# Evolutionary Neural Networks for Option Pricing: Multi-Assets Option and Exotic Option

**Yang Li**[1, 2 *], **Zelin Wu**[1], **Feiyang Ye** [1, 3]

[1]Southern University of Science and Technology
[2]National University of Singapore
[3]University of Technology Sydney
e0321295@u.nus.edu, {12012229, 12060007}@mail.sustech.edu.cn

## Abstract

This paper presents a novel framework based on the evolutionary neural network to solve the generalized Black-Scholes equation arising in the financial market efficiently and accurately. We first employ evolutionary neural networks to parameterize the Partial Differential Equations (PDEs) involved in option pricing. This approach allows us to simplify the Black-Scholes PDEs and convert them into the corresponding Ordinary Differential Equations (ODEs). Thus we can use standard ODE solvers such as Euler's method to solve the simplified ODE problem. The proposed framework is flexible and can handle various boundary conditions and terminal conditions, allowing customization based on specific market requirements and scenarios. Moreover, our method offers a reliable and deterministic solution methodology for the pricing framework, as it does not rely on stochastic training. Unlike other approaches that incorporate stochastic elements in their training process, our method eliminates the need for such training and provides consistent results. This deterministic nature enhances the reliability and stability of our approach, making it well-suited for real-world applications in option pricing and financial markets. The experiments on multiple settings are carried out to illustrate the applicability and accuracy of the proposed framework.

## Introduction

In financial markets, options are commonly utilized by investors for arbitrage, speculation or to hedge against investment risks. Consequently, determining the fair price of an option contract, known as option pricing, has emerged as a crucial problem in both theoretical research and practical implementations. The Black-Scholes option pricing model (Black and Scholes 1973), which was used to price European options originally, was the first widely adopted mathematical formula for pricing options. It has a significant influence on modern financial pricing. Some subsequent models (Leland 1985) have been proposed to address its limitations and better align with observed market characteristics. One interesting extension of the Black-Scholes models in option pricing is the modeling of multiple underlying assets, which is commonly referred to as the multi-asset Black-Scholes model (Khodayari and Ranjbar 2018).

*Corresponding author.


Nonetheless, closed-form formulas for option prices, even for European vanilla options, are generally unavailable or difficult to derive under these advanced models, non-homogeneous volatility for instance. Therefore, several numerical methods have been proposed to approximate solutions for option pricing problems. One widely employed method to address the option pricing problems is to solve the corresponding PDEs using finite difference methods (Salvador, Oosterlee, and Van Der Meer 2020; Kim et al. 2020) or finite element methods (Prohl 2019). Another alternative method is the Monte Carlo (MC) simulation method, which offers flexibility and ease of implementation, although it converges slowly to the exact solution and comes at the expense of computational time. In this paper, we mainly study the approach to address option pricing problems by solving the corresponding PDEs.

Although solving PDEs is effective for option pricing, classic PDE solvers often encounter the curse of dimensionality, where the computational cost exponentially increases with the problem's dimensionality (Poggio and Liao 2018). Therefore, developing efficient numerical algorithms for high-dimensional PDEs has been one of the most attractive challenges in option pricing problems.

Machine learning and deep learning have gained significant traction in recent years and have made remarkable strides in diverse scientific domains (Mikalef and Gupta 2021), including text classification, sentiment analysis, image recognition, speech recognition, natural language processing, and computational mathematics, among others. Therefore, with the universal approximation property of deep neural networks (DNNs), it is both intuitive and convenient to utilize DNNs as approximators for solutions to high-dimensional PDEs.

In this paper, we present a novel framework for pricing multi-asset options and exotic options. Our framework leverages the exceptional representational power of DNNs and effectively transforms any PDE model with diverse terminal payoff conditions into an ODE model. Notably, our approach efficiently tackles high-dimensional PDEs while mitigating the curse of dimensionality. It offers a reliable and deterministic methodology, eliminating the need for stochastic training. This enhances the robustness and stability of the pricing framework. The proposed framework is flexible and can handle various boundary conditions and terminal pay-

offs, allowing customization based on specific market requirements and scenarios.

The main contributions of this paper are: **1)** We propose a novel framework that effectively reduces complex PDEs to ODE. This framework allows for a more efficient and tractable solution approach; **2)** The proposed framework is applicable to pricing options with various terminal payoff functions, making it versatile and applicable to a wide range of financial instruments; **3)** We demonstrated the capability of the framework to handle nonlinear PDEs. By leveraging the power of DNNs, our approach is able to effectively tackle exotic and nonlinear option pricing problems.

## Problem Statement

In this section, we introduce the notations used throughout the paper and formulate the problem of pricing options by solving the corresponding PDEs.

In the valuation of a European option contract based on the underlying stock price without dividends, the Black-Scholes PDE plays a crucial role. The corresponding equation can be derived from Itô's Lemma using either a replicating portfolio approach or a martingale approach.

We denote the option price by $v(s,t)$, then the Black-Scholes equation is expressed as follows:

$$\frac{\partial v(s,t)}{\partial t}+\frac{1}{2}\sigma^2 s^2\frac{\partial^2 v(s,t)}{\partial^2 s}+rs\frac{\partial v(s,t)}{\partial s}=rv(s,t). \quad (1)$$

where $t \leq T$ is the time, $\sigma$ represents the constant volatility and $r$ is the risk-free interest rate. Indeed, the Black-Scholes PDE is accompanied by a final condition that represents the specific payoff of the option at the expiration time $T$.

$$v(s,T)=v_T(s_T,K). \quad (2)$$

While the classical Black-Scholes equation has made significant progress, it is important to consider the impact of transaction costs (Soner and Touzi 1999), large investor preferences (Merton 1973), and incomplete markets (Wilmott and Schönbucher 2000). These factors can render the classical option pricing model unrealistic. Consequently, the pricing equations may transform into strongly or fully nonlinear, potentially degenerate, parabolic diffusion-convection equations. In such scenarios, the volatility $\sigma$ can vary with time and depend on the stock price $s$ as well as the derivatives of the option price $v$ itself.

$$\frac{\partial v(s,t)}{\partial t}+\frac{1}{2}\hat{\sigma}^2(t,s,v_s,v_{ss})s^2\frac{\partial^2 v(s,t)}{\partial^2 s}+rs\frac{\partial v}{\partial s}=rv. \quad (3)$$

Furthermore, it is natural and practical to extend equation (1) from pricing single-asset options to pricing multi-asset options, as the latter are more commonly encountered in real-world markets. Let $v\left(s_1,s_2,\ldots,s_n,t\right)$ be the value of the option, where $s_i$ is the underlying $i$-th asset value. We consider the following generalized $n$-asset BS equation:

$$\frac{\partial v(\mathbf{s},t)}{\partial t}+\frac{1}{2}\sum_{i,j=1}^{n}\sigma_i\sigma_j\rho_{ij}s_is_j\frac{\partial^2 v(\mathbf{s},t)}{\partial s_i\partial s_j} + r\sum_{i=1}^{n}s_i\frac{\partial v(\mathbf{s},t)}{\partial s_i}=rv. \quad (4)$$

for $(\mathbf{s},t)=(s_1,s_2,\ldots,s_n,t)\in\mathbf{R}_+^n\times[0,T)$. The corresponding final condition for equation (4) is

$$v(\mathbf{s},T)=v_T(\mathbf{s}_T,K) \quad (5)$$

Here, $r$ is an interest rate, $\sigma_i$ is a constant volatility of the $i$-th asset, and $\rho_{ij}$ is the correlation coefficient between $i$-th and $j$-th underlying assets.

To simplify the notation and unify these PDE problems, we denoted the operator $\mathcal{L}_{BS}$ by

$$\mathcal{L}_{BS}v=rv-\frac{1}{2}\sum_{i,j=1}^{n}\hat{\sigma_i}\hat{\sigma_j}\rho_{ij}s_is_j\frac{\partial^2 v}{\partial s_i\partial s_j}-r\sum_{i=1}^{n}s_i\frac{\partial v}{\partial s_i}. \quad (6)$$

Then the PDE problem (1), (3) and (4) can be formulated in an united form:

$$\frac{\partial v(\mathbf{s},t)}{\partial t}=\mathcal{L}_{BS}v(\mathbf{s},t),\ \ v(\mathbf{s},T)=v_T(\mathbf{s}_T,K). \quad (7)$$

Therefore, the pricing of options can be framed as the task of solving those nonlinear PDEs (7).

## Methodology

In this section, we present the detailed methodology of the proposed framework. We consider a spatial domain $\mathcal{S}\subseteq\mathcal{R}^d$, then Equation (7) can be expressed as the evolution of a time-dependent field $v:\mathcal{S}\times[0,T]\rightarrow\mathcal{R}$. The field $v$ belongs to a function space $\mathcal{V}$ at all times, and its dynamics are governed by the equation

$$\frac{\partial v(\mathbf{s},t)}{\partial t}=f(t,\mathbf{s},v,v_{s_i},v_{s_is_j}),\quad(\mathbf{s},t)\in\mathcal{S}\times[0,T], \quad (8)$$

subject to the terminal condition (5), where $v_T\in\mathcal{V}$. By appropriately choosing the function $f$, equation (8) can represent different equations of interest. We assume that suitable boundary conditions exist for Equation (8), ensuring that it is well-posed for all $t\in[0,T]$. This implies that the solution to the equation exists, is unique, and depends continuously on the terminal condition $v_T\in\mathcal{V}$.

### Parametrizing the Solution

To proceed, we adopt a parametric representation of the solution $v(\mathbf{s},t)$ at time $t$ as $V(\mathbf{s},\boldsymbol{\theta}(t))\in\mathcal{V}$, where $\boldsymbol{\theta}(t)\in\boldsymbol{\Theta}$ are the parameters, and $V:\mathcal{V}\times\boldsymbol{\Theta}\rightarrow\mathcal{R}$. This is achieved by utilizing the ansatz:

$$v(\mathbf{s},t)=V(\mathbf{s},\boldsymbol{\theta}(t)),\quad(\mathbf{s},\boldsymbol{\theta}(t))\in\mathcal{S}\times\boldsymbol{\Theta}. \quad (9)$$

It is important to highlight that the dependence of $V$ on $\boldsymbol{\theta}(t)$ can be nonlinear, which stands in contrast to the linear dependence found in the majority of classical approximations in scientific computing. For the representation in Equation (9) to be complete, ensuring that for any $v(\mathbf{s},t)\in\mathcal{V}$ there exists at least one $\boldsymbol{\theta}(t)\in\boldsymbol{\Theta}$ such that $V(\mathbf{s},\boldsymbol{\theta}(t))\equiv v(\mathbf{s},t)$, the parameter space $\boldsymbol{\Theta}$ is typically an infinite-dimensional function space.

In practical applications, we are often interested in situations where the parametric representation $V(\mathbf{s},\boldsymbol{\theta}(t))$ depends on a finite-dimensional parameter $\boldsymbol{\theta}(t)$. For example,

in a DNN, $\boldsymbol{\theta}(t)$ represents the vector of adjustable parameters or weights in the network. Although the number of these parameters can be large, it is important to note that they are still finite in number.

In this case, the use of Eq. (9) with DNN approximation as an ansatz for the solution of Eq. (8) introduces approximation errors. It is important to be aware of these errors and aim to control their magnitude throughout the solution process.

## Evolutionary Neural Networks

An evolutionary neural network (ENN) (Du and Zaki 2021) is a mathematical mapping that takes an input vector $\boldsymbol{x} \in \mathcal{R}^d$ and transforms it into an output vector $\boldsymbol{z} \in \mathcal{R}^m$ at time $t$. This transformation is achieved by composing a series of vector-valued functions known as layers. In an L-layer network, there are L such layers, each consisting of $\beta_l$ neurons.

The output of an L-layer network can be expressed as $\boldsymbol{z} = ENN(\boldsymbol{x}; \boldsymbol{\theta}(t))$, where

$$ENN(\boldsymbol{x}; \boldsymbol{\theta}(t)) := \boldsymbol{h_L}\left(\cdot, \boldsymbol{\theta}^L(t)\right) \circ \cdots \circ \boldsymbol{h_1}\left(\cdot, \boldsymbol{\theta}^1(t)\right)(\boldsymbol{x}). \tag{10}$$

For the $l$-th layer, where $1 \leq l \leq L$, we have

$$\begin{aligned} &\boldsymbol{h_l}\left(\boldsymbol{z_l}; \theta^l(t)\right) = \sigma_l\left(\boldsymbol{W_l}(t)\boldsymbol{z_l}(t) + \boldsymbol{b_l}(t)\right), \\ &\boldsymbol{W_l}(t) \in \mathcal{R}^{\beta_{l+1} \times \beta_l}, \boldsymbol{z_l}(t) \in \mathcal{R}^{\beta_l}, b_l(t) \in \mathcal{R}^{\beta_{l+1}}, \end{aligned}$$

with $\boldsymbol{z_1} = \boldsymbol{x}, \beta_1 = d$ ,$\beta_L = m$ and $\sigma_l$ is activation functions.

## Controlling the Dynamics of $\boldsymbol{\theta}(t)$

To control the approximation error in the ansatz solution, we introduce the residual function $r_t(\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{s})$, defined in Equation (11), which quantifies the discrepancy between the left-hand side and the right-hand side of the Equation (9). This residual function depends on the parameters $\boldsymbol{\theta}$, the time derivative $\boldsymbol{\eta} = \dot{\boldsymbol{\theta}}(t)$, and the spatial variable $\mathbf{s}$.

$$r_t(\boldsymbol{\theta}, \boldsymbol{\eta}, \mathbf{s}) := |\nabla_{\boldsymbol{\theta}} V(\mathbf{s}, \boldsymbol{\theta}(t)) \cdot \boldsymbol{\eta} - f(t, \mathbf{s}, V(\mathbf{s}, \boldsymbol{\theta}(t)))|^2. \tag{11}$$

To minimize the approximation error, we aim to find the time derivative $\dot{\boldsymbol{\theta}}(t)$ such that for all $t > 0$. It satisfies the minimization problem:

$$\dot{\boldsymbol{\theta}}(t) \in \arg\min_{\boldsymbol{\eta}} L_t(\boldsymbol{\theta}(t), \boldsymbol{\eta}), \tag{12}$$

where $L_t(\boldsymbol{\theta}, \boldsymbol{\eta})$ is the objective function defined as:

$$L_t(\boldsymbol{\theta}, \boldsymbol{\eta}) = \frac{1}{2}\int_{\mathcal{S}} r_t(\boldsymbol{\theta}, \boldsymbol{\eta}, \boldsymbol{s}) d\mathbf{s}. \tag{13}$$

By solving the minimization problem for each time step $t$, we can iteratively update the parameters $\boldsymbol{\theta}(t)$, leading to an approximation dynamics for the original PDE (i.e. Eq. (8)) solution.

Fortunately, the minimization problem (12) has an explicit form, which leads to solving the corresponding Euler-Lagrange equation (14):

$$\nabla_{\boldsymbol{\eta}} L_t(\boldsymbol{\theta}(t), \dot{\boldsymbol{\theta}}(t)) = 0. \tag{14}$$

Algorithm 1: ENN-PDE Solver

**Require:** The PDE (8), terminal condition (5) and regularity term $\lambda$.

1: Set $t_k, k = 0, \ldots, N, t_0 = 0, t_N = T, \theta_k = \theta(t_k)$.
2: Train $\boldsymbol{\theta}_N$ by equation (18).
3: **while** $k \geq 0$ **do**
4: Calculate the coefficient $\mathbf{C}(\boldsymbol{\theta}_{k+1})$ and $\mathbf{b}(\boldsymbol{\theta}_{k+1})$.
5: Update $\boldsymbol{\theta}_k$ explicitly by equation

$$(\mathbf{C}(\boldsymbol{\theta}_{k+1}) + \lambda I)\frac{\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k}{\tau_k} = \mathbf{b}(\boldsymbol{\theta}_{k+1}).$$

6: **end while**
7: **return** Discretized solution $V(\mathbf{s}, \boldsymbol{\theta}_k), k = 0, ..., N$.

In explicit form, equation (14) becomes a system of ordinary differential equations (ODEs) for $\boldsymbol{\theta}(t)$:

$$\mathbf{C}(\boldsymbol{\theta}(t))\dot{\boldsymbol{\theta}} = \mathbf{b}(\boldsymbol{\theta}(t)), \quad \boldsymbol{\theta}(T) = \boldsymbol{\theta}_T, \tag{15}$$

where we define:

$$\mathbf{C}(\boldsymbol{\theta}) = \int_{\mathcal{S}} (\nabla_{\boldsymbol{\theta}} V(\mathbf{s}, \boldsymbol{\theta}(t)))^T \nabla_{\boldsymbol{\theta}} V(\mathbf{s}, \boldsymbol{\theta}(t)) d\mathbf{s}, \tag{16}$$

$$\mathbf{b}(\boldsymbol{\theta}) = \int_{\mathcal{S}} \nabla_{\boldsymbol{\theta}} V(\mathbf{s}, \boldsymbol{\theta}(t)) f(\mathbf{s}, V(\mathbf{s}, \boldsymbol{\theta}(t))) d\mathbf{s}. \tag{17}$$

The terminal condition $\boldsymbol{\theta}_T$ can be obtained, for example, by minimizing the least-squares loss between $v_T(\mathbf{s})$ and $V(\mathbf{s}, \boldsymbol{\theta})$:

$$\boldsymbol{\theta}_T \in \arg\min \int_{\mathcal{S}} |v_T(\mathbf{s}) - V(\mathbf{s}, \boldsymbol{\theta})|^2 d\mathbf{s}. \tag{18}$$

Similarly, boundary conditions can be enforced naturally by choosing $V(\mathbf{s}, \boldsymbol{\theta}(t))$ such that these conditions are satisfied for all $\boldsymbol{\theta}(t) \in \Theta$. In summary, the minimization problem leads to solving the system of ODEs (15) for $\boldsymbol{\theta}(t)$, which is determined by the matrix $\mathbf{C}(\boldsymbol{\theta})$ and the vector $\mathbf{b}(\boldsymbol{\theta})$. The terminal condition $\boldsymbol{\theta}_T$ can be obtained through minimization and boundary conditions can be enforced by choosing appropriate $V(\mathbf{s}, \boldsymbol{\theta}(t))$. Solving this system allows us to find the optimal parameter trajectory $\boldsymbol{\theta}(t)$ that minimizes the approximation error and satisfies the given conditions.

To numerically approximate the system of ODEs in equation (15), we employed the explicit Euler scheme to update $\boldsymbol{\theta}_k$ forwardly. Let $\boldsymbol{\theta}_k$ denote the numerical approximation to $\boldsymbol{\theta}(t_k)$. The times $t_k$ are defined recursively as $t_0 = 0, t_N = T$, and subsequent time points are obtained by adding the time step $\tau_k$ adaptively. The time steps, denoted by $\tau_k > 0$, can be chosen to be non-uniform.

$$(\mathbf{C}(\boldsymbol{\theta}_{k+1}) + \lambda I)\frac{\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k}{\tau_k} = \mathbf{b}(\boldsymbol{\theta}_{k+1}), \tag{19}$$

where $\lambda I$ is a diagonal matrix added to ensure the invertibility of the coefficient matrix. In addition to the explicit Euler scheme, other numerical schemes such as high-order schemes and implicit schemes can be employed. The entire algorithm to solve problem (8) is shown in Algorithm 1.

## Related Work

To solve equation (7), one notable approach called DeepBSDE numerical method was proposed by (E, Han, and Jentzen 2017). It tackles the difficulties arising from the backward characteristic of equation (6). DeepBSDEs have shown promise in overcoming the curse of dimensionality associated with certain types of PDEs. However, their application has been primarily limited to numerical experiments involving the Black-Scholes equation with constant coefficients, which is different from our settings.

The Physics-Informed Neural Networks (PINNs) (Raissi, Perdikaris, and Karniadakis 2019) have gained significant attention in recent years for their remarkable performance in solving PDEs. Recently, (Tanios 2021) successfully applied PINNs to directly solve Black-Scholes equations, and (P. Villarino, Leitao, and García Rodríguez 2023) introduced improvements to the methodology by addressing boundary conditions. Furthermore, (Zang et al. 2020) presented the WAN method, which provides an approach to solve weak solutions of PDEs. Nonetheless, these methods encounter challenges in effectively training the neural networks and selecting appropriate hyperparameters.

## Experiments

In this section, we present an evaluation of the ENN-PDE solver on various options, including the vanilla European call option, European call option with transaction cost, down-and-out barrier option, and exchange option on multiple assets.

**Implementation Details.** To facilitate the computations, we transform the original infinite domain into a finite domain by restricting $\mathbf{s}$ up to $\mathbf{s}_{\max}$. Additionally, we rescale the domain to $[0,1]^d$ without loss of generality. We set $\mathbf{s} \in [0,1]^d$, $K = 0.4$, $r = 0.05$, $\sigma = 0.25$,$T = 1$. The neural network architecture used in our approach is fully connected, comprising 6 hidden layers with 60 neurons per layer. We apply a tanh activation function to enhance the network's representational capacity. For all the experiments conducted, we utilize the Monte Carlo method to approximate the integrals involved in the computations.

### Case 1: Vanilla European Call Option

The vanilla European call option with a terminal payoff $v(s,T) = (s_T - K)^+$ has an analytic solution:

$$v_c(s,t) = sN(d_1) - Ke^{-r\tau}N(d_2), \tag{20}$$

where $d_1 = \frac{\log(s/K)+\left(r+0.5\sigma^2\right)\tau}{\sigma\sqrt{\tau}}$, $d_2 = d_1 - \sigma\sqrt{\tau}$, $\tau := T - t$, and $N(\cdot)$ represents the cumulative distribution function (c.d.f.) of the standard normal distribution.

The ENN ansatz can be expressed as equation (21) to impose the one-side boundary condition $V(0, \boldsymbol{\theta}(t)) = 0$.

$$V(s, \boldsymbol{\theta}(t)) = s \cdot ENN(s; \boldsymbol{\theta}(t)). \tag{21}$$

In our experiment, we set $\tau_k = 10^{-3}, \lambda = 10^{-7}$. The results are shown in Figure 1. It shows that the ENN-PDE solver effectively and accurately approximated the analytical solution of the vanilla European call option. Furthermore, as time progressed, the ENN-PDE showed strong convergence towards the steady state of the analytical solution, further validating the accuracy and efficacy of our proposed method.

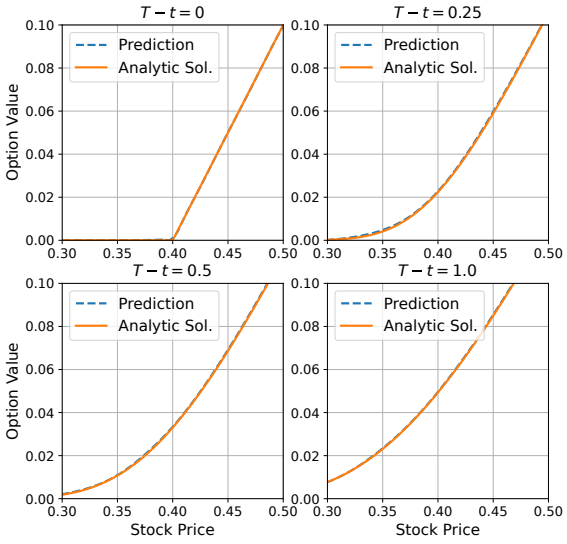


Figure 1. Vanilla European Call Option: compare ENN-PDE predictions and analytic solutions at four timestamps.

### Case 2: Call Option with Transaction Cost

For the European call option with transaction cost, we consider a Black-Scholes equation (3) with the modified volatility (Leland 1985) to model the influence of transaction costs.

$$\hat{\sigma}^2 = \sigma^2\left(1 + \sqrt{\frac{2}{\pi}}\frac{\kappa}{\sigma\sqrt{\delta_t}}sign(v_{ss})\right), \tag{22}$$

where $v_{ss}$ is the second derivative of $v$, $\kappa$ stands for the round trip transaction cost per unit and $\delta_t$ is the transaction frequency.

In the experiment, the ENN ansatz is the same as equation (21), and we set $\kappa = 0.05$, $\delta_t = 0.01$, $\lambda = 10^{-7}$, $\tau_k = 10^{-3}$. Figure 2 shows the excellent approximation of the reference solution by the ENN-PDE solver indicating that our solver provides a robust and reliable approach for pricing options in realistic market conditions.

### Case 3: Down-and-Out Call Option

A European-style down-and-out call option is a financial derivative that has a similar payoff structure to a regular European call option but with an additional barrier condition. This option pays out the standard call payoff $(S_T - K)^+$ at its expiration, but only if the underlying asset price $S_t$ (where $t \leq T$) has not fallen to or below a predetermined barrier level $B$ during the option's lifetime. If at any point before expiration, the underlying asset price reaches or goes below the barrier level of $B$, the option becomes worthless.

In this case, the ENN ansatz can be expressed as equation (23) to impose the one-side boundary condition

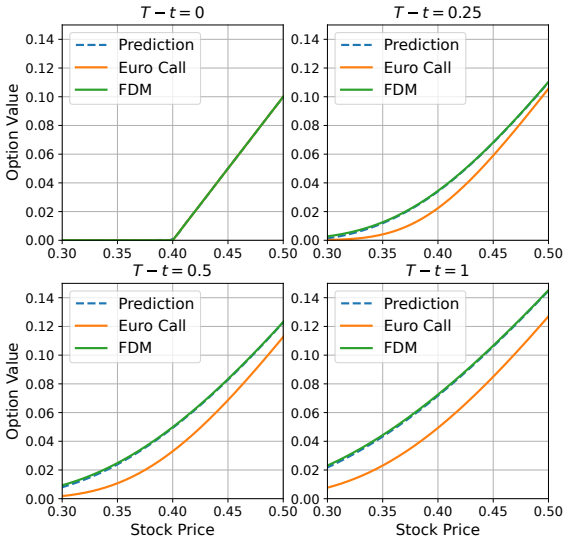


Figure 2. European Call Option with Transaction Cost: compare ENN-PDE predictions with reference solutions (finite difference method with $\Delta x = 0.0001$ and $\Delta t = 0.001$) at four timestamps.

$V(B, \boldsymbol{\theta}(t)) = 0.$

$$V(s, \boldsymbol{\theta}(t)) = (s - B) \cdot ENN(s; \boldsymbol{\theta}(t)). \tag{23}$$

helloThe analytical solution of this down-and-out call option is given by the following equation

$$v(s,t) = v_c(s,t) - (\frac{B}{s})^{\frac{2r}{\sigma^2}-1} v_c(\frac{B^2}{s}, t), \tag{24}$$

where $v_c$ is defined in equation (20), the analytic solution of vanilla European call option, and $B$ is less than the strike price $K$.

In our experiment, we set $B = 0.35$, $\lambda = 10^{-7}$, $\tau_k = 10^{-3}$. Figure 3 demonstrates the consistency between the ENN-PDE solver and the analytical solution, highlighting the power of our framework in pricing exotic options. The close agreement between the two solutions validates the accuracy and reliability of our approach in tackling exotic option pricing problems.

### Case 4: Multi-Assets Option Pricing

In addition to pricing options on a single underlying asset, our approach extends to pricing multi-asset options. Specifically, we focus on the case of two underlying assets, formulating the problem as a PDE problem (4) with $n = 2$. By specifying different payoff functions (i.e. Eq. (25)), various options can be defined, such as exchange options, rainbow options, or average put options.

We specifically address the exchange option, for which an analytic solution known as Margrabe's formula exists. The payoff function for an exchange option is given by:

$$v(s_1(T), s_2(T)) = (s_1(T) - s_2(T))^+. \tag{25}$$

According to Margrabe's formula (Margrabe 1978), the fair value of a European exchange option at a time can be expressed as

$$v(s_1, s_2, t) = s_1(t)N(d_1) - s_2(t)N(d_2), \tag{26}$$

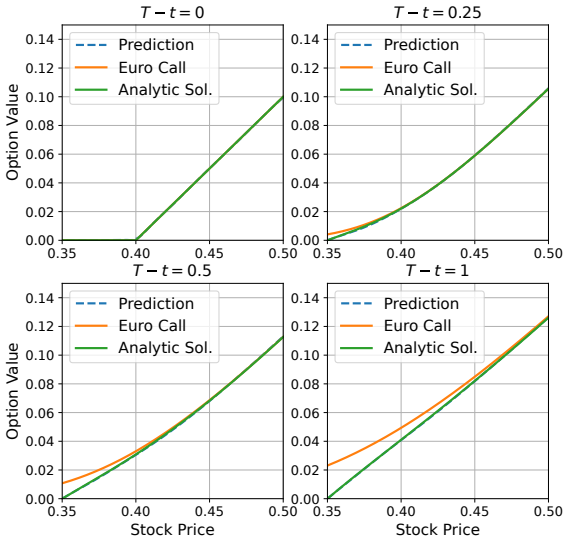


Figure 3. Down-and-Out Call Option: compare ENN-PDE predictions and analytic solutions at four timestamps.

where $d_1 = \frac{(2\log(s_1(t)/s_2(t)) + \sigma^2)(T-t)}{2\sigma\sqrt{T-t}}$, $d_2 = d_1 - \sigma\sqrt{T-t}$, and $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2 - 2\sigma_1\sigma_2\rho}$.

In this case, the ENN ansatz can be expressed as

$$V(s_1, s_2, \boldsymbol{\theta}(t)) = ENN(s_1, s_2; \boldsymbol{\theta}(t)). \tag{27}$$

We set $r = 0.05, \sigma_1 = 0.25, \sigma_2 = 0.20, \rho = 0.1$, the evolving parameters are $\tau_k = 10^{-3}$, $\lambda = 10^{-8}$ and the results are shown in Figure 4. Our method demonstrates exceptional approximation of the analytic solution, even in the presence of non-smooth terminal payoffs such as the exchange option. Although the non-smoothness of the terminal payoff introduces initial approximation errors, our ENN-PDE solver does not amplify these errors as time evolves. This characteristic highlights the stability and robustness of our approach, allowing for accurate and reliable pricing of options over time. The ability to mitigate and control approximation errors is a significant advantage of our method, as it ensures the accuracy and consistency of the pricing framework.

## Conclusion

Our paper introduces a novel framework for pricing options using evolutionary neural networks. We demonstrate the effectiveness of our approach in addressing exotic options, options with transaction costs, and multi-asset option pricing. A key advantage of our framework is its deterministic and reliable solution methodology, eliminating the need for stochastic training. This enhances the robustness and stability of the pricing framework, making it suitable for real-world applications. Additionally, our framework provides flexibility in handling various boundary conditions and terminal payoffs, allowing customization for specific market requirements. This adaptability makes it a versatile tool for option pricing in different market contexts. In conclusion, our approach offers a powerful and practical solution for option pricing by combining evolutionary neural networks and partial differential equations. Through empirical experiments, we demonstrate its effectiveness and applicability.

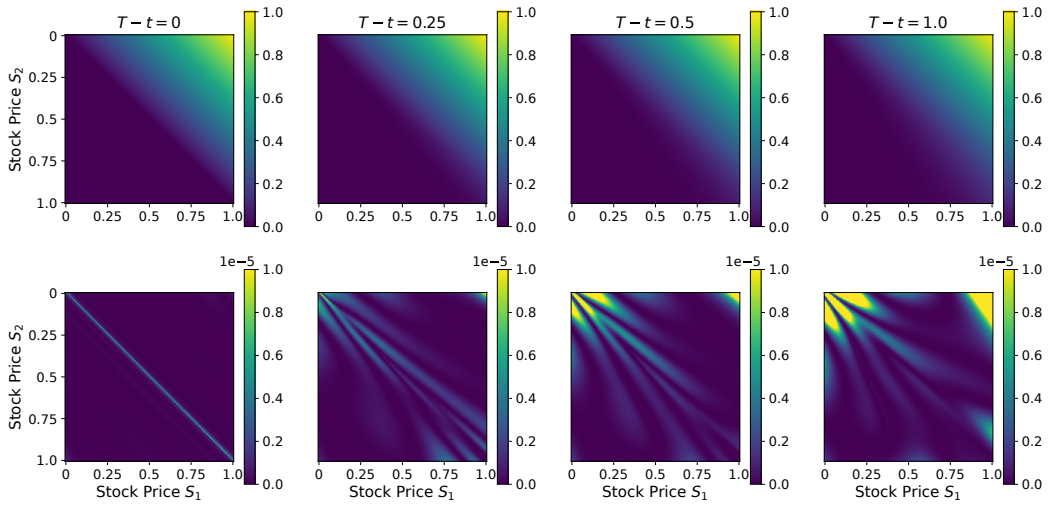


Figure 4. Exchange Option: compare ENN-PDE predictions and analytic solutions at four timestamps. Up row: NN predictions; Down row: Error between NN predictions and analytic solution $|V(s_1, s_2, \boldsymbol{\theta}(t)) - v(s_1, s_2, t)|^2$.

We believe our work opens up new possibilities for pricing complex options and provides valuable insights for the financial industry. Future research directions could involve exploring the application of our framework to other path-dependent derivative instruments and extending it to handle coupled stochastic pricing models. These advancements would further contribute to the field of option pricing and expand the scope of our framework's applicability.

## References


Black, F.; and Scholes, M. 1973. The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81(3): 637.

Du, Y.; and Zaki, T. A. 2021. Evolutional deep neural network. *Physical Review E*, 104(4): 045303.

E, W.; Han, J.; and Jentzen, A. 2017. Deep Learning-Based Numerical Methods for High-Dimensional Parabolic Partial Differential Equations and Backward Stochastic Differential Equations. *Communications in Mathematics and Statistics*, 5(4): 349–380.

Khodayari, L.; and Ranjbar, M. 2018. A computationally efficient numerical approach for multi-asset option pricing. *International Journal of Computer Mathematics*, 96: 1–13.

Kim, S.; Jeong, D.; Lee, C.; and Kim, J. 2020. Finite Difference Method for the Multi-Asset Black–Scholes Equations. *Mathematics*, 8(3): 391.

Leland, H. 1985. Option Pricing and Replication with Transactions Costs. *Journal of Finance*, 40(5): 1283–1301.

Margrabe, W. 1978. The Value of an Option to Exchange One Asset for Another. *The Journal of Finance*, 33(1): 177.

Merton, R. 1973. The Theory of Rational Option Pricing. *Bell J Econ Manage Sci*, 4: 141–183.

Mikalef, P.; and Gupta, M. 2021. Artificial intelligence capability: Conceptualization, measurement calibration, and empirical study on its impact on organizational creativity and firm performance. *Information Management*, 58(3): 103434.

P. Villarino, J.; Leitao, ; and García Rodríguez, J. 2023. Boundary-safe PINNs extension: Application to non-linear parabolic PDEs in counterparty credit risk. *Journal of Computational and Applied Mathematics*, 425: 115041.

Poggio, T.; and Liao, Q. 2018. Theory I: Deep networks and the curse of dimensionality. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, 66.

Prohl, S. 2019. Finite Element Methods for Partial Differential Equations for Option Pricing. *SSRN Electronic Journal*.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378: 686–707.

Salvador, B.; Oosterlee, C. W.; and Van Der Meer, R. 2020. Financial Option Valuation by Unsupervised Learning with Artificial Neural Networks. *Mathematics*, 9(1): 46.

Soner, H.; and Touzi, N. 1999. Super-Replication Under Gamma Constraints. *Université Panthéon-Sorbonne (Paris 1), Papiers d'Economie Mathématique et Applications*.

Tanios, R. 2021. Physics Informed Neural Networks in Computational Finance: High Dimensional Forward & Inverse Option Pricing. 59 p. Artwork Size: 59 p. Medium: application/pdf Publisher: ETH Zurich.

Wilmott, P.; and Schönbucher, P. 2000. The Feedback Effect of Hedging in Illiquid Markets. *SIAM Journal of Applied Mathematics*, 61: 232–272.

Zang, Y.; Bao, G.; Ye, X.; and Zhou, H. 2020. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411: 109409.