Computational Mechanics
https://doi.org/10.1007/s00466-023-02333-8

ORIGINAL PAPER

Convolution Hierarchical Deep-Learning Neural Network Tensor Decomposition (C-HiDeNN-TD) for high-resolution topology optimization

Hengyang Li[1] · Stefan Knapik[1] · Yangfan Li[1] · Chanwook Park[1] · Jiachen Guo[2] · Satyajit Mojumder[2] · Ye Lu[1,4] · Wei Chen[1] · Daniel W. Apley[3] · Wing Kam Liu[1]

Received: 30 November 2022 / Accepted: 27 March 2023

Abstract

High-resolution structural topology optimization is extremely challenging due to a large number of degrees of freedom (DoFs). In this work, a Convolution-Hierarchical Deep Learning Neural Network-Tensor Decomposition (C-HiDeNN-TD) framework is introduced and applied to solve the computationally challenging giga-scale topology optimization problem using only a single personal computer (PC). Utilizing the idea of convolution, the C-HiDeNN opens a new avenue for the development of topology optimization theory with arbitrarily high-order smoothness under given DoFs. The convolution involves a set of controllable parameters, including patch size, dilation parameter, and polynomial order. These parameters allow built-in control of the design length-scale, accuracy, and smoothness of solutions. From the point of view of neural networks, increasing the "patch size" is analogous to adding an extra hidden layer with extra neurons, leading to an enhanced approximation capability. Under the separation of variables, the C-HiDeNN-TD can greatly reduce the computational cost of finding a 3D high-resolution topology design by decomposing the ultra-large-scale 3D mechanical problem into several tractable small 1D problems. Orders of magnitude speedups compared to traditional finite element-based topology optimization have been demonstrated through numerical examples. Furthermore, the C-HiDeNN-TD method enables much more efficient concurrent multi-scale topology design than traditional approaches. The proposed framework opens numerous opportunities for high-resolution design, n-scale concurrent design, and structure-lattice-materials design.

Keywords High resolution topology optimization · Multi-scale concurrent design · Reduced-order modeling · 3D printing and metamaterial design · Hierarchical deep-learning neural networks with GPU

1 Introduction

Topology optimization (TO) aims to find an optimal structure under certain constraints to achieve structural design needs [1]. Its applicability has so far been limited to the design of single components or simple structures, owing to the resolution limits of current optimization methods [2]. Recent advancements in manufacturing technologies have led to the need for high-resolution engineering structures that require billions or trillions of degrees of freedom (DoFs) in design, presenting a significant challenge to the design community, particularly when computational resources are limited [3, 4]. "Record resolutions are by now measured in billions, which still require advanced algorithms and supercomputers", as stated in Ole Sigmund's recent paper [5]. As a result, achieving high-resolution design through TO remains a major challenge.

✉ Ye Lu
yelu@umbc.edu

✉ Wing Kam Liu
w-liu@northwestern.edu

[1] Department of Mechanical Engineering, Northwestern University, 2145 Sheridan Rd, Tech B224, Evanston, IL 60208-3109, USA

[2] Theoretical and Applied Mechanics Program, Northwestern University, 2145 Sheridan Rd, Evanston, IL 60208, USA

[3] Department of Industrial Engineering and Management Sciences, Northwestern University, 2145 Sheridan Rd, Evanston, IL 60208, USA

[4] Department of Mechanical Engineering, University of Maryland Baltimore County, 1000 Hilltop Cir, Baltimore, MD 21250, USA

Published online: 03 May 2023

In this paper, we propose a C-HiDeNN-TD-TO framework opening a new avenue for the development of high-resolution topology optimization with the following highlights:

- Conducting large-scale topology optimization with a single PC.
- Providing a new topology optimization technique that exhibits high continuity in both partial differential equation (PDE) solutions and represented structures without requiring increased DoFs.
- Convolution approximation-based reduced-order models lead to highly accurate and efficient solutions.
- The convolution parameters provide built-in filtering for structure and solutions and can improve the stability of topology design such as avoid checkerboard pattern and length-scale control.
- Possessing the potential for extension to an n-scale concurrent design and structure-lattice-materials design, which are prohibitively expensive to achieve with existing methods.

To start, we will first review current approaches in dealing with large-scale topology optimization problems, which include parallel computing, reduced-order methods, and data science approaches [3]. With the development of computational hardware such as Graphics Processing Units (GPUs) and parallel computing technologies such as Message Passing Interface (MPI) and Open Multi-Processing (OpenMP) [6, 7], the utilization of parallel computing to reduce calculation time has become a topic of interest. In the earlier stage, A. Mahdav et al. [8] used domain decomposition to implement parallel computing in TO. Other research works have used multiple processors and GPUs to accelerate the computation (A. Niels et al. [9], Evgrafov et al. [10], Wadbro and Berggren [11]). These papers provide methods to utilize parallel computing hardware to accelerate the calculation time. However, the number of required CPU hours is still extremely large [2, 4]. The limited availability of computing resources mostly hinders the widespread implementation of high-resolution TO in high-performance computing environments.

Another way to reduce the calculation time of TO is reduced-order modeling (ROM). Contrary to the nested approach, where the finite element method (FEM) is used to solve the equilibrium equation during every iteration, reduced-order models can be adopted as surrogate models to approximate the FEM solution with acceptable accuracy while drastically lessening the computational cost [12]. In these models, a reduced subspace/basis typically has a much lower dimension compared to the original solution space. Recently, dimension reduction techniques such as principal component analysis (PCA) and its equivalent, proper orthogonal decomposition (POD), and singular value decomposition (SVD) have been used to solve TO problems. For example, Xia et al. [12] proposed the combination of POD with homogenization techniques for 2-scale concurrent topology optimization. Ferro et al. [13] applied POD to derive the density map, but the basis cannot be updated during the analysis. Xiao et al. [3] used POD to generate the basis on the fly during FE solution and design optimization. Although ROMs show potential for reducing the computational burden, the reduction error of such ROMs may cause some numerical instability, and additional offline computation may persist as a bottleneck [14].

Apart from high-performance computing TO and ROM TO, data science approaches are another way to alleviate the exorbitant computational cost of nested TO methods. In terms of machine learning (ML) techniques, Lei et al. [15] combined supported vector regression (SVR) and K-nearest-neighbors (KNN) methods to solve the layout of an optimized structure under different external loads. Jiang et al. [16] proposed an ML-based parameter tuning strategy to automatically select hyperparameters of the moving morphable component (MMC) scheme. Deep learning (DL) methods have been recently used as an end-to-end model to perform TO. For example, Sosnovik and Oseledets [17] used a convolutional neural network (CNN) to obtain optimal structures based on intermediate structures at each iteration. Yu et al. [18] combined CNN and conditional generative adversarial network (cGAN) to predict the optimal solutions based on 100,000 training data points. Sasaki et al. [19] trained a DL surrogate model to replace expensive FEM computations at each iteration. Deng et al. [20] reported a Self-directed Online Learning Optimization (SOLO) scheme that can automatically generate new training data based on the previously predicted optimal structure and dynamically adapt the new training data for better prediction until convergence. One of the main disadvantages of data science approaches is their dependence on offline databases and their inability to make accurate predictions beyond the range of these training databases (extrapolation). Very recently, Huang et al. proposed a Problem-Independent Machine Learning (PIML)-based TO that has the capability to solve problems with millions of design variables [21]. The database-free machine learning approach is still a developing area. Furthermore, the real ability of data-driven approaches to solve large-scale problems is still unclear [22].

In order to tackle the current challenges in high-performance computing TO (large memory and computation requirements), reduced-order model-based TO (loss in accuracy), and data-driven TO (dependent on an offline database), we propose the Convolution-Hierarchical Deep Learning Neural Network-Tensor Decomposition for Topology Optimization (C-HiDeNN-TD-TO). This is based on a newly developed Convolution-Hierarchical Deep Learning Neural Network (C-HiDeNN) [23] framework, which leverages the partition of unity (POU) of the patch functions with higher order approximations such as meshfree and radial basis functions without increasing the degrees of freedom compared to FEM. This work presents in detail the C-HiDeNN topology optimization methodology and the numerous advantages for large-scale structural design, including efficient high-order smoothness and automatic length-scale filtering with different controllable parameters. Other applications of C-HiDeNN related to graphics processing unit (GPU) computing can be found in [24], which can be combined with this work as a future extension. The proposed C-HiDeNN design framework has the following advantages:

- *Storage and speed* The C-HiDeNN-TD enables high-resolution topology optimization by decomposing the whole 3D/2D space problem into several small 1D problems; it can solve a giga-scale problem using a single PC's hardware, while FEM-TO requires a highly parallel supercomputer environment with multiple nodes. Our method allows thousands of times speedup in terms of CPU hours for a giga-scale problem.
- *Solution accuracy and smoothness* The C-HiDeNN solver improves accuracy and enables higher convergence rates in solving physical problems. The C-HiDeNN-TD high-order continuity is implemented only by changing the convolution shape functions, similar to adding extra layers in the neural network. No extra DoFs are required to smoothen solutions, empowering increased accuracy for a given mesh.
- *Design smoothness* The design variables are interpolated using C-HiDeNN shape functions with high-order continuity. Therefore the design can reach higher smoothness than with the Solid Isotropic Material with Penalization (SIMP) method, where ρ is usually discontinuous through space. Same as for solutions, C-HiDeNN-TD can smoothen designs without introducing more DoFs or refining a mesh.
- *No offline database* The "training" procedure of C-HiDeNN-TD solves the governing PDE as an optimization problem. No offline training dataset is needed. C-HiDeNN-TD-TO can handle arbitrary boundary conditions and design setups without generating an offline database.

We will first briefly review topology optimization to make this paper more readable for a general audience. Depending on different descriptions of the designed geometry, it can be categorized into but not limited to: Solid Isotropic Material with Penalization (SIMP) [1] method, level-set method [25], and moving morphable components (MMC) method [26]. This paper focuses on the SIMP method, which defines the structure with a density variable $\rho(\boldsymbol{x})$. The notation used in TO is defined in Table 1.

Table 1 Notation used for topology optimization

c	Objective function
ρ	Design variable
Π	Virtual work
∇_s	Differential operator
$\boldsymbol{D}$	Constitutive relation
Ω	Design domain
Γ	Design boundary
f	External loading
t	External traction
ρ_{min}	Minimal density
$\boldsymbol{g}(\rho)$	Design constraints

A general optimization problem for the SIMP method [27] can be written as

$$\begin{aligned}
&min \quad c(\rho)\\
&s.t.: \ \delta\Pi = \int_\Omega \nabla_s \boldsymbol{u}^T \boldsymbol{D}(\rho) \nabla_s \delta\boldsymbol{u} d\Omega - \int_\Omega \boldsymbol{f}^T \delta\boldsymbol{u} \rho d\Omega\\
&\qquad - \int_\Gamma \boldsymbol{t}^T \delta\boldsymbol{u} d\Gamma = 0,\\
&\qquad \rho_{min} \le \rho \le 1, \ 0 < \rho_{min} \ll 1,\\
&\qquad V(\rho) \le V^*,\\
&\qquad r(\rho) \ge r_{min},\\
&\qquad \boldsymbol{g}(\rho) \le 0,
\end{aligned} \tag{1}$$

where $c(\rho)$ is the objective function with design variables ρ. The equilibrium equation is written as $\delta\Pi$ in the principle of virtual work form. The design variable constraint is defined as $\rho_{min} \le \rho \le 1$, where $\rho = \rho_{min}$ corresponds to no material, and $\rho = 1$ is solid material. The total volume is $V(\rho)$ with V^* as the volume constraint. The minimal length-scale, which is the minimum allowable size of structural features in the design, is defined as $r(\rho)$ and with constraints $r(\rho) \ge r_{min}$. Other design constraints, such as stress constraints, are defined in $\boldsymbol{g}(\rho) \le 0$. The optimization Eq. (1) can be solved by optimization algorithms such as the Optimum Criterion (OC) method [1] or the Method of Moving Asymptotes (MMA) [10]. For nested TO problems, one solves the equilibrium equations during each optimization iteration. As the resolution increases, solving equilibrium equations will become much more expensive. The developed C-HiDeNN-TD method aims to accelerate topology optimization by reducing the computation with TD. More specifically, we focus on accelerating the solution time of equilibrium equation $\delta\Pi$. The proposed C-HiDeNN-TD-TO framework opens numerous opportunities for designing high-resolution, 3D printable, lightweight, and high-stiffness structures required by the advanced manufacturing industry.

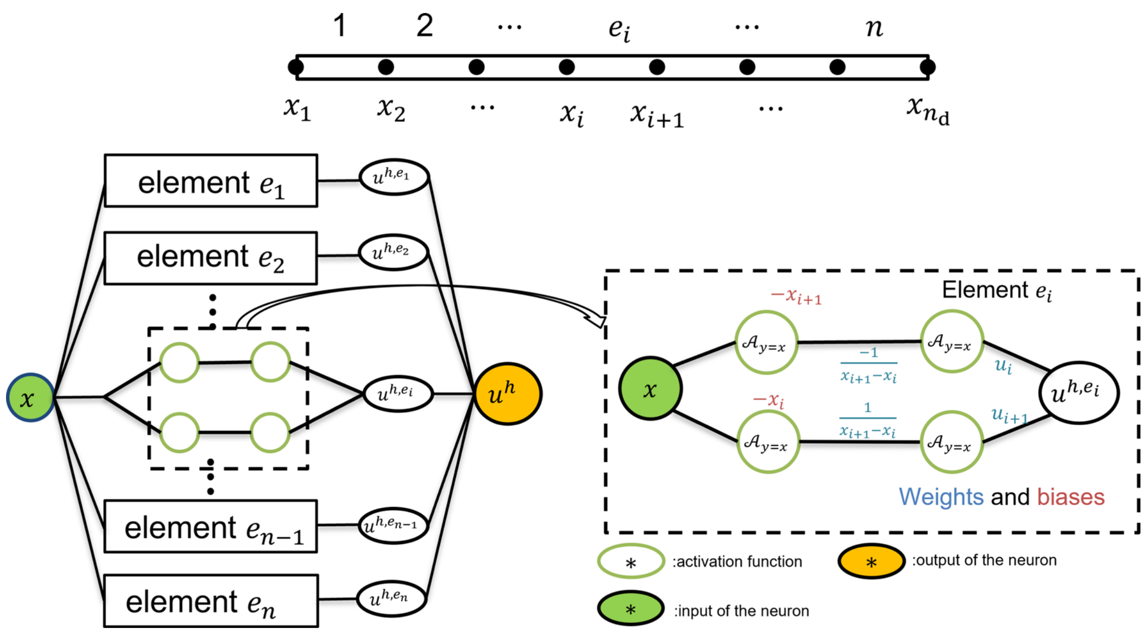


Fig. 1 Structure of the HiDeNN for the displacement in a 1D problem with linear elements [15]. The whole neural network consists of small neural network blocks shown inside the dashed line box

The paper is organized in the following order. In Sect. 2, the C-HiDeNN-TD for topology optimization theory will be introduced. After that, we will show the unique features of convolution shape functions in Sect. 3. In Sect. 4, the influence of the convolution parameters on the optimization results will be studied. Numerical examples of cantilever beam and drone design problems are given in Sect. 5. We will discuss the future directions and conclusions in Sects. 6, 7, respectively.

2 C-HiDeNN-TD for topology optimization

In this section, the C-HiDeNN-TD for topology optimization is introduced. We start by describing the Convolution HiDeNN (C-HiDeNN) theory with a 1D problem to illustrate the basic ideas. Then, we introduce Tensor Decomposition (TD) for topology optimizations to decompose the multi-dimensional (3D) problem into several 1D problems. The whole flowchart for C-HiDeNN-TD-TO is also discussed in this part.

2.1 Convolution-HiDeNN theory

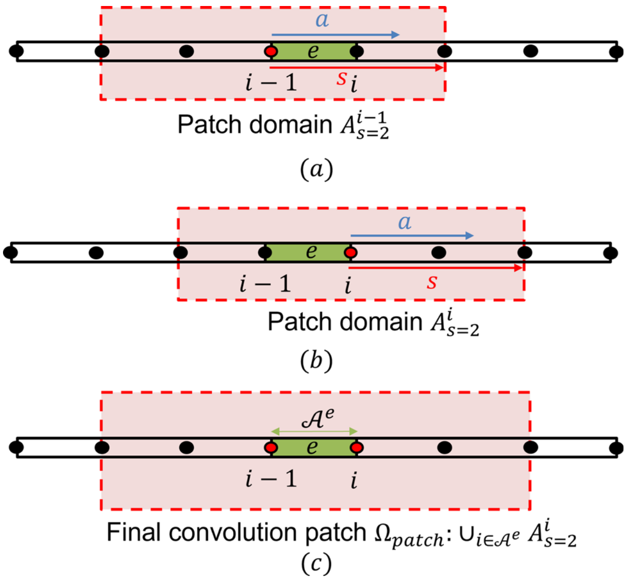
To better understand the C-HiDeNN, we briefly review the idea of HiDeNN. The idea of HiDeNN is to use deep learning neural networks to construct the FE-like shape functions [28] so as to solve partial differential equations for science and engineering problems. For example, for a 1D problem, the HiDeNN shape function on node I can be written as

$$\mathcal{N}_i(x; x_i^*, \boldsymbol{\mathcal{A}}) := \mathcal{F}_i(x; \boldsymbol{w}, \boldsymbol{b}, \boldsymbol{\mathcal{A}}), \tag{2}$$

where $\mathcal{F}_i$ stands for a deep learning neural network with inputs x, weights $\boldsymbol{w}$, biases $\boldsymbol{b}$, and the activation function $\boldsymbol{\mathcal{A}}$. $\mathcal{N}_i$ denotes the shape function for the node at position x_i^*. The detailed structure of the neural network blocks can be found in Zhang et al. [28]. The whole neural network is assembled by adding all the neural network blocks together, as shown in Fig. 1. For a 1D problem, the input for the HiDeNN is the location x on the domain. The output is the displacement at this location

$$u^{\text{HiDeNN}}(x) = \sum_{i=1}^{n} u^{h,e_i}(x) = \sum_{i=1}^{n_d} \mathcal{N}_i(x; x_i^*, \boldsymbol{\mathcal{A}}) u_i, \tag{3}$$

where u^{h,e_i} is the approximated displacement in each element, u_i is the discretized nodal solution of the problem, and u^{HiDeNN} is the approximated solution function. After the construction of the HiDeNN shape function, the differentiation and integration of the shape function can be implemented similarly to the finite element method. The problem can be solved for displacement via optimization algorithms [28]. Several advantages of the HiDeNN method include: (1) it is independent of an offline database; it can solve arbitrary problems with different boundary conditions; (2) it can be extended to r-h-p adaptivity to improve the mesh and increase solution accuracy; (3) it can utilize advanced machine learning hardware such as GPU and TPU. To further improve



Patch domain $A^{i-1}_{s=2}$

(a)

Patch domain $A^{i}_{s=2}$

(b)

Final convolution patch Ω_{patch}: $\cup_{i\in\mathcal{A}^e} A^i_{s=2}$

(c)

Fig. 2 Illustration of the patch domain and final convolution patch for linear element e. The path size is defined as $s = 2$. **a** shows the patch domain associated with node $i-1$. **b** shows the patch domain associated with node i. **c** shows the final convolution patch associated with element e with the relation

the accuracy of the HiDeNN method, Convolution-HiDeNN (C-HiDeNN) is developed [23]. It can be seen as a generalization of the reproducing kernel particle methods (RKPM) [29, 30]. The detailed theoretical foundation and formulation of the C-HiDeNN method can be found in [23].

C-HiDeNN is based on a newly developed convolution approximation, which is formed by two functions: (1) a general polynomial function; (2) a local patch function that operates on a set of nodal points adjacent to the node as a convolution operator. Within an element e, the displacement can be written as:

$$u^{\text{C-HiDeNN}}(\boldsymbol{x}) = \sum_{i\in\mathcal{A}^e} \mathcal{N}_i(x) \sum_{j\in A^i_s} \mathcal{W}^{x_i}_{a,j}(\boldsymbol{x}) u_j = \sum_{k\in\Omega_{\text{patch}}:=\bigcup_{i\in\mathcal{A}^e} A^i_s} \widetilde{N}_k(x)u_k, \qquad (4)$$

where $\mathcal{A}^e$ denotes the support domain of the piecewise polynomial associated with element e, and A^i_s is the local convolution patch domain for each supporting node i in the polynomial function. The relation and definition of the support domain, patch domain, and final convolution patch can be found in Fig. 2. $\mathcal{N}_i(x)$ is the polynomial function associated with node i, and $\mathcal{W}^{x_i}_{a,j}(x)$ is the local convolution patch function that considers the contribution from the neighborhood (A^i_s) of node i. The convolution patch function is designed to have a similar role as the kernel function in a convolution neural network. Details of constructing $\mathcal{N}_i(x)$ and $\mathcal{W}^{x_i}_{a,j}(x)$ can be found in appendix B and papers [23, 24]. It should be noted that there are three key controlling parameters in the convolution patch functions: (1) the patch size s defining the size of the patch domain; (2) the polynomial order p; (3) the dilation parameter a. These parameters dictate the smoothness of the approximated solutions and convergence rates, as discussed in [23, 24]. We will study their influence on TO design in this work.

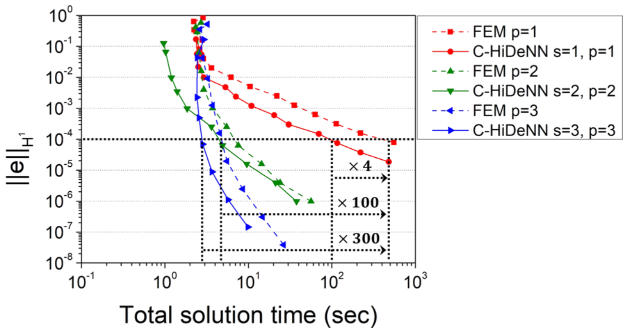


Fig. 3 H1 norm error versus computation time plot for a 1D Poisson problem. FEM uses dashed lines, while C-HiDeNN uses solid lines. Curves with the same polynomial order p have the same color. Both FEM and C-HiDeNN are solved via a Jacobi-preconditioned conjugate gradient iterative solver implemented on a small GPU (NVIDIA GeForce GTX 1650, 4GB memory). For more information on the GPU implementation, readers may refer to [24] or GitHub: https://github.com/hachanook/C-HiDeNN_tutorial

The convolution shape function $\widetilde{N}_k(x)u_k$ only changes the formulation of the original shape function $N_k(x)u_k$ without adding extra DoFs. This unique feature can increase the smoothness and accuracy of the approximation while avoiding extra DoFs, resulting in faster computation time for a given accuracy. Figure 3 shows the speedup of C-HiDeNN compared to higher-order FEM for a given level of accuracy. A 1D Poisson's equation is solved, and H1 norm error estimator is computed (details of the problem can be found in appendix C). Overall, when the polynomial order p is the same, C-HiDeNN is faster than FEM for a given error. Note that higher-order FEM uses higher-order elements while both linear FEM ($p = 1$) and C-HiDeNN use linear elements. That is, quadratic FEM ($p = 2$) and cubic FEM ($p = 3$) use 3-node and 4-node elements, respectively, while linear FEM ($p = 1$) and C-HiDeNN use 2-node elements in a 1D problem. Considering this, C-HiDeNN can achieve accurate solutions orders of magnitude faster than linear FEM ($p = 1$). For example, as illustrated in Fig. 3, C-HiDeNNs with ($s = 1$, $p = 1$), ($s = 2$, $p = 2$), and ($s = 3$, $p = 3$) are around 4 times, 100 times, and 300 times faster than linear FEM ($p = 1$) to achieve 10^{-4} H1 norm error. This clearly demonstrates the improved efficiency of C-HiDeNN.

The structure of C-HiDeNN is shown in Fig. 4. Compared to HiDeNN, there is an extra layer of convolution neurons that perform the convolution operations. The accuracy and

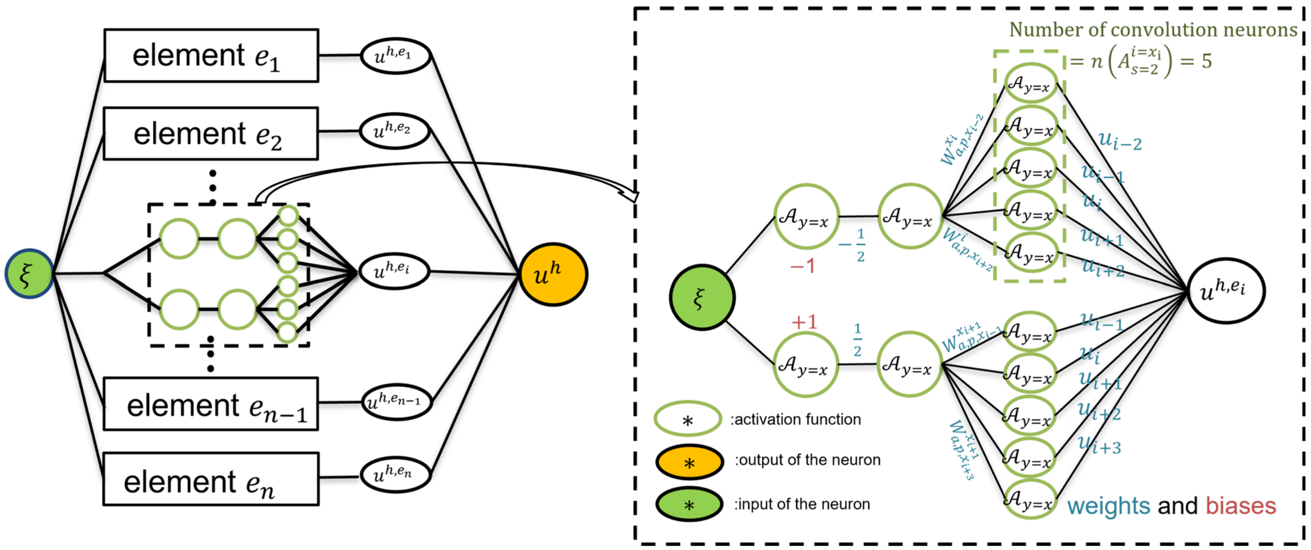


Fig. 4 Structure of the Convolution HiDeNN (C-HiDeNN) for the displacement in a 1D problem with linear elements [24]. The convolution neural network block is shown in the dashed line box. C-HiDeNN only changes the structure of neural network blocks without increasing the number of DoFs of displacement

convergence study of C-HiDeNN can also be found in [23, 24]. To simplify numerical implementation, Eq. (4) can be converted into the natural coordinate system:

$$\begin{aligned} u^{\text{C-HiDeNN}}(\boldsymbol{\xi}) &= \sum_{i\in\mathcal{A}^e} \mathcal{N}_i(\xi) \sum_{j\in A_s^i} \mathcal{W}_{a,j}^{\xi_i}(\boldsymbol{\xi}) u_j \\ &= \sum_{k\in\Omega_{\text{patch}}:=\bigcup_{i\in\mathcal{A}^e} A_s^i} \widetilde{N}_k(\xi) u_k, \end{aligned} \tag{5}$$

where $\boldsymbol{\xi}$ denotes general natural (or parametric) coordinates.

As discussed in [23, 24], the benefits of C-HiDeNN include:

- C-HiDeNN significantly improves accuracy and convergence rates only by changing the formulation of approximation functions. The DoFs in discretized systems remain unchanged compared to FEM with the same node/mesh used;
- C-HiDeNN can achieve arbitrary orders of smoothness and convergence rates by adjusting the different controlling parameters in the convolution patch function;
- C-HiDeNN provides a highly scalable framework that can be accelerated using deep learning libraries and parallel computing hardware such as JAX [24] and TPU [31].

The focus of this paper is the implementation of C-HiDeNN for topology optimization, which is discussed in the following sections.

2.2 C-HiDeNN-TD for topology optimization

C-HiDeNN-TD is a reduced-order modeling technique of the C-HiDeNN methods. For structural TO, C-HiDeNN-TD can serve the same role as FEM for solving partial differential equations and enforcing physics constraints. As discussed in the introduction, one of the major difficulties in solving a large number of simultaneous equations using numerical methods is the computational cost of large matrix inversion. To this end, the Proper Generalized Decomposition (PGD) and Tensor Decomposition (TD) based reduced-order modeling techniques have been developed [32–36]. The basic idea is to assume the physical quantities of the problem can be written into a form with a separation of variables. For example, the displacement $\boldsymbol{u} = [u, v, w]$ for a 3D design problem can be written as

$$\begin{aligned} u\,(x, y, z) &= \sum_{q=1}^{Q} u_x^q\,(x)\,u_y^q\,(y)\,u_z^q\,(z), \\ v\,(x, y, z) &= \sum_{q=1}^{Q} v_x^q\,(x)\,v_y^q\,(y)\,v_z^q\,(z), \\ w\,(x, y, z) &= \sum_{q=1}^{Q} w_x^q\,(x)\,w_y^q\,(y)\,w_z^q\,(z), \end{aligned} \tag{6}$$

where the index q denotes mode number q out of Q total modes, u_x, u_y, u_z are 1D functions to be computed, and similarly for v_x, v_y, v_z and w_x, w_y, w_z. Similarly, the design variable ρ can also be decomposed into

$$\rho\,(x, y, z) = \sum_{k=1}^{K} \rho_x^k\,(x)\,\rho_y^k\,(y)\,\rho_z^k\,(z). \tag{7}$$

To distinguish the modes of u and ρ, the index of mode in ρ is defined as k, and the maximum number of modes is K. Assuming the density distribution is updated during each design iteration, this decomposition can be done repeatedly by the Higher-Order PGD (HOPGD) method [37–39]. Consequently, the original 3D problem is converted to finding the corresponding 1D modes of those u_x, u_y, u_z, v_x, v_y, v_z, w_x, w_y, w_z, and ρ_x, ρ_y, ρ_z. Therefore, the computational complexity is significantly reduced. The equilibrium equation in the form of the principle of virtual work can be written as

$$\begin{aligned}\delta\Pi &= \int_{\Omega} \nabla_s \delta \boldsymbol{u}^T (\boldsymbol{D}\rho) \nabla_s \boldsymbol{u} d\Omega - \int_{\Omega} \delta \boldsymbol{u}^T \boldsymbol{f} \rho d\Omega \\ &\quad - \int_{\Gamma} \delta \boldsymbol{u}^T \boldsymbol{t} d\Gamma = 0, \end{aligned} \tag{8}$$

where ∇_s is the symmetric gradient operator, and $\nabla_s \boldsymbol{u}$ is the strain in Voigt notation. Considering the separation of variables, the principle of virtual work becomes

$$For\ each\ mode\ q,\ \begin{cases} \delta\Pi_{,\delta u_x^q} = 0, & \forall \delta u_x^q \\ \delta\Pi_{,\delta u_y^q} = 0, & \forall \delta u_y^q \\ \delta\Pi_{,\delta u_z^q} = 0, & \forall \delta u_z^q \\ \delta\Pi_{,\delta v_x^q} = 0, & \forall \delta v_x^q \\ \delta\Pi_{,\delta v_y^q} = 0, & \forall \delta v_y^q \\ \delta\Pi_{,\delta v_z^q} = 0, & \forall \delta v_z^q \\ \delta\Pi_{,\delta w_x^q} = 0, & \forall \delta w_x^q \\ \delta\Pi_{,\delta w_y^q} = 0, & \forall \delta w_y^q \\ \delta\Pi_{,\delta w_z^q} = 0, & \forall \delta w_z^q \end{cases}, \tag{9}$$

where $\delta\Pi_{,\delta u}$ denotes the partial derivative with respect to δu. From the Eq. (9), the whole 3D problem has been decomposed into several 1D problems. There are many ways to solve the above variational equations. In this paper, we applied the C-HiDeNN approximation for each separated function [23]. The element-wise approximations of solutions then read

$$For\ each\ mode\ q,\ \begin{cases} u_x^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(x) u^q_{x,k} \\ u_y^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(y) u^q_{y,k} \\ u_z^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(z) u^q_{z,k} \\ v_x^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(x) v^q_{x,k} \\ v_y^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(y) v^q_{y,k} \\ v_z^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(z) v^q_{z,k} \\ w_x^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(x) w^q_{x,k} \\ w_y^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(y) w^q_{y,k} \\ w_z^{q,e} = \sum_{k\in A_s} \widetilde{N}_k(z) w^q_{z,k} \end{cases}, \tag{10}$$

where $u^q_{x,k}$, $u^q_{y,k}$, $u^q_{z,k}$, $v^q_{x,k}$, $v^q_{y,k}$, $v^q_{z,k}$, $w^q_{x,k}$, $w^q_{y,k}$, $w^q_{z,k}$ are the nodal solution values of each separated function. $\widetilde{N}$ is the convolution shape function. Let $\boldsymbol{u}_x^q$, $\boldsymbol{u}_y^q$, $\boldsymbol{u}_z^q$, $\boldsymbol{v}_x^q$, $\boldsymbol{v}_y^q$, $\boldsymbol{v}_z^q$, $\boldsymbol{w}_x^q$, $\boldsymbol{w}_y^q$, $\boldsymbol{w}_z^q$ denote the global nodal solution vector for each function, then Eq. (9) can be discretized and rearranged to become

$$For\ each\ mode\ q,\ \begin{cases} \tilde{\boldsymbol{K}}_{u_x^q} \boldsymbol{u}_x^q = \boldsymbol{f}_{u_x^q} \\ \tilde{\boldsymbol{K}}_{u_y^q} \boldsymbol{u}_y^q = \boldsymbol{f}_{u_y^q} \\ \tilde{\boldsymbol{K}}_{u_z^q} \boldsymbol{u}_z^q = \boldsymbol{f}_{u_z^q} \\ \tilde{\boldsymbol{K}}_{v_x^q} \boldsymbol{v}_x^q = \boldsymbol{f}_{v_x^q} \\ \tilde{\boldsymbol{K}}_{v_y^q} \boldsymbol{v}_y^q = \boldsymbol{f}_{v_y^q} \\ \tilde{\boldsymbol{K}}_{v_z^q} \boldsymbol{v}_z^q = \boldsymbol{f}_{v_z^q} \\ \tilde{\boldsymbol{K}}_{w_x^q} \boldsymbol{w}_x^q = \boldsymbol{f}_{w_x^q} \\ \tilde{\boldsymbol{K}}_{w_y^q} \boldsymbol{w}_y^q = \boldsymbol{f}_{w_y^q} \\ \tilde{\boldsymbol{K}}_{w_z^q} \boldsymbol{w}_z^q = \boldsymbol{f}_{w_z^q} \end{cases}, \tag{11}$$

Finally, the solution of Eq. (8) can be approximated by solving the above set of equations. The directional stiffness matrix of each mode for the separated displacement function is defined as $\tilde{\boldsymbol{K}}$ with the integrated external force $\boldsymbol{f}$. Details of the derivation from Eqs. (9) to (11) can be found in the appendix A. Equation (11) can be solved sequentially by looping over each mode of $\boldsymbol{u}$. The number of modes is determined by setting a convergence criterion. After convergence, all modes can be optimized again to increase the accuracy further, as discussed in [35].

It is evident that for each iteration in Eq. (11), the dimension of the directional stiffness matrix is only determined by the number of DoFs in each direction. This results in a drastic reduction in the matrix dimension compared to the traditional FEM. For a problem with a mesh size of $1000 \times 1000 \times 1000$ using linear elements, the DoFs of the problem including design density on each node is $(1001 \times 1001 \times 1001 \times 4) \rightarrow (4, 012, 012, 004)$. Solving such a large number of equations is extremely expensive. However, in our work, the number of the DoFs for every one dimensional problem would be only 4004. Ultimately, this can lead to orders of magnitude reduction in the overall computational cost.

With the new C-HiDeNN-TD solver, the TO flow chart can be redesigned as in Fig. 5. The overall procedure is similar to the traditional nested TO. The main difference comes from the solver part. In our design framework, the C-HiDeNN-TD is used for solving the equilibrium Eq. (8), whereas traditional TO usually uses FEM. The initial design variable is set to be uniformly distributed and satisfies the volume fraction constraint in the whole design domain. The design variable ρ is decomposed into three directions for a 3D problem or two directions for a 2D problem, as defined in Eq. (7). In this paper, we use HOPGD to perform the decomposition of the whole density field [37]. After the decomposition, the decomposed density is sent to the C-HiDeNN-TD solver with the

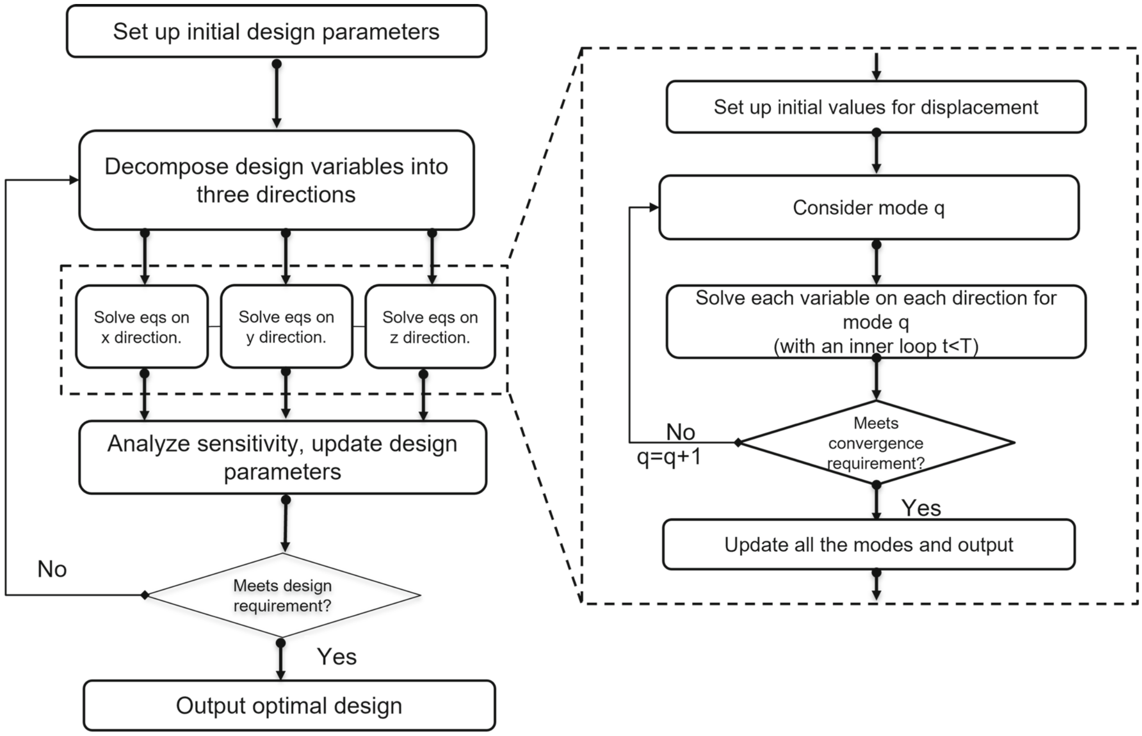

Set up initial design parameters
Decompose design variables into three directions
Solve eqs on x direction.
Solve eqs on y direction.
Solve eqs on z direction.
Analyze sensitivity, update design parameters
Meets design requirement?
No
Yes
Output optimal design
Set up initial values for displacement
Consider mode q
Solve each variable on each direction for mode q (with an inner loop t<T)
Meets convergence requirement?
No
q=q+1
Yes
Update all the modes and output

Fig. 5 Flow chart of C-HiDeNN-TD for topology optimization

convolution shape function defined in Eq. (10). The design variables are optimized using the OC method [1]. With each design variable update, a convergence criterion is tested to determine whether or not the design has converged.

We remark that the overall design framework is flexible and can be coupled with traditional FEM or other C-HiDeNN methods in a hybrid way. For example, FEM can be used during the first few iterations with a coarse mesh to get some initial design before ultimately switching to C-HiDeNN-TD with a higher resolution to achieve finer structures that perform better. This kind of hybrid strategy will be further investigated in our future work.

3 Convolution interpolation for built-in density filtering and higher-order density smoothness without extra DoFs

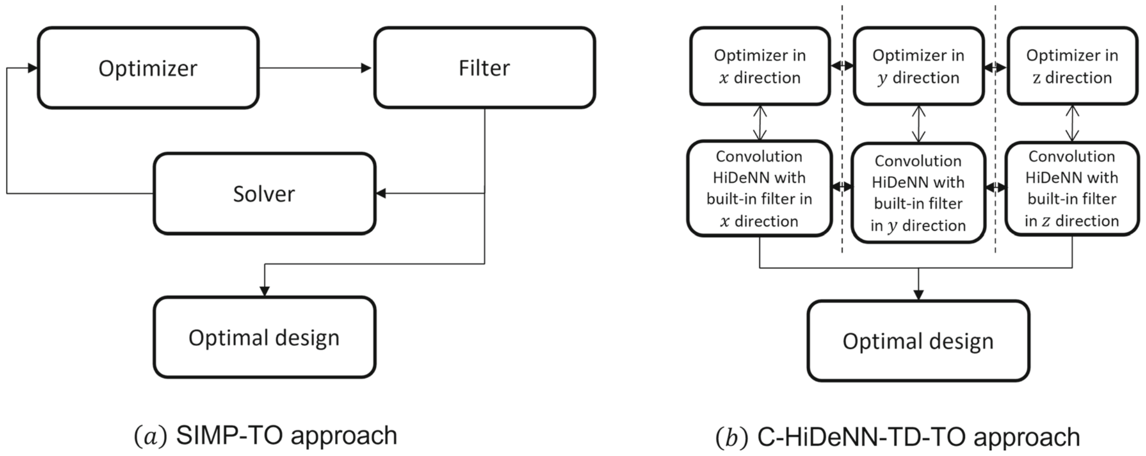
This section studies the effect of C-HiDeNN approximation on the optimized density distribution. In TO with the SIMP approach, the design variable, i.e., density field, is considered constant inside each element of the background mesh. Let Ω_e denote an element region. The SIMP-based density description is defined as

$$\rho^{\text{SIMP}}(\boldsymbol{\xi}) = \rho_e, \quad \forall \boldsymbol{\xi} \in \Omega_e. \tag{12}$$

Consequently, the SIMP approach can lead to discontinuous density distributions and checkerboard patterns across multiple elements. Using higher-order elements for a smooth description of the density field is one of the solutions to resolve this issue, as suggested by [40, 41]. However, using higher-order finite elements can result in excessive increases in the computational cost of TO due to the additional DoFs in both the physical equations and the design variables, as indicated by Sigmund et al. [42]. Another way to mitigate checkerboarding and enforce length-scale control is the filtering technique [43]. Several filter methods have been developed, including sensitivity filter, density filter, Heaviside filter, and averaging filter [44–46]. However, the extra filtering stage increases the computational burden in the design loop, especially for high-resolution problems where the filter may be applied over millions or billions of design DoFs.

From the above perspectives, the C-HiDeNN approximation is advantageous in providing a built-in filter to improve the design smoothness without increasing the overall number of design variable DoFs and DoFs of the numerically discretized physical system. Therefore, the proposed C-HiDeNN-TD-TO framework is expected to lead to a higher-order smooth density description compared to SIMP if the same background mesh (resolution) is used.

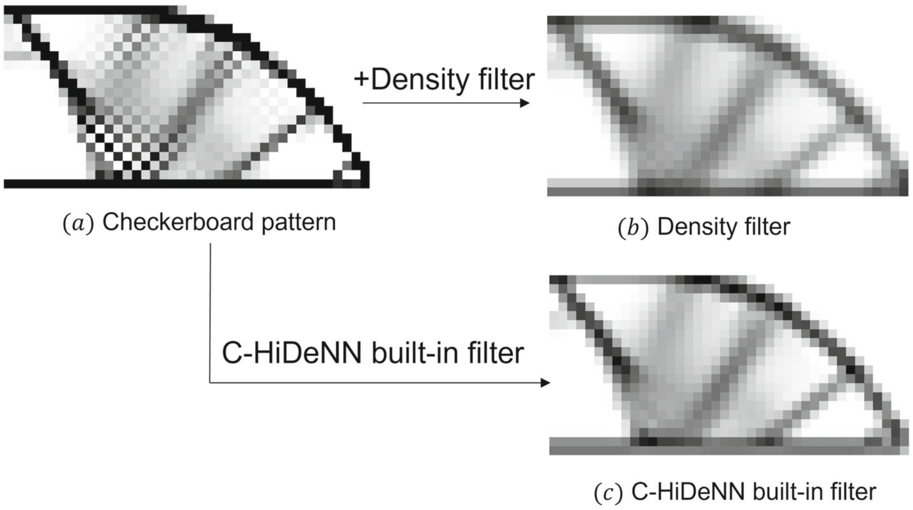
The C-HiDeNN approximated density field can be defined using the nodal values of the background mesh. Similar to


Optimizer
Filter
Solver
Optimal design
Optimizer in x direction
Optimizer in y direction
Optimizer in z direction
Convolution HiDeNN with built-in filter in x direction
Convolution HiDeNN with built-in filter in y direction
Convolution HiDeNN with built-in filter in z direction
Optimal design

(*a*) SIMP-TO approach

(*b*) C-HiDeNN-TD-TO approach

Fig. 6 **a** SIMP-TO approach, an extra filter is needed to control the length-scale and avoid a checkerboard pattern. **b** C-HiDeNN-TD-TO approach, no extra filter is needed. The length-scale control can be done by choosing the patch size s, polynomial order p, and dilation parameter a



(*a*) Checkerboard pattern

(*b*) Density filter

(*c*) C-HiDeNN built-in filter

Fig. 7 **a** Checkerboard pattern. **b** A density filter is applied to the density shown in **a** as traditional TO approaches. The checkerboard pattern is reduced. **c** The C-HiDeNN built-in filter is applied to **a**. It has a similar effect as the density filter but without the extra filtering step

the displacement field, the nodal density can be denoted by ρ_j, and the approximated density then reads:

$$\begin{aligned}\rho^{\text{C-HiDeNN}}(\boldsymbol{\xi}) &= \sum_{i\in\mathcal{A}^e}\mathcal{N}_i(\xi)\sum_{j\in A_s^i}\mathcal{W}^{\xi_i}_{a,j}(\boldsymbol{\xi})\rho_j\\ &= \sum_{k\in\Omega_{\text{patch}}:=\bigcup_{i\in\mathcal{A}^e}A_s^i}\widetilde{N}_k(\xi)\rho_k,\qquad \forall\boldsymbol{\xi}\in\Omega_e,\end{aligned}\tag{13}$$

where the support domain $\mathcal{A}^e$ and local convolution patch domain A_a^i are defined the same as for the displacement approximation. The polynomial function associated with node i is $\mathcal{N}_i(\xi)$, and the local convolution patch function is $\mathcal{W}^{\xi_i}_{a,j}(\xi)$. In a general sense, they can be different from the displacement approximation.

The dilation parameter a, the polynomial order p, and the patch size s serve as built-in filtering parameters to control the minimum design length-scale and remove checkerboard patterns. Figure 6 shows the difference between the SIMP-TO approach and the C-HiDeNN-TD-TO approach. Unlike SIMP-TO, the density is interpolated using C-HiDeNN-TD-TO shape functions and the convolution operation serves as a filter on the density field, which will automatically prevent checkerboarding and incur length-scale control. One example is shown in Fig. 7. To illustrate the idea, we generate a density field with a checkerboard pattern, as shown in Fig. 7. This is commonly seen if the filter is not applied during the design iteration using SIMP. We applied the density filter and the C-HiDeNN built-in filter on the same density field. The minimum length-scale of the density filter is set to 2. For a fair comparison, the patch size s is set to 2 with a dilation parameter $a = 2$ and polynomial order $p = 2$. Figure 7 shows that the C-HiDeNN built-in filter has the capability to remove a checkerboard pattern similar to a dedicated density filter. It is noted that for some 3D examples with a volume fraction less than 0.3, the C-HiDeNN-TD-TO built-in filter may be insuf-

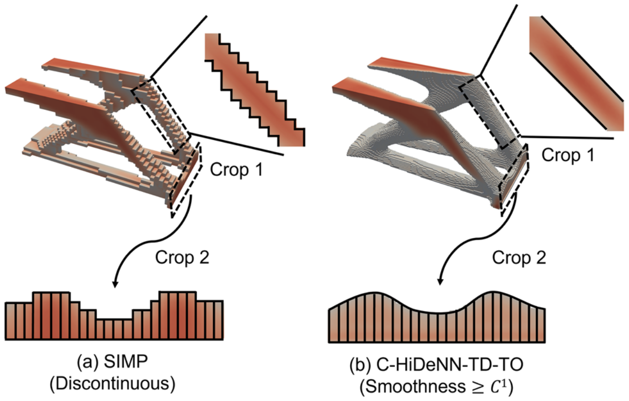


Fig. 8 **a** Topology optimization SIMP density plot. **b** C-HiDeNN-TD-TO density plot with 27 interpolation points in each element. With the same design representation mesh, C-HiDeNN-TD-TO has a continuity $\geq C^1$ and is much smoother and more 3D printable with desirable smoothness

ficient to remove all isolated substructures. More studies are needed, including designing a more stable built-in filter and finding optimal parameter sets.

The other feature is that the convolution interpolation does not increase the DoFs in the design variables but improves the smoothness of interpolation. One example to compare the difference between the SIMP approach and the C-HiDeNN-TD-TO approach is shown in Fig. 8, which depicts a cantilever beam design example with a mesh size of 100 × 50 × 50 using SIMP and C-HiDeNN descriptions. As shown in Fig. 8a, the density in the SIMP approach is discontinuous in the design domain. For C-HiDeNN-TD-TO, the density is smooth, with a desired higher-order continuity defined by the C-HiDeNN shape function. To represent the C-HiDeNN-based continuous density function in the 3D domain, we select a few interpolation points in each element. Figure 8b shows the plot of C-HiDeNN-TD-TO with 27 interpolation points in each element. It can be seen that, without adding any additional DoFs in the design, the C-HiDeNN-TD-TO gives a smoother design structure than the SIMP-TO. With more interpolation points added, the plot can be a very smooth structure with details on the surface. It should be noted that although the C-HiDeNN approximation increases the smoothness of the design, it can not lead to the same results as high-resolution design since higher resolutions can result in greater design flexibility creating finer structures with better design objective values.

4 Parametric study of the convolution parameters on the final design

In this section, we study the effect of the convolution parameters on the final design structures. As mentioned earlier, there are three parameters in the convolution shape functions: the patch size s defining the size of the patch domain; the polynomial order p defining the polynomial order of $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$; the dilation parameter a in $\mathcal{W}_{a,j}^{\boldsymbol{\xi}_i}(\boldsymbol{\xi})$ controlling the size of the window function, as shown in Fig. 9b.

As shown in Fig. 9, the cantilever beam is used with the left side clamped and a uniform loading on the right side. The background design mesh size is 100 × 50 × 50. The design formulations are defined as:

$$\begin{aligned} min\ c(\rho) &= \int_\Omega \boldsymbol{f}^T \boldsymbol{u} d\Omega + \int_\Gamma \boldsymbol{t}^T \boldsymbol{u} d\Gamma \\ s.t.: \delta\Pi &= \int_\Omega \nabla_s \boldsymbol{u}^T \boldsymbol{D}(\rho) \nabla_s \delta \boldsymbol{u} d\Omega - \int_\Omega \boldsymbol{f}^T \delta \boldsymbol{u} \rho d\Omega \\ &\quad - \int_\Gamma \boldsymbol{t}^T \delta \boldsymbol{u} d\Gamma = 0, \\ &\quad \rho_{min} \leq \rho \leq 1,\ 0 < \rho_{min} \ll 1, \\ &\quad V(\rho) \leq V^*, \\ &\quad r(\rho) \geq r_{min}. \end{aligned} \tag{14}$$

The objective function is system compliance as defined in Eq. (14). The volume fraction V^* is set to 0.5 for the design region. The convolution parameters are defined as follows. The patch size s is defined as the size of the extended region outside the element (bold green region). The dilation parameter a is defined as the tuning parameter of the window function. The polynomial order of the convolution patch function is defined as p. In this study, we fix the patch size s but vary the polynomial order p and the dilation parameter a.

Figure 10 shows the different results related to the polynomial order p. The patch size is set as $s = 4$. To better understand the synergism of C-HiDeNN with the tensor decomposition, we set Fig. 10a with a very small and positive a, which produces the Tensor Decomposition-TO (TD-TO) result. From these results, higher order p does give a better design (lower objective function) with more support structures, especially when compared to $p = 1$ (the case of TD-TO). The reason is that the higher-order approximation gives a smoother and more accurate solution of the physics field and significantly improves the sensitivity of the density to displacement. Finally, the optimal design converges with $p = 3$ in this case.

Figure 11 shows another example to test the dilation parameter a. The polynomial order and patch size are $p = 3, s = 4$. From the figure, it is obvious that a large dilation parameter gives a lower objective function. The optimal design (objective function) converges with $a = 4$. Furthermore, the dilation parameter seems to contribute to the length-scale filtering, as a larger a might remove more small support structures in the final design. This can be seen inside the black circles in the figure.

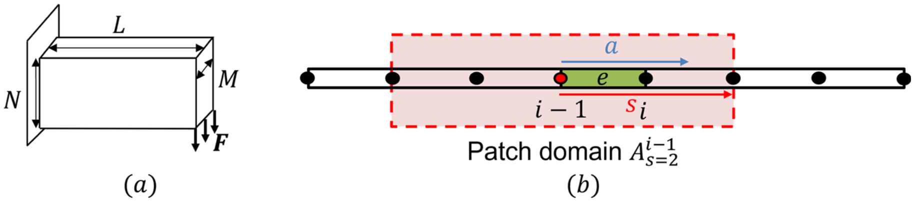
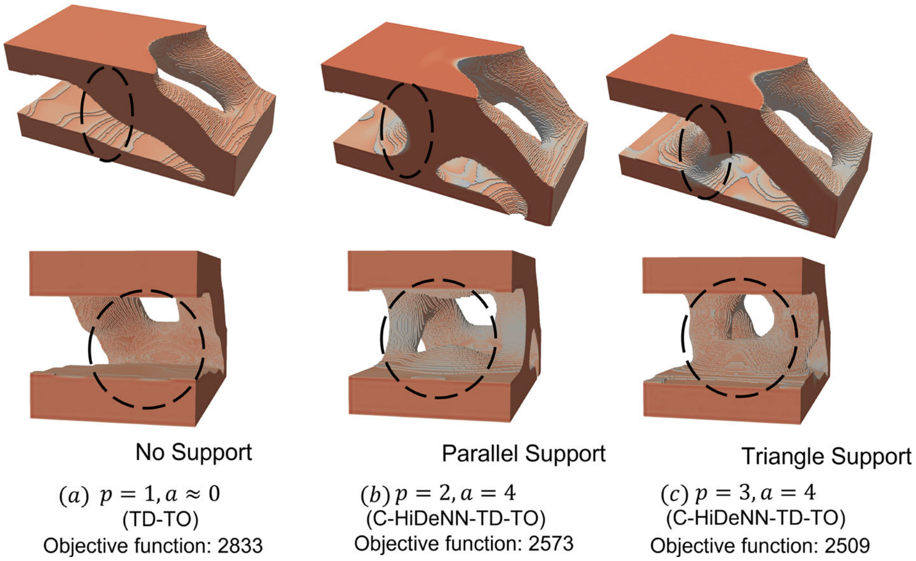


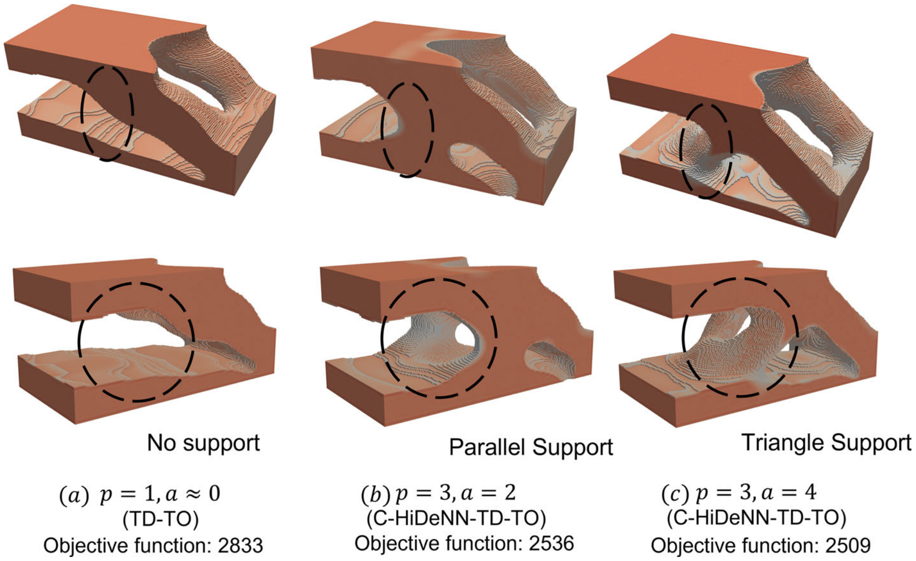
Fig. 9 **a** Cantilever beam example with a uniform loading on the right side and a clamped boundary condition on the left side. The size is set to $M = N = 50$ and $L = 100$. **b** Illustration of the patch size s, and dilation parameter a


No Support
$(a)\ p = 1, a \approx 0$
(TD-TO)
Objective function: 2833

Parallel Support
$(b)\, p = 2, a = 4$
(C-HiDeNN-TD-TO)
Objective function: 2573

Triangle Support
$(c)\, p = 3, a = 4$
(C-HiDeNN-TD-TO)
Objective function: 2509

Fig. 10 Convolution polynomial order study. The patch size is set to $s = 4$. Example (a) uses a small dilation parameter to degenerate into the TD-TO method. From the comparison, polynomials of higher order produce a better objective value with more support structures because of the smoothness of the corresponding solutions


No support
$(a)\ p = 1, a \approx 0$
(TD-TO)
Objective function: 2833

Parallel Support
$(b)\, p = 3, a = 2$
(C-HiDeNN-TD-TO)
Objective function: 2536

Triangle Support
$(c)\, p = 3, a = 4$
(C-HiDeNN-TD-TO)
Objective function: 2509

Fig. 11 Convolution polynomial order study. The patch size is set to $s = 4$. Example (a) uses a small dilation parameter to degenerate into the TD-TO method. From the comparison, a larger dilation parameter produces a better design and contributes to the design length-scale control

From the above two comparisons, we can see that the greater the polynomial order, p, and the larger the dilation parameter, a, the more accurate the solved displacements are, leading to better topology-optimized designs. However, it should be noted that there exists an optimal pair of p and a for each problem. Using larger numbers for p and a can result in a higher computational cost. The balance between accuracy and cost should be considered when choosing the optimal convolution parameters. The effect of this dilation on the length-scale control and the relationship to physical quantities in the TO can be further investigated in the future.

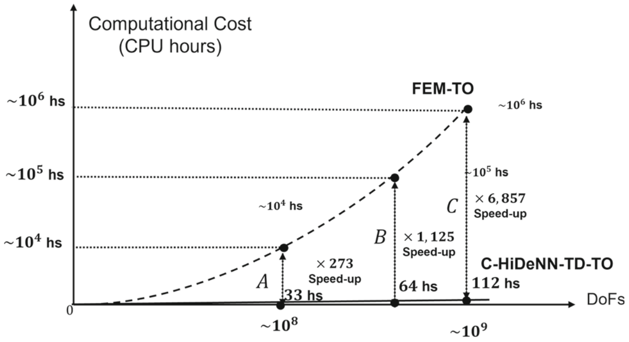


Fig. 12 DoFs vs computational cost between FEM-TO and C-HiDeNN-TD-TO. C-HiDeNN-TD-TO can solve a giga-scale problem only using hundreds of CPU hours with an affordable personal computer

5 Numerical examples for high-resolution design

5.1 Example 1: high-resolution cantilever beam

The first example is still the cantilever example, as illustrated in Fig. 9. This time, the efficiency of C-HiDeNN-TD-TO is studied with an increasing number of DoFs (higher resolution). In order to make a comparison to the example in [4], the volume fraction is set to be 0.12, and the minimal length-scale parameter r_{min} in Eq. (1) is set to be 0.1 of the short edge. The objective function and optimization formulations are same as Eq. (14). Three different design resolutions are considered at this time: 480 × 240 × 240, 960 × 480 × 480, and 1440 × 720 × 720. The dilation parameter a is set to 4 with the polynomial order $p = 3$ and a patch size $s = 4$.

The computational cost with different DoFs for FEM-TO and the proposed C-HiDeNN-TD-TO are shown in Fig. 12. The number of FEM DoFs also considering the density variable on each node. The CPU hours in FEM-TO with the SIMP method are estimated according to work done by Aage et al. [2, 4] using supercomputer resources. For the FEM-TO method, the CPU time increases from 10^4 hours to 10^6 hours as the DoFs increase from 10^8 to 10^9. However, for C-HiDeNN-TD-TO, the CPU time only increases from 33 to 112 h, with a 6857 times speedup for the larger mesh when compared to the FEM-TO method. Table 2 lists the detailed numbers of the three cases, including the number of equations, the number of modes in the C-HiDeNN-TD-TO method, the CPU time, and the speed-up relative to FEM-TO.

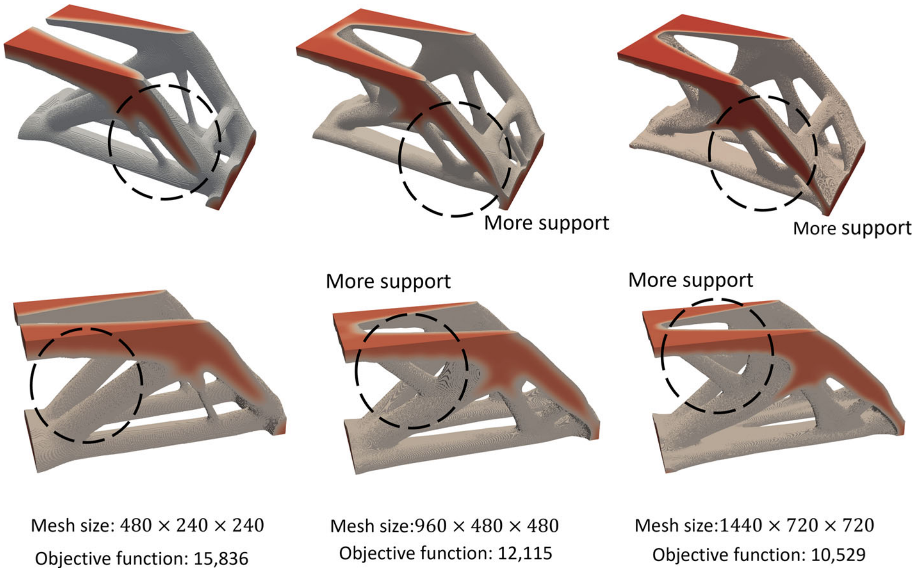
The final design structures of the three cases are shown in Fig. 13. As the design resolution increases, the overall structure becomes more and more similar. For the same volume fraction and the given length-scale control, the topology design tends to have a convergent result. However, for some

Table 2 Comparison of speed-up in different cases

Resolution	FEM DoFs	C-HiDeNN-TD-TO DoFs	Number of modes		CPU time		Speed-up
			ρ	u	FEA-TO	C-HiDeNN-TD-TO	
480 ×240 × 240	111,747,844	964,926	34	334	9000 h	33 h	273
960 × 480× 480	889,351,684	3,830,616	36	664	72,000 h	64 h	1125
1440 × 720× 720	2,996,363,524	7,273,809	42	841	768,000 h	112 h	6857

The CPU time of the FEM-TO method is estimated based on [4]

Fig. 13 Three cantilever beam cases with different mesh sizes. Higher resolution TO produces more optimal designs with more supporting structural features


More support
More support
More support
More support
Mesh size: 480 × 240 × 240
Objective function: 15,836
Mesh size:960 × 480 × 480
Objective function: 12,115
Mesh size:1440 × 720 × 720
Objective function: 10,529

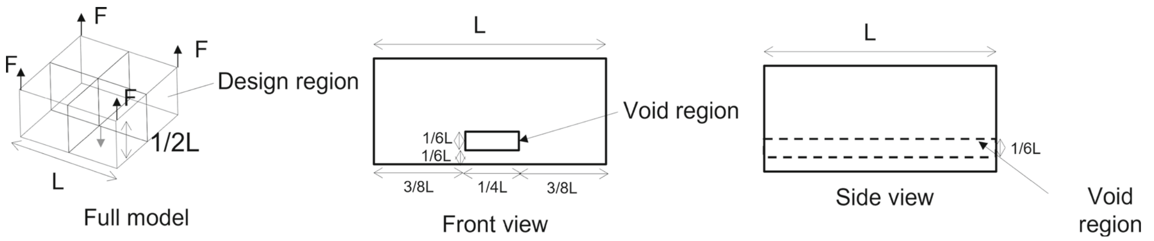


Fig. 14 Drone design problem definition

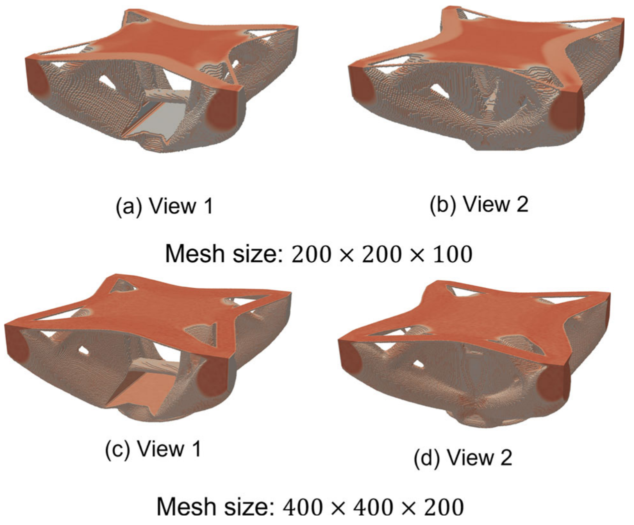


Fig. 15 Drone design results with volume constraint 0.2 with different design resolutions

regions, such as the region in a dashed circle, the increasing resolution results in more support structures and an improved, lesser objective function.

5.2 Example 2: Drone design problem

In the second example, the C-HiDeNN-TD-TO is applied to a drone design problem shown in Fig. 14. The design region is a rectangular box with loads on the four top corners and the center of the bottom. For the numerical implementation, the four corners are treated as the fixed boundary conditions in all directions. Furthermore, there is a void region in the design space to leave room for batteries and a control unit, as shown in Fig. 14. The design goal is to minimize system compliance under some predefined design constraints on the volume fraction and minimum length-scale control, as defined in Eq. (14). Here we study two different volume fractions, 0.12 and 0.2, with the minimum length-scale set at $0.03L$ as shown in Fig. 14. The void region is performed by enhancing a passive region in the design loop, which means the void region is forced to have zero density at each design iteration.

The final TO structures are shown in Figs. 15, 16 with different design resolutions. In Fig. 15, the drone is designed with a volume fraction constraint of 0.2, whereas in Fig. 16, the drone is designed with a volume constraint of 0.12. It can be observed that the lower volume fraction will produce thinner truss structures. The objective function of design with volume constraint 0.2 is 3.3618, and the objective function with volume constraint 0.12 is 3.4162.

6 Future development

In this section, we will discuss several future developments of C-HiDeNN-TD-TO. One direction is how to utilize advanced computing hardware to improve the efficiency of C-HiDeNN-TD-TO further. We will use one example to illustrate the idea. We will also discuss the extension of the method to a two-scale design C-HiDeNN-TD^2. A concurrent drone problem will be given from the preliminary developed C-HiDeNN-TD^2 framework. More details and results will be introduced in a future paper, including r- or mesh adaptivity for which the C-HiDeNN shape functions are expressed as functions of position $\boldsymbol{x}$, giving rise to a mesh refinement criterion around local regions.

6.1 GPU acceleration in C-HiDeNN-TD-TO

One feasible direction is to use GPU to accelerate C-HiDeNN-TD-TO. In particular, several subroutines of the overall framework are highly parallelizable. As shown in Fig. 17, the HOPGD subroutine can be accelerated by parallelizing operations and utilizing GPU hardware, with a slight modification of the convergence criterion used to indicate a sufficient number of decomposed modes. For the high-resolution mesh, the HOPGD algorithm ultimately saw a 142 times reduction in compute time on an Nvidia GeForce RTX 3070 GPU relative to the original standard implementation [37] executed on an AMD Ryzen 5900X CPU with 12 cores. Decomposition modes must be calculated sequentially during HOPGD, which results in compute time scaling linearly with the number of decomposed modes to be calculated. This leads to CPUs, which generally have faster

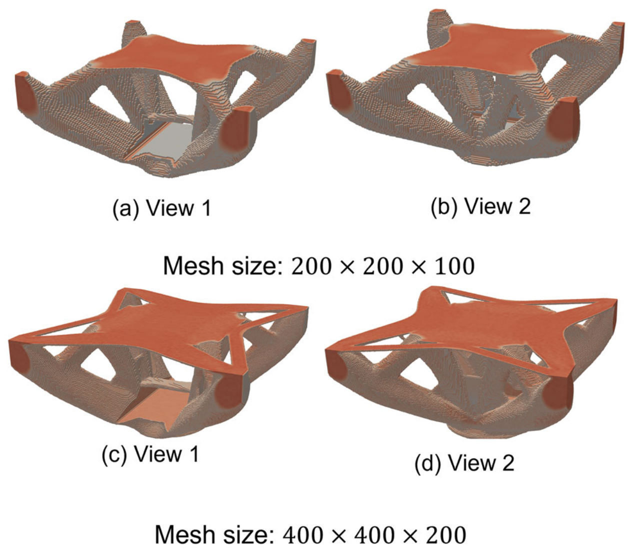


Fig. 16 Drone design results with volume constraint 0.12 with different design resolutions

clock rates, outperforming GPUs on small resolution meshes where the vector operations within each mode calculation offer limited opportunity for parallelization. GPU execution offers a massive advantage for high-resolution meshes where C-HiDeNN-TD-TO can excel compared to traditional FEM TO. For ultra-fine meshes where C-HiDeNN-TD-TO may exceed the memory limits of GPU hardware, Message Passing Interface (MPI) to distribute computation across multiple processors should further improve performance. The linear scaling of HOPGD compute time with the number of tensor decomposition modes is not particularly problematic for two reasons. First, the bulk of the computational expense of C-HiDeNN-TD-TO for fine meshes is in the solution subroutine, for which the equations of each mode can be solved in parallel. Second, the use of inordinately many modes is not essential to achieve sufficient representation accuracy. Hundreds of modes have been observed to capture complex geometries adequately, and if extreme accuracy is desired, one could choose to implement more modes only during the final iterations of a topology optimization procedure. This example shows how to utilize the massive parallelization offered by GPUs to accelerate the original C-HiDeNN-TD-TO code. More studies will be performed in the future.

6.2 Multi-unit cell concurrent drone design

In the second direction, we will discuss C-HiDeNN-TD^2 for concurrent multi-unit cell design. Concurrent multi-scale design is based on the multi-scale theory and aims to optimize the two or more scales simultaneously [47]. Considering homogenization theory, we extend the original C-HiDeNN-TD to a two-scale design method called C-HiDeNN-TD^2.

The design problem definition of the drone structure is shown in Fig. 18. The boundary conditions and the loading type are the same as in the previous example. The difference is in the predefined regions of the design domain. For each region, we assume it consists of the same unit cells. At the initial design stage, all the unit cells of each region have the same initial set-up with a volume fraction of 0.5. The drone structure and the unit cells are simultaneously updated during the concurrent design stage. Results are shown in Fig. 19. From the figure, different regions have different unit cell structures that better reduce the objective function. By applying the reinforcing connectivity conditions, we ensure each unit cell is perfectly connected with its neighbors. This example shows one potential application of the C-HiDeNN-TD^2 method. In the future, it can be extended to an n-scale concurrent design framework with high-resolution design routines at each scale.

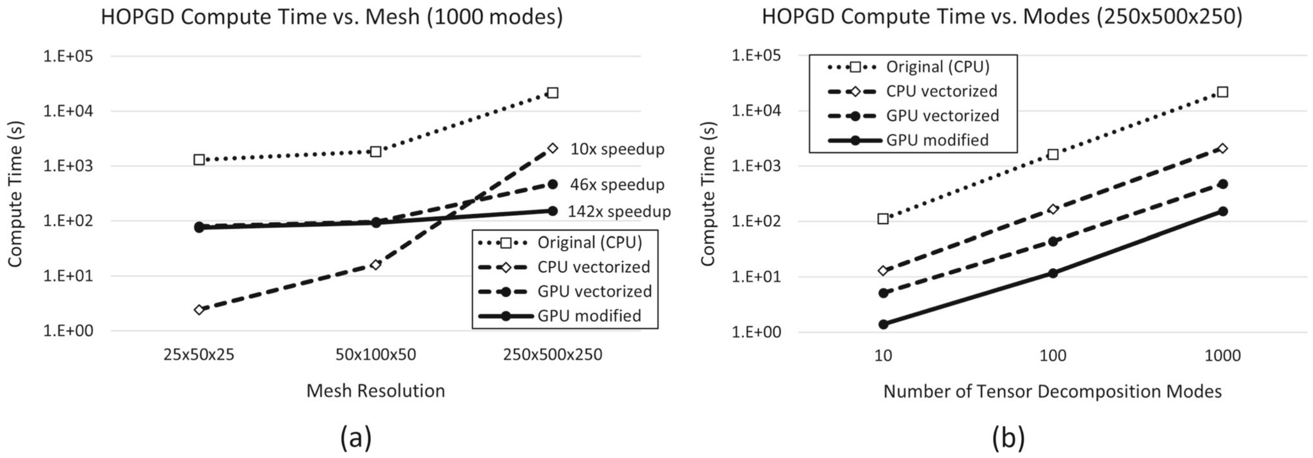


Fig. 17 GPU acceleration within the C-HiDeNN-TD-TO. With operation vectorization, the HOPGD subroutine can execute 142 times faster on GPU than the original implementation on CPU

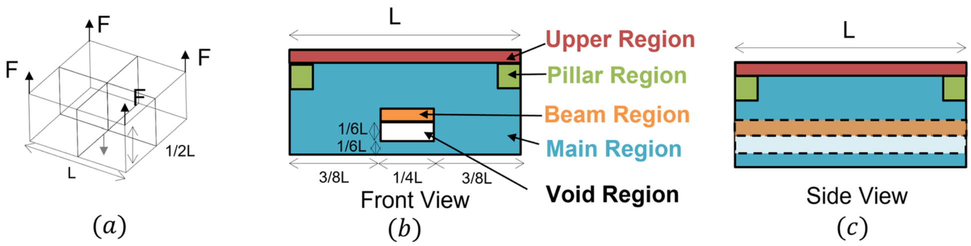


Fig. 18 Multi unit cell concurrent drone design problem definition. The initial design domain is divided into different regions as shown in **b** front view and **c** side view

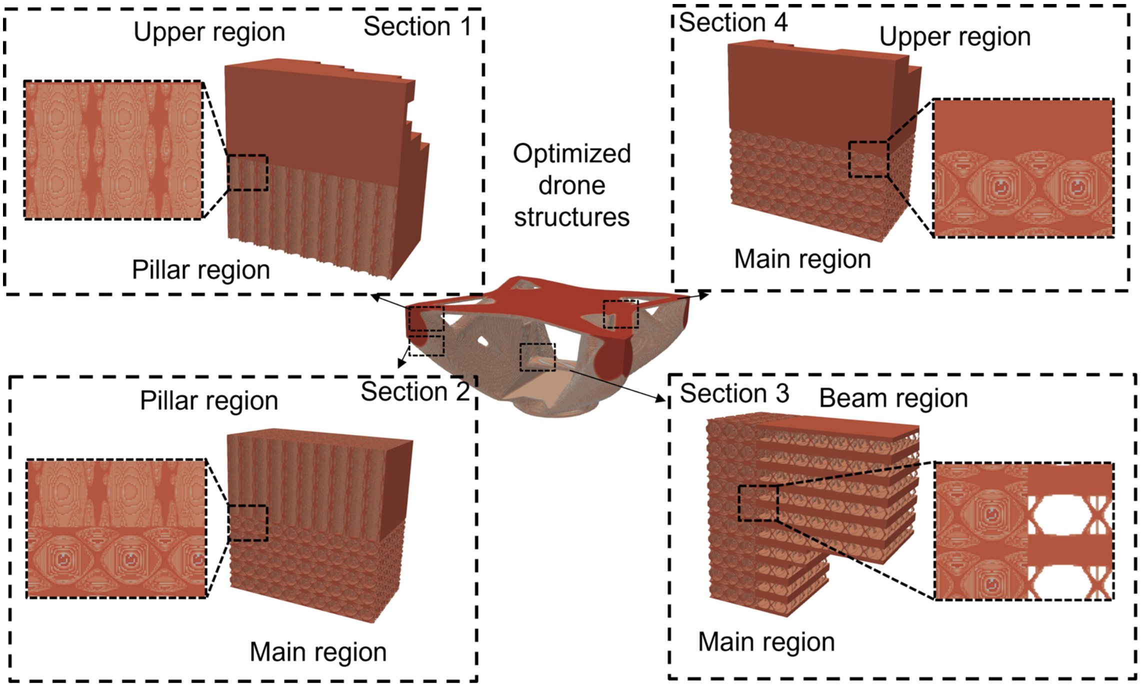


Fig. 19 Multi unit cell concurrent drone designed structure. Each section showed good connectivity between each region

Apart from the two directions, another future direction is to leverage the computational efficiency of C-HiDeNN-TD-TO for concurrent multiscale topology optimization, for which the scheme can be nested to optimize structures at the part and element scales simultaneously, further increasing the feasibility of structure optimization for extremely high-resolution meshes. Additionally, the incorporation of finite deformation representation would empower the framework to produce designs for large deformation problems. Finally, further study of the convolution scheme to smoothen design representation and the current improvement of solution accuracy is expected to increase TO stability and the manufacturability of resulting structures going forward.

7 Conclusions

In this paper, we proposed the Convolution-Hierarchical Deep Learning Neural Network-Tensor Decomposition (C-HiDeNN-TD) for topology optimization. We introduced the C-HiDeNN theory, which is a deep learning-based discretization method with advantages in terms of both accuracy and computational efficiency. Then, the convolution tensor decomposition-based topology optimization is developed to reduce the calculation cost. Numerical examples demonstrate the superior performance of the C-HiDeNN-TD-TO, especially for ultra-large-scale topology design and concurrent multi-scale topology optimization. In addition, the effects of controlling the dilation parameter and polynomial order in convolution approximation have been studied. We summarize the key features of the C-HiDeNN-TD-TO:

- Overcomes the memory limitations of high-resolution topology optimization by avoiding the storage of a full matrix.
- Reduces the computational cost, enabling thousands of times speedup for giga-scale topology optimization problems and efficient concurrent multi-scale topology design.
- Improved solution accuracy compared to traditional SIMP methods by using a convolution approximation that

leads to higher-order smooth descriptions of structures for a given mesh.
- The convolution approximation provides a built-in length-scale filter for topology optimization and avoids the extra filtering step in traditional topology optimization.
- Local design refinement can be easily achieved due to the location dependence of the convolution shape functions. This feature can be exploited for mesh adaptivity as demonstrated by HiDeNN [28] and HiDeNN-TD [35].

The proposed topology design framework can be further accelerated by GPU and has considerable potential to enable fabrication-specific topology design for additive manufacturing, which usually requires extreme model resolutions to predict the physical shape properties accurately. C-HiDeNN-TD-TO democratizes topology optimization for practical problems by preserving solution accuracy while alleviating the formerly extravagant memory and computation requirements [48].

Appendix A: derivation of C-HiDeNN-TD for topology optimization solver

To start with, the displacement $\boldsymbol{u} = [u, v, w]$ and the density ρ are assumed to have the separated form.

$$
\begin{aligned}
u\left(x, y, z\right) &= \sum_{q=1}^{Q} u_x^q\left(x\right) u_y^q\left(y\right) u_z^q\left(z\right), \\
v\left(x, y, z\right) &= \sum_{q=1}^{Q} v_x^q\left(x\right) v_y^q\left(y\right) v_z^q\left(z\right), \\
w\left(x, y, z\right) &= \sum_{q=1}^{Q} w_x^q\left(x\right) w_y^q\left(y\right) w_z^q\left(z\right), \\
\rho\left(x, y, z\right) &= \sum_{k=1}^{K} \rho_x^k\left(x\right) \rho_y^k\left(y\right) \rho_z^k\left(z\right),
\end{aligned}
\tag{15}
$$

To distinguish the modes of u and ρ, the index of mode in ρ is defined as k, and the maximum number of modes is K; the index of mode in $\boldsymbol{u}$ is defined as q with the maximum number as Q. The equilibrium equation in principle of virtual work can be written as

$$
\begin{aligned}
\delta\Pi &= \int_{\Omega} \nabla_s \delta\boldsymbol{u}^T \boldsymbol{D} \nabla_s \boldsymbol{u} \rho d\Omega - \int_{\Omega} \delta\boldsymbol{u}^T \boldsymbol{f} \rho d\Omega - \int_{\Gamma} \delta\boldsymbol{u}^T \boldsymbol{t} d\Gamma \\
&= \delta\Pi^{int} - \delta\Pi^{ext} = 0,
\end{aligned}
\tag{16}
$$

To solve the problem, we can borrow the alternating fix point algorithm, like PGD methods [32]. In this case, the maximum number of modes Q of u is a dynamic changing variable. For example, initially, we start from $Q = 1$. All the separated functions of displacement are then computed for the mode 1. Once they are computed, we can consider they are constant and update the Q as $Q = Q + 1$ for computing the second mode. This procedure successively enrich the modes until convergence. For the density ρ, the maximum number of modes K is determined by the HOPGD procedure [37].

In the next, we illustrate the procedure for the Q-th mode, assuming $Q - 1$ modes have been computed earlier. Then we can compute first the $u_x^{q=Q}(x)$ by assuming the other displacement components are constant. The variation of the displacement becomes

$$
\delta\boldsymbol{u} = \left[\delta u_x^Q(x) u_y^Q\left(y\right) u_z^Q\left(z\right), 0, 0\right]^T, \tag{17}
$$

Therefore, the variation of Eq. (16) under current mode with variable $u_x^{q=Q}(x)$ can be expressed as

$$
\begin{aligned}
&\delta\Pi_{u_x^Q 1}^{int} + \delta\Pi_{u_x^Q 2}^{int} + \delta\Pi_{u_x^Q 3}^{int} + \delta\Pi_{u_x^Q 4}^{int} + \delta\Pi_{u_x^Q 5}^{int} + \delta\Pi_{u_x^Q 6}^{int} \\
&\quad + \delta\Pi_{u_x^Q 7}^{int} - \delta\Pi_{u_x^Q}^{ext} = 0,
\end{aligned}
\tag{18}
$$

where each term of the $\Pi_{u_x^Q}^{int}$ is defined as below

$$
\begin{aligned}
\delta\Pi_{u_x^Q 1}^{int} = \frac{E(1-\mu)}{(1+\mu)(1-2\mu)} \sum_{k=1}^{K} \sum_{q=1}^{Q} &\left[\int_x \left(\frac{\partial \delta u_x^Q}{\partial x} \frac{\partial u_x^q}{\partial x}\right) \rho_x^k dx \int_y \left(u_y^Q u_y^q\right) \rho_y^k dy \right. \\
&\left. \int_z \left(u_z^Q u_z^q\right) \rho_z^k dz\right],
\end{aligned}
\tag{19}
$$

$$
\begin{aligned}
\delta\Pi_{u_x^Q 2}^{int} = \frac{E\mu}{(1+\mu)(1-2\mu)} \sum_{k=1}^{K} \sum_{q=1}^{Q} &\left[\int_x \left(\frac{\partial \delta u_x^Q}{\partial x} v_x^q\right) \rho_x^k dx \int_y \left(u_y^Q \frac{\partial v_y^q}{\partial y}\right) \rho_y^k dy \right. \\
&\left. \int_z \left(u_z^Q v_z^q\right) \rho_z^k dz\right],
\end{aligned}
\tag{20}
$$

$$
\begin{aligned}
\delta\Pi_{u_x^Q 3}^{int} = \frac{E\mu}{(1+\mu)(1-2\mu)} \sum_{k=1}^{K} \sum_{q=1}^{Q} &\left[\int_x \left(\frac{\partial \delta u_x^Q}{\partial x} w_x^q\right) \rho_x^k dx \int_y \left(u_y^Q w_y^q\right) \rho_y^k dy \right. \\
&\left. \int_z \left(u_z^Q \frac{\partial w_z^q}{\partial z}\right) \rho_z^k dz\right],
\end{aligned}
\tag{21}
$$

$$
\delta\Pi_{u_x^Q 4}^{int} = \frac{E}{2(1+\mu)} \sum_{k=1}^{K} \sum_{q=1}^{Q}
$$

$$\left[\int_x \left(\delta u_x^Q u_x^q\right) \rho_x^k dx \int_y \left(\frac{\partial u_y^Q}{\partial y}\frac{\partial u_y^q}{\partial y}\right) \rho_y^k dy \right.$$
$$\left.\int_z \left(u_z^Q u_z^q\right) \rho_z^k dz\right], \tag{22}$$

$$\delta\Pi_{u_x^Q 5}^{int} = \frac{E}{2(1+\mu)} \sum_{k=1}^{K}\sum_{q=1}^{Q}$$
$$\left[\int_x \left(\delta u_x^Q \frac{\partial v_x^q}{\partial x}\right) \rho_x^k dx \int_y \left(\frac{\partial u_y^Q}{\partial y} v_y^q\right) \rho_y^k dy \right.$$
$$\left.\int_z \left(u_z^Q v_z^q\right) \rho_z^k dz\right], \tag{23}$$

$$\delta\Pi_{u_x^Q 6}^{int} = \frac{E}{2(1+\mu)} \sum_{k=1}^{K}\sum_{q=1}^{Q}$$
$$\left[\int_x \left(\delta u_x^Q u_x^q\right) \rho_x^k dx \int_y \left(u_y^Q u_y^q\right) \rho_y^k dy \right.$$
$$\left.\int_z \left(\frac{\partial \delta u_z^Q}{\partial z}\frac{\partial u_z^q}{\partial z}\right) \rho_z^k dz\right], \tag{24}$$

$$\delta\Pi_{u_x^Q 7}^{int} = \frac{E}{2(1+\mu)} \sum_{k=1}^{K}\sum_{q=1}^{Q}$$
$$\left[\int_x \left(\delta u_x^Q \frac{\partial w_x^q}{\partial x}\right) \rho_x^k dx \int_y \left(u_y^Q w_y^q\right) \rho_y^k dy \right.$$
$$\left.\int_z \left(\frac{\partial \delta u_z^Q}{\partial z} w_z^q\right) \rho_z^k dz\right], \tag{25}$$

where the original 3D integration in the Eq. (16) is decomposed into the product of 1D integration. This is the advantage due to the separation of variables for reducing the computational complexity.

The external virtual work can be defined as

$$\Pi_{u_x^Q}^{ext} = \int_\Omega \sum_{m=1}^{M} f_{rx}^m \delta u_x^Q f_{ry}^m u_y^Q f_{rz}^m u_z^Q \rho d\Omega$$
$$- \int_\Gamma \sum_{n=1}^{N} t_{rx}^n \delta u_x^Q t_{ry}^n u_y^Q t_{rz}^n u_z^Q \Gamma, \tag{26}$$

with the decomposition of the external body force $\boldsymbol{f} = [f_r, f_s, f_t]^T$ and boundary traction: $\boldsymbol{t} = [t_r, t_s, t_t]^T$.

$$f_r(x, y, z) = \sum_{m=1}^{M} f_{rx}^m(x) f_{ry}^m(y) f_{rz}^m(z), \tag{27}$$

$$t_r(x, y, z) = \sum_{n=1}^{N} t_{rx}^n(x) t_{ry}^n(y) t_{rz}^n(z), \tag{28}$$

In the expression for the external virtual work (26), other components of the body force and traction force become zero as it will not contribute to u_x^Q.

With these definitions, we can discretize the problem (18) using C-HiDeNN. The final discretized form reads

$$\tilde{\boldsymbol{K}}_{u_x^Q} \boldsymbol{u}_x^Q = \boldsymbol{f}_{u_x^Q} \tag{29}$$

where $\tilde{\boldsymbol{K}}_{u_x^Q}$ is the directional stiffness matrix associated with component u_x^Q.

By analogy, we can derive the equations for $u_{y,k}^Q$ by considering

$$\delta \boldsymbol{u} = \left[u_x^Q(x) \delta u_y^Q(y)\, u_z^Q(z), 0, 0\right]^T, \tag{30}$$

and similar for other displacement components $u_{z,k}^Q$, $v_{x,k}^Q$, $v_{y,k}^Q$, $v_{z,k}^Q$, $w_{x,k}^Q$, $w_{y,k}^Q$, $w_{z,k}^Q$. The equations will be solved repetitively until the Q-th mode remains unchanged before passing the next mode.

Appendix B: derivation of window function [23]

Consider the following 1D case for illustration purposes.

$$u^{\text{C-FEM}}(\xi) = \sum_{i \in \mathcal{A}^e} N_i(\xi) \sum_{j \in A_s^i} W_{a,j}^{\xi_i}(\xi) u_j \tag{31}$$

Then we can define the following interpolation as the part of the approximation centering around the i-th node in the element domain.

$$u^i(\xi) = \sum_{j \in A_s^i} W_{a,j}^{\xi_i}(\xi) u_j, \tag{32}$$

where the supporting node set of W is A_s^i with a given patch size s. Assuming the nodal solution value for the 4 nodes is $[u_1, u_2, u_3, u_4]$, we illustrate the radial basis interpolation procedure for the part centering around $i = 2$. In this case, the parametric coordinates for the support nodes are $\{-3, -1, 1\}$. Then we can consider the radial basis interpolation $u^{i=2}(\xi)$ has the following form

$$u^i(\xi) = \boldsymbol{\Psi}_a(\xi)\mathbf{k} + \mathbf{p}(\xi)\mathbf{l}, \tag{33}$$

where $\boldsymbol{\Psi}_a$ is a defined kernel function, which can be the reproducing kernel or cubic spline kernel [29, 49] with the dilation parameter a, $\mathbf{p}(\xi)$ is the polynomial basis vector of p-th order, $\mathbf{k} = [k_1, k_2, k_3]^T$ and $\mathbf{l} = [l_1, l_2, l_3]^T$ are the coefficient vector that helps to enforce the reproducing condition and Kronecker delta property. We give here a specific example for $\boldsymbol{\Psi}_a$ and $\mathbf{p}(\xi)$ using a cubic spline Kernel and a second-order polynomial.

$$\begin{aligned} \boldsymbol{\Psi}_a(\xi) &= [\Psi_a(\xi-\xi_1),\ \Psi_a(\xi-\xi_2),\ \Psi_a(\xi-\xi_3)] \\ \text{where}\quad \Psi_a(\xi-\xi_I): &= \Psi_a(z) \quad \text{with} \quad z=\frac{|\xi-\xi_I|}{a} \\ &= \begin{cases} \frac{2}{3}-4z^2+4z^3 & \forall z\in[0,\,\frac{1}{2}] \\ \frac{4}{3}-4z+4z^2-\frac{4}{3}z^3 & \forall z\in[\frac{1}{2},\,1] \\ 0 & \forall z\in(1,+\infty) \end{cases}, \end{aligned} \tag{34}$$

and

$$\mathbf{p}=[1,\ \xi,\ \xi^2] \tag{35}$$

Now we can compute $\mathbf{k}$ and $\mathbf{l}$ by enforcing the below conditions.

$$\begin{cases} u^i(\xi_1)=u_1 \\ u^i(\xi_2)=u_2 \\ u^i(\xi_3)=u_3 \\ \sum\mathbf{k}=0 \\ [\xi_1,\xi_2,\xi_3]\,\mathbf{k}=0 \\ [\xi_1^2,\xi_2^2,\xi_3^2]\,\mathbf{k}=0 \end{cases}, \tag{36}$$

Solving the above equations gives the solution to $\mathbf{k}$ and $\mathbf{l}$, which reads

$$\begin{cases} \mathbf{k}=\mathbf{Ku} \\ \mathbf{l}=\mathbf{Lu} \end{cases}, \tag{37}$$

with

$$\begin{cases} \mathbf{u}=[u_1,\ u_2,\ u_3]^T \\ \mathbf{L}=(\mathbf{P}^T\mathbf{R}_0\mathbf{P})^{-1}\mathbf{P}^T\mathbf{R}_0^{-1} \\ \mathbf{K}=\mathbf{R}_0^{-1}(\mathbf{I}-\mathbf{PL}) \end{cases}, \tag{38}$$

and

$$\begin{cases} \mathbf{R}_0 = \begin{pmatrix} \boldsymbol{\Psi}_a(\xi_1) \\ \boldsymbol{\Psi}_a(\xi_2) \\ \boldsymbol{\Psi}_a(\xi_3) \end{pmatrix} = \begin{pmatrix} \Psi_a(\xi_1-\xi_1) & \Psi_a(\xi_1-\xi_2) & \Psi_a(\xi_1-\xi_3) \\ \Psi_a(\xi_2-\xi_1) & \Psi_a(\xi_2-\xi_2) & \Psi_a(\xi_2-\xi_3) \\ \Psi_a(\xi_3-\xi_1) & \Psi_a(\xi_3-\xi_2) & \Psi_a(\xi_3-\xi_3) \end{pmatrix} \\ \mathbf{P} = \begin{pmatrix} \mathbf{p}(\xi_1) \\ \mathbf{p}(\xi_2) \\ \mathbf{p}(\xi_3) \end{pmatrix} = \begin{pmatrix} 1 & \xi_1 & \xi_1^2 \\ 1 & \xi_2 & \xi_2^2 \\ 1 & \xi_3 & \xi_3^2 \end{pmatrix} \end{cases}, \tag{39}$$

Finally, the radial basis interpolation with the computed coefficients reads:

$$\begin{aligned} u^i(\xi) = \boldsymbol{\Psi}_a(\xi)\mathbf{k}+\mathbf{p}(\xi)\mathbf{l} &= \boldsymbol{\Psi}_a(\xi)\mathbf{Ku}+\mathbf{p}(\xi)\mathbf{Lu} \\ &= (\boldsymbol{\Psi}_a(\xi)\mathbf{K}+\mathbf{p}(\xi)\mathbf{L})\mathbf{u} \\ &= W_{a,1}^{\xi_i}(\xi)u_1 + W_{a,2}^{\xi_i}(\xi)u_2 \\ &\quad + W_{a,3}^{\xi_i}(\xi)u_3 \\ &= \sum_{j\in A_s^i} W_{a,j}^{\xi_i}(\xi)u_j, \end{aligned} \tag{40}$$

where $W_{a,j}^{\xi_i}$ is obtained by identifying the corresponding coefficient of u_j. By analogy, we can compute the other convolution patch functions W with the support $A_s^{i=3}$. Detailed mathematical derivation and analysis of the radial basis interpolation can be found in [50].

Appendix C: 1D Poisson's problem

Poisson's equation is commonly used to demonstrate the efficiency of a numerical method because the analytical solution can be derived. In this paper, we solve a 1D Poisson's problem defined as:

$$\begin{aligned} \Delta u(x)+b(x)=0 &\quad \text{in} \quad \Omega \\ u=0 &\quad \text{on} \quad \Gamma \end{aligned}$$

where $\Omega=[0,\,10]$ is the problem domain and Γ is the domain boundary. We first construct an analytical solution $u(x)$ and then derive the body force $b(x)$ from the equation. The solution $u(x)$ and the body force $b(x)$ are given as:

$$\begin{aligned} u(x) = &\left[e^{-5(x-2.5)^2}-e^{-31.25}\right]+2\left[e^{-5(x-7.5)^2}-e^{-281.25}\right] \\ &-\frac{e^{-31.25}-e^{-281.25}}{10}x, \end{aligned} \tag{41}$$

$$\begin{aligned} b(x) = &-\left[100(x-2.5)^2-10\right]e^{-5(x-2.5)^2} \\ &-2\left[100(x-7.5)^2-10\right]e^{-5(x-7.5)^2}. \end{aligned} \tag{42}$$

To measure the accuracy, the H^1 norm error estimator was used:

$$\|e\|_{H^1} = \frac{\left(\int_\Omega \left(u-u^h\right)^2 dx + \int_\Omega \|\nabla u-\nabla u^h\|_2^2 dx\right)^{1/2}}{\left(\int_\Omega u^2 dx + \int_\Omega \|\nabla u^h\|_2^2 dx\right)^{1/2}}. \tag{43}$$

References

1. Bendsøe MP (1989) Optimal shape design as a material distribution problem. Struct Optim 1(4):193–202
2. Aage N, Andreassen E, Lazarov BS, Sigmund O (2017) Giga-voxel computational morphogenesis for structural design. Nature 550(7674):84–86
3. Xiao M, Lu D, Breitkopf P, Raghavan B, Dutta S, Zhang W (2020) On-the-fly model reduction for large-scale structural topology optimization using principal components analysis. Struct Multidiscip Optim 62(1):209–230
4. Aage N, Andreassen E, Lazarov BS (2015) Topology optimization using petsc: an easy-to-use, fully parallel, open source topology optimization framework. Struct Multidiscip Optim 51(3):565–572
5. Sigmund O (2022) On benchmarking and good scientific practise in topology optimization. Struct Multidiscip Optim 65(11):315
6. Gabriel E, Fagg GE, Bosilca G, Angskun T, Dongarra JJ, Squyres JM, Sahay V, Kambadur P, Barrett B, Lumsdaine A et al (2004) Open mpi: goals, concept, and design of a next generation mpi implementation. European parallel virtual machine/message passing interface users' group meeting. Springer, New York, pp 97–104
7. Walker DW, Dongarra JJ (1996) Mpi: a standard message passing interface. Supercomputer 12:56–68
8. Mahdavi A, Balaji R, Frecker M, Mockensturm E (2006) Topology optimization of 2d continua for minimum compliance using parallel computing. Struct Multidiscip Optim 32(2):121–132
9. Aage N, Poulsen TH, Gersborg-Hansen A, Sigmund O (2008) Topology optimization of large scale stokes flow problems. Struct Multidiscip Optim 35(2):175–180
10. Evgrafov A, Rupp CJ, Maute K, Dunn ML (2008) Large-scale parallel topology optimization using a dual-primal substructuring solver. Struct Multidiscip Optim 36(4):329–345
11. Wadbro E, Berggren M (2009) Megapixel topology optimization on a graphics processing unit. SIAM Rev 51(4):707–721
12. Xia L, Breitkopf P (2014) A reduced multiscale model for nonlinear structural topology optimization. Comput Methods Appl Mech Eng 280:117–134
13. Ferro N, Micheletti S, Perotto S (2019) Pod-assisted strategies for structural topology optimization. Comput Math Appl 77(10):2804–2820
14. Xiao M, Lu D, Breitkopf P, Raghavan B, Zhang W, Dutta S (2020) Multi-grid reduced-order topology optimization. Struct Multidiscip Optim 61(6):1–23
15. Lei X, Liu C, Du Z, Zhang W, Guo X (2019) Machine learning-driven real-time topology optimization under moving morphable component-based framework. J Appl Mech 86(1):011004
16. Jiang X, Wang H, Li Y, Mo K (2020) Machine learning based parameter tuning strategy for mmc based topology optimization. Adv Eng Softw 149:102841
17. Sosnovik I, Oseledets I (2019) Neural networks for topology optimization. Russ J Numer Anal Math Model 34(4):215–223
18. Yu Y, Hur T, Jung J, Jang IG (2019) Deep learning for determining a near-optimal topological design without any iteration. Struct Multidiscip Optim 59(3):787–799
19. Sasaki H, Igarashi H (2019) Topology optimization accelerated by deep learning. IEEE Trans Magn 55(6):1–5
20. Deng C, Wang Y, Qin C, Fu Y, Lu W (2022) Self-directed online machine learning for topology optimization. Nat Commun 13(1):1–14
21. Huang M, Du Z, Liu C, Zheng Y, Cui T, Mei Y, Li X, Zhang X, Guo X (2022) Problem-independent machine learning (piml)-based topology optimization-a universal approach. Extreme Mech Lett 56:101887
22. Woldseth RV, Aage N, Bærentzen JA, Sigmund O (2022) On the use of artificial neural networks in topology optimisation. Struct Multidiscip Optim 65(10):1–36
23. Lu Y, Li H, Zhang L, Park C, Mojumder S, Knapik S, Sang Z, Tang S, Apley DW, Wagner GJ, Liu WK (2023) Convolution hierarchical deep-learning neural networks (c-hidenn): finite elements, isogeometric analysis, tensor decomposition, and beyond, Computational Mechanics
24. Park C, Lu Y, Saha S, Xue T, Guo J, Mojumder S, Apley D, Wagner G, Liu W (2023) Convolution hierarchical deep-learning neural network (c-hidenn) with graphics processing unit (gpu) acceleration, Computaitonal Mechanics
25. Wang MY, Wang X, Guo D (2003) A level set method for structural topology optimization. Comput Methods Appl Mech Eng 192(1–2):227–246
26. Guo X, Zhang W, Zhong W (2014) Doing topology optimization explicitly and geometrically-a new moving morphable components based framework. J Appl Mech 81(8)
27. Bendsoe MP, Sigmund O (2003) Topology optimization: theory, methods, and applications. Springer, New York
28. Zhang L, Cheng L, Li H, Gao J, Yu C, Domel R, Yang Y, Tang S, Liu WK (2021) Hierarchical deep-learning neural networks: finite elements and beyond. Comput Mech 67(1):207–230
29. Liu WK, Jun S, Zhang YF (1995) Reproducing kernel particle methods. Int J Numer Meth Fluids 20(8–9):1081–1106
30. Liu WK, Jun S, Li S, Adee J, Belytschko T (1995) Reproducing kernel particle methods for structural dynamics. Int J Numer Meth Eng 38(10):1655–1679
31. Jouppi N, Young C, Patil N, Patterson D (2018) Motivation for and evaluation of the first tensor processing unit. IEEE Micro 38(3):10–19
32. Ammar A, Mokdad B, Chinesta F, Keunings R (2006) A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. J Nonnewton Fluid Mech 139(3):153–176
33. Chinesta F, Ladeveze P, Cueto E (2011) A short review on model order reduction based on proper generalized decomposition. Arch Comput Methods Eng 18(4):395–404
34. Scanff R, Néron D, Ladevèze P, Barabinot P, Cugnon F, Delsemme J-P (2022) Weakly-invasive latin-pgd for solving time-dependent non-linear parametrized problems in solid mechanics. Comput Methods Appl Mech Eng 396:114999
35. Zhang L, Lu Y, Tang S, Liu WK (2022) Hidenn-td: reduced-order hierarchical deep learning neural networks. Comput Methods Appl Mech Eng 389:114414
36. Goutaudier D, Berthe L, Chinesta F (2021) Proper generalized decomposition with time adaptive space separation for transient wave propagation problems in separable domains. Comput Methods Appl Mech Eng 380:113755
37. Lu Y, Blal N, Gravouil A (2018) Multi-parametric space-time computational vademecum for parametric studies: application to real time welding simulations. Finite Elem Anal Des 139:62–72
38. Lu Y, Blal N, Gravouil A (2018) Adaptive sparse grid based hopgd: toward a nonintrusive strategy for constructing space-time welding computational vademecum. Int J Numer Meth Eng 114(13):1438–1461
39. Saha S, Kafka OL, Lu Y, Yu C, Liu WK (2021) Microscale structure to property prediction for additively manufactured in625 through advanced material model parameter identification. Integr Mater Manuf Innov 10(2):142–156
40. Diaz A, Sigmund O (1995) Checkerboard patterns in layout optimization. Struct Opt 10(1):40–45
41. Haber RB, Jog CS, Bendsøe MP (1996) A new approach to variable-topology shape design using a constraint on perimeter. Struct Opt 11(1):1–12

42. Sigmund O, Petersson J (1998) Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima. Struct Opt 16(1):68–75
43. Bourdin B (2001) Filters in topology optimization. Int J Numer Meth Eng 50(9):2143–2158
44. Sigmund O (1997) On the design of compliant mechanisms using topology optimization. J Struct Mech 25(4):493–524
45. Guest JK, Prévost JH, Belytschko T (2004) Achieving minimum length scale in topology optimization using nodal design variables and projection functions. Int J Numer Meth Eng 61(2):238–254
46. Svanberg K, Svärd H (2013) Density filters for topology optimization based on the pythagorean means. Struct Multidiscip Optim 48:859–875
47. Wu J, Sigmund O, Groen JP (2021) Topology optimization of multi-scale structures: a review. Struct Multidiscip Optim 63:1455–1480
48. Wu J, Dick C, Westermann R (2015) A system for high-resolution topology optimization. IEEE Trans Vis Comput Graphics 22(3):1195–1208
49. Chen J-S, Liu WK, Hillman MC, Chi S-W, Lian Y, Bessa MA (2017) Reproducing kernel particle method for solving partial differential equations. Encycl Comput Mech Second Edition 1–44
50. Schaback R, Wendland H (2001) Characterization and construction of radial basis functions. Multivar Approx Appl 1–24

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.