



WEAK ADVERSARIAL NETWORKS FOR INTERFACE OPTIMAL DESIGN PROBLEMS UNDER PHYSICAL CONSTRAINTS

GANG BAO^{✉1,2}, DONG WANG^{✉*3,4,5} AND BOYI ZOU^{✉2}

¹Center for Interdisciplinary Applied Mathematics,
Zhejiang University, Hangzhou, Zhejiang 310058, China

²School of Mathematical Sciences, Zhejiang University,
Hangzhou, Zhejiang 310058, China

³School of Science and Engineering, The Chinese University of Hong Kong (Shenzhen),
Shenzhen, Guangdong 518172, China

⁴Shenzhen International Center for Industrial and Applied Mathematics,
Shenzhen Research Institute of Big Data, Guangdong 518172, China

⁵Shenzhen Loop Area Institute, Shenzhen, Guangdong 518048, China

(Communicated by Hao-Min Zhou)

ABSTRACT. Interface optimal design problems, fundamental to diverse scientific and engineering applications, present computational challenges due to their inherent complex geometries and physical constraints. This paper introduces a novel framework integrating Weak Adversarial Networks (WAN) and the Augmented Lagrangian Method (ALM) to efficiently solve physics constrained interface optimal design problems modeled in the phase field based representation. We reformulate these problems as minimax problems via ALM and represent the primal and dual variables using distinct deep neural networks. These networks are then trained through an adversarial process, enabling the method to naturally handle complex interface geometries and physical constraints. The proposed approach demonstrates superior constraint enforcement through learned Lagrange multipliers, significantly reduces sensitivity to constraint scaling and penalty hyperparameters compared to penalty-based methods, and an enhanced ability to escape local minimizers during optimization. Moreover, empirical study indicates that the $\tanh^3$ activation function exhibits enhanced robustness during practical hyperparameter tuning while also improving solution accuracy. Extensive numerical experiments, including Ginzburg–Landau energy minimization with mass constraint, optimal partition problems with L^2 norm preservation (demonstrated up to 5 dimensions), and fluid-solid topology optimization with partial differential equation constraints – all featuring physical constraints – validate the capability, robustness, and high-dimensional extensibility of the method.

2020 *Mathematics Subject Classification.* 68T07, 65N25, 49M41, 49Q10.

Key words and phrases. Interface problems, deep neural networks, augmented Lagrangian method, adversarial networks, optimal design.

*Corresponding author: Dong Wang.

1. Introduction. Interface problems play a crucial role in diverse scientific and engineering fields, as they provide a powerful framework for modeling complex physical phenomena. These problems arise in contexts such as heat transfer, fluid dynamics, elasticity, materials science, image processing, and industrial design. Specific examples include, for instance, in materials science, where they are used to simulate phenomena such as grain boundary evolution and bubble coalescence [7]; in image processing, where they facilitate image denoising and contour extraction by smoothing image boundaries while effectively removing noise and preserving critical structural information [11, 51, 57].

Various methods have been developed to represent and track interfacial dynamics, including front tracking methods [50], level set methods [40], phase field methods [48], and threshold dynamics methods [38]. For numerical solutions, well-established approaches include finite difference methods [3, 41], finite element methods [5, 19, 24], and discontinuous Galerkin methods [9, 37]. These methods primarily focus on representing the interface, approximating the energy functional or motion law within this representation, and employing optimization and time-marching techniques for specific problems. However, explicit representations typically struggle to track topology changes, while implicit methods such as phase field approaches often require a fine mesh to resolve sharp transitions. Moreover, level set methods can be difficult to extend to multiphase problems involving an arbitrary number of phases.

Interface optimal design problems under physical constraints represent a critical subclass of general interface problems, characterized by the optimization of free interfaces subject to complex multi-physics constraints. These problems are mathematically defined by free interfaces as decision variables, performance metrics (dependent on structure, shape, or topology) as objective functionals, and physical mechanisms - typically governed by coupled systems of partial differential equations with corresponding boundary conditions - as constraints. Such optimization challenges arise across diverse engineering applications, from precision structural design in intelligent manufacturing and 3D printing path optimization to advanced material design (particularly for micro/nano composites), flow battery electrode engineering, and thermal management component development. Despite their importance, conventional solution approaches [25, 45, 10] face persistent limitations: the curse of dimensionality from excessive design variables, prohibitive computational costs, and sensitivity to initialization conditions and parameter selection. Consequently, developing computationally efficient methods that maintain sufficient design freedom while ensuring solution robustness remains a fundamental unsolved challenge in engineering optimization.

In recent years, artificial intelligence (AI) for science has undergone rapid development, with physics-informed machine learning in scientific computing attracting particular attention. The universal approximation capability of neural networks for continuous functions, as proven in [18], has gained significant attention in the past decade. This has led to a growing trend of employing deep neural networks (DNNs) to represent a function and solve differential operator related problems. Numerous studies have demonstrated the effectiveness of deep learning approaches in solving partial differential equations, optimization problems with differential equation constraints, and inverse problems. Notable methods include physics-informed neural network (PINN) [42], deep Galerkin method (DGM) [46], deep Ritz method (DRM) [22], weak adversarial network (WAN) [56, 1], particle weak-form based

neural networks (ParticleWNN) [2], random feature method (RFM) [13], local extreme learning machines (locELM) [20], physics-driven orthogonal feature method (PD-OFM) [30] and many others. Regarding the utilization of deep neural networks to solve interface related problems, several existing works have been established [54, 8, 26, 33]. In contrast to traditional methods, deep learning approaches are typically mesh-free, making them less susceptible to the curse of dimensionality and more efficient for high-dimensional problems. For a comprehensive overview of using DNNs to solve differential operator related problems, we refer to [34, 31, 17] and references therein.

The framework for solving interface optimal design problems under physical constraints using DNNs involves constructing a problem-specific loss function and solving an optimization problem with complicated constraints. A common approach employs penalty methods to convert these constrained optimization problems into unconstrained ones. The results critically depend on the weighting of different terms in the loss function. Specifically, careful consideration must be given to the relative scaling of constraint values to ensure balanced minimization of all terms during training. Construction methods (e.g., [36, 49]) provide another way to handle specific boundary constraints, though their efficiency is limited to particular constraint types and computational domains.

Beyond traditional penalty methods, the Lagrange multiplier and augmented Lagrangian (ALM) approaches reformulate constrained optimization as minimax problems, finding successful applications in neural networks for natural language processing (NLP) [39], classification [43], and differential equation related problems. In these implementations, networks typically represent solutions while Lagrange multipliers are discretized on fixed grids - used to enforce conservation laws in kinetic PDEs [29], inverse design problems [35], and boundary conditions in PINNs [47]. Despite these advances, current ALM implementations remain fundamentally constrained by their reliance on fixed-grid multiplier discretization, limited scalability in high-dimensional spaces due to the curse of dimensionality, and inherent incompatibility with adaptive frameworks requiring dynamic collocation point sampling. The augmented Lagrangian deep learning (ALDL) [28] method constructed the Lagrange multipliers in the sense of adversarial networks to impose the essential boundary conditions of variational problems, avoiding the discretization of the Lagrange multipliers which is defined on the boundary.

Inspired by the framework of WAN [56, 1], we propose to integrate the idea of WAN and the augmented Lagrangian method, denoted as WAN-AL, to deal with interface optimal design problems with different types of physical constraints. The proposed method offers broad applicability, being capable of handling diverse constraint types, including scalar constraints, linear and nonlinear constraints, and PDE constraints. It employs an augmented Lagrangian formulation where both decision variables and Lagrange multipliers are represented by separate neural networks, trained through an adversarial process. The proposed method demonstrates reduced sensitivity to the weights of different terms in the loss function. By leveraging Lagrange multipliers, it achieves more effective constraint enforcement while simultaneously avoiding local optima that commonly may stuck in the conventional approach and it can be readily integrated with other techniques to enhance the performance of training results. Through numerical experiments, we validate its robustness and effectiveness across various interface optimal design problems with different physical constraints, including Ginzburg–Landau energy minimization problems

with mass constraints, Dirichlet optimal partition problems in different dimensions, and fluid-solid optimal design problems.

Specifically, our main contributions can be summarized as follows:

1. A novel framework is proposed, combining the idea of weak adversarial network with the augmented Lagrangian method (WAN-AL) to address interface optimal design problems under physical constraints, to effectively handle complex interface related optimization problems under various types of constraints and high-dimensional problems.
2. Compared with other deep learning approaches, the proposed method demonstrates faster convergence, reduced hyperparameter sensitivity, and effective avoidance of local optima.
3. A novel activation function, $\tanh^3$, is proposed within the current network architectural, demonstrating both high accuracy and enhanced robustness to practical hyperparameter tuning (e.g., neural network depth, width, learning rate).

The rest of this paper is organized as follows. In Section 2, we present the proposed method along with its network architecture. In Section 3, we use the mass preserving Ginzburg–Landau energy minimization problem as an example to demonstrate the performance of WAN-AL. In particular, Section 3.1 presents results comparing with other deep learning approaches to demonstrate the insensitivity of WAN-AL to hyperparameters, and Section 3.2 shows comparison results to shed light on the choice of activation functions. In Section 4, we investigate the high-dimensional performance of WAN-AL by considering Dirichlet partition problems up to five dimensions. The PDE constrained optimization problems are considered in Section 5, focusing on fluid-solid optimization problems. We draw a conclusion and discussion in Section 6.

2. Framework of WAN-AL. Interface optimal design problems with physical constraints can typically be formulated as constrained optimization problems. To introduce the proposed method, we consider the following general formulation:

$$\begin{aligned} \min_u L(u) \\ \text{s.t. } C(u) = 0, \end{aligned} \tag{1}$$

where $u = u(x)$ represents the decision variable to be optimized, L denotes the objective functional of u , and $C(u) = 0$ represents physical constraints that may include scalar constraints, boundary constraints, and PDE constraints. This problem finds diverse applications, including fluid mechanics, material science, optimal design, and topology optimization.

For problem (1), we transform it into a minimax problem with the Lagrange multiplier λ and a penalty term as follows:

$$\min_u \max_{\lambda} L_{\beta}(u, \lambda), \tag{2}$$

with

$$L_{\beta}(u, \lambda) = C_0 L(u) - \langle \lambda, C(u) \rangle + \frac{\beta}{2} \|C(u)\|^2, \tag{3}$$

in which C_0 and β are scalar coefficients. Here, C_0 is a fixed scaling hyperparameter used to avoid the objective functional value being too small in specific problems, while β is adaptively increased during optimization, as in the augmented Lagrangian method.

In WAN-AL, we propose to use two (or several) individual deep neural networks (similar to WAN and IWAN) to represent $u = u(x; \theta)$ and $\lambda = \lambda(x; \eta)$ with parameters θ and η , respectively. Problem (2) is then transformed into a minimax problem with respect to θ and η :

$$\min_{\theta} \max_{\eta} L_{\beta}(u(x; \theta), \lambda(x; \eta)), \quad (4)$$

with

$$L_{\beta}(u(x; \theta), \lambda(x; \eta)) = C_0 L(u(x; \theta)) - \langle \lambda(x; \eta), C(u(x; \theta)) \rangle + \frac{\beta}{2} \|C(u(x; \theta))\|^2. \quad (5)$$

The parameters θ and η are then alternatively updated by gradient descent and gradient ascent of $L_{\beta}(u(x; \theta), \lambda(x; \eta))$, with a gradually increasing value of β . The algorithmic framework of WAN-AL is summarized in Algorithm 1, and the neural network diagram of WAN-AL is illustrated in Figure 1.

Algorithm 1 WAN-AL: Weak adversarial network integrated with augmented Lagrangian method.

Inputs:

Two (or several) individual neural networks to represent the decision variable $u = u(x; \theta)$ and Lagrange multiplier $\lambda = \lambda(x; \eta)$.

C_0, β : weight term of the objective functional and constraint.

α : the amplifier of β in the augmented Lagrangian method.

N : number of iterations.

N_u, N_{λ} : number of primal and adversarial network parameter updates in each iteration.

$\tau_{\theta}, \tau_{\eta}$: learning rate of primal and adversarial network.

for $i = 1 : N$ **do**

 Update the Loss function $L_{\beta}(u(x; \theta), \lambda(x; \eta))$ in Equation (4).

for $j = 1 : N_u$ **do**

 Update $\theta \leftarrow \theta - \tau_{\theta} \nabla_{\theta} L_{\beta}$.

end for

for $k = 1 : N_{\lambda}$ **do**

 Update $\eta \leftarrow \eta + \tau_{\eta} \nabla_{\eta} L_{\beta}$.

end for

 Update $\beta \leftarrow \alpha * \beta$.

end for

Outputs: Neural networks $u(x; \theta)$ and $\lambda(x; \eta)$ with arbitrary input x and trained parameters θ and η .

In terms of the network architecture, we represent the decision variables using residual neural networks (ResNet). The Lagrange multiplier is represented by different networks depending on the type of physical constraints. For example:

1. For scalar or vector constraints, $\lambda(x; \eta)$ can be simply set as $\lambda(0; \eta)$ with a constant input 0 and a shallow neural network,
2. For constraints defined over the entire computational domain (e.g., PDE constraints), $\lambda(x; \eta)$ is set to be a function in the domain represented by a ResNet.

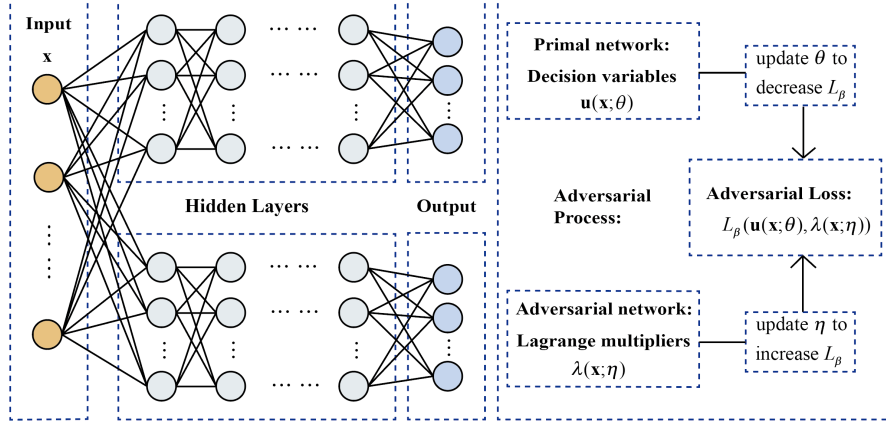
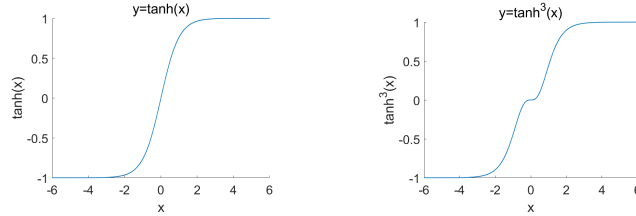


FIGURE 1. Network structure, see Section 2.

To be specific, we use a ResNet [27] with N_d hidden layers and a width of N_w , expressed as follows:

$$\begin{aligned} \mathbf{u}(\mathbf{x}_0; \theta) &= \mathbf{W}_{out} \mathbf{x}_{N_d+1} + \mathbf{b}_{out}, \\ \mathbf{x}_{k+1} &= f_k(\mathbf{x}_k) = \Phi_k(\mathbf{W}_k \mathbf{x}_k + \mathbf{b}_k) + \mathbf{x}_k, \quad k = N_d, \dots, 1, \\ \mathbf{x}_1 &= \mathbf{W}_{in} \mathbf{x}_0 + \mathbf{b}_{in}. \end{aligned} \quad (6)$$

Here, $\mathbf{x}_0 \in \Omega$ is the input of dimension d , $\{\mathbf{W}_{in} \in \mathbb{R}^{N_w \times d}, \mathbf{W}_k \in \mathbb{R}^{N_w \times N_w}, \mathbf{W}_{out} \in \mathbb{R}^{n \times N_w}\}$ are the weight matrices of the input layer, k -th hidden layer, and output layer, $\{\mathbf{b}_{in} \in \mathbb{R}^{N_w}, \mathbf{b}_k \in \mathbb{R}^{N_w}, \mathbf{b}_{out} \in \mathbb{R}^n\}$ are the corresponding bias vectors, and Φ_k represents the activation function. The parameters N_d and N_w define the size of the weight matrices and bias vectors in the neural network, which ultimately determine the network's performance and capabilities based on many empirical observations and the universal approximation theorem. The ResNet [27] effectively mitigates gradient vanishing and explosion through residual connections (represented by the $+\mathbf{x}_k$ term in the second equation of (6)). For the activation function Φ_k , we propose to use a novel activation function $\tanh^3$ instead of common activation functions [44] (e.g., sigmoid, tanh, softmax, ReLU, and ReLU³). As shown in Figure 2, this new activation function introduces a plateau near zero compared to tanh, and numerical experiments demonstrate that it typically achieves higher accuracy and is relatively robust to hyperparameter. A detailed numerical comparison is provided in Section 3.

FIGURE 2. Comparison of two activation functions, $\tanh(x)$ and $\tanh^3(x)$, in the range of $[-6, 6]$. See Section 3.2.

After constructing the saddle point problem (4), the loss function is optimized iteratively by the standard optimizer Adam [32] with the gradient descent and ascent method. When computing different terms of the loss function, we employ several standard techniques, including the Monte Carlo (MC) method for integrals and automatic differentiation for derivatives. To avoid getting stuck in undesirable local minima, we incorporate training point shuffling during the training process, as demonstrated in [22]. To simplify the optimization process, we design specialized networks that automatically satisfy boundary or positivity constraints. For cases where these constraints are not straightforward to enforce, they can be implemented as additional adversarial networks within the framework of WAN-AL. The numerical results presented in Sections 3-5 demonstrate that WAN-AL is a versatile algorithm capable of being combined with various DNN techniques to enhance accuracy and accelerate the optimization process. Specific techniques will be discussed when they are used in different examples.

The proposed method can be used to deal with general constrained optimization problems. In this work, we focus specifically on problems involving differential operators in either the objective functional or constraints, particularly those related to free interface optimization with certain physical constraints, where the interface is represented in the phase field based framework. To demonstrate the performance of WAN-AL, we consider the following problems:

1. Ginzburg–Landau energy minimization with mass preservation. This example showcases WAN-AL’s robustness in hyperparameter selection and experimentally validates the choice of activation function (See Section 3).
2. Dirichlet optimal partition problems in dimensions 2 to 5 with nonlinear vector constraints. These demonstrate WAN-AL’s capability in handling high-dimensional, non-convex optimization landscapes with multiple local minima (See Section 4).
3. Fluid-solid optimization under PDE constraints. This demonstrates WAN-AL’s effectiveness in handling partial differential equation (PDE)-constrained problems (See Section 5).

3. Minimization of the Ginzburg–Landau energy with mass conservation.

In this section, we empirically investigate hyperparameter sensitivity, activation function selection, and comparisons with penalty-based methods and adversarial networks using only Lagrange multipliers, illustrating them through the minimization of the Ginzburg–Landau energy with mass conservation as an example.

Consider the minimization of the Ginzburg–Landau energy with mass conservation and homogeneous Dirichlet boundary conditions in $\Omega = [0, 1] \times [0, 1]$ as follows:

$$\begin{aligned} \min_u \quad & \int_{\Omega} \frac{\epsilon}{2} |\nabla u|^2 + \frac{1}{\epsilon} (u^2 - 1)^2 \, d\mathbf{x} \\ \text{s.t.} \quad & \int_{\Omega} u \, d\mathbf{x} = V, \quad \text{and} \quad u = -1 \quad \text{on} \quad \partial\Omega, \end{aligned} \tag{7}$$

where $u : (x, y) \rightarrow \mathbb{R}$ is the phase order parameter and V is a constant used to enforce the integral of u , which is equivalent to the total mass in a two-phase system if u represents rescaled density.

The solution u is modeled by a ResNet $u(x, \theta)$, while λ is parameterized by a shallow network (one hidden layer, few neurons). At $\mathbf{x} = 0$, $\lambda(\mathbf{x}; \eta)$ becomes a scalar $\lambda(0; \eta)$. Here, θ and η are parameters in the distinct networks. As for the boundary

conditions of the primal network $u(\mathbf{x}; \theta)$ in this case, we enforce them explicitly at the output layer:

$$u(\mathbf{x}_0; \theta) = (W_{out} f_{N_d}(\mathbf{x}_{N_d}) + b_{out}) \times (x_0^{(1)}) \times (1 - x_0^{(1)}) \times (x_0^{(2)}) \times (1 - x_0^{(2)}) - 1, \quad (8)$$

in which $x_0^{(1)}$ and $x_0^{(2)}$ correspond to the two coordinates of the input $\mathbf{x}_0$, and $u(\mathbf{x}_0; \theta) = -1$ for $\mathbf{x}_0 \in \partial\Omega$.

Remark 3.1. Here, we use a shallow neural network with constant input to represent the Lagrange multiplier since this approach simplifies the learning rate tuning for the adversarial term. Using a single parameter to represent the Lagrange multiplier would make precise learning rate tuning difficult during gradient ascent, as a small learning rate leads to poor acceleration while a large one causes convergence issues. In contrast, employing a neural network allows multiple parameters to collectively adjust the Lagrange multiplier's updates, enabling stable and effective optimization with a relatively small learning rate. This strategy achieves both accelerated training and reliable convergence.

Remark 3.2. This approach of dealing with boundary conditions is suitable for relatively regular geometries, such as the unit square we considered in this example. For more general boundary conditions, we can treat them as constraints and use the framework of WAN-AL to convert them into another adversarial network, as demonstrated in Section 5.

3.1. Sensitivity to hyperparameters and comparison to other deep learning approaches. For problem (7), we obtain the following minimax problem:

$$\min_{\theta} \max_{\eta} \mathcal{L}_{\beta}(\theta, \eta), \quad (9)$$

with

$$\begin{aligned} \mathcal{L}_{\beta}(\theta, \eta) = & C_0 \int_{\Omega} \left[\frac{\epsilon}{2} |\nabla u(\mathbf{x}; \theta)|^2 + \frac{1}{\epsilon} (u(\mathbf{x}; \theta)^2 - 1)^2 \right] d\mathbf{x} \\ & - \lambda(0; \eta) \left(\int_{\Omega} u(\mathbf{x}; \theta) d\mathbf{x} - V \right) + \frac{\beta}{2} \left(\int_{\Omega} u(\mathbf{x}; \theta) d\mathbf{x} - V \right)^2. \end{aligned} \quad (10)$$

Since the problem under consideration is formulated in a variational framework, we compare the proposed method with the deep Ritz method (DRM) [22] incorporating constraints to evaluate the contribution of each term. Specifically, we consider the following variants:

1. DRM combined with penalty method (DRM-P);
2. DRM combined with adaptive penalty method (DRM-AP);
3. DRM combined with Lagrange multipliers (DRM-L).

For problem (7), the loss function of DRM is defined as follows (with identical boundary condition handling to the proposed method):

$$\mathcal{L}_{DRM}(\theta) = C_0 \int_{\Omega} \left[\frac{\epsilon}{2} |\nabla u(\mathbf{x}; \theta)|^2 + \frac{1}{\epsilon} (u(\mathbf{x}; \theta)^2 - 1)^2 \right] d\mathbf{x} + \frac{\beta}{2} \left(\int_{\Omega} u(\mathbf{x}; \theta) d\mathbf{x} - V \right)^2. \quad (11)$$

Here, DRM-P denotes the case where the penalty coefficient is a fixed value, whereas DRM-AP refers to the scenario where β increases at the same rate as in the proposed method. The loss function for DRM combined with Lagrange multipliers

is given as follows:

$$\begin{aligned} \mathcal{L}_{DRM-L}(\theta, \eta) = & C_0 \int_{\Omega} \left[\frac{\epsilon}{2} |\nabla u(\mathbf{x}; \theta)|^2 + \frac{1}{\epsilon} (u(\mathbf{x}; \theta)^2 - 1)^2 \right] d\mathbf{x} \\ & - \lambda(0; \eta) \left(\int_{\Omega} u(\mathbf{x}; \theta) d\mathbf{x} - V \right). \end{aligned} \quad (12)$$

From the classical analysis of problem (7), the zero level set of the solution should be a circle with the enclosed area conserved. It is worth noting that the solution to this problem is not unique with shifting, and therefore we compare the results of different methods by examining how well they satisfy the constraints. In this example, we set $\epsilon = 0.05$, $V = -0.5$, $C_0 = 400$, and $\alpha = 1.0003$. For the primal network $u(\mathbf{x}; \theta)$, we use $N_d = 4$ hidden layers with width $N_w = 50$. The adversarial network $\lambda(0; \eta)$ employs a shallow neural network with 1 hidden layer and width 10, since the Lagrange multiplier λ is a scalar in this case. The training step is set to $N = 5000$ with inner iteration steps $N_u = N_\lambda = 2$. In each iteration, $N_r = 40,000$ points are used for training. Unless otherwise specified, uniformly distributed random sampling is employed. In this example and all subsequent examples, the initial learning rate is $\tau_\theta = \tau_\eta = 0.016$ and halved at certain training milestones. The activation function used here is $\tanh^3(x)$. Both DRM-P and DRM-AP share the same network architecture as the primal network described above.

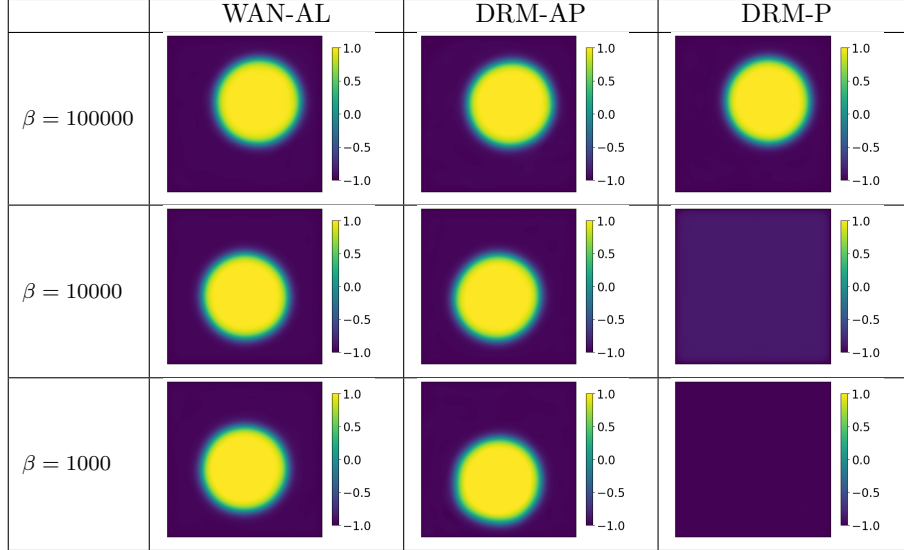


FIGURE 3. Comparison of the training results among WAN-AL, DRM-AP, and DRM-P under different β . See Section 3.1.

In Figure 3, we display the results from WAN-AL, DRM-P, and DRM-AP for different choices of $\beta = 100000, 10000, 1000$, together with the convergence of the constraint error, defined as $\frac{V - \int_{\Omega} u d\mathbf{x}}{V}$, during the training process in Figure 4. WAN-AL employs identical hyperparameters across different β values in these results. In contrast, both DRM-P and DRM-AP require individual hyperparameter tuning for each case to obtain satisfactory results (which may still completely fail due to the poor selection of β). This implies that the proposed WAN-AL is relatively

insensitive to hyperparameter settings. As shown in Figure 3, under three different initial penalty terms, the proposed WAN-AL consistently outperforms DRM-P and DRM-AP.

For the first case with $\beta = 100000$, the profiles of WAN-AL, DRM-AP, and DRM-P are close to circles, but WAN-AL and DRM-AP show superior constraint error performance. For $\beta = 10000$ and $\beta = 1000$, DRM-P fails, and DRM-AP yields inaccurate (non-circular) results due to unbalanced loss terms. In contrast, WAN-AL maintains robust performance. This indicates that relying solely on the adaptive penalty method is still sensitive to hyperparameter selection, and comparable results are achieved only when $\beta = 100000$.

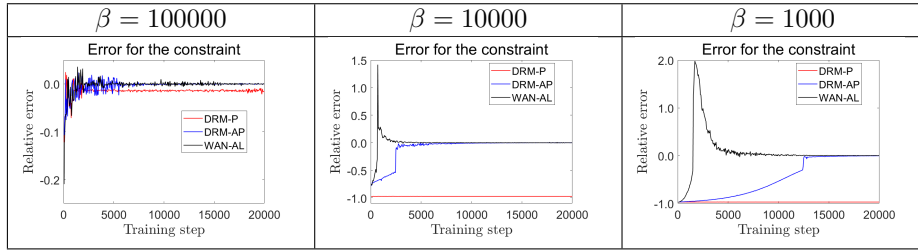


FIGURE 4. The relative error of the constraint for three schemes with corresponding β . The first, second, and third pictures correspond to $\beta = 100000, 10000, 1000$, respectively. The relative error of the constraint, defined as $\frac{V - \int_{\Omega} u \, d\mathbf{x}}{V}$, is recorded every 50 step. See Section 3.1.

In all cases, WAN-AL can achieve a relative error of the constraint less than 0.005. As shown in Figure 4, compared to DRM-AP, WAN-AL can significantly reduce the training steps due to the adversarial term in (9). To be more specific, due to the Dirichlet boundary condition and the random initialization of network parameters, the output of the primal network in the initial steps closely approximates -1 . Consequently, the initial integral of u is also close to -1 , falling below the constraint of -0.5 . With the inclusion of the adversarial term λ , during the early stages of training, the mass of u increases at a faster rate compared to DRM-AP. As the adversarial term λ reaches a sufficiently large value, it becomes dominant, causing the minimization step to shift the mass in the positive direction. This corresponds to the initial steps shown in Figure 4. Once the error of the constraint becomes positive, λ is reduced. Subsequently, when the third term becomes dominant again, the constraint error decreases rapidly. This corresponds to the stage of rapid descent in Figure 4, which is more evident in the last two pictures. To conclude, the inclusion of the adversarial term λ allows for adaptive adjustment of its magnitude, enabling the constraint to be rapidly satisfied from both sides. This contrasts with DRM-AP, where the mass can only increase rapidly as the penalty term reaches a threshold value. This further demonstrates that WAN-AL is relatively insensitive to hyperparameter settings, as the adversarial term exhibits notable adaptive weighting and acceleration effects. Moreover, from the training perspective, in WAN-AL, half of the training steps are devoted to optimizing the adversarial network, which is usually constructed with a smaller network. This typically results in faster training compared to other methods.

We also consider the Lagrange multiplier formula without penalty terms. We observe that the Lagrange multiplier only has a corresponding ascent direction during training, and its effectiveness in imposing the constraint is affected by the step size (learning rate). Therefore, relying solely on the Lagrange multiplier may result in difficulties in convergence.

However, with the augmented Lagrangian method, we observe that the adversarial term adjusts its value scale based on the current level of constraint satisfaction to accelerate the process of satisfying the constraint, while the penalty term gradually increases to guarantee convergence. As a result, the adversarial process is effectively similar to assigning different weights to the training data, enabling WAN-AL to impose the constraints more effectively.

Remark 3.3. In practice, setting β to a moderately large value (e.g., 100 or 1000) and scaling it by a factor α (e.g., 1.0003) enables the adversarial term to adjust automatically. Furthermore, multiplying the objective functional by a coefficient C_0 (e.g., 10 or 100) prevents it from becoming too small, which helps ensure stable training. Code is available at <https://github.com/BoyiZou>.

Remark 3.4. Notably, WAN-AL outperforms DRM in training efficiency: Lagrange multipliers accelerate constraint satisfaction, and the use of a smaller adversarial network also reduces training cost.

3.2. Choice of activation functions. To illustrate the observations and reasons for the utilization of $\tanh^3$ (see the profile in Figure 2) in our examples, we present two numerical simulations to validate the advantages of combining ResNet with the activation function $\tanh^3$ in the proposed WAN-AL.

For the above problem (9), we consider:

1. Replacing the activation function directly with other activation functions in the model that has been well-tuned for $\tanh^3$;
2. Replacing the activation function with $\tanh^3$ in the model that has been well-tuned for other common activation functions.

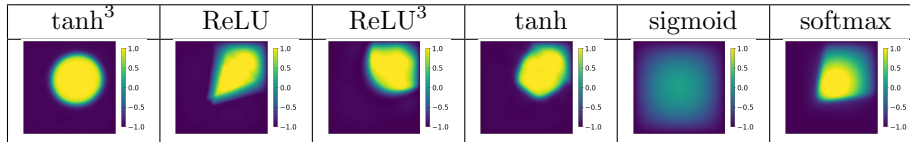


FIGURE 5. Replacing $\tanh^3$ with other activation functions while keeping the other network hyperparameters unchanged. The training results for the same network hyperparameters with activation functions $\tanh^3$, ReLU, ReLU^3 , tanh, sigmoid, and softmax. See Section 3.2.

As shown in Figure 5, when replacing the activation function $\tanh^3$ in the well-tuned neural network with other activation functions, the training results are often poor or even incorrect. However, as shown in Figure 6, when initially using a well-tuned neural network with a different activation function and subsequently changing it to $\tanh^3$, we observe that similar or even better results can be achieved.

Based on the observations from these two examples, we believe that combining ResNet with $\tanh^3$ exhibits favorable properties and can reduce hyperparameter tuning time for WAN-AL. This choice is also used and verified in the subsequent problems.

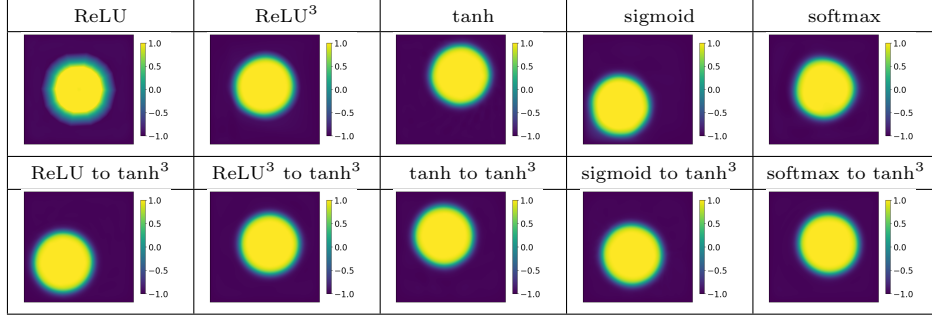


FIGURE 6. First row: training results obtained by well-tuned networks with the activation functions ReLU, ReLU^3 , tanh, sigmoid, and softmax. Second row: training results after replacing the activation function with $\tanh^3$, while keeping the other hyperparameter settings unchanged throughout the process. See Section 3.2.

4. Dirichlet optimal partition problems. In this section, we consider the optimal partition problem, which minimizes the Dirichlet energy with L^2 norm preserving under different dimensions and different boundary conditions [6, 21], as follows:

$$\begin{aligned} \min_{\mathbf{u}} \int_{\Omega} \left[\frac{\epsilon}{2} \sum_i |\nabla u_i|^2 + \frac{1}{\epsilon} f(\mathbf{u}) \right] d\mathbf{x} \\ \text{s.t. } \int_{\Omega} u_i^2 d\mathbf{x} = 1, i = 1, 2, \dots, n, \text{ and } \mathbf{u} \text{ satisfies some boundary conditions,} \end{aligned} \quad (13)$$

where $\Omega = [0, 1]^d$, d denotes the dimension, n represents the number of phases, $\mathbf{u} = [u_1, u_2, \dots, u_n]'$ denotes the phase variables, with each u_i mapping $\mathbf{x}$ to $\mathbb{R}_+$, ϵ is the model parameter related to the thickness of the transition layer, and f takes the form $f(\mathbf{u}) = \sum_{i,j=1, i \neq j}^n u_i^2 u_j^2$.

Following the augmented Lagrangian method, one can obtain the minimax problem:

$$\begin{aligned} \min_{\mathbf{u}} \max_{\boldsymbol{\lambda}} C_0 \int_{\Omega} \left[\frac{\epsilon}{2} \sum_i |\nabla u_i|^2 + \frac{1}{\epsilon} \sum_{i=1, j=1, i \neq j}^n u_i^2 u_j^2 \right] d\mathbf{x} \\ - \boldsymbol{\lambda} \cdot \left(\int_{\Omega} \mathbf{u}^2 d\mathbf{x} - \mathbf{1} \right) + \frac{\beta}{2} \left\| \int_{\Omega} \mathbf{u}^2 d\mathbf{x} - \mathbf{1} \right\|_2^2, \end{aligned} \quad (14)$$

where $\boldsymbol{\lambda}$ denotes n Lagrange multipliers for n constraints in (13), and $\mathbf{1}$ is the vector in $\mathbb{R}^n$ whose component are all ones.

We then construct the primal and adversarial networks to represent $\mathbf{u} = \mathbf{u}(\mathbf{x}; \theta)$ and $\boldsymbol{\lambda} = \boldsymbol{\lambda}(\mathbf{x}; \eta)$, respectively. Since the L^2 norm preserving constraint is independent of $\mathbf{x}$, we construct the adversarial network as $\boldsymbol{\lambda}(0; \eta) = [\lambda_1(0; \eta), \lambda_2(0; \eta), \dots, \lambda_n(0; \eta)]'$ using a shallow neural network. Consequently, problem (14) is transformed into the following form:

$$\min_{\theta} \max_{\eta} L_{\beta}(\theta, \eta), \quad (15)$$

with

$$L_\beta(\theta, \eta) = C_0 \int_{\Omega} \left[\frac{\epsilon}{2} \sum_{i=1}^n |\nabla u_i(\mathbf{x}; \theta)|^2 + \frac{1}{\epsilon} \sum_{i=1, j=1, i \neq j}^n u_i^2(\mathbf{x}; \theta) u_j^2(\mathbf{x}; \theta) \right] d\mathbf{x} \\ - \left[\boldsymbol{\lambda}(0; \eta) \cdot \left(\int_{\Omega} \mathbf{u}^2(\mathbf{x}; \theta) d\mathbf{x} - \mathbf{1} \right) \right] + \frac{\beta}{2} \left\| \int_{\Omega} \mathbf{u}^2(\mathbf{x}; \theta) d\mathbf{x} - \mathbf{1} \right\|_2^2. \quad (16)$$

In the following, we present the results for the Dirichlet and periodic boundary conditions separately. The hyperparameter settings are detailed for each boundary condition type. The results in both 2- and 3-dimensional spaces are consistent with the results presented in [52, 4, 53], indicating the effectiveness and robustness of the proposed method. In addition, we present the 4- and 5-dimensional partition results to demonstrate the computational efficiency of WAN-AL for high-dimensional problems.

4.1. Dirichlet optimal partition with Dirichlet boundary conditions. In this section, we consider the Dirichlet partition problem with Dirichlet boundary conditions in the unit square $\Omega = [0, 1] \times [0, 1]$ with the L^2 norm preserved. The Dirichlet boundary conditions are imposed in strong form at the output layer, similar to the first numerical example:

$$\mathbf{u}(\mathbf{x}_0; \theta) = \text{ReLU}(\mathbf{W}_{out} f_{N_d}(\mathbf{x}_{N_d}) + \mathbf{b}_{out}) \times \prod_{i=1}^d \left[(x_0^{(i)}) \times (1 - x_0^{(i)}) \right], \quad (17)$$

in which $x_0^{(i)}$ corresponds to the i -th entry of the input $\mathbf{x}_0 \in \Omega$, and ReLU here is used to enforce the non-negativity of $\mathbf{u}$.

The hyperparameters for the primal networks are set as follows: $\epsilon = 0.05$, $C_0 = 100$, $\beta = 10000$, $\alpha = 1.0003$, $N_d = 8$, and $N_w = 120$. A shallow neural network $\boldsymbol{\lambda}(0; \eta)$ with 1 hidden layer and width 10 is used to represent the adversarial network. The training procedure consists of $N = 20000$ steps with inner iteration steps $N_u = N_\lambda = 1$, using $N_r = 10000$ sample points per iteration. The testing results are obtained by evaluating the trained neural network on a uniform 1000×1000 grid.

In Figure 7, we present the optimal n -partitions with Dirichlet boundary conditions. These results are consistent with those reported in [52, 4, 14], which were obtained using traditional numerical methods. Furthermore, Table 1 compares the final energy values, defined as $\frac{1}{2} \int_{\Omega} \sum_i |\nabla u_i|^2 d\mathbf{x}$, between the proposed method and those from [14]. This comparison demonstrates the consistency and capability of the proposed method in handling such complicated partition problems. The first and third rows of Figure 7 represent the network output after being well trained. The second and fourth rows represent the plot of n partitions based on the following projection:

$$\phi(\mathbf{x}) = m, \quad m = \arg \max_{1 \leq k \leq n} u_k(\mathbf{x}; \theta), \quad \text{for } \mathbf{x} \in \Omega. \quad (18)$$

	$n = 2$	$n = 3$	$n = 4$	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$
WAN-AL	42.96	84.08	135.2	216.6	304.0	406.3	520.8	637.9
Ref. [14]	45.94	90.00	140.9	211.1	298.7	391.4	529.1	616.2

TABLE 1. Comparison of final energy between WAN-AL and the traditional algorithm [14]. See Section 4.1.

In the traditional method for this optimization problem, one has to carefully handle the boundary condition by employing concave relaxation or Schrödinger operator relaxation and design the optimization scheme to obtain reasonable and regular results. In WAN-AL, the aforementioned hyperparameters can effectively train the network for all numbers (i.e., $n = 2-9$) of partitions, which correspond to different dimensions of the network output. This implies the robustness of WAN-AL to hyperparameter settings.

It is important to highlight that under identical hyperparameter settings except for the adversarial term, if the DRM is directly used to solve the aforementioned Dirichlet partition problems, improper selection of β may easily lead to entry vanishing. For example, given $n = 9$, the solution incorrectly yield 7-partition, as shown in Figure 8. In contrast, the utilization of Lagrange multipliers in WAN-AL ensures that when certain entries vanish during a training step, their corresponding Lagrange multipliers become dominant in the loss function. This will help vanishing entries to become nonzero in subsequent training steps. This observation further demonstrates WAN-AL's insensitivity to hyperparameter selection.

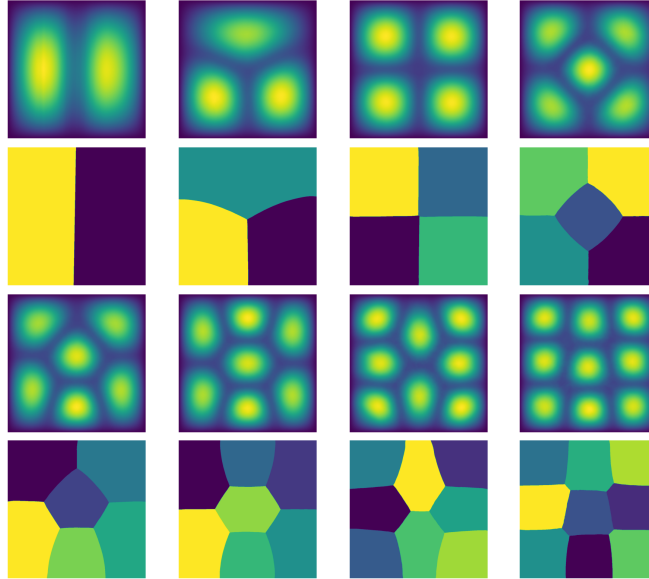


FIGURE 7. Results for the Dirichlet partition problem with Dirichlet boundary conditions in the unit square $[0, 1] \times [0, 1]$. First and third rows: the training solution $\sum_{i=1}^n (u_i(\mathbf{x}; \theta))$ for $n = 2-9$. Second and fourth rows: the corresponding partitions after the projection (18) for 2-9 partitions. These results are based on the evaluation of the trained neural network on 1000×1000 grid points. See Section 4.1.

4.2. Dirichlet optimal partition with periodic boundary conditions. In this section, we consider the Dirichlet partition problem in 2-, 3-, 4-, and 5-dimensional

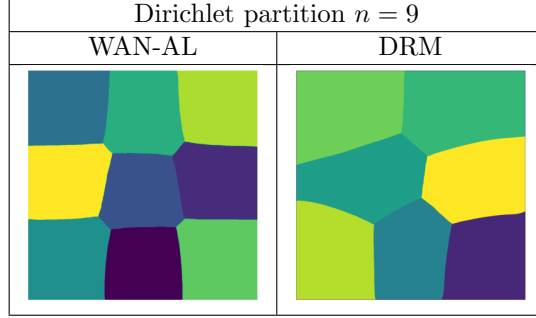


FIGURE 8. Comparison results of WAN-AL and DRM for the Dirichlet partition problem with Dirichlet boundary conditions in the unit square $[0, 1] \times [0, 1]$, with $n = 9$. Under identical hyperparameter settings except for the adversarial term, DRM exhibits entry vanishing ($n = 9 \rightarrow 7$), whereas WAN-AL obtains consistent results. See Section 4.1.

flat tori. The periodic boundary conditions on $\Omega = [0, 1]^d$ are imposed by transforming the input layer as follows [36]:

$$\begin{aligned}
 f_{in}(\mathbf{x}_0) &= f((x_0^{(1)}, x_0^{(2)}, \dots, x_0^{(d)})') \\
 &= (\cos(2\pi x_0^{(1)}), \sin(2\pi x_0^{(1)}), \cos(2\pi x_0^{(2)}), \sin(2\pi x_0^{(2)}), \dots, \cos(2\pi x_0^{(d)}), \sin(2\pi x_0^{(d)}))',
 \end{aligned}
 \tag{19}$$

in which $x_0^{(i)}$ corresponds to the i -th entry of the input $\mathbf{x}_0 \in \Omega$.

The hyperparameters for the primal networks are set as follows: $\epsilon = 0.04$, $C_0 = 2500$, $\beta = 100000$, $\alpha = 1.0003$, $N_d = 3$, and the width is increasing from $N_w = 50$ to $N_w = 200$ as the dimension or the number of phases increases. A shallow neural network $\lambda(0; \eta)$ with 1 hidden layer and width 10 is used to represent the adversarial network. The training is performed for $N = 5000$ steps with inner iteration steps $N_u = N_\lambda = 2$. $N_r = 10000, 40000, 60000, 80000$ points are used for 2-, 3-, 4-, and 5-dimensional cases in each iteration, respectively. The sampling points used are fixed as the Hammersley sequence in the corresponding dimension, as in [55].

Two-dimensional flat torus: In Figure 9, we display the results of n -partition in a 2-dimensional flat torus trained by the proposed WAN-AL for $n = 3-9, 11, 12, 15, 16, 18, 20, 23$, and 24, following the same projection described in (18). The testing results are obtained by evaluating the trained neural network on a uniform 1000×1000 grid. To our knowledge, even in the 2-dimensional case, there are not too many efficient numerical methods that can compute the n -partition problems for a large n (e.g., $n = 24$). Existing methods [15, 16] usually require thousands of steps to converge for the problem with a large number of partitions, even in the 2-dimensional case. In all of the previous studies, hexagons are ubiquitous for large values of n . Figure 9 shows that WAN-AL obtains consistent results for a large range of n . Since the domain has an aspect ratio equal to one, regular hexagons cannot be used to tile the domain, so the hexagons are slightly distorted.

Three-dimensional flat torus: For the 3-dimensional flat torus, we consider the cases for $n = 3, 4$, and 8. The testing results are obtained by evaluating the trained neural network on a uniform 100^3 grid. For $n = 3$, we obtain a novel partition consisting of three identical structural units, as shown in Figure 10. To our

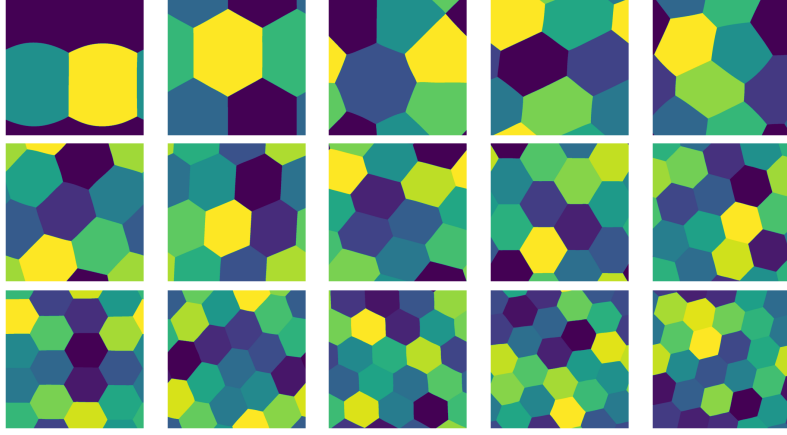


FIGURE 9. The projected partitions for the Dirichlet partition problem with periodic boundary conditions in the 2-dimensional flat torus for $n = 3-9, 11, 12, 15, 16, 18, 20, 23$, and 24 . See Section 4.2.

knowledge, this specific geometric configuration has not been previously reported in the literature. We tried to compute the $n = 3$ case using the conventional algorithm [52] from a random initial guess but found it difficult to obtain the similar structure. However, when using the output of WAN-AL as the initial value for the traditional algorithm, the computation stabilizes and converges to this structure. This observation indicates that network representations enable the discovery of non-trivial local minima under random initialization, whereas classical optimization methods may struggle.

For $n = 4$, we obtain a partition of the cube consisting of four identical rhombic dodecahedron structures, as displayed in Figure 11. For $n = 8$, we obtain a partition of the cube that is similar to the Weaire–Phelan structure. Figure 12 shows different views of the first and second type Weaire–Phelan structures, together with different views of a periodic extension of the partition.

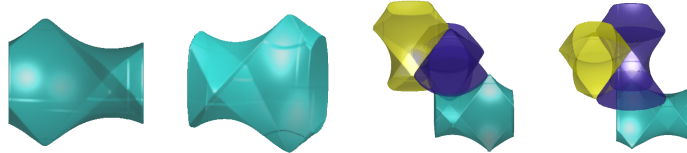


FIGURE 10. Partition results for the Dirichlet partition problem with periodic boundary conditions in the 3-dimensional flat torus with $n = 3$. See Section 4.2.

Four-dimensional flat torus: We then apply the proposed WAN-AL to 4-dimensional partition problems. In Figures 13 and 14, the four columns correspond to slices perpendicular to the x_1 -, x_2 -, x_3 -, and x_4 -axes, respectively. The four rows correspond to the slices at $x_j = 0, 0.25, 0.5, 0.75$, respectively. These partitions are based on testing the trained neural network on a uniform 100^3 grid at corresponding 3-dimensional slices.

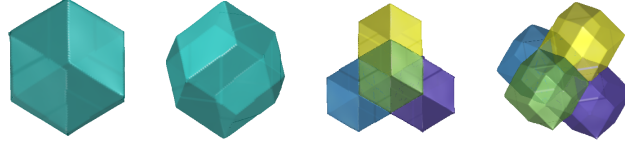


FIGURE 11. Partition results for the Dirichlet partition problem with periodic boundary conditions in the 3-dimensional flat torus with $n = 4$. See Section 4.2.

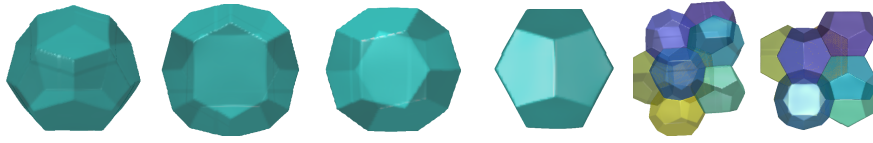


FIGURE 12. Partition results for the Dirichlet partition problem with periodic boundary conditions in the 3-dimensional flat torus with $n = 8$. Left two: different views of the first type Weaire-Phelan structure. Middle two: different views of the second type Weaire-Phelan structure. Right two: different views of the periodic extension of the structure. See Section 4.2.

For $n = 2$, we obtain a constant extension of the structure along the fourth direction, similar to the Schwarz P-surface, as displayed in Figure 13. Traditional methods usually result in slab partitions for 2 partitions in 3-dimensional flat torus with a random initialization of the partition. However, based on the evidence from the numerical study in [53], the Schwarz P-surface also appears to be a local minimizer for the 3-dimensional case. In this 4-dimensional flat torus, the random initialization represented by the neural network can find different local minimizers or stationary solutions more easily.

For $n = 4$, we obtain a partition structure similar to a constant extension along the fourth direction of the rhombic dodecahedron. For $n = 8$, we obtain a partition that is symmetric along all four directions, as shown in Figure 14. The obtained partition is similar to the structure known as the 24-cell honeycomb, which is a regular space-filling tessellation of 4-dimensional Euclidean space by regular 24-cells.

Five-dimensional flat torus: To further demonstrate the advantages of the proposed method in solving high-dimensional problems, we extend our investigation to 5-dimensional partition problems. We compute both $n = 4$ and $n = 8$ scenarios. For $n = 4$, the 5-dimensional training results are a constant extension of the corresponding 3-dimensional case in Figure 11 along another two directions. For $n = 8$, the 5-dimensional training results are a constant extension of the corresponding 4-dimensional case in Figure 14 along the other direction. For these high-dimensional cases, traditional numerical methods struggle to compute due to the curse of dimensionality. In contrast, the proposed method can easily obtain partition results up to 5 dimensions, with only a slight increase in the number of points required during the process. For example, increasing from 10,000 points in 2 dimensions to 80,000 points in 5 dimensions yields promising results, whereas 100^5 would become prohibitively expensive for traditional algorithms.

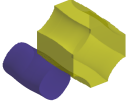
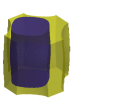
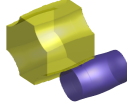
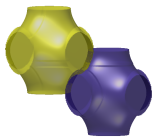
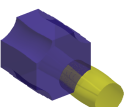
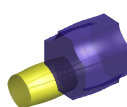
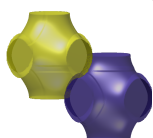
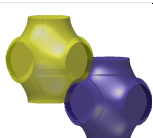
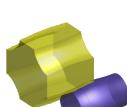
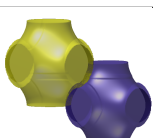
	$j = 1$	$j = 2$	$j = 3$	$j = 4$
$x_j = 0$				
$x_j = 0.25$				
$x_j = 0.5$				
$x_j = 0.75$				

FIGURE 13. A $n = 2$ Dirichlet partition problem with periodic boundary conditions in the 4-dimensional flat torus. The four columns correspond to the slides perpendicular to the x_1 -, x_2 -, x_3 -, and x_4 -axis respectively. The four rows correspond to the slices at $x_j = 0, 0.25, 0.5, 0.75$, respectively. These partitions are based on testing the trained neural network on a uniform 100^3 grid. See Section 4.2.

5. Fluid-solid optimal design problems. Topology optimization is a technique that aims to find the optimal layout or distribution of materials to maximize the objective within a specified design space while satisfying a set of constraints. Typically, these constraints are usually described by coupled partial differential equations. Such problems can be formulated as interface optimal design problems with PDE constraints. In the following, we consider the application of WAN-AL to solve fluid-solid optimization problems.

To be specific, following [23, 12], we consider an optimization model to find the optimal fluid-solid channel inside a designated region with given inflow and outflow boundary conditions, where the interface is represented by the phase field order parameter ϕ , and the objective is the total dissipation of the system in the following model:

$$\begin{aligned}
& \min_{\mathbf{u}, \phi} J_\alpha(\phi, \mathbf{u}) + J_\epsilon(\phi) \\
& \text{s.t.} \quad \begin{cases} -\Delta \mathbf{u} + \nabla p + \alpha(\phi) \mathbf{u} = \mathbf{0}, & \mathbf{x} \in D, \\ \nabla \cdot \mathbf{u} = 0, & \mathbf{x} \in D, \\ \int_D \phi \, d\mathbf{x} = C_V |D|, & \mathbf{x} \in D, \\ \mathbf{u} = \mathbf{g}, & \mathbf{x} \in \partial D, \end{cases} \quad (20)
\end{aligned}$$

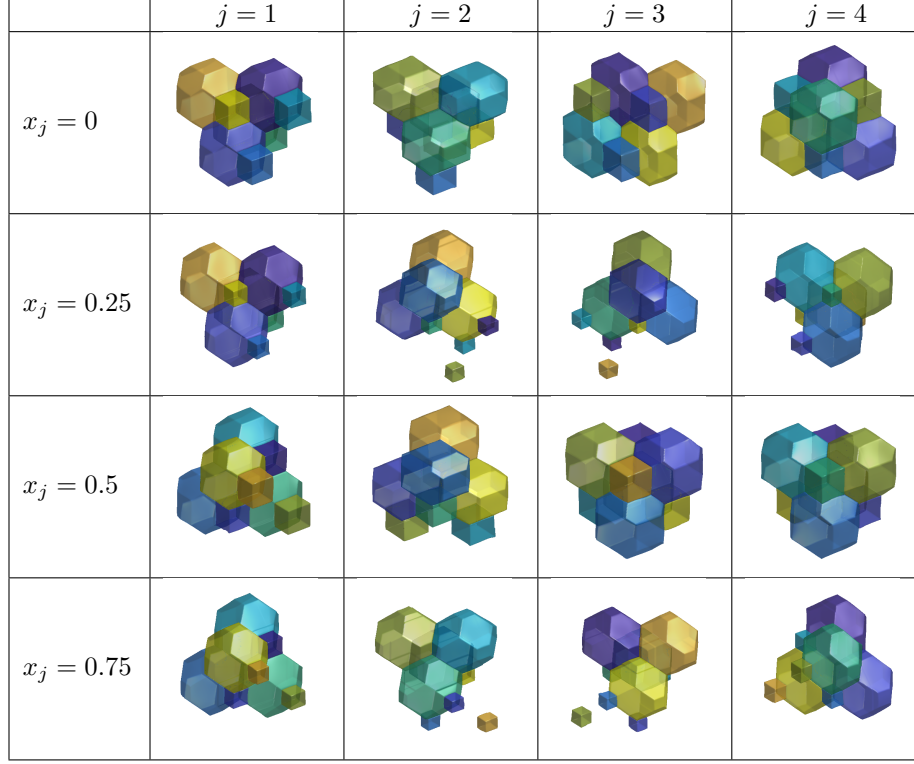


FIGURE 14. A $n = 8$ Dirichlet partition problem with periodic boundary conditions in the 4-dimensional flat torus. The four columns correspond to the slides perpendicular to the x_1 -, x_2 -, x_3 -, and x_4 -axis respectively. The four rows correspond to the slices at $x_j = 0, 0.25, 0.5, 0.75$, respectively. These partitions are based on testing the trained neural network on a uniform 100^3 grid. See Section 4.2.

where

$$J_\alpha(\phi, \mathbf{u}) = \int_D \frac{1}{2} |\nabla \mathbf{u}|^2 d\mathbf{x} + \int_D \frac{1}{2} \alpha(\phi) |\mathbf{u}|^2 d\mathbf{x}, \quad (21)$$

$$J_\epsilon(\phi) = \int_D \frac{\epsilon}{2} |\nabla \phi|^2 + \frac{1}{\epsilon} F(\phi) d\mathbf{x}. \quad (22)$$

Here, $\alpha(\phi) = \alpha_0(1 - \phi)^2$ is a penalty function with a constant $\alpha_0 > 0$, and $F(\phi) = \frac{1}{4}\phi^2(1 - \phi)^2$ is a double-well potential. $|D|$ represents the volume of the computational domain D , and C_V denotes the volume fraction. p is the fluid pressure, and $\mathbf{g}$ is a given function defined on the boundary ∂D . To simplify the above problem, we interpret the pressure p as the Lagrange multiplier for the divergence-free constraint, resulting in the following modified objective functional:

$$J = \int_D \frac{1}{2} |\nabla \mathbf{u}|^2 d\mathbf{x} + \int_D \frac{1}{2} \alpha(\phi) |\mathbf{u}|^2 d\mathbf{x} + \int_D \frac{\epsilon}{2} |\nabla \phi|^2 + \frac{1}{\epsilon} F(\phi) d\mathbf{x} - \int_D p \nabla \cdot \mathbf{u} d\mathbf{x}. \quad (23)$$

Setting the variational derivative of J with respect to p and u equal to zero yields:

$$\begin{aligned}\frac{\delta J}{\delta p} &= \nabla \cdot \mathbf{u} = 0, \\ \frac{\delta J}{\delta \mathbf{u}} &= -\Delta \mathbf{u} + \alpha(\phi)\mathbf{u} + \nabla p = \mathbf{0}.\end{aligned}\tag{24}$$

This indicates that the first PDE constraint of Equation (20) is inherently satisfied, and thus the solution to Equation (25) is also a solution to Equation (20).

$$\begin{aligned}\min_{\mathbf{u}, \phi} & J_\alpha(\phi, \mathbf{u}) + J_\epsilon(\phi) \\ \text{s.t.} & \begin{cases} \nabla \cdot \mathbf{u} = 0, & \mathbf{x} \in D, \\ \int_D \phi \, d\mathbf{x} = C_V |D|, & \mathbf{x} \in D, \\ \mathbf{u} = \mathbf{g}, & \mathbf{x} \in \partial D. \end{cases}\end{aligned}\tag{25}$$

The proposed WAN-AL is then applied to model (25). For this specific problem, we introduce three Lagrange multipliers represented by three individual neural networks: $p(\mathbf{x}; \eta_0)$, $\lambda_1(0; \eta_1)$, $\lambda_2(\mathbf{x}; \eta_2)$, to incorporate the divergence-free constraint, boundary condition constraint, and volume preserving constraint, respectively. We then arrive at the following minimax problem:

$$\min_{\theta} \max_{\eta_0, \eta_1, \eta_2} L_\beta(\mathbf{u}(\mathbf{x}; \theta), \phi(\mathbf{x}; \theta), p(\mathbf{x}; \eta_0), \lambda_1(0; \eta_1), \lambda_2(\mathbf{x}; \eta_2)),\tag{26}$$

where

$$\begin{aligned}L_\beta(\mathbf{u}(\mathbf{x}; \theta), \phi(\mathbf{x}; \theta), p(\mathbf{x}; \eta_0), \lambda_1(0; \eta_1), \lambda_2(\mathbf{x}; \eta_2)) &= C_\alpha J_\alpha(\phi(\mathbf{x}; \theta), \mathbf{u}(\mathbf{x}; \theta)) \\ &+ C_\epsilon J_\epsilon(\phi(\mathbf{x}; \theta)) - \int_D p(\mathbf{x}; \eta_0) \nabla \cdot \mathbf{u}(\mathbf{x}; \theta) d\mathbf{x} + \frac{\beta_u}{2} \int_D (\nabla \cdot \mathbf{u}(\mathbf{x}; \theta))^2 d\mathbf{x} \\ &- \lambda_1(0; \eta_1) \left(\int_D \phi(\mathbf{x}; \theta) d\mathbf{x} - C_V |D| \right) + \frac{\beta_\phi}{2} \left(\int_D \phi(\mathbf{x}; \theta) d\mathbf{x} - C_V |D| \right)^2 \\ &- \int_{\partial D} \lambda_2(\mathbf{x}; \eta_2) (\mathbf{u}(\mathbf{x}; \theta) - \mathbf{g}(\mathbf{x})) \, ds + \frac{\beta_b}{2} \int_{\partial D} (\mathbf{u}(\mathbf{x}; \theta) - \mathbf{g}(\mathbf{x}))^2 \, ds.\end{aligned}\tag{27}$$

Here, the primal network has an output dimension of 3, which includes the two entries in $\mathbf{u}(\mathbf{x}; \theta)$ and the phase variable $\phi(\mathbf{x}; \theta)$. The adversarial network consists of three individual networks, with one involving the Lagrange multipliers corresponding to $\nabla \cdot \mathbf{u}$ as pressure $p(\mathbf{x}; \eta_0)$, and the other two involving the multipliers $\lambda_1(0; \eta_1)$ and $\lambda_2(\mathbf{x}; \eta_2)$ corresponding to the volume preserving and boundary condition constraints, respectively.

In the following, we consider two examples of the fluid-solid optimization problem using WAN-AL. The hyperparameters are set as follows: $\epsilon = 0.01$, $\alpha_0 = 250000$, $C_\alpha = 100$, $C_\epsilon = 10$, $\beta_u = 100$, $\beta_\phi = 100$, $\beta_b = 1000$, and $\alpha_u = \alpha_\phi = \alpha_b = 1.0003$. In Example 1, the primal network is set to $N_d = 4$ and $N_w = 50$, while in Example 2, $N_d = 6$ and $N_w = 80$. The constructions of three adversarial neural networks are as follows. For the Lagrange multiplier p , we use a ResNet with the same depth and width as the primal network to represent it as $p(\mathbf{x}; \eta_0)$ for each example. For the Lagrange multiplier λ_1 , which enforces the area constraint, we use a shallow neural network with 1 hidden layer and width 10, denoted as $\lambda_1(0; \eta_1)$ to represent it. For the Lagrange multiplier $\lambda_2(\mathbf{x}; \eta_2)$, which enforces the boundary conditions, we use a ResNet with 4 hidden layers and width 50 to represent it. The training is performed for $N = 5000$ steps with inner iteration steps $N_u = 3$, $N_p = N_\phi = N_b = 1$, and

$N_r = 40000$ points are used for training in each iteration. The testing results are obtained by evaluating the trained neural network on a uniform 1000×1000 grid.

Example 1: In Figure 15, we display the results of fluid-solid optimization in $D = [0, 1] \times [0, 1]$ with $C_v = \frac{1}{2}$ and the following boundary conditions:

$$\begin{cases} u_1(0, y) = \frac{1}{2} \sin(y\pi), & y \in [0, 1], \\ u_1(1, y) = \frac{3}{2} \sin((3y - 1)\pi), & y \in [\frac{1}{3}, \frac{2}{3}], \\ u_1(x, y) = u_2(x, y) = 0, & \text{elsewhere on } \partial D. \end{cases} \quad (28)$$

The training results, including the phase variable $\phi(\mathbf{x}; \theta)$ and the velocity field $\mathbf{u}(\mathbf{x}; \theta)$, are presented in Figure 15. The training result exhibits a smooth transition from a large opening on the left to a smaller opening on the right. The second column shows the phase variable based on the projection given below:

$$\begin{cases} \tilde{\phi}(\mathbf{x}; \theta) = 1 & \text{if } \phi(\mathbf{x}; \theta) \geq \frac{1}{2}, \\ \tilde{\phi}(\mathbf{x}; \theta) = 0 & \text{if } \phi(\mathbf{x}; \theta) < \frac{1}{2}. \end{cases} \quad (29)$$

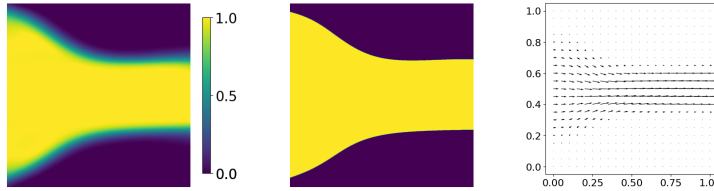


FIGURE 15. The phase variable (left), projection fluid-solid region (middle), and the velocity field (right) for Example 1. See Section 5

Example 2: In Figure 16, we display the results of fluid-solid optimization problems for cases with the same boundary conditions but different optimization regions, denoted as $D = [0, l] \times [0, 1]$, where l represents the length of the desired domain and takes two values: $\frac{3}{2}$ and $\frac{1}{2}$. The boundary conditions are specified below and $C_v = \frac{1}{3}$,

$$\begin{cases} u_1(0, y) = u_1(l, y) = 1 - (12y - 3)^2, & y \in [\frac{1}{6}, \frac{2}{6}], \\ u_1(0, y) = u_1(l, y) = 1 - (12y - 9)^2, & y \in [\frac{4}{6}, \frac{5}{6}], \\ u_1(x, y) = u_2(x, y) = 0, & \text{elsewhere on } \partial D. \end{cases} \quad (30)$$

The training results are presented in Figure 16. The first and second rows correspond to the results of $l = \frac{3}{2}$ and $l = \frac{1}{2}$, respectively. It is noteworthy that in this example, the boundary conditions, network architecture, hyperparameter settings, and other configurations are the same, with the only difference being the optimization region, i.e., different values of l . By employing the identical WAN-AL procedure, we can obtain the corresponding optimal channels. Specifically, for $l = \frac{1}{2}$, two independent, disconnected channels form, while for $l = \frac{3}{2}$, the channels merge and then separate, creating an X-shaped configuration. These results agree well with the results in [12], further highlighting the capabilities of WAN-AL.

6. Conclusion and discussion. In this paper, we introduced WAN-AL, a novel framework inspired by Weak Adversarial Networks (WAN) [56, 1]. WAN-AL leverages an augmented Lagrangian formulation with separate networks for decision variables and Lagrange multipliers, enhancing robustness to hyperparameter tuning and

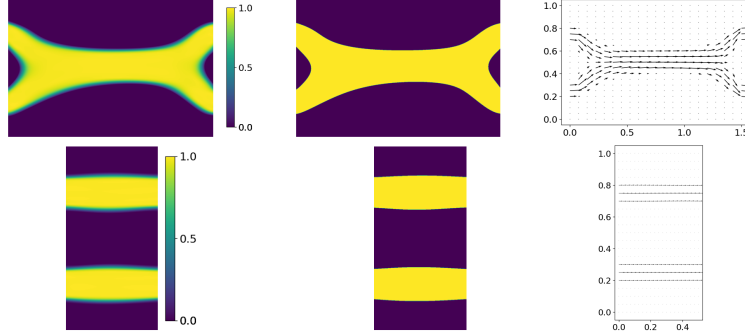


FIGURE 16. The phase variable (left), projection fluid-solid region (middle), and the velocity field (right) for Example 2. See Section 5.

enabling adaptive training through an adversarial term. As a general framework, WAN-AL improves training efficiency and achieves superior results when combined with other methods to deal with interface optimal design problems under physical constraints. Numerical experiments demonstrate its performance across diverse interface optimal design problems with different kinds of physical constraints, including scalar linear/nonlinear constraints and PDE constraints. The results demonstrate WAN-AL as a stable, adaptable, and scalable approach, highlighting its potential for broader applications. The WAN-AL framework is inherently extensible, designed for seamless integration with diverse deep learning architectures and training paradigms. Its formulation naturally generalizes to challenging problems beyond the current scope, particularly in constrained optimization with coupled PDE systems, structure-preserving dynamical systems, and priority-weighted data-driven discovery.

Notably, the empirical success of the $\tanh^3$ -ResNet combination in our implementations motivates future theoretical study of $\tanh^3$ activation in adversarial networks—particularly its approximation properties, role in constrained optimization, and connections to interface related problem analysis.

Acknowledgments. G. Bao was partially supported by the National Key Research and Development Program of China (2023YFA1009100), the Key Project of Joint Funds for Regional Innovation and Development (U21A20425), and the Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China, JYB2025XDXM103. D. Wang was partially supported by National Natural Science Foundation of China (Grant No. 12422116), Guangdong Basic and Applied Basic Research Foundation (Grant No. 2023A1515012199), Shenzhen Science and Technology Innovation Program (Grant No. RCYX20221008092843046, JCYJ20220530143803007), Guangdong Provincial Key Laboratory of Mathematical Foundations for Artificial Intelligence (2023B1212010001), and Hetao Shenzhen-Hong Kong Science and Technology Innovation Cooperation Zone Project (No.HZQ SWS-KCCYB-2024016).

REFERENCES

- [1] G. Bao, X. Ye, Y. Zang and H. Zhou, [Numerical solution of inverse problems by weak adversarial networks](#), *Inverse Problems*, **36** (2020), 115003.

- [2] G. Bao and Y. Zang, [A weak-form deep learning framework for solving partial differential equations and inverse problems](#), *ParticleWNN*.
- [3] T. Beale and A. Layton, [On the accuracy of finite difference methods for elliptic problems with interfaces](#), *Communications in Applied Mathematics and Computational Science*, **1** (2006), 91–119.
- [4] B. Bogosel and B. Velichkov, [A multiphase shape optimization problem for eigenvalues: Qualitative study and numerical results](#), *SIAM Journal on Numerical Analysis*, **54** (2016), 210–241.
- [5] J. H. Bramble and J. T. King, [A finite element method for interface problems in domains with smooth boundaries and interfaces](#), *Advances in Computational Mathematics*, **6** (1996), 109–138.
- [6] L. A. Cafferelli and F. H. Lin, [An optimal partition problem for eigenvalues](#), *Journal of Scientific Computing*, **31** (2006), 5–18.
- [7] J. W. Cahn and J. E. Taylor, [A unified approach to motion of grain boundaries, relative tangential translation along grain boundaries, and grain rotation](#), *Acta Materialia*, **52** (2004), 4887–4898.
- [8] S. Cai, Z. Mao, Z. Wang, M. Yin and G. E. Karniadakis, [Physics-informed neural networks \(PINNs\) for fluid mechanics: A review](#), *Acta Mechanica Sinica*, **37** (2021), 1727–1738.
- [9] Z. Cai, X. Ye and S. Zhang, [Discontinuous Galerkin finite element methods for interface problems: A priori and a posteriori error estimations](#), *SIAM Journal on Numerical Analysis*, **49** (2011), 1761–1787.
- [10] L. Cen, W. Hu, D. Wang and X. Wang, [An iterative thresholding method for the minimum compliance problem](#), *Communications in Computational Physics*, **33** (2023), 1189–1216.
- [11] T. Chan and L. Vese, [Active contours without edges](#), *IEEE Transactions on Image Processing*, **10** (2001), 266–277.
- [12] H. Chen, H. Leng, D. Wang and X.-P. Wang, [An efficient threshold dynamics method for topology optimization for fluids](#), *CSIAM Transactions on Applied Mathematics*, **3** (2022), 26–56.
- [13] J. Chen, X. Chi, W. E and Z. Yang, [Bridging traditional and machine learning-based algorithms for solving PDEs: The random feature method](#), *Journal of Machine Learning*, **1** (2022), 268–298.
- [14] Q. Cheng, J. Guo and D. Wang, [Computing optimal partition problems via Lagrange multiplier approach](#), *Journal of Scientific Computing*, **102**.
- [15] Q. Cheng and J. Shen, [Global constraints preserving scalar auxiliary variable schemes for gradient flows](#), *SIAM Journal on Scientific Computing*, **42** (2020), A2489–A2513.
- [16] K. Chu and S. Leung, [A level set method for the Dirichlet k-partition problem](#), *Journal of Scientific Computing*, **86**.
- [17] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi and F. Piccialli, [Scientific machine learning through physics-informed neural networks: Where we are and what's next](#), *Journal of Scientific Computing*, **92**.
- [18] G. Cybenko, [Approximation by superpositions of a sigmoidal function](#), *Mathematics of Control, Signals, and Systems*, **2** (1989), 303–314.
- [19] J. Dolbow and I. Harari, [An efficient finite element method for embedded interface problems](#), *International Journal for Numerical Methods in Engineering*, **78** (2008), 229–252.
- [20] S. Dong and Z. Li, [Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations](#), *Computer Methods in Applied Mechanics and Engineering*, **387** (2021), 114129.
- [21] Q. Du and F. Lin, [Numerical approximations of a norm-preserving gradient flow and applications to an optimal partition problem](#), *Nonlinearity*, **22** (2008), 67–83.
- [22] W. E and B. Yu, [The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems](#), *Communications in Mathematics and Statistics*, **6** (2018), 1–12.
- [23] H. Garcke, C. Hecht, M. Hinze and C. Kahle, [Numerical approximation of phase field based shape and topology optimization for fluids](#), *SIAM Journal on Scientific Computing*, **37** (2015), A1846–A1871.
- [24] M. D. Gunzburger and P. B. Bochev, *Least-Squares Finite Element Methods*, Springer, New York, 2009.
- [25] R. T. Haftka and Z. Gürdal, *Elements of Structural Optimization*, Springer Science & Business Media, 2012.
- [26] C. He, X. Hu and L. Mu, [A mesh-free method using piecewise deep neural network for elliptic interface problems](#), *Journal of Computational and Applied Mathematics*, **412** (2022), 114358.

- [27] K. He, X. Zhang, S. Ren and J. Sun, *Deep Residual Learning for Image Recognition*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016.
- [28] J. Huang, H. Wang and T. Zhou, *An augmented Lagrangian deep learning method for variational problems with essential boundary conditions*, *Communications in Computational Physics*, **31** (2022), 966–986.
- [29] H. J. Hwang and H. Son, *Lagrangian dual framework for conservative neural network solutions of kinetic equations*, *Kinetic and Related Models*, **15** (2022), 551.
- [30] Q. Jia and D. Wang, *Physics Informed Learning of Orthogonal Features with Applications in Solving Partial Differential Equations*, [arXiv:2602.03247](#).
- [31] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang and L. Yang, *Physics-informed machine learning*, *Nature Reviews Physics*, **3** (2021), 422–440.
- [32] D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, [arXiv:1412.6980](#).
- [33] Z. Li, B. Zou, H. Wang, J. Su, D. Wang and X. Xu, *Deep learning-based computational method for soft matter dynamics: Deep Onsager-Machlup method*, *Communications in Computational Physics*, **37** (2025), 353–382.
- [34] L. Lu, X. Meng, Z. Mao and G. E. Karniadakis, *DeepXDE: A deep learning library for solving differential equations*, *SIAM Review*, **63** (2021), 208–228.
- [35] L. Lu, R. Pestourie, W. Yao, Z. Wang, F. Verdugo and S. G. Johnson, *Physics-informed neural networks with hard constraints for inverse design*, *SIAM Journal on Scientific Computing*, **43** (2021), B1105–B1132.
- [36] L. Lyu, K. Wu, R. Du and J. Chen, *Enforcing exact boundary and initial conditions in the deep mixed residual method*, *CSIAM Transactions on Applied Mathematics*, **2** (2021), 748–775.
- [37] R. Massjung, *An unfitted discontinuous Galerkin method applied to elliptic interface problems*, *SIAM Journal on Numerical Analysis*, **50** (2012), 3134–3162.
- [38] B. Merriman, J. K. Bence and S. Osher, *Diffusion Generated Motion by Mean Curvature*, Department of Mathematics, University of California, Los Angeles, 1992.
- [39] Y. Nandwani, A. Pathak and P. Singla, *A primal dual formulation for deep learning with constraints*, *Advances in Neural Information Processing Systems*, **32**.
- [40] S. Osher and R. P. Fedkiw, *Level set methods: An overview and some recent results*, *Journal of Computational Physics*, **169** (2001), 463–502.
- [41] M. N. Özişik, H. R. Orlande, M. J. Colaço and R. M. Cotta, *Finite Difference Methods in Heat Transfer*, CRC Press, 2017.
- [42] M. Raissi, P. Perdikaris and G. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, *Journal of Computational Physics*, **378** (2019), 686–707.
- [43] S. Sangalli, E. Erdil, A. Hötker, O. Donati and E. Konukoglu, *Constrained optimization to train neural networks on critical and under-represented classes*, *Advances in Neural Information Processing Systems*, **34** (2021), 25400–25411.
- [44] S. Sharma, S. Sharma and A. Athaiya, *Activation functions in neural networks*, *International Journal of Engineering Applied Sciences and Technology*, **04** (2020), 310–316.
- [45] O. Sigmund and K. Maute, *Topology optimization approaches: A comparative review*, *Structural and Multidisciplinary Optimization*, **48** (2013), 1031–1055.
- [46] J. Sirignano and K. Spiliopoulos, *DGM: A deep learning algorithm for solving partial differential equations*, *Journal of Computational Physics*, **375** (2018), 1339–1364.
- [47] H. Son, S. W. Cho and H. J. Hwang, *Enhanced physics-informed neural networks with augmented Lagrangian relaxation method (AL-PINNs)*, *Neurocomputing*, **548** (2023), 126424.
- [48] I. Steinbach, *Phase-field models in materials science*, *Modelling and Simulation in Materials Science and Engineering*, **17** (2009), 073001.
- [49] N. Sukumar and A. Srivastava, *Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks*, *Computer Methods in Applied Mechanics and Engineering*, **389** (2022), 114333.
- [50] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas and Y.-J. Jan, *A front-tracking method for the computations of multiphase flow*, *Journal of Computational Physics*, **169** (2001), 708–759.
- [51] L. A. Vese and T. F. Chan, *A multiphase level set framework for image segmentation using the Mumford and Shah model*, *International Journal of Computer Vision*, **50** (2002), 271–293.
- [52] D. Wang, *An efficient unconditionally stable method for dirichlet partitions in arbitrary domains*, *SIAM Journal on Scientific Computing*, **44** (2022), A2061–A2088.

- [53] D. Wang and B. Osting, [A diffusion generated method for computing Dirichlet partitions](#), *Journal of Computational and Applied Mathematics*, **351** (2019), 302–316.
- [54] Z. Wang and Z. Zhang, [A mesh-free method for interface problems using the deep learning approach](#), *Journal of Computational Physics*, **400** (2020), 108963.
- [55] C. Wu, M. Zhu, Q. Tan, Y. Kartha and L. Lu, [A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks](#), *Computer Methods in Applied Mechanics and Engineering*, **403** (2023), 115671.
- [56] Y. Zang, G. Bao, X. Ye and H. Zhou, [Weak adversarial networks for high-dimensional partial differential equations](#), *Journal of Computational Physics*, **411** (2020), 109409.
- [57] W. Zhu and T. Chan, [Image denoising using mean curvature of image surface](#), *SIAM Journal on Imaging Sciences*, **5** (2012), 1–32.

Received August 1, 2025; revised December 22, 2025; early access February 28, 2026.