Finite Volume Neural Network (FVNN) with Reduced Derivative Order for Incompressible Flows

Zijie Su[1], Mengke Ren[1], Yunpu Liu[1], Sheng Pan[3], Zheng Li[1,*], Changyu Shen[1,2]

[1]State Key Laboratory of Structural Analysis, Optimization and CAE Software for Industrial Equipment, Department of Engineering Mechanics, Dalian University of Technology, Dalian 116024, China

[2]School of Materials Science and Engineering, The Key Laboratory of Material Processing and Mold of Ministry of Education, Zhengzhou University, Zhengzhou 450002, China

[3]Department of Micro Engineering, Kyoto University, 615-8540, Kyoto, Japan

*Correspondence: lizheng@dlut.edu.cn;

Abstract—Physics-Informed Neural Networks (PINN) have emerged as a powerful tool for solving partial differential equations (PDEs) and have been extensively applied in various fields such as energy, environment, and engineering. Typically, PINNs utilize Automatic Differentiation (AD) to compute the residuals of governing equations, which may lead to precision loss. Although recent studies have attempted to integrate traditional numerical methods with PINNs, these approaches are mostly data-driven and remain dependent on traditional numerical solvers for generating training data. In contrast, this paper proposes a purely physics-driven Finite Volume Neural Network (FVNN) method, completely independent of conventional numerical solvers, specifically designed to solve steady-state incompressible flow problems. Inspired by the Finite Volume Method, the FVNN approach divides the solution domain into multiple grids and employs Gauss's theorem to evaluate the residuals of the Navier-Stokes equations at Gaussian integral points located on grid boundaries, rather than at collocation points within the grids as in traditional PINNs. The loss function is constructed using the Gaussian integral approach, effectively reducing the order of derivatives required for velocity calculations. To validate the effectiveness of the proposed method, we predict velocity and pressure fields for two representative examples in fluid topology optimization. Results are compared against both commercial software and traditional PINNs. Numerical cases demonstrate that the FVNN significantly improves the accuracy of velocity and pressure field predictions while accelerating the network's training speed compared to traditional PINNs.

Keywords: Physics-Informed Neural Networks; Finite Volume Method; Navier-Stokes Equations; Incompressible Flow; Steady-State Problems;

1. Introduction

In recent years, Physics-Informed Neural Network (PINN) (1) has emerged as a promising numerical method that, unlike traditional data-driven machine learning techniques, directly incorporates governing equations and boundary conditions into the loss function. This integration enhances model interpretability and ensures that predictions are more consistent with physical laws. PINN can provide accurate solutions to partial differential equations with minimal or even no prior data. Compared to conventional discrete numerical methods, such as finite difference and finite element methods, PINNs offer greater flexibility in handling complex geometries, as they do not require mesh generation. Furthermore, when applied to high-dimensional problems, PINNs effectively overcome the curse of dimensionality, making them especially suited for tackling such challenges(2). Consequently, PINNs have found widespread applications in various fields, including diffusion equations, materials science, quantum mechanics, solid mechanics, and fluid dynamics(3–7) .

In particular, researchers have successfully applied PINNs to a wide range of fluid dynamics problems, demonstrating strong performance in both laminar and turbulent flow regimes. For example, Raissi et al. (1) pioneered the use of PINNs to model the Navier-Stokes (NS) equations, ,enabling the prediction of velocity and pressure fields for incompressible flows. However, most existing models based on PINNs for solving partial differential equations achieve this by directly embedding the residuals of the governing equations at the collocation points into the loss function(8,9). When applying PINNs to solve incompressible flow, the mass conservation equation Eq.(1) is typically enforced by treating the stream function ϕ of the velocity as the network output. By differentiating ϕ with respect to the spatial coordinates, the velocity components in each direction can be obtained (3). To compute the residuals of the NS equations, the second derivatives of the velocity must be calculated, which effectively requires computing the third derivatives of the network outputs. In this context, Automatic Differentiation (AD) is commonly used to compute the derivatives of the network output with respect to the network inputs (10). To compute first-order derivatives, AD uses both forward and backward passes. In solving high-order partial differential equations, AD can be recursively applied n times to compute nth-order derivatives, leading to substantial computational overhead and potential accuracy loss(11). To address this issue, numerous researchers (12–16) have proposed loss functions in weak form, which reduce the order of derivatives required for the PDEs. Several examples have demonstrated the potential of this approach in solving high-dimensional partial differential equations.

Inspired by the finite volume method, we propose the FVNN. This method leverages Gauss's theorem to reduce the order of the NS equations and employs the reduced integral expressions as the loss function for PINN, thereby enabling unsupervised training. Without requiring any prior data, the steady-state velocity and pressure fields of incompressible laminar flow can be predicted solely based on the reduced integral expressions of the NS equations and the problem's boundary conditions. To systematically explain this theory and verify its effectiveness, Chapter 2 introduces the theoretical framework, loss function construction, and sampling strategies of FVNN. Chapter 3 demonstrates the reliability and effectiveness of the FVNN model through two numerical examples. Finally, the main contributions and findings of this study are summarized in the concluding chapter, along with a discussion of the potential of this method to be further extended to fluid topology optimization.

2. Finite Volume Neural Network (FVNN)

2.1 Governing equation

In this paper, we present FVNN, which is used to predict the velocity and pressure fields for incompressible laminar flow under steady-state conditions. Let us first consider the governing equations of this problem in the steady-state case:

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

$$(\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\nabla^2 \mathbf{u} \tag{2}$$

Where ∇ is the Nabla operator, $\mathbf{u} = (u_1, u_2)$ is the velocity vector (in 2D problem), p is the pressure, μ is the viscosity of the fluid, ρ is the density of fluid. To naturally satisfy the mass conservation equation (1), researchers(1,8) define the stream function ϕ of the velocity as the output of the neural network. This allows the velocity components in each direction to be obtained through Eq. (3).

$$u_1, u_2 = \frac{\partial \phi}{\partial y}, -\frac{\partial \phi}{\partial x} \tag{3}$$

2. 2 Integral Formulation of the NS Equations Using the Finite Volume Method

In solving the NS equations, the Finite Volume Method applies integration of the governing equations over discrete control volumes, converting the differential form into an integral form. This transformation facilitates the handling of complex boundaries and discontinuities. For incompressible flows, to satisfy the mass conservation equation, the stream function ϕ is typically used as the output in PINNs, as previously discussed, eliminating the need for further adjustments. As for the momentum conservation equation, the divergence theorem(17), also known as Gauss's theorem, is applied as shown below to convert volume integrals into boundary integrals, which correspond to the flux of the momentum conservation equations on the cell boundaries.

$$\int_V (\nabla \cdot \mathbf{u}) dV = \oint_S \mathbf{u} \cdot \boldsymbol{n} dS \tag{4}$$

The following section addresses each term in the momentum equation Eq. (2). The inertial term $(\mathbf{u} \cdot \boldsymbol{\nabla})\mathbf{u}$ from momentum equation Eq. (2) is transformed into a surface flux form through Gauss's theorem:

$$\int_{\Omega_i} (\mathbf{u} \cdot \nabla)\mathbf{u} dV = \oint_{\partial\Omega_i} \mathbf{u}(\mathbf{u} \cdot dA) \tag{5}$$

The pressure gradient ∇p is also converted into a surface flux:

$$\int_{\Omega_i} \frac{1}{\rho}\nabla p dV = \oint_{\partial\Omega_i} \frac{p}{\rho} dA \tag{6}$$

For the viscous diffusion term, it is treated as a surface flux on the boundary:

$$\int_{\Omega_i} \frac{\mu}{\rho} \nabla^2 \mathbf{u} dV = \oint_{\partial\Omega_i} \frac{\mu}{\rho} \nabla \mathbf{u} \cdot dA \tag{7}$$

After these transformations, the momentum conservation equation becomes an integral equation over the control volume, expressed the fluxes on the control volume boundary.

$$\begin{aligned}\int_{\Omega} \left[(\mathbf{u} \cdot \nabla)\mathbf{u} + \frac{1}{\rho}\nabla p - \frac{\mu}{\rho}\nabla^2 \mathbf{u} \right] dV \\ = \oint_{\partial\Omega} \mathbf{u}(\mathbf{u} \cdot d\boldsymbol{A}) + \oint_{\partial\Omega} \frac{p}{\rho} d\boldsymbol{A} \\ - \oint_{\partial\Omega} \frac{\mu}{\rho} \nabla \mathbf{u} \cdot d\boldsymbol{A}\end{aligned} \tag{8}$$

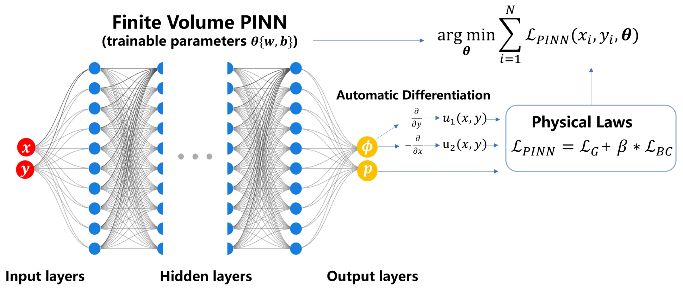
According to the above modifications, the two-dimensional form of the boundary flux integral is as follows:

$$\begin{aligned}\int_{\Gamma} \left[u_1(u_1 n_1 + u_2 n_2) - \frac{\mu}{\rho}\left(\frac{\partial u_1}{\partial x_1} n_1 + \frac{\partial u_1}{\partial x_2} n_2\right) \right. \\ \left. + p n_1 \right] d\Gamma \quad x\ direction\end{aligned} \tag{9}$$

$$\begin{aligned}\int_{\Gamma} \left[u_2(u_1 n_1 + u_2 n_2) - \frac{\mu}{\rho}\left(\frac{\partial u_2}{\partial x_1} n_1 + \frac{\partial u_2}{\partial x_2} n_2\right) \right. \\ \left. + p n_2 \right] d\Gamma \quad y\ direction\end{aligned} \tag{10}$$

Here, u_1 and u_2 represent the velocities in the x- and y-directions at each integration point, respectively; n_1 and n_2 are the components of the normal vector to the cell boundary in the x- and y-direction; and p is the pressure at this point.

Figure 1 Schematic of FVNN. Here, x and y are the horizontal and vertical coordinates of the flow field integration points. $\boldsymbol{\theta}\{w, b\}$ represents the weights and biases of FVNN, which are continuously updated during training.


Finite Volume PINN
(trainable parameters $\theta\{w,b\}$)
$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \mathcal{L}_{PINN}(x_i, y_i, \boldsymbol{\theta})$
Automatic Differentiation
$\frac{\partial}{\partial y}$ $u_1(x,y)$
$-\frac{\partial}{\partial x}$ $u_2(x,y)$
Physical Laws
$\mathcal{L}_{PINN} = \mathcal{L}_G + \beta * \mathcal{L}_{BC}$
x
y
ϕ
p
Input layers
Hidden layers
Output layers

2. 3 Implementation of FVNN

In the present work, the proposed FVNN takes a standard multilayer fully connected neural network as a prototype (shown in

). For this network, we use the hyperbolic tangent (tanh) as the activation function which is suitable to capture nonlinear patterns in flow field, as shown below:

$$\sigma(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{11}$$

The network input is the spatial coordinates $\boldsymbol{x} = (x, y)$ of the sampled points in the flow field, and the outputs are the stream function ϕ and pressure p, as formulated in Eq.(1) and Eq.(2). This approach allows us to obtain a continuous solution that satisfies the weak form of the NS equations. To compute the derivatives of the output quantities with respect to the spatial coordinates, we leverage automatic differentiation, which efficiently calculates both first- and second-order derivatives. Specifically, given the neural network output $f(\boldsymbol{x}, \boldsymbol{\theta})$, where $\boldsymbol{x}$ represents the spatial coordinates and $\boldsymbol{\theta}$ denotes the network parameters, automatic differentiation provides the derivatives:

$$\frac{\partial f}{\partial x}, \frac{\partial^2 f}{\partial x^2}, \frac{\partial^2 f}{\partial x \partial y} \tag{12}$$

These derivatives are then substituted into the integral expressions of the finite volume formulations shown in Eq.(9) and Eq.(10), allowing us to evaluate the governing equations over each cell. By doing so, we ensure that the network output satisfies the integral form of the conservation laws within each finite volume cell.

Furthermore, we introduce a boundary condition loss function $\mathcal{L}_{BC}$, which, along with the physical loss $\mathcal{L}_G$, constitutes the total loss function $\mathcal{L}_{PINN}$:

$$\mathcal{L}_{PINN} = \mathcal{L}_G + \beta * \mathcal{L}_{BC} \tag{13}$$

The physical loss $\mathcal{L}_G$ is computed by evaluating the NS equation residuals after transforming them via the Gauss's theorem into integrals, which are then approximated using Gaussian quadrature. The boundary condition loss function $\mathcal{L}_{BC}$ is computed by substituting the velocity field and pressure field predicted by the FVNN into either Dirichlet or Neumann boundary conditions, depending on the specific problem. β is a constant that balances the contributions of the two types of loss functions.

The PINN minimizes the loss function through gradient descent algorithms, such as Adam and L-BFGS (18,19). Through iterative optimization of the network parameters, the predicted solution gradually converges towards the true physical solution.

2.4 Gaussian Quadrature-based Sampling Strategy and Loss Function Computation

Unlike the uniform sampling of points in the entire flow field in PINN for fluid(1), in this study, the selection of sampling points is based on Gaussian Quadrature(20). Specifically, like Finite Volume Method, the flow field is discretized into elements, and the sampling points involved in the FVNN loss function calculation are located on the boundaries of these elements, as shown in Figure 2. These points are selected through Gaussian Quadrature to ensure that the approximation of physical quantities at each integration point achieves higher accuracy. Gaussian Quadrature allows for efficient sampling in high-dimensional spaces, especially when dealing with complex boundary conditions and flow fields, significantly reducing errors.

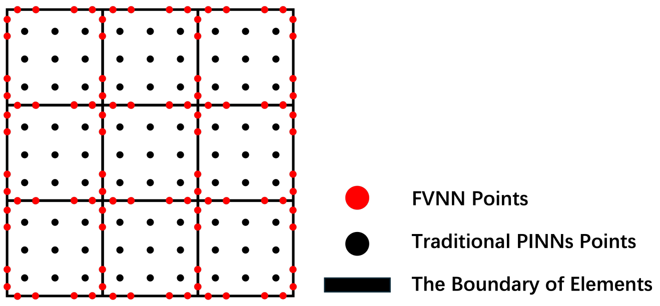


Figure 2 Diagram of integration points.

Through Gaussian Quadrature, we can convert the loss function from an integral form into a discrete numerical form. By applying the Gaussian Quadrature coefficients at the sampling points, we can compute the contribution of loss function at each point, and thus obtain the loss function of the governing physical equations, as shown in Eq.(14).

$$\mathcal{L}_G \approx \sum_{j=1}^{N}\sum_{k=1}^{n}\Bigg(\bigg(u_1(u_1n_1+u_2n_2)-v\left(\frac{\partial u_1}{\partial x_1}n_1+\frac{\partial u_1}{\partial x_2}n_2\right) + pn_1\bigg) + \bigg(u_1(u_1n_1+u_2n_2) - v\left(\frac{\partial u_1}{\partial x_1}n_1+\frac{\partial u_1}{\partial x_2}n_2\right)+pn_1\bigg)\Bigg)_{j,k} w_k \tag{14}$$

Where N represents the number of elements in the flow field, n denotes the number of integration points in each element, and w_k represents the Gaussian quadrature weight.

3. Model Validation and Results Analysis

The pipe bend problem is classic example in fluid topology optimization, first introduced by Borrvall and Peterson in 2003(21). In the following chapter, we will use the proposed FVNN to predict the steady-state velocity and pressure fields for these two cases and compare the results with the commercial software COMSOL and Fluent, thereby validating its effectiveness. The FVNN is implemented through PyTorch and trained on NVIDIA RTX 4090.

3.1 Pipe Bend Problem

This problem is illustrated in Figure 3.a. On the left-hand side, there is an inlet at which parabolic normal velocity profiles are prescribed with a maximum velocity of U_{in} $\left(U_{max}=1\frac{m}{s}\right)$n the bottom, there is a zero-pressure outlet at which the flow is specified to exit in the normal direction. For the steady case, the dynamic viscosity and density of the fluid is $0.02\frac{kg}{m\cdot s}$ and $1\frac{kg}{m^3}$ respectively.

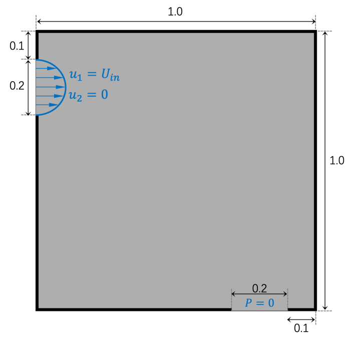


Figure 3 Problem setup

For the boundary conditions, instead of performing additional sampling, we select integration points located at the boundaries, compute the associated boundary condition loss terms $\mathcal{L}_{BC}$, and incorporate them into the final loss function. The Adam optimizer is used for training. The specific neural network parameters are as follows:

TABLE 1. THE PARAMETERS OF FVNN

Parameter	Value
Architecture: layers and neurons	[2,40,40,40,40, 40,40,40,40,2]
Training epochs	20000
Learning rate	0.003
β (scaling $\mathcal{L}_G$ and $\mathcal{L}_{BC}$)	0.0001

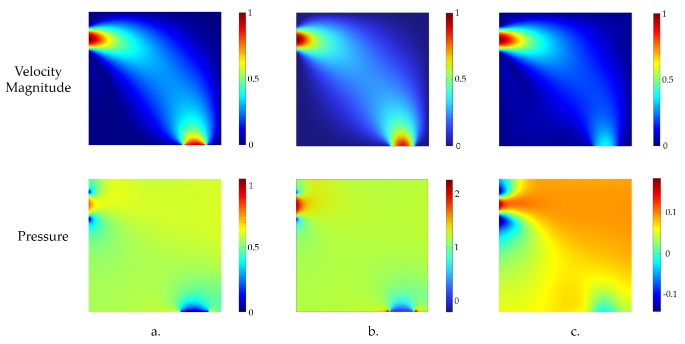


Figure 4. Comparison of Results for the Pipe Bend Problem: (a) Predicted results from FVNN, (b) Computational results from COMSOL, and (c) Predicted results from the traditional NS equation-based PINN.

Figure 4 shows the prediction results of FVNN for the Pipe Bend problem. As can be seen, accurate predictions of the velocity field are achieved, while the pressure field can be reasonably predicted qualitatively. The numerical predictions of the pressure field differ from those of COMSOL. This phenomenon has already been explained by Raissi in (1), where it was stated that the absolute pressure field in incompressible flows is indeterminate and can only be predicted up to an arbitrary constant. The model predicts the relative pressure variations, but the absolute pressure value remains undetermined. It can be also observed that the traditional PINN, which directly incorporates the residuals of the Navier–Stokes equations into the loss function, fails to provide accurate predictions for both the velocity and pressure fields in this problem. Consequently, it can be concluded that FVNN, by reducing the order of differentiation in automatic differentiation, significantly enhances the accuracy of predictions for incompressible flow fields.

Table 2 presents a comparison between FVNN and the traditional NS-based PINN in terms of training time and the number of sampling points (with their respective prediction results shown in Figure 4.a and Figure 4.c). As shown in the table, after 20,000 iterations, FVNN achieves more accurate predictions than the traditional NS-based PINN while utilizing fewer sampling points, resulting in a 40% reduction in training time. Figure 5 illustrates the variations of the loss functions during the training process for both models, demonstrating that FVNN exhibits significantly faster convergence compared to the traditional NS-based PINN.

TABLE 2. COMPARISON OF TRAINING TIME AND SAMPLING POINTS

Parameter	Training Time(s)	The number of Points
FV-PINN	442.72	29280
Traditional PINN	745.77	42202

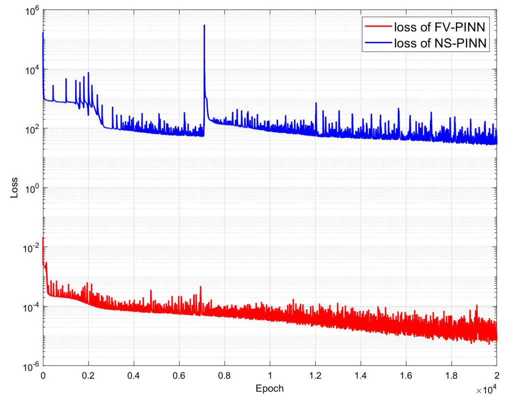


Figure 5. Comparison of Loss Functions: FVNN vs. Traditional PINN

3.2 Steady-state flow past a cylinder problem

This problem is illustrated in Figure 3.b. In this subsection, the classical steady-state flow past a cylinder problem is investigated to further validate the effectiveness of the proposed FVNN. In this case, a uniform inlet velocity profile is applied, and the outlet is subjected to a zero-pressure boundary condition. All other physical parameters, as well as the neural network architecture and hyperparameters, remain consistent with those used in the previous subsection. Figure 6 presents a comparative analysis analysis between the FVNN predictions and the results obtained from the commercial CFD software Fluent. The results indicate that FVNN achieves excellent predictive performance for the steady-state flow past a cylinder.

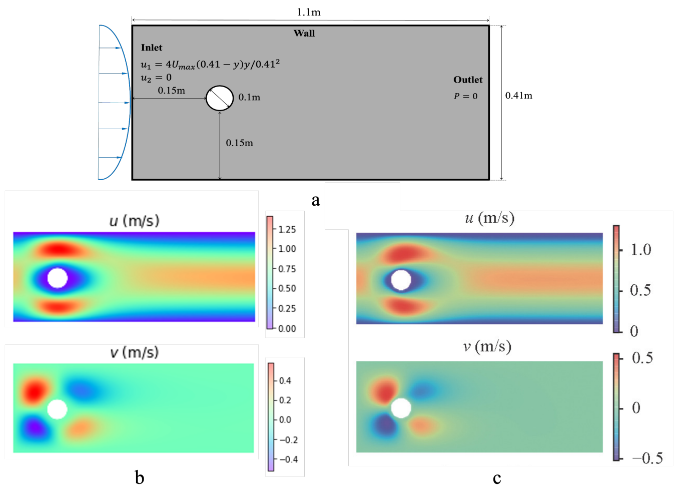


Figure 6 Comparison of Results : (a) Problem setup (b) Predicted results from FVNN, (c) Computational results from Fluent,.

3.3 Summary of Case Study Results

The predictions of FVNN model closely resemble those from commercial softwares, demonstrating a high level of agreement. The accuracy of these predictions highlights the effectiveness of the PINN approach in capturing the underlying flow dynamics, making it comparable to traditional computational methods. Through comparison, it is evident that FVNN significantly outperforms traditional NS-based PINNs in terms of prediction accuracy, achieving more precise results with fewer computational resources. In addition to its enhanced accuracy, FVNN demonstrates a substantial reduction in time cost, as it requires fewer sampling points. Furthermore, FVNN exhibits a much faster convergence rate than conventional NS-based PINNs, making it a highly efficient approach for solving complex flow problems.

4. Conclusions

This paper presents FVNN, a novel Physics-Informed Neural Network inspired by the Finite Volume Method. By leveraging the Gauss's theorem to reformulate the residuals of Navier-Stokes equations, FVNN reduces the reliance on high-order derivatives and achieves enhanced prediction accuracy and faster convergence. The proposed method has been validated on steady-state incompressible laminar flow problems by comparing with commercial softwares, demonstrating superior accuracy and efficiency compared to traditional PINNs. FVNN offers a promising framework for solving complex fluid dynamics problems with reduced computational cost and higher precision, paving the way for broader applications in computational physics.

In future work, we aim to leverage FVNN to achieve highly accurate predictions of velocity and pressure fields in flow scenarios with dynamically evolving fluid-solid interfaces. This will lay the foundation for fully AI-driven fluid topology optimization, offering a novel technical framework and solutions to the challenges of fluid mechanics design.

References:

1. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys. 2019;378:686–707.
2. Hu Z, Shukla K, Karniadakis GE, Kawaguchi K. Tackling the curse of dimensionality with physics-informed neural networks. Neural Netw. 2024;176:106369.
3. Goswami S, Yin M, Yu Y, Karniadakis GE. A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials. Comput Methods Appl Mech Eng. 2022;391:114587.
4. Li D, Yan B, Gao T, Li G, Wang Y. PINN Model of Diffusion Coefficient Identification Problem in Fick's Laws. ACS Omega. 2024;9(3):3846–57.
5. Haghighat E, Raissi M, Moure A, Gomez H, Juanes R. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Comput Methods Appl Mech Eng. 2021;379:113741.
6. Hu H, Qi L, Chao X. Physics-informed Neural Networks (PINN) for computational solid mechanics: Numerical frameworks and applications. Thin-Walled Struct. 2024;112495.
7. Markidis S. On physics-informed neural networks for quantum computers. Front Appl Math Stat. 2022;8:1036711.
8. Rao C, Sun H, Liu Y. Physics-informed deep learning for incompressible laminar flows. Theor Appl Mech Lett. 2020;10(3):207–12.
9. Haghighat E, Raissi M, Moure A, Gomez H, Juanes R. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. Comput Methods Appl Mech Eng. 2021;379:113741.
10. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, et al. Automatic differentiation in pytorch. 2017;
11. Lu L, Meng X, Mao Z, Karniadakis GE. DeepXDE: A deep learning library for solving differential equations. SIAM Rev. 2021;63(1):208–28.
12. Yu B. The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. Commun Math Stat. 2018;6(1):1–12.
13. Kharazmi E, Zhang Z, Karniadakis GE. Variational physics-informed neural networks for solving partial differential equations. ArXiv Prepr ArXiv191200873. 2019;
14. Lyu L, Zhang Z, Chen M, Chen J. MIM: A deep mixed residual method for solving high-order partial differential equations. J Comput Phys. 2022;452:110930.
15. De Ryck T, Mishra S, Molinaro R. wPINNs: Weak physics informed neural networks for approximating entropy solutions of hyperbolic conservation laws. SIAM J Numer Anal. 2024;62(2):811–41.
16. Zang Y, Bao G, Ye X, Zhou H. Weak adversarial networks for high-dimensional partial differential equations. J Comput Phys. 2020;411:109409.
17. Byron FW, Fuller RW. Mathematics of classical and quantum physics. Courier Corporation; 2012.
18. Kingma DP. Adam: A method for stochastic optimization. ArXiv Prepr ArXiv14126980. 2014;
19. Liu DC, Nocedal J. On the limited memory BFGS method for large scale optimization. Math Program. 1989;45(1):503–28.
20. Gautschi W. Numerical analysis. Springer Science & Business Media; 2011.
21. Borrvall T, Petersson J. Topology optimization of fluids in Stokes flow. Int J Numer Methods Fluids. 2003;41(1):77–107.
22. Alexandersen J. A detailed introduction to density-based topology optimisation of fluid flow problems with implementation in MATLAB. Struct Multidiscip Optim. 2023;66(1):12.