Available online at www.sciencedirect.com
ScienceDirect

Comput. Methods Appl. Mech. Engrg. 411 (2023) 116052

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Topology optimization via implicit neural representations

Zeyu Zhang[a], Wen Yao[b,*], Yu Li[b,*], Weien Zhou[b], Xiaoqian Chen[b]

[a] College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China
[b] Defense Innovation Institute, Chinese Academy of Military Science, Beijing 100071, China

Received 19 January 2023; received in revised form 14 March 2023; accepted 5 April 2023
Available online 24 April 2023

Abstract

Along with the rapid development of artificial intelligence (AI) technology, scientific research enters a new era of AI. Topology optimization (TO) and AI technology are recently showing a growing trend of cross development, which has received continuous attentions from relative researchers. In this paper, we introduce a concept of Implicit Neural Representations from AI into TO field and establish a novel TO framework which is named as TOINR. In TOINR, the Topology Description Function is the inherent combination factor, which is constructed by a Neural Network (NN) and determines the topology statuses of structural materials in the design domain. We adopt sine as a periodic activation function combined with MLP as the architecture of the NN. The inputs of NN are a predefined set of spatial points' coordinates, while the outputs are the implicit representations of corresponding topological boundaries. Along with updating NN's parameters (i.e., design variables), the structural topologies iteratively evolve according to the responses analysis results and optimization functions. A boundary-adaptive multi-resolution finite element analysis method is developed to improve the physical response accuracy. At each step of TOINR, we ensure the computational differentiability. Thus, the automatic differentiation is used in sensitivity analysis. The multi-penalty function approach is applied to handle objects with constraints. Besides, we design an adaptive adjustment scheme for learning rates to enhance the stability of the optimization process. Numerical examples illustrate that TOINR can stably obtain optimized structures for different problems with high performance and robustness.

Keywords: Topology optimization; Neural Network; Implicit representation; Automatic differentiation

1. Introduction

The rapid development of human society and the brilliant modern civilization cannot be achieved without various complex structural systems, such as aircraft and automobiles. In this sense, structures have become the basis for carrying out the physical life of human beings. In order to design structures with better performance, many researchers have devoted themselves to digging and delve structural optimization design methods. Among them, **Topology Optimization** (TO), as an essential means of structural design, has broken through the early limitation of size and shape optimization with the aim of optimizing the spatial layout of materials, greatly enhancing the design freedom and offering a systematic platform for achieving and high-performance structural innovative configuration design.

* Corresponding authors.
E-mail addresses: zhangzeyu_work@outlook.com (Z. Zhang), wendy0782@126.com (W. Yao), liyu_npu@outlook.com (Y. Li), weienzhou@outlook.com (W. Zhou), chenxiaoqian@nudt.edu.cn (X. Chen).

https://doi.org/10.1016/j.cma.2023.116052

As the most effective approach to improving structural performance, TO has been widely used in many industrial fields, e.g., aircraft and car design, architectural design. Generally, TO is an iterative process based on finite element analysis (other PDE solvers can also be used) to seek the optimized material distribution in a design domain by minimizing the object function, meanwhile fulfilling one or more constraints. Since the pioneering work made by Bendsøe and Kikuchi [1], TO has undergone rapid development and evolved into several classes of methods. Nowadays, according to the different schemes of updating the design variables, the current widely used TO methods are as follows: the elemental/nodal density-based method including the Solid Isotropic Material with Penalization (SIMP) approach [2,3] and the Evolutionary Structural Optimization method (ESO) [4]/Bi-Evolutionary Structural Optimization method (BESO) [5], the Level-set methods (LSMs) [6,7], and the Feature-Mapping Methods (e.g., Moving Morphable Component/Void method (MMC/MMV) [8–10]). Refs. [11–15] provide a good overview of these works. Besides, there are also some distinctive methods, e.g., Ref. [16–18] .

Now, TO and Artificial Intelligence (AI) are showing a new trend of intermingling development, especially Deep Learning (DL), a very promising subfield of machine learning based on Deep Neural Network(DNN), which sheds light on the development of new methods for TO. Thus, the field of combining TO with AI frameworks is thriving. Next, we provide a brief overview of these studies, while a more detailed literature review can be found in the Refs. [19–21]. In the following, we refer to the DNN-based TO approach as DNNTO.

In the authors' view, the methodology of DNNTO can be generalized from three different perspectives (see Fig. 1).

1. Categorized according to how the data is learned as regression model-based approaches and generative model-based approaches.

    - *Regression model-based approaches* [22,23] is based on the idea of supervised learning. A dataset with enough labeled samples is prepared in advance to construct a surrogate model for mapping the design condition to the optimized structure. Essentially, this approach regards TO as an image-to-image regression problem, ignoring the intrinsic mechanical properties embedded in TO.
    - *Generative model-based approaches* [24] can be used for both supervised and unsupervised learning. An alluring advantage of using generative models is that the generative characteristics of GAN and VAE in the latent space makes it possible to obtain designs with diversity while satisfying the design conditions.

2. Categorized according to the role played by DNN in TO. Under this criterion, DNNTO can be divided into four categories: *Direct design*, *Sub-procedure substitution*, *Post-processing*, and *Reparameterization*.

    - *Direct design* [25,26] refers to an alternative to the whole TO process by establishing a surrogate model to directly predict an optimal structure by a given design condition to achieve iteration-free TO. As pointed out in Refs. [19,20], such methods face three inevitable problems. Firstly, the high computation cost makes constructing the dataset required for training challenging. Secondly, the trained model often has to severely restrict the displacement and external load configurations, making it difficult to generalize well. Thirdly, the optimized structure does not perform as well as the conventional methods, especially the structural disconnection is inevitable. Although currently subject to many application limitations, this methods gained the most research attention under the great temptation of possibly achieving near "real-time" optimization design.
    - *Sub-procedure substitution* [27,28] refers to the substitution of a part of the TO process, such as sensitivity analysis and finite element analysis, with aiming to reduce computational costs. This approach is more flexible than previously described direct design but also encounters the problems of difficulty in achieving generalizability and the high cost of dataset construction.
    - *Post-processing* is the process of reprocessing the obtained optimization results, such as super-resolution [29] and de-homogenization [30]. The post-processing process involves fewer physical processes, so the DNN does not need to try to understand the complex physical processes during training, and thus this kind of method shows better generalization.
    - *Reparameterization* [19,31–33] differs from the learning-based DNNTO method, which does not need to construct a dataset. It is a method to map the original TO design parameters (e.g., the element density) to DNN parameters on the parameter space.

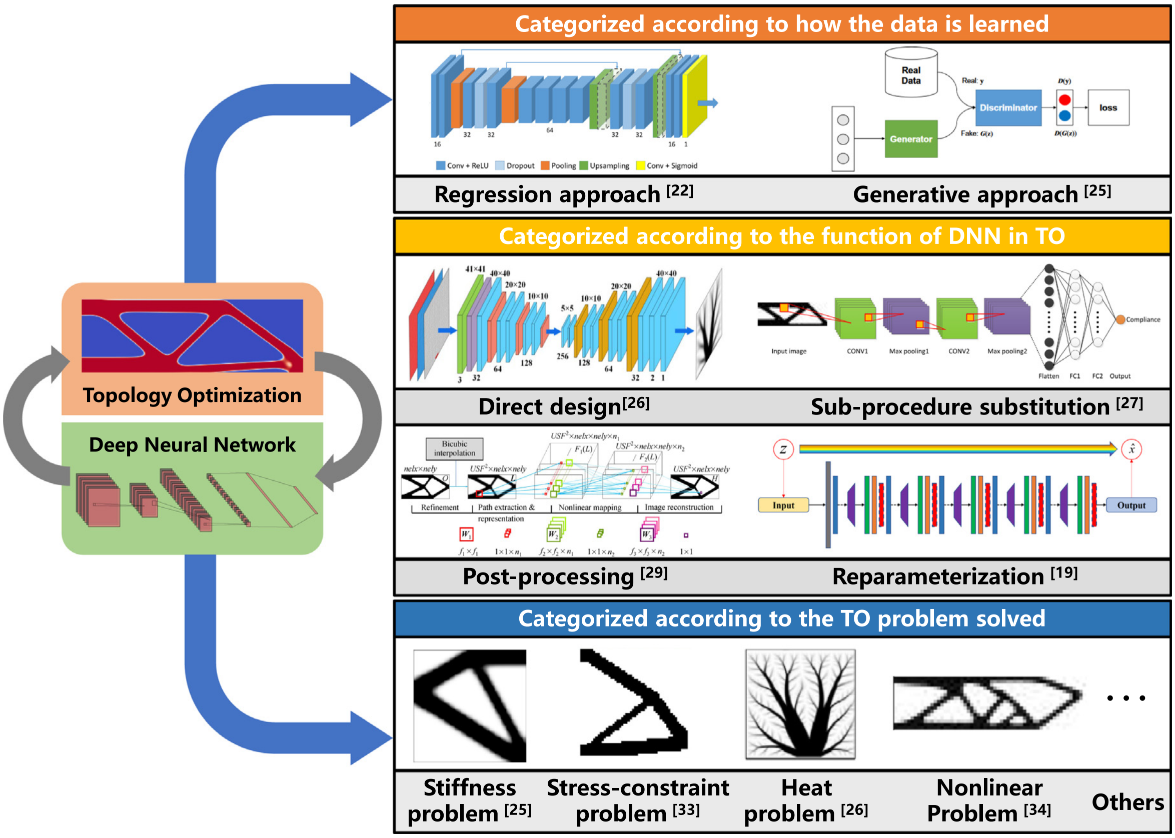


Fig. 1. The DNNTOs are categorized according to different criteria.

3. Categorized according to the characteristics of the TO problem solved.
   As with other TO methods, DNNTO was initially used to solve structural stiffness maximization problems [25] and then gradually expanded to solve other problems, including but not limited to stress constraint problems [34], heat conduction system design problems [26], and nonlinear problems [35].

The summary of the existing research shows that although the research papers on DNNTO abound, as pointed out in Refs. [19,20], most of them follow the baseline of using DL for image segmentation and generation tasks, which aeriform treats the structural topology as "*an image composed of a series of black and white grids*". Such approaches inevitably introduce several side effects, as discussed in the previous review of the *direct design* approach. In contrast, as a learning-free method, the *reparameterization* approach is more similar to the conventional TO method in the problem-solving procedure, where training the neural network (NN) is equivalent to solving the given optimization problem. The design variables are defined as NN's parameters in the *reparameterization* approach. The structural topology is optimized automatically with the update of NN's parameters [19].

In our previous work [19], we conducted an in-depth exploration and analysis of the reparameterization approach and verified the effectiveness of the method in several topology optimization problems, proving that the reparameterization approach has a promising prospect. However, as the research progresses, we argue that simply reparameterizing the pseudo density as NN's parameters does not fully exhibit the advantages and characteristics of NN. As universal approximators, NN naturally has powerful geometric description capabilities. Therefore, searching for a more suitable reparameterization way that better exploits the characteristics of NN has always haunted us.

Meanwhile, the research progress of NN in ***implicit neural representations*** (INR) [36–40] has recently sparked great attention. INR, also referred to as implicit surface/shape learning in some studies, is a novel research field that uses NN to perform approximate representations of a given surface or shape. In INR, the traditional discrete signal representation (e.g., pixel/voxel grids or mesh points) is replaced by a continuous function parameterized by

NN that maps the input signal (e.g., the coordinates of a pixel in the image) to a color, occupancy, or density at the input location. The INR method adequately embodies the advantage properties of NN, including *compactness* and *smoothness* [41–43], which ensure that, theoretically, we can achieve infinite resolution representation of functions in a mesh-free way (decoupling from spatial resolution), as well as the whole process is differentiable. Besides, researches show that INR is not limited to the representation of geometric shapes or surfaces. Just by a brief description, we can find a high degree of similarity between the idea of INR and TO (in particular, TDF methods [44] and LSMs [6,7]). Also, the excellent properties of INR are desired for TO.

Motivated by the aforementioned considerations and on the basis of existing work, this paper attempts to explore a new TO framework by combining INR with TO, which we name **Topology Optimization via Implicit Neural Representations (TOINR)**. TOINR combines the ideas of INR and topology optimization using a topology description function (TDF). TOINR uses the NN to construct the TDF, whereafter it is used to determine the presence or absence of material at an arbitrary point in the design domain. In TOINR, the design variables are the NN's parameters, and the NN's input is a predefined set of spatial point coordinates. Through the predefined spatial points, we achieve the representation of the structural topology configuration with a clear and continuous boundary. Some advanced methods and technologies are proposed or introduced to solve the optimization problem better. We use sine as a periodic activation function combined with MLP as the architecture of the network. A boundary-adaptive multi-resolution finite element analysis method is developed to improve the physical response accuracy. We ensure that every step of the TOINR is differentiable. Thus, we adopt automatic differentiation techniques in the sensitivity analysis, making it unnecessary for us to perform manual sensitivity solving. The multi-penalty function approach is further applied to handle objects with constraints. Besides, we design an adaptive adjustment scheme for learning rates to enhance the stability of the optimization process. We believe that the proposed framework is a novel optimization paradigm combining TO and AI.

Some characteristics are briefly summarized as follows. TOINR is a fully-differentiable TO framework with clear and continuous boundaries, and it uses a single network to represent highly complex geometries without additional "overlap" or "separate" operations. TOINR decoupled from the mesh, enabling a performance-oriented optimization design without any restrictions on the mesh discretization way or mesh size. Meanwhile, it has theoretically infinite spatial resolution. As a method using DL technology, the performance of the optimized structure and the generalization ability is guaranteed. A more systematic assessment of the methodology is in the subsequent section. Based on the characteristics of TOINR, further development and potential extensions could be made in multiscale structure design, multi-physics structure design.

The remainder of this paper is organized as follows. In Section 2, we introduce some preliminary knowledge, mainly including the two ideological origins of the TOINR. In Section 3, a comprehensive and detailed description of the proposed TOINR framework is given, including the procedure, finite element analysis, and some optimization strategies. In Section 4, some representative examples are presented for illustrating the validity of the TOINR. A comprehensive assessment of the TOINR is made in Section 5. Finally, it is summarized in Section 6.

2. Preliminary

As mentioned above, TOINR is conceptually an organic combination of TO and INR. In this section, we will give a brief introduction of the two basic ideas of TOINR.

2.1. Topology optimization using a topology description function

De Ruiter and Van Keulen [44] originally proposed a TO method using *topology description functions* (TDFs). The main idea of the TDF approach is mapping a function $f_\theta : \mathbb{R}^n \rightarrow \mathbb{R}$ on the reference domain into a geometry [44]. Conventionally, the TDF is constructed by the superpositions of a set of basis functions h_i, as shown in Eq. (1). The parameters affecting the properties of this set of basis functions are the design variables for topology optimization.

$$f_\theta(\boldsymbol{X}) = \sum_{i=1}^{N} h_i(\boldsymbol{X}) \tag{1}$$

For example, if the Gaussian probability distribution function is used to construct the TDF, the design variables are the height and width of each kernel function [44]; if the hyperelliptic function is used to construct the TDF, the

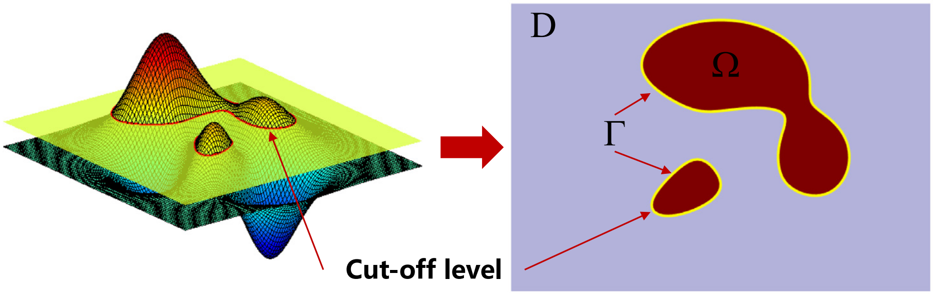


Fig. 2. A TDF f_θ and the corresponding material domain Ω.

design variables are (x_0, y_0, L, t, θ), which denote the center coordinates, the length, the thickness and the inclined angle of each hyperelliptic function, respectively [8].

For a given point $\boldsymbol{X}$ in the design domain $\mathrm{D} \subset \mathbb{R}^2$ or $\mathbb{R}^3$, the material assignment of the point is determined by its corresponding TDF value $f_\theta(\boldsymbol{X})$. As shown in Fig. 2, if the TDF value is higher than the cut-off level $\mathcal{C}$, then the point is assigned as a solid, and vise versa. The point lies on the boundary contour where the TDF value is equal to $\mathcal{C}$ will be the interface between solid and void, i.e.,

$$\begin{cases} f_\theta(\boldsymbol{X}) > \mathcal{C} \Leftrightarrow \boldsymbol{X} \in \Omega \text{ (Material)} \\ f_\theta(\boldsymbol{X}) = \mathcal{C} \Leftrightarrow \boldsymbol{X} \in \Gamma \text{ (Interface)} \\ f_\theta(\boldsymbol{X}) < \mathcal{C} \Leftrightarrow \boldsymbol{X} \in (\mathrm{D}\backslash\Omega) \text{ (Void)} \end{cases} \tag{2}$$

When performing the physical response analysis of the structure, the TDF approach can use any of the available numerical analysis methods. In the case of finite element analysis (FEA) with a fixed grid, finite element discretization of the design domain is required. It should be noted that the mesh discretization is only used to carry out the FEA and is completely decoupled from the design variables. Based on the TDF values at the corners of the element, the equivalent element stiffness matrix is calculated according to the area-fraction of the element [44]. Besides, the "ersatz material" model has been applied to fill the void holes with weak material [7].

The reader may be confused about the concepts of LSMs and the TDF approach because they both represent the structural topology with the help of the cut-off level. The difference between the conventional LSM (Specifically, it refers to the LSM that solves the H-J equation in a discrete way) and TDF is significant. The TDF approach does not involve solving any PDE equations in any way. Some similarities and differences between the TDF approach and conventional LSM are explained in the Ref. [44]. As for now more popular Parameterized-LSM, in most cases, the meaning is similar to that of TDF since they both emphasize describing the structural topology with a set of basis functions with parameters. Especially, Ref. [44] can be regarded as a variation of RBF-LSM.

Although the idea of the TDF approach is straightforward, but the selection of the basis function for constructing the TDF is not arbitrary, and Ref. [44] suggests three principles for the selection of the basis function.

1. It should be computationally convenient to prevent the TDF computation from becoming too expensive.
2. The parameters should be continuously varying.
3. It must be topologically versatile, i.e., it should have the ability to complex geometric representations.

Although the functions that can be used as basis functions usually have better properties, however, it is difficult to achieve the representation of the entire structure topology by a single basis function, and multiple basis functions are required to work together. The involved superposition processing among basis functions brings additional computational overhead and oscillations in the optimization process. Besides, the superposition process of some specific basis functions may be non-differentiable. Therefore, in addition to the above principles, we expect the TDF to have another two properties as follows.

1. the computation of the TDF values is straightforward and does not require additional function superposition processing.
2. the computation process of TDF is fully differentiable.

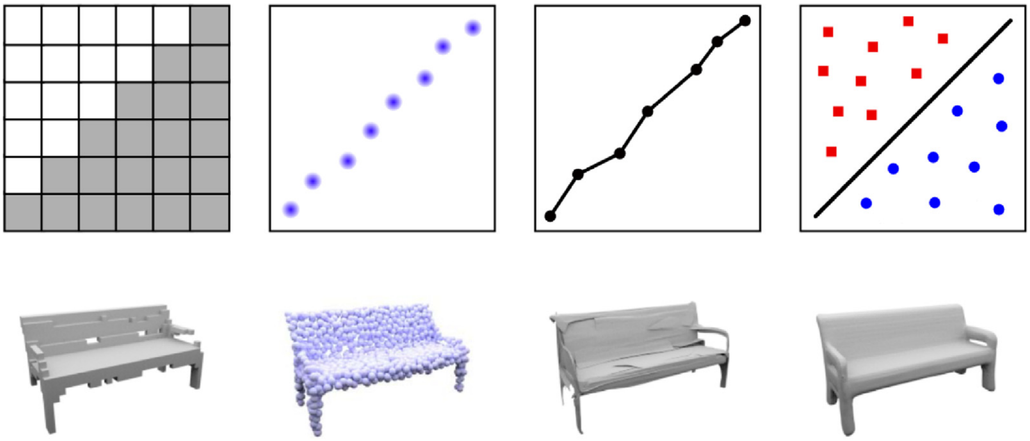


Fig. 3. The existing representations methods are categorized according to the output forms, from left to right, as voxel-based methods, point cloud-based methods, mesh-based methods, and INR methods [45].

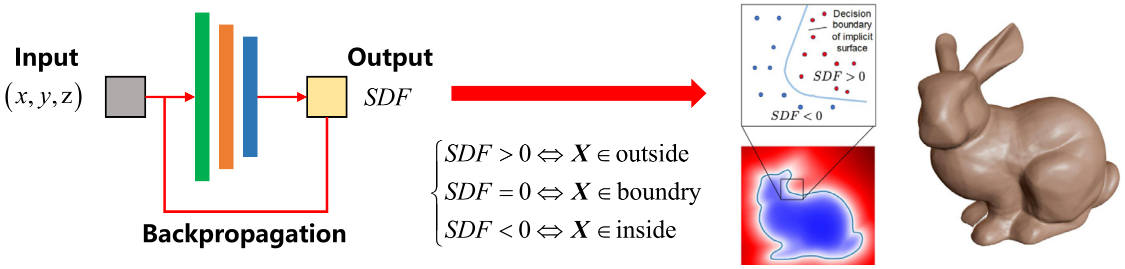

Input
(x, y, z)
Output
SDF
Backpropagation
$SDF > 0 \Leftrightarrow \boldsymbol{X} \in$ outside
$SDF = 0 \Leftrightarrow \boldsymbol{X} \in$ boundry
$SDF < 0 \Leftrightarrow \boldsymbol{X} \in$ inside
Decision boundary of implicit surface
$SDF > 0$
$SDF < 0$

Fig. 4. A typical INR-based surface representation procedure [36], not surprisingly, is highly similar to that of TO.

The TDF constructed based on NN perfectly fulfills the above five properties. Even a simple Multilayer Perceptron (MLP) can achieve complex geometric representations without other superposition operations (In Section 4.1, we give an example for verification). Therefore, NN are well-suited for building TDF. Therefore, NN is well-suited for building a TDF.

2.2. Implicit neural representations

NN-based shape/surface representations is a novel research field with increasing popularity in recent years. It was used mainly for reconstructing continuous and renderable surfaces from unstructured and incomplete 3D point clouds, which is a fundamental problem in robotics, vision, and graphics [46]. The NN-based representations methods can be categorized into discrete-form and continuous-form methods according to the output form, the latter being the INR method. Before the advent of INR, the representation of shape/surface were mainly in discrete forms, such as point-based [47], voxel-based [48], and mesh-based [49] representations methods. Compared to representations methods using discrete forms, INR does not require voxel/mesh discretization and has theoretically infinite resolution while providing continuous shape/surface representations. The result of different representations methods based on NN is shown in Fig. 3.

A typical INR-based surface representation [36] is shown in Fig. 4. The input of the NN is the coordinates of a point $\boldsymbol{X}$ in the reference domain, the output is the value of the *Sign Distance Function* (SDF) of the point, and the NN's architecture adopts the commonly used MLP. The SDF is defined as, for a given spatial point $\boldsymbol{X}$, outputs the point's distance to the closest surface, whose sign encodes whether the point is inside (negative) or outside (positive) of the boundary surface [36]. Therefore, a surface $\mathcal{S} \subset \mathbb{R}^3$ can be implicitly represented by the SDF's zero cut-off level. By backpropagation, the NN's parameters are updated, and the accuracy of surface representation is optimized.

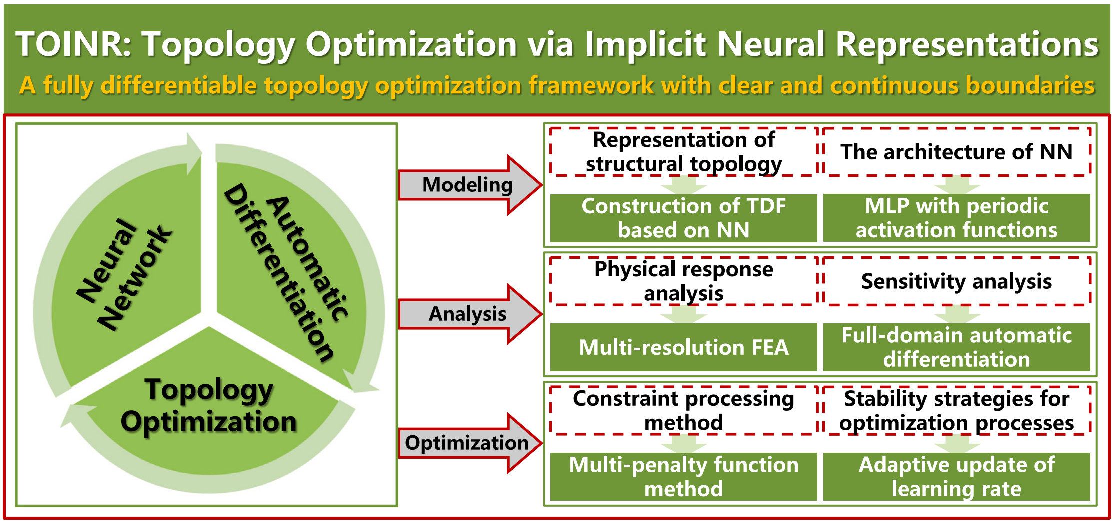

TOINR: Topology Optimization via Implicit Neural Representations
A fully differentiable topology optimization framework with clear and continuous boundaries
Neural Network
Automatic Differentiation
Topology Optimization
Modeling
Analysis
Optimization
Representation of structural topology
The architecture of NN
Construction of TDF based on NN
MLP with periodic activation functions
Physical response analysis
Sensitivity analysis
Multi-resolution FEA
Full-domain automatic differentiation
Constraint processing method
Stability strategies for optimization processes
Multi-penalty function method
Adaptive update of learning rate

Fig. 5. TOINR: Topology Optimization via Implicit Neural Representations.

Thus, several similarities between INR and TDF in TO could be concluded as

1. Same input form and a particular property of a point in the reference domain need to be computed.
2. Same output form and a surface $\mathcal{S}$ is internally implied.
3. Both INR and TDF-based TO are gradient-driven iterative processes.

The above three assertions reflect the inherent similarities between the TDF and INR methods. TDF can be considered a process of representation of a particular surface from the latent space. The initial and final structural topologies are equivalent to a worse representation and a better representation of the theoretical optimal structure, respectively.

Therefore, a natural idea is to combine the TDF approach with INR, which gives rise to the core of what we are trying to elaborate on in this paper—the TOINR method.

3. Methodology and strategy

After introducing the two sources of TOINR, we now spotlight the introduction of TOINR. As highlighted in Fig. 5, TOINR is a deep integration of Topology Optimization (TDF-based TO), Neural Network (INR), and Automatic Differentiation (AD-based gradient computation and collaborative parameter updating). Several innovative/advanced technical approaches are implemented and make up the three main modules: Modeling, Analysis and Optimization. In the follow-up part of this section, each method applied in TOINR will be described in detail.

3.1. Framework

The framework of TOINR is shown in Fig. 6. Here, the structural stiffness optimization problem under the linear elastic model is taken as an example to expound each part of TOINR. The mathematical optimization model takes

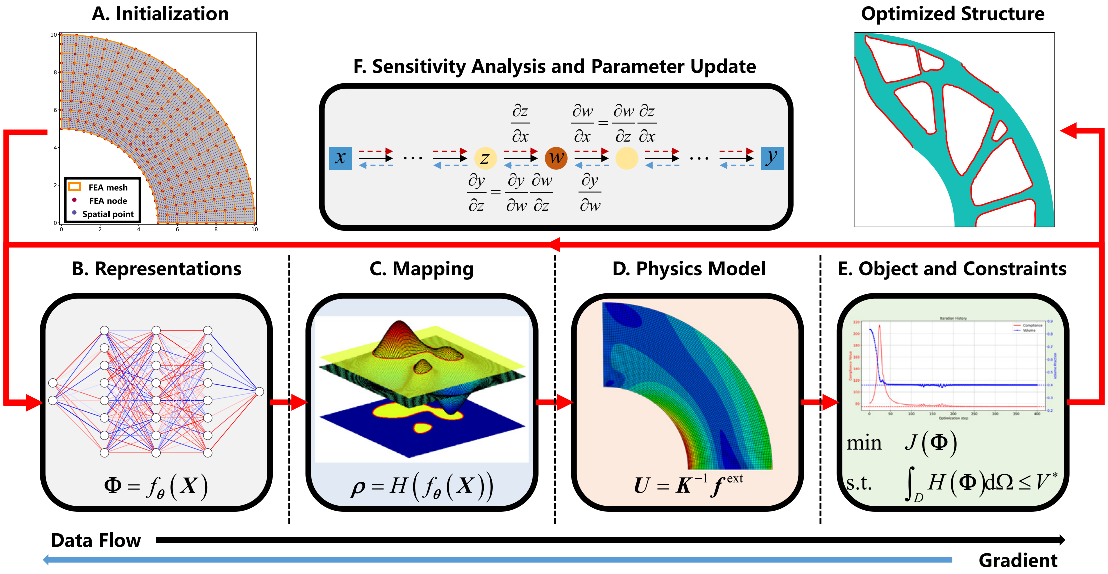

A. Initialization
FEA mesh
FEA node
Spatial point
Optimized Structure
F. Sensitivity Analysis and Parameter Update
x z w y
$\frac{\partial z}{\partial x}$
$\frac{\partial w}{\partial x} = \frac{\partial w}{\partial z}\frac{\partial z}{\partial x}$
$\frac{\partial y}{\partial z} = \frac{\partial y}{\partial w}\frac{\partial w}{\partial z}$
$\frac{\partial y}{\partial w}$
B. Representations
C. Mapping
D. Physics Model
E. Object and Constraints
$\Phi = f_\theta(X)$
$\rho = H(f_\theta(X))$
$U = K^{-1} f^{\text{ext}}$
min $J(\Phi)$
s.t. $\int_D H(\Phi)\mathrm{d}\Omega \le V^*$
Data Flow
Gradient

Fig. 6. The procedure of TOINR, taking the stiffness optimization of the linear elastic model as an example. TOINR consists of six steps before the final optimized structure is obtained.

the following form [6,50]:

$$\begin{aligned}
\min_{\boldsymbol{\Phi}} \quad & J(\boldsymbol{\Phi}) = \int_D (\boldsymbol{\varepsilon}(\boldsymbol{u}) : \boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{u})) H(\boldsymbol{\Phi}) \mathrm{d}\Omega \\
\text{s.t.} \quad & G(\boldsymbol{\Phi}) = \int_D H(\boldsymbol{\Phi}) \mathrm{d}\Omega \le V^* \\
& a(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\Phi}) = l(\boldsymbol{v}, \boldsymbol{\Phi}), \forall \boldsymbol{v} \in \boldsymbol{U} \\
& \boldsymbol{u} = \boldsymbol{u}_0, \text{ in } \Gamma_u \\
& \boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{u}) \cdot \boldsymbol{n} = \boldsymbol{\tau}, \text{ in } \Gamma_\tau
\end{aligned} \tag{3}$$

where the compliance J quantified as the total strain energy of the structure, $\boldsymbol{\Phi}$ represents TDF. $\boldsymbol{u}$ is the displacement field, $\boldsymbol{\varepsilon}$ is the strain tensor, and $\boldsymbol{C}$ denotes the fourth-order material elasticity tensor. The operator (:) denotes tensor contraction. $G(\boldsymbol{\Phi})$ is the volume constraint to limit material usage, while V^* is the maximum allowable volume fraction of the design domain. The linearly elastic equilibrium equation is written in its weak variation form in terms of the energy bilinear form $a(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\Phi})$ and the load linear form $l(\boldsymbol{v}, \boldsymbol{\Phi})$, with $\boldsymbol{v}$ denoting the virtual displacement in the test space $\boldsymbol{U}$, respectively described by Eq. (4).

$$\begin{aligned}
a(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\Phi}) &= \int_D (\boldsymbol{\varepsilon}(\boldsymbol{u}) : \boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{v})) H(\boldsymbol{\Phi}) \mathrm{d}\Omega \\
l(\boldsymbol{v}, \boldsymbol{\Phi}) &= \int_D \boldsymbol{b} \cdot \boldsymbol{v} H(\boldsymbol{\Phi}) \mathrm{d}\Omega + \int_{\Gamma_\tau} \boldsymbol{\tau} \cdot \boldsymbol{v} \mathrm{d}\Gamma
\end{aligned} \tag{4}$$

$\boldsymbol{u}_0$, $\boldsymbol{n}$, $\boldsymbol{\tau}$, $\boldsymbol{b}$ are the given displacement, the boundary normal vector, the traction at boundary, and body force in the domain, respectively. Γ_u and Γ_τ denote the displacement and force boundary, respectively. $H(\cdot)$ is the smoothed Heaviside function defined as follows:

$$H(\boldsymbol{\Phi}) = \begin{cases} \alpha, & \boldsymbol{\Phi} < -\Delta \\ \frac{3(1-\alpha)}{4}\left(\frac{\boldsymbol{\Phi}}{\Delta} - \frac{\boldsymbol{\Phi}^3}{3\Delta^3}\right) + \frac{1+\alpha}{2}, & -\Delta \leqslant \boldsymbol{\Phi} < \Delta \\ 1, & \boldsymbol{\Phi} \geqslant \Delta \end{cases} \tag{5}$$

where a is assigned a small value to avoid the singularity of the global stiffness matrix, and Δ denotes the half-bandwidth. The use of smooth Heaviside function ensures that the whole TO process is differentiable. And to further alleviate the calculation errors caused by the intermediate density elements in the half-bandwidth, we adopt AD and multi-resolution FEA which will be demonstrated later.

As shown in Fig. 6, TOINR can be divided into six parts of A to F:

Part A. *Initialization* is a step performed outside the iteration. The initialization accomplishes two tasks. One is to determine the coordinates of the spatial points in the design domain. A series of predefined spatial points $\boldsymbol{X}$ is required for the TDF to depict the actual structural topology. And the other one is to generate the FEA mesh. As mentioned in the previous section, mesh discretization is used only to carry out the FEA and is completely decoupled from the TDF's design variables. The mesh discretization affects the accuracy of FEA but not the accuracy of structural topology representation, which depends entirely on the geometric representation capability of TDF. As shown in Fig. 6, the blue points are the predefined spatial points, the yellow rectangle represents the FEA mesh, and the red points are the FEA nodes.

Part B. *Representations* is the core part of TOINR. In this step, we construct the TDF based on NN, denoted as f_θ, and the NN's architecture will be introduced in Section 3.2. The input of the NN is the predefined discrete spatial points $\boldsymbol{X}$, and the output is the continuous implicit representation field (i.e., TDF), which is mathematically expressed as follows:

$$\boldsymbol{\Phi} = f_\theta(\boldsymbol{X}) \tag{6}$$

After determining the cut-off level C (usually zero), the structural topology can be obtained according to Eq. (2). With the update of the NN's parameters θ (i.e., the design variables), the TDF $\boldsymbol{\Phi}$ evolves consequently, and thus different structural topologies are obtained. An SDF-based method is employed to initialize the NN's parameters, which will be given in Section 3.6

Part C. *Mapping* is an important part of the formulation of a TDF-based TO. In this step, the continuous geometric model described by TDF is mapped to the discrete mechanical model for FEA. Here, the most commonly used density-based mapping is adopted. The mechanical model is described by a density field $\boldsymbol{\rho}$ (it is referred to as the material model in some studies), which indicating the amount of material at each point of the design domain:

$$\boldsymbol{\rho}(\boldsymbol{X}) = H\left(f_\theta(\boldsymbol{X})\right) \tag{7}$$

In the present work, the fixed grid FEA is used to calculate the structural response, and the well-know ersatz material model is adopted for FEA [6,51]. Since the fixed grid boundary does not match the moving structural boundary, the ersatz material model assumes that the Young's modulus within an FEA element is constant and uses the Young's modulus defined at the center point of that element. With use of the ersatz material model, the Young's modulus of each FEA element E^e can be interpolated as:

$$E^e = E_0 \frac{\sum_{i=1}^{n} \rho_i}{n} \tag{8}$$

where E_0 represents the Young's modulus of solid material, $\rho_i, i = 1, \ldots, n$ are the values of the density $\boldsymbol{\rho}$ on each node of FEA element Ω_e.

Part D. *Physics Model* is the process of carrying on FEA. After obtaining Young's modulus of each EFA element, the element stiffness matrix can be calculated, and then the global stiffness matrix of the structure can be assembled. In order to improve the computational accuracy of the structure near the interface, we introduce multi-resolution FEA as a substitution for the traditional FEA, which will be given in Section 3.3 .

Part E. *Object and Constraints* calculates the objective function and constraints. This step corresponds to the loss function calculation in DL.

Part F. *Sensitivity Analysis and Parameter Update* corresponds to the last part of Fig. 6. As in conventional TO methods, the gradient of the objective and constraint functions with respect to the design variables are vital to the optimization problem. In TOINR, AD is adopted for sensitivity analysis, free the researchers from manual derivations, which is a complex and error-prone process. The choice of the optimizer is crucial for updating the NN's parameters. The MMA algorithm [52] commonly used in conventional TO methods is not suitable for updating the NN's parameters or at least requires extremely tedious parameter adjustment to apply. Here we choose Adam [53] as the optimizer.

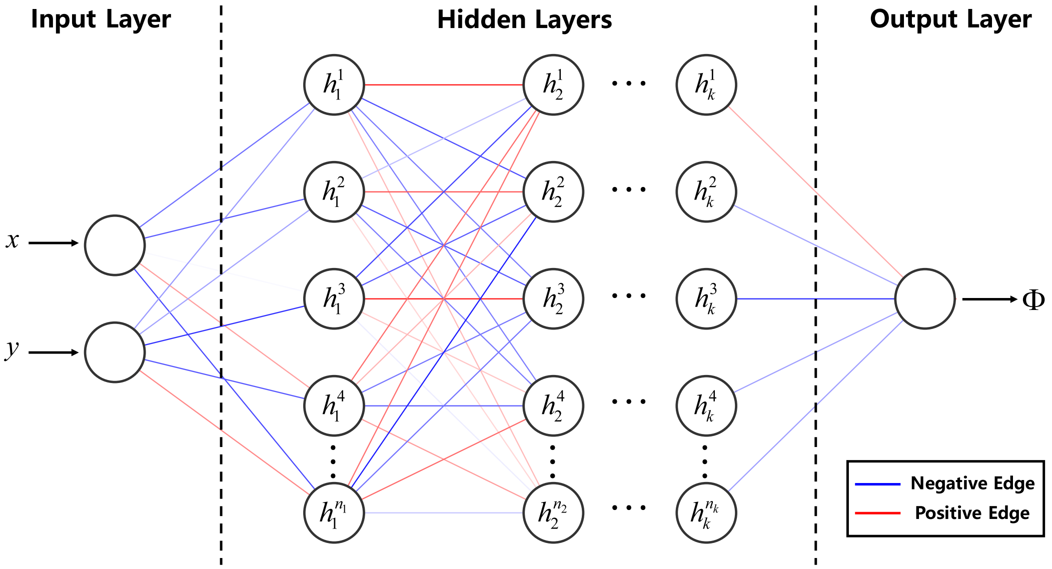


Fig. 7. The architecture of f_θ, which is constructed based on MLP, includes an input layer, several hidden layers, and an output layer. The shades of connecting line colors indicate the magnitude of neuron weight values. The input to the f_θ is the spatial point coordinates and the output is the TDF $\boldsymbol{\Phi}$.

After the above steps, the final optimized structure with a clear and continuous boundary is obtained. To more clearly express the core idea of TOINR, we summarize TOINR as follows:

Definition 1 (***TOINR***)**.** *Training a NN f_θ that parameterizes θ to map spatial coordinates $\boldsymbol{X}$ to TDF value while satisfying some constraints.*

The use of NN to construct the TDF not only satisfies the three principles originally proposed by Ref. [44] but also perfectly meets the additional two properties we raised in Section 2.1. In addition, benefiting from the NN represents the TDF as a continuous field, which implies a theoretically infinite spatial resolution. In the following, we further introduce some of the key technical points of TOINR.

3.2. The architecture of neural network

In this section, we introduce the architecture of NN applied to TOINR. Refs. [41,42] proved that even simple MLP have excellent function approximation capabilities. The study of INR-based representation methods further demonstrates that MLP can remarkably perform shape/surface representations. Inherited from INR, the architecture of NN in this study is designed based on MLP.

Usually, a MLP contains an input layer, an output layer, and several hidden layers, as shown in Fig. 7. The hidden layers may consist of several fully-connected layers and activation functions. The fully-connected layers contain several neurons, and the neurons in adjacent fully-connected layers are connected. Essentially, the fully-connected layer is a linear transformation from one feature space to another feature space. The activation function introduces nonlinearity into the NN, giving it the ability to arbitrarily approximate any nonlinear function. There are various activation functions, e.g., Relu, Sigmoid, and Tanh. The activation function's form affects the NN's performance, and selecting the appropriate activation function according to the task target is important. Therefore, designing the architecture of the NN can be construed as designing the number of fully-connected layers, the number of neurons in each fully-connected layer, the form of the activation function, and whether to introduce some other technical means, such as batch normalization and dropout.

Undoubtedly, NN's architecture affects the optimization effect of TOINR, as similar as the DNN architecture in DL models predictions. An appropriate architecture can improve the optimization effect while accelerating convergence. However, constructing a network is a heuristic design that largely depends on the human experience. It is an inherent challenge to construct an NN suitable for a specific task, even in the traditional DL field. Therefore, the architecture presented in this section does not represent an optimal network architecture but a effective architecture within our experience to illustrate the TOINR framework.

The current commonly used MLP+Relu architecture is incapable of modeling signals with fine detail. Also, the TDF describing the structural topology can be regarded as a multimodal function, and we observe that using MLP+Relu or MLP+Tanh is not efficient in fitting such functions. Inspired by the Ref. [39], we choose the sine as a periodic activation function in TOINR, as described in Eq. (9).

$$\begin{aligned} f_\theta(\boldsymbol{X}) &= \boldsymbol{W}_k \circ h_{k-1} \circ h_{k-2} \circ \cdots \circ h_1(\boldsymbol{X}) + \boldsymbol{b}_k \\ h_i\left(\boldsymbol{Y}_i\right) &= \sin\left(\boldsymbol{W}_i \boldsymbol{Y}_i + \boldsymbol{b}_i\right), i \in [1, k-1] \end{aligned} \tag{9}$$

Here, $h_i : \mathbb{R}^{n_{i-1}} \mapsto \mathbb{R}^{n_i}$ is the ith hidden layer of the network, and $\boldsymbol{Y}_i$ represents the output of h_{i-1} (for $i = 1$, $\boldsymbol{Y}_1 = \boldsymbol{X}$). h_i consists of an affine transformation and a smooth activation function sine, where the former is defined by the weight matrix $\boldsymbol{W}_i \in \mathbb{R}^{n_i \times n_{i-1}}$ and the bias vector $\boldsymbol{b}_i \in \mathbb{R}^{n_i}$, and the activation function is applied to the output of the affine transform. The integer k is the depth of f_θ and n_i means the widths of ith layer. Therefore, the union of their coefficients corresponds to the parameters θ of f_θ. As illustrate in Fig. 7, the input of NN is the spatial point coordinates $\boldsymbol{X}(x, y)$ (for 2D cases), and the output is the TDF $\boldsymbol{\Phi}$.

Using sine as the activation function is not our whim and contains several factors. It has been demonstrated in the Ref. [39] that, based on the same MLP structure, using sine as the activation function can remarkably improve the NN's ability to represent fine structures. A reasonable inference is that TDF constructed using sine as the activation function have better fine structure representation capability. For some specific structural design problems, this may be important. In addition to the preference for representing fine structures, the close correlation between the sine and the Fourier mapping makes it possible to achieve minimum length scale control further. Minimal length scale control based on Fourier representation has been studied in both conventional TO [54] and DNNTO [55]. Among them, Ref. [55] adopts a similar approach to that studied [56] in INR, where the input is the first Fourier mapped before being passed into the fully-connected layer. In fact, such an approach can be viewed as a perceptron with sine as the activation function with two hidden layers. Since cosine is the phase shift of sine, sine and cosine can be written uniformly in the form of sine. Then the perceptron is written as:

$$\boldsymbol{Y} = \boldsymbol{W} \sin(2\pi \boldsymbol{A}\boldsymbol{X} + \boldsymbol{\phi}) + \boldsymbol{b} \tag{10}$$

The similarity can be found by comparing Eqs. (9) and (10). The difference is reflected in the fact that $\boldsymbol{A}$ and $\boldsymbol{\phi}$ are fixed and untrainable when performing the Fourier mapping. Therefore, the MLP with sine as the activation function can achieve minimum length control after careful design, which is exemplified by the Ref. [55].

We believed that through special manipulations TOINR could fulfill the length scale control. And other feature design different NN architecture such as CNN will be further research.

3.3. Multi-resolution finite element analysis

Multi-resolution finite element analysis (MFEA) is introduced in the physical response analysis process of TOINR. Combining MFEA with TO was initially proposed by Ref. [57] and called multi-resolution topology optimization (MTOP). Subsequently, MFEA was successfully applied in the SIMP-based TO [58], MMC/MMV-based TO [59,60], and thermal layout optimization [61].

As introduced in Section 3.1, in the *Mapping* step, from continuous Eq. (5) to discrete Eq. (8), several implementation issues need to be noticed:

- The physical response analysis accuracy is required, such as mesh size and width of narrow-band affect the result. First, using the ersatz material model is based on the premise that the mesh should be fine enough to be smaller than the geometric features of the structure. Suppose the mesh is coarse, where the completely different geometrical features may give the same mechanical model, as detailed in Ref. [60]. This could be a very noticeable problem, especially since the TOINR is decoupled from the background mesh and is possible to produce arbitrary fine geometric features. Second, the mapping based on the smooth Heaviside function introduces a narrow-band of width 2Δ. The spatial points inside the narrow-band region correspond to an intermediate density value $\rho_i \in (0, 1)$. Even if the mesh size is smaller than the geometric features of the structure at this time, the element stiffness matrix calculated in the narrow-band region using the ersatz model is inaccurate.

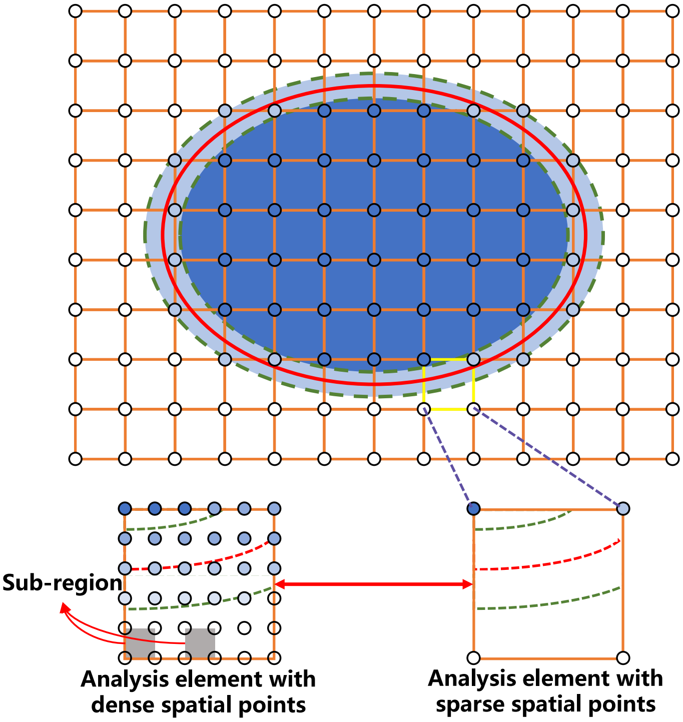

Sub-region
Analysis element with dense spatial points
Analysis element with sparse spatial points

Fig. 8. Topology optimization using BA-MFEA. The green dashed line represents the narrow-band, the dots represent the spatial points, and the orange mesh represents the FEA mesh. As exemplified by the yellow area in the figure, 36 spatial points are used to form 25 subregions within one analysis element.

- The sensitivity accuracy should also be guaranteed. Based on Eq. (5), the derivative of the smooth Heaviside function is given as

$$\delta(\boldsymbol{\Phi}) = \begin{cases} \frac{3(1-\alpha)}{4\Delta}\left(1-\frac{\boldsymbol{\Phi}^2}{\Delta^2}\right), & |\boldsymbol{\Phi}| \leqslant \Delta \\ 0, & |\boldsymbol{\Phi}| > \Delta \end{cases} \tag{11}$$

  From the above equation, it is clear that only the spatial points where the TDF takes values in $[-\Delta, \Delta]$ have the non-zero gradient. More explicitly, for TOINR, only the spatial points that both fall within the narrow-band region and participate in the FEA are genuinely involved in the sensitivity analysis. Suppose we choose a small narrow-band to improve the FEA accuracy, for which the spatial points that participate in the sensitivity analysis may be very scarce, so the sensitivity accuracy cannot be guaranteed, leading to an unfavorable situation for the optimization.

The key to solving the above problems issues collectively is a sufficient spatial resolution near the narrow-band region, which typically by means of additional Gaussian integration points at the boundary. Methods that can satisfy the above needs include X-FEM, FCM, and MFEA. In TOINR, we suggest to apply MFEA. As for MFEA, the additional Gaussian integration points can be predefined without adaptive generation during the optimization, and it is convenient to apply. In addition, in order to reduce the computational cost, we further propose boundary-adaptive multi-resolution finite elements analysis (BA-MFEA).

TO using MFEA introduces two sets of meshes with different resolutions to solve the optimization problem. The analysis element uses a coarser resolution, and the material element uses a finer resolution. TOINR is slightly different because the concept of “material element” does not exist. In TOINR, we no longer use the mechanical model calculated only from the TDF values at the four nodes of the analysis element. More spatial points are selected within an analysis element. Every four adjacent spatial points constitute a subregion, as schematically shown in Fig. 8, the dense subregions are used to perceive the material distribution of the structure. Based on the ersatz material model, the mechanical model is obtained for each subregion, and the contribution of each subregion to the stiffness of the analysis element in which it is located is calculated. It should be emphasized that each subregion has both its unique corresponding Young's modulus and Gaussian integration points, thus truly calculating the stiffness contribution of each subregion to the overall analysis element.

The mechanical models of solid and void regions inside the structure are independent of the spatial resolution, so the high-resolution analysis elements do not need to be constructed for these regions, and the element stiffness matrix for these regions are defined in advance. Hence, BA-MFEA is further proposed to reduce the computational cost. BA-MFEA adaptively identify the structural boundaries and construct analysis elements containing multiple subregions. For TOINR, it is convenient to identify the boundaries adaptively and output the TDF values of the dense spatial points. Because the TDF value of an arbitrary point in the design domain can be determined by only a single NN without any additional operations in TOINR, increasing the number of outputs TDF of spatial points hardly adds any computational burden. Since the boundary region of the structure is a low percentage of the overall structural topology, the use of BA-MFEA can significantly reduce the number of constructed high-resolution analysis elements. After vectorization programming, the additional computational overhead of using BA-MFEA is almost negligible compared to that of using conventional FEA.

The formula for calculating the stiffness matrix of any analysis element located in the boundary region is shown below.

$$\begin{aligned} \boldsymbol{K}^{\mathrm{AE}} &= \sum_{i=1}^{N_{\mathrm{sub}}} A_i \int_{\Omega_i} \left(\boldsymbol{B}^{\mathrm{T}} \boldsymbol{D}_i \boldsymbol{B} J h \right) \mathrm{d}\Omega \\ &= \sum_{i=1}^{N_{\mathrm{sub}}} \left(\left(\sum_{j=1}^{4} w_j \boldsymbol{B}_{i,j}^{\mathrm{T}} \boldsymbol{D}_0 \boldsymbol{B}_{i,j} J_{i,j} \right) E_i \right) h A_i \end{aligned} \tag{12}$$

where $\boldsymbol{K}^{\mathrm{AE}}$ represents the element stiffness matrix of the analysis element. Assuming that an analysis element contains N_{sub} subregions, the subscript i implies the ith subregion, and the corresponding weight of the integrand is the area of the subregion A_i. Ω_i represents the region occupied by the ith subregion, $\boldsymbol{B}$ is the strain–displacement matrix, and $\boldsymbol{D}_i$ is the constitutive matrix of the ith subregion (in Section 3.1, the elasticity tensor is denoted as $\boldsymbol{C}$, and here it is expressed as a matrix form $\boldsymbol{D}$ in Voigt notation). J means the determinant of the jacobian, and h represents the element thickness. j represents the jth Gaussian integration point used in a subregion, ω_j represents the weight of jth integration point, and the subscript i, j means that the quantity is calculated at the jth integration point in the ith subregion. $\boldsymbol{D}_0$ represents the constitutive matrix of the solid material with unit Young's modulus, and E_i is the Young's modulus of the ith subregion. The Young's modulus E_i of each subregion can be calculated from the TDF values of the four spatial points enclosing it, according to Eqs. (7) and (8). As the example shown in Fig. 8, the original analysis element contains only 4 spatial points and uses 4 Gaussian integration points for calculation. In the BA-MFEA, the new analysis element contains 36 spatial points forming 25 subregions and uses 100 Gaussian integration points for calculation.

In summary, the use of BA-MFEA significantly improves the spatial resolution of the boundary region. The equivalent stiffness matrix of the analysis element calculated based on BA-MFEA can better reflect the heterogeneous nature of its internal material distribution, which improves the accuracy of the physical response analysis. At the same time, more spatial points participate in FEA and fall into narrow-band region, enhancing sensitivity analysis accuracy. Even better, BA-MFEA does not increase the degrees of freedom of the linear equation system, and the scale of FEA does not change. Except for a little extra programming ingenuity, we have improved the accuracy of physical response analysis and sensitivity analysis at almost no cost.

3.4. The full domain sensitivity analysis based on automatic differentiation

Both TO and DL update the design variables based on the derivatives of the objective function (the name loss function is preferred in DL) with respect to the design variables. Therefore, a top priority is obtaining gradient information conveniently and accurately. Currently, the following three gradient calculation methods are commonly used [62]:

- *Numerical Differentiation* is represented by the finite difference method, which calculates the gradient based on variable perturbation.

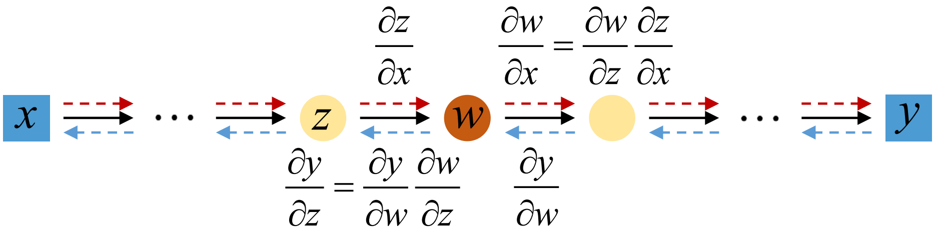

$\frac{\partial z}{\partial x}$
$\frac{\partial w}{\partial x} = \frac{\partial w}{\partial z}\frac{\partial z}{\partial x}$
x
z
w
y
$\frac{\partial y}{\partial z} = \frac{\partial y}{\partial w}\frac{\partial w}{\partial z}$
$\frac{\partial y}{\partial w}$

Fig. 9. Two modes of AD: forward-mode and reverse-mode. The solid black arrow represents the direction of data flow. Forward-mode propagates gradients in the same direction as forward computation, which is depicted as a red dashed arrow in the figure. Reverse-mode propagates gradients in the reverse direction of forward computation, corresponding to the blue dashed arrow in the figure.

- *Manual Differentiation*, i.e., the analytical method, can obtain the exact derivative by deriving manually.
- *Automatic Differentiation* is a set of techniques to efficiently and accurately evaluate the derivative of a numeric function specified by a computer program.

Conventional TO usually uses the analytical method. However, some simplified and approximate methods are also chosen for some complex problems where it is challenging to derive analytical gradients directly, such as stability, fluid, and electric problems. Here, we focus on the powerful technique of AD, which is used to alternative the conventional sensitivity analysis process, freeing the researcher from the tedious manual computing of derivatives while ensuring the accuracy of sensitivity information. As shown in Fig. 6, if TO and NN are viewed as the two sides of the balance, then AD is the middle lever for bridging TO and NN. TOINR conducts sensitivity analysis based on AD to obtain the gradient of the objective function w.r.t. the NN's parameters and then achieves the update of the NN's parameters.

The principle of AD is summarized briefly as follows: any function can be regarded as an ordered composite of a series of elementary operations and elementary functions (e.g., addition, multiplication, sine), no matter how complicated the form of these functions themselves. By predefining the Jacobian matrix of these elementary operations, the computer automatically calculates and accumulates the derivative of the objective function w.r.t. the design parameters accurately according to the chain rule. Specifically, the way of accumulating the gradients contain two basic modes: forward-mode and reverse-mode. The two modes correspond to the direct method and the adjoint method in the analytic method [63,64]. Hence, AD is essentially a fully automatic implementation of the analytic method, which is why AD guarantees high accuracy (We provide the accuracy check in Section 4.2). Two AD models are shown schematically in Fig. 9. The forward-mode suited for few-to-many mappings $f : \mathbb{R}^n \to \mathbb{R}^m, n \ll m$, and the reverse-mode suited for many-to-few mappings $f : \mathbb{R}^n \to \mathbb{R}^m, n \gg m$. For TOINR, the number of design variables is much larger than the amount of output, hence the reverse-mode is the more efficient way.

AD cannot handle operations that are inherently non-differentiable. Thus, we strictly ensure that each of the mathematical operations included in the construction of TOINR is differentiable. Therefore, as a fully differentiable framework, we only need to define the forward computation for TOINR, while the derivative computation is performed automatically by tracing program execution. In particular, in combination with the BA-MFEA method proposed in the previous section, the accuracy of the sensitivity analysis of the boundary region is significantly improved.

Specially, since the design variables of TOINR are NN's parameters decoupled from both the background mesh and the spatial points, all design variables jointly determine the TDF of each spatial point. Therefore, although only the spatial points that fall within the narrow-band region participate in the sensitivity analysis, each parameter can be passed back to a non-zero gradient, and any parameter update will change the whole structural topology. This means that the evolution of the structural topology does not occur only in the boundary region, but any region within the design domain can have a change in the structural topology. Therefore, we call the sensitivity analysis in TOINR as *the full domain sensitivity analysis based on automatic differentiation*.

3.5. The methods for handling constraints

TO is a typical class of optimization problems with constraints. In actual structural engineering design, various criteria could be taken into account as constraints. However, the problems faced in DL are usually unconstrained,

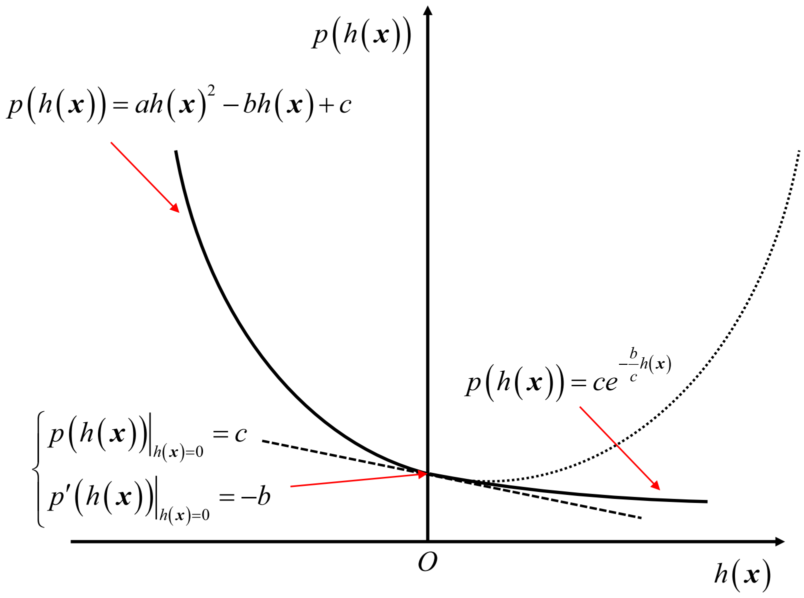


Fig. 10. Curve of the multi-penalty term $p(h(\boldsymbol{x}))$.

and most optimizers belong to unconstrained algorithms that cannot directly deal with constrained problems. Since our research is based on the DL framework (The research in this paper used JAX, but it can be easily implemented in other DL frameworks, such as PyTorch and TensorFlow), an essential question is how to deal with the constraint. Here we borrow the way of solving constrained problems in BESO and LSM, which transform the original constrained problem into an unconstrained problem using the penalty function approach [65].

The unconstrained problem in the kth iteration is

$$\min \varphi\left(\boldsymbol{x}, r_k, t_k\right) = f(\boldsymbol{x}) + r_k \sum_{i=1}^{M} [h_i(\boldsymbol{x})]^2 u_i\left(h_i\right) + t_k \sum_{j=1}^{N} \left[g_j(\boldsymbol{x})\right]^2 \tag{13}$$

where h_i and g_j represent the inequality constraints and equality constraints respectively, r_k and t_k are the penalty coefficients in the kth iteration. Here,

$$u_i\left(h_i\right) = \begin{cases} 1, & h_i(\boldsymbol{x}) < 0 \\ 0, & h_i(\boldsymbol{x}) \geq 0 \end{cases} \tag{14}$$

In each iteration, the penalty coefficients are increased. It is well known that the exterior penalty method will only penalize the violated constraints and cause oscillations in the optimization process. Here we adopt a more robust treatment, using the multi-penalty method [66] instead of the exterior penalty method. The formulation of the multi-penalty function is shown below

$$\begin{aligned} &\varphi(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{i=1}^{M} p_i(h(\boldsymbol{x})) \\ &p(h(\boldsymbol{x})) = \begin{cases} ah(\boldsymbol{x})^2 - bh(\boldsymbol{x}) + c, & (a > 0, b > 0, c > 0), h(\boldsymbol{x}) < 0 \\ ce^{-\frac{b}{c}h(\boldsymbol{x})}, h(\boldsymbol{x}) \geq 0 \end{cases} \end{aligned} \tag{15}$$

where $p(h(\boldsymbol{x}))$ is the penalty term about $h(\boldsymbol{x})$. a, b, and c are the user defined and adjustable parameters, as depicted in Fig. 10. Parameter a is used to regulate the intensity of the penalization of the violated constraints, b and c are defined as the slope and height of the curve at $h(\boldsymbol{x}) = 0$, respectively, while adjusting the penalization of the satisfied constraints at $h(\boldsymbol{x}) \geq 0$. The multi-penalty method combines the features of both the exterior and interior penalty methods. The violated constraints will be penalized. Meanwhile, for a satisfying constraint, a small nonzero value and a small negative derivative of $p(h(\boldsymbol{x}))$ exist to prevent the constraint been violated.

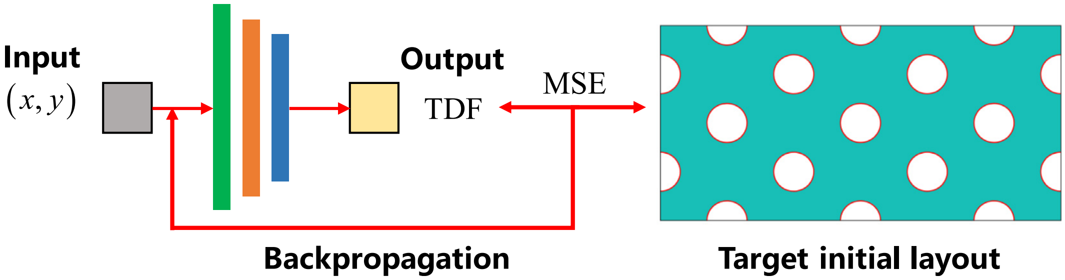


Fig. 11. The procedure of pre-train.

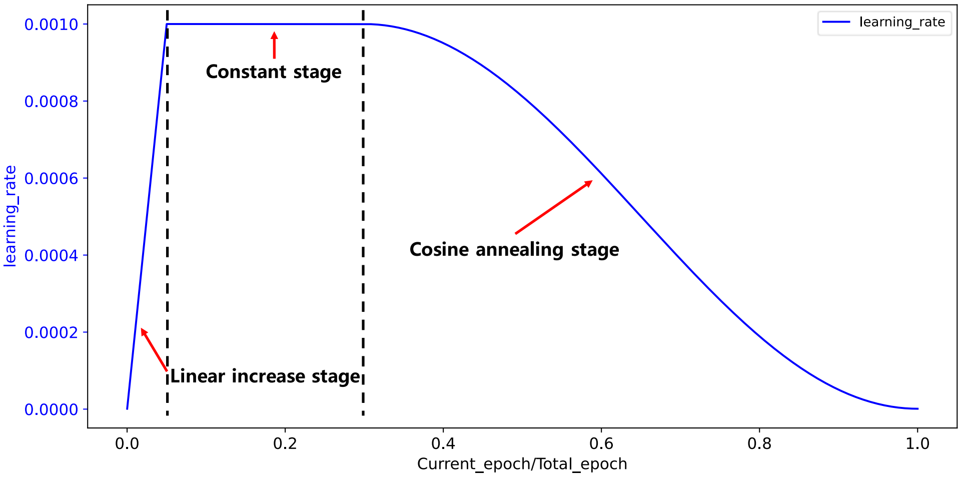


Fig. 12. Adaptive learning rate adjustment scheme.

Here we just provide a effective way to handle constraints, and it is well known that the penalty method requires careful and *ad hoc* tuning of the coefficient of each penalty term in the overall function. Other strategies to handle constraints need to be further developed.

3.6. Algorithm

In the previous section, corresponding to Fig. 5, we have elaborated on the technical details of TOINR. As a method that relies on NN, the inescapable element is the initialization method and the training method of the network (The process of optimizing a NN's parameters is called training).

Here we pre-train the network so that the output structural topology contains multiple uniformly distributed holes, similar to the initial layout often used by the LSMs and the Feature-Mapping methods. The pre-training process is shown in Fig. 11. We use the SDF to construct the implicit representation of the target topology, and Mean Squared Error (MSE) is used as the loss function. The training makes the output of NN close to the target topology, and it stops after reaching the desired representation accuracy, at which the corresponding parameters θ are the initial parameters $\bar{\theta}$ for the subsequent TO task. Here we emphasize that this pre-training does not require the construction of any dataset and is essentially a representation of the target geometry. The pre-training process is fast and does not impose an expensive computational burden. Since the pre-training process does not involve physical processes, only one pre-training is required under a design domain. The trained parameters can be used for arbitrary TO problems (including arbitrary boundary conditions and physical fields).

A key aspect of using a gradient optimization algorithm is selecting the appropriate step size, which corresponds to the concept of learning rate in the NN's optimizer. A large learning rate means a large step size, accelerating the model's training and making it easier to approach the local or global optimal solution. However, a constant learning

Algorithm 1: The entire procedure of TOINR

```
Input: The spatial point coordinates $(x, y)$, the boundary conditions fixeddofs and force conditions $f^{\mathrm{ext}}$,
       the material parameter Young, Poisson, and the predescribed volume $V^*$
Result: The optimized TDF $\Phi$ and plot the final design
According to Section 3.6, determine the initial parameters $\bar{\theta}$ of the network;
for $k = 0$ to $Iter_{\max}$ do
    Acquire the TDF $\Phi$, according to Eq. (6) ;
    According to Eq. (5) and Eq. (7), convert the geometric model to the material model $\boldsymbol{\rho}$;
    According to ersatz material model, calculate the Young's modulus $E^e$ of each FEA element;
    Calculate the stiffness matrix of each FEA element according to Eq. (12);
    Assemble the global stiffness matrix $\boldsymbol{K}$;
    Solve the state equation, $\boldsymbol{U} = \boldsymbol{K}^{-1}\boldsymbol{f}^{\mathrm{ext}}$;
    Calculate the loss function and constraint function according to Eq. (3);
    Update NN's parameters $\theta$ used Adam;
    Save the value of the loss function and the corresponding TDF $\Phi$;
end
Find the minimum value of the loss function and the corresponding TDF $\Phi$;
Plot final design;
```

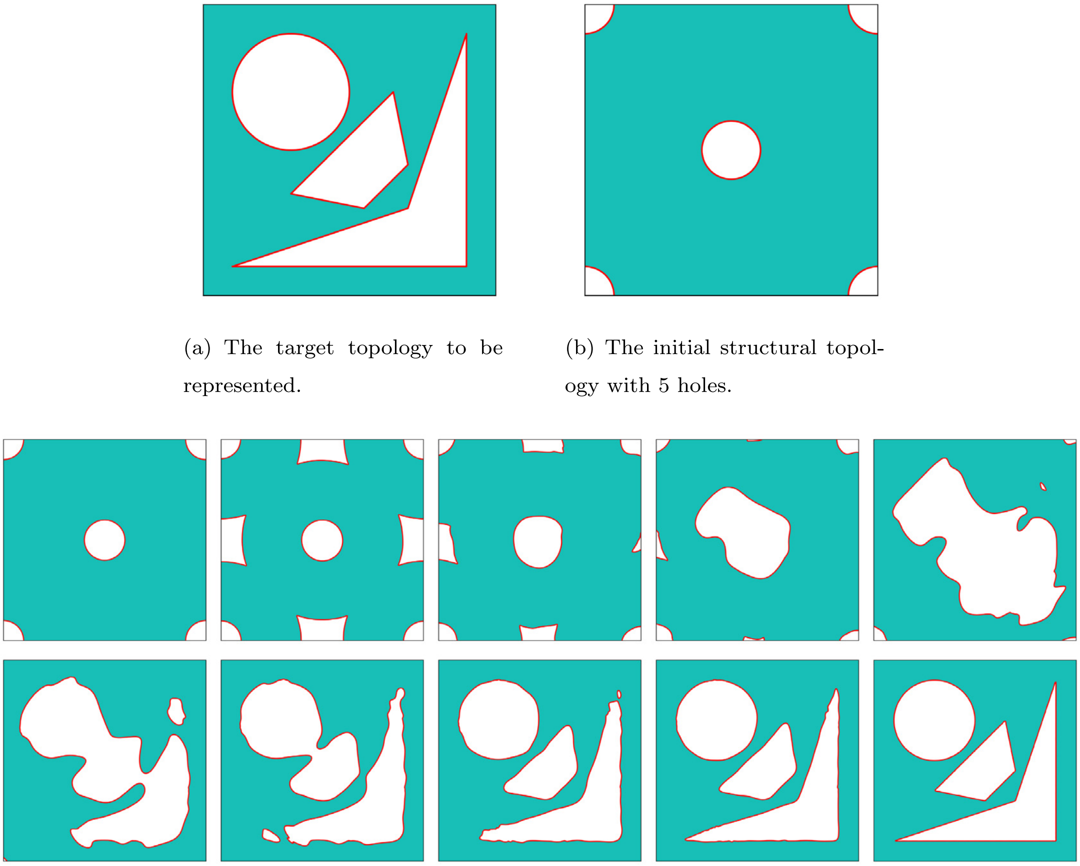
rate is unsuitable for the whole training process. A large learning rate may cause the model to diverge directly at the beginning of training, so a small learning rate is preferable for the beginning of training. Meanwhile, a large learning rate leads to more pronounced oscillations in the late stage, with the loss function values hovering around minimum values and challenging to converge. An effective way to meet the requirements for learning rates at different stages is to adopt an adaptive adjustment scheme for learning rates. At the beginning of the training, it starts with a small learning rate, which is gradually increased to a larger one and maintained for some time so that the model rapidly approaches the optimal solution. Subsequently, the learning rate is gradually reduced as the training continues to ensure that the model does not fluctuate too much in the late stages and thus gets closer to the optimal solution. In TOINR, as shown in Fig. 12, the network is trained with a small learning rate at the initial stage, and then the learning rate is increased linearly until a predetermined maximum learning rate is reached. After maintaining the maximum learning rate for a while, the learning rate decreases following the cosine annealing scheme.

Above all, the entire procedure of TOINR is summarized in Algorithm 1.

4. Numerical examples

In this section, we conduct several numerical experiments to illustrate the effectiveness of TOINR framework. The research in this paper is based on JAX, and it is feasible to use other NN frameworks such as TensorFlow and PyTorch. Through the experiments, we investigate the following.

- *The geometric representation capability of NN*: The first task is to verify the geometric representation capability of the NN to illustrate the rationality of using NN to construct TDF.
- *Accuracy of BA-MFEA and AD*: The second experiment contains two parts, one comparing the calculation results of BA-MFEA with those of ABAQUS to verify the accuracy of the MFEA. And the other verify the accuracy of AD.
- *Validation*: The third task to validate the TOINR by comparing the structural topology for standard 2D benchmark problems against those obtained via conventional methods.
- *Complex design problem*: Next, TOINR is used to solve topology optimization problem with complex design domain.
- *Extensions*: Finally, we extend the experiment to some other design goals, including conduct high-resolution design and consider the non-designed area.



(a) The target topology to be represented.

(b) The initial structural topology with 5 holes.

(c) The process of representation of the target geometry starts from the initial structural topology.

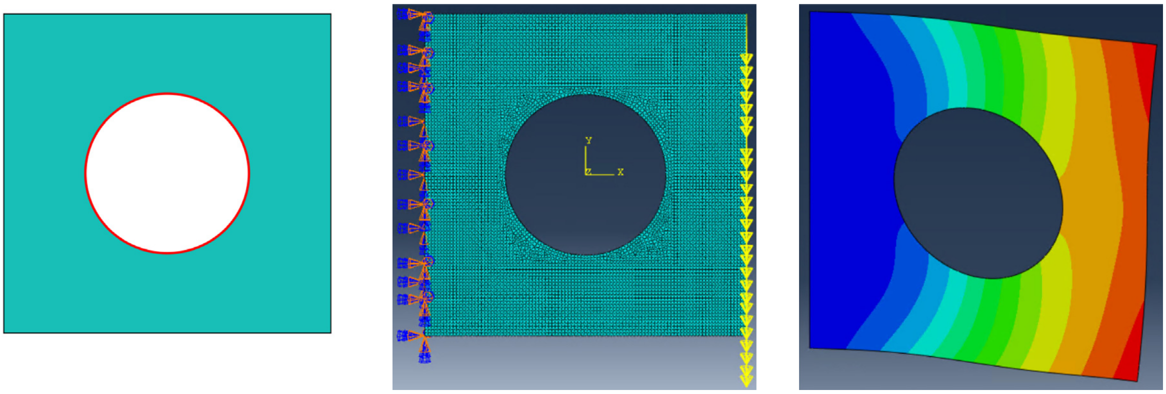
Fig. 13. Demonstration of the geometric representation capability of NN.

4.1. Verification of the geometric representation capability of NN

We employ NN to construct TDF because of NN's powerful geometric representation capability, and in this section, we verify this pre-condition. Besides, assuming that optimized configurations exist, TO can be considered the process of representing optimized configurations with geometries described by topological variables driven by sensitivity. In other words, the essence of representing a particular geometry is the same as TO, the only difference being the choice of a different objective function. Therefore, under the TOINR framework, we construct a geometric representation case as depicted in Fig. 13.

As shown in Fig. 13(a), three non-contiguous holes are inside the target geometry, including a circular hole, a convex quadrilateral hole, and a concave quadrilateral hole. These three different geometric patterns are sufficient to examine the geometric representation capability of the TDF. The initial structural topology is shown in Fig. 13(b). Here we deliberately choose an initial layout with significant differences between the target geometry. There is only one hole inside the initial layout and four other holes at the corners. If we want to represent the target geometry accurately, the structural topology must be adaptive to generate new holes in the interior.

As introduced in Section 3.2, we construct a network containing five fully-connected layers with 100 neurons within each layer. The optimization variables are the NN's parameters, and the objective is to minimize the MSE between the constructed TDF and the target geometry. The process of representing the target geometry, starting from the initial layout, is shown in Fig. 13(c). The figure shows that the target geometry is accurately represented, even from a very different initial layout, which fully demonstrates the NN's powerful geometric representation capability.



(a) Design domain: square plate with a circular hole

(b) Boundary condition and force condition

(c) The analysis result obtained by ABAQUS

Fig. 14. A test case using the square plate with a circular hole for validating the accuracy of BA-MFEA by ABAQUS.

Table 1
ABAQUS analysis results for different hole radii.

Radius	5.000	4.99	5.01	4.999	5.001
Compliance	69.5222	69.2954	69.7515	69.4997	69.5452

At the same time, the process also demonstrates the ability of TOINR for hole nucleation. In Fig. 13(c), it clearly shows that new holes are generated adaptively during the topology evolution, driven by the AD-based full-domain sensitivity.

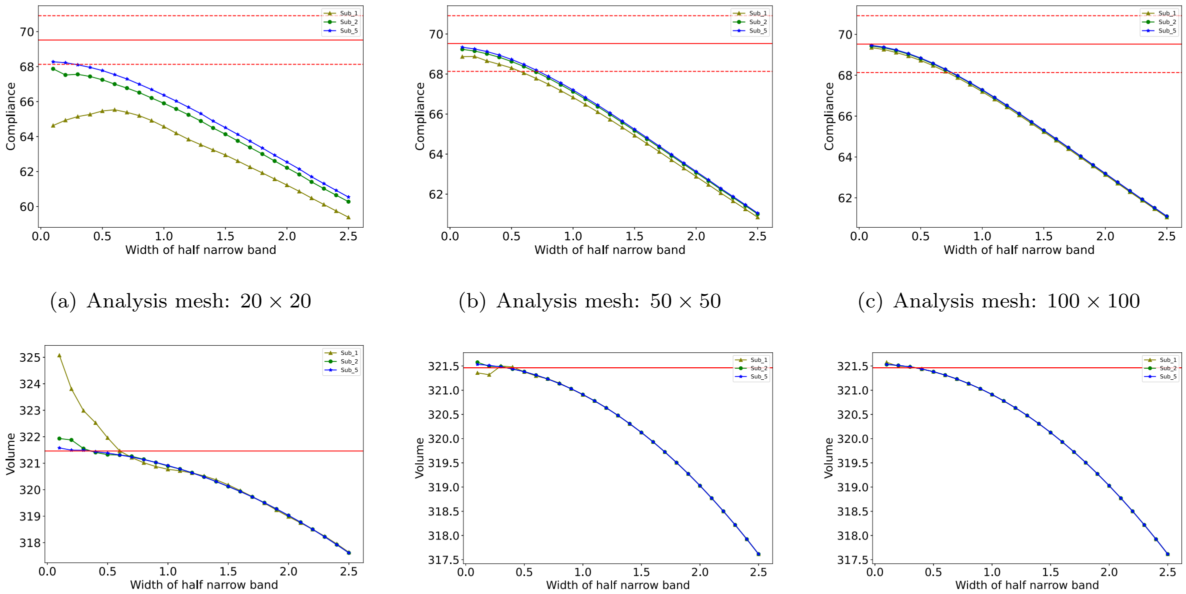
4.2. The verification of MFEA and AD accuracy

In TOINR, we propose the BA-MFEA method as an alternative to conventional FEA and use AD to substitute the conventional manual sensitivity analysis. Here we first verify the accuracy of these two methods. The test case takes a square plate with a circular hole as an example, with structural compliance as the objective function. In this case, the design parameters are the position (x, y) and radius r of the circular hole, and the objective function is the structural compliance. As shown in Fig. 14(a), the size of square plate is $20 \times 20 \times 0.1$, and there is a circular hole with radius $r = 5$ at the center of the plate. Here we use SDF to construct the implicit geometric description field. The boundary condition and force condition is shown in Fig. 14(b). The left side is fixed, and a uniformly distributing force is applied along the edge on the right side with $\sum_i f_i = -1$. The FEA model is built in ABAQUS and meshing is performed using C3D8I element. The dimensionless Young's modulus and Poisson's ratio are $E = 1$ and $\nu = 0.3$, respectively. The analysis results obtained by ABAQUS for different hole radii are listed in Table 1. Fig. 14(c) shows the structure deformation (The deformation effect is scaled up).

4.2.1. Physical response analysis

First we verify the accuracy of the physical response analysis. In BA-MFEA, three factors affect the accuracy of physical response analysis: the size of the analysis mesh, the number of subregions contained in each analysis element, and the half narrow-band width of the boundary region. Therefore, we need to consider the influence of these three factors simultaneously when performing the accuracy calibration of FEA. The plate is discretized using three types of mesh: 20×20, 50×50, and 100×100, and the subregion is divided. The structural compliance and volume with different width of half narrow-band $\varDelta$ are obtained, as shown in Fig. 15. In Fig. 15, the solid red line is the baseline value obtained from the ABAQUS at the high-density mesh. Two deviation values with ±2% errors are also displayed with two red dashed lines for better comparison.

The conclusion can be drawn from Fig. 15 that the BA-MFEA significantly improves the analysis accuracy, especially with a rough analysis mesh. As depicted in Fig. 15(a), the analysis results without BA-MFEA deviate



(a) Analysis mesh: 20 × 20 (b) Analysis mesh: 50 × 50 (c) Analysis mesh: 100 × 100

(d) Analysis mesh: 20 × 20 (e) Analysis mesh: 50 × 50 (f) Analysis mesh: 100 × 100

Fig. 15. The validation of the physical response analysis accuracy of the structural compliance uses different sizes of analysis mesh: 20 × 20, 50 × 50, and 100 × 100. Sub_1: FEA without subregions, Sub_2: BA-MFEA with $2^2 = 4$ subregions, Sub_5: BA-MFEA with $5^5 = 25$ subregions.

far from the baseline when using the rough mesh. At the same time, this deviation is remarkably reduced by using the BA-MFEA with 25 subregions. In the 50 × 50 mesh, BA-MFEA can still improve the computational accuracy noticeably. Finally, the three curves in the 100 × 100 mesh almost overlap because 100 × 100 is already a high resolution for a circle of radius 5, and adding subregions does not significantly impact the analysis accuracy. This experiment can also guide our choice of narrow-band width. With BA-MFEA, a larger narrow-band width can be chosen without compromising the computational accuracy, implying a better perception of possible structural topology changes. Combining the experimental results and previous experience, we believe the narrow-band width of 2 times the minimum element size can balance the computational accuracy and topology change perception ability when BA-MFEA is adopted.

4.2.2. Sensitivity analysis

Here we verify the accuracy of AD. In TOINR, AD is used to calculate the derivative of the loss function $\mathcal{L}$ with respect to NN's parameters θ. Based on the chain rule, the backpropagation can be expressed as:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\rho}} \frac{\partial \boldsymbol{\rho}}{\partial \boldsymbol{\Phi}} \frac{\partial \boldsymbol{\Phi}}{\partial \theta} \tag{16}$$

Instead of using a NN to represent the geometric topology, here we directly use the SDF to represent the geometry, which enables a comparison using conventional sensitivity analysis methods. In the following, we compare the results of AD, conventional sensitivity analysis, and the finite difference method based on the results of ABAQUS in Table 1, using the sensitivity of the circular hole radius variation as an example. Compute the finite difference results for $\Delta r_1 = 0.01$ and $\Delta r_2 = 0.001$:

$$\begin{aligned} &\frac{\text{Comp}(5.01) - \text{Comp}(4.99)}{0.02} = 22.805 \\ &\frac{\text{Comp}(5.001) - \text{Comp}(4.999)}{0.002} = 22.75 \end{aligned} \tag{17}$$

Table 2
Comparison of calculation results by AD, conventional sensitivity analysis method, and the finite difference methods In the table, the conventional sensitivity analysis is abbreviated as CSA. Diff represents the difference between AD and CSA calculation results, and Error means the deviation between the results of AD and the finite difference methods. Other symbols have the same meaning as in Fig. 15.

Mesh	20 × 20			50 × 50			100 × 100		
Subregion	Sub_1	Sub_2	Sub_5	Sub_1	Sub_2	Sub_5	Sub_1	Sub_2	Sub_5
CSA	67.14	34.00	21.57	15.33	22.37	22.61	22.53	22.44	22.65
AD	67.14	34.00	21.57	15.33	22.37	22.61	22.53	22.44	22.65
Diff	<1E−10	<1E−10	<1E−10	<1E−10	<1E−10	<1E−10	<1E−10	<1E−10	<1E−10
Error	194.86%	49.32%	−5.27%	−32.67%	−1.76%	−0.70%	−1.05%	−1.44%	−0.52%

It can be seen that the first-order difference of compliance with respect to radius is stable between 22.75 and 22.8 under the high-density meshes. Using 22.775 as a baseline, the sensitivities calculated by different methods are compared in Table 2. In addition, the analytical sensitivity derivation procedure we give in Appendix.

In this experiment, the narrow-band widths are taken as 0.1. Two conclusions are drawn from Table 2: First, the difference between AD and conventional sensitivity analysis method results is negligible, which further demonstrates that AD has the same accuracy as the analytical method if the programming strictly complies with the formulation. Second, the accuracy of sensitivity analysis is significantly improved by using BA-MFEA, regardless of whether rough or fine meshes are used.

The above two experiments fully demonstrate the effectiveness of BA-MFEA and AD. Especially the adoption of AD reduces the difficulty of solving many complex problems. After defining the forward computation, the computer automatically generates the required gradient by tracing program execution or ahead of time via source code analysis [67]. Hence, TOINR can flexibly compose various predefined elementary functions to build far more complex computations with end-to-end gradient support.

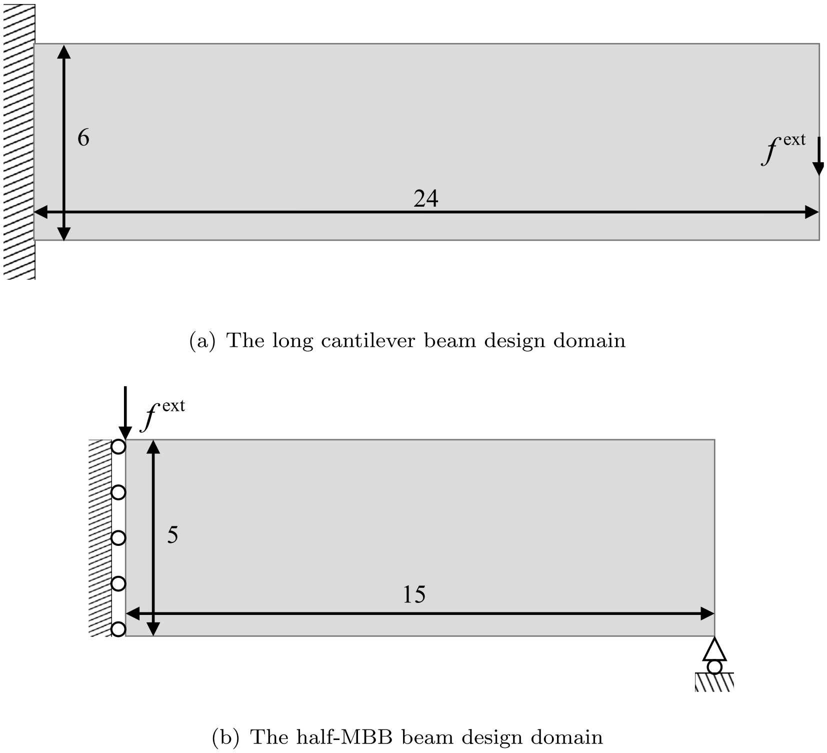
4.3. Validation

In this section, we validate the classic TO task using TOINR, which is used to demonstrate the effectiveness of TOINR. Further, we compare the structural topologies obtained from the TOINR against those obtained using conventional approaches. The optimization formulation refer to Eq. (3). Here, two classic numerical examples are used for validation. The long cantilever beam and the half-MBB beam design domains are shown in Fig. 16. As shown in Fig. 16(a), the long cantilever beam has an aspect ratio of 4 : 1 and is discretized using a mesh of 120 × 30 4-node finite elements. The left side of the beam is fixed, while a concentrated force is applied at the bottom of the right side. The half-MBB beam has an aspect ratio of 3 : 1, and it is discretized using a mesh of 120 × 40 4-node finite elements. A concentrated force is applied at the top of the left side, as illustrated in Fig. 16(b).

In the following numerical examples, we construct a network containing five fully-connected layers with 100 neurons within each layer. Each analysis element in BA-MFEA is subdivided into 25 subregions if not otherwise specified. All physical quantities are assumed to be dimensionless. The Young's moduli and Poisson's ratio for solid material are set to be $E_0 = 1$ and $\nu = 0.3$, respectively. The thickness of the structures are defined as 1. In the long cantilever beam design problem, we choose 601 × 151 spatial points for the subsequent calculation of TDF values. Similarly, in the MBB beam design problem, we selected 601 × 201 spatial points.

The convergence history of the optimization of the long cantilever beams is shown in Fig. 17. The figure depicts the evolution of the compliance and volume fraction of the structural topology with increasing iteration steps. Meanwhile, the optimized structure for some intermediate iteration steps is presented in the figure. The initial structural topology contains several uniformly distributed holes inside, and it can be clearly seen that the shape of these holes evolves as the optimization proceeds. After several iterative steps, the volume fraction of the structural topology gradually approaches the target volume fraction, while the fluctuation of the objective function value gradually becomes smaller until convergence. This also shows that TOINR's constraint-handling approach and optimization strategy are effective.

Next, we compare the optimization results of TOINR with those of the conventional density-based TO methods. As shown in Fig. 18, the optimization results of TOINR, SIMP, and BESO are shown simultaneously. As seen from the figure, the optimized structural configurations obtained by the three methods are very close. In this experiment,



(a) The long cantilever beam design domain

(b) The half-MBB beam design domain

Fig. 16. The design domains.

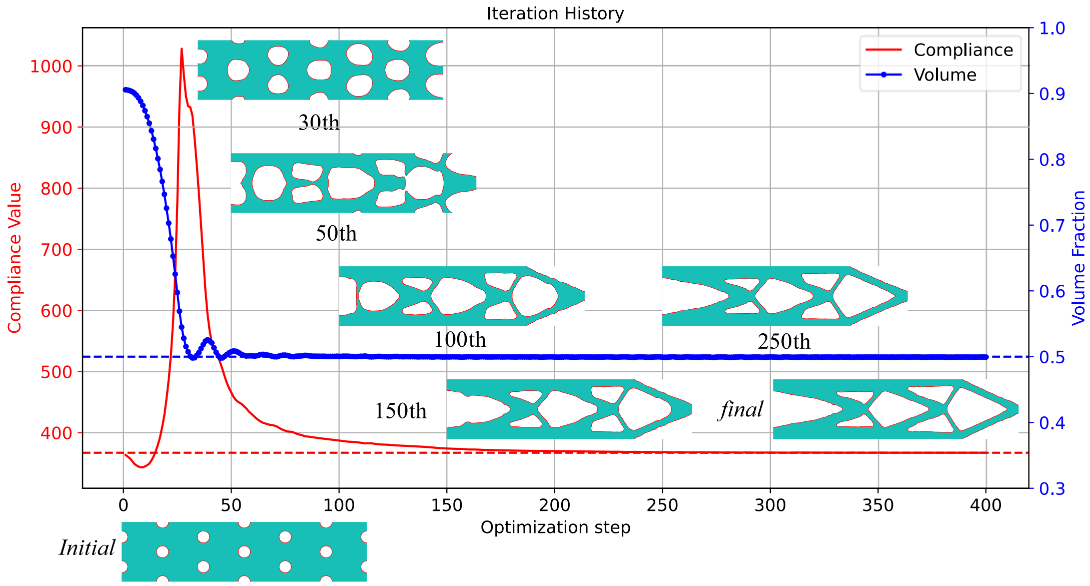
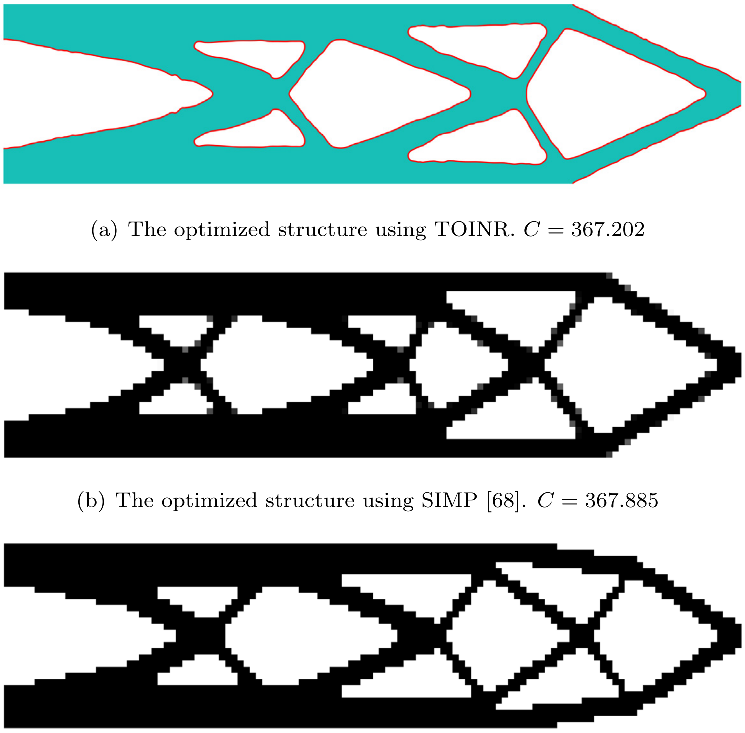


Fig. 17. The convergent histories of the optimization of the long cantilever. The solid red line represents the objective function value, and the red dashed line is the reference line made by the minimum objective function value. The solid blue line with dots represents the volume fraction, and the blue dashed line is the reference line according to the target volume fraction. After some iterative steps, the structural topology finally converges into a clear, smooth topology configuration with $C = 367.202$ and $V = 0.5$.



(a) The optimized structure using TOINR. $C = 367.202$

(b) The optimized structure using SIMP [68]. $C = 367.885$

(c) The optimized structure using BESO [69]. $C = 368.041$

Fig. 18. The optimization results of TOINR are compared with those of conventional density-based TO methods. The results of the SIMP method are obtained using the top99neo code [68], and the results of the BESO method are obtained using the softbeso code [69]. The topology configuration of the final structure obtained using the three methods is highly similar, while the performance of the objective function is very close.

This experiment does not utilize filtering as well and demonstrates good performance.

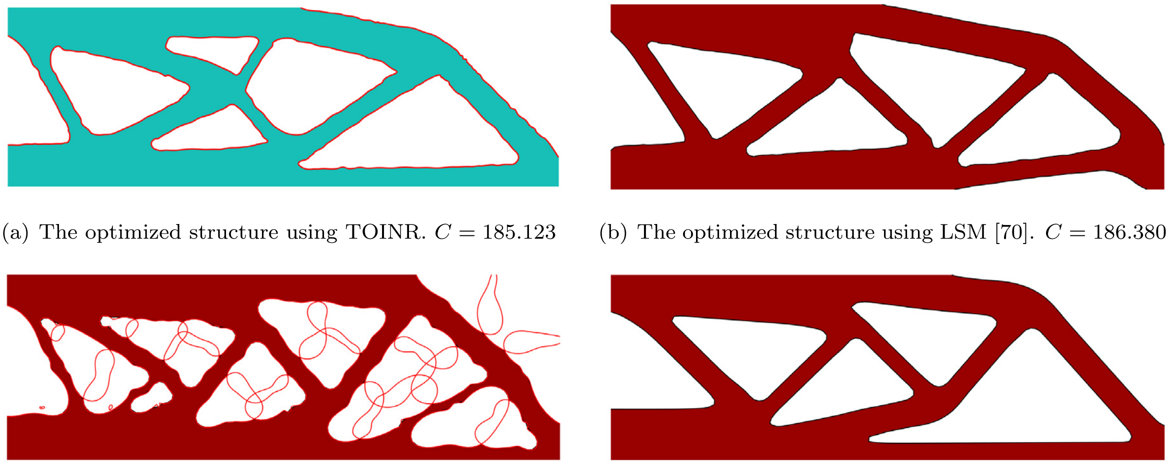
TOINR does not introduce any filtering scheme, while SIMP and BESO adopt a filtering scheme with a filtering radius of 2. In terms of the performance of the objective function, the three methods are close to each other, which fully demonstrates the structural optimization capability of TOINR. It should be noted here that in this experiment, we did not perform careful manual hyperparameter tuning. In addition, the output of TOINR is an implicit field representing the structural topology, so it has clear continuous boundaries without a zig-zag effect.

Next, for the MBB beam design problem, we compare the optimization results of TOINR with other methods that also produce smooth boundaries, including the LSM [70], the MMV [10], and the ETO [71] (which resulting from the combination of BESO and LSM). As shown in Fig. 19, the optimized structures obtained by all four methods have smooth boundaries. Since different methods have different hyperparameters and the manually chosen hyperparameters impact the optimization results, the fine internal structures of the four optimized structures are slightly different, but the main configurations remain highly similar. The performance of the objective functions of the four optimized structures is very close.

By conducting the optimization design of long cantilever beams and MBB beams, it has been demonstrated that TOINR can effectively carry out the structural topology optimization design. Compared with several mainstream topology optimization methods, TOINR does not have any disadvantage in the performance of the objective function and can obtain the same excellent design scheme. Moreover, the optimized structure has a clear and smooth boundary.

4.4. Validation of effectiveness on complex design problem

In the previous section, we verified the effectiveness of TOINR using two classic numerical examples. In this section, we perform the verification on more complex design domains to further demonstrate the effectiveness and



(a) The optimized structure using TOINR. $C = 185.123$

(b) The optimized structure using LSM [70]. $C = 186.380$

(c) The optimized structure using MMV [10]. $C = 186.892$

(d) The optimized structure using ETO [71]. $C = 183.959$

Fig. 19. We compare the results of MBB beam design using different methods. Here we choose the LSM method, the MMV method, and the ETO method as references. The methods used as references are all studies that have appeared within the last five years. We thank the authors for their generosity in making the code open-source to facilitate our comparison. The optimization results obtained by TOINR and the other methods have similar performance, and the optimized configurations are close to each other.

characteristics of TOINR. As shown in Fig. 20, we carry out the optimized design for the quarter annulus in this section.

The quarter annulus with unit concentrated load and the boundary conditions are defined in Fig. 20(a), and two indices are set as $r = 5$ and $R = 10$, respectively. The thickness of the plate is defined as $h = 1$. The spatial point and FEA mesh are defined as shown in Fig. 20(b). Here, the design domain is discretized using a mesh of 100×50 4-node isoparametric finite elements. Each analysis element in BA-MFEA is subdivided into 25 subregions. The prescribed volume fraction is $V^* = 0.4$.

As shown in Fig. 21, the optimized structure of the quarter annulus and several intermediate structures during the optimization process are depicted. The initial designs of the quarter annulus is presented in Fig. 21(a), with some holes evenly distributed inside the structure. Along with the evolution of the internal holes of the structure, the structural volume fraction decreases continuously, and the objective function performance changes. The topology configurations corresponding to the four intermediate iteration steps are shown in the figure. It shows that the structural topology corresponding to the intermediate iteration steps has a clear topological outline, which, combined with the convergence history in Fig. 21(g), can show from the side that the optimization process of TOINR is smooth. After several iterations, the structure reaches the target volume fraction and converges into a final optimized topological configuration, as depicted in Fig. 21(f)·.

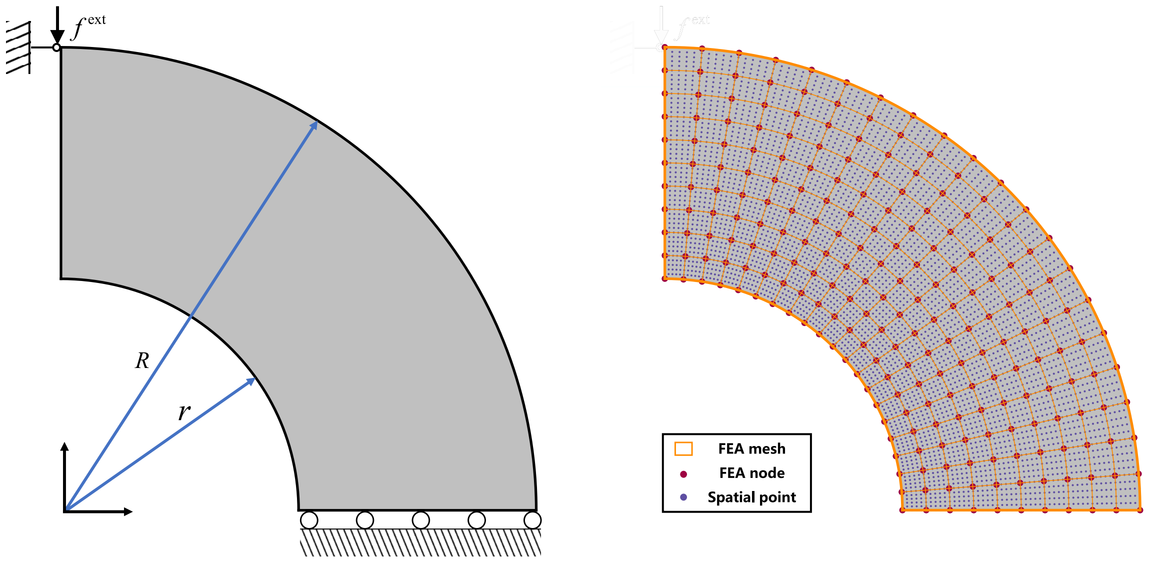
In this experiment, we also compare the optimization results obtained by TOINR with the other methods, as shown in Fig. 22. Here we use the design results of IGA-TO [72] and IGA-MMV [60] to make a comparison with the design results of TOINR. The optimized configurations obtained by TOINR are similar to those obtained by these two methods.

The ability to carry out optimization of the quarter annulus means that TOINR can perform optimization of complex curved-edge structures. Benefits from decoupling the design elements, the optimization capability of TOINR is not limited by the shape of the design domain.

4.5. Extensions

In this section, we consider two application scenarios that are common in practice: the structural optimization design with non-designed domains and the structural super-resolution optimization design.

Firstly, we extend the experiment to consider the non-designed area. which is the basis for many designs considering engineering features and manufacturing constraints to be carried out. Consider a short cantilever beam



(a) The boundary conditions as well as the applied loads of the quarter annulus design problem.

(b) The blue points are the predefined spatial points, the yellow rectangle represents the FEA mesh, and the red points are the FEA nodes.

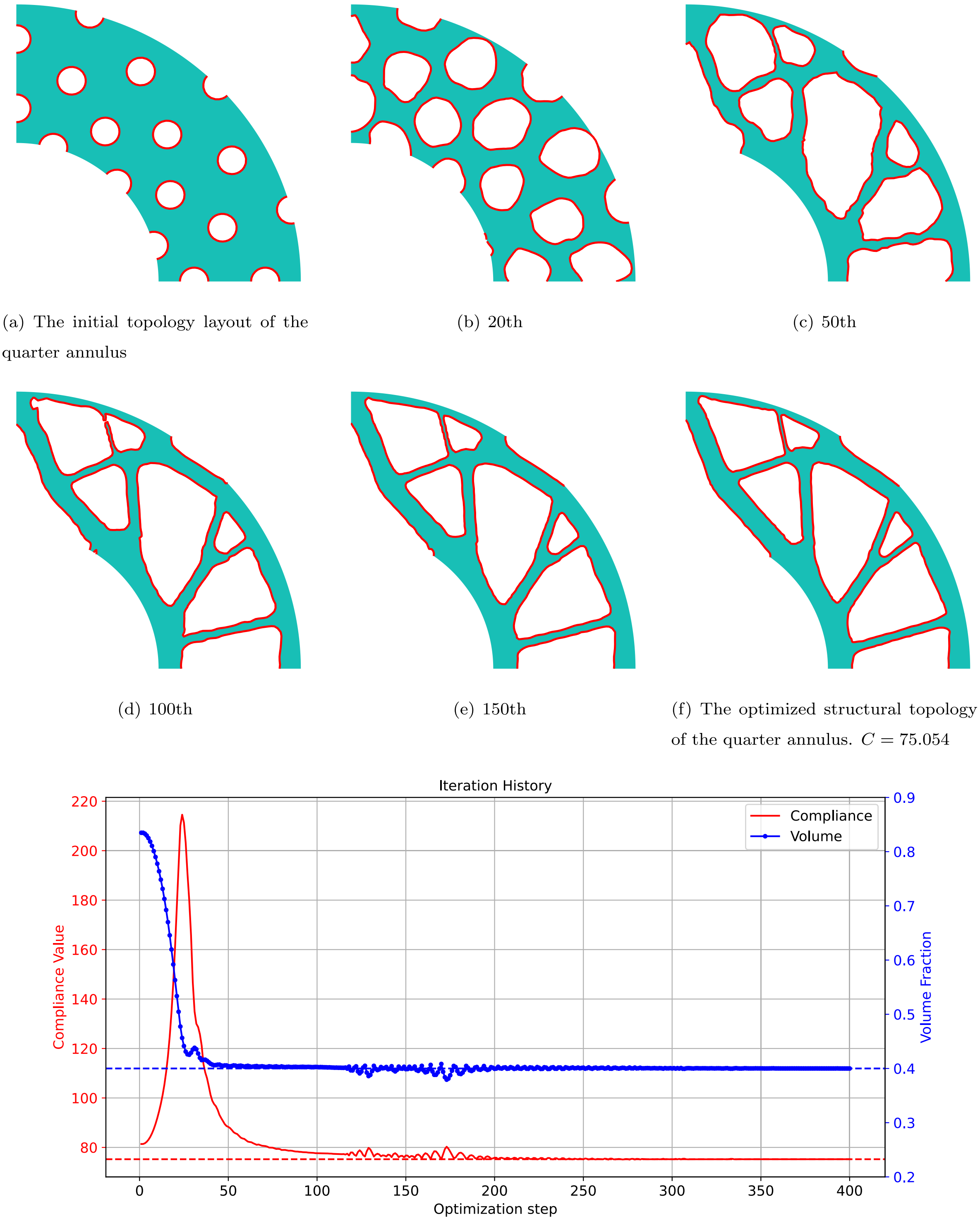
Fig. 20. The design domain of the quarter annulus.

design domain, as shown in Fig. 23, with a circular non-designed area of radius $r = 0.5$, inside which the non-designed area is filled with material. The boundary conditions and the loads are the same as in the long cantilever beam example in Section 4.3. The beam is discretized using a mesh of 80 × 40 4-node finite elements. Here we choose 401 × 201 spatial points for the subsequent calculation of TDF values.

As shown in Fig. 24, the optimized structures ignoring and considering the non-designed area are shown respectively. As indicated in the figure, there is a big difference between the two structural topologies. When the non-design area is ignored, the center of the structure is void, and all the rods bypass the center. After considering the non-design area, the center of the structure is filled with material, becoming the intersection of the rod structures. The side effect is that the structure's compliance increases to 72.065 from 69.336. This experiment demonstrates that TOINR can handle structural optimization problems with non-designed area.

The second scenario we consider is the structural super-resolution design. A long-standing challenge for structural topology optimization is to achieve an efficient high-resolution design. In conventional TO methods, the design element is usually coupled with the analysis element, which may significantly increase computation if the quantity of design elements is increased. A common approach to achieving high-resolution design is using high-density design elements with low-density analysis elements, similar to the idea of the MFEA, using multiple design elements to form a single analysis element. With this approach, it is possible to improve the design resolution without increasing the dimensionality of the linear equation. However, the number of design elements still needs to be determined at the beginning of the design. Suppose the design resolution needs to be further increased after the optimization is completed or in localized areas. Only approximate interpolation can be used to obtain approximate results or to perform the complete optimization again. In TOINR, the structural topology configuration is implicitly embedded in the trained network and is represented by selecting an appropriate number of spatial points. After a complete optimization design using TOINR, we can obtain the TDF value for an arbitrary spatial point in the design domain. Theoretically, inherited from the nature of the NN itself, TOINR has infinite resolution, and we can quickly obtain design results with different resolutions.

Taking the MBB beam design problem shown in Fig. 16(b) as an example, the design results at different resolutions are given after a complete NN training, as shown in Fig. 25. The low-resolution design result is shown in Fig. 25(a), where it uses 121 × 41 spatial points to represent the topological configurations. Fig. 25(b) shows



(a) The initial topology layout of the quarter annulus

(b) 20th

(c) 50th

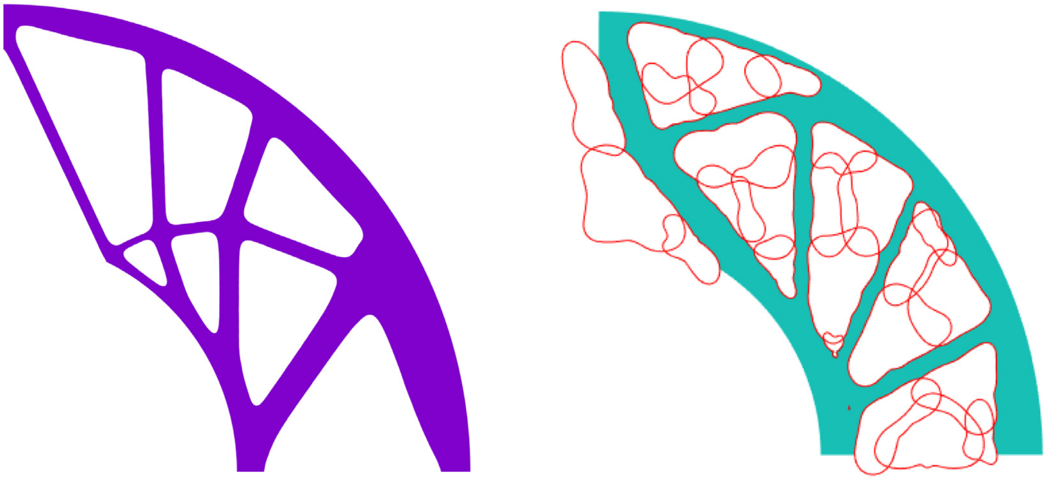
(d) 100th

(e) 150th

(f) The optimized structural topology of the quarter annulus. $C = 75.054$

(g) The convergent histories of the optimization of the quarter annulus.

Fig. 21. The optimization design the quarter annulus.



(a) The optimized structure using IGA-TO [72]. (b) The optimized structure using IGA-MMV [60].

Fig. 22. The optimization results of the quarter annulus using different methods. In such a complex design problem, TOINR can obtain optimized results similar to those of the conventional method.

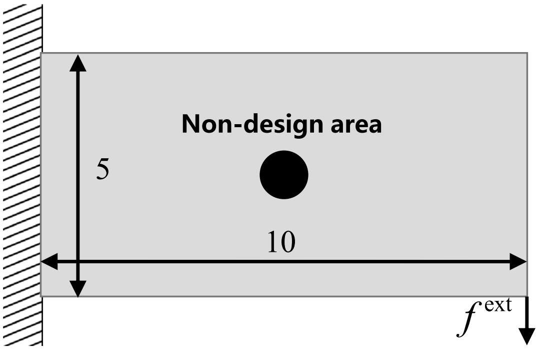
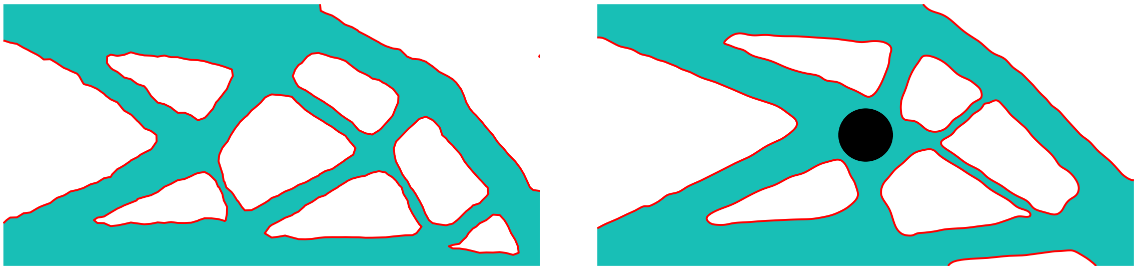


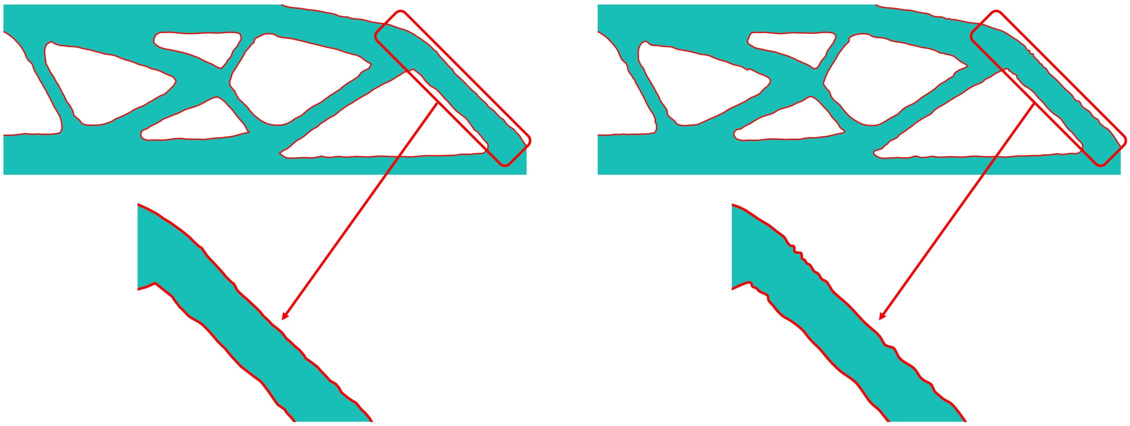
Fig. 23. The short cantilever beam design domain. At the center of the beam, there is a circular non-designed area.



(a) The optimized structure of the non-designed area is ignored. It is clear that no material exists at the center of the structure. $C = 69.336$

(b) The optimized structure considers the non-designed area. The center of the structure is filled with material and becomes the intersection of several rod-like structures. $C = 72.065$

Fig. 24. The optimization structures ignore the non-design domain and consider the design domain.



(a) Low-resolution design results: 121 × 41 spatial points are used.

(b) High-resolution design results: 3001 × 1001 spatial points are used.

Fig. 25. The optimization results of half-MBB beams at different resolutions.

the design results at high resolution, using 3001 × 1001 spatial points to represent the topological configuration. It shows that the design results at both resolutions are highly similar. However, a zoomed-in display of the local part of the structure shows that the high-resolution design results have more detailed features, which also leads to the high-resolution design results not looking as smooth as the low-resolution design results. The experimental results show that TOINR can conveniently generate design results with multiple resolutions, and generating results with different resolutions only requires changing the number of spatial points of the NN's input without any training process. This feature is meaningful for some application scenarios that require high-resolution design results.

5. Discussion

As mentioned in the introduction, TO has undergone tremendous development during the last three decades. Many approaches have been proposed since the pioneering work of Bendsøe and Kikuchi [1]. An important thing to answer for a new TO framework is what are the advantages and distinctive features? Why another new TO framework? To answer this question, in the following, we provide a comprehensive assessment of TOINR.

1. The design variables of TOINR are the parameters of the NN, which makes it completely decoupled from the mesh, truly enabling a completely performance-oriented optimization design without any restrictions on the mesh discretization way or mesh size. In addition, this feature allows TOINR to eliminate the need for additional filtering schemes.
2. The structural topology configuration is implicitly embedded in the trained NN. Beneficial from the properties of the NN itself, TOINR has theoretically infinite resolution, similar to the infinite resolution property of PINN. This characteristic also means that for any spatial point in the design domain, the TDF value of that point can be output as long as its corresponding coordinates are known. Thus the presence or absence of material at that point is known.
3. The inputs of NN are spatial point coordinates with a clear geometric meaning rather than a set of abstract parameters.
4. The structures obtained using TOINR have clear and continuous boundaries. Although TOINR is an implicit method for structural topology representation, well-established methods can extract structural contours and transform them into forms such as the patch.
5. Highly complex geometries can be represented using a single network without additional "overlap" or "separate" operations, thus avoiding introducing the additional computational overhead and causing oscillations in the optimization process.

6. TOINR is fully differentiable, eliminating the need to solve for sensitivity manually. It has some distinctive advantages and certain potential for complex physics design, especially in derivations of complicated states functions in TO.
7. TOINR can automatically generate new holes during the optimization process.
8. In terms of objective function performance, TOINR is able to obtain results comparable to those of conventional methods.

TOINR as an approach to combine AI and TO, we need to respond to some questions that researchers have focused on. In the review published by Sigmund's team on AI combined with TO [20], some questions to consider when working with AI in TO are presented. The insightful questions urge us to think about combining AI and TO better. Here, TOINR is again reviewed from the perspective of combining AI and TO.

1. Compared to most DNNTO based on data-driven ideas, TOINR is significantly different because it is not belong to the data-driven methods. In TOINR, we take advantage of the ability of NN to accurately represent geometry and use NN to construct TDFs, which enables the description of topological configurations. Therefore, we do not need to construct TO datasets composed of a large number of samples in advance.
2. The procedure of TOINR for solving TO problems is similar to the conventional method. Therefore, TOINR has the same generalization ability as the conventional method and can solve various kinds of TO problems. As pointed out in TOINR's "mother", in our previous research on TONR [19], the method based on reparameterization has good generalization and is not limited by boundary conditions and problem types. After replacing the boundary conditions, we need to perform a complete optimization process, which has an entirely different meaning than re-training on a large dataset in DL.
3. TOINR's objective function performance is comparable to that of conventional optimization methods, and we have verified this conclusion in the numerical examples in Section 4, where we are not troubled by the "structural disconnection" phenomenon. Furthermore, we do not want to show that TOINR has made a breakthrough in the objective function performance, but rather that TOINR has some properties that can help us to do better structural optimization design in some problems.

6. Conclusions

In this paper, we construct a new fully-differentiable TO framework with clear and continuous boundaries, TOINR, by combining INR with TO using AD as a bridge. In TOINR, we adopt MLP as the architecture of the network and use the sine as a periodic activation function. In order to improve the accuracy of physical response analysis and sensitivity analysis, we propose the BA-MFEA as a substitution for conventional FEA. To deal with the constraints, we introduce the multi-penalty function method. In addition, we design an adaptive adjustment scheme for learning rates, which makes the optimization process more stable and robust. Numerical examples illustrate that TOINR can stably obtain optimized structures for different problems with high performance and robustness.

As Prof. Sigmund's team pointed out in their review paper [20]: "*The research into using AI-technology in TO has barely begun*". AI-technology has brought revolutionary progress to various research fields. It is still unknown in what kind of the most suitable way it will reshape TO, which needs to be discussed and explored by researchers worldwide. The TOINR method proposed in this paper is only to provide a possible way of combination. We praise its advantages, but we do not deny that there are still some areas that need further improvement. In the future, we will work on improving the TOINR and look forward to seeing more research on the combination of AI and TO.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgments

The authors gratefully acknowledge the support provided by the National Natural Science Foundation of China under Grant No. 51675525, 11725211, and 52005505, and Post-graduate Scientific Research Innovation Project of Hunan Province under Grant No.CX20220059. The author also expresses gratitude to all the anonymous reviewers.

Appendix

Each step in the forward calculation of TOINR is differentiable. Hence we can naturally give an analytical sensitivity analysis. In the following, we give an example of the sensitivity analysis process for structural compliance with respect to the radius of the circular hole, using the numerical example in Section 4.2.2 as an example. The optimization formulation (3) is expressed in finite element form as follows

$$\begin{aligned} \min \quad & J = \boldsymbol{f}_{\text{ext}}^{\text{T}} \boldsymbol{U} \\ \text{s.t.} \quad & \boldsymbol{K}\boldsymbol{U} = \boldsymbol{f}_{\text{ext}} \\ & G(\boldsymbol{\Phi}) = \int_D H(\Phi) \mathrm{d}\Omega \leq V^* \end{aligned} \tag{18}$$

Besides, the BA-MFEA-based expression for the element stiffness matrix (Eq. (12) in manuscript) is presented here for easy reading

$$\begin{aligned} \boldsymbol{K}^{\text{AE}} &= \sum_{i=1}^{N_{\text{sub}}} A_i \int_{\Omega_i} \left(\boldsymbol{B}^{\text{T}} \boldsymbol{D}_i \boldsymbol{B} J h \right) \mathrm{d}\Omega \\ &= \sum_{i=1}^{N_{\text{sub}}} \left(\left(\sum_{j=1}^{4} w_j \boldsymbol{B}_{i,j}^{\text{T}} \boldsymbol{D}_0 \boldsymbol{B}_{i,j} J_{i,j} \right) E_i \right) h A_i \end{aligned} \tag{19}$$

Where the stiffness matrix of the ith subregion $\boldsymbol{K}_i^{\text{ME}}$ can be expressed as following (it is considered that the subregion is full of materials) :

$$\boldsymbol{K}_i^{\text{ME}} = \sum_{j=1}^{4} w_j \boldsymbol{B}_{i,j}^{\text{T}} \boldsymbol{D}_0 \boldsymbol{B}_{i,j} J_{i,j} h A_i \tag{20}$$

By adding the adjoint term to the original objective function (Eq. (18)), the adjoint equation is constructed as follows:

$$h = J + \boldsymbol{\lambda}^{\text{T}} \left(\boldsymbol{f}_{\text{ext}}^{\text{T}} - \boldsymbol{K}\boldsymbol{U} \right) \tag{21}$$

where $\boldsymbol{\lambda}$ is the adjoint vector. Since $\frac{\partial \boldsymbol{f}_{\text{ext}}}{\partial x} = \frac{\partial \boldsymbol{\lambda}}{\partial x} = 0$, the derivation of the Eq. (21) yields

$$\frac{\partial J}{\partial x} = \frac{\partial h}{\partial x} = \boldsymbol{f}_{\text{ext}}^{\text{T}} \frac{\partial \boldsymbol{U}}{\partial x} - \boldsymbol{\lambda}^{\text{T}} \frac{\partial \boldsymbol{K}}{\partial x} \boldsymbol{U} - \boldsymbol{\lambda}^{\text{T}} \boldsymbol{K} \frac{\partial \boldsymbol{U}}{\partial x} = \left(\boldsymbol{f}_{\text{ext}}^{\text{T}} - \boldsymbol{\lambda}^{\text{T}} \boldsymbol{K} \right) \frac{\partial \boldsymbol{U}}{\partial x} - \boldsymbol{\lambda}^{\text{T}} \frac{\partial \boldsymbol{K}}{\partial x} \boldsymbol{U} \tag{22}$$

Let $\boldsymbol{\lambda} = \boldsymbol{U}$, the derivation of the Eq. (21) is expressed as

$$\frac{\partial J}{\partial x} = -\boldsymbol{U}^{\text{T}} \frac{\partial \boldsymbol{K}}{\partial x} \boldsymbol{U} \tag{23}$$

Considering that the design variable in the numerical example is the radius r of the circular hole, the sensitivity equation is further expressed as

$$\begin{aligned} \frac{\partial J}{\partial r} &= -\sum_{ne=1}^{N_A} \boldsymbol{U}_{ne}^{\text{T}} \frac{\partial \boldsymbol{K}_{ne}^{\text{AE}}}{\partial x} \boldsymbol{U}_{ne} \\ &= -\sum_{ne=1}^{N_A} \boldsymbol{U}_{ne}^{\text{T}} \left(\sum_{i=1}^{N_{\text{sub}}} \boldsymbol{K}_i^{\text{ME}} \left(\frac{1}{4} \sum_{k=1}^{4} p \left(H_{i,k} \right)^{p-1} \frac{\partial H_{i,k}}{\partial r} \right) \right) \boldsymbol{U}_{ne} \\ &= -\sum_{ne=1}^{N_A} \boldsymbol{U}_{ne}^{\text{T}} \left(\sum_{i=1}^{N_{\text{sub}}} \boldsymbol{K}_i^{\text{ME}} \left(\frac{1}{4} \sum_{k=1}^{4} p \left(H_{i,k} \right)^{p-1} \delta_{i,k} \frac{\partial \Phi_{i,k}}{\partial r} \right) \right) \boldsymbol{U}_{ne} \end{aligned} \tag{24}$$

where $\boldsymbol{K}_{ne}^{\mathrm{AE}}$ represents the element stiffness matrix of the neth analysis element, $\boldsymbol{U}_{ne}$ means the nodal displacement of the neth analysis element. Eq. (24) is the expression for the analytical sensitivity equation of the structural compliance with respect to the radius r of the circular hole.

Besides, if we consider the derivation of the objective function with respect to the NN's parameters, we need to calculate $\frac{\partial \boldsymbol{\Phi}}{\partial \theta}$, which is not complicated and is usually a derivative of the combined linear function and activation function. This process is not related to TO, and there is a detailed procedure in DL-related monographs [73].

References

[1] M.P. Bendsøe, N. Kikuchi, Generating optimal topologies in structural design using a homogenization method, Comput. Methods Appl. Mech. Engrg. 71 (2) (1988) 197–224, http://dx.doi.org/10.1016/0045-7825(88)90086-2.

[2] M.P. Bendsøe, O. Sigmund, Material interpolation schemes in topology optimization, Arch. Appl. Mech. 69 (9–10) (1999) 635–654, http://dx.doi.org/10.1007/s004190050248.

[3] O. Sigmund, A 99 line topology optimization code written in Matlab, Struct. Multidiscip. Optim. 21 (2) (2001) 120–127, http://dx.doi.org/10.1007/s001580050176.

[4] Y. Xie, G. Steven, A simple evolutionary procedure for structural optimization, Comput. Struct. 49 (5) (1993) 885–896, http://dx.doi.org/10.1016/0045-7949(93)90035-C.

[5] O. Querin, G. Steven, Y. Xie, Evolutionary structural optimisation (ESO) using a bidirectional algorithm, Eng. Comput. 15 (8) (1998) 1031–1048, http://dx.doi.org/10.1108/02644409810244129.

[6] M.Y. Wang, X. Wang, D. Guo, A level set method for structural topology optimization, Comput. Methods Appl. Mech. Engrg. 192 (1–2) (2003) 227–246, http://dx.doi.org/10.1016/S0045-7825(02)00559-5.

[7] G. Allaire, F. Jouve, A.-M. Toader, Structural optimization using sensitivity analysis and a level-set method, J. Comput. Phys. 194 (1) (2004) 363–393, http://dx.doi.org/10.1016/j.jcp.2003.09.032.

[8] X. Guo, W. Zhang, W. Zhong, Doing topology optimization explicitly and geometrically—A new moving morphable components based framework, Trans. ASME, J. Appl. Mech. 81 (8) (2014) 081009, http://dx.doi.org/10.1115/1.4027609.

[9] W. Zhang, W. Yang, J. Zhou, D. Li, X. Guo, Structural topology optimization through explicit boundary evolution, Trans. ASME, J. Appl. Mech. 84 (1) (2016) 011011, http://dx.doi.org/10.1115/1.4034972.

[10] B. Du, W. Yao, Y. Zhao, X. Chen, A moving morphable voids approach for topology optimization with closed B-splines, J. Mech. Des. 141 (8) (2019) 081401, http://dx.doi.org/10.1115/1.4043369.

[11] O. Sigmund, K. Maute, Topology optimization approaches: A comparative review, Struct. Multidiscip. Optim. 48 (6) (2013) 1031–1055, http://dx.doi.org/10.1007/s00158-013-0978-6.

[12] J.D. Deaton, R.V. Grandhi, A survey of structural and multidisciplinary continuum topology optimization: Post 2000, Struct. Multidiscip. Optim. 49 (1) (2014) 1–38, http://dx.doi.org/10.1007/s00158-013-0956-z.

[13] N.P. van Dijk, K. Maute, M. Langelaar, F. van Keulen, Level-set methods for structural topology optimization: A review, Struct. Multidiscip. Optim. 48 (3) (2013) 437–472, http://dx.doi.org/10.1007/s00158-013-0912-y.

[14] D.J. Munk, G.A. Vio, G.P. Steven, Topology and shape optimization methods using evolutionary algorithms: A review, Struct. Multidiscip. Optim. 52 (3) (2015) 613–631, http://dx.doi.org/10.1007/s00158-015-1261-9.

[15] F. Wein, P.D. Dunning, J.A. Norato, A review on feature-mapping methods for structural optimization, Struct. Multidiscip. Optim. 62 (4) (2020) 1597–1638, http://dx.doi.org/10.1007/s00158-020-02649-6.

[16] Y. Liang, G. Cheng, Topology optimization via sequential integer programming and canonical relaxation algorithm, Comput. Methods Appl. Mech. Engrg. 348 (2019) 64–96, http://dx.doi.org/10.1016/j.cma.2018.10.050.

[17] Z. Zhang, Y. Zhao, B. Du, X. Chen, W. Yao, Topology optimization of hyperelastic structures using a modified evolutionary topology optimization method, Struct. Multidiscip. Optim. 62 (6) (2020) 3071–3088, http://dx.doi.org/10.1007/s00158-020-02654-9.

[18] Y. Li, X. Li, M. Li, Y. Zhu, B. Zhu, C. Jiang, Lagrangian–Eulerian multidensity topology optimization with the material point method, Internat. J. Numer. Methods Engrg. 122 (14) (2021) 3400–3424, http://dx.doi.org/10.1002/nme.6668.

[19] Z. Zhang, Y. Li, W. Zhou, X. Chen, W. Yao, Y. Zhao, TONR: An exploration for a novel way combining neural network with topology optimization, Comput. Methods Appl. Mech. Engrg. 386 (2021) 114083, http://dx.doi.org/10.1016/j.cma.2021.114083.

[20] R.V. Woldseth, N. Aage, J.A. Bærentzen, O. Sigmund, On the use of artificial neural networks in topology optimisation, Struct. Multidiscip. Optim. 65 (10) (2022) 294, http://dx.doi.org/10.1007/s00158-022-03347-1.

[21] P. Ramu, P. Thananjayan, E. Acar, G. Bayrak, J.W. Park, I. Lee, A survey of machine learning techniques in structural and multidisciplinary optimization, Struct. Multidiscip. Optim. 65 (9) (2022) 266, http://dx.doi.org/10.1007/s00158-022-03369-9.

[22] I. Sosnovik, I. Oseledets, Neural networks for topology optimization, Russian J. Numer. Anal. Math. Modelling 34 (4) (2019) 215–223, http://dx.doi.org/10.1515/rnam-2019-0018.

[23] J. Luo, Y. Li, W. Zhou, Z. Gong, Z. Zhang, W. Yao, An improved data-driven topology optimization method using feature pyramid networks with physical constraints, CMES Comput. Model. Eng. Sci. 128 (3) (2021) 823–848, http://dx.doi.org/10.32604/cmes.2021.016737.

[24] S. Oh, Y. Jung, S. Kim, I. Lee, N. Kang, Deep generative design: integration of topology optimization and generative models, J. Mech. Des. 141 (11) (2019) 111405, http://dx.doi.org/10.1115/1.4044229.

[25] Y. Yu, T. Hur, J. Jung, I.G. Jang, Deep learning for determining a near-optimal topological design without any iteration, Struct. Multidiscip. Optim. 59 (3) (2019) 787–799, http://dx.doi.org/10.1007/s00158-018-2101-5.

[26] B. Li, C. Huang, X. Li, S. Zheng, J. Hong, Non-iterative structural topology optimization using deep learning, Comput. Aided Des. 115 (2019) 172–180, http://dx.doi.org/10.1016/j.cad.2019.05.038.

[27] S. Lee, H. Kim, Q.X. Lieu, J. Lee, CNN-based image recognition for topology optimization, Knowl.-Based Syst. 198 (2020) 105887, http://dx.doi.org/10.1016/j.knosys.2020.105887.

[28] C. Qian, W. Ye, Accelerating gradient-based topology optimization design with dual-model artificial neural networks, Struct. Multidiscip. Optim. 63 (4) (2021) 1687–1707, http://dx.doi.org/10.1007/s00158-020-02770-6.

[29] L. Xue, J. Liu, G. Wen, H. Wang, Efficient, high-resolution topology optimization method based on convolutional neural networks, Front. Mech. Eng. (2021) http://dx.doi.org/10.1007/s11465-020-0614-2.

[30] M.O. Elingaard, N. Aage, J.A. Bæ rentzen, O. Sigmund, De-homogenization using convolutional neural networks, Comput. Methods Appl. Mech. Engrg. 388 (2022) 114197, http://dx.doi.org/10.1016/j.cma.2021.114197, arXiv:2105.04232.

[31] H. Deng, A.C. To, Topology optimization based on deep representation learning (DRL) for compliance and stress-constrained design, Comput. Mech. 66 (2) (2020) 449–469, http://dx.doi.org/10.1007/s00466-020-01859-5.

[32] A. Chandrasekhar, K. Suresh, TOuNN: Topology optimization using neural networks, Struct. Multidiscip. Optim. 63 (3) (2021) 1135–1149, http://dx.doi.org/10.1007/s00158-020-02748-4.

[33] S. Hoyer, J. Sohl-Dickstein, S. Greydanus, Neural reparameterization improves structural optimization, 2019, arXiv:1909.04240 [Cs, Stat].

[34] D.W. Abueidda, S. Koric, N.A. Sobh, Topology optimization of 2D structures with nonlinearities using deep learning, Comput. Struct. 237 (2020) 106283, http://dx.doi.org/10.1016/j.compstruc.2020.106283.

[35] J. Li, H. Ye, B. Yuan, N. Wei, Cross-resolution topology optimization for geometrical non-linearity by using deep learning, Struct. Multidiscip. Optim. 65 (4) (2022) 133, http://dx.doi.org/10.1007/s00158-022-03231-y.

[36] J.J. Park, P. Florence, J. Straub, R. Newcombe, S. Lovegrove, DeepSDF: Learning continuous signed distance functions for shape representation, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Long Beach, CA, USA, 2019, pp. 165–174, http://dx.doi.org/10.1109/CVPR.2019.00025.

[37] M. Michalkiewicz, J.K. Pontes, D. Jack, M. Baktashmotlagh, A. Eriksson, Implicit surface representations as layers in neural networks, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV, IEEE, Seoul, Korea (South), 2019, pp. 4742–4751, http://dx.doi.org/10.1109/ICCV.2019.00484.

[38] A. Gropp, L. Yariv, N. Haim, M. Atzmon, Y. Lipman, Implicit geometric regularization for learning shapes, in: H. Daumé, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 119, PMLR, 2020, pp. 3789–3799.

[39] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, G. Wetzstein, Implicit neural representations with periodic activation functions, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 7462–7473.

[40] Z. Huang, S. Bai, J.Z. Kolter, (Implicit)2 : Implicit layers for implicit representations, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (Eds.), Advances in Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 9639–9650.

[41] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Systems 2 (4) (1989) 303–314, http://dx.doi.org/10.1007/BF02551274.

[42] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw. 4 (2) (1991) 251–257, http://dx.doi.org/10.1016/0893-6080(91)90009-T.

[43] T. Novello, G. Schardong, L. Schirmer, V. da Silva, H. Lopes, L. Velho, Exploring differential geometry in neural implicits, Comput. Graph.-UK 108 (2022) 49–60, http://dx.doi.org/10.1016/j.cag.2022.09.003.

[44] M. de Ruiter, F. van Keulen, Topology optimization using a topology description function, Struct. Multidiscip. Optim. 26 (6) (2004) 406–416, http://dx.doi.org/10.1007/s00158-003-0375-7.

[45] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, A. Geiger, Occupancy networks: Learning 3D reconstruction in function space, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Long Beach, CA, USA, 2019, pp. 4455–4465, http://dx.doi.org/10.1109/CVPR.2019.00459.

[46] J. Chibane, M.A. mir, G. Pons-Moll, Neural unsigned distance fields for implicit function learning, in: H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 21638–21652.

[47] R.Q. Charles, H. Su, M. Kaichun, L.J. Guibas, PointNet: Deep learning on point sets for 3D classification and segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Honolulu, HI, 2017, pp. 77–85, http://dx.doi.org/10.1109/CVPR.2017.16.

[48] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3D ShapeNets: A deep representation for volumetric shapes, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, CVPR, IEEE, Boston, MA, USA, 2015, pp. 1912–1920, http://dx.doi.org/10.1109/CVPR.2015.7298801.

[49] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, Y.-G. Jiang, Pixel2Mesh: Generating 3D mesh models from single RGB images, in: V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (Eds.), Computer Vision – ECCV 2018, Vol. 11215, Springer International Publishing, Cham, 2018, pp. 55–71, http://dx.doi.org/10.1007/978-3-030-01252-6_4.

[50] P. Wei, Z. Li, X. Li, M.Y. Wang, An 88-line MATLAB code for the parameterized level set method based topology optimization using radial basis functions, Struct. Multidiscip. Optim. 58 (2) (2018) 831–849, http://dx.doi.org/10.1007/s00158-018-1904-8.

[51] W. Zhang, J. Yuan, J. Zhang, X. Guo, A new topology optimization approach based on moving morphable components (MMC) and the ersatz material model, Struct. Multidiscip. Optim. 53 (6) (2016) 1243–1260, http://dx.doi.org/10.1007/s00158-015-1372-3.

[52] K. Svanberg, The method of moving asymptotes—a new method for structural optimization, Internat. J. Numer. Methods Engrg. 24 (2) (1987) 359–373, http://dx.doi.org/10.1002/nme.1620240207.

[53] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, arXiv:1412.6980 [Cs].

[54] D.A. White, M.L. Stowell, D.A. Tortorelli, Toplogical optimization of structures using Fourier representations, Struct. Multidiscip. Optim. 58 (3) (2018) 1205–1220, http://dx.doi.org/10.1007/s00158-018-1962-y.

[55] A. Chandrasekhar, K. Suresh, Approximate length scale filter in topology optimization using Fourier Enhanced Neural Networks, Comput. Aided Des. 150 (2022) 103277, http://dx.doi.org/10.1016/j.cad.2022.103277.

[56] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, R. Ng, Fourier features let networks learn high frequency functions in low dimensional domains, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 7537–7547.

[57] T.H. Nguyen, G.H. Paulino, J. Song, C.H. Le, A computational paradigm for multiresolution topology optimization (MTOP), Struct. Multidiscip. Optim. 41 (4) (2010) 525–539, http://dx.doi.org/10.1007/s00158-009-0443-8.

[58] N. Aage, M. Nobel-Jørgensen, C.S. Andreasen, O. Sigmund, Interactive topology optimization on hand-held devices, Struct. Multidiscip. Optim. 47 (1) (2013) 1–6, http://dx.doi.org/10.1007/s00158-012-0827-z.

[59] C. Liu, Y. Zhu, Z. Sun, D. Li, Z. Du, W. Zhang, X. Guo, An efficient moving morphable component (MMC)-based approach for multi-resolution topology optimization, Struct. Multidiscip. Optim. 58 (6) (2018) 2455–2479, http://dx.doi.org/10.1007/s00158-018-2114-0.

[60] B. Du, Y. Zhao, W. Yao, X. Wang, S. Huo, Multiresolution isogeometric topology optimisation using moving morphable voids, CMES Comput. Model. Eng. Sci. 122 (3) (2020) 1119–1140, http://dx.doi.org/10.32604/cmes.2020.08859.

[61] X. Chen, W. Yao, W. Zhou, Z. Zhang, Y. Li, A general differentiable layout optimization framework for heat transfer problems, SSRN Electron. J. (2022) http://dx.doi.org/10.2139/ssrn.4150423.

[62] A.G. Baydin, B.A. Pearlmutter, A.A. Radul, J.M. Siskind, Automatic differentiation in machine learning: A survey, J. Mach. Learn. Res. 18 (153) (2018) 43.

[63] J.R.R.A. Martins, A. Ning, Engineering Design Optimization, first ed., Cambridge University Press, 2021, http://dx.doi.org/10.1017/9781108980647.

[64] W. Zhu, K. Xu, E. Darve, G.C. Beroza, A general approach to seismic inversion with automatic differentiation, Comput. Geosci. 151 (2021) 104751, http://dx.doi.org/10.1016/j.cageo.2021.104751.

[65] J. Nocedal, S.J. Wright, Numerical Optimization, in: Springer Series in Operations Research, Springer, New York, 1999.

[66] J.-H. Zhu, W.-J. Guo, W.-H. Zhang, T. Liu, Integrated layout and topology optimization design of multi-frame and multi-component fuselage structure systems, Struct. Multidiscip. Optim. 56 (1) (2017) 21–45, http://dx.doi.org/10.1007/s00158-016-1645-5.

[67] M. Minkov, I.A.D. Williamson, L.C. Andreani, D. Gerace, B. Lou, A.Y. Song, T.W. Hughes, S. Fan, Inverse design of photonic crystals through automatic differentiation, ACS Photonics 7 (7) (2020) 1729–1741, http://dx.doi.org/10.1021/acsphotonics.0c00327.

[68] F. Ferrari, O. Sigmund, A new generation 99 line Matlab code for compliance topology optimization and its extension to 3D, Struct. Multidiscip. Optim. 62 (4) (2020) 2211–2228, http://dx.doi.org/10.1007/s00158-020-02629-w.

[69] X. Huang, Y.M. Xie, Evolutionary Topology Optimization of Continuum Structures: Methods and Applications, John Wiley & Sons, Ltd, Chichester, UK, 2010, http://dx.doi.org/10.1002/9780470689486.

[70] Y. Wang, Z. Kang, MATLAB implementations of velocity field level set method for topology optimization: An 80-line code for 2D and a 100-line code for 3D problems, Struct. Multidiscip. Optim. (2021) http://dx.doi.org/10.1007/s00158-021-02958-4.

[71] D. Da, L. Xia, G. Li, X. Huang, Evolutionary topology optimization of continuum structures with smooth boundary representation, Struct. Multidiscip. Optim. 57 (6) (2018) 2143–2159, http://dx.doi.org/10.1007/s00158-017-1846-6.

[72] J. Gao, L. Wang, Z. Luo, L. Gao, IgaTop: An implementation of topology optimization for structures using IGA in MATLAB, Struct. Multidiscip. Optim. 64 (3) (2021) 1669–1700, http://dx.doi.org/10.1007/s00158-021-02858-7.

[73] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, in: Adaptive Computation and Machine Learning, The MIT Press, Cambridge, Massachusetts, 2016.