Published in AJSE, Vol: 24, Issue:1
Received on 20th October 2024
Revised on 21st April 2025
Published on 31th August 2025

AIUB JOURNAL OF SCIENCE AND ENGINEERING
ISSN: 1608 - 3679 (print) 2520 - 4890 (Online)
https://doi.org/10.53799/9qpmy141

Option Pricing of the Black–Scholes Equation: A Study Using Physics-Informed and Artificial Neural Networks

M. M. H. Imran[1], Muhammad Sajjad Hossain[2, *], Md. Abdul Alim[3]

***Abstract*—This study aims to carry out a comparative analysis of Artificial Neural Networks (ANN) and Physics-Informed Neural Networks (PINNs) in solving the Black-Scholes (BS) model for European call option pricing. PINNs have gained prominence in various engineering and financial fields due to their effectiveness in solving practical applications. In the last two decades, researchers have attempted to resolve BS model using numerical techniques; nevertheless, the difficulty of mesh generation complicates the numerical solution, particularly when addressing complex geometry. To address the BS model using an ANN, we initially trained our ANN with empirical data to obtain weights and biases leading to a fairly accurate solution for the untrained data. But, while solving the BS model with PINNs, no preexisting data is needed; we trained our PINNs model with the initial and boundary conditions. We employ the backpropagation technique in conjunction with the Adam optimizer to reduce the error. This study demonstrates that PINNs yield more accurate results than ANNs and that the predictions made by PINNs fit well with the exact findings. Thus, we can employ the PINNs method in substitution of the conventional numerical method.**

***Index Terms*— Black–Scholes equation, Option pricing, ANN, PINNs, Adam optimizer**

I. INTRODUCTION

Fischer Black and Myron Scholes introduced a new financial mathematical model called The Black-Scholes, is a mathematical model that describes the dynamics of financial markets in particular the pricing of the options: call option and put option.

Md. Majibul Hasan Imran[1] is with the Department of Mathematics, American International University-Bangladesh (AIUB), 408/1, Kuratoli, Dhaka-1229, Bangladesh. (email: imran.math@aiub.edu , imrandu065@gmail.com)

Muhammad Sajjad Hossain[2] is with the Department of Mathematics, Ahsanullah University of Science and Technology (AUST), 141 & 142, Love Road, Tejgaon Industrial Area, Dhaka-1208, Bangladesh. (*corresponding author email: msh.as@aust.edu , msh80edu@gmail.com)

Md. Abdul Alim[3] is with the Department of Mathematics, Bangladesh University of Engineering and Technology (BUET), Dhaka-1000, Bangladesh. (email: a0alim@gmail.com)

Pricing financial derivatives is of interest because it is employed to minimize the losses resulting from fluctuations in the prices of the underlying assets. The Black-Scholes Model has a wide range of practical applications, encompassing not only pricing options but also risk management, trading strategies, and financial engineering. Determining the fair value of exchange-traded possibilities worldwide is the most popular use.

The theoretical option price can be established by obtaining the exercise price, underlying asset price, option period to expiration, and risk-free rate of return. Traditional numerical methods have been used to solve BS over the last few decades. However, recent progress in AI driven machine learning based neural network techniques: ANN and PINN paving the new way to solve data based and differential equations in a more efficient way. The accuracy of option prices is crucial because even small pricing errors can lead to significant financial consequences.

Physics Informed Neural Networks (PINNs) which integrates the governing equations, initial and boundary conditions directly into training process. This approach enables the network to learn solutions without labeled trained data.

Using PINN we can solve the partial governing equations (PDE):

$$\mathcal{N}\left[u(x,t)\right]=0,\quad (x,t)\in\Omega$$

Subject to some initial and boundary conditions, where, $\mathcal{N}[\bullet]$ represent PDE operator and $u(x,t)$ is the target function and Ω is spatio-temporal domain.

ANN can identify the patterns but involves no physics from the equation. So, in this paper, we are planning to apply PINN method to solve the BS model, which doesn't require any trained or previously obtained data. PINN involves the hidden physics of the problem. PINNs are deep learning networks that generate an estimated solution for the given inputs. PINNs is an unsupervised learning method that converts the governing equations with the boundary and initial condition into an optimization problem. In our best knowledge, this is the first attempt to apply PINNs on Black-Scholes model also compare the ANN and PINN prediction numerically and graphically.

This work aims to fill the following gap:

a) *Implementing ANN and PINN for the BS equation with European call options.*

b) *Comparing its performance with a data driven ANN and PINN outcomes with numerical solutions.*

c) *Highlighting the advantages and limitations of ANN and PINNs in the context of option pricing.*

II. Literature Review

Anuwar [1], solved the BS model for European call option with the help of numerical method. In his study, L1-norm estimate was used to assess the model's correctness once the stability requirement was established via convex combination. Julia [2] investigated BS model by considering multiple factors that affect an option's value. She solved the BS model numerically after converting it into a convection-diffusion problem. Heider [3] used Crank-Nicolson scheme for the BS model and gave the stability analysis for problems. Pradip and Goura [4] presented a new computational approach for numerically solving the generalized BS model, utilizing a Crank-Nicolson scheme and sixth order B-spline collocation method which was proven stable, has second-order convergence for time and sixth-order convergence for space variables. Hizaz et al. [5] studied linear and non-linear time fractional BS model and suggested a meshless collocation approach based on hybrid Gaussian cubic radial basis functions with polynomials and obtained good accuracy for valuing the option price. Many researchers around the globe are still trying to find more accurate and suitable method to find the solution of BS model.

In this new era, machine learning methods have become more and more popular due to their ability to identify the pattern in large data sets and can handle complex scenarios. A neural network is a branch of machine learning idea that replicates the human brain's organic computing mechanism. Neural Network research has received great interest in numerous fields of science. The applications include but are not limited to computational material and chemical research, the pharmaceutical sector, biomedical engineering, forecasting science, robotics, aerospace, and mechanical engineering. The rationale behind employing neural networks originates from offering universal approximation for continuous functions. Among these ML methods, multilayer perceptron neural network techniques, namely ANN and PINNs are used to solve differential equations, optimization problem, pattern recognition, simulations (Deng [6], Blechschmidt [7], Khan [8]). Fran [9] used PINNs method for solving PDEs based on Convolution Neural Network (CNN). Cuomo [10], in his work he showed the ways of solving DEs using the ML based techniques through PINN and answered their question where it can be used and what's next. Imran *et al.* [11,12] applied ANN and PINNs methodology to solve fluid dynamics applications: boundary layer problems.

Neural networks, specifically ANN, have been utilized in option pricing to identify market patterns and rectify deficiencies in conventional pricing models. The integration of machine learning into finance started with basic models such as support vector machines and has subsequently advanced to deep learning frameworks adept at simulating intricate financial processes (Sirignano [13]). Eskiizmirliler [14] studied BS equation for European call option value using neural networks and approximates BS model solutions using a trial function involving initial and boundary conditions and hence predicted over the unseen data. The authors considered optimization methods like particle swarm optimization and gradient-type monotone iteration process. Chen [15] studied Laguerre neural network where he used Laguerre activation function to solve the BS model. Gunon [16], Daniel [17], trained neural network using real world data from the stock options and then solve BS equation to get the price of the options.

III. BSM MODEL FOR OPTION PRICING

Assume that the option price of the underlying assets follows Brownian geometric motion. The option pricing PDE model is the parabolic differential model given in Eq. 1 [18,19].

$$\frac{\partial V}{\partial t}+\frac{1}{2}S^2\sigma^2\frac{\partial^2 V}{\partial S^2}+rs\frac{\partial V}{\partial S}-rV=0,$$
$$(S,t)\in(0,\infty)\times(0,T] \quad (1)$$

Here, $V(S,t)$ represents the value of European Call option. Assume that $V(S,t)$ is sufficiently smooth and first order time and second order space continuous. $V(S,t)$ is a contract where a holder may purchase the asset $S(t)$ at a prescribed time in the future, known as maturity time T, for a prescribed amount of price K called strike price. The writer has the obligation to sell the asset if the holder chooses to exercise his right. Therefore, the value of the option at maturity is known as pay-off function, defined by $V(S,T)=(S-K)^+=\max(S-K,0)$. Again, at $S=0$, option has no value, therefore $V(0,t)=0$. Now, as $S\to\infty$, for a call option, the option value approaches the $S-Ke^{-r(T-t)}$. Here $Ke^{-r(T-t)}$ is discounted strike price. In summary the associated boundary condition for Eq. (1) can be written in the following form: [20,21]

$$V(S,T)=\max(S-K,0),0<S<\infty \quad (2)$$
$$V(0,t)=0, and$$
$$V(S\to\infty,t)\to S-Ke^{-r(T-t)},0<t\le T \quad (3)$$

IV. SOLUTION OF THE BLACK-SCHOLES MODEL:

To get non-dimensional form of Eq. 1 introduce a new variable

$$\zeta=\frac{S}{K}\text{ and }\tau=T-t, \quad (4)$$

where ζ represents non-dimensional stock price and τ represents non-dimensional time. Using these transformation Eq.1 becomes

$$\frac{\partial\psi}{\partial\tau}-\frac{1}{2}\zeta^2\sigma^2\frac{\partial^2\psi}{\partial\zeta^2}-r\zeta\frac{\partial\psi}{\partial\zeta}+r\psi=0 \quad (5)$$

With the initial and boundary conditions:

$$\psi(\zeta,0)=\max(\zeta-1,0),0<\zeta<\zeta_{max} \quad (6)$$
$$\psi(0,\tau)=0, and$$
$$\psi(\zeta\to\zeta_{max},\tau)\to S-e^{-r\tau},0<\tau\le T \quad (7)$$

Here, ψ =non-dimensional option price, ζ_{max} =maximum stock price.

A. *Numerical Solution:*

Eq. (5) is a *parabolic* equation. To solve this numerically we have adopted 2nd order central difference scheme for spatial derivatives

$$\frac{\partial \psi}{\partial \zeta} = \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2\Delta\zeta} \quad (8)$$

$$\frac{\partial^2 \psi}{\partial \zeta^2} = \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{(\Delta\zeta)^2} \quad (9)$$

And to obtain explicit difference scheme, we have adopted 1st order backward difference scheme for temporal derivative

$$\frac{\partial \psi}{\partial \tau} = \frac{\psi_{i,j} - \psi_{i+1,j}}{\Delta\tau} \quad (10)$$

Substituting Eq. (8), (9) & (10) into (5)

$$\frac{\psi_{i,j} - \psi_{i+1,j}}{\Delta\tau} - \frac{1}{2}\zeta_j^2\sigma^2 \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{(\Delta\zeta)^2} - r\zeta_j \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2\Delta\zeta} + r\psi_{i,j} = 0$$

$$\psi_{i+1,j} = \psi_{i,j} + \Delta\tau\left(\frac{1}{2}\zeta_j^2\sigma^2 \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{(\Delta\zeta)^2} + r\zeta_j \frac{\psi_{i,j+1} - \psi_{i,j-1}}{2\Delta\zeta} - r\psi_{i,j}\right) \quad (11)$$

With $\psi_{0,j} = \max(\zeta_j - 1, 0)$, $\psi_{i,0} = 0$ *and* $\psi_{i,n} = \zeta_n - e^{-r\tau_i}$

Where, $i = \overline{0, \dots, m}$ *and* $j = \overline{0, \dots, n}$.

The CFL condition for Eq. (11) is approximately,

$$\Delta\tau \leq \frac{\Delta\zeta^2}{\sigma^2\zeta_{max}^2}.$$

B. *ANN Computation*

ANN is a computing model inspired by the human brain's system. ANN is made up of three layers: input Layer: that gets the input feature; hidden Layers: This is where all the computational work is executed. Each neuron takes information, adds up the weights of those inputs, and then runs that number through an activation function, and lastly, the output layer gives the desired effects. [22,23,24]:

From each input to hidden layers

$$y_j \to f(w_{ji}x_i + \beta_j), i = 1,2, \dots, n \quad (12)$$

From each hidden layer to output layer

$$\psi(\zeta, \tau) = \theta\left(\sum_{j=1}^{j=n} v_j y_j + \beta\right) \quad (13)$$

Here, $x = [\zeta, \tau]^T$, w_i, v_i = weights, β =bias, f, θ = denotes the activation function.

In this study inputs and outputs are positive, so we chose "ReLU" as an activation function defined by $f(x) = \max(x, 0)$.

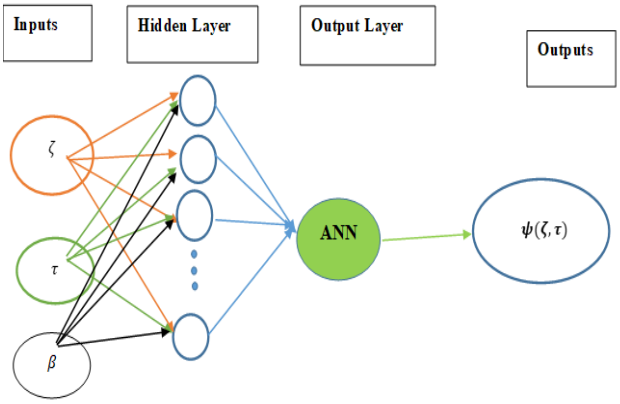

Figure 1: ANN general flow-structure

Now, we have to calculate the loss, defined by $L = \frac{1}{2}\sum_k \left(\psi(\zeta, \tau) - \hat{\psi}(\zeta, \tau)\right)^2$

To minimize the loss L with respect to weights and biases, we have used backpropagation technique. To achieve this, firstly we have chosen the weights and biases randomly, then adjust the initially guessed weights and biases using backpropagation algorithm. To update the weights and biases we have applied Adam (Adaptive Moment Estimation) optimizer. It's one of the most popular optimizers in the machine learning tools, and useful while working with big data. It computes adaptive learning rates for each parameter. Adam is the modified version of stochastic gradient descent namely AdaGrad and RMSProp. It utilizes the first moment (mean) and second moment (uncentered variance) estimates of the gradients. Now consider, our parameter, $\phi = [w, \beta]^T$, m, v represent first and second moment respectively, η =learning late, β_1, β_2represents decay rate and $\epsilon =$ small constant to avoid division by zero.To minimize the loss function using Adam, we have followed the following algorithms: [25,26]

Algorithm 1: Adam updating weights and biases

/*Initialization*/

$t \leftarrow 0$

for $i \in$ N do

input $\leftarrow \phi_i$

$m_0(\phi_i) \leftarrow 0_j$

$v_0(\phi_i) \leftarrow 0_j$

/*Perform T iteration of training*/

while $t < T$ do

$t \leftarrow t + 1$;

/*update $\boldsymbol{m}, \boldsymbol{v}, \boldsymbol{\theta}$*/

for $i \in$ N do

$g_t \leftarrow \frac{\partial L}{\partial \phi}$

$m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$

$v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$

$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$

$\hat{v}_t \leftarrow \frac{m_t}{1-\beta_2^t}$

$\phi \leftarrow \phi - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \epsilon}}$

update ϕ_T

C. Train Data sets for ANN

To train the ANN model, firstly we must choose the number of hidden layers, nodes, training data. For this, we apply trial and error methods to choose the above hyperparameters.

Table 1:Comparing MSE, RMSE, Relative L_2 loss for different ANN hyperparameters

Controlling parameters	ANN hyper-parameters	MSE	RMSE	Relative L_2 error
$r = 12\%$, $\sigma = 0.1$	HL=2, Nodes=20; $lr = 10^{-3}$	5.03×10^{-5}	7.1×10^{-3}	1.67×10^{-2}
	HL=2, Nodes=32; $lr = 10^{-3}$	8.24×10^{-6}	2.9×10^{-3}	6.73×10^{-3}
	HL=2, Nodes=64; $lr = 10^{-3}$	3.5×10^{-8}	5.9×10^{-4}	1.38×10^{-4}

and We have set a sequential feed forward neural network with 2 hidden layers each containing 64 nodes. Data sets to train ANN models are generated from numerical solution. For this, we choose

$\zeta \in (0, \zeta_{max}, 100)$ and $\tau \in (0, T, 100)$

$[\zeta_t, \tau_t]$ =meshgrid (ζ, τ), # total 101*101 datasets

We have used 80% data set for training and 20% data set for testing. We have chosen $\eta = 10^{-3}, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$.

ANN predicted option price $= \psi(\zeta_t, \tau_t)$.

D. PINN Computation

Physics–Informed Neural Networks (PINNs) are a scientific machine learning technique has been widely used to solve PDEs in various filed as an alternative solver of tradition numerical technique. PINNs approximate PDE solutions by training a neural network that minimize a loss function; loss function combines with the initial and boundary conditions along the space-time domain's boundary and the governing PDE residual at interior points in the domain (called collocation point). It integrates mathematical models into networks and strengthens the loss function with a residual term, restricting acceptable solutions [16,27]. Following the framework of PINNs proposed in [28], V_{PINN} is approximated option value given by a fully connected network, which takes the coordinates (ζ, τ) as inputs and $V_{PINN}(\zeta, \tau)$ as output. The neural network is composed of multiple hidden layers, where the inputs of each hidden layer and outputs are propagated through the network as

$$y_j = f\left(w_{i,j} x_i + \beta_j\right) \tag{14}$$

where $w_{i,j}$and β_j are learnable weights and biases, respectively; f is the activation function representing a simple nonlinear transformation. Mean square error (MSE) is defined as

$$MSE = \frac{1}{n-1} \sum_{i=1}^{n} \left(y_i - y_i^{(numerical)}\right)^2 \tag{15}$$

$$\text{RMSE} = \sqrt{MSE} \tag{16}$$

$$L_2 = \frac{\langle y_i - y_{ref} \rangle_2}{\langle y_{ref} \rangle_2} \tag{17}$$

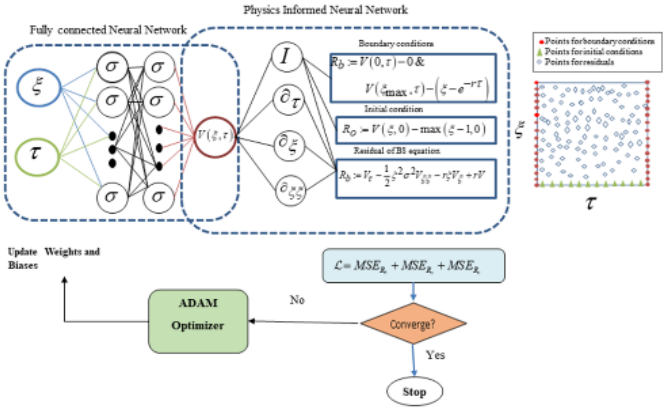

Fig 2: Working flow of Physics Informed Neural Network

The parameters involving in the neural network can be updated by minimizing a composite loss function with the following form

$$\mathcal{L} = MSE_{R_b} + MSE_{R_o} + MSE_{R_e} \tag{16}$$

Where, $MSE_{R_b} = \frac{1}{N_b} \sum_{n=1}^{N_b} |V(0,\tau) - 0|^2 + |V(\zeta_{max}, \tau) - (\zeta - e^{-r\tau})|^2$,

$MSE_{R_o} = \frac{1}{N_o} \sum_{n=1}^{N_o} |V(\zeta, 0) - \max(\zeta - 1, 0)|^2$,

$MSE_{R_e} = \frac{1}{N_r} \sum_{n=1}^{N_r} \left| V_\tau - \frac{1}{2}\zeta^2 \sigma^2 V_{\zeta\zeta} - r\zeta V_\zeta + rV \right|^2$, are respectively boundary loss, initial loss and PDE loss. and N_b, N_o, N_r denote number of boundary points, initial points and

colocation points. The PINN procedure graphical representation is given below in fig.2.

In table 2, PINN initial set up is shown with the number of neurons, hidden layers gradient tolerance and stop tolerance. [12]

Table 2: PINN set up with ADAM optimizer

Neurons	Hidden Layers	Gradient Tolerance	Stop Tolerance	Max Iteration
20	5	10^{-7}	10^{-7}	10000

Others Properties:

(1) Line Search Method: "Weak Wolf" Method to identify optimum learning rate. This technique provides a positive definite estimation of the inverse Hessian matrix.

(2) Gradient threshold technique used to eliminate gradient values that surpass the gradient threshold: "L2 norm" (MSE), If the L2 norm of the gradient of a trainable guess parameter is bigger than Threshold value of Gradient, then this technique will adjust the gradient in a manner that the L2 norm or MSE matches Threshold value of the Gradient.

V. Validation

We compare the ANN and PINN results with the numerical obtained results. To perform the error analysis, we plotted the value of option price at $t = 0$ in fig. 3. From fig. 3, one can easily observe that PINN outcomes are superior to ANN and very close to the approximate results. Sometimes PINN predictions are more accurate than the Numerical and approximate results when option value are zero or close to zero.

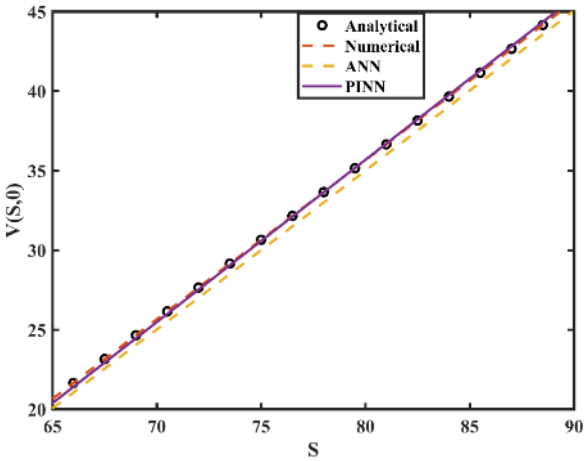

Fig 3: Comparative study of BSM model with ANN and PINN

VI. Result and Discussion

In this section, we demonstrated the solution the non-dimensional Eq. 5 with three methods: FDM, ANN and PINN. For numerical solution, considered an example with strike price, $K = 50$, $T = 1, S_{max} = 150$ and risk-free interest rate, $r = 0.12$ and volatility, $\sigma = 0.1$. In the non-dimensional case above parameter becomes $K = 50, T = 1, \zeta_{max} = 3, r = 0.12$ & $\sigma = 0.1$. We applied backward time and spatial center scheme to solve Eq.5 with the help of "MATLAB" software. To retrieve the actual solution, we have to multiply $\psi(\zeta, \tau)$ by the strike price that is, $V(S,t) = K \times \psi(\zeta, \tau)$.

The approximate solution of BSM model is shown in fig. 4a also the numerical solution of this model is shown in fig. 4b.

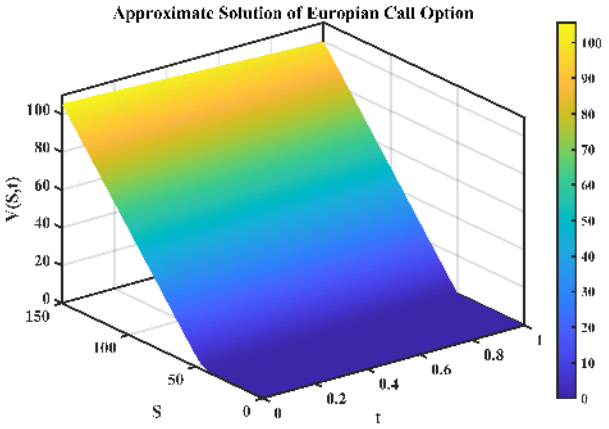

Fig 4a: Approximate Solution of BSM

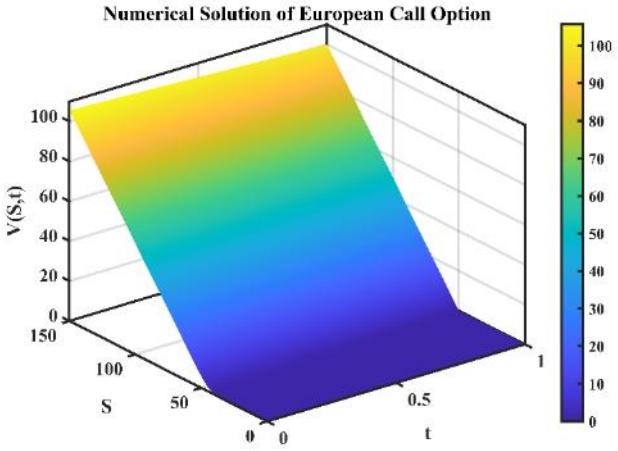

Fig 4b: FDM Solution of BSM

In machine learning study, it is common to normalize the data before the train. In that case, For ANN prediction, to get better outcome firstly we normalized our data using Min-Max normalization given by Eq. 17,

$$X_i = \frac{y - \max(y)}{\max(y) - \min(y)} \quad (17)$$

Then applied feed forward neural network to predict the ANN results. Lastly, we de-normalize the ANN outcomes using Min-Max algorithm to retrieve the actual solutions. ANN predicted outcomes for unseen data plotted in fig. 5. In fig. 6, ANN training and validation loss have been graphically illustrated. From fig. 6, we can see the initial loss training and validation was 0.01409 and 5.65×10^{-5} respectively after 100 epoch training and validation is 2.10×10^{-6} and 3.49×10^{-7} respectively. In figure 7, we have shown actual value with the ANN predicted value, ANN predicted option values nearly close the actual solution.

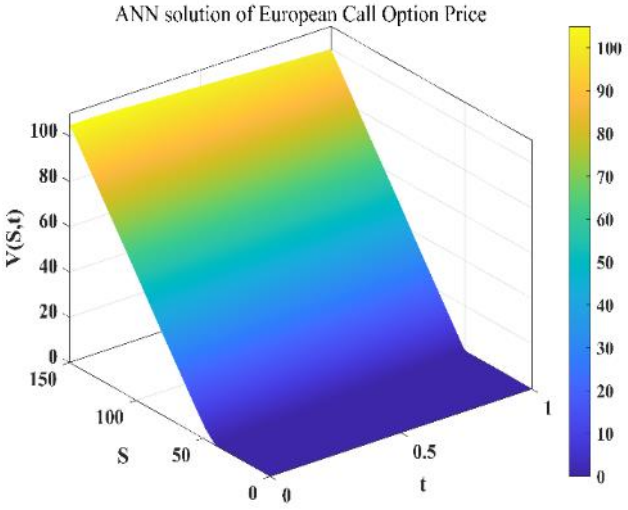

Fig 5: ANN Solution of BSM

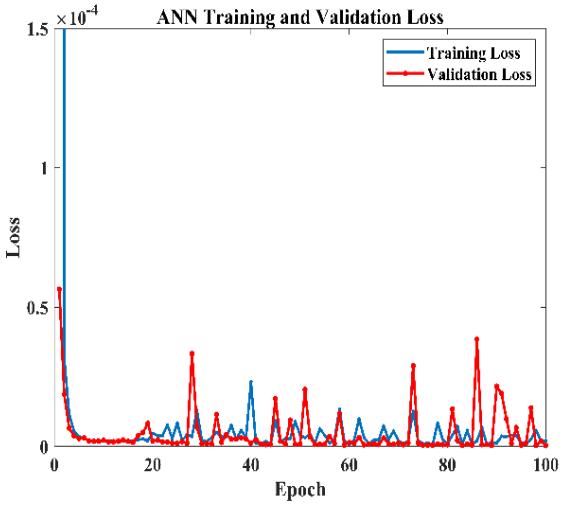

Fig 6: ANN Training and Validation Loss

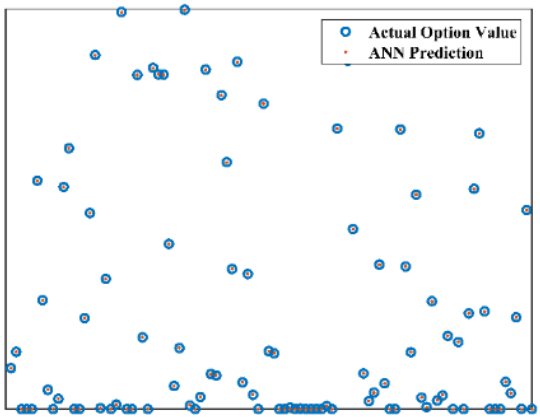

Fig 7: Comparison between ANN and Actual option value

For PINN application, we solve Eq. 5 with the initial and boundary condition (6) and (7). To minimize the total L in Eq. 16, we have applied algorithm 1 with learning rate, $\eta = 0.001$ and number of iterations is 10000. In hidden layers we have applied "tanh" as an activation function and for output layer "ReLU". While updating the weights and biases, we have to determine the gradient of L in Eq. 16, with respect to inputs, weights and biases. To achieve this, we use deep learning frameworks, "TensorFlow" called automatic differentiation package in "Python" environment. It gives us the flexibility to avoid large number derivations involving the values or numerical discretization while computing derivatives of all orders in space–time. Again, to retrieve the actual solution, we have to multiply PINN generated solution $\psi_{PINN}(\zeta, \tau)$ by the strike price that is $V_{PINN}(S, t) = K \times \psi_{PINN}(\zeta, \tau)$.

In fig. 8, we demonstrated the PINN outcomes graphically and the training loss for BSM model with the PINN is also shown in fig.9. Started from the initial loss: 3.888, after 10,000 iteration loss is 9.48×10^{-5}. To perform the error analysis, we plotted the value of option price at $t = 0$ in fig. 10. From fig. 10, one can easily observe that PINN outcomes are superior to ANN and very close to the approximate results. Sometimes PINN predictions are more accurate than the Numerical and approximate results when option value are zero or close to zero.

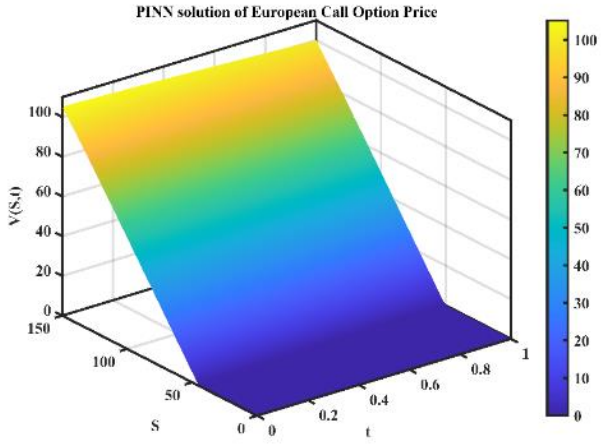

Fig 8: PINN Solution of BSM

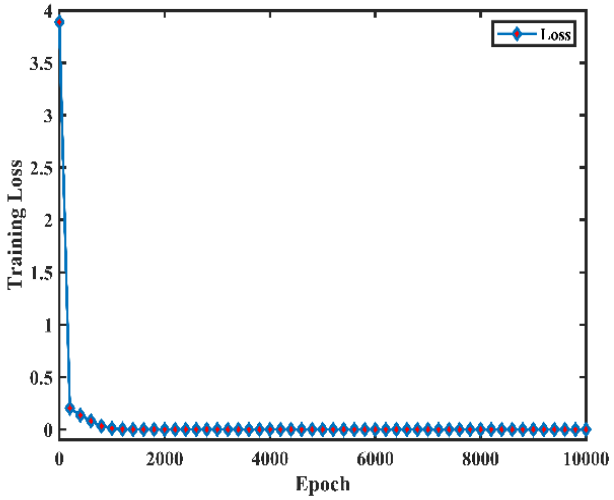

Fig 9: PINN training loss with Epoch

The performance time of ANN and PINNs dependent on CPU\GPU configuration. All computations were performed on

a workstation with an Intel ® core ™ i7-1270H CPU @2.30 GHz, 24GB RAM, and an NVIDIA ® RTX 3060 GPU with 4GB VRAM using python 3.11, TensorFlow-2.15 and MATLAB-2023.

Since we choose random weights and biases in every calculation training time and loss update will vary. Using trial and error methods, for the best optimum mode we have found the following optimal solution present in table.3.

Table 3: Convergence time of ANN and PINN

Model	**Hidden Layer**	**Neuron / layer**	**Activation function**	**Training time(s)**	**Epochs**	**MSE**
ANN	2	64	ReLu	155	5000	3.5×10^{-8}
PINN	5	20	Tanh	447	10000	9.48×10^{-5}

Table 4: Option price of European call option

Stock price	Analytical	Numerical (FDM)	ANN	PINN
73.5	29.15398	29.15717	28.50557	29.10508
88.5	44.15398	44.15717	43.56467	44.14815
103.5	59.15398	59.15717	58.65866	59.15401
118.5	74.15398	74.15706	73.78530	74.14369
133.5	89.15398	89.15623	88.82779	89.13826
148.5	104.15398	104.15422	103.55378	103.98105
150.0	105.65398	105.65398	104.95925	105.58664

In table 4, we represents tabular form of option price value for unseen data (not used for training). From this table we can observe that PINN shows excellent results in comparison to ANN results.

VII. Model Achievements & Limitations

This PINN frameworks are easily extendable to other financial PDEs and Stochastic PDEs, such as Heston or local volatility models with minor modification. PINNs produce smooth differentiable solutions that facilitate analytical post-processing. However, certain limitations remain Training PINNs is computationally more expensive than classical solvers for low dimensional PDEs. This need more careful balancing loss function terms to ensure optimality.

In practical scenarios, PINN can predict option prices that align closely with real data by embedding directly into the loss function. While the Numerical and ANN methos may faster than PINN may deviate in volatile or sparse-data conditions due to lack explicit physics enforcement.

Conclusion

In this paper, we investigated multilayer neural network techniques ANN and PINN for solving Black-Scholes option pricing model. Through the use of these comprehensive computational methods, we exhibited how they may improve the accuracy of option pricing and offer a better understanding of financial modeling. We have compared the outcomes of ANNs and PINNs with numerical and approximate findings. ANN gives the prediction based on the trained data but in PINNs techniques, we don't need any previous data. Thus, PINN may serve as a substitute for conventional numerical techniques. Additionally, PINNs are mesh-free procedures, whereas the numerical method sometimes fails to construct mesh in complex boundaries. PINNs have been shown to be a superior method over ANNs. The results highlight the importance of integrating machine learning approaches into conventional financial models and open the door for more studies that may improve these methods even more.

Table: Nomenclature, Greek symbols and Aberration

Nomenclature		Greek Symbols	
K	Strike price	σ	Volatility
S	Stock price	ψ	Non-dimensional option value
T	Maturity time	ζ	Non-dimensional stock price
V	Option value	τ	Non-dimensional time
L	Loss function	θ	activation function
r	Risk free interest rate	β	Biases
t	time	Abirritation	
m, n	constant	PINN	Physics Informed Neural Network
f	Activation function	ADAM	Adaptive Moment Estimation
w	Weights	ANN	Artificial Neural Network
-	-	BS	Black-Scholes
-	-	MSE	Mean Square Error
-	-	FDM	Finite difference Method

References

1. Anwar, M.N. and Andallah, L.S., 2018. A study on numerical solution of Black-Scholes model. *Journal of Mathematical Finance*, *8*(2), pp.372-381.
2. Ankudinova, J. and Ehrhardt, M., 2008. On the numerical solution of nonlinear Black–Scholes equations. Computers & Mathematics with Applications, 56(3), pp.799-812.
3. Heider, P., 2010. Numerical methods for non-linear Black–Scholes equations. Applied Mathematical Finance, 17(1), pp.59-81.
4. Roul, P. and Goura, V.P., 2020. A sixth order numerical method and its convergence for generalized Black–Scholes PDE. Journal of Computational and Applied Mathematics, 377, p.112881.

5. Ahmad, H., Khan, M.N., Ahmad, I., Omri, M. and Alotaibi, M.F., 2023. A meshless method for numerical solutions of linear and nonlinear time-fractional Black-Scholes models. *AIMS Math*, *8*(8), pp.19677-19698.
6. Deng, S. and Gu, S., 2021. Optimal conversion of conventional artificial neural networks to spiking neural networks. arXiv preprint arXiv:2103.00476.
7. Blechschmidt, J. and Ernst, O.G., 2021. Three ways to solve partial differential equations with neural networks—A review. *GAMM-Mitteilungen*, *44*(2), p.e202100006.
8. Khan, I., Raja, M.A.Z., Shoaib, M., Kumam, P., Alrabaiah, H., Shah, Z. and Islam, S., 2020. Design of neural network with Levenberg-Marquardt and Bayesian regularization backpropagation for solving pantograph delay differential equations. *IEEE Access*, *8*, pp.137918-137933.
9. Fang, Z., 2021. A high-efficient hybrid physics-informed neural networks based on convolutional neural network. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(10), pp.5514-5526.
10. Cuomo, S., Di Cola, V.S., Giampaolo, F., Rozza, G., Raissi, M. and Piccialli, F., 2022. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, *92*(3), p.88.
11. Imran, M. M. H., Hossain, M. S., Billah, M. M., & Farzana, H. (2024). On the applications of neural network technique for electro-viscoplastic Casson hybrid ferrofluid with a permeable channel. *International Journal of Thermofluids*, *24*, 100976.
12. Imran, M. M. H., Mojumdar, S., & Uddin, M. J. (2025). Investigation of magnetized slip flow of hybrid nanofluid past nonlinearly radiative sheet with Newtonian heating: Physics informed neural network simulation. *Proceedings of the Institution of Mechanical Engineers, Part N: Journal of Nanomaterials, Nanoengineering and Nanosystems*, 23977914251353307.
13. Sirignano, J., & Cont, R. (2019). Universal features of price formation in financial markets: Perspectives from deep learning. *Quantitative Finance*, 19(9), 1449-1459.
14. Eskiizmirliler, S., Günel, K. and Polat, R., 2021. On the solution of the black–scholes equation using feed-forward neural networks. *Computational Economics*, *58*, pp.915-941.
15. Chen, Y., Yu, H., Meng, X., Xie, X., Hou, M. and Chevallier, J., 2021. Numerical solving of the generalized Black-Scholes differential equation using Laguerre neural network. *Digital Signal Processing*, *112*, p.103003.
16. Gonon, L., 2023. Random feature neural networks learn Black-Scholes type PDEs without curse of dimensionality. *Journal of Machine Learning Research*, *24*(189), pp.1-51.
17. Santos, D.D.S. and Ferreira, T.A.E., 2024. Neural Network Learning of Black-Scholes Equation for Option Pricing. *arXiv preprint arXiv:2405.05780*.
18. Shinde, A.S. and Takale, K.C., 2012. Study of Black-Scholes model and its applications. Procedia Engineering, 38, pp.270-279.
19. Capiński, M. and Kopp, E., 2012. The Black–Scholes Model. Cambridge University Press.
20. Coelen, N., 2002. Black-Scholes Option Pricing Model. Recuperado de http://ramanujan. math. trinity. edu/tumath/research/studpapers/s11. pdf.
21. Emery, D.R., Guo, W. and Su, T., 2008. A closer look at Black–Scholes option thetas. Journal of economics and finance, 32, pp.59-74.
22. Nasir, S., Berrouk, A.S., Gul, T. *et al.* Develop the artificial neural network approach to predict thermal transport analysis of nanofluid inside a porous enclosure. *Sci Rep, vol.* **13**, ID.21039, 2023.
23. Mohamed E. Ghoneim, Zeeshan Khan, Samina Zuhra, Aatif Ali, Elsayed Tag-Eldin,Numerical solution of Rosseland's radiative and magnetic field effects for Cu-Kerosene and Cu-water nanofluids of Darcy-Forchheimer flow through squeezing motion, Alexandria Engineering Journal,Vol.64, pp. 191-204, 2023.
24. Khan, Z., Zuhra, S., Islam, S. *et al.* Modeling and simulation of Maxwell nanofluid flows in the presence of Lorentz and Darcy–Forchheimer forces: toward a new approach on Buongiorno's model using artificial neural network (ANN). *Eur. Phys. J. Plus*, vol. **138**, no.107, 2023.
25. Zhang, Z., 2018, June. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)* (pp. 1-2). Ieee.
26. Tato, A. and Nkambou, R., 2018. Improving adam optimizer.
27. Cai, S., Wang, Z., Wang, S., Perdikaris, P. and Karniadakis, G.E., 2021. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, *143*(6), p.060801.
28. Raissi, M., Perdikaris, P., and Karniadakis, G. E., 2019, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," J. Comput. Phys., 378, pp. 686–707.10.1016/j.jcp.2018.10.045.

Md. Majibul Hasan Imran is a Lecturer in the Department of Mathematics at the American International University-Bangladesh (AIUB), Dhaka. He earned his BSc and MSc in Applied Mathematics from the University of Dhaka. His research lies at the intersection of fluid dynamics and scientific machine learning, with a focus on Artificial Neural Networks (ANN) and Physics-Informed Neural Networks (PINN) for solving and inferring parameters in partial differential equations. He has worked on boundary-layer and magnetohydrodynamic (MHD) flows, hybrid nanofluids, turbulent channel flow, fractional and thermo-convective stability problems, option-pricing models based on the Black–Scholes equation, and reaction–diffusion systems for biomedical transport. His current projects emphasize data-driven modeling, numerical simulation, and high-performance computing for complex multiphysics flows.

Dr. Muhammad Sajjad Hossain is an Associate Professor in the Division of Mathematics, Department of Arts and Sciences at Ahsanullah University of Science and Technology with a Ph.D. in Mathematics from Jahangirnagar University, Savar, Dhaka, Bangladesh with a research focus on MHD convective flows in complex enclosures using the finite element method. He had also completed his M.Phil. in Mathematics from Bangladesh University of Engineering and Technology (BUET), Bangladesh following an M.S. and B. Sc. (Honors) at the University of Chittagong, Bangladesh. Before joining AUST he was a faculty member of University of Asia Pacific, Dhaka, Bangladesh for more than 02 (two) years. His research spans computational fluid dynamics, magnetohydrodynamics, nanofluids, computational hemodynamics, and machine-learning methods including ANN, DNN, and PINN. He earned Gold Medals for securing first position in MS and B. Sc. (Honors) level and also earned UGC merit scholarship in MS level. Dr. Hossain has over 17 years of university-level teaching experience across institutions like UAP, AUST and BUET. His work has been widely published, with more than 41 research articles in Scopus and Web of Science-indexed journals, many in Q1-ranked journals by Elsevier, MDPI and Springer Nature. He also explores applications of artificial intelligence in engineering analysis. Dr. Hossain serves as a reviewer for several international journals and has led and participated in multiple research projects funded by AUST. At AUST, he also coordinates the M.S. in Mathematics program and contributes to academic governance and curriculum initiatives. Already five (05) students of MS in Mathematics at AUST completed their degree and four (04) students are continuing their thesis under supervision of Dr. Hossain. As an internal member, he examined many MS students' thesis. He is currently supervising two Ph.D. students at SUST and JU contributing to the academic development of the next generation of researchers.

Dr. Md. Abdul Alim is a Professor of Mathematics at Bangladesh University of Engineering and Technology (BUET) and Fellow of the Bangladesh Academy of Sciences (FBAS). A pioneer in nanofluid-based solar collector research, he has made significant contributions to computational fluid dynamics, heat and mass transfer, combustion, radiation, and porous media flows. He has over 450 publications and 475+ international reviews. He has supervised 20 Ph.D. and 37 M.Phil./M.Sc. theses. Dr. Alim earned his Ph.D. from Loughborough University, UK, and has served as Head of Department and Provost at BUET. He is a life member of the Bangladesh Mathematical Society and actively contributes to global scientific and educational advancement.