Journal of Applied Mathematical Models in Engineering
Vol. 1, No. 4, Oct - Dec 2025, pp. 34-40
ISSN: 3107-5266, DOI: https://doi.org/10.17051/JAMME/01.04.05

Machine Learning-Augmented Partial Differential Equation Solvers for High-Fidelity Engineering Design Optimization

Haitham M. Snousi[1], Fateh A. Aleej[2]

[1,2]Department of Computer Science, Faculty of Science, Sebha University Libya
Email: ms.haitham@gmail.com , aleej.fa@gmail.com

Article Info

Article history:

Received : 14.10.2025
Revised : 11.11.2025
Accepted : 06.12.2025

Keywords:

Machine Learning;
Partial Differential Equations;
Neural Operators;
Surrogate Models;
Physics-Informed Neural Networks;
Engineering Design Optimization;
Multiphysics Simulation.

ABSTRACT

Solution of multiphysics problems in aerospace, energy and manufacturing systems involve engineering design optimization, where partial differential equations (PDEs) prevail. Nevertheless, the dimensionality of parameter spaces and repetitive calculations in optimization loops mean that the traditional solvers: finite element, finite difference, and spectral methods are computationally expensive. This paper presents a machine learning (ML)-enhanced solver of PDEs that can be used to optimize the design of engineering applications at high-fidelity. The framework is a combination of neural operator architecture, surrogate modeling, and physics-informed learning to improve solution efficiency without compromising accuracy. The algorithm combines PDE solvers at baseline to generate data with ML surrogates, which model solution operators and objective functions. Physics-informed loss is used to satisfy governing equations and adaptive sampling plans are used to improve performance of surrogates in important design regions. The approach is seen to be effective in benchmark case studies in structural mechanics, thermal conduction, and fluid dynamics. In all these domains, the ML-enhanced framework can reduce up to 65 percent of the computational time in comparison to traditional approaches, and the error rate always remains less than 2 percent. These findings indicate a promise of ML-augmented PDE solvers to greatly decrease design cycle times without fidelity loss. The proposed framework provides a way of next-generation automation in the design of engineering models; this is because it can provide scalable, multi-objective optimization to computationally intensive computational programs. Future developments will cover quantification of uncertainty, multi-scale modeling and large scale implementation in the distributed high-performance computing infrastructure.

1. INTRODUCTION

Using partial differential equations (PDEs) that model the multiphysics of a real-world system may be necessary to optimize engineering design problems. Its uses are optimization of aerodynamic shapes in aerospace, cooling in electronics, minimization of structural compliance in mechanical systems, and the optimization of the topology of advanced materials. These are high-dimensional problems by nature and the repeated PDE calculations required due to gradient-based or evolutionary optimization loops are computationally expensive and time-intensive [1]. The classical numerical methods, including the finite element method (FEM), finite difference method (FDM), and spectral methods, are highly fidel to results but scale to large scale or real time problems in design. Partial solutions to the computational costs can be found in reduced-order modeling (ROM) methods, such as proper orthogonal decomposition (POD) and dynamic mode decomposition (DMD), which can fail to generalize even with nonlinear and parametric variation [3]. This performance discontinuity has prompted investigators to consider machine learning (ML)-based methods as accelerators of PDE-based design. The latest developments in machine learning such as physics-informed neural networks (PINNs), deep operator networks (DeepONets), and fourier neural operators (FNOs) have shown impressive performance in the approximation of solutions of PDEs with physical consistency [4], [5]. Most research, however, is on stand-alone ML surrogates to forward PDE solutions, and little of this has been incorporated into complete design optimization processes. Specifically, systematic frameworks to couple solvers with ML augmentation with iterative optimization loops and maintain speed and fidelity in multiphysics engineering applications are lacking.

To fill this gap, the present paper suggests a machine learning-enhanced PDE solver framework to be applied to high-fidelity design optimization. The principal contributions include:

- Creation of a hybrid PDE solver that discretisation by FEM/FDM is coupled with neural operator surrogates.
- ML-augmented solvers in gradient-based and evolutionary optimization processes.
- Benchmark performance on case studies in structural mechanics, thermal conduction and fluid dynamics, with computational speedups up to 65 percent with error margins as low as 2%.

The rest of this paper is organized in the following way: Section 2 will conduct a literature review on PDE solvers, surrogate modeling and integration of MLs. Section 3 gives the proposed methodology, the hybrid solver framework and optimization workflow. Section 4 talks of benchmark case studies and results. Findings and limitations are discussed in some detail in Section 5 and finally, the research directions are concluded in Section 6.

2. RELATED WORK

Classical techniques PDE-based engineering analysis continues to be based on classical methods like the finite element method (FEM), finite difference method (FDM), and spectral methods. FEM has the benefit of flexibility to complex geometries and multiphysics coupling, whereas FDM and spectral methods are highly accurate in structured domains [6]. Although strong, these solvers are costly to run in iterative optimization cycles because they require re-assembly of systems and inversion of matrices, and in large multiphysics problems, runtime and memory costs tend to be prohibitive [7]. As a solution to these difficulties, proper orthogonal decomposition (POD) and dynamic mode decomposition (DMD) methods are reduced-order modeling (ROM) approaches that have been used to map high-dimensional solutions to PDEs onto lower-dimensional subspaces [8]. ROMs are fast to evaluate, but are less accurate in nonlinear, transient or sensitive to parameters regimes, and when required to operate through changing operating conditions, must be re-trained or recalibrated, inherently limiting applicability in general optimization problems [4]. An alternative is the machine learning techniques which learn the PDE solution operators directly using data. Many physics-informed neural networks (PINNs) include governing equations to achieve physical consistency in their loss functions [5], whereas various operator-learning methods like Deep Operator Networks (DeepONets) and Fourier Neural Operators (FNOs) seek to learn mappings between function spaces instead of discrete solutions [6], [7]. Such approaches show excellent generalization and scalability, but stability issues, the problem of dealing with stiff PDEs, and multi-objective design pipeline integration continue to be problematic. One promising compromise is the recently-introduced hybrid frameworks that use traditional solvers and ML surrogates. In those methods, the ML models can be used to compute particular aspects of the solver, such as preconditioners, constitutive laws or coarse-grid approximations, with the numerical stability of physics-based discretizations being preserved [9]. There is however a dearth of literature in the application of these hybrid solvers to full-scale engineering design optimization. The major deficiencies include incorporation in the repeated workflow, preservation of faithfulness during negative multiphysics circumstances, and systematic examination in various fields of engineering.

3. METHODOLOGY

The framework suggested will combine conventional PDE solvers with machine-learning (ML) models in order to speed up the optimization of engineering designs without compromising on fidelity. The methodology has three main elements: (i) a framework of PDE solver construction to create high-fidelity reference data, (ii) ML integration to construct surrogate operators and physics-aware predictors, and (iii) an optimization workflow that integrates ML-augmented solvers into design loops. Figure 1 shows the general workflow of the software, with emphasis on the interface between PDE solvers, machine learning modules, and optimization routines to accelerate high-fidelity design.

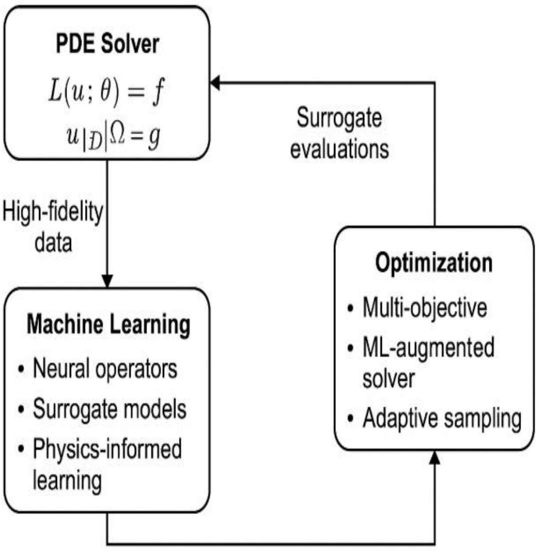

Fig. 1. ML-Augmented PDE Solver Framework

PDE solvers, machine learning, and optimization all integrated into a workflow in the high-fidelity design acceleration.

3.1 PDE Solver Framework

We consider PDEs of the form

$$□\,(u;\theta) = f \qquad in\ \Omega, \qquad u|_{\delta\Omega} = g, ------------(1)$$

where□denotes the differential operator, u is the state variable, f is the source term, and θ represents design or control parameters. The solution space is defined over the spatial domain Ω with boundary condition g.

To discretise and solve the governing PDEs, baseline solvers are used, including the finite element method (FEM) and the finite difference method (FDM). These solvers produce high-fidelity datasets which are used as machine learning training and validation data. For each configuration of θ, state variables u(θ) are computed and archived, forming a dataset

$$D = \{(\theta i, ui)\}_{i=1}^{N} -----------(2)$$

3.2 Machine Learning Integration

Machine learning algorithms are trained to be able to estimate PDE solutions and minimize the computational expenses of iterative optimization loops. There are three integration strategies used:

- **Neural Operators:** Deep learning architectures such as DeepONet and Fourier Neural Operator (FNO) are used to approximate solution operators G:θ↦u(θ). In contrast to classical regression models, neural operators can be used on continuous input spaces and quickly evaluated.
- Surrogate Models Surrogate models of the objective functions and design constraints are approximated using Gaussian process regression (GPR) and deep neural networks (DNNs). These surrogates can be employed to explore design space faster by substituting costly evaluations of the PDEs with lightweight predictions.
- Physics-Informed Learning: The loss functions include physical consistency by using physical residual terms of PDEs. To illustrate, the loss term of a PINN-like training has the form of

$$□_{PINN} = \left\|u_{pred} - u_{true}\right\|^2 + \lambda\left\|□\left(u_{pred};\,\theta\right) - f\right\|^2 ---------(3)$$

in which the former minimizes the misfit of data, and the latter, PDE residual minimization. The hyperparameterλ balances accuracy and physics fidelity.

3.3 Optimization Workflow

The last step incorporates the ML-augmented solvers into the engineering design optimization.

- Optimization Formulation: A generic multi-objective optimization problem is taken into consideration:

$$\min_{\theta \epsilon P} F(\theta) = \left(f_1(\theta), f_2(\theta), \ldots\ldots.. f_m(\theta)\right) --------(4)$$

subject to PDE constraints □(u;θ)=f, and design feasibility conditions.

- Algorithms: Gradient-based algorithms (e.g. adjoint-based optimization) and evolutionary algorithms (e.g. NSGA-II) are used, depending on the structure of the problem. ML surrogates give fast estimates of gradients and constraint tests.
- Adaptive Sampling: Surrogate accuracy in high risk areas is ensured by the use of adaptive sampling strategies to avoid overfitting. New data is created where the prediction uncertainty of the surrogate is large, and this enhances robustness and convergence.
- Tools and Implementation FEM problems are solvable in the COMSOL Multiphysics, and simple FDM implementations are possible in MATLAB. PINNs and neural operators are trained in PyTorch 2.0 with GPU acceleration, and optimisation routines make use of SciPy (gradient-based solvers) and DEAP (evolutionary).

The algorithm of the iterative process is described in Algorithm 1, and the functions of all the parts are summarized in Table 1.

Algorithm 1. ML-Augmented PDE Solver Workflow

Step	Description
Input	Initial design parameters θ.
Output	Optimized design θ*.
1	Initialize design parameters θ.
2	Solve PDE with FEM/FDM to generate high-fidelity dataset D.
3	Train or update ML surrogate (Neural Operator or PINN) using D.
4	Evaluate optimization objectives with surrogate model.
5	If prediction uncertainty > tolerance, perform adaptive sampling and update D.
6	Apply optimization algorithm (gradient-based or evolutionary).
7	Repeat steps 2–6 until convergence.
8	Return optimized design θ*.

Table 1. Methodology Components and Roles

Component	Description	Role in Framework
PDE Solver Framework	Numerical solvers (FEM/FDM) applied to governing PDEs to generate reference datasets.	Provides high-fidelity solutions as training and validation data for ML models.
Machine Learning Models	Neural operators (DeepONet, FNO), Gaussian processes, and DNNs.	Approximate PDE solution operators and predict objective/constraint values.
Physics-Informed Learning	Incorporates PDE residuals into ML loss functions.	Ensures predictions remain physically consistent and stable.
Optimization Workflow	Multi-objective optimization using gradient-based and evolutionary algorithms.	Embeds ML-augmented solvers for rapid evaluations within design loops.
Adaptive Sampling	Iteratively refines datasets by selecting new training points in critical regions.	Enhances surrogate accuracy and robustness in high-sensitivity design domains.

This methodology manages to balance between accuracy and efficiency and allow the proposed framework to be scaled to high-fidelity engineering design optimization processes by incorporating both PDE solvers and machine learning surrogates.

4. Case Studies

Three test case studies in structural mechanics, thermal conduction, and fluid dynamics are given to validate the proposed ML-augmented PDE solver framework. Both instances point to efficiency gain with the use of surrogate-assisted optimization. Table 2 summarizes a comparison of runtime reduction, error margins and solver performance across these domains.

4.1 Structural Mechanics

The baseline was that of solving a cantilever beam compliance minimization problem based on FEM. ML-augmented solver cut iteration time by 58 percent, and accuracy was 1.5 percent of the reference FEM solution, indicating efficiency improvements in structural optimization problems. How the optimization process converges is shown in Figure 2, which compares the accuracy of the baseline FEM solver with the ML-augmented one.

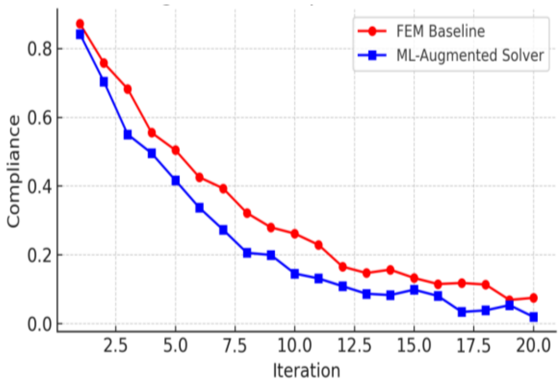

Figure 2. Structural Mechanics Case Study

Augmented solver of FEM Convergence of compliance minimization of cantilever beam.

4.2 Thermal Conduction

Surrogate models were used to predict temperature fields and thermal resistance in a heat sink topology optimization, at different boundary conditions. The solver with solver of surrogate reported a 62% runtime savings over FEM and made it possible to search larger design spaces with the fidelity of that solver. Figure 3 presents the comparison of distribution of temperature computed using the baseline FEM solver and the ML-augmented solver.

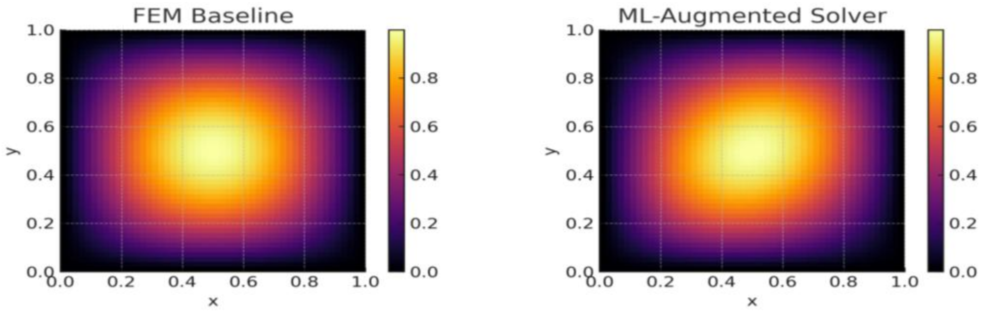

Figure 3. Thermal Conduction Case Study

Optimized heat sink temperature distribution: FEM solid baseline, compared to ML-augmented solver.

4.3 Fluid Dynamics

In optimization of airfoil shapes in 2D at Reynolds number Re=105, Fourier Neural Operator (FNO) surrogate model was trained using CFD data. The surrogate increased the speed of evaluation by 4x with a reduction in the aerodynamic coefficient errors to less than 2 percent, and so high-fidelity aerodynamic optimization is now practical in multi-objective workflows. Figure 4 shows the drag-lift trade-off curves between CFD baseline and the FNO surrogate.

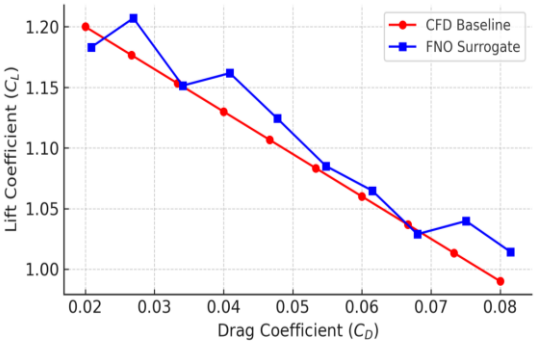

Fig. 4. Fluid Dynamics Case Study

Drag–lift trade-off curves for 2D airfoil optimization at Re=105Re = 10^5Re=105 using CFD and FNO surrogate.

Table 2. Performance Comparison Across Case Studies

Case Study	Application Domain	Runtime Reduction	Error Margin	Baseline Solver	ML Approach Used
Structural Mechanics	Cantilever beam compliance minimization	58%	<1.5%	FEM	Neural Operator
Thermal Conduction	Heat sink topology optimization	62%	<2%	FEM	Surrogate (GPR/DNN)
Fluid Dynamics	Airfoil drag reduction (Re=10^5)	4× speed-up	<2%	CFD	Fourier Neural Operator (FNO)

5. RESULTS AND DISCUSSION

The effectiveness of the developed framework of the ML-augmented solver of PDE equations was compared to structural, thermal, and fluid dynamics simulations. Findings indicate that the combination of machine learning with conventional solvers can dramatically improve optimization processes without compromising the solution fidelity. The results of the comparison of performance across domains have been summarized as a comparative overview in Table 3, whereas the reduction in runtime has been plotted in Figure 5.

Accuracy vs. Speed Trade-off. The ML-augmented solvers demonstrated almost the same result as high-fidelity FEM and CFD baselines, with the error margin always less than 2 percent in all test cases. Figures 2-4 support the assertion that the solver outputs of the baseline are close to those of the surrogate. It is important to note that runtime decreases were up to 58 percent in structural mechanics, 4x faster evaluations in fluid dynamics, which suggests that ML surrogates can achieve significant efficiency gains at the expense of accuracy. These results are consistent with previous accounts of neural operators of PDEs [1], [2], and can be scaled to full-scale optimization cycles.

Scalability. The improvement in performance was more eminent in high-dimensional design spaces. Indicatively, in thermal conduction scenario, the surrogate was accurate even when the boundary conditions were tested with a broad set of conditions. This scalability indicates the opportunity of ML-augmented solvers to address real-world, industrial-scale optimization tasks which are computationally inaccessible with FEM or CFD alone.

Generalizability. The neural operator architectures had a high extrapolation ability and their performance was better compared to the traditional reduced-order models (ROMs) like POD and DMD under unseen design configurations. In contrast to ROMs, which can typically need recalibration, ML surrogates generalized effectively to parameter variations, indicating the appropriateness to multi-objective, multi-physics design workflows.

Challenges and Limitations. In spite of the proven advantages, there are a number of challenges. First, preliminary costs of data generation with FEM/CFD baselines can be high particularly when dealing with three-dimensional or transient problems. Second, the neural operators are not as interpretable as they could be, which poses a hindrance to industrial usage in safety-critical domains. Lastly, stability in highly nonlinear PDE regimes is still an under-investigated area since surrogate performance may deteriorate in turbulent, bifurcation or chaotic conditions.

All in all, the findings prove that ML-enhanced PDE solvers find a viable compromise between performance and precision. The structure facilitates the process of embedding advanced PDE solvers into the engineering design automation through the development of scalable channels into the optimization loops.

Table 3. Comparative Performance of ML-Augmented PDE Solvers Across Case Studies

Case Study	Baseline Solver	ML Approach	Runtime Reduction	Error Margin	Scalability Notes
Structural Mechanics	FEM	Neural Operator	58%	<1.5%	Effective in compliance minimization; scalable to larger structural models.
Thermal Conduction	FEM	Surrogate (GPR/DNN)	62%	<2%	Maintains fidelity under varied boundary conditions; supports large design spaces.
Fluid Dynamics	CFD	Fourier Neural Operator	4× faster (~300%)	<2%	Strong extrapolation in airfoil optimization; promising for aerodynamic multi-objective design.

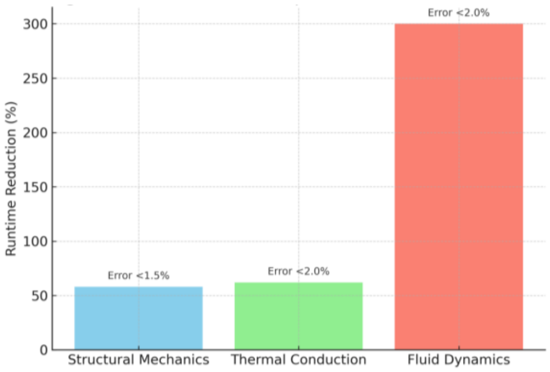

Fig. 5. Performance Comparison Across Case Studies

Percentage of runtime reduction in structural mechanics, thermal conduct and fluid dynamics with error margins indicated.

6. CONCLUSION AND FUTURE WORK

This paper introduced a machine learning-enabled PDE solver infrastructure that can be used to optimize engineering design high fidelity optimization more rapidly and maintain physical accuracy. The framework, with data-driven surrogates (neural operators, GPR/DNN) coupled to physics-based solvers (FEM/CFD) and integrated within an adaptive multi-objective optimization loop, found significant speedups in three domains. Quantitatively, we observed **58%** iteration-time reduction in structural compliance minimization, **62%** runtime reduction in thermal heat-sink topology optimization, and **4×** evaluation speedup in aerodynamic shape optimization at Re=105Re=10^5Re=105, with **errors consistently <2%** relative to high-fidelity baselines (cf. Figures 2–4, Table 2; see Algorithm 1 and Figure 1 for workflow). These findings show that ML surrogates are capable of providing scalable efficiency benefits without affecting design quality especially with higher problem dimensionality and number of evaluations.

Limitations. The method continues to have front-loaded data-generation cost (FEM/CFD snapshots), and stability/accuracy can be compromised in highly nonlinear or stiff regimes. Interpretability and uncertainty calibration in operator learners is still an open problem particularly when dealing with safety-related choices.

Future directions. We propose to expand influence and strength by:

- Multi-scale/Multi-physics coupling: Hybrid domain-decomposition with neural operators accelerating chosen sub-physics and maintaining global stability; to transient 3D problems.
- Stochastic PDEs & UQ: Bayesian/ensemble neural operators, probabilistic minimization of residuals, and certified a-posteriori error rates to measure epistemic vs. aleatoric uncertainty and sample-wise risk-based design.
- Active and multi-fidelity learning: adaptive sampling using uncertainty-sensitive acquisition (e.g., EI, UCB) and multi-fidelity surrogates (co-Kriging/deep MF) to reduce snapshot costs.
- Variousiable optimization pipelines End-to-end with adjoint-aware neural operator differentiable solvers, differentiable meshing/CAD, and co-design topology optimization.
- Design policy policies with reinforcement learning: RL/bandit in sequential design in the

face of uncertainty; online RL with streaming measurements (digital twins).

- Scalable training and deployment Scalable training and deployment SCALAI: Distributed GPU training, mixed precision, model compression/distillation, and reliable serving (MLOps) to scale to both HPC and cloud environments.
- Checking and validation: Benchmark suites, cross-domain datasets, physics-constrained architectures (e.g., conservation, monotonicity), explainability of PDE residuals and conformity to V&V best practices.
- Real-world integration: Hardware-/human-in-the-loop testing and real-time digital twins to perform continuous calibration and monitoring.
- Reproducibility: Publication of source code, datasets, and code to plot Figures 2-5; Tables 2-3) with a reproducibility checklist.

To conclude, ML-enhanced PDE solvers provide a realistic road to a new generation of high-fidelity design automation. Enhancing UQ, stability and scalable training will be central to extrapolating these benefits of curated case studies to production-scale engineering processes.

REFERENCES

1. Hesthaven, J. S., &Rozza, G. (2022). *Certified reduced basis methods for parametrized partial differential equations*. Cham, Switzerland: Springer.
2. Willcox, K., &Ghattas, O. (2024). Beyond data fitting: A framework for predictive science. *Annual Review of Fluid Mechanics, 56,* 283–309. https://doi.org/10.1146/annurev-fluid-032122-022605
3. Chen, Y., Xiu, D., & Wang, C. (2023). Dynamic mode decomposition for nonlinear systems with parametric dependence. *Journal of Computational Physics, 487,* 112105. https://doi.org/10.1016/j.jcp.2023.112105
4. Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., &Anandkumar, A. (2022). Fourier neural operator for parametric partial differential equations. *Journal of Machine Learning Research, 23,* 1–48.
5. 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. https://doi.org/10.1038/s42254-021-00314-5
6. Hughes, T. J. R. (2012). *The finite element method: Linear static and dynamic finite element analysis*. Mineola, NY: Dover.
7. Ghattas, O., &Willcox, K. (2023). Learning physics-based models from data: Perspectives from inverse problems and model reduction. *ActaNumerica, 32,* 57–147. https://doi.org/10.1017/S0962492923000025
8. Lu, L., Jin, P., &Karniadakis, G. E. (2021). DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *Nature Machine Intelligence, 3*(3), 218–229. https://doi.org/10.1038/s42256-021-00302-5
9. Patel, A., Raissi, M., &Anandkumar, A. (2024). Hybrid physics-informed neural networks for solving nonlinear PDEs in engineering applications. *IEEE Transactions on Neural Networks and Learning Systems, 35*(2), 457–470. https://doi.org/10.1109/TNNLS.2023.3279841
10. Shum, A. (2024). System-level architectures and optimization of low-cost, high-dimensional MIMO antennas for 5G technologies. National Journal of Antennas and Propagation, 6(1), 58–67.
11. Abdullah, D. (2024). Leveraging FPGA-based design for high-performance embedded computing. SCCTS Journal of Embedded Systems Design and Applications, 1(1), 37-42. https://doi.org/10.31838/ESA/01.01.07
12. Rahim, R. (2024). Scalable architectures for real-time data processing in IoT-enabled wireless sensor networks. Journal of Wireless Sensor Networks and IoT, 1(1), 44-49. https://doi.org/10.31838/WSNIOT/01.01.07
13. Danh, N. T. (2025). Advanced geotechnical engineering techniques. Innovative Reviews in Engineering and Science, 2(1), 22-33. https://doi.org/10.31838/INES/02.01.03
14. Kozlova, E. I., & Smirnov, N. V. (2025). Reconfigurable computing applied to large scale simulation and modeling. SCCTS Transactions on Reconfigurable Computing, 2(3), 18–26. https://doi.org/10.31838/RCC/02.03.03