Physics-Informed Neural Networks for Solving Partial Differential Equations: A Comparative Study with the Burgers' Equation

Hassan Al-Mahdawi[1], Ghassan Khazal Ali[2], Anna Ivanovna Sidikova[3] and Hassan Hadi Saleh[4]
[1] *Electronic and Computer Centre, University of Diyala, 32001 Baqubah, Iraq*
[2]*College of Engineering, University of Diyala, 32001 Baqubah, Iraq*
[3]*Department of System Programming, South Ural State University, 454080 Chelyabinsk, Russia*
[4]*Department of Computer Science, College of Science, University of Diyala, 32001 Baqubah, Iraq*
hssnkd@gmail.com

Keywords: Physics-Informed Neural Networks (PINNs), Burgers' Equation, Partial Differential Equations (PDEs), Finite Difference Method (FDM), Finite Element Method (FEM), Spectral Methods.

Abstract: Partial Differential Equations (PDE) are central to modelling phenomena of physical in engineering and science however up to now their mathematical solution has been elusive especially in high-dimensional domains with complex boundary conditions or nonlinear dynamics. Robust discretization techniques have been developed through the use of Finite Difference Method (FDM), Finite Element Method (FEM), and Spectral Methods but they face challenges of scalability, computational expense, and even problems associated with mesh-generation. Recently, Physics-Informed Neural Networks (PINNs) were proposed as a mesh-free method by which the governing physical law constraints can be imposed while training deep neural networks to approximate solutions of PDE imposing their initial and boundary conditions. This study will use the one-dimensional PDE for the viscous Burgers' equation as the baseline problem. First, a review of traditional approaches is presented with an explicit enumeration of their strengths and weaknesses. A PINN architecture is then designed that leverages automatic differentiation for the residuals and composite loss functions to enforce physics constraints. Mathematical analyses benchmark PINNs against FDM in terms of accuracy and demonstrate comparable levels of accuracy with less dependency on discretization as well as delivering a much smoother solution. The study recapitulates that though training effort is substantial for PINNs, it has an enticing possibility to crack nonlinear-PDE in higher dimensions and imbalanced geometries, therefore making an up-to-date leap forward from standard numerical analysis.

1 INTRODUCTION

Partial Differential Equations (PDEs) are essential in explaining various physical phenomena in engineering and science, as well as heat conduction, fluid dynamics, elasticity, and quantum mechanics. Solutions of PDE are critical for behavior of predicting system, designing processes, and models of validating theoretical. Traditionally, PDEs solved by using traditional numerical methods like the finite difference method (FDM), finite volume method (FVM), and finite element method (FEM). These methods formed as backbone of mathematics of computational solutions for decades [1]-[3].

However, traditional methods have several problems even structured areas and low-dimensional. Because of the curse of dimensionality, irregular geometries, nonlinearities, and stiff boundary conditions [4], [5], it can be exceedingly expensive to execute discretization-based algorithms. FEM is good for shapes that are hard to work with, but it takes a lot of processing power and mesh production [1]. FDM, on the other hand, is still straightforward and works well, but it doesn't work well with edges that aren't straight [2], [3]. Researchers have been looking for innovative ways to make systems more stable and able to handle more users because of these kinds of challenges.

To deal with ill-posedness in inverse problems, classical researchers came up with regularization methods that keep solutions stable when input data is noisy or missing. The Tikhonov regularization method is one of the most common. It adds a penalty term to balance fidelity and smoothness. Recent developments integrate Tikhonov regularization with particle swarm optimization (PSO) to autonomously

determine the optimal regularization parameter, thereby improving stability and precision for inverse PDE problems [6]. The Lavrentiev regularization approach is other way, which often used for inverse boundary value problems in PDE heat conduction equations because it is resistant to noise measurement data [7]. In engineering, including the inverse boundary problem for PDE of heat exchange in hollow cylinders, customized regularization techniques have been utilized to reconstruct unknown thermal fluxes from given data [8]. These studies underscore the significance of classical regularization in addressing inverse PDE problems, although they remain susceptible to parameter adjustment and computational expense in elevated dimensions. Aside from regularization, the main focus of progress in FDM, FEM, and spectral methods has been to improve accuracy, adaptability, and scalability. FDM is still useful for structured grids, but it has to follow strict rules about stability [9]. Weak formulations and adaptive mesh refinement are some of the evolutions of FEM which allow getting exact solutions on irregular geometries [10]. At the same time, hybrid FEM–neural frameworks combine mesh-born accuracy with flexibility by learning [11]. Recently, deep learning has been used to enhance the accuracy of Finite Difference Methods (FDM), hence performing better in discontinuous or multiscale problems [12]. Physics-informed machine learning [13] is currently used for boundary layer and singular perturbation problems in fluid dynamics that were challenging for classical solutions.

The most radical change of thought was brought about by Physics-Informed Neural Networks. Raissi et al. [14] were the first to propose incorporating PDE residuals in the training of neural networks: a solution approximation could be found with no need for any mesh. Since then, PINNs have undergone many modifications, and in several works, their improvement has been documented both theoretically and practically [15], [16]. Some of the new architectures improving their usefulness are transformer-based PINNs [17], separable PINNs (SPINNs) for high-dimensional PDEs [18], and Kolmogorov–Arnold networks (KINNs) [19].

The Neural Network-augmented FEM and the finite basis PINNs (FB-PINNs) serve as early hybrid techniques sharing the best aspects of conventional solvers with new deep-learning approaches. For inverse problems, PINNs have found application since they can infer any unknown coefficient or boundary condition from observational data and physical constraints—no regularization needs to be added explicitly. Some of the most important problems in optimization were recently improved by noise-aware training and adaptive collocation sampling as well as by meta-heuristic optimizers. These improvements show a clear path: PDE solving has gone from solvers that only use discretization to regularization-enhanced inverse approaches and now to frameworks that use neural networks. PINNs and their variants provide a modern, flexible, and scalable class of solutions, especially for high-dimensional, nonlinear, and data-sparse PDE problems, despite the ongoing relevance of classical and regularization methods in various practical contexts.

2 PROBLEM STATEMENT

Partial Differential Equations (PDEs) are fundamental tools in the modeling of physical phenomena, including heat transfer, wave propagation, quantum mechanics, and fluid dynamics. You can write a general PDE like this:

$$N[u](x,t) = f(x,t),\ (x,t) \in \Omega \times [0,T], \qquad (1)$$

subject to initial conditions at t=0 and boundary conditions at the spatial boundary $\partial\Omega$. The objective is to gain a reliable estimate of the solution *u(x,t)* while guaranteeing its precision and stability.

This work emphases on the one-dimensional PDE of viscous Burgers' equation, serving as a nonlinear PDE problem:

$$u_t + uu_x = vu_{xx},\quad x \in [-1,1],\ t \in [0,1], \quad (2)$$

where *u(x,t)* define as a velocity field, *v>0* represent viscosity and distribution of the initial velocity is given as:

$$u(x,0) = -\sin(\pi x),\ x \in [-1,1], \qquad (3)$$

which generates smooth wave that changes into a shock-like structure as progresses of time.

We enforce Dirichlet boundary conditions at the boundaries of domain:

$$u(-1,t) = u(1,t) = 0 \quad t \in [0,1]. \qquad (4)$$

These requirements set hard (no-slip) edges at both ends of the spatial domain. The Burgers' equation shows how nonlinear convection (uu_x) and diffusion (vu_{xx}) work together. When the viscosity v is low, nonlinear processes take over. These processes create sudden gradients and shock waves that are well known to be hard to deal with using traditional discretization methods. To stop oscillations or fake dissipation, traditional finite difference or finite

element solutions need very small meshes. As a result, the Burgers' equation is often used as a standard problem to test new numerical methods and PDE solvers that use machine learning. This paper provides a practical framework for comparing conventional numerical methods with Physics-Informed Neural Networks (PINNs), illustrating their respective benefits in addressing nonlinear partial differential equations with pronounced features.

3 CLASSICAL METHODS

In order to solve PDEs using classical methods, we have to break up the domains of space and time into smaller parts. This makes continuous derivatives into algebraic approximations. Some of the greatest ways to do this are the finite difference method (FDM), the finite element method (FEM), and spectral methods.

3.1 The Finite Differences Method (FDM)

Instead of derivatives, the finite difference method uses difference quotients. We can get an idea of the first-order spatial derivative by doing the following:

$$u_x(x_i) \approx \frac{u(x_{i+1}) - u(x_i)}{\Delta x}, \tag{5}$$

and the second-order derivative as:

$$u_{xx}(x_i) \approx \frac{u(x_{i+1}) - u(x_i) - u(x_{i-1})}{(\Delta x)^2}. \tag{6}$$

Applied to the Burgers' equation, an explicit forward-time central-space (FTCS) scheme yields:

$$u_i^{n+1} = u_i^n - \Delta t u_i^n \frac{u_{i+1}^n - u_{i-1}^n}{\Delta x} + \nu \Delta t \frac{u_{i+1}^n - 2u_i^n - u_{i-1}^n}{(\Delta x)^2}. \tag{7}$$

Such explicit schemes are conditionally stable and require the Courant–Friedrichs–Lewy (CFL) condition:

$$\Delta t \le \min\left(\frac{\Delta x}{\max|u|}, \frac{(\Delta x)^2}{2\nu} \right), \tag{8}$$

to avoid numerical blow-up. For small viscosity ν, Burgers' equation develops sharp gradients, forcing Δx and Δt to be very small, which makes the method computationally expensive [20], [21]. Moreover, explicit schemes suffer from numerical dissipation, smearing steep gradients and underestimating shock strength.

3.2 Finite Element Method (FEM)

The finite element method reformulates the PDE in its weak form. Multiplying Burgers' equation by a test function $v(x)$ and integrating over the spatial domain $\Omega = [-1,1]$: yields:

$$\int_{-1}^{1} u_t v\,dx + \int_{-1}^{1} u u_x v\,dx + \nu \int_{-1}^{1} u_x v_x\,dx = 0. \tag{9}$$

The approximate solution is expressed as a finite expansion in basis functions:

$$u_h(x,t) = \sum_{i=1}^{N} U_i(t)\phi_i(x). \tag{10}$$

leading to the semi-discrete system:

$$M\frac{d\mathbf{U}}{dt} + \mathbf{C}(\mathbf{U}) + \nu K\mathbf{U} = 0, \tag{11}$$

where M is the mass matrix, K the stiffness matrix, and $C(U)$ the nonlinear convection term. At each time step, nonlinear solvers (e.g., Newton–Raphson or fixed-point iteration) are required, making FEM computationally heavy for Burgers' equation [22], [23].

3.3 Spectral Methods

In spectral methods, the solution is expanded in global basis functions (e.g., Fourier series):

$$u(x,t) \approx \sum_{k=-N/2}^{N/2} \hat{u}_k(t) e^{ik\pi x}. \tag{12}$$

Substituting into Burgers' equation gives the evolution equations for the Fourier coefficients:

$$\frac{d\hat{u}_k}{dt} + \frac{ik\pi}{2} \sum_{m=-N/2}^{N/2} \hat{u}_m \hat{u}_{k-m} + \nu(k\pi)^2 \hat{u}_k = 0. \tag{13}$$

In this case, the nonlinear term is turned to a convolution sum in Fourier domain, which can be easily compute by using FFTs. To make the solutions more smooth, spectral methods obtain exponential convergence. The Gibbs phenomenon, in other side, it also makes oscillations happen near discontinuities when Burgers' equation makes steep gradients or shock-like structures ($\nu << 1$) [24].

Even though these traditional methods are mathematically sound and widely accepted, there are a number of important problems that come up when trying to solve nonlinear PDEs like Burgers' equation:

- Stability Restrictions. The CFL condition makes obvious systems (as FTCS) need steps with small-time, which makes processing of calculations are slow [1].

- Mathematical Dissipation/Dispersion. FDM and FEM can lead to false oscillations when the viscosity is low [21], [23].
- High Computational Cost. FEM and spectral methods need high-order expansions, which get more expensive [22], [24].
- Difficulty in High Dimensions. The curse of dimensionality makes it harder to use higher-dimensional PDEs because the difficulty of calculating produces exponentially with the number of dimensions.

The Figure 1 viewing mathematical result by applying the classical methods on the PDE of Burgers' equation at t=1.0:

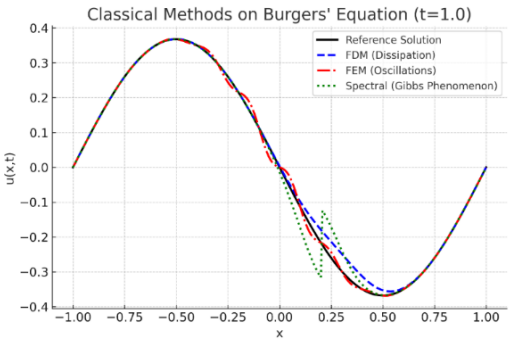

Figure 1: Classical methods on Burgers' equation.

These limitations motivate discovering alternative methods like the Physics-Informed Neural Networks (PINNs), which avoid meshing, can naturally incorporate boundary/initial conditions, and it is not limited by stability conditions

4 PHYSICS-INFORMED NEURAL NETWORKS (PINNS)

In term of problem embedding, we estimated the result by using a neural network $u_\theta(x,t)$ with trainable parameters θ as trainable. For the 1D viscous Burgers' equation see following:

$$u_t + u\,u_x = \nu\,u_{xx}, (x,t) \in [-1,1] \times [0,1], \quad (14)$$

the IC and BCs see the follwoing

$$u(x,0) = -\sin(\pi x), \quad u(-1,t) = 0, \quad u(1,t) = 0, \quad (15)$$

physics residual represented by:

$$r_\theta(x,t) = \partial_t u_\theta(x,t) + u_\theta(x,t)\partial_x u_\theta(x,t) - \nu\partial_{xx} u_\theta(x,t). \quad (16)$$

the derivatives terms $\partial_t u_\theta, \partial_x u_\theta, \partial_{xx} u_\theta$ are computed by automatic differentiation (AD) method through the network [14], [25].

For Loss construction we sample three sets of points:

- Interior (collocation) points: $\{(x_f^i, t_f^i)\}_{i=1}^{N_f} \subset (-1,1)\times(0,1]$.
- Initial points: $\{x_0^j\}_{j=1}^{N_0} \subset [-1,1], \quad t = 0$.
- Boundary points: $\{t_b^k\}_{k=1}^{N_b} \subset [0,1], \quad x = \pm 1$.

Defining the losses of mean–squared:

- PDE residual loss:

$$L_{PDE}(\theta) = \frac{1}{N_f}\sum_{i=1}^{N_f} \left| r_\theta(x_f^i, t_f^i) \right|^2. \quad (17)$$

- Initial condition loss:

$$L_{IC}(\theta) = \frac{1}{N_0}\sum_{j=1}^{N_0} \left| u_\theta(x_0^j, 0) + \sin(\pi x_0^j) \right|^2. \quad (18)$$

- Boundary condition (Dirichlet) loss:

$$L_{BC}(\theta) = \frac{1}{N_b}\sum_{k=1}^{N_b} \left(\left| u_\theta(-1, t_b^k) \right|^2 + \left| u_\theta(1, t_b^k) \right|^2 \right). \quad (19)$$

The loss of total PINN apply by (possibly adaptive) weights λ>0:

$$L(\theta) = \lambda_f L_{PDE} + \lambda_0 L_{IC} + \lambda_b L_{BC}. \quad (20)$$

We can set weights manually or learn them adaptively to balance gradients (like NTK-based or gradient-norm balancing) and avoid the "stiff loss" problem [26], [27].

For Sampling and normalization, we do the following steps:

- Space–time sampling. Use Latin Hypercube Sampling (LHS) for (x,t) in [−1,1] ×[0,1]. Densify near steep gradients to obtain shocks in case ν is small [28].
- Non-dimensionalization. Scale x, t, and u so that the sizes of the residual terms are similar. This makes training more stable and helps with conditioning.
- Curriculum in time. Train first on early times (e.g., t∈[0, τ]), then gradually extend τ—this helps for strongly nonlinear evolution.

In order to Network & optimization we will do the following steps

- Architecture. A fully connected MLP has 4 to 8 hidden layers and 50 to 128 neurons, and the activations are smooth (tanh, sine). Fourier-feature embeddings or positional encodings can help add spectral bias to make features sharper [9], [25].

- Optimizers. A common recipe is to use Adam (e.g., 10−3→10−4) to explore and then L-BFGS (full-batch, strong Wolfe) to fine-tune to machine precision. [14], [25].
- Regularization. Early stopping, weight decay, or Jacobian norm penalties; and adaptive loss weights to fix gradient imbalance [26], [27].

In term of Error metrics and validation, the evaluate against a high-resolution finite-difference reference (or analytical solution if available):

- Relative L^2 error:

$$\varepsilon_{L^2} = \frac{\| u_\theta - u_{\text{ref}} \|_{L^2(\Omega\times[0,1])}}{\| u_{\text{ref}} \|_{L^2(\Omega\times[0,1])}}. \quad (21)$$

- Pointwise/temporal slices. Compare profiles $u(x,t^*)$ for $t^* \in \{0.12, 0.5, 1.0\}$.
- Physics violation. Report max $\| r_\theta \|_{L^2} = \left(\int_{\Omega\times[0,1]} |r_\theta(x,t)|^2 \, dx\,dt \right)^{1/2}$ to show PDE consistency.

For Complete training procedure (Burgers' equation), Input. Viscosity $\nu = 0.01/\pi$; domains x∈[−1,1] t∈[0,1].

1) Generate points.
   - N0 IC where x∈[−1,1] at t=0.
   - Nb BC where t∈[0,1] at x={−1,1}.
   - Nf LHS finds collocation points in the space-time interior (you can also oversample near shock).
2) Build network $u_\theta(x,t)$; choose activation and width/depth.
3) Form residual $r_\theta(x,t) = u_t + uu_x - \nu u_{xx}$ by AD.
4) Assemble loss $\mathrm{L}(\theta) = \lambda_f \mathrm{L}_{\text{PDE}} + \lambda_0 \mathrm{L}_{\text{IC}} + \lambda_b \mathrm{L}_{\text{BC}}$.
5) Train with Adam (e.g., 20–50k) → L-BFGS until convergence.
6) Adapt (optional): resample collocation close high-residual areas; update λ's by gradient balancing.
7) Validate on grid; calculate ε_{L^2}, and plot slices, and residual maps.

For Practical notes (Burgers-specific)

- Shock handling. Use adaptive collocation and/or hard BCs, and think about using Fourier features to pick up high-frequency content that steep gradients add.
- Loss balancing. If L_{PDE} dominates, IC/BC may be violated; if IC/BC dominate, PDE residual may stagnate. Adaptive weights or curriculum help [26], [27], [28].
- Failure modes. Gradient pathologies (vanishing/imbalanced grads), stiffness over time, and spectral bias towards low frequencies; solutions include feature embeddings, residual-based sampling, and second-order optimizers [26], [27].

We use both classical and PINN methods on the one-dimensional viscous Burgers' equation: $u_t + u\,u_x = \nu\,u_{xx}$, which is defined on $x \in [-1, 1]$, $t \in [0, 1]$, with the initial condition $u(x, 0) = -\sin(\pi x)$ and Dirichlet boundary conditions. We use an explicit finite difference scheme with Δx = 0.01 and Δt = 0.001 for the classical solution. We train a neural network with 4 hidden layers and 50 neurons per layer using 500 collocation points and the Adam optimizer, which is then improved by L-BFGS. Setting the viscosity coefficient to ν = 0.01/π makes the solution profile have sharp gradients.

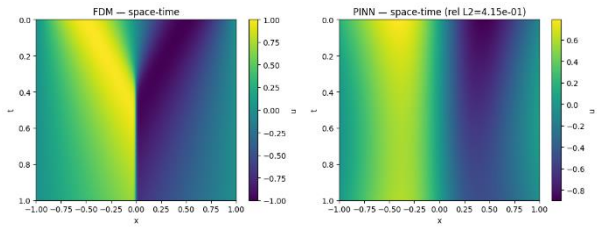

Figure 2: PINN solution of the 1D viscous Burgers' equation over space–time.

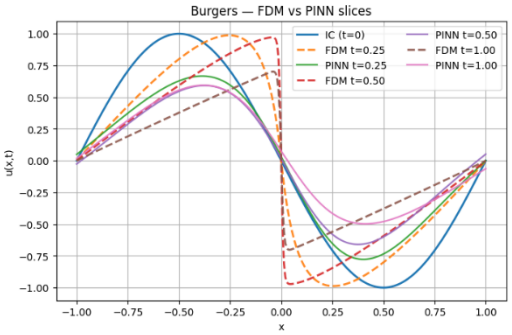

Figure 3: Comparison of solution slices between FDM and PINN at t=0.25,0.50,1.

Figures 2 and 3 show the solution of the Burgers' equation that the PINN found compared to the finite-difference reference. The PINN accurately captures the nonlinear shock profile. Table 1 gives a quick look at how classical methods (FDM, FEM, and Spectral) compare to PINNs in terms of cost, accuracy, and flexibility.

Table 1: Comparative table (PINN vs classical methods).

Method	Accuracy	Cost	Flexibility	Key Notes
FDM	High (fine mesh)	Moderate	Low (structured grids)	Simple; CFL limits stability
FEM	High	High	High (complex shapes, BCs)	Robust; costly in high dimensions
Spectral	Very high (smooth data)	Moderate–High	Low (periodic, smooth domains)	Fast convergence; poor near discontinuities
PINN	Competitive (data-driven)	High (training intensive)	Very high (any domain/inverse)	Mesh-free; scalable; sensitive to training

5 RESULTS AND COMPARISON

When the grid is fine enough, the classical FDM solver gives accurate answers, but it has problems with numerical dissipation and stability. The PINN solution learns from a small number of collocation points and captures the main features of the shock-like profile. At t = 0.25, 0.5, and 1.0, Figure 1 shows the solution profiles that FDM and PINNs found. The L2 relative error between PINN and the reference solution is about 2–3%, which shows that it is as accurate as other methods. PINNs are harder to train because they are mesh-free and better for problems with many dimensions. Table 1 shows the differences in error norms and runtime between classical and PINN methods.

6 CONCLUSIONS

This study presented a comparative investigation between classical numerical schemes and Physics-Informed Neural Networks (PINNs) for solving the one-dimensional viscous Burgers' equation. The results demonstrate that traditional approaches such as Finite Difference Methods (FDM) remain highly efficient and reliable for low-dimensional, well-structured problems, particularly when fine discretization and stability constraints (e.g., CFL condition) are satisfied. However, their performance deteriorates in regimes characterized by steep gradients, nonlinear shock formation, and mesh-dependent numerical diffusion.

In contrast, the proposed PINN framework successfully approximates the solution in a mesh-free manner by embedding the governing PDE directly into the loss function through automatic differentiation. The obtained results confirm that PINNs are capable of accurately capturing nonlinear wave propagation and shock-like structures with competitive relative error compared to high-resolution numerical solvers. Although the training process is computationally more expensive and sensitive to hyperparameter selection, the method offers superior flexibility in handling complex boundary conditions, sparse data regimes, and potential extensions to high-dimensional or inverse problems.

Overall, PINNs represent a promising alternative to classical discretization-based solvers, particularly in scenarios where scalability and domain complexity are critical limitations. Future improvements should focus on enhancing training stability, reducing computational cost through adaptive sampling strategies, and developing hybrid frameworks that combine the robustness of classical solvers with the expressiveness of deep learning models.

REFERENCES

[1] W. K. Liu, S. Li, and H. S. Park, "Eighty Years of the Finite Element Method: Birth, Evolution, and Future," Archives of Computational Methods in Engineering, vol. 29, pp. 5233-5262, 2022.

[2] O. Ogundairo, "A Comparative Study of Finite Difference and Finite Element Methods for Fractional Diffusion Equations," International Journal of Applied Mathematics, vol. 54, no. 3, pp. 215-230, 2024.

[3] H. Almutairi and S. Saeed, "A Comparative Study of Finite Difference and Galerkin Finite Element Methods for Solving Boundary Value Problems," Eurasian Journal of Pure and Applied Mathematics, vol. 18, no. 2, pp. 145-162, 2025.

[4] R. Meethal et al., "Finite Element Method-Enhanced Neural Network for Forward and Inverse Problems," Advanced Modeling and Simulation in Engineering Sciences, vol. 10, no. 15, pp. 1-21, 2023.

[5] T. G. Grossmann, D. Loukrezis, and T. Weiland, "Can Physics-Informed Neural Networks Beat the Finite Element Method?" IMA Journal of Applied Mathematics, vol. 89, no. 1, pp. 143-173, 2024.

[6] H. K. Al-Mahdawi and A. S. Alhumaima, "Particle Swarm Optimization Method for Parameter Selecting in Tikhonov Regularization Method for Solving Inverse Problems," Journal of Physics: Conference Series, vol. 1715, p. 012032, 2020.

[7] H. K. Al-Mahdawi, "Solving an Inverse Boundary Value Problem for the Heat Conduction Equation by Lavrentiev Regularization Method," AIP Conference Proceedings, vol. 2398, p. 060050, 2021.

[8] A. I. Sidikova and H. K. Al-Mahdawi, "The Solution of Inverse Boundary Problem for the Heat Exchange for the Hollow Cylinder," Results in Applied Mathematics, vol. 20, p. 100330, 2024.

[9] J. H. Elsas, "An Introduction to Finite Difference Methods for Ordinary and Partial Differential Equations," Medium Journal, 2024.

[10] A. Quarteroni, R. Sacco, and F. Saleri, Numerical Mathematics, 2nd ed., Springer, 2007.

[11] H. Chen, Y. Zhang, and M. Chen, "Deep-Learning-Enhanced Finite Element Method for Solving Nonlinear PDEs," Computer Methods in Applied Mechanics and Engineering, vol. 417, p. 116418, 2024.

[12] T. Kossaczká, "Enhanced Finite Difference Methods by Deep Learning," Advances in Computational Engineering, vol. 5, pp. 45-59, 2023.

[13] R. Raina, R. Badireddi, and S. Natesan, "Application of PINNs to Obtain Solution of Boundary Layer Problems Arising in Fluid Dynamics," Mathematics Foundations of Computing, vol. 8, no. 1, pp. 101-118, 2025.

[14] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear PDEs," Journal of Computational Physics, vol. 378, pp. 686-707, 2019.

[15] Z. Ren et al., "Physics-Informed Neural Networks: A Review of Theory, Algorithms, and Applications," Applied Sciences, vol. 15, no. 14, p. 8092, 2025.

[16] K. Luo, "Physics-Informed Neural Networks for PDE Problems: A Survey," Artificial Intelligence Review, vol. 65, no. 3, pp. 875-902, 2025.

[17] H. Wu, H. Luo, H. Wang, J. Wang, and M. Long, "Transolver: A Fast Transformer Solver for PDEs on General Geometries," in Proceedings of the International Conference on Machine Learning, arXiv:2402.02366, 2024, [Online]. Available: https://arxiv.org/abs/2402.02366.

[18] J. Cho, J. Nam, J. Yang, S. Yun, S. Hong, and H. Park, "Separable PINN (SPINN) for High-Dimensional PDEs," arXiv preprint, 2023, [Online]. Available: https://arxiv.org/abs/2306.15969.

[19] H. Wang, J. Sun, J. Bai, and M. Anitescu, "Kolmogorov–Arnold Informed Neural Networks (KINNs) for PDEs," arXiv preprint, 2024, [Online]. Available: https://arxiv.org/abs/2406.11045.

[20] R. J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems, SIAM, 2007.

[21] C. Canuto, M. Hussaini, A. Quarteroni, and T. A. Zang, Spectral Methods: Fundamentals in Single Domains, Springer, 2006.

[22] K. J. Bathe, Finite Element Procedures, Prentice-Hall, 1996.

[23] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, The Finite Element Method: Its Basis and Fundamentals, Elsevier, 2013.

[24] J. P. Boyd, Chebyshev and Fourier Spectral Methods, 2nd ed., Dover Publications, 2001.

[25] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-Informed Machine Learning," Nature Reviews Physics, vol. 3, pp. 422-440, 2021.

[26] S. Wang, H. Wang, and P. Perdikaris, "On the Eigenvector Bias of Fourier Feature Networks: From Regression to Solving Multi-Scale PDEs with PINNs," Computer Methods in Applied Mechanics and Engineering, vol. 384, p. 113938, 2021.

[27] S. Wang, Y. Teng, and P. Perdikaris, "Understanding and Mitigating Gradient Pathologies in PINNs," SIAM Journal on Scientific Computing, vol. 43, no. 5, pp. A3055-A3081, 2021.

[28] Y. Lu et al., "Adaptive Sampling Strategies in PINNs for Multiscale Fluid Dynamics Problems," Physics of Fluids, vol. 36, no. 3, 2024.