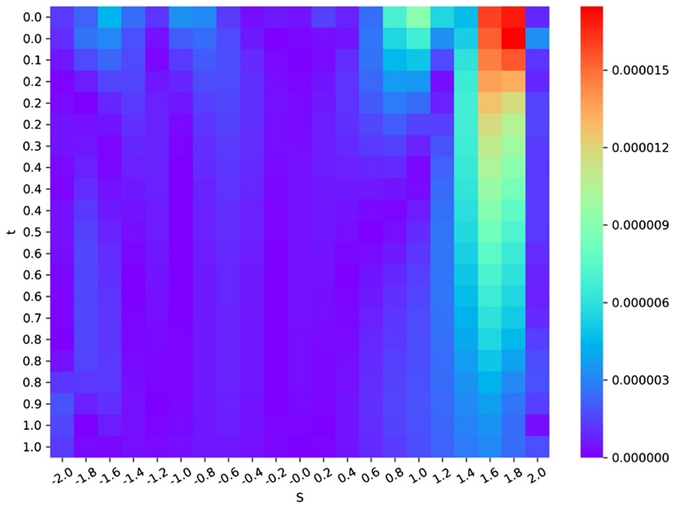


Fig. 11. The absolute error of the approximation obtained by the LNN at test points.

Furthermore, by increasing the number of neurons in the hidden layer, a higher-precision approximate solution can be obtained. Fig. 7 demonstrated the numerical results when 40 neurons are used, which is similar to the result in Fig. 5. In Fig. 8, we compare the maximum absolute error of each test points.

5.3. Numerical Experiment 3

Consider the following generalized BS model [48]:

$$u_t(x,t) = 0.02exp(-x - e^x - 2t) - e^{x-t}$$

$$+ 0.08\left(2 + (1-t)\sin\left(e^x\right)\right)^2 u_{xx}(x,t)$$

$$+ \left(0.06\left(1 + texp\left(-e^x\right)\right) - 0.02exp\left(-t - e^x\right)\right.$$

$$\left. - 0.08\left(2 + (1-t)\sin\left(e^x\right)\right)^2\right) u_x(x,t)$$

$$- 0.06\left(1 + texp\left(-e^x\right)\right) u(x,t) \quad (x,t) \in (-2,2) \times (0,1). \tag{28}$$

The initial and boundary conditions are set as:

$$\begin{cases} u(-2,t) = e^{-2-t} \\ u(2,t) = e^{2-t} \\ u(x,0) = e^x \end{cases},$$

the generalized Black-Scholes equation (28) has an exact solution, which is $u(x,t) = e^{x-t}$.

We apply the proposed LNN model to solve this differential equation. Ten hidden layer neurons are set in each direction. One hundred training points are randomly selected in the domain. Fig. 9 plots the exact solution, and Fig. 10 describes the approximate solution. The absolute error in the test points is plotted in Fig. 11. We can find that LNN can obtain an approximation with a maximum error of less than 2×10^{-5}, only a few training points are required.

Besides, we also compared the approximation with other neural network methods, such as Bernstein neural network (BNN) and Radial basis function neural network (RBF). Fig. 12 and 13 shows the absolute errors in test points by BNN and RBF, respectively. The comparison of MSE and MAE of different methods is listed in Table 4. The calculation time is also listed in Table 4.

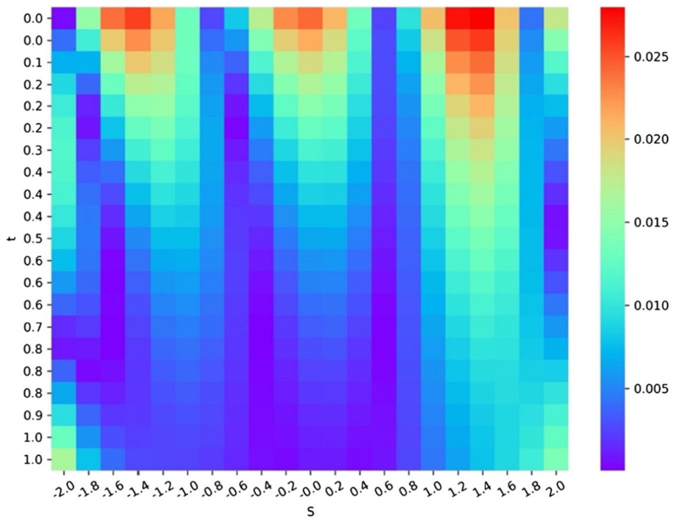


Fig. 12. The absolute error of the approximate solution obtained by the BNN at test points.

Table 4
The comparison of different methods.

Methods	MSE	MAE	Calculation time
LNN	1.16×10^{-11}	1.74×10^{-5}	0.463
BNN	9.11×10^{-5}	2.80×10^{-2}	0.482
RBF	3.10×10^{-6}	6.48×10^{-3}	0.385
CNN	8.13×10^{-8}	1.10×10^{-3}	0.404
HOCFDM	2.31×10^{-7}	7.31×10^{-4}	0.123
B-spline	2.92×10^{-6}	1.39×10^{-3}	0.473
FEM	1.44×10^{-7}	3.97×10^{-3}	0.252

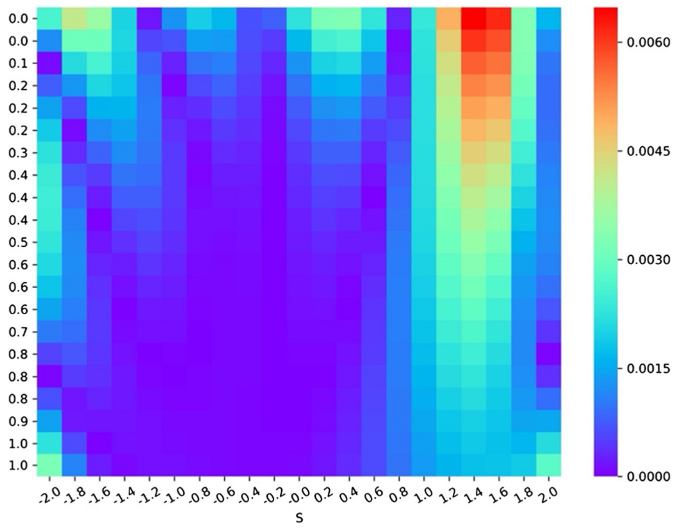


Fig. 13. The absolute error of the approximate solution obtained by the RBF at test points.

6. Conclusion and prospects

In this paper, we present a novel neural network algorithm which is called Laguerre neural network for numerical solving generalized Black-Scholes differential equations. The loss function is constructed by initial and boundary value conditions, also with the form of differential equations. Which is used to measure how well the neural network approximates the exact solution. Modifying the boundary conditions for different option pricing model only needs to change one item of the projection mapping, simplifying the application of the model in the real world problem. The extreme learning machine method is used to minimize the loss function, and only a few equidistant or randomly selected training points are used for training the parameters in the Laguerre neural network. Three numerical experiments show that within an acceptable calculation time, the numerical solution obtained by the proposed LNN method has a smaller mean square error and the maximum absolute error. It can be observed that the LNN solution matches well with the exact solution in most points. The proposed LNN method is suitable for numerical solving BS equations and is efficient and high-precision. Meanwhile, we find that using the neural network method to solve the BS equation can avoid the curse of dimensionality, and numerical solving the high-dimensional BS equation will become our future research direction. In addition, the deeper models are proved to have better representation capabilities, although they are difficult to train. We have considered increasing the depth of neural network to improve the accuracy in some recent works.

7. Code availability

If necessary, you can contact by email chenyinghao@csu.edu.cn.

CRediT authorship contribution statement

All authors contributed to the study conception and design.

Yinghao Chen: Conceptualization, Methodology, Formal analysis, Writing – original;

Hanyu Yu: Conceptualization, Methodology, Visualization;

Xiangyu Meng: Conceptualization, Methodology;

Xiaoliang Xie: Writing – review, Project administration, Funding acquisition;

Muzhou Hou: Conceptualization, Writing – review & editing, Supervision, Funding acquisition;

Julien Chevallie: Writing – review, Validation;

All authors read and approved the final manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and material

Not applicable, all data are giving in this paper.

Acknowledgments

This work was supported by Projects of the National Social Science Foundation of China under Grants No: 19BTJ011.

References

[1] B. Alziary, J.-P. Décamps, P.-F. Koehl, A P.D.E. approach to Asian options: analytical and numerical evidence, J. Bank. Finance 21 (5) (1997) 613–640, https://doi.org/10.1016/S0378-4266(96)00057-X.
[2] P. Amster, C.G. Averbuj, M.C. Mariani, D. Rial, A Black–Scholes option pricing model with transaction costs, J. Math. Anal. Appl. 303 (2) (2005) 688–695, https://doi.org/10.1016/j.jmaa.2004.08.067.
[3] C. Anitescu, E. Atroshchenko, N. Alajlan, T. Rabczuk, Artificial neural network methods for the solution of second order boundary value problems, Comput. Mater. Continua 59 (1) (2019) 345–359, https://doi.org/10.32604/cmc.2019.06641.
[4] J. Ankudinova, M. Ehrhardt, On the numerical solution of nonlinear Black–Scholes equations, Comput. Math. Appl. 56 (3) (2008) 799–812, https://doi.org/10.1016/j.camwa.2008.02.005.
[5] A.J. Arenas, G. González-Parra, B.M. Caraballo, A nonstandard finite difference scheme for a nonlinear Black–Scholes equation, Math. Comput. Model. 57 (7) (2013) 1663–1670, https://doi.org/10.1016/j.mcm.2011.11.009.
[6] F. Black, M. Scholes, The pricing of options and corporate liabilities, J. Polit. Econ. 81 (3) (1973) 637–654, https://doi.org/10.1086/260062.
[7] T. Chan, Pricing contingent claims on stocks driven by Levy processes, Ann. Appl. Probab. 9 (2) (1999) 504–528, Retrieved from <Go to ISI>://WOS:000080974700013.
[8] Y. Chen, X. Xie, T. Zhang, J. Bai, M. Hou, A deep residual compensation extreme learning machine and applications, J. Forecast. (2020), https://doi.org/10.1002/for.2663.
[9] R. Company, A.L. González, L. Jódar, Numerical solution of modified Black–Scholes equation pricing stock options with discrete dividend, Math. Comput. Model. 44 (11) (2006) 1058–1068, https://doi.org/10.1016/j.mcm.2006.03.009.
[10] J.C. Cox, S.A. Ross, M. Rubinstein, Option pricing: a simplified approach, J. Financ. Econ. 7 (3) (1979) 229–263, https://doi.org/10.1016/0304-405X(79)90015-1.

[11] H. Cui, S. Rajagopalan, A.R. Ward, Predicting product return volume using machine learning methods, Eur. J. Oper. Res. 281 (3) (2020) 612–627, https://doi.org/10.1016/j.ejor.2019.05.046.

[12] D. Dufresne, Laguerre series for Asian and other options, Math. Finance 10 (4) (2000) 407–428, https://doi.org/10.1111/1467-9965.00101.

[13] P.A. Forsyth, K.R. Vetzal, R. Zvan, A finite element approach to the pricing of discrete lookbacks with stochastic volatility, Appl. Math. Finance 6 (2) (1999) 87–106, https://doi.org/10.1080/135048699334564.

[14] M.C. Fu, R.W. Wu, G. Gurkan, A.Y. Demir, A note on perturbation analysis estimators for American-style options, Probab. Eng. Inf. Sci. 14 (3) (2000) 385–392, Retrieved from <Go to ISI>://WOS:000087888000008.

[15] F. Godin, V.S. Lai, D.-A. Trottier, Option pricing under regime-switching models: novel approaches removing path-dependence, Insur. Math. Econ. 87 (2019) 130–142, https://doi.org/10.1016/j.insmatheco.2019.04.006.

[16] A.T. Hansen, P.L. Jorgensen, Analytical valuation of American-style Asian options, Manag. Sci. 46 (8) (2000) 1116–1136, https://doi.org/10.1287/mnsc.46.8.1116.12027.

[17] M. Hou, Y. Yang, T. Liu, W. Peng, Forecasting time series with optimal neural networks using multi-objective optimization algorithm based on AICc, Front. Comput. Sci. 12 (6) (2018) 1261–1263, https://doi.org/10.1007/s11704-018-8095-8.

[18] G.-B. Huang, X. Ding, H. Zhou, Optimization method based extreme learning machine for classification, Neurocomputing 74 (1–3) (2010) 155–163, https://doi.org/10.1016/j.neucom.2010.02.019.

[19] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (1–3) (2006) 489–501, https://doi.org/10.1016/j.neucom.2005.12.126.

[20] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: a new learning scheme of feedforward neural networks, in: 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541), vol. 2, Budapest, Hungary, 2004, pp. 985–990, https://doi.org/10.1109/IJCNN.2004.1380068.

[21] H. Huang, X. Gan, L. Lei, Neural Network Prediction of Stock Price Trend Based on RS with Entropy Discretization, 2017.

[22] R.A. Jarrow, A partial differential equation that changed the world, J. Econ. Perspect. 13 (4) (1999) 229–248, https://doi.org/10.1257/jep.13.4.229. In honor of the Nobel laureates Robert C. Merton and Myron S. Scholes.

[23] R. Kangro, R. Nicolaides, Far field boundary conditions for Black-Scholes equations, SIAM J. Numer. Anal. 38 (4) (2000) 1357–1368, https://doi.org/10.1137/s0036142999355921.

[24] M.H.M. Khabir, K.C. Patidar, Spline approximation method to solve an option pricing problem, J. Differ. Equ. Appl. 18 (11) (2012) 1801–1816, https://doi.org/10.1080/10236198.2011.596150.

[25] A. Kim, Y. Yang, S. Lessmann, T. Ma, M.C. Sung, J.E.V. Johnson, Can deep learning predict risky retail investors? A case study in financial risk behavior forecasting, Eur. J. Oper. Res. 283 (1) (2020) 217–234, https://doi.org/10.1016/j.ejor.2019.11.007.

[26] A. Kulaglic, B.B. Üstündağ, Stock price forecast using wavelet transformations in multiple time windows and neural networks, in: 2018 3rd International Conference on Computer Science and Engineering (UBMK), Sarajevo, 2018, pp. 518–521, https://doi.org/10.1109/UBMK.2018.8566614.

[27] K.P. Lam, P.Y. Mok, Stock price prediction using Intraday and AHIPMI data, 2002.

[28] W. Leigh, R. Purvis, J.M. Ragusa, Forecasting the NYSE composite index with technical analysis, pattern recognizer, neural network, and genetic algorithm: a case study in romantic decision support, Decis. Support Syst. 32 (4) (2002) 361–377, https://doi.org/10.1016/s0167-9236(01)00121-x.

[29] Y. Lu, G. Chen, Q. Yin, H. Sun, M. Hou, Solving the ruin probabilities of some risk models with Legendre neural network algorithm, Digit. Signal Process. 99 (2020), https://doi.org/10.1016/j.dsp.2019.102634.

[30] Y. Lu, Q. Yin, H. Li, H. Sun, Y. Yang, M. Hou, The LS-SVM algorithms for boundary value problems of high-order ordinary differential equations, Adv. Differ. Equ. (2019), https://doi.org/10.1186/s13662-019-2131-3.

[31] S. Mall, S. Chakraverty, Single layer Chebyshev neural network model for solving elliptic partial differential equations, Neural Process. Lett. 45 (3) (2017) 825–840, https://doi.org/10.1007/s11063-016-9551-9.

[32] A. Marco, P. Antonio, Dynamic hedging portfolios for derivative securities in the presence of large transaction costs, Appl. Math. Finance 1 (2) (1994) 165–194, https://doi.org/10.1080/13504869400000010.

[33] H. Markowitz, Portfolio selection*, J. Finance 7 (1) (1952) 77–91, https://doi.org/10.1111/j.1540-6261.1952.tb01525.x.

[34] A. Martinez, C. Schmuck, S. Pereverzyev Jr., C. Pirker, M. Haltmeier, A machine learning framework for customer purchase prediction in the non-contractual setting, Eur. J. Oper. Res. 281 (3) (2020) 588–596, https://doi.org/10.1016/j.ejor.2018.04.034.

[35] A.J. Menkveld, T. Vorst, A pricing model for American options with Gaussian interest rates, Ann. Oper. Res. 100 (2000) 211–226, https://doi.org/10.1023/a:1019275302878.

[36] Merton, Rational theory of option pricing, Bell J. Econ. 4 (1973) 141–183, https://doi.org/10.2307/3003143.

[37] Merton, On the pricing of corporate debt: the risk structure of interest rates*, J. Finance 29 (2) (1974) 449–470, https://doi.org/10.1111/j.1540-6261.1974.tb03058.x.

[38] Merton, Option prices when underlying stock returns are discontinuous, J. Financ. Econ. 3 (1976) 125–144, https://doi.org/10.1016/0304-405X(76)90022-2.

[39] K. Michalak, P. Lipinski, Prediction of high increases in stock prices using neural networks, Neural Netw. World 15 (4) (2005) 359–366, Retrieved from <Go to ISI>://WOS:000232532500011.

[40] N. Ozdemir, M. Yavuz, Numerical solution of fractional Black-Scholes equation by using the multivariate Pade approximation, Acta Phys. Pol. A 132 (3) (2017) 1050–1053, https://doi.org/10.12693/APhysPolA.132.1050.

[41] M. Pakdaman, A. Ahmadian, S. Effati, S. Salahshour, D. Baleanu, Solving differential equations of fractional order using an optimization technique based on training artificial neural network, Appl. Math. Comput. 293 (2017) 81–95, https://doi.org/10.1016/j.amc.2016.07.021.

[42] H. Qu, X. Liu, A numerical method for solving fractional differential equations by using neural network, Adv. Math. Phys. 2015 (2015), https://doi.org/10.1155/2015/439526.

[43] A.N. Refenes, A.N. Burgess, Y. Bentz, Neural networks in financial engineering: a study in methodology, IEEE Trans. Neural Netw. 8 (6) (1997) 1222–1267, https://doi.org/10.1109/72.641449.

[44] J.C. Rhim, H. Kim, An estimation of early exercise premium for American put options, Glob. Bus. Finance Rev. 5 (1) (2000) 13–30, Retrieved from <Go to ISI>://KJD:ART002171101.

[45] S. Rikukawa, H. Mori, T. Harada, Recurrent neural network based stock price prediction using multiple stock brands, Int. J. Innov. Comput. Inf. Control 16 (3) (2020) 1093–1099, https://doi.org/10.24507/ijicic.16.03.1093.

[46] F.B. Rizaner, A. Rizaner, Approximate solutions of initial value problems for ordinary differential equations using radial basis function networks, Neural Process. Lett. 48 (2) (2018) 1063–1071, https://doi.org/10.1007/s11063-017-9761-9.

[47] P. Roul, V.M.K. Goura, A high order numerical method and its convergence for time-fractional fourth-order partial differential equations, Appl. Math. Comput. 366 (2019), https://doi.org/10.1016/j.amc.2019.124727.

[48] P. Roul, V.M.K.P. Goura, A new higher order compact finite difference method for generalised Black-Scholes partial differential equation: European call option, J. Comput. Appl. Math. 363 (2020) 464–484, https://doi.org/10.1016/j.cam.2019.06.015.

[49] L. Sen Tan, Z. Zainuddin, P. Ong, Solving ordinary differential equations using neural networks, in: D. Mohamad, A.B. Akbarally, H. Maidinsah, M.M. Jaffar, M. Mohamed, S.R. Sharif, W. Rahman (Eds.), Proceeding of the 25th National Symposium on Mathematical Sciences, vol. 1974, 2018.

[50] S. Simon, M.J. Goovaerts, J. Dhaene, An easy computable upper bound for the price of an arithmetic Asian option, Insur. Math. Econ. 26 (2–3) (2000) 175–183, https://doi.org/10.1016/s0167-6687(99)00051-7.

[51] H. Sun, M. Hou, Y. Yang, T. Zhang, F. Weng, F. Han, Solving partial differential equation based on Bernstein neural network and extreme learning machine algorithm, Neural Process. Lett. 50 (2) (2019) 1153–1172, https://doi.org/10.1007/s11063-018-9911-8.

[52] N. Touzi, American options exercise boundary when the volatility changes randomly, Appl. Math. Optim. 39 (3) (1999) 411–422, https://doi.org/10.1007/s002459900112.

[53] H. Wang, A study on the stock market prediction based on genetic neural network, in: Proceedings of the 2018 International Conference on Information Hiding and Image Processing (IHIP 2018), Association for Computing Machinery, New York, NY, USA, 2018, pp. 105–108, https://doi.org/10.1145/3292425.3292441.

[54] J.-Z. Wang, J.-J. Wang, Z.-G. Zhang, S.-P. Guo, Forecasting stock indices with back propagation neural network, Expert Syst. Appl. 38 (11) (2011) 14346–14355, https://doi.org/10.1016/j.eswa.2011.04.222.

[55] Z. Wang, Y. Meng, F. Weng, Y. Chen, F. Lu, X. Liu, M. Hou, J. Zhang, An effective CNN method for fully automated segmenting subcutaneous and visceral adipose tissue on CT scans, Ann. Biomed. Eng. 48 (1) (2020) 312–328, https://doi.org/10.1007/s10439-019-02349-3.

[56] Z. Wang, Y. Xiao, Y. Li, J. Zhang, F. Lu, M. Hou, X. Liu, Automatically discriminating and localizing COVID-19 from community-acquired pneumonia on chest X-rays, Pattern Recognit. 110 (2021), https://doi.org/10.1016/j.patcog.2020.107613.

[57] B. Wu, T. Duan, A performance comparison of neural networks in forecasting stock price trend, Int. J. Comput. Intell. Syst. 10 (1) (2017) 336–346, https://doi.org/10.2991/ijcis.2017.10.1.23.

[58] L. Xi, M. Hou, M.H. Lee, J. Li, D. Wei, H. Hai, Y. Wu, A new constructive neural network method for noise processing and its application on stock market prediction, Appl. Soft Comput. 15 (2014) 57–66, https://doi.org/10.1016/j.asoc.2013.10.013.

[59] F. Yakuwa, Y. Dote, M. Yoneyama, S. Uzurabashi, Novel time series analysis and prediction of stock trading using fractal theory and time delayed neural network, in: SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483), vol. 1, Washington, DC, USA, 2003, pp. 134–141, https://doi.org/10.1109/ICSMC.2003.1243804.

[60] Y. Yang, M. Hou, J. Luo, A novel improved extreme learning machine algorithm in solving ordinary differential equations by Legendre neural network methods, Adv. Differ. Equ. (2018), https://doi.org/10.1186/s13662-018-1927-x.
[61] Y. Yang, M. Hou, J. Luo, Z. Tian, Numerical solution of several kinds of differential equations using block neural network method with improved extreme learning machine algorithm, J. Intell. Fuzzy Syst. 38 (3) (2020) 3445–3461, https://doi.org/10.3233/jifs-190406.
[62] Y. Yang, M. Hou, H. Sun, T. Zhang, F. Weng, J. Luo, Neural network algorithm based on Legendre improved extreme learning machine for solving elliptic partial differential equations, Soft Comput. 24 (2) (2020) 1083–1096, https://doi.org/10.1007/s00500-019-03944-1.
[63] M. Yavuz, European option pricing models described by fractional operators with classical and generalized Mittag-Leffler kernels, Numer. Methods Partial Differ. Equ. (2020), https://doi.org/10.1002/nurn.22645.
[64] M. Yavuz, N. Ozdemir, A different approach to the European option pricing model with new fractional operator, Math. Model. Nat. Phenom. 13 (1) (2018), https://doi.org/10.1051/mmnp/2018009.
[65] M. Yavuz, N. Ozdemir, European vanilla option pricing model of fractional order without singular kernel, Fractal Fract. 2 (1) (2018), https://doi.org/10.3390/fractalfract2010003.
[66] M. Yavuz, N. Ozdemir, A Quantitative Approach to Fractional Option Pricing Problems with Decomposition Series 6, 2018.
[67] M. Yavuz, N. Ozdemir, Y. Yolcu Okur, Generalized differential transform method for fractional partial differential equation from finance, 2016.
[68] C. Yinghao, C. Yi, X. Xie, M. Hou, Y. Cheng, Solution of ruin probability for continuous time model based on block trigonometric exponential neural network, Symmetry 12 (6) (2020) 876, https://doi.org/10.3390/sym12060876.
[69] T. Zhou, X. Liu, M. Hou, C. Liu, Numerical solution for ruin probability of continuous time model based on neural network algorithm, Neurocomputing 331 (2019) 67–76, https://doi.org/10.1016/j.neucom.2018.08.020.

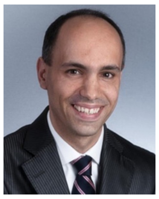
Yinghao Chen received the B.S. degree from Northwest Normal University in 2018. He is currently a Master student in school of Mathematics and Statistics, Central South University, P.R. China. His current research interests include machine learning, deep learning, neural networks, computer vision, financial risks management, quantitative finance and computational intelligence.

Hanyu Yu is currently an undergraduate student majoring in financial management at the Business School of Central South University, P.R. China. Her original major is material chemistry before switching. And her current research interests include quantitative finance, financial risks management, game theory and information economics.

Xiangyu Meng is currently an undergraduate student majoring in statistics at Central South University. Her current research interests include machine learning methods, financial statistics.

Xiaoliang Xie received the M.Sc. degree in mathematics from Hunan Normal University, Changsha, China, in 1998, and the Ph.D. degree in mathematics from Central South University, Changsha, in 2010. He is currently a Professor with the School of Mathematics and Statistics, Hunan Technology and Business University. His research interests include optimization theory, rough set theory, and information systems.

Muzhou Hou received the B.S. degree from Hunan Normal University, Changsha, China, the M.S. degree and the Ph.D. degree from Central South University, Changsha, China, in 1985, 2002 and 2009, respectively. He is currently a Professor and doctoral advisor in School of Mathematics and Statistics, Central South University, China. His current research interests include machine learning, computational intelligence, neural networks, data mining and numerical approximation.



Julien Chevallier is a Professor of Economics at University Paris VIII, Affiliated Professor of Commodity Markets at IPAG Business School (IPAG Lab) and Director of the MSc Money, Banking, Finance & Insurance. He undertakes research and lectures on empirical finance, applied time-series econometrics, and commodity markets. He received his Ph.D. in Economics from the University Paris Nanterre in 2008, and his M.Sc. in Economics from the London School of Economics in 2005. He has previously held visiting research positions at the Imperial College Business School, the Centre for Economic Performance (London School of Economics) and the World Bank (Washington DC).