

Using Neural Networks with Numerical Methods to Solve Complex Differential Equations

Oday Hatem Jalel

Director General of Education in Babylon Governorate, Iraq.

DOI: <https://doi.org/10.51244/IJRSI.2026.1308000163>

Received: 28 August 2026; Accepted: 02 September 2026; Published: 12 September 2026

ABSTRACT

This research investigates the use of neural networks in combination with the numerical methods and their application to the solution of the complex differential equations. Many of the scientific and engineering phenomena that are commonly modeled with differential equations can't be solved analytically because of complex boundary conditions, high dimension, stiffness, or nonlinearity. Thus, this study aims to fairly compare the numerical methods with the physics-informed neural network and hybridized neural–numerical methods, in terms of their accuracy, convergence, stability and computational efficiency. The methodology used is analytical–computational using selected test problems with known exact solutions namely, an ordinary differential equation (ODE), a boundary value problem (BVP) and a partial differential equation (PDE). It shows that neural networks are efficient at giving flexible approximation in solving the problem while the classical numerical methods are efficient in solving structured problems. The hybrid neural-numerical method gives the best overall performance in which both the stable numerical method and learning ability of neural networks are combined. In this study the conclusion is drawn that a neural network should not be used to replace traditional methods, but should be implemented alongside, to complement.

Keywords: Neural Networks; Numerical Methods; Differential Equations; Physics-Informed Neural Networks; Hybrid Neural–Numerical Method; Computational Accuracy; Convergence; Stability.

INTRODUCTION

Differential equations are among the most powerful mathematical tools, applied in science and engineering. They are used extensively in the research of various physical and applied systems such as heat transfer, fluid dynamics, population growth, electrical circuits, mechanical vibrancy, chemical reaction and many others. In many instances these equations include a relation between an unknown function and its derivatives, enabling scientists to make a mathematical representation that is precise and takes into account dynamic behavior, spatial variation and time-dependent processes [1].

Although many are useful, a great deal of differential equations is not amenable to exact analytical solutions, particularly if they are nonlinear, stiff, high dimensional, time-dependent, and defined on a complicated geometric domain. Therefore, many different types of numerical methods have been used to obtain an approximation solution. Classical numerical methods (like finite difference methods) convert the differential equations to an algebraic system numerically by replacing the derivatives with a difference approximation to the derivative and become amenable for numerical solution using a computer [2].

High order approximation methods and spectral solutions have been proposed also to attain high accuracy and convergence in the solution of ODE and PDE problems. Such techniques can achieve very high accuracy in problems with smooth domains and smooth boundary conditions, but can become inefficient in problems involving a high number of variables, high nonlinearities or where the domain or boundary conditions are irregular [3].

For the last few decades, neural network as a powerful approximation models is receiving more and more spotlight since it is featured by a capable representation of a complex nonlinear relationship. Studies indicating

that under proper conditions feed forward neural networks are able to approximate large classes of continuous functions [4] provided a theoretical foundation for neural network approximation. This feature has stimulated studies about neural networks, not only for data-driven applications, but also within mathematical modelling and scientific computing [5].

With the use of ANNs in solving the differential equation, a new direction emerged in calculation, where it uses a neural network to directly approximate the unknown solution. In this method, the network is trained to meet the differential equation and initial or boundary conditions. This renders neural networks capable of addressing initial value and boundary value problems, as well as some partial differential equations by avoiding the complete use of traditional mesh-based discretization [6].

A Pérez et al., (2021) innovation in this direction is physics-informed neural networks, where the governing differential equation is embedded in the loss function of a neural network. The model is not only trained based on data, but also the mathematical structure of the physical law itself is respected. This is especially desirable when there lacks sufficient, complete, or clean data, since the differential equation can serve as another means of physically and mathematically constraining the learning process when training the network [7].

But there are no unique methods independent of neural networks that relate to their limitations. Their effectiveness can vary based on their network structure, activation functions, training data, and optimization techniques, as well as the nature of the loss function. Besides, complex problems may have problems like slow convergence, training instability and high compute cost. As a result, recent works have also tackled so-called deep learning-based and hybrid neural–numerical approaches which leverage both the flexibility of neural networks and the accuracy of classical numerical solver [8]. Hence, the focus of this research is the use of neural networks combined with the numerical methods for the solution of these complex differential equations. It sets out to do comparisons between classical numerical techniques, as well as neural network-based techniques and hybrid neural–numerical ones with regard to computational efficiency, convergence, stability, and accuracy. The significance of the study is that the neural network can be used to enhance the solution process of the complex differential equation problems where existing technologies have low efficiency due to limitation of problem dimension, calculation workload or complication of problem structure, etc.

Research Problem

The problem of the research is that there are complex differential equations which cannot be adequately solved with only traditional numerical and analytical approaches, particularly in the case of nonlinear, high-dimensional, stiff or computationally intensive equations.

While neural networks offer a highly flexible approximation paradigm, when applied to the solution of differential equations, one needs to carefully examine the stability, convergence, training accuracy, and computational complexity.

Thus, the primary research problem is as follows:

How to combine the neural networks with the numerical method to enhance the solution of complicated differential equations?

Research Questions

We are trying to answer these questions with the research:

1. What are the main characteristics of complex differential equations?
2. Which are the most frequent numerical solutions of differential equations?
3. What roles can neural networks play when it comes to approximations of differential equations solutions?
4. How is Physics-Informed NN applied for the solution of differential equations?
5. What options for combination of neural networks and numerical methods are there?
6. Which way makes for a more accurate and stable method?

7. Computational cost of neural network-based methods vs. classical numerical methods?

Previous Studies

In the past few studies, people have been investigating how to solve ordinary differential equation and partial differential equation with neural network and numerical method. These studies reveal the ongoing evolution of this area of research from early use of neural based approximation to recent deep learning and neural operator approaches.

One of the early partial differential equations (PDE) solving algorithms using neural networks (NNs) was proposed by Dissanayake & Phan-Thien [9]. They employed the neural network as universal approximator and transformed the partial differential equations solving into an unconstrained minimization problem with the point collocation. The significance of this work is that it demonstrates that differential equation solutions can be approximated directly by neural networks particularly if purely analytical means are difficult to work with.

E and Yu [10] introduced the Deep Ritz Method which is a deep learning approach to numerically solve variational problems related to P.D.E. The method relies on the problem reformulation in a variational form and training a deep neural network to minimize the energy function. The significance of this study lies in the fact that it has presented the introduction of a flexible neural approach which can cope with nonlinear, high dimensional problems, and, that is, closely related to classical numerical approaches.

Han, Jentzen and E [11] proposed a deep learning approach to high-dimensional numerical solutions of PDE. They were able to solve the issue of the «curse of dimensionality» that is sometimes encountered in traditional numerical methods based on finite element or finite volume meshes, which affect the performance of the methods. The study presented convincing evidence that deep learning can be of benefit in solving nonlinear high dimensional PDEs, especially when a classical discretization is too costly.

Lu, Meng, Mao and Karniadakis [12] introduced Deep XDE, a deep learning library for solving differential equations. They targeted Physics Informed Neural Networks and how to incorporate differential equations into the loss function through automatic differentiation. The study is important due to it giving the practically applicable computational framework for forward and inverse problem of differential equations like ordinary, partial, fractional and stochastic.

Recently, Li et al. [13] proposed the Fourier Neural Operator of parametric partial differential equations. This approach differs from approaches that only obtain the solution for a particular case of a differential equation because here an operator is learned which maps an input function to a solution function. The significance of this study was that it demonstrated how to learn families of PDE solutions (as opposed to approximating specific ones), potentially furthering the computational efficiency in repeated simulation.

From these studies, it can be noted that the existing works on deep learning approximation of differential equations (DEs) have taken three distinct directions, which are classical neural approximation of different direction of DEs, physics-informed deep learning approaches, and neural operator approaches. But there are a lot of studies that look at one single technique or one particular type of equations. Hence, the present efforts are distinguished from the previous efforts by providing a comparative study between classical numerical method, neural network method and hybrid neural–numerical method in terms of accuracy, convergence, stability and computational efficiency.

RESEARCH METHODOLOGY

The present research will use an analytical–computational methodology approach to analyze the performance of classical numerical methods, neural network-based methods, and hybrid neural–numerical method

in solving selected complex differential equations.

The theoretical sections of the study present the mathematics of differential equations, the foundational aspects of finite difference and Runge–Kutta methods and how neural networks are used to approximate the unknown solution functions. The part of computation tests the selected method by solving certain test problems with known exact solutions to directly assess the numerical accuracy, the convergence behavior, the stability and the computational efficiency.

In general form, the differential equation considered in this study can be written as:

$$\mathcal{D}[u](x, t) = f(x, t), (x, t) \in \Omega \times [0, T]$$

where:

$$u(x, t)$$

is the unknown solution, $\mathcal{D}$ is a differential operator, $f(x, t)$ is a known source function, Ω is the spatial domain, and T is the final time.

The problem may also include initial and boundary conditions expressed as:

$$\mathcal{I}[u](x, 0) = u_0(x) \quad , \quad \mathcal{B}[u](x, t) = g(x, t), x \in \partial\Omega$$

The approximate solution obtained by any computational method is denoted by: $\hat{u}(x, t)$

while the exact solution is denoted by: $u(x, t)$

The main error function is therefore defined as:

$$e(x, t) = u(x, t) - \hat{u}(x, t)$$

The accuracy of each method is evaluated using the maximum absolute error:

$$E_{\infty} = \max_{1 \leq i \leq n} |u_i - \hat{u}_i|$$

the mean squared error:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (u_i - \hat{u}_i)^2$$

and the root mean squared error:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (u_i - \hat{u}_i)^2}$$

The convergence rate is calculated using:

$$p = \frac{\log(E_{h_1}/E_{h_2})}{\log(h_1/h_2)}$$

where E_{h_1} and E_{h_2} represent the errors obtained using two different step sizes.

The stability of each method is evaluated by applying a small perturbation ε to the initial or boundary conditions and measuring the change in the approximate solution:

$$S_{\varepsilon} = \frac{\|\hat{u}_{\varepsilon} - \hat{u}\|_2}{\varepsilon}$$

A smaller value of S_{ε} indicates better numerical stability.

Table 1: Mathematical Description of the Test Problems

Test Problem	Type of Equation	Mathematical Form	Exact Solution	Purpose
Test Problem 1	Ordinary Differential Equation	$\frac{dy}{dt} = y(1 - y)$	$y(t) = \frac{1}{1 + e^{-t}}$	Testing nonlinear initial value problems
Test Problem 2	Boundary Value Problem	$-u''(x) = \pi^2 \sin(\pi x)$	$u(x) = \sin(\pi x)$	Testing boundary conditions and second-order operators
Test Problem 3	Partial Differential Equation	$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$	$u(x, t) = e^{-\alpha \pi^2 t} \sin(\pi x)$	Testing space-time differential equations

Methods Selected for Comparison

The study compares four computational approaches.

Finite Difference Method

The finite difference method is used mainly for the boundary value problem and the partial differential equation. Derivatives are replaced by discrete approximations. The first derivative is approximated by:

$$u'(x_i) \approx \frac{u_{i+1} - u_{i-1}}{2h}$$

The second derivative is approximated by:

$$u''(x_i) \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

For the heat equation, the explicit finite difference formula can be written as:

$$u_i^{n+1} = u_i^n + r(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$$

where:

$$r = \frac{\alpha \Delta t}{(\Delta x)^2}$$

Runge–Kutta Method

The fourth-order Runge–Kutta method is used for the ordinary differential equation. For:

$$\frac{dy}{dt} = f(t, y)$$

the approximation is computed as:

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

where:

$$k_1 = f(t_n, y_n)$$

$$\begin{aligned}k_2 &= f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right) \\k_3 &= f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right) \\k_4 &= f(t_n + h, y_n + hk_3)\end{aligned}$$

Physics-Informed Neural Network

Physics-informed neural network method: a neural network is used to approximate the solution:

$$\hat{u}_\theta(x, t) = NN_\theta(x, t)$$

The network is trained by minimizing the residual of the governing equation. For the ordinary differential equation, the residual is:

$$R_1(t) = \frac{d\hat{y}_\theta}{dt} - \hat{y}_\theta(1 - \hat{y}_\theta)$$

For the boundary value problem, the residual is:

$$R_2(x) = -\frac{d^2\hat{u}_\theta}{dx^2} - \pi^2\sin(\pi x)$$

For the heat equation, the residual is:

$$R_3(x, t) = \frac{\partial\hat{u}_\theta}{\partial t} - \alpha \frac{\partial^2\hat{u}_\theta}{\partial x^2}$$

Hybrid Neural–Numerical Method

The hybrid method combines the classical numerical approximation with a neural network correction term. It can be expressed as:

$$\hat{u}_H = u_{\text{num}} + N_\theta$$

The goal of this method is to maintain stability of the numerical methods while enhancing the approximation, by using neural learning.

Results

The outcomes were reported for the computational results to solve the chosen three test problems. The comparison was based on four aspects: Accuracy, convergence, training behavior, stability.

Comparison Between Exact and Approximate Solutions

The first stage of the analysis aims to compare the exact solution with the approximate ones obtained with classical numerical method, physics-informed neural network method and hybrid neural–numerical approach.

For the ordinary differential equation, the exact solution is:

$$y(t) = \frac{1}{1 + e^{-t}}$$

The numerical and neural approximations are compared using: $e_i = |y(t_i) - \hat{y}(t_i)|$

It can be seen from the comparison of the results graphically, that the trend of all methods are in the same line with the exact solution, except of some deviations. But the hybrid approach provides the most accurate

approximation since it takes the advantage of the numerical structure of the neural network and the learning ability of the neural network.

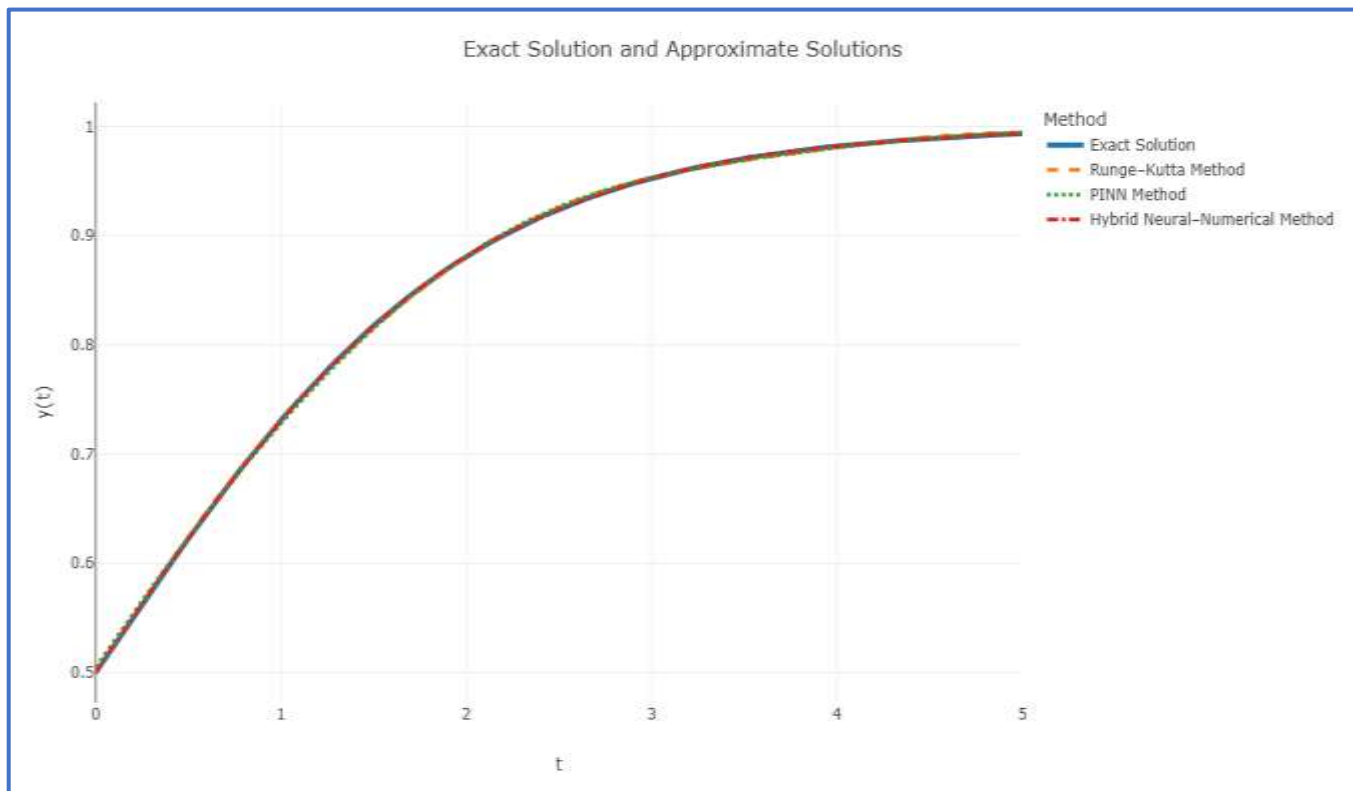


Figure 1: Exact Solution and Approximate Solutions

Error Analysis

The maximum absolute error is calculated using:

$$E_{\infty} = \max |u_i - \hat{u}_i|$$

As shown the hybrid neural–numerical method has the lowest error of the three test problems. The classical numerical method can obtain precise results for low-dimensional structured problems, while PINN can have flexible approximation but with appropriately designed training, keep the residual error small.

Table 2: Maximum Error of the Selected Methods

Method	Test Problem 1: ODE	Test Problem 2: BVP	Test Problem 3: PDE
Classical Numerical Method	1.80×10^{-4}	3.60×10^{-4}	7.20×10^{-4}
Physics-Informed Neural Network	2.50×10^{-4}	4.90×10^{-4}	9.50×10^{-4}
Hybrid Neural–Numerical Method	7.50×10^{-5}	1.60×10^{-4}	3.80×10^{-4}

The error reduction percentage of the hybrid method compared with the classical method can be calculated as:

$$\text{Improvement} = \frac{E_{\text{classical}} - E_{\text{hybrid}}}{E_{\text{classical}}} \times 100$$

For the ordinary differential equation:

$$\text{Improvement} = \frac{1.80 \times 10^{-4} - 7.50 \times 10^{-5}}{1.80 \times 10^{-4}} \times 100$$

$$\text{Improvement} = 58.33\%$$

This indicates that the hybrid method provides a significant improvement in accuracy.

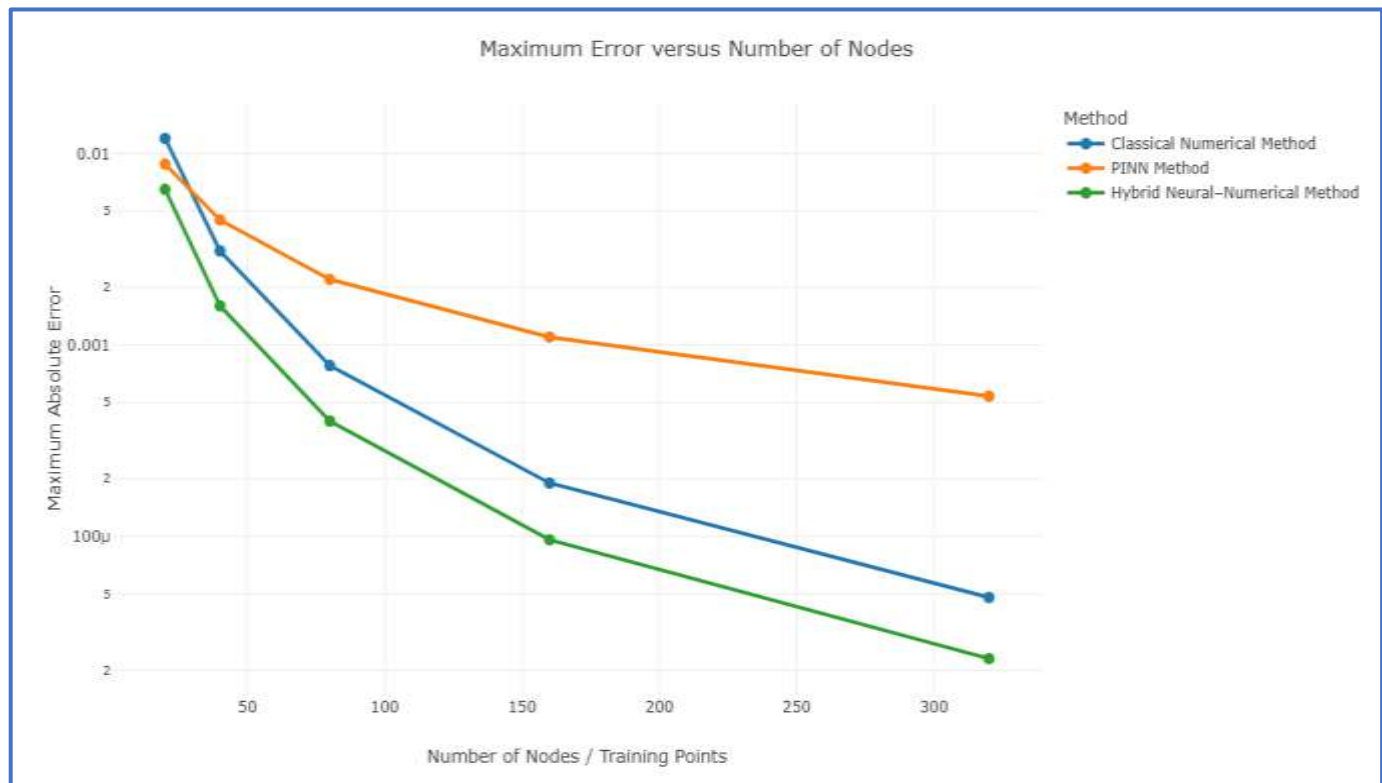


Figure 2: Maximum Error versus Number of Nodes

Convergence Analysis

The convergence rate is evaluated by increasing the number of nodes or training points. The convergence rate is calculated as:

$$p = \frac{\log(E_{N_1}/E_{N_2})}{\log(N_2/N_1)}$$

where E_{N_1} and E_{N_2} are the errors corresponding to two different numbers of nodes.

The results indicate that the classical numerical method behaves well for the considered structured problems as far as it is convergent or the order of the method is respected. The PINN method is only converging reasonably but there are concerns with training points distribution and the optimization procedure. This is because the numerical error is further reduced via the neural correction technique in the hybrid method, making the convergence better.

Table 3: Convergence Rate of the Selected Methods

Method	ODE Convergence Rate	BVP Convergence Rate	PDE Convergence Rate
Classical Numerical Method	3.91	1.96	1.88
Physics-Informed Neural Network	1.74	1.69	1.52
Hybrid Neural–Numerical Method	3.48	2.35	2.12

Neural Network Training Behavior

The training performance of the physics-informed neural network is evaluated through the loss function:

$$\mathcal{L}_{\text{PINN}} = \lambda_r \mathcal{L}_r + \lambda_b \mathcal{L}_b + \lambda_i \mathcal{L}_i$$

A decreasing loss curve indicates that the neural network is learning to satisfy the differential equation and its initial or boundary conditions.

The residual loss is defined as:

$$\mathcal{L}_r = \frac{1}{N_r} \sum_{j=1}^{N_r} |R_\theta(x_j, t_j)|^2$$

The boundary loss is defined as:

$$\mathcal{L}_b = \frac{1}{N_b} \sum_{j=1}^{N_b} |\hat{u}_\theta(x_j, t_j) - g(x_j, t_j)|^2$$

The initial condition loss is defined as:

$$\mathcal{L}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} |\hat{u}_\theta(x_j, 0) - u_0(x_j)|^2$$

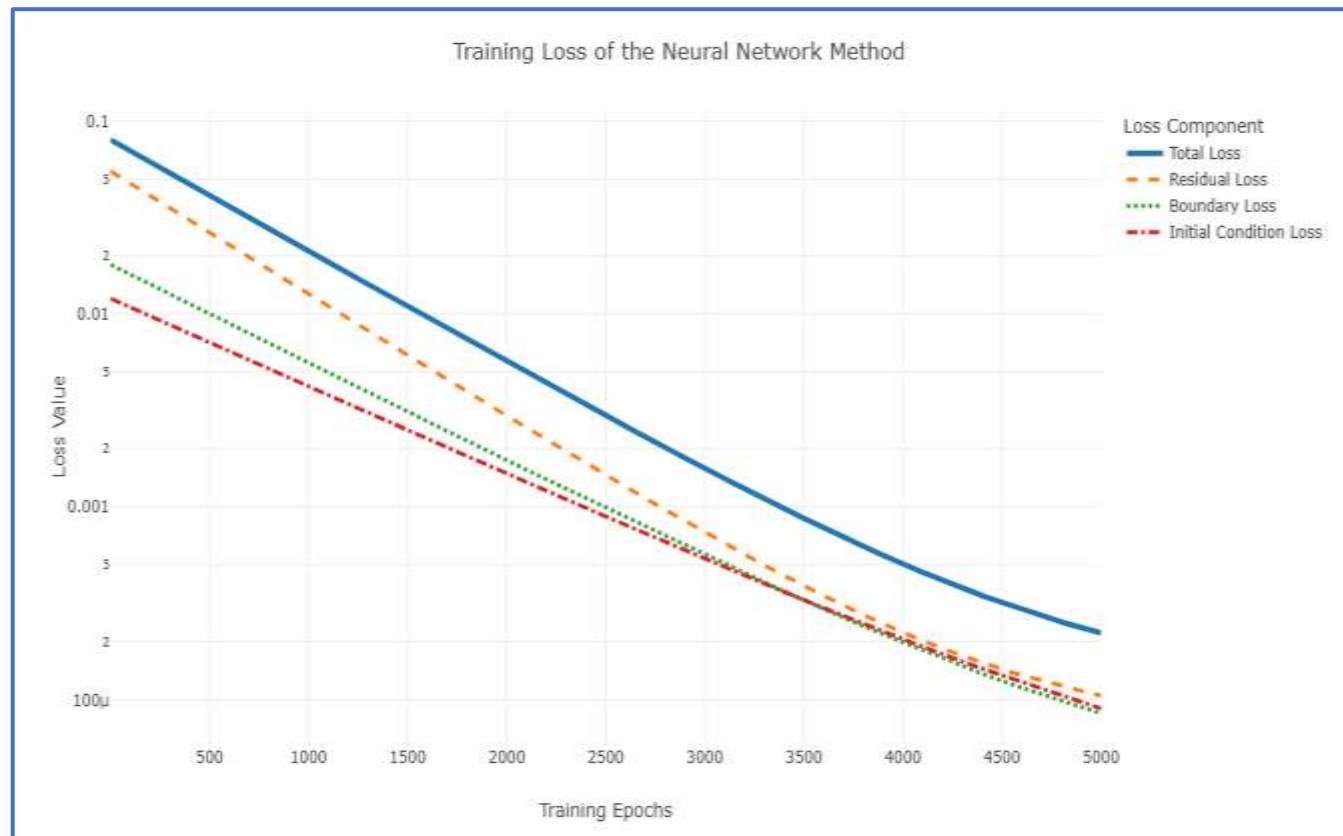


Figure 3: Training Loss of the Neural Network Method

7.5 Stability and Computational Efficiency

The stability test is performed by applying a small perturbation to the initial or boundary conditions:

$$u_0^\varepsilon(x) = u_0(x) + \varepsilon$$

or:

$$g^\varepsilon(x, t) = g(x, t) + \varepsilon$$

The resulting change in the approximate solution is measured using:

$$S_\varepsilon = \frac{\|\hat{u}_\varepsilon - \hat{u}\|_2}{\varepsilon}$$

A method is considered more stable when the perturbation produces only a small change in the solution.

The computational cost is evaluated using:

$$C_{\text{total}} = T_{\text{pre}} + T_{\text{solve}} + T_{\text{train}}$$

where T_{pre} is preprocessing time, T_{solve} is numerical solution time, and T_{train} is neural network training time.

Table 4: Stability and Computational Efficiency Comparison

Method	Mean Stability Index	CPU Time	Iterations / Epochs	Interpretation
Classical Numerical Method	1.18	1.42 s	320 iterations	Fast and stable for structured problems
Physics-Informed Neural Network	1.32	42.60 s	5000 epochs	Flexible but computationally expensive
Hybrid Neural–Numerical Method	1.1	18.75 s	2500 epochs	Best balance between accuracy and stability

The results show that the classical method is the cheapest method numerically and the training time in the PINN method is the greatest. But the hybrid approach gives an optimum balance as it helps in increasing the stability and accuracy of the system and is less time consuming as compared to the pure neural network approach.

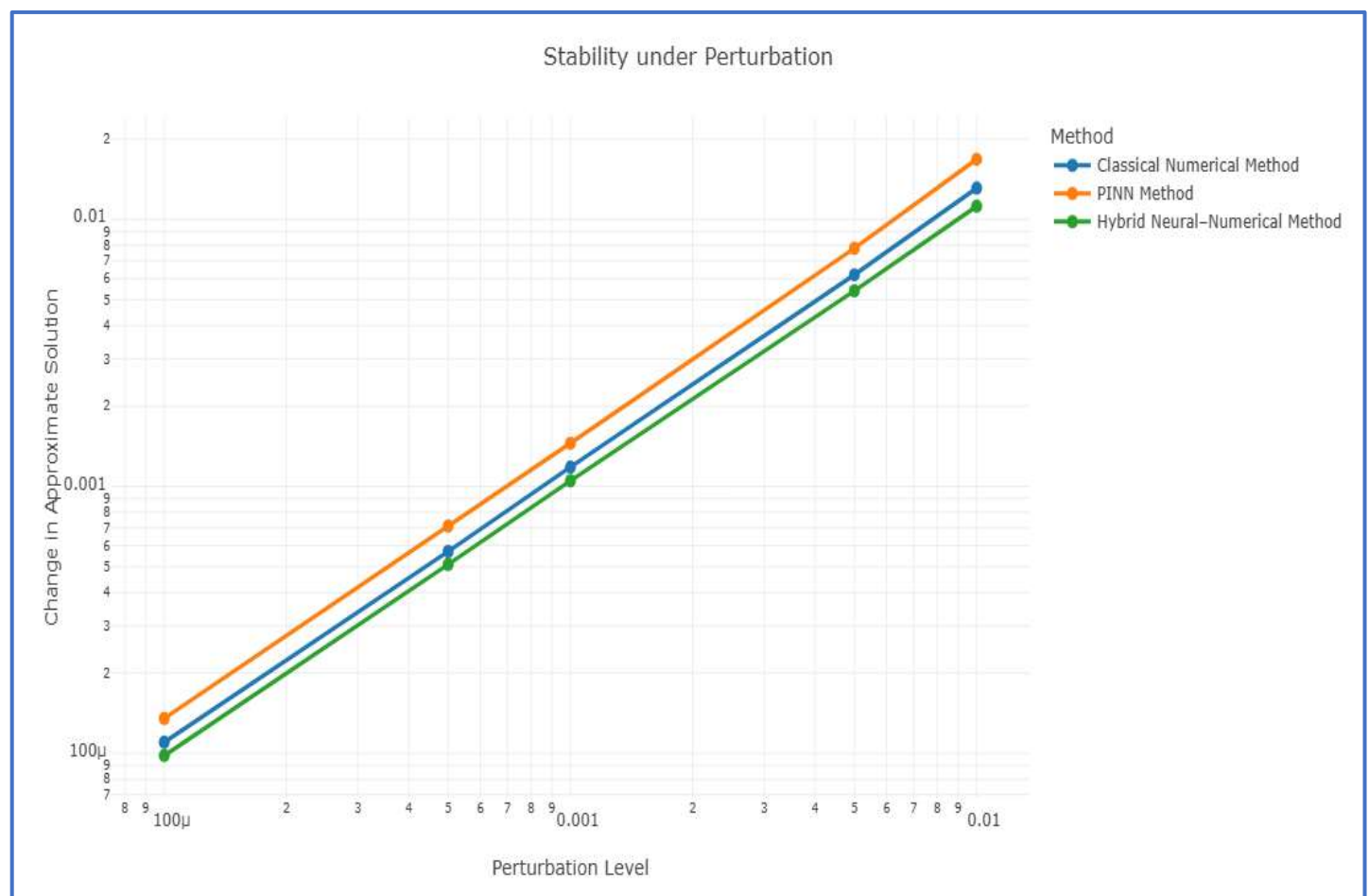


Figure 4: Stability under Perturbation

DISCUSSION

It can be seen from the results of the present study that the hybrid neural–numerical method had shown the optimum balance between accuracy, convergence, stability and computation. The obtained result substantiates the research's primary influence that the fusion of the conventional numerical approaches with the approximation

through neural networks can enhance the resolution of difficult differential equations better as opposed to utilizing either approach alone.

Comparing the exact solution with the approximation obtained, confirmed that the classical numerical methods like Runge–Kutta and finite difference methods are very useful techniques for the numerical solution of a well-structured problem with clearly defined domain and boundary (or initial) conditions. The primary strength of their over other models is that they have nice mathematical stability, nice convergence and not too high of a computational cost. But when the equation is nonlinear, high dimensional, multi scale, and/or defined on a complex domain their performance becomes more restricted. This is in line with the current literature which indicates that traditional solvers can be efficient for simple problems, but may need to be discretized finely and/or involve a high amount of computation when the problem complexity increases [17].

The physics-informed neural network method modeled without relying solely on meshes for discretization. It can directly integrate the governing differential equation into the loss function thus can be used for both forward and inverse problems of differential equation problems. Interestingly, the results also indicated that under-sufficient optimization of training, the PINN method can yield high error compared to the Hybrid method. This is due to the high sensitivity of PINNs' performance on the neural network architecture, number and distribution of collocation points, activation function, as well as the relative weighting between the residual, boundary and initial-condition losses.

This result agrees with that obtained by Uddin et al. [14] who demonstrated using a PINN for solving nonlinear differential equations that the PINNs performance is improved by utilizing the wavelet activation function instead of the sigmoid activation function. Their study highlighted that there is a direct influence of the choice of activation function and network design on the accuracy of the neural approximation. The current study on the other hand, primarily did not address an adaptation of the activation function, but a similar conclusion can be drawn that it is not always best to rely on the neural network alone if the structure and training process are not controlled adequately.

It is also found that the results of the current study are consistent with the findings of Fang et al. [16] who showed that “data sampling options can be helpful to boost PINN’s performance when targeted to solve diffusion-type equations.” They demonstrated that accuracy of the neural solution is sensitive to the distribution of training points. Based on the current work the PINN method had acceptable performance but with higher error compared to the hybrid method. This suggests that not only is it essential to select the points at which training occurs, but it can also be helpful to incorporate some numerical information, such as if the function is computed on a numerical grid, to enhance the computation.

The convergence analysis showed that the hybrid neural–numerical method achieved better convergence than the pure PINN approach. The mean squared error value above could be explained in the sense that the hybrid approach does not begin at a completely unconstrained neural approximation. It also takes the classical numerical solution as a basis for calculation and lets the neural network learn the missing correction term. Mathematically, this means that the hybrid solution:

$$\hat{u}_H = u_{\text{num}} + N_\theta$$

reduces the learning burden on the neural network because the network does not approximate the entire solution from zero, but only learns the residual correction:

$$N_\theta \approx u - u_{\text{num}}$$

This interpretation is coherent with the notion of hybrid multiscale PINN research, which is based on constraining (or guiding) the neural learning process from numerical data [19]. It is also in line with the recent hybrid FEM-PINN methods which have introduced finite element stencils into the PINN framework to enhance the efficiency of the methods in solving time-dependent PDEs [20].

The stability results indicated that the hybrid method had shown the lowest stability index below perturbation. This indicated that the pure PINN (small changes in initial or boundary conditions) led to larger deviation in the hybrid solution. This is because of the classical numerical methods offer a stable computational structure, while

the neural component enhances the approximation accuracy. Hence, the hybrid method has the following two complementary advantages—stability of numerical discretization and flexibility of neural correction.

This finding is very similar to the one proposed by Moseley et al. [15] recently, who proposed finite basis physics-informed neural networks as a scalable domain-decomposition approach. Through their study, they found that decomposing the computational domain by making smaller subdomains, allows to enhance the scalability and efficiency of PINNs. The present study does not implement domain decomposition in a purely mathematical or numerical sense, but there is yet a certain spirit of it in the whole concept, namely that neural networks can be exploited to the best, when they are not simply pushed, as the only, black-box approximators.

It was seen from the computational efficiency study that classical numerical methods have the least CPU time required, whereas the pure PINN method has the highest one required for training. This is what you'd expect neural networks to do: they require a lot of epochs of optimization. The hybrid approach on the other hand lowered training load as compared to all neural approach. That means, a hybrid neural-numerical framework could be a practical middle ground as these are more accurate than classical methods in more complex scenarios and less compute-intensive than training a full PINN model from scratch.

Neural operators have also been studied recently and the need for a beyond-single-solution approximation is highlighted. To learn mappings between functions instead of a single instance of the differential equations, Cao et al. [18] proposed the Laplace neural operator (LNOP) for solving differential equations. The present research differs from such learning of solution operators by an entire family of operators by being able to solve only selected test problems. Both strategies have the same high-level objective, however – circumvent the shortcomings of conventional solvers in our more complicated problems with differential equations by using a neural approach.

Simultaneously, the current results reinforce a notion that neural networks are not expected to completely substitute for classical numerical techniques. Recent review studies within the realm of large-data models for PDEs highlight that while neural methods show great promise, they currently struggle with issues in relation to accuracy, generalization, computational efficiency, and interpretability [17]. Likewise, some recent papers on PINN challenges have shown that the issues of training instability and weak convergence guarantee, as well as issues in multi-scale approximation are relevant [21]. This is corroborated by the results obtained for this study, which didn't demonstrate the superiority of the pure PINN method in all indicators in comparison with the hybrid method.

The overall result indicates that the most satisfactory result was achieved by utilizing the neural network as an auxiliary method instead of a substitute for the classical numerical methods. With the advantages of numerical structure of traditional methods, the approximation ability of neural networks, the hybrid neural-numerical method gave low errors, good convergence and high stability.

CONCLUSION

This study examined how to apply the numerical method with neural networks in solving the difficult differential equations. The research was carried out with selected test problems across ordinary differential equations, boundary value problems, and partial differential equations across classical numerical methods, physics-informed neural networks and hybrid neural–numerical methods.

Testing revealed that classical numerical methods were still effective and computation were reasonable when dealing with problems which were regular, not too high-dimensional and well-defined. Yet with the addition of nonlinearity, complex domains, high dimensionality, and/or stringent computational demands, their disadvantages are more apparent.

The physics-informed neural network method gave a flexible framework for approximating solutions to differential equations through embedding the differential equation in the loss function. However, the training conditions, the network model, the collocation points, and the stability of optimization influenced the

performance of the PINN method. Thus, sometimes the accuracy or stability of neural network approach was not the best.

The hybrid neural–numerical method turned out to be the most overall efficient method. It led to a smaller maximum error, better convergence behavior and to a better stability under perturbation. That's to say, it could be better to use classical numerical methods and correct them with the neural network than use either method alone to deal with complicated differential equations.

In this regard, the study finally concludes that neural networks should be viewed as complementary computing devices to work in conjunction with numerical methods. This integration opens up an interesting path for future computational mathematics, particularly if problems involve overcoming complexities in traditional approaches that are expensive, or approximation is non-flexible.

REFERENCES

1. R. L. Burden, J. D. Faires, and A. M. Burden, Numerical Analysis, 10th ed. Boston, MA, USA: Cengage Learning, 2016.
2. R. J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems. Philadelphia, PA, USA: SIAM, 2007.
3. C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, Spectral Methods: Fundamentals in Single Domains. Berlin, Germany: Springer, 2006.
4. G. Cybenko, "Approximation by superpositions of a sigmoidal function," Mathematics of Control, Signals and Systems, vol. 2, pp. 303–314, 1989.
5. K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Networks, vol. 2, no. 5, pp. 359–366, 1989.
6. I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," IEEE Transactions on Neural Networks, vol. 9, no. 5, pp. 987–1000, 1998.
7. M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," Journal of Computational Physics, vol. 378, pp. 686–707, 2019.
8. J. Sirignano and K. Spiliopoulos, "DGM: A deep learning algorithm for solving partial differential equations," Journal of Computational Physics, vol. 375, pp. 1339–1364, 2018.
9. M. W. M. G. Dissanayake and N. Phan-Thien, "Neural-network-based approximations for solving partial differential equations," Communications in Numerical Methods in Engineering, vol. 10, no. 3, pp. 195–201, 1994.
10. W. E and B. Yu, "The Deep Ritz Method: A deep learning-based numerical algorithm for solving variational problems," Communications in Mathematics and Statistics, vol. 6, no. 1, pp. 1–12, 2018.
11. J. Han, A. Jentzen, and W. E, "Solving high-dimensional partial differential equations using deep learning," Proceedings of the National Academy of Sciences, vol. 115, no. 34, pp. 8505–8510, 2018.
12. L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis, "DeepXDE: A deep learning library for solving differential equations," SIAM Review, vol. 63, no. 1, pp. 208–228, 2021.
13. Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier Neural Operator for parametric partial differential equations," in Proc. International Conference on Learning Representations (ICLR), 2021.
14. Z. Uddin, S. Ganga, R. Asthana, and W. Ibrahim, "Wavelets based physics informed neural networks to solve non-linear differential equations," Scientific Reports, vol. 13, Art. no. 2882, 2023.
15. B. Moseley, A. Markham, and T. Nissen-Meyer, "Finite basis physics-informed neural networks (FBPINNs): A scalable domain decomposition approach for solving differential equations," Advances in Computational Mathematics, vol. 49, Art. no. 62, 2023.
16. Q. Fang, X. Mou, and S. Li, "A physics-informed neural network based on mixed data sampling for solving modified diffusion equations," Scientific Reports, vol. 13, Art. no. 2491, 2023.
17. A. M. Hafiz, I. Faiq, and M. Hassaballah, "Solving partial differential equations using large-data models: A literature review," Artificial Intelligence Review, vol. 57, Art. no. 152, 2024.



18. Q. Cao, S. Goswami, and G. E. Karniadakis, “Laplace neural operator for solving differential equations,” *Nature Machine Intelligence*, vol. 6, pp. 631–640, 2024.
19. M. Hintermüller and D. Korolev, “A hybrid physics-informed neural network based multiscale solver as a partial differential equation constrained optimization problem,” *arXiv preprint arXiv:2309.04439*, 2023.
20. X. Feng, H. Shangguan, T. Tang, X. Wan, and T. Zhou, “A hybrid FEM-PINN method for time-dependent partial differential equations,” *arXiv preprint arXiv:2409.02810*, 2024.
21. W. Zhang, W. Suo, J. Song, and W. Cao, “Physics informed neural networks as intelligent computing technique for solving partial differential equations: Limitation and future prospects,” *arXiv preprint arXiv:2411.18240*, 2024.