Sungwon PARK, PhD Candidate
sungwpark@korea.ac.kr
Korea University, Seoul, South Korea

Kyoung-Sook MOON, PhD (corresponding author)
ksmoon@gachon.ac.kr
Gachon University, Gyeonggi, South Korea

Hongjoong KIM, PhD (corresponding author)
hongjoong@korea.ac.kr
Korea University, Seoul, South Korea

Asian Option Pricing Using the Physics-Informed Neural Networks Method

Abstract. *Accurately calculating the prices of Asian options is challenging due to their path-dependent characteristics and the high-dimensional nature of the problem. This study addresses this issue using a novel Physics-Informed Neural Network (PINN) approach, which leverages the strengths of both neural networks and partial differential equation methods. By applying this PINN method to the pricing problems of one-asset and two-asset Asian options, we demonstrate that it can efficiently produce accurate price estimates compared to the traditional Monte Carlo method.*

Keywords: *Asian option, Physics-informed neural network, Meshless method, Path-dependent option pricing.*

JEL Classification: C15, C45, G13.

1. Introduction

We introduce an effective approach for pricing Asian options utilising the Physics-Informed Neural Networks (PINNs) methodology. An Asian option is a type of financial derivative where the payoff is determined by the average price of the underlying asset over a specific period, rather than its price at a single point in time. This averaging mechanism reduces the option's sensitivity to short-term volatility and market manipulation. Asian options are typically used in markets with high volatility or where price manipulation is a concern, providing a more stable and predictable hedging instrument. They are popular in commodities and foreign exchange markets, helping investors manage risks associated with fluctuating prices over time (Wilmott et al., 1995; Kwok, 2008).

However, due to the path-dependent nature of these options, determining the price of Asian options is challenging. Even within the framework of the Black–Scholes model, a closed-form solution for the price of an arithmetic Asian option does not exist. Consequently, a variety of methods have been proposed in the

DOI: 10.24818/18423264/59.1.25.01

literature to address this challenge. Among these methods, Monte Carlo simulation (Glasserman, 2004; Shiraya and Takahashi, 2017), methods based on partial differential equations (Shi and Yang, 2014; Vecer, 2014), and expansion techniques (Lo et al., 2014; Lin and Chang, 2020) are widely employed.

With the advancement of computational capabilities driven by big data, numerous machine learning and deep learning techniques have demonstrated significant success in addressing challenges across various domains, including the finance sector (Fischer and Krauss, 2018; Sezer et al., 2020; Goodell et al., 2021; Nabipour et al., 2020; Moon and Kim, 2019, 2023). Recently, PINNs have emerged, leveraging the strengths of traditional neural networks, particularly their capacity to approximate complex functions, by incorporating physical laws directly into the training process (Psaros et al., 2023; Cuomo et al., 2022). This approach has also been applied to financial option pricing problems, demonstrating performance improvements (Bai et al., 2022; Gatta et al., 2023; Wang et al., 2023).

In this paper, we apply PINNs to the problem of pricing Asian options. Our approach aims to overcome the limitations of the Black-Scholes partial differential equation, particularly in addressing the path-dependent nature of averaging options and high-dimensional problems, thereby enhancing both the accuracy and computational efficiency of the pricing process. In Section 4, we demonstrate the efficiency of our method based on PINNs through examples involving both single-asset and multi-asset cases. As shown in Figures 3 and 5, when PINNs are used, Asian option prices can be calculated more accurately and efficiently than with the traditional Monte Carlo (MC) method.

The following sections of this paper are structured as follows. Section 2 introduces the problem of pricing Asian options. Section 3 details the PINN methodology, including the structure of neural networks and its application to pricing issues. In Section 4, we compare the proposed method with the MC results by applying it to examples of single-asset and multi-asset Asian options. Section 5 concludes the paper with insights and future research directions.

2. Problem description

In this work, we consider the pricing of continuously monitored Asian options. Let us assume that the asset price, $S(t)$, follows the stochastic differential equation:

$$dS(t) = \mu S(t)dt + \sigma S(t)dW(t) \tag{1}$$

where $W(t)$ is the standard Brownian motion process, and μ and σ represent the expected rate of return and volatility of the asset, respectively. Asian options are path-dependent derivatives whose payoffs depend on the average of the underlying asset prices over a predefined period. Under the assumption of a frictionless market, the value of a claim contingent on the stock at time t can be represented as

$$V(S(t), I(t), t) = e^{-r(T-t)}E^*[g(S(T), I(T), T) \mid F_t] \tag{2}$$

where $g(S(T), I(T), T)$ represents the payoff function at T and $E^*[\cdot]$ denotes the expectation conditioned on the information available at time t, represented by the filtration F_t with risk neutral assumption $\mu = r$ in (1). r is the risk-free interest rate and $I(t) = \int_0^t S(\tau)d\tau$. Then $A(t) = \frac{1}{t}I(t)$ represents the continuous average of the asset price. Applying the no-arbitrage principle, we derive the governing partial differential equation (PDE) for the price, V, of the one-asset Asian option:

$$\mathcal{L}[V] \equiv \frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} + S\frac{\partial V}{\partial I} - rV = 0\,. \tag{3}$$

The terminal payoff at maturity, T, is $g(S(T), I(T), T) = \left(\frac{1}{T}I(T) - K\right)^+$ for the fixed strike call option with the strike price K, and $g(S(T), I(T), T) = \left(S(T) - \frac{1}{T}I(T)\right)^+$ for the floating strike call option, where $(\cdot)^+ = \max(\cdot, 0)$. The boundary conditions are approximated by

$$V(S(t), I(t), t) \approx \left(e^{-r(T-t)}\left(\frac{I(t)}{T} - K\right) + \frac{S(t)}{rT}\left(1 - e^{-r(T-t)}\right)\right)^+,$$

see in (Barraquand and Pudet,1996).

For the Asian basket option based on two assets, $S_1(t)$ and $S_2(t)$ following $dS_i(t) = rS_i(t)dt + \sigma S_i(t)\, dW_i(t),\ i = 1, 2$, with the correlation ρ between two assets, the governing PDE becomes

$$\begin{aligned}\mathcal{L}[V] \equiv \frac{\partial V}{\partial t} + \frac{1}{2}\sigma_1^2 S_1^2 \frac{\partial^2 V}{\partial S_1^2} + \frac{1}{2}\sigma_2^2 S_2^2 \frac{\partial^2 V}{\partial S_2^2} + \rho\,\sigma_1\sigma_2 S_1 S_2 \frac{\partial^2 V}{\partial S_1 \partial S_2} + rS_1\frac{\partial V}{\partial S_1} \\ + rS_2\frac{\partial V}{\partial S_2} + \frac{S_1 + S_2}{2}\frac{\partial V}{\partial I} - rV = 0\end{aligned} \tag{4}$$

where $I(t) = \frac{1}{2}\int_0^t S_1(\tau) + S_2(\tau)\, d\tau$.

3. Computational methods

We are interested in solving the following problem: *Given a set of known data, what can be inferred about the solution, V, of the PDE in (3) or (4) governing the pricing of Asian options based on one or two underlying assets, respectively*? Traditionally, the field of scientific computing has developed robust theoretical frameworks and many numerical algorithms to derive exact or approximate solutions for such problems. In this study, we apply the approach based on PINNs to determine the value of European Asian call options and compare the results with those obtained from the traditional MC method.

3.1 Monte Carlo method

For the value $V(S(t), I(t), t)$ in (2) of the European Asian option for one asset with the payoff function $g(S(T), I(T), T)$, the MC method generates M sample paths $\{S_j(t_i), i = 1, \dots, N\}_{j=1}^{M}$ over N time points, $0 = t_1 \leq t_2 \leq \cdots \leq t_N = T$ and estimates the expectation in (2) by replacing the continuous average $A(T) = \int_0^T S(\tau)d\tau \,/\, T$ with an arithmetic average $\frac{1}{N}\sum_{i=1}^{N} S(t_i)$ of the samples.

The MC method is flexible and can be easily applied to various financial problems; however, it also has weaknesses. The MC method is heavily dependent on the quality of the input data, and the computational intensity is another drawback due to the requirement of generating a large number of sample paths. In fact, by the Central Limit Theorem, it can be shown that

$$V^{true} - V_M^{MC} = O\left(\frac{1}{\sqrt{M}}\right)$$

where V^{true} is the true (unknown) solution and V_M^{MC} is an approximate solution from the MC method with M sample paths.

3.2 Neural Networks and PINNs

A neural network (NN) is a computational model inspired by biological neural networks in the human brain, composed of interconnected nodes (neurons) and weighted connections (synapses). As illustrated in Figure 1, each node processes the input signals, applies a nonlinear transformation to the weighted sum of its inputs, and passes the output to the next layer. Neurons are organised into layers, with information flowing from the input layer through hidden layers to the output layer. A neural network is considered deep if it has multiple hidden layers.

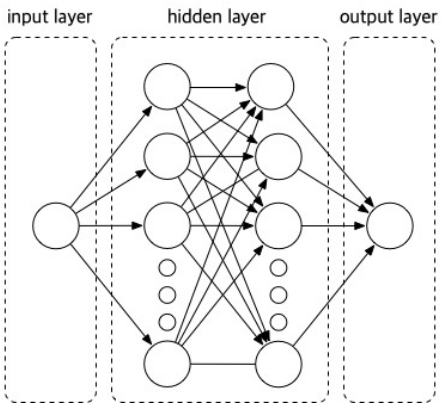


Figure 1. Structure of a neural network
Source: Illustration by authors.

With the increasing power of machine learning techniques, scientific algorithms have been developed that incorporate physical principles into machine learning models. Given known data as input, a NN can be trained such that its output V^{NN} closely matches the expected target data V^{true}, achieved by minimizing the error between V^{NN} and V^{true}. However, one limitation of such standard NNs is that while they can model physical or financial processes well based on the training data, their predictive capability is limited to the regions spanned by the training data, lacking generalisation beyond the training domain.

PINNs address this limitation by incorporating the governing PDEs into the loss function during the training process. PINNs specify a set of collocation points within the domain of the PDE, as well as additional points to describe the terminal and boundary conditions. Note that the way the points are generated is different from that of the MC method (Psaros et al., 2023; Cuomo et al., 2022).

3.3 PINNs for Asian Options

For the one-asset Asian option, we address a function approximation problem for the option price $V(S, I, t, r, \sigma)$ within the domain $[0, S_{Max}] \times [0, I_{Max}] \times [0, T] \times [0, r_{Max}] \times [0, \sigma_{Max}]$, and generate a set of N_p uniformly distributed collocation points $\mathcal{D}_{PDE}$.

To determine the option price, we additionally specify a terminal condition (TC) and three boundary conditions (BCs) as follows:

(i) *Terminal Condition at* $t = T$

$$V(S, I, T, r, \sigma) = \left(\frac{I}{T} - K\right)^{+} \tag{5}$$

(ii) *Boundary Condition 1 for* $S = 0$

$$V(0, I, t, r, \sigma) \approx e^{-r(T-t)} \left(\frac{I}{T} - K\right)^{+} \tag{6}$$

(iii) *Boundary Condition 2 for* $S = S_{Max}$

$$V(S_{Max}, I, t, r, \sigma) \approx \left(e^{-r(T-t)} \left(\frac{I}{T} - K\right) + S_{Max} \frac{1 - e^{-r(T-t)}}{rT}\right)^{+} \tag{7}$$

(iv) *Boundary Condition 3 for* $I = I_{Max}$ $(\geq KT)$

$$V(S, I_{Max}, t, r, \sigma) = e^{-r(T-t)} \left(\frac{I_{Max}}{T} - K\right) + S \frac{1 - e^{-r(T-t)}}{rT}. \tag{8}$$

BC (7) is approximated to provide an upper bound that is asymptotically consistent with the option price. BC (8) gives the exact value only when $I_{Max} \geq KT$. For TC and BCs, we generate n_D vectors uniformly for each of the temporal and three spatial axes, resulting in $N_D = 4n_D$ vectors in total, denoted as $\mathcal{D}_{data}$.

For the two-asset Asian basket options, the procedure is similar to that for the one-asset case, with differences only in the domain and boundary conditions. We consider the option price $V(S_1, S_2, I, t, r, \sigma_1, \sigma_2, \rho)$ within the domain $[0, S_{Max}] \times$

$[0, S_{Max}] \times [0, I_{Max}] \times [0, T] \times [0, r_{Max}] \times [0, \sigma_{Max}] \times [0, \sigma_{Max}] \times [-\rho_{Max}, \rho_{Max}]$. We consider the same ranges for S_1 and S_2 in this study for simplicity, but this can be easily adjusted to different ranges as required. We uniformly generate N_P vectors in the domain for the collocation points $\mathcal{D}_{PDE}$.

The TC and five BCs for the two-asset case are as follows:

(i) *Terminal Condition at* $t = T$

$$V(S_1, S_2, I, T, r, \sigma_1, \sigma_2, \rho) = \left(\frac{I}{T} - K\right)^+ \tag{9}$$

(ii) *Boundary Condition 1 for* $S_1 = 0$

$$V(0, S_2, I, t, r, \sigma_1, \sigma_2, \rho) = V_{\text{one-asset}}\left(\frac{S_2}{2}, I, t, r, \sigma_2\right) \tag{10}$$

(iii) *Boundary Condition 2 for* $S_1 = S_{Max}$

$$V(S_{Max}, S_2, I, t, r, \sigma_1, \sigma_2, \rho) \approx \left(e^{-r(T-t)}\left(\frac{I}{T} - K\right) + \frac{S_{Max} + S_2}{2} \times \frac{1 - e^{-r(T-t)}}{rT}\right)^+ \tag{11}$$

(iv) *Boundary Condition 3 for* $S_2 = 0$

$$V(S_1, 0, I, t, r, \sigma_1, \sigma_2, \rho) = V_{\text{one-asset}}\left(\frac{S_1}{2}, I, t, r, \sigma_1\right) \tag{12}$$

(v) *Boundary Condition 4 for* $S_2 = S_{Max}$

$$V(S_1, S_{Max}, I, t, r, \sigma_1, \sigma_2, \rho) \approx \left(e^{-r(T-t)}\left(\frac{I}{T} - K\right) + \frac{S_1 + S_{Max}}{2} \times \frac{1 - e^{-r(T-t)}}{rT}\right)^+ \tag{13}$$

(vi) *Boundary Condition 5 for* $I = I_{Max}$ $(\geq KT)$

$$V(S_1, S_2, I_{Max}, t, r, \sigma_1, \sigma_2, \rho) = e^{-(T-t)r}\left(\frac{I_{Max}}{T} - K\right) + \frac{S_1 + S_2}{2} \times \frac{1 - e^{-r(T-t)}}{rT} \tag{14}$$

BCs (11) and (13) are approximations, and BC (14) holds only when $I_{Max} \geq KT$, like the one-asset option. For BCs (10) and (12), the one-asset option price is required, which can be obtained from a PINN trained for the one-asset option. For TC and BCs, we uniformly generate n_D vectors for each of the temporal and five spatial axes, resulting in $N_D = 6n_D$ vectors in total for $\mathcal{D}_{data}$.

We train a NN V_θ^{NN}, parameterised by θ, which consists of weights and biases, to approximate the option value. The NN is trained by minimising the weighted sum of errors, defined as the loss function:

$$\ell \equiv w * \ell_{PDE} + \ell_{data},$$

where w is a weight parameter, and ℓ_{PDE} and ℓ_{data} are the mean squared errors (MSE) for the PDE residuals and data (TC and BC), respectively,

$$\ell_{PDE} \equiv \frac{1}{N_P} \sum_{\boldsymbol{x} \in \mathcal{D}_{PDE}} \left(\mathcal{L}\left[V_\theta^{NN}\right]\Big|_{\boldsymbol{x}}\right)^2,$$

$$\ell_{data} \equiv \frac{1}{N_D} \sum_{(\boldsymbol{x}, V) \in \mathcal{D}_{data}} \left(V - V_\theta^{NN}(\boldsymbol{x})\right)^2.$$

$\mathcal{L}[\cdot]$ is the operator in (3) or (4) for one or two assets respectively, see Algorithm 1.

Algorithm 1: Training of PINNs for Asian Option Pricing

Input: number of epochs (N_{epochs}), learning rate (η)
Output: a trained model
For i in $\{1, 2, \dots, N_{epochs}\}$
  $\ell_{PDE} \leftarrow 0$
  Randomly sample the datasets $\mathcal{D}_{PDE}$, $\mathcal{D}_{data}$
  For $\boldsymbol{x}$ in $\mathcal{D}_{PDE}$
    $\ell_{PDE} \leftarrow \ell_{PDE} + \frac{1}{N_P}\left(\mathcal{L}\left[V_\theta^{NN}\right]\big|_{\boldsymbol{x}}\right)^2$
  End
  $\ell_b \leftarrow 0$
  For $(\boldsymbol{x}, V)$ in $\mathcal{D}_b$
    $\ell_{data} \leftarrow \ell_{data} + \frac{1}{N_D}\left(V - V_\theta^{NN}(\boldsymbol{x})\right)^2$
  End
  $\ell \leftarrow w\ell_{PDE} + \ell_{data}$
  $\theta \leftarrow \theta - \eta\nabla\ell$
End
Return u_θ^{NN}

PINNs offer several advantages over classical numerical schemes. First, NNs, including PINNs, are universal function approximators, meaning that they can approximate any function with sufficient complexity. Second, partial derivatives can be computed efficiently using automatic differentiation without requiring finite difference approximations. Third, PINNs are mesh-free and do not require discretisation of the simulation domain. Additionally, once the training is completed, option valuation can be performed almost instantly, providing significant computational speedup compared to traditional numerical methods. This advantage is particularly notable when the model is trained to handle multiple input variables simultaneously, such as the risk-free interest rate r and volatility σ. In this case, option values for various scenarios can be obtained promptly and simultaneously without the need for retraining or additional computation.

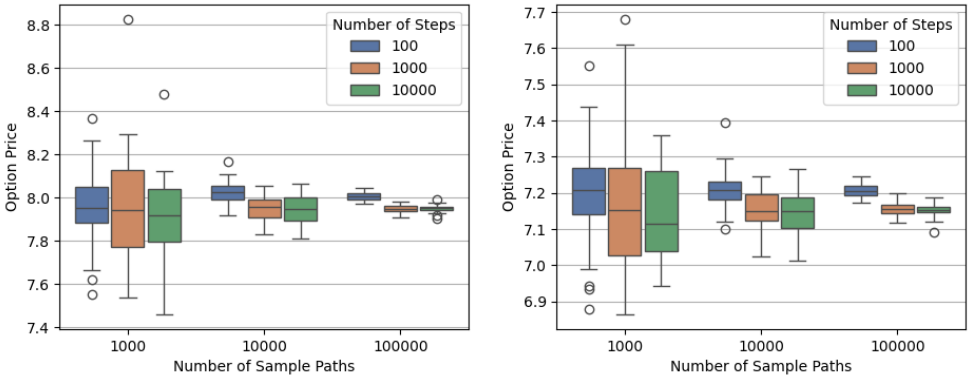
4. Numerical Results

This section presents numerical experiments to evaluate the performance of the PINNs in pricing Asian options. The results obtained from the PINNs are compared with those from the traditional MC method, which serves as a baseline. The comparison focuses on two key aspects: *relative error* and *computational efficiency*. All experiments were conducted on an Intel(R) Xeon(R) Platinum 8175M CPU @ 2.50GHz with 4 cores and 16GB of memory. The PINN training was performed on

a single NVIDIA L4 GPU, while the execution times reported herein are measured on the CPU, excluding the offline training time of the PINN.

4.1 Monte Carlo Results

To establish reference values for both one-asset and two-asset Asian option prices, we implemented the MC method with the antithetic variate technique for variance reduction. Figure 2 visually presents the estimated prices across various numbers of sample paths (M) and time steps (N). Option prices tend to be overestimated when a small number of time steps is used, and the variance of the estimation is high when the number of sample paths is low. Therefore, to obtain accurate prices using the MC method, it is necessary to simulate with sufficiently large numbers of sample paths and time steps, which leads to inefficiencies in both time and memory usage. For this study, the values obtained using 100,000 sample paths over 10,000 time steps are used as the reference values, V^{Ref}, resulting in execution times of 53 seconds and 120 seconds per option value for the one-asset and two-asset cases, respectively.



(a) One-asset Asian option **(b) Two-asset Asian option**

Figure 2. Boxplots of the prices from MC simulations
Source: Calculation made by authors.

To measure the error for comparison, we use the mean absolute error (MAE) between the reference option prices (V^{Ref}) and the estimated option prices $(\tilde{V})$, calculated across diverse asset prices. The MAE is defined as:

$$\mathrm{MAE}(V^{Ref}, \tilde{V}) = \frac{1}{N_{price}} \sum_i \left| V^{Ref}(S_i) - \tilde{V}(S_i) \right| ,$$

where N_{price} is the total number of option prices compared.

Table 1 presents the results obtained from the MC method for the one-asset case with $K = 100, r = 0.05, \sigma = 0.3, T = 1$, and $S(t_1)$ ranging from 0 to 300. Table 2 shows the results for the two-asset case with $K = 100, r = 0.05, \sigma_1 = 0.2, \sigma_2 = 0.4, \rho = 0.5, T = 1$, and $S_1(t_1)$, $S_2(t_1)$ ranging from 0 to 300. In both tables,

increasing the number M of sample paths leads to a decrease in the corresponding MAE; however, this also results in longer execution time.

Table 1. The errors of the one-asset Asian option prices from MC simulations ($K = 100, r = 0.05, \sigma = 0.3, T = 1, S(t_1) \in [0, 300]$)

M **(Number of sample paths)**	***N*** **(Number of time steps)**	**MAE**	**Execution time (s)**
1,000	100	0.088	2.8×10^{-3}
	1,000	0.103	2.2×10^{-2}
	10,000	0.073	3.1×10^{-1}
10,000	100	0.034	2.4×10^{-2}
	1,000	0.032	3.9×10^{-1}
	10,000	0.033	5.0×10^{0}
100,000	100	0.029	4.8×10^{-1}
	1,000	0.015	4.6×10^{0}

Source: Calculation made by authors.

Table 2. The errors of the two-asset Asian basket option prices from MC simulations ($K = 100, r = 0.05, \sigma_1 = 0.2, \sigma_2 = 0.4, \rho = 0.5, T = 1, S(t_1), S(t_2) \in [0, 300]$)

M **(Number of sample paths)**	***N*** **(Number of time steps)**	**MAE**	**Execution time (s)**
1,000	100	0.111	6.8×10^{-3}
	1,000	0.101	6.2×10^{-2}
	10,000	0.113	7.0×10^{-1}
10,000	100	0.046	7.1×10^{-2}
	1,000	0.035	1.0×10^{0}
	10,000	0.035	1.0×10^{1}
100,000	100	0.036	9.2×10^{-1}
	1,000	0.016	1.1×10^{1}

Source: Calculation made by authors.

4.2 PINNs for the one-asset Asian options

We trained the PINNs to solve the PDE in Equation (3) for one-asset Asian options with $K = 1$, $T = 1$, and various values of $S, I, r,$ and σ from $S \in [0, S_{Max}]$, $I \in [0, I_{Max}]$, $r \in [0, r_{Max}]$, and $\sigma \in [0, \sigma_{Max}]$, where $S_{Max} = 3, I_{Max} = 1, r_{Max} = 0.5$, and $\sigma_{Max} = 1$. It is important to note that although the PINNs are trained with $K = 1$, once trained, the model can be extended to any strike value K' using the following transformation:

$$\text{option value for strike } K' = K' \cdot V_\theta^{NN}\left(\frac{S'}{K'}, 0, 0, r, \sigma\right),$$

where $V_\theta^{NN}(S, I, t, r, \sigma)$ is the solution from the PINNs for one-asset options trained with $K = 1$. This extension is valid under the assumption that the market model follows a geometric Brownian motion. In all evaluations conducted in this study, we consider the strike price $K' = 100$ and the asset price $S' \in [0, 100 * S_{Max}]$.

We employed a 4-layer multi-layer perceptron (MLP) with a Gaussian Error Linear Unit (GELU) activation function (Hendrycks and Gimpel, 2016), defined by:

$$\mathrm{GELU}(x) = x\Phi(x),$$

where Φ is the cumulative distribution function of the standard Gaussian distribution. The model was trained using the Adam optimiser (Kingma and Ba, 2015), a first-order method with momentum acceleration, for 500,000 epochs. To facilitate convergence in this non-convex problem, we decreased the learning rate from 0.001 to 0 using cosine annealing (Loshchilov and Hutter, 2016). The learning rate η_k in the k^{th} iteration is given by:

$$\eta_k = \frac{\eta_0}{2}\left(1 + \cos\left(\frac{\pi k}{N_{epochs}}\right)\right),$$

where η_0 is the initial learning rate. We used a single cycle, making this equivalent to weight decay. During each training iteration, we uniformly sampled $N_P = 1000$ collocation points in the domain and $n_D = 100$ points for each of temporal and three spatial axes, resulting in $N_D = 4 \times n_D = 400$ vectors in $\mathcal{D}_{data}$.

Table 3 presents the results for various numbers of nodes in a NN with $K = 100, r = 0.05, \sigma = 0.3, T = 1$ and $S(t_1)$ from 0 to 300. The offline training times, ranging from approximately 446 to 909 seconds, occur only once during the training phase. As the number of nodes increases, the network size grows, enhancing its representation capacity and consequently reducing both the training error and the MAE. Since we trained the model with parameters such as the risk-free interest rate and volatility, we can calculate option prices for any parameter values almost instantaneously. Notably, even when the number of nodes in NN is increased to 160 nodes, the CPU execution time remains extremely fast at just 2.3×10^{-5} seconds, significantly outperforming the MC method, which requires up to 4.6 seconds.

Table 3. Performance of PINNs on the one-asset Asian options with different numbers of nodes ($K = 100, r = 0.05, \sigma = 0.3, T = 1, S(t_1) \in [0, 300]$)

Number of nodes	Final training error	MAE	Execution time (s)
10	3.412×10^{-5}	0.140	1.6×10^{-5}
20	2.294×10^{-6}	0.030	1.4×10^{-5}
40	4.821×10^{-7}	0.023	1.6×10^{-5}
80	2.242×10^{-7}	0.011	1.8×10^{-5}
160	9.284×10^{-8}	0.012	2.3×10^{-5}

Source: Calculation made by authors.

Figure 3 provides a compelling visual comparison of the MAE and the execution time for both methods in the one-asset case across various parameters. The results demonstrate the superior performance of PINNs. Not only do PINNs exhibit remarkably shorter computation times, but they also achieve lower error rates compared to the MC method. Most notably, PINN configurations with 80 and 160

nodes consistently outperform all MC results in accuracy, while dramatically reducing execution time.

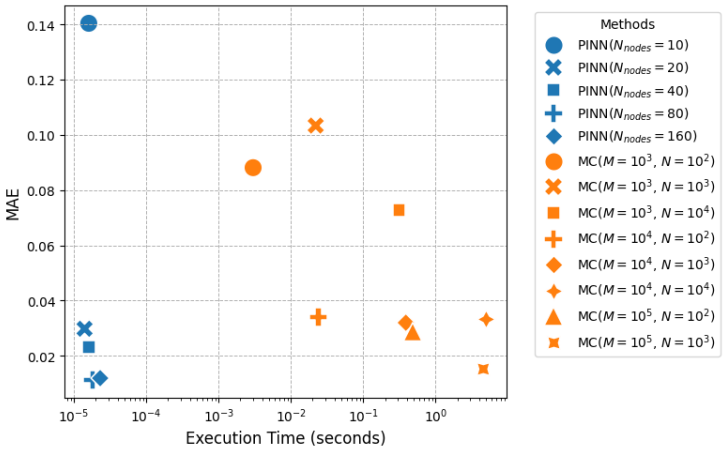
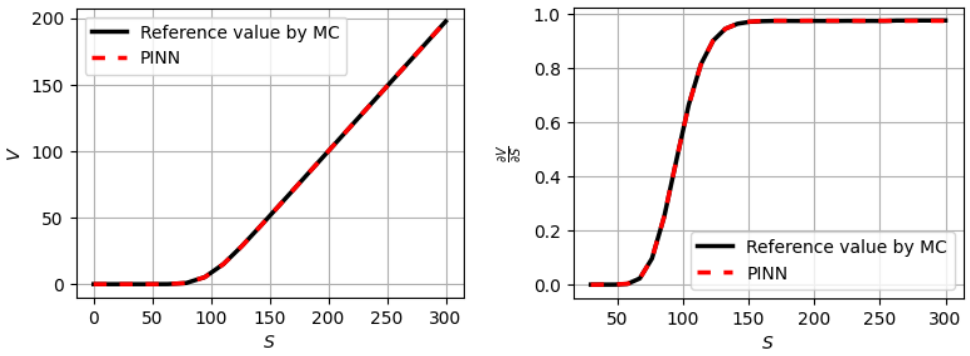


Figure 3. Comparison between PINNs and MC for one-asset options

Source: Calculation made by authors.

Figure 4(a) illustrates the option prices calculated by the PINN with 160 nodes compared to the reference values, while Figure 4(b) shows the delta values. Both sets of results are almost identical to their respective reference values. The PINNs demonstrate the ease of calculating Greeks, such as delta, due to the automatic differentiation capabilities of neural networks. The PINN computes delta in 6.3×10^{-4} seconds, while the reference value from the MC method takes 146 seconds. The proposed approach allows for efficient and accurate computation of these sensitivities without relying on finite difference approximations, providing valuable insights into the behaviour of option prices with respect to underlying parameters.



(a) prices **(b) delta**

Figure 4. Estimated prices and delta for the one-asset Asian option using the PINN ($K = 100, r = 0.05, \sigma = 0.3, T = 1, S(t_1) \in [0, 300]$)

Source: Calculation made by authors.

Table 4 presents the comparisons of one-asset option prices calculated by the PINNs and the MC for various values of S and σ. The PINN with 160 nodes was used and the reference MC results were obtained with $M = 100{,}000$ and $N = 1{,}000$. PINNs achieve consistently low errors in various scenarios, demonstrating efficiency and accuracy. This result highlights the PINN method's ability to deliver high-quality outcomes with much less computational resources, marking a substantial advancement in option pricing techniques.

Table 4. Prices of the one-asset Asian options calculated using PINN and MC ($K = 100, r = 0.05, T = 1$)

S	σ	V^{Ref}	PINNs	MC (90% confidence interval)
90	0.1	0.210	0.213	[0.207, 0.212]
	0.2	1.576	1.564	[1.557, 1.581]
	0.3	3.366	3.381	[3.365, 3.412]
100	0.1	3.646	3.647	[3.638, 3.653]
	0.2	5.787	5.772	[5.747, 5.786]
	0.3	7.905	7.947	[7.927, 7.987]
110	0.1	12.214	12.214	[12.211, 12.216]
	0.2	13.049	13.052	[13.042, 13.068]
	0.3	14.651	14.653	[14.626, 14.681]

Source: Calculation made by authors.

4.3 PINNs for the two-asset Asian options

Neural networks offer advantages in approximating high-dimensional functions, which circumvent the curse of dimensionality, where the problem space expands exponentially with dimensionality. Therefore, PINNs can be effectively used for calculating option prices for multi-asset options as well. In this section, we trained the PINNs to solve the two-asset PDE in Equation (4) with $K = 1$, $T = 1$, and various values of $S_1, S_2, I, r, \sigma, \rho$ in $S_1 \in [0, S_{Max}], S_2 \in [0, S_{Max}], I \in [0, I_{Max}], r \in [0, r_{Max}], \sigma \in [0, \sigma_{Max}]$ and $\rho \in [-\rho_{Max}, \rho_{Max}]$, where $S_{Max} = 3$, $I_{Max} = 1$, $r_{Max} = 0.5$, $\sigma_{Max} = 1$, and $\rho_{Max} = 1$. Like the one-asset options, despite training the PINNs for $K = 1$, we can compute the option value for an arbitrary strike value K' using the following transformation:

$$\text{option values for strike } K' = K' \cdot V_\theta^{NN}\left(\frac{S_1'}{K'}, \frac{S_2'}{K'}, 0, 0, r, \sigma_1, \sigma_2, \rho\right),$$

where $V_\theta^{NN}(S_1, S_2, I, t, r, \sigma_1, \sigma_2, \rho)$ is the PINNs for two-asset options trained with $K = 1$. Throughout our evaluations of two-asset options, we specifically consider $K' = 100$ and the asset prices $S_1',\ S_2' \in [0, 100 * S_{Max}]$.

We employed a model with the same configurations as in the one-asset case, consisting of a 4-layer MLP with a GELU activation function. The model was trained using the Adam optimiser for 500,000 iterations, starting with an initial learning rate

of 0.001, which was decreased to 0 using cosine annealing. In each training iteration, we uniformly sampled $N_P = 1000$ collocation points within the domain and $n_b = 100$ points for each of temporal and five spatial axes, resulting in $N_D = 6 \times n_D = 600$ vectors in $\mathcal{D}_{data}$.

Table 5 presents the performance of the PINNs with varying numbers of nodes for $K = 100, r = 0.05, \sigma_1 = 0.2, \sigma_2 = 0.4, \rho = 0.5, T = 1$, and S_1, S_2 ranging from 0 to 300. The offline training times, ranging from 748 to 1928 seconds, occur only once. Similar to the one-asset case (as shown in Table 3), as the number of nodes increases, the PINNs achieve lower final training errors and MAE values. Remarkably, the PINN's execution time is reported at 2.4×10^{-5} seconds per option price even with 160 nodes, significantly faster than the MC method.

Table 5. Performance of PINNs on the two-asset Asian basket option with different numbers of nodes
($K = 100, r = 0.05, \sigma_1 = 0.2, \sigma_2 = 0.4, \rho = 0.5, T = 1, S(t_1), S(t_2) \in [0, 300]$)

Number of nodes	Final training error	MAE	Execution time (s)
10	1.239×10^{-4}	0.196	1.5×10^{-5}
20	2.893×10^{-5}	0.054	1.6×10^{-5}
40	4.372×10^{-6}	0.032	1.8×10^{-5}
80	1.844×10^{-6}	0.023	2.0×10^{-5}
160	5.929×10^{-7}	0.016	2.4×10^{-5}

Source: Calculation made by authors.

Figure 5 visually compares the MAE and the execution time of the two methods across various parameters. Extending its superiority from the one-asset scenario, PINNs show remarkably faster computation times while achieving similar or even lower MAE. Most impressively, PINNs with 160 nodes consistently outperform all configurations of the MC method, excelling in both speed and accuracy.

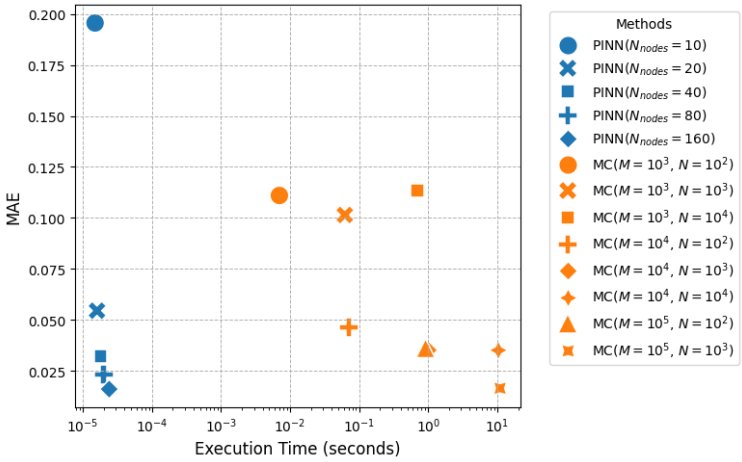
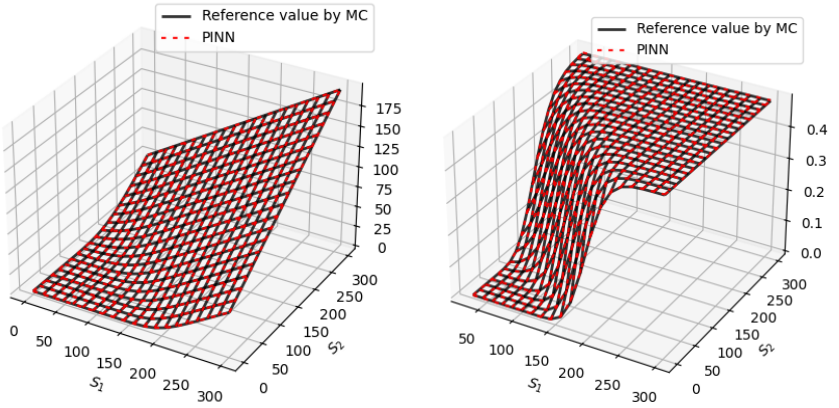


Figure 5. Comparison between PINNs and MC for two-asset options
Source: Calculation made by authors.

Figure 6(a) and (b) present the prices and delta values of the two-asset Asian basket option calculated using the PINN with 160 nodes, compared to reference values. Like the one-asset case, the PINN computes both accurately and efficiently, calculating delta values in 9.8×10^{-4} seconds compared to the MC method's 225 seconds. This performance highlights the PINN's scalability and potential as a powerful tool for pricing and risk management in multi-dimensional financial instruments.



(a) prices **(b) delta**

Figure 6. Estimated prices and delta for the two-asset Asian basket option using PINN ($K = 1, r = 0.05, \sigma_1 = 0.2, \sigma_2 = 0.4, \rho = 0.5, T = 1$)

Source: Calculation made by authors.

Table 6 compares two-asset Asian basket option prices calculated by the PINN and the MC for various values of S_1, σ_1, and σ_2. The PINN with 160 nodes was used and the MC results were obtained with $M = 100{,}000$ and $N = 1{,}000$. PINN consistently achieves low errors across a variety of scenarios, highlighting its effective and precise calculation and robustness in complex multi-asset scenarios.

Table 6. Prices of the two-asset Asian basket options calculated using PINN and MC ($K = 100, S_2 = 100, r = 0.05, \rho = 0.5, T = 1$)

S_1	$\sigma_1 (= \sigma_2)$	V^{Ref}	PINN	MC (90% confidence interval)
90	0.1	0.916	0.908	[0.905, 0.916]
	0.2	2.683	2.658	[2.665, 2.695]
	0.3	4.519	4.509	[4.503, 4.549]
100	0.1	3.376	3.375	[3.371, 3.382]
	0.2	5.188	5.175	[5.172, 5.213]
	0.3	7.060	7.064	[7.046, 7.104]
110	0.1	7.451	7.433	[7.447, 7.454]
	0.2	8.619	8.604	[8.601, 8.628]
	0.3	10.282	10.248	[10.229, 10.283]

Source: Calculation made by authors.

5. Conclusions

This study introduces an efficient method for pricing Asian options in financial markets using PINNs. By integrating physical laws into the learning process, PINNs enhance both accuracy and interpretability. This approach minimises the need for extensive training data and improves generalisation in solving differential equations.

As illustrated in Figures 3 and 5, PINNs achieve low error rates more efficiently than the traditional MC method for both one-asset and two-asset Asian options. Unlike the MC method, which requires recalculations for each set of parameters, PINNs can instantly compute prices for arbitrary parameters once trained.

Furthermore, the PINN methodology has the potential for accurate pricing of other complex exotic options and could be extended to applications in optimal control and asset allocation problems.

Acknowledgements: *This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A1054766).*

References

[1] Bai, Y., Chaolu, T., Bilige, S. (2022), *The application of improved physics-informed neural network (IPINN) method in finance. Nonlinear Dynamics*, 107, 3655-3667.

[2] Barraquand, J., Pudet, T. (1996), *Pricing of American path-dependent contingent claims. Mathematical Finance*, 6(1), 17-51.

[3] Cuomo, S., Schiano, V., Giampaolo, F., Rozza, G., Raissi, M., Piccialli, F. (2022), *Scientific machine learning through Physics-Informed Neural Networks: Where we are and What's next. Journal of Scientific Computing*, 92(3).

[4] Fischer, T., Krauss, C. (2018), *Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research*, 270, 654-669.

[5] Gatta, F., Di Cola, V.S., Giampaolo, F., Piccialli, F., Cuomo, S. (2023), *Meshless methods for American option pricing through Physics-Informed Neural Networks. Engineering Analysis with Boundary Elements*, 151, 68-82.

[6] Glasserman, P. (2003), *Monte Carlo methods in financial engineering (Stochastic modelling and applied probability*, 53), Springer, New York, USA.

[7] Goodell, J.W., Kumar, S., Lim, W.M., Pattnaik, D. (2021), *Artificial intelligence and machine learning in finance: Identifying foundations, themes, and research clusters from bibliometric analysis. Journal of Behavioral and Experimental Finance*, 32 (100577).

[8] Hendrycks, D., Gimpel, K. (2016), *Gaussian error linear units (GELUs),* arXiv:1606.08415.

[9] Kingma, D.P., Ba, J. (2015), *Adam: A method for stochastic optimization*, Proceedings of the 3rd *International Conference on Learning Representations.*

[10] Kwok, Y.-K. (2008), *Mathematical models of financial derivatives. Springer Berlin Heidelberg, Germany*.

[11] Lin, C.-G., Chang, C.-C. (2020), *Approximate analytic solution for Asian options with stochastic volatility, The North American Journal of Economics and Finance*, 54 (100949).

[12] Lo, C.-L., Palmer, K.J., Yu, M.T. (2014), *On Moment-Matching Approximations for Asian Options, Journal of Derivatives*, 21(4), 103-122.

[13] Loshchilov, I., Hutter, F. (2017*), SGDR: Stochastic Gradient Descent with Warm Restarts*, Proceedings of the 5th *International Conference on Learning Representations,* Toulon, France, 24–26 April 2017, 1-16.

[14] Moon, K.-S., Kim, H. (2019), *Performance of deep learning in prediction of stock market volatility. Economic Computation and Economic Cybernetics Studies and Research*, 53(2), 77-92.

[15] Moon, K.-S., Kim, H. (2023), *Efficient asset allocation based on prediction with adaptive data selection. Economic Computation and Economic Cybernetics Studies and Research*, 57(1), 57-72.

[16] Nabipour, M., Nayyeri, P., Jabani, H., Mosavi, A., Salwana, E., Shahab, S. (2020), *Deep learning for stock market prediction. Entropy*, 22(8), 840.

[17] Psaros, A.F., Meng, X., Zou, Z., Ling, G., Karniadakis, G.E. (2023), *Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons, Journal of Computational Physics*, 477 (111902).

[18] Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M. (2020), *Financial time series forecasting with deep learning: A systematic literature review: 2005-2019. Applied soft computing journal*, 90 (106181).

[19] Shi, Q., Yang, X. (2014), *Pricing Asian options in a stochastic volatility model with jumps, Applied Mathematics and Computation*, 228, 411-422.

[20] Shiraya, K., Takahashi, A. (2017), *A general control variate method for multi-dimensional SDEs: An application to multi-asset options under local stochastic volatility with jumps models in finance. European Journal of Operational Research*, 258, 358-371.

[21] Vecer, J. (2014), *Black-Scholes representation for Asian options, Mathematical Finance*, 24(3), 598-626.

[22] Wang, X., Li, J., Li, J. (2023), *A deep learning based numerical PDE method for option pricing, Computational Economics*, 62, 149-164.

[23] Wilmott, P., Howison, S., Dewynne, J. (1995), *The mathematics of financial derivatives. Cambridge University Press*, Cambridge, UK.