iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** VI **Month of publication:** June 2025

DOI: https://doi.org/10.22214/ijraset.2025.71980

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue VI June 2025- Available at www.ijraset.com

Deep Learning Approaches for Solving Differential Equations in Scientific Computing

Dr. A. Prashanthi[1], Dr. D. Saraswathi[2], Jagdip Singh[3], Bangari Manasa[4], Kuchimanchi Jayasri[5], Mr. A. Durai Ganesh[6]
[1]*Associate Professor, Department of CSE, Nalla Narasimha Reddy Education Society's Group of Institutions, Hyderabad, Telangana, India*
[2]*Assistant Professor, Mathematics, M. I. E. T Engineering College,Trichy-620007,TAMILNADU, INDIA*
[3]*Assistant Professor, PG dept of C.S and I.T, B.U.C College, Batala, Punjab*
[4]*Assistant Professor, CSE, Nalla Narasimha Reddy Education Society Group of Institutions*
[5]*Assistant Professor, Computer science and engineering, Aurora Deemed to be University*
[6]*Assistant Professor, Department of Mathematics, PET Engineering College, Vallioor, Tirunelveli, Tamil Nadu*

Abstract: ***Differential equations are fundamental in modeling dynamic systems across physics, engineering, and finance. Traditional numerical methods, while robust, often struggle with high-dimensional problems and computational complexity. Recent advances in deep learning have introduced novel frameworks such as Physics-Informed Neural Networks (PINNs), Deep Operator Networks (DeepONets), and neural ordinary differential equations (Neural ODEs), offering efficient and scalable alternatives. This paper explores the integration of deep learning with differential equation solvers, comparing their accuracy, computational efficiency, and application scope. We provide theoretical insights, mathematical formulations, and implementation details to demonstrate how these models outperform traditional solvers in various scientific computing scenarios.***

Keywords: ***Differential equations, deep learning, PINNs, Neural ODEs, scientific computing, numerical methods, machine learning, DeepONets, high-dimensional PDEs.***

I. INTRODUCTION

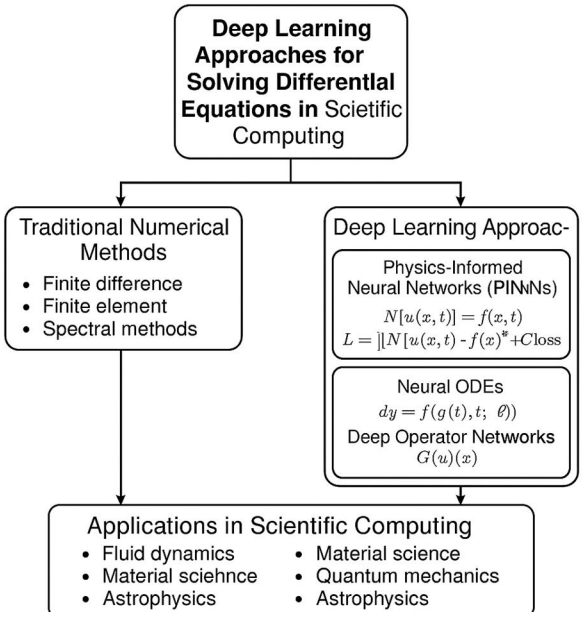
Differential equations are foundational tools used to model dynamic systems in diverse fields such as physics, biology, engineering, finance, and environmental science. They provide mathematical formulations that describe how a particular quantity evolves over time or space. Solving these equations accurately and efficiently is essential for scientific simulations, predictions, and real-world applications. Traditional techniques for solving ordinary and partial differential equations (ODEs and PDEs), including the finite difference method (FDM), finite element method (FEM), and spectral methods, rely heavily on discretization of the domain. Although these methods have been extensively studied and are well-established, they exhibit limitations when applied to high-dimensional systems, complex geometries, or problems requiring real-time computations.

In recent years, the intersection of deep learning and differential equation solving has opened a promising new direction. Deep learning, a subfield of machine learning focused on neural networks with multiple layers, has demonstrated the capacity to approximate highly nonlinear functions and capture complex relationships within data. This ability has led to the development of models like Physics-Informed Neural Networks (PINNs), Neural Ordinary Differential Equations (Neural ODEs), and Deep Operator Networks (DeepONets), which embed the structure and constraints of differential equations into the training process of neural networks.

Unlike traditional methods that require dense meshes and fine-tuned solvers, these deep learning models are mesh-free and data-driven, which significantly enhances their flexibility and generalization. Moreover, they can be trained directly on sparse data and still infer the underlying physical dynamics, offering advantages in scenarios where data is limited or noisy. These models not only learn the solutions but can also generalize across different input conditions, making them highly suitable for parametric studies and inverse problems.

This paper explores the theoretical foundations and practical applications of deep learning-based approaches for solving differential equations. It aims to compare these modern methods with classical numerical techniques, highlight their advantages and limitations, and demonstrate their role in advancing scientific computing. As computational resources continue to grow and algorithms become more refined, deep learning is poised to become a central tool in solving complex differential equations across disciplines.

©IJRASET: All Rights are Reserved



II. TRADITIONAL NUMERICAL METHODS: A BRIEF OVERVIEW

Traditional numerical methods have long been the cornerstone for solving differential equations in scientific computing. These include techniques like Euler's method, Runge-Kutta methods, finite difference methods (FDM), finite element methods (FEM), and spectral methods. Each of these methods discretizes the continuous domain—either in time, space, or both—to transform differential equations into systems of algebraic equations that can be solved numerically.

The Euler method, one of the simplest techniques for ordinary differential equations (ODEs), approximates solutions through linear extrapolation. Though computationally inexpensive, it is limited by low accuracy and stability issues, particularly for stiff equations. To improve upon this, Runge-Kutta methods (especially the fourth-order RK4) provide better accuracy without significantly increasing computational complexity. These are widely used in simulations of physical systems, such as motion dynamics and control systems.

For partial differential equations (PDEs), finite difference methods approximate derivatives by finite differences using grid points. This technique is intuitive and effective for regular geometries but suffers when applied to complex or irregular domains. Finite element methods, on the other hand, offer greater flexibility in handling irregular geometries by dividing the domain into elements and using local polynomial basis functions. FEM is particularly powerful in structural mechanics, heat transfer, and electromagnetics.

Spectral methods utilize global basis functions like trigonometric or orthogonal polynomials to achieve high accuracy for smooth problems. However, they may not perform well with discontinuities or sharp gradients, commonly leading to issues like Gibbs phenomena. Despite their effectiveness, these traditional methods face major challenges. Chief among them is the Curse of Dimensionality, which causes an exponential increase in computational resources as the number of dimensions grows. This becomes especially problematic in high-dimensional PDEs and stochastic differential equations (SDEs), where mesh-based discretization becomes impractical. Additionally, these methods are inherently sequential in many cases, limiting their scalability on parallel hardware like GPUs.

As scientific computing problems grow in complexity and dimensionality, the limitations of traditional numerical methods have spurred the exploration of deep learning-based alternatives, which aim to address issues of scalability, mesh-dependence, and high-dimensional approximation with more adaptive and data-driven techniques.

III. DEEP LEARNING FOUNDATIONS RELEVANT TO PDE SOLVING

Deep learning has emerged as a transformative tool in computational science due to its remarkable ability to approximate complex nonlinear functions. At the heart of deep learning models are artificial neural networks (ANNs), which are composed of interconnected layers of nodes (neurons). These networks can learn mappings between inputs and outputs by adjusting their internal parameters (weights and biases) through training on data.

The Universal Approximation Theorem states that a feedforward neural network with a sufficient number of neurons in a hidden layer can approximate any continuous function on a compact domain to any desired accuracy. This theoretical foundation makes neural networks suitable for learning the solutions of differential equations, which often involve complex and high-dimensional mappings.

In the context of solving PDEs and ODEs, several components of deep learning become particularly important:

1) Loss Functions: These quantify how closely the neural network's output satisfies the differential equation and boundary conditions. For example, in Physics-Informed Neural Networks (PINNs), the loss function typically includes the residual of the differential equation and terms for enforcing initial/boundary conditions.

2) Backpropagation and Optimization: The training process involves computing gradients of the loss with respect to the network parameters using backpropagation. Optimization algorithms such as Stochastic Gradient Descent (SGD), Adam, or L-BFGS are then used to minimize the loss and update the network weights.

3) Regularization and Constraints: To avoid overfitting and ensure physical plausibility, regularization terms or physics-based constraints can be added to the loss. These help the model converge to physically meaningful solutions rather than just interpolating the data.

4) Architecture Design: Depending on the problem, different architectures such as Convolutional Neural Networks (CNNs) or Recurrent Neural Networks (RNNs) may be used. For spatial-temporal PDEs, CNNs are advantageous for capturing local features, while RNNs can be effective for time-dependent dynamics.

By leveraging these foundational concepts, deep learning models are being tailored to not only fit data but also adhere to the governing laws of physics and dynamics, thus enabling them to serve as powerful solvers for differential equations in scientific computing.

IV. PHYSICS-INFORMED NEURAL NETWORKS (PINNS)

Physics-Informed Neural Networks (PINNs) represent a class of deep learning models that integrate physical laws directly into the training process. Unlike traditional neural networks that learn solely from data, PINNs utilize the governing differential equations to guide learning. This makes them particularly suitable for scientific computing applications where data may be scarce or expensive to obtain.

The core idea behind PINNs is to construct a neural network that approximates the solution of a given partial differential equation (PDE) such as:

$$\mathcal{N}[u(x,t)] = f(x,t)$$

$$\mathcal{L}(\theta) = \frac{1}{N_r}\sum_{i=1}^{N_r} |\mathcal{N}[\hat{u}(x_i,t_i)] - f(x_i,t_i)|^2 + \frac{1}{N_b}\sum_{j=1}^{N_b} |\hat{u}(x_j,t_j) - g(x_j,t_j)|^2$$

One of the strengths of PINNs is that they are mesh-free and rely solely on automatic differentiation to compute derivatives, which avoids discretization errors common in finite difference or finite element methods. Furthermore, PINNs are highly parallelizable and can generalize well to new inputs within the same domain, provided the network has been properly trained.

However, training PINNs can be computationally intensive and sensitive to the choice of collocation points, network architecture, and optimizer settings. Researchers often employ techniques such as adaptive sampling, loss weighting, or transfer learning to improve convergence and performance.

PINNs have been successfully applied to various scientific problems including fluid flow, heat transfer, and electromagnetics, showing promising results in terms of accuracy, interpretability, and generalization compared to traditional numerical solvers.

V. NEURAL ORDINARY DIFFERENTIAL EQUATIONS (NEURAL ODES)

Neural Ordinary Differential Equations (Neural ODEs) introduce a continuous-depth paradigm for modeling time-dependent phenomena using deep learning. Proposed by Chen et al. (2018), Neural ODEs replace discrete hidden layers in neural networks with a continuous transformation governed by a learned differential equation. Instead of mapping layer-by-layer transformations as in residual networks, the hidden state evolves over time as:

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

To obtain the solution $y(t_1)$ given an initial value $y(t_0)$, one integrates the neural ODE using a numerical solver:

$$y(t_1) = y(t_0) + \int_{t_0}^{t_1} f(y(t), t; \theta)\, dt$$

\

Neural ODEs have multiple advantages in scientific computing. They provide adaptive time stepping, ensuring precision in stiff or multi-scale systems. Unlike fixed-layer networks, Neural ODEs adjust the number of evaluations based on the complexity of the dynamics. This continuous modeling framework is particularly suitable for dynamical systems such as population dynamics, chemical reactions, or mechanical systems.

Moreover, Neural ODEs offer parameter efficiency, using fewer parameters than deep residual networks for comparable accuracy. They also allow continuous interpolation and extrapolation over time, making them ideal for irregularly sampled data and forecasting tasks.

However, Neural ODEs are not without limitations. Solving the integral often requires multiple evaluations of , leading to increased computational cost. Additionally, stiffness in the learned dynamics can degrade training stability.

Despite these challenges, Neural ODEs have opened new directions for integrating deep learning with classical differential equation theory. Their ability to blend data-driven modeling with continuous-time dynamics makes them a powerful tool in scientific computing and time-series analysis.

VI. DEEP OPERATOR NETWORKS (DEEPONETS)

Deep Operator Networks (DeepONets) are a groundbreaking class of neural network architectures specifically designed to learn operators—mappings between infinite-dimensional function spaces—rather than conventional finite-dimensional input-output relationships. This allows DeepONets to generalize across entire families of functions and solve parameterized differential equations with remarkable efficiency.

In scientific computing, many problems involve solving differential equations with varying inputs such as boundary conditions, material properties, or source terms. Traditional methods require solving the equation anew for each variation. DeepONets overcome this limitation by learning the solution operator such that:

$$\mathcal{G}(u)(x) \approx y(x)$$

The architecture of a DeepONet consists of two sub-networks:

- **Branch Network**: Takes samples of the input function at predefined sensor points and maps them into a latent feature vector.
- **Trunk Network**: Takes the evaluation point as input and also maps it to a latent space. The outputs of both networks are combined via an inner product:

$$\mathcal{G}(u)(x) \approx \sum_{i-1}^{p} b_i(u) \cdot t_i$$

This separation of input function and evaluation point enables the model to generalize to unseen inputs and predict solution functions at any query location. Unlike traditional solvers, once trained, a DeepONet can predict solutions instantly for a wide range of inputs without re-solving the differential equation.

DeepONets are particularly effective in high-dimensional PDE problems, uncertainty quantification, and real-time simulation. They have been applied successfully in fluid mechanics, elasticity, and biological systems where rapid inference is critical.

Training DeepONets typically involves generating datasets of input-output function pairs using high-fidelity solvers. The model minimizes the discrepancy between predicted and true outputs across multiple function realizations.

However, challenges include the need for large and diverse training datasets, computational cost during training, and ensuring physical consistency in the learned operator. Recent extensions like Physics-Informed DeepONets aim to mitigate these issues by embedding governing equations into the learning process.

DeepONets represent a significant leap in modeling general-purpose solution operators, enabling scalable, mesh-free, and versatile solvers in scientific computing.

VII. TRAINING TECHNIQUES AND OPTIMIZATION STRATEGIES

Training deep learning models to solve differential equations involves more complexity than standard machine learning tasks. Unlike traditional models that minimize a loss function based solely on data, scientific computing models must also satisfy physical laws, boundary conditions, and initial values. To achieve this, specialized training techniques and optimization strategies are employed, ensuring convergence, stability, and physical accuracy.

For models like Physics-Informed Neural Networks (PINNs), Deep Operator Networks (DeepONets), and Neural Ordinary Differential Equations (Neural ODEs), the training process often involves defining a composite loss function. This includes terms that measure how well the neural network satisfies the governing equations, the boundary or initial conditions, and any available data points. The loss function must be carefully weighted to balance the influence of each component during training.

Optimizers play a crucial role in minimizing this loss. Popular choices include gradient-based algorithms like Adam, which adaptively adjusts learning rates for each parameter, and L-BFGS, a quasi-Newton method that is particularly useful for physics-informed models. These optimizers help the network converge to a solution that honors both data fidelity and physical consistency.

Gradient computation is typically done using automatic differentiation, which enables the precise evaluation of derivatives required to formulate the physics-based loss. For stiff or sensitive systems, training can be unstable. To address this, researchers employ techniques such as adaptive learning rate schedules, where the learning rate changes based on performance, and gradient clipping, which prevents exploding gradients during training.

Regularization methods are also employed to avoid overfitting, especially when training data is scarce. Techniques like dropout, early stopping, and weight decay help the model generalize better. Another important strategy is curriculum learning, where the model is first trained on simplified versions of the problem and gradually exposed to more complex scenarios. This progressive approach improves learning efficiency and stability.

Normalization of input features and output targets can also speed up convergence and make the training more robust. Moreover, pretraining models on synthetic data generated from known solvers can give them a strong initialization, reducing training time on real-world problems.

Overall, the training of deep learning models for differential equations is a multifaceted process that requires a blend of modern optimization methods, domain knowledge, and tailored strategies to ensure accuracy, efficiency, and physical validity.

VIII. APPLICATIONS IN SCIENTIFIC COMPUTING

Deep learning methods for solving differential equations are increasingly being adopted across various domains in scientific computing. These models are particularly attractive due to their ability to approximate complex physical systems without requiring mesh generation or domain discretization, which are common in traditional numerical solvers. Their flexibility, generalization capability, and scalability make them suitable for a wide range of scientific and engineering applications.

In fluid dynamics, Physics-Informed Neural Networks (PINNs) have been used to solve the Navier-Stokes equations for incompressible flow. These models can capture complex flow patterns around objects and in turbulent regimes, offering fast and accurate predictions. Researchers have employed PINNs to analyze airflow over aircraft wings and blood flow through arteries, demonstrating efficiency and reduced computational costs compared to classical solvers.

In material science, deep learning models have helped characterize the mechanical behavior of materials by learning stress-strain relationships governed by partial differential equations. Deep Operator Networks (DeepONets) have been used to model heterogeneous materials and predict how materials respond under varying loads, helping in the design of new composites and smart materials.

In quantum mechanics, neural networks have been trained to solve the Schrödinger equation, enabling the modeling of atomic and subatomic systems. These approaches are especially valuable when dealing with high-dimensional quantum systems where traditional solvers struggle with scalability.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

In astrophysics, deep learning has facilitated simulations of stellar dynamics and galaxy formation. By solving differential equations that describe gravitational interactions and energy transfer, these models help understand the evolution of celestial systems over time.

Other areas include climate modeling, where neural differential solvers are applied to predict atmospheric dynamics, and biomechanics, where neural models simulate the behavior of biological tissues. In all these applications, the major benefits include faster inference, the ability to learn from sparse or noisy data, and adaptability to parameter changes.

By embedding physical laws directly into their training processes, deep learning approaches maintain consistency with established scientific principles while offering modern computational advantages, transforming the landscape of scientific computing across disciplines.

IX. CHALLENGES AND LIMITATIONS

Despite the promising advancements in using deep learning to solve differential equations, several challenges and limitations remain that hinder their widespread adoption in scientific computing.

One major challenge is training stability and convergence, especially for stiff systems or equations with rapid changes. Neural networks may struggle to converge to accurate solutions when the underlying physical system involves sharp gradients or discontinuities. This can cause optimization algorithms to get stuck in poor local minima or exhibit oscillatory behavior during training.

Handling discontinuities and sharp gradients is particularly difficult because neural networks are inherently smooth function approximators. Capturing sudden changes, such as shock waves in fluid dynamics or phase transitions in materials, often requires specialized architectures or training schemes, which are still under active research.

Another limitation is the lack of theoretical error bounds and guarantees on the accuracy of the learned solutions. Unlike classical numerical methods, which come with well-established convergence theories and error estimates, deep learning approaches rely heavily on empirical validation. This uncertainty can limit trust and acceptance, especially in critical applications like aerospace or medicine.

The high computational cost associated with training large neural networks is also a significant bottleneck. Training can require extensive computational resources and time, especially for high-dimensional problems or when a large dataset of solved equations is needed. This contrasts with traditional solvers that, although sometimes slower for complex problems, do not require upfront training.

Moreover, the quality and quantity of training data pose challenges. Many deep learning methods require large datasets of input-output pairs generated from high-fidelity solvers. Generating such datasets can be costly or infeasible for complex systems.

Lastly, integrating deep learning models with existing numerical solvers and ensuring robustness under varying conditions remains an ongoing challenge. Hybrid methods that combine traditional and data-driven approaches are promising but require careful design. while deep learning methods bring flexibility and scalability, overcoming these challenges is essential for reliable, accurate, and practical deployment in scientific computing. Continued research in model architectures, training techniques, and theoretical analysis is crucial to address these limitations.

X. FUTURE DIRECTIONS

The integration of deep learning with differential equation solvers has opened exciting avenues for scientific computing. However, to fully realize the potential of these approaches, several future research directions and innovations are necessary.

One promising area is the integration of symbolic artificial intelligence with neural solvers. By combining symbolic reasoning and neural networks, models could achieve better interpretability and robustness. This fusion would enable not only the approximation of solutions but also the discovery of underlying physical laws and analytical expressions, improving trust and insight into the modeled phenomena.

Another key direction is the development of self-supervised and unsupervised learning models that require minimal labeled data or precomputed solutions. Such models would be particularly valuable in real-time applications, such as weather forecasting or control systems, where obtaining large amounts of training data is impractical. Self-supervised models can learn directly from physical laws and sparse observations, enhancing adaptability and generalization.

Extending deep learning architectures to solve stochastic differential equations is another critical challenge. Many real-world systems involve uncertainty and randomness, and capturing these stochastic dynamics is essential for accurate modeling. Developing neural models that incorporate probabilistic frameworks and uncertainty quantification will significantly expand the applicability of deep learning in scientific domains.

The application of federated learning and distributed training methods to physics-informed models offers another exciting direction. Scientific simulations often involve large datasets distributed across multiple locations or institutions. Federated learning would allow models to be trained collaboratively without sharing sensitive data, preserving privacy and security while benefiting from diverse datasets.

Additionally, hybrid approaches that combine classical numerical methods with neural solvers are expected to become more prominent. These approaches leverage the strengths of both worlds — the reliability and theoretical guarantees of numerical solvers and the flexibility and scalability of neural networks.

Finally, advances in hardware, such as specialized AI accelerators, will further accelerate training and inference, making deep learning-based differential equation solvers more accessible for complex, large-scale scientific problems.

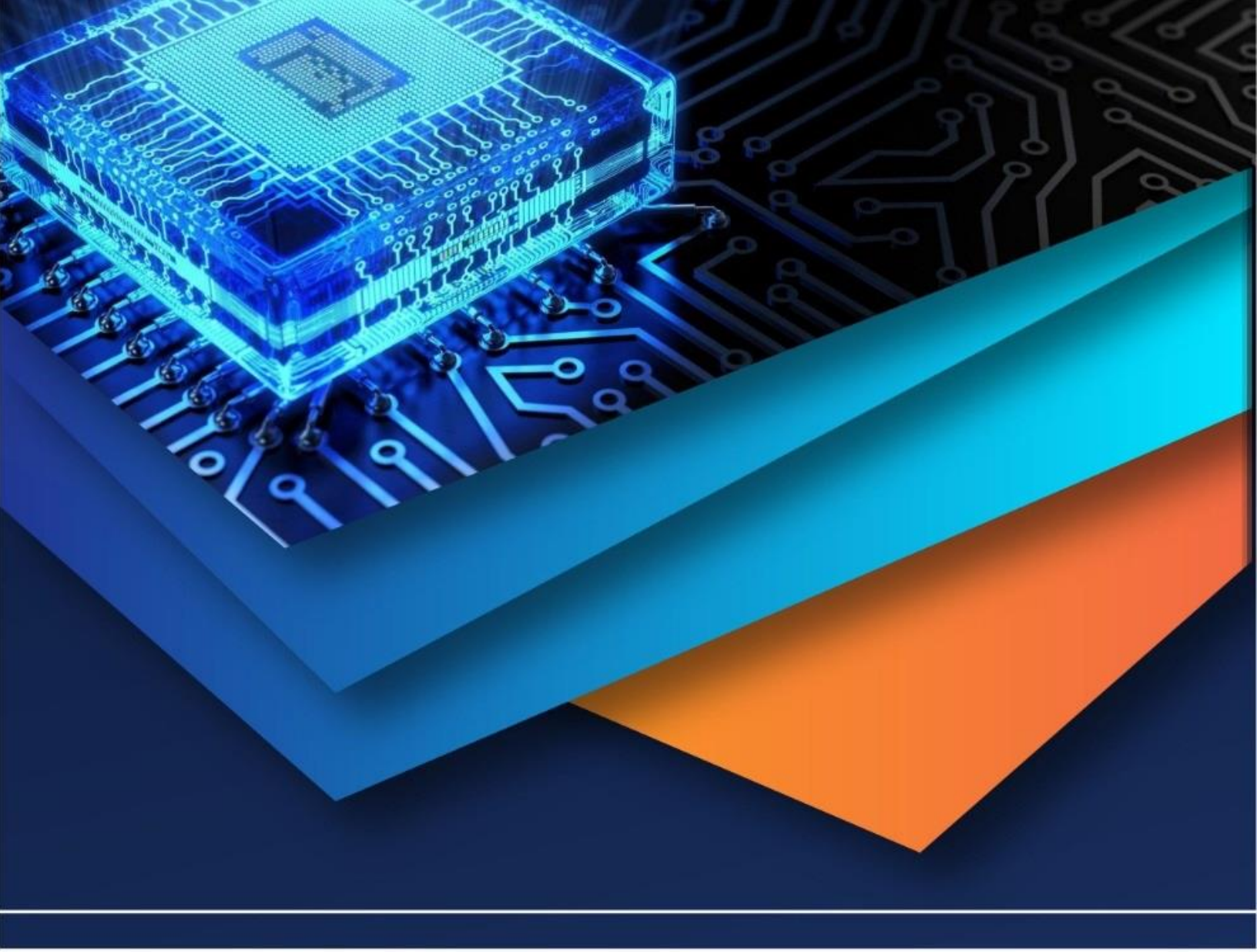
XI. CONCLUSION

Deep learning has emerged as a transformative approach for solving differential equations in scientific computing. Techniques like Physics-Informed Neural Networks, Neural Ordinary Differential Equations, and Deep Operator Networks have demonstrated remarkable capability in handling complex, high-dimensional problems where traditional methods face limitations. These approaches combine data-driven learning with physical laws to produce accurate, scalable, and mesh-free solutions. While challenges such as training stability, computational cost, and theoretical guarantees remain, ongoing research is addressing these issues through innovative architectures, optimization strategies, and hybrid models. The future holds promising developments in integrating symbolic reasoning, self-supervised learning, stochastic modeling, and distributed training, all of which will expand the reach and effectiveness of neural solvers. Ultimately, the synergy between deep learning and numerical analysis is poised to redefine computational methods across science and engineering, enabling more efficient, interpretable, and adaptable solutions to complex differential equations.

REFERENCES

[1] Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686–707.
[2] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., & Duvenaud, D. (2018). Neural Ordinary Differential Equations. Advances in Neural Information Processing Systems, 31.
[3] Lu, L., Jin, P., & Karniadakis, G. E. (2021). Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nature Machine Intelligence, 3(3), 218–229.
[4] Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. Nature Reviews Physics, 3, 422–440.
[5] Raissi, M., & Karniadakis, G. E. (2018). Hidden physics models: Machine learning of nonlinear partial differential equations. Journal of Computational Physics, 357, 125–141.
[6] Sirignano, J., & Spiliopoulos, K. (2018). DGM: A deep learning algorithm for solving partial differential equations. Journal of Computational Physics, 375, 1339–1364.
[7] Zang, Y., Zhang, Y., & Karniadakis, G. E. (2020). Weak adversarial networks for high-dimensional partial differential equations. Journal of Computational Physics, 429, 109949.
[8] Han, J., Jentzen, A., & E, W. (2018). Solving high-dimensional partial differential equations using deep learning. Proceedings of the National Academy of Sciences, 115(34), 8505–8510.
[9] Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021). DeepXDE: A deep learning library for solving differential equations. SIAM Review, 63(1), 208–228.
[10] Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed generative adversarial networks for stochastic differential equations. Journal of Computational Physics, 397, 108050.
[11] Rackauckas, C., Ma, Y., Martensen, J., Warner, C., Zubov, K., Supekar, R., ... & Ramadhan, A. (2020). Universal Differential Equations for Scientific Machine Learning. arXiv preprint arXiv:2001.04385.
[12] Kovachki, N. B., Azizzadenesheli, K., Bauer, S., et al. (2021). Neural Operator: Learning Maps Between Function Spaces. arXiv preprint arXiv:2108.08481.
[13] Tartakovsky, A. M., Marrero, C. O., Perdikaris, P., et al. (2020). Physics-informed deep learning for nonlinear multiphysics problems. Computers & Chemical Engineering, 133, 106675.
[14] Meng, X., Li, Z., Zhang, D., & Karniadakis, G. E. (2020). PPINNs: Parallel Physics-Informed Neural Networks based on domain decomposition. Journal of Computational Physics, 429, 109927.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)