Advances in Engineering Software 139 (2020) 102733

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

An efficient isogeometric topology optimization using multilevel mesh, MGCG and local-update strategy

Yingjun Wang[a], Zhongyuan Liao[a], Ming Ye[b,c,*], Yu Zhang[d], Weihua Li[d], Zhaohui Xia[e,f,**]

[a] National Engineering Research Center of Novel Equipment for Polymer Processing, The Key Laboratory of Polymer Processing Engineering of the Ministry of Education (South China University of Technology), Guangdong Provincial Key Laboratory of Technique and Equipment for Macromolecular Advanced Manufacturing, South China University of Technology, Guangzhou 510641, China
[b] Guangzhou Huagong Motor Vehicle Inspection Technology Co., LTD, Guangzhou 510641, China
[c] National Engineering Research Center of Near-Net-Shape Forming for Metallic Materials, South China University of Technology, Guangzhou 510641, China
[d] School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510641, China
[e] National Enterprise Information Software Engineering Research Center, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China
[f] Center for Modeling, Simulation, and Imaging in Medicine, Rensselaer Polytechnic Institute, Troy, NY, USA

ARTICLE INFO

Keywords:
Isogeometric topology optimization
High-efficiency
Multilevel mesh
Multigrid conjugate gradient method
Local-update

ABSTRACT

This paper proposes a new high-efficiency isogeometric topology optimization (HITO), including three part: multilevel mesh, multigrid conjugate gradient method (MGCG) and local-update strategy, which improves the efficiency in three aspects: mesh scale reduction, solving acceleration and design variables reduction. Four benchmark examples are used to evaluate proposed method, and the results show that the proposed HITO successfully reduces 37%–93% computational time compared to the conventional isogeometric topology optimization (CITO) and obtains consistent optimization results, which demonstrates the high-efficiency of the HITO. Furthermore, the efficiency improvement of the HITO will be more significant for the large-scale problems.

1. Introduction

Isogeometric analysis (IGA) [1] combines the basis functions used in computer-aided design (CAD), such as nonuniform rational B-splines (NURBSs), with the variational framework of the finite element method (FEM). Compared to the conventional FEM, the most improvement of IGA is direct design-to-analysis, meaning the analysis is direct from computer-aided design (CAD) files without discretization. Furthermore, the IGA also has advantages in accuracy per degree of freedom (DOF), partial differential equation (PDE) solving and computational efficiency of high order elements [2]. With these merits of IGA, IGA has been successfully applied to a variety of domains, such as beams and shells [3,4], fluid [5], fluid-solid interaction [6], topology optimization [7,8], fracture [9] and many other analysis fields [10–12].

Topology optimization (TO) is a mathematical method aiming at finding the optimal material distribution subjected to some constrains within a given domain. A series of impressive TO methods have emerged in the past 30 years, such as homogenization method [13], solid isotropic material with penalization (SIMP) approach [14–17], evolutionary approach [18,19], level set method [20–22] and MMC method [23,24]. In recent years, IGA has been applied in topology optimization (TO) due to its advantages such as high accuracy and high continuity. Seo et al. [8] proposed a method for isogeometric TO (ITO) with the usage of trimmed spline surfaces, while the computational time for the analysis may significantly increase when the number of trim curves in a trimmed surface is large. According to work of Dedè et al. [25], the ITO provides higher accuracy and makes the design domain can be exactly represented on a phase field model for both 2D and 3D problems. Kang and Youn [26] applied ITO on shell structures using trimmed NURBS surfaces, which made the surface curvatures and the topology of a shell could be optimized effectively. Inherited from TO, a set of new ITO methods by integrated IGA with different TO methods (such as level-set [7,27] and MMC [2,28]) have also been proposed. Wang et al. [29,30] developed an arbitrary geometric constrained ITO and accelerated ITO by GPU parallel strategy. Moreover, using the theory of asymptotic homogenization, an ITO scheme for lattice materials with both isotropic and anisotropic is proposed by Wang et al. [31]. More ITO applications can be found in [32–34] and a

* Corresponding author at: Guangzhou Huagong Motor Vehicle Inspection Technology Co., LTD
** Corresponding author at: Huazhong University of Science and Technology.
E-mail addresses: yeming@scut.edu.cn (M. Ye), xiaz@hust.edu.cn (Z. Xia).

https://doi.org/10.1016/j.advengsoft.2019.102733
Received 10 June 2019; Received in revised form 24 September 2019; Accepted 20 October 2019
Available online 05 November 2019

recent review [35].

Although ITO can avoid mesh discretization, the optimization still costs much computational time duo to the iterative procedure of topological evolution. For engineering problems, the computational scales of TOs are usually large [36] and continuously growing due to complex problems with more details in the future. Therefore, pursuing higher computational efficiency is the eternal objective of TO. Mainly, there are two types of methods to improve the computational efficiency: one is parallel computing [30,37–40], and the other is the algorithm improvement [41–43]. Apparently, the later one is capable to fundamentally improve the efficiency of TO so that it will be studied in this paper. To improve the efficiency of the TO, several aspects can be worked including convergence acceleration, solution acceleration, and design variable reduction.

In order to accelerate the convergence rate, a two-stage approach for TO was presented by Lin and Zhou [44], the optimal result of the first stage with a coarse mesh was utilized as an initial design for the second stage with a finer mesh, and the optimized result of the first stage can accelerate the convergence of second stage, but only two different mesh sizes are considered in this two-stage approach. To obtain a high accuracy result with less computational cost, Stainko [45] presented a multilevel scheme that adaptively refined mesh along the interface between solid and void. Nguyen et al. [46] proposed an efficient multiresolution topology optimization (MTOP) technique, where a coarser mesh for the finite element analysis (FEA) was used to decrease the computational cost and the design variables were separated from the finite element (FE) mesh with a high resolution. Lieu et al. [47] combined multi-resolution approach with IGA for the multi-material TO problem, obtaining higher-resolution designs with a lower computational cost. Although the above MTOP approaches can decrease the degree of freedom (DOFs) in the FEA with a coarse mesh, but the accuracy of FEA is sacrificed due to the coarse mesh.

In ITO, it is required to implement IGA (similar to FEA) in each iteration, and a high-efficiency solution method to solve the equations of IGA can also reduce the computational cost of ITO. Equation solution methods are mainly divided into two types: direct methods [48] and iterative methods [49]. For large-scale problems, the iterative methods have better performance than direct methods in memory requirements and computational efficiency [50]. Many iterative methods have been used to solve TO problem, and one of most effective methods for FE problems is Preconditioned conjugate-gradients method (PCG) [51]. Amir et al. [52] used a standard PCG solver to terminate the iterative solution of the nested equations in a short time. Liu and Tovar [53] presented an efficient 3D TO code using the PCG in Matlab. Furthermore, Amir et al. [54] combined multigrid method and PCG into multigrid preconditioned conjugate gradients (MGCG), which can reduce the solving time of the linear equations for large-scale TOs. For the large-scale ITO, solving equations costs a large part of computational time, and thereby it is necessary to utilize an efficient solving method.

The number of design variables is one of the most important factors determining the computational cost, and Guest and Genut [55] found that reducing some design variables could reduce the number of iterations without losing the accuracy compared to that using all design variables. A reducible design variable method (RDVM) was presented by Yong et al. [41], which removed some quickly-converge design variables from iterations to reduce computational cost. Yoo and Lee [56] presented a variable grouping method which reduced variables according to histogram of multiplication of sensitivity and density variable. For large-scale ITO, the design variables will increase to a huge quantity, which brings many difficulties (e.g., large computational cost and slow convergence) in optimization. Therefore, reducing some unimportant design variables using a special strategy will improve the optimization efficiency.

As a burgeoning method, ITO has been demonstrated some advantages over conventional TO. Unavoidably, conventional challenges in TO, such as iterative solution and huge computational cost for large-scale problems, still exist in ITO. To improve the efficiency of ITO, this paper presents a new high-efficiency ITO method (HITO), which reduces the computational time by three methods: multilevel mesh, MGCG, and local-update strategy. In the reminder of this paper is organized as follows: Section 2 introduces the basic theories of ITO. A detailed model of the proposed high-efficiency ITO method is presented in Section 3, and the process of algorithm implementation is shown in Section 4. Section 5 discusses the efficiency improvement of the proposed method by several benchmark examples, and the final conclusions are drawn in Section 6.

2. Theoretical basis

IGA, SIMP-based TO and ITO-SIMP are the theoretical bases of this paper, which will be summarized in the following sections. For more detail discussion on IGA, SIMP-base TO and ITO-SIMP, we refer readers to [1,2,10,14,15,35,57].

2.1. . Basic fundamentals of IGA

In computer-aided design (CAD) and computer graphics (CG), Non-uniform rational B-splines (NURBS) is the most common representation for curves and surfaces [58]. The knot vector $\Xi = [\xi_1,\xi_2,\ldots, \xi_{n+p+1}]$ is a sequence of non-decreasing real numbers in the parametric space, where n is the number of control points (also the number of basis functions) and p is the degree of spline. For the sake of simplicity, the knot vectors in following contents are open knot vectors in which the first and last knots have multiplicity $p + 1$.

The B-spline basis functions $B_{i,p}(\xi)$ used in the construction of NURBS can be defined recursively by Cox-de Boor recursion formula [59]:

$$B_{i,0}(\xi) = \begin{cases} 1 if \xi_i \le \xi < \xi_{i+1} \\ 0 otherwise \end{cases}$$
$$B_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi) if \xi_i \le \xi < \xi_{i+1} \quad (1)$$

where the convention $0/0 = 0$. Fig. 1 gives an example of spline basis functions constructed by the knot $\Xi = [0, 0, 0, 1, 2, 3, 3, 3]$.

Introducing a positive weight w_i to each B-spline basis function, a univariate NURBS basis function is obtained as

$$N_{i,p}(\xi) = \frac{B_{i,p}(\xi) w_i}{\sum_{j=1}^{n} B_{j,p}(\xi) w_j}. \quad (2)$$

According to the tensor-product property, the two-dimensional (2D) or high-dimensional NURBS basis functions $N_{\boldsymbol{i},\boldsymbol{p}}(\boldsymbol{\xi})$ are constructed as:

$$N_{\boldsymbol{i},\boldsymbol{p}}(\boldsymbol{\xi}) = \prod_{m=1}^{d_p} N_{i_m,p_m}(\xi_{i_m}^m). \quad (3)$$

On the left of Eq. (3), $\boldsymbol{\xi} = [\xi_{i_1}^1, \xi_{i_2}^2, \ldots, \xi_{i_{d_p}}^{d_p}]$, represents the parametric coordinate, where $\boldsymbol{i} = [i_1, \ldots, i_{d_p}]$, represents the index position in the tensor product structure, and $\boldsymbol{p} = [p_1, \ldots, p_{d_p}]$, indicates the polynomial degrees for all parameter directions. On the right hand, N_{i_m,p_m} is the univariate B-spline basis functions along the mth parametric direction, obtained by Eq. (2). d_p denotes the dimension of the parameter space, $\Xi^m = \{\xi_1^m, \xi_2^m, \xi_3^m, \ldots, \xi_{n_m+p_m+1}^m\}$ $(m = 1, \ldots, d_p)$ are the d_p univariate knot vector, n_m indicates the associated number of functions and p_m is the polynomial degree along the parametric direction m.

2.2. . SIMP-based TO

Since the 99-lines Matlab codes for SIMP-based TO was proposed [14], more and more researches devoted themselves to the TO, and the

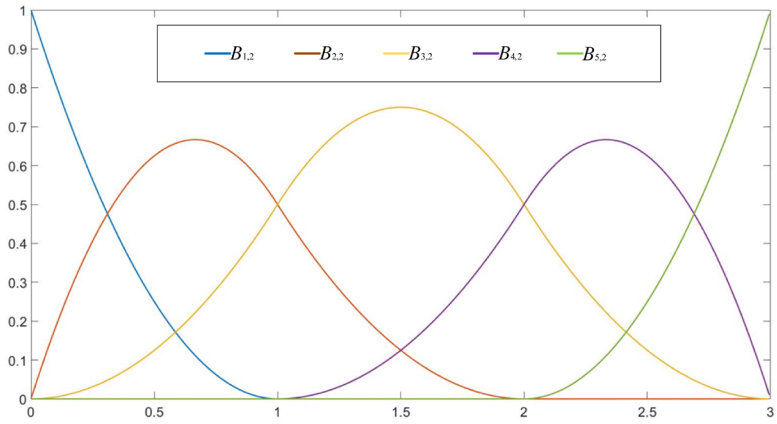


Fig. 1. An example of spline basis functions.

SIMP-based TO becomes the most widespread TO methods in the world. In the SIMP-based TO, the relation between element density ρ_e and Young's modulus E_e is represented as:

$$E_e(\rho_e) = E_{min} + \rho_e^s(E_0 - E_{min})\ \rho_e \in [0, 1], \tag{4}$$

where E_0 and E_{min} are the Young's moduli of standard solid and void elements, respectively. s is the penalization factor.

2.2.1. . Minimum compliance

For the classic compliance optimization problem, the mathematical description can be written as follows:

$$\text{Min } c(\boldsymbol{\rho}) = \boldsymbol{U}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{U} = \sum_{e=1}^{N} E_e(\rho_e)\boldsymbol{u}_e{}^{\mathrm{T}}\boldsymbol{k}_e\boldsymbol{u}_e,$$
$$\text{Subject to}\begin{cases}\boldsymbol{K}\boldsymbol{U} = \boldsymbol{F}\\ \dfrac{V(\boldsymbol{\rho})}{V_0} \le VF,\\ 0 \le \rho_e \le 1\end{cases} \tag{5}$$

where c is the compliance, N is the number of the design variables, ρ is the vector of design variables (i.e. the element densities), $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{k}_e$ is the element stiffness matrix with unit Young's modulus, $\boldsymbol{U}$ is the global displacement vector, $\boldsymbol{u}_e$ is the element displacement vector, $\boldsymbol{F}$ is the global force vector, $V(\rho)$, V_0 are the material volume and design domain volume, and VF is the volume fraction.

The sensitivities of the objective function c and the material volume V with respect to an element density ρ_e are calculated as:

$$\frac{\partial c}{\partial \rho_e} = -s(\rho_e)^{s-1}\boldsymbol{u}_e^T\ \boldsymbol{k}_e\boldsymbol{u}_e, \tag{6}$$

$$\frac{\partial V}{\partial \rho_e} = 1, \tag{7}$$

it is worth noting that Eq. (7) is established when each element has unit volume.

2.2.2. . Compliant mechanism

For the compliant mechanism problem, whose typical goal is to maximize the output port displacement, the mathematical description can be written as follows:

$$\text{Min } c(\boldsymbol{\rho}) = -u_{out} = -\boldsymbol{L}^{\mathrm{T}}\boldsymbol{U} = \boldsymbol{U}_d{}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{U},$$
$$\text{Subject to}\begin{cases}\boldsymbol{U}_d{}^{\mathrm{T}}\boldsymbol{K} = -\boldsymbol{L}\\ \boldsymbol{K}\boldsymbol{U} = \boldsymbol{F}\\ \dfrac{V(\boldsymbol{\rho})}{V_0} \le VF,\\ 0 \le \rho_e \le 1\end{cases} \tag{8}$$

where c is the objective function, u_{out} is the output displacement, ρ is the vector of design variables (i.e. the element densities). $\boldsymbol{L}$ is a unit length vector with zeros at all degrees of freedom expect at the output point where it is one, while $\boldsymbol{U}_d$ is defined as the global adjoint vector. $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{F}$ is the global force vector, $V(\rho)$, V_0 are the material volume and design domain volume, and VF is the volume fraction.

The sensitivities of the objective function c with respect to an element density ρ_e are calculated as:

$$\frac{\partial c}{\partial \rho_e} = -s(\rho_e)^{s-1}\boldsymbol{u}_{de}^T\ \boldsymbol{k}_e\boldsymbol{u}_e, \tag{9}$$

where $\boldsymbol{u}_{de}$ is the part of the adjoint vector associated with element e.

To avoid the defects of TO including mesh-dependency, local minima and checkboard, the filter techniques are usually used in TOs, and more detail can be found in [15,60]. The TO problems can be solved by a series of gradient-based methods such as sequential quadratic programming (SQP) [61], the optimality criteria (OC) method [10] and the method of moving asymptotes (MMA) [62].

2.3. . ITO-SIMP

Different from conventional SIMP, the numerical computations of a given physical field are calculated at control points in ITO-SIMP. Therefore, a variable x (e.g., coordinate, force, or displacement) whose parametric coordinate is $\boldsymbol{\xi}$ can be evaluated from the control point values

$$x(\boldsymbol{\xi}) = \sum_i N_i(\boldsymbol{\xi})x_i, \tag{10}$$

where N_i is the basis function of the ith control point influencing the position of $\boldsymbol{\xi}$, and x_i is the corresponding value of the control point.

In ITO-SIMP, an element stiffness matrix $\boldsymbol{k}_e$ can be calculated as:

$$\boldsymbol{k_e} = \int_{\hat{\Omega}_e} \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}|\boldsymbol{J}_1|d\hat{\Omega} = \int_{\bar{\Omega}_e} \boldsymbol{B}^T\boldsymbol{D}\boldsymbol{B}|\boldsymbol{J}_1||\boldsymbol{J}_2|d\bar{\Omega} \tag{11}$$

where $\boldsymbol{B}$ is strain-displacement matrix, $\boldsymbol{D}$ is stress-strain matrix, $\hat{\Omega}$ and $\bar{\Omega}$ are the domain of the parametric domain in the NURBS parametric space Ξ^m and the integration parametric space $\bar{\Xi}^m$, and $\boldsymbol{J}_1$ and $\boldsymbol{J}_2$ represent the transformation relation from the NURBS parametric space to the physical space and the integration parametric space to the NURBS parametric space, respectively.

Take the 2D case as an example, the NURBS parametric space is represented as $\Xi^m = \{\xi_1^m, \xi_2^m, \xi_3^m, ..., \xi_{n_m+p_m+1}^m\}$ $(m = 1, 2)$, the Jacobian $\boldsymbol{J}_1$ is represented as

$$\boldsymbol{J}_1 = \begin{bmatrix} \frac{\partial \boldsymbol{x}}{\partial \xi^1} & \frac{\partial \boldsymbol{y}}{\partial \xi^1} \\ \frac{\partial \boldsymbol{x}}{\partial \xi^2} & \frac{\partial \boldsymbol{y}}{\partial \xi^2} \end{bmatrix}, \tag{12}$$

and the Jacobian $\boldsymbol{J}_2$ may be written as

$$\boldsymbol{J}_2 = \begin{bmatrix} \frac{\partial \xi^1}{\partial \tilde{\xi}^1} & \frac{\partial \xi^2}{\partial \tilde{\xi}^1} \\ \frac{\partial \xi^1}{\partial \tilde{\xi}^2} & \frac{\partial \xi^2}{\partial \tilde{\xi}^2} \end{bmatrix} = \begin{bmatrix} \frac{\xi_{i+1}^1 - \xi_i^1}{2} & 0 \\ 0 & \frac{\xi_{i+1}^2 - \xi_i^2}{2} \end{bmatrix}, \tag{13}$$

if the mapping from the Gauss quadrature domain $[-1, 1]$ to the NURBS parametric domain $[\xi_i^1, \xi_{i+1}^1) \times [\xi_i^2, \xi_{i+1}^2)$ is linear.

When the IGA is used, it is beneficial to replace the design variables from the element density to the control point density. The element density is represented by the density at the element center $\rho_n(ic)$, which can be calculated by

$$\rho_{e_i} = \rho_n(ic) = \sum_{j \in \boldsymbol{c}_i} N_{ij}(ic)\rho_{n_{ij}}, \tag{14}$$

where ρ_n denotes the control point density, ic represents the center of the ith element, $\boldsymbol{c}_i$ denotes the control point set which influences element i, c_{ij} is the jth control point of $\boldsymbol{c}_i$, and $N_{ij}(ic)$ is the NURBS basis function of control point c_{ij} corresponding to the center of the ith element. Fig. 2 gives an illustration of an element and its corresponding control points, which is based on a NURBS patch [0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1] × [0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1].

According to Eq. (14), the Young's modulus $E_e(\rho_n)$ is represented as:

$$E_e(\rho_n) = E_e(\rho_n(ic)) = E_{min} + \rho_n(ic)^s(E_0 - E_{min}), \tag{15}$$

where $\rho_n(ic)$ is a combination of NURBS basis functions at the element center. The sensitivity of the objective function c with respect to a control point for minimum compliance cases is written as

$$\frac{\partial c}{\partial \rho_{n_i}} = \sum_{j \in \boldsymbol{e}_i} \frac{\partial c}{\partial \rho_{e_{ij}}} \frac{\partial \rho_{e_{ij}}}{\partial \rho_{n_i}} = \sum_{j \in \boldsymbol{e}_i} -s\rho_{e_{ij}}^{s-1}(E_0 - E_{min})\boldsymbol{u}_{e_{ij}}^{\mathrm{T}} \boldsymbol{k}_{e_{ij}} \boldsymbol{u}_{e_{ij}} \frac{\partial \rho_{e_{ij}}}{\partial \rho_{n_i}}, \tag{16}$$

where ρ_{n_i} denotes the variable (i.e., density) of the ith control point, $\boldsymbol{e}_i$ is the element set on which the ith control point influences, and e_{ij} is the jth element of $\boldsymbol{e}_i$. For the compliant mechanism, the $\boldsymbol{u}_{e_{ij}}^{\mathrm{T}}$ should be replaced by $\boldsymbol{u}_{de_{ij}}^{\mathrm{T}}$. Fig. 3 illustrates the relationship between a control point and its corresponding elements, which is based on the same NURBS patch as Fig. 2.

According to Eq. (14), the sensitivity of ρ_{e_i} with respect to a design variable ρ_{n_k} is calculated by

$$\frac{\partial \rho_{e_i}}{\partial \rho_{n_k}} = N_k(ic). \tag{17}$$

Based on the above equation, the $\frac{\partial \rho_{e_{ij}}}{\partial \rho_{n_i}}$ in Eq. (16) can be obtained.

Due to the ITO-SIMP based on control points, we prefer to NURBS filter rather than conventional distance-based filter in this work. The differences between the conventional distance-based filter and NURBS

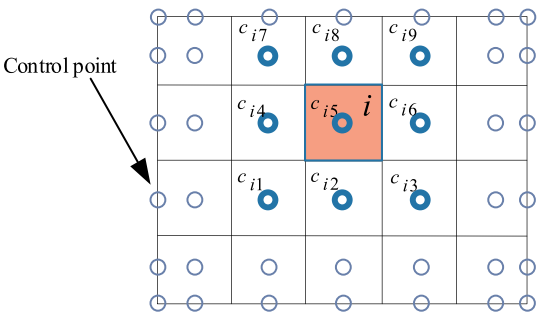


Fig. 2. Illustration of the relation between an element i and its control points. ($p_1 = p_2 = 2$).

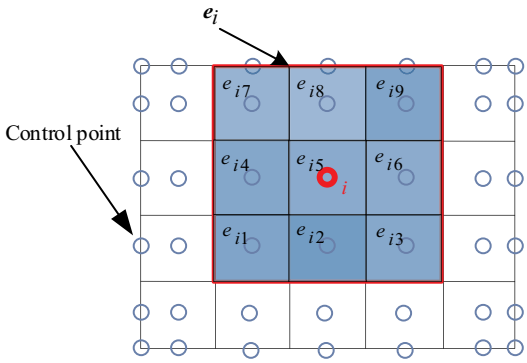


Fig. 3. Illustration of the relation between a control point i and its corresponding elements. ($p_1 = p_2 = 2$).

filter are mainly present in 3 aspects:

(1) Weight function. The weight function of the conventional filter is a distance based linear hat function controlled by the filter radius, but that for the NURBS filter is the NURBS basis function with piecewise polynomials. In the conventional distance-based filter, the weight value depends on distance between adjacent density variables, normally represented based on finite elements, e.g., element node or element center. In the NURBS filter, the weight value depends on the NURBS basis functions as shown in Eq. (14).
(2) Influence region. Taking the place of conventional axial-symmetric circular filter region controlled by filter radius, the influence region of NURBS filter is a rectangle. When the number of knot intervals increases (i.e., the design resolution increases), the length of knot interval in physical space decreases. We can enlarge the filtering region by increasing the order of the NURBS filter, which can be reflected in the comparison between Figs. 3 and 4.
(3) Resolution constraint. To prevent the numerical instability appearance such as checkerboard in optimized process, the filter size should be larger than the element size in the conventional distance-based filters. As for NURBS filter, the support domain of B-spline basis functions must contain multiple elements, more details can be found in [57].

In Eqs. (16) and (17), the NURBS basic functions act as weight functions, i.e., the sensitivity value is calculated by the spatial average of the adjacent NURBS basis functions, in which the influence region is a rectangular region, e.g., the $(p_1 + 1) \times (p_2 + 1)$ knot spans as shown in Fig. 3. Therefore, we can use different p_1 and p_2 for IGA computation and the NURBS filter as long as the NURBS spans keep constant, meaning that we can adjust the filtering region by increasing/decreasing the NURBS order of the filter without changing the NURBS order of the IGA. Taking Fig. 3 as an example, the influence regions corresponding to higher order ($p_1 = p_2 = 3$) is shown in Fig. 4. It should

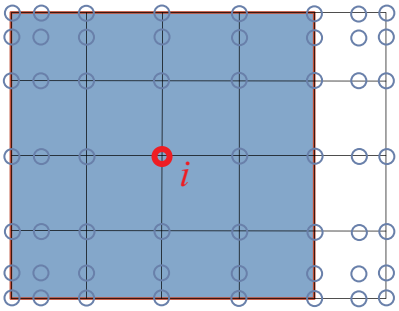


Fig. 4. Influence region comparison of higher NURBS orders ($p_1 = p_2 = 3$) compared to Fig. 3.

be noted that the control points of the IGA used in the structural response analysis may be different from control points of the NURBS filter which is associated with the design variables. In the IGA, we just need the element density to compute the Young's modulus as Eq. (15), and the element density can be interpolated by the shape functions of the NURBS filter as Eq. (14), so these two NURBS orders can be different. With the usage of NURBS filter, the defects of ITO-SIMP including mesh-dependency, local minima and checkboard can also be solved.

3. Model of high-efficiency isogeometric topology optimization

To reduce the computational cost of ITO, three parts of high-efficiency ITO including multilevel mesh, MGCG and local-update strategy will be proposed in this section from different aspects.

3.1. . Multilevel mesh approach

To accelerate the convergence rate in ITO, a multilevel mesh method is proposed, which solves the ITO in a coarse mesh first and then maps the optimized result to a finer mesh as an initial design, and this procedure can be repeated for multiple levels to further increase the efficiency. In ITO-SIMP, the mesh scale determines the number of design variables (i.e., control-point densities). The relationship between control point number and mesh scale is shown as follows:

$$n = \prod_{m=1}^{d_p} (N_m + p_m), \tag{18}$$

where d_p denotes the dimension of the parameter space, p_m indicates the polynomial degrees for mth direction. N_m is the number of elements for mth direction and n is the total number of control points. Fig. 5 illustrates the positional relation between control points and elements, which is based on a NURBS patch [0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1] × [0, 0, 0, 0.25, 0.5, 0.75, 1, 1, 1].

The design variables play an important role during the ITO, the same as that in conventional TO. Large number of design variables will result in huge computational cost and bring huge difficulty to solve the large-scale problems. Therefore, reducing the mesh scale can improve the efficiency of ITO. On the other hand, if the mesh is too coarse, it is difficult to obtain an accurate density distribution and a clear boundary. To balance coarse and fine meshes, multilevel mesh method provides a better choice.

With mesh from coarse to fine, the multilevel mesh method divides the ITO process into several levels. Firstly, the ITO is performed on a coarse mesh with small number of elements, in which the iteration can converge quickly and occupy only small amount of computational cost. By refining the mesh, the density field of the coarser mesh can be mapped to a refined mesh as an initial density field. In terms of the practical requirement, we can repeat above procedure several times for multiple levels. The final ITO result is obtained from the finest mesh.

In multilevel mesh method, the computational cost on coarser mesh only occupies a little part of whole, while the major part is taken by the

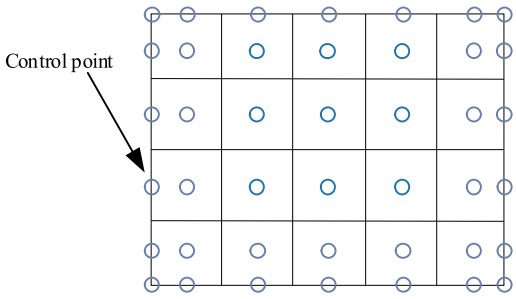


Fig. 5. Illustration of the positional relation between control points and elements. ($p_1 = p_2 = 2$).

computation on the finest mesh. Through multilevel mesh method, the ITO obtains an initial density field and converges much faster on the finer meshes, which will reduce significant computational time.

How to inherit the density field from coarser mesh to finer mesh is the core of multilevel mesh approach. In this work, we use h-refinement, for which the design domain subdivision actually can be regarded as knot insertion. When the coarse mesh is subdivided into a fine mesh, which is geometrically and parametrically identical to the original mesh, and the number of control point increases accordingly. Take univariate B-spline as example, the original knot vector is $\Xi = [\xi_1, \xi_2, ..., \xi_{n_o+p+1}]$, whose control point density is $\boldsymbol{\rho} = [\rho_1, \rho_2, ..., \rho_{n_o}]$, the after-subdivision knot vector is $\bar{\Xi} = [\bar{\xi}_1, \bar{\xi}_2, ..., \bar{\xi}_{n_a+p+1}]$, whose control point density is $\bar{\boldsymbol{\rho}} = [\bar{\rho}_1, \bar{\rho}_2, ..., \bar{\rho}_{n_a}]$, n_o and n_a are the number of control points in original mesh and after-subdivision mesh, respectively. The after-subdivision control point density $\bar{\boldsymbol{\rho}}$ is formed from linear combinations of the original control point density

$$\bar{\boldsymbol{\rho}} = \boldsymbol{T}^{\boldsymbol{p}}\boldsymbol{\rho}, \tag{19}$$

where $\boldsymbol{T}^{\boldsymbol{p}}$ is a $n_o \times n_a$ matrix. For $p = 0$

$$T_{ij}^0 = \begin{cases} 1, & \bar{\xi}_i \in [\xi_j, \xi_{j+1}) \\ 0, & \text{otherwise} \end{cases}, \tag{20}$$

for the $p = 1, 2, 3...$, they are defined by

$$T_{ij}^{q+1} = \frac{\bar{\xi}_{i+q} - \xi_j}{\xi_{j+q} - \xi_j} T_{ij}^q + \frac{\xi_{j+q+1} - \bar{\xi}_{i+q}}{\xi_{j+q+1} - \xi_{j+1}} T_{ij+1}^q, (q = 0, 1, 2, \cdots, p-1). \tag{21}$$

In this paper, an element is halved in each direction, therefore, a 2D element is subdivided into 4, and a 3D element is divided into 8. In each direction, uniform meshes are utilized, the new control points are inserted in middle of each mesh, and the after-subdivision knot vector is

$$\bar{\Xi} = \left[\bar{\xi}_1, \bar{\xi}_2, \cdots, \bar{\xi}_{n_a+p+1}\right], \bar{\xi}_i = \begin{cases} \xi_1, i = 1, 2 \cdots, p+1 \\ \frac{1}{2}\left(\xi_{p+\frac{(i-p)}{2}} + \xi_{p+1+\frac{(i-p)}{2}}\right) \\ , i = p+2, p+4, \cdots \\ \xi_{p+\frac{(i-p)+1}{2}}, i = p+3, p+5, \cdots \\ \xi_{n_o+p+1}, i = n_a+1, n_a+2, \cdots, n_a+p+1 \end{cases}. \tag{22}$$

The after-subdivision NURBS basis function can be calculated by Eq. (2). The h-refinement can be performed in each dimension to obtain new control points, and Eq. (19) is used to obtain the corresponding control point density distribution for both the 2D and 3D cases. Fig. 6 shows the transformation of NURBS basis function in h-refinement progress for a 2D case. The flowchart of multilevel mesh algorithm is shown in Fig. 7.

3.2. . Multigrid conjugate gradient method (MGCG)

To solve the equations in ITO, the MGCG (Multigrid Conjugate Gradients method) [54,63] which has successfully solved equations of FEA is utilized in this work to calculate the displacement vector $\boldsymbol{U}$ of equations $\boldsymbol{KU} = \boldsymbol{F}$. MGCG is a high-efficiency iterative method to solve large-scale linear equations, which is essentially a preconditioned conjugate gradient (PCG) method with the multigrid preconditioner.

3.2.1. Preconditioned conjugate gradient method (PCG)

In IGA, the equations $\boldsymbol{KU} = \boldsymbol{F}$ has similar properties as that of FEA. Therefore, the coefficient matrix $\boldsymbol{K}$ must be symmetric, positive definite and sparse. The algorithm of PCG with a preconditioning matrix $\boldsymbol{M}$ to reduce the total execution time can be described as Table 1.

In Algorithm 1, $\boldsymbol{M}$ is the preconditioning matrix, $\boldsymbol{p}_i$ is search direction vector, $\boldsymbol{r}_i$ is the residual vector, and ε is the convergence tolerance

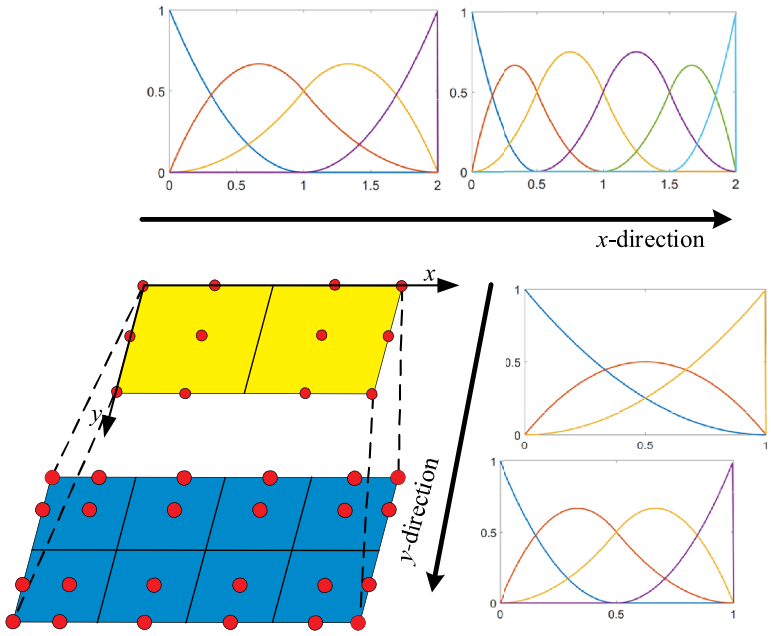


Fig. 6. The illustration of *h*-refinement in multilevel mesh method.

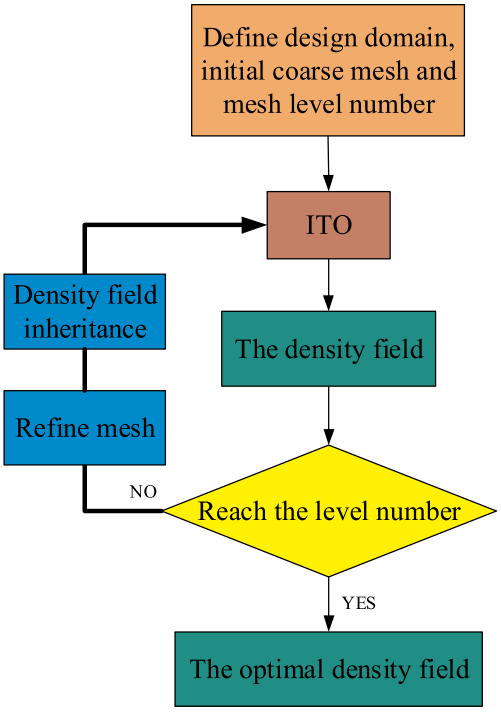


Fig. 7. The flowchat of multilevel mesh method.

(a small value, e.g., 10^{-6}). When the residuals $||\boldsymbol{r}_i|| \leq \varepsilon$, the iteration stops. The convergence rate of PCG is given by [54]

$$||\boldsymbol{U} - \boldsymbol{U}_i||_{\boldsymbol{K}^{-1}} \leq ||\boldsymbol{U} - \boldsymbol{U}_0||_{\boldsymbol{K}^{-1}} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^i, \tag{23}$$

where i is the iteration number, κ is the condition number of matrix $\boldsymbol{K}$. If $\kappa \gg 1$, the convergence rate will be very slow. Therefore, the multigrid preconditioner $\boldsymbol{M}$ is utilized in the iterations, which makes $\kappa(\boldsymbol{M}^{-1}\boldsymbol{K}) \ll \kappa(\boldsymbol{K})$ to improve the convergence rate.

3.2.2. Multigrid (MG) approach

Multigrid (MG) is a multilevel iterative method to solve the linear system of equations, which divides the grids into different scales and the exact solution is computed at the coarsest grid corresponding to the smallest scale. Here we denote the finest level as $l = 1$, whereas $l = L$ corresponds to the coarsest level, which is different from the multilevel mesh approach that $l = 1$ for the coarsest mesh level. A series of cycling

Table 1
Algorithm implementation for PCG.

Algorithm 1
Input: $\boldsymbol{K}$, $\boldsymbol{M}$, $\boldsymbol{F}$, $\boldsymbol{U}_0$
Step1: Initialization
$\boldsymbol{r}_0 = \boldsymbol{F} - \boldsymbol{K}\boldsymbol{U}_0$, $\boldsymbol{z}_0 = \boldsymbol{M}^{-1}\boldsymbol{r}_0$
$\rho_0 = \boldsymbol{r}_0^T \boldsymbol{z}_0$, $\boldsymbol{p}_1 = \boldsymbol{r}_0$, $i = 1$
Step2: Circulation calculation
While $||\boldsymbol{r}_i|| > \varepsilon$
$\boldsymbol{w} = \boldsymbol{K}\boldsymbol{p}_i$; $\alpha_i = \frac{\rho_{i-1}}{\boldsymbol{p}_i^T \boldsymbol{w}}$
$\boldsymbol{U}_i = \boldsymbol{U}_{i-1} + \alpha_i \boldsymbol{p}_i$; $\boldsymbol{r}_i = \boldsymbol{r}_{i-1} - \alpha_i \boldsymbol{w}$
$\boldsymbol{z}_i = \boldsymbol{M}^{-1}\boldsymbol{r}_i$; $\rho_i = \boldsymbol{r}_i^T \boldsymbol{z}_i$
$\beta_i = \frac{\rho_i}{\rho_{i-1}}$; $\boldsymbol{p}_{i+1} = \boldsymbol{z}_i + \beta_i \boldsymbol{p}_i$,
$i = i + 1$.
End while
Output: $\boldsymbol{U}_i$

Table 2
Algorithm implementation for multigrid V-cycle strategy.

Algorithm 2 MG_V-cycle ($\boldsymbol{F}^{(1)}$,$\boldsymbol{K}^{(1)}$)
Input: $\boldsymbol{F}^{(1)}$,$\boldsymbol{K}^{(1)}$
Setting the initial guess: $\boldsymbol{U}^{(1)} = 0$,
For $l = 1$ to L-1
smooth ($\boldsymbol{K}^{(l)}$, $\boldsymbol{U}^{(l)}$,$\boldsymbol{F}^{(l)}$),
$\boldsymbol{r}^{(l)} = \boldsymbol{F}^{(l)} - \boldsymbol{K}^{(l)}\boldsymbol{U}^{(l)}$,
$\boldsymbol{F}^{(l+1)} = \boldsymbol{P}^T \boldsymbol{r}^{(l)}$,
$\boldsymbol{K}^{(l+1)} = \boldsymbol{P}^T \boldsymbol{K}^{(l)} \boldsymbol{P}$,
$\boldsymbol{U}^{(l+1)} = 0$.
End for
Solve $\boldsymbol{U}^{(L)} = (\boldsymbol{K}^L)^{-1}\boldsymbol{F}^L$
For $l = L$-1 down to 1
$\boldsymbol{U}^{(l)} = \boldsymbol{U}^{(l)} + \boldsymbol{P}\boldsymbol{U}^{(l+1)}$,
smooth ($\boldsymbol{K}^{(l)}$, $\boldsymbol{U}^{(l)}$,$\boldsymbol{F}^{(l)}$).
End for
Output $\boldsymbol{U}^{(1)}$

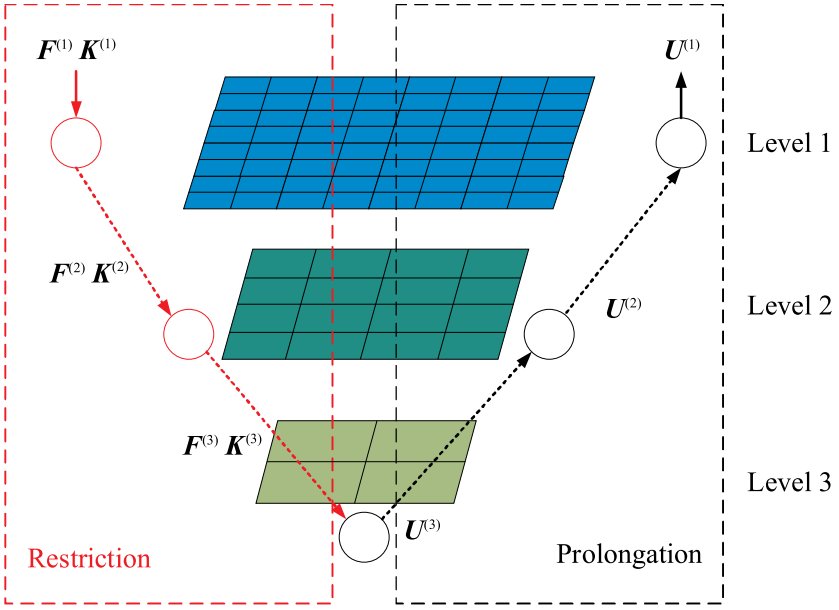


Fig. 8. The depiction of multigrid V-cycle strategy with 3 levels.

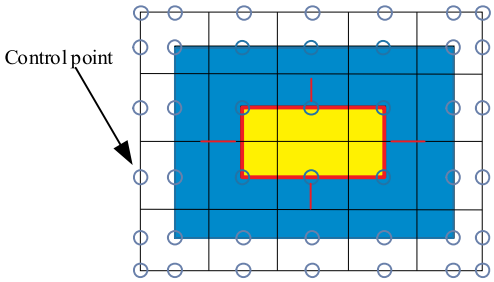


Fig. 9. The boundary moves 1 layer toward outside area (diffused layer number is 1).

Table 3
Algorithm implementation for design domain division by diffused-layer number.

Algorithm 3
Choose a diffused layer number D_{num}
$\Omega_n^{\text{original}} = \Omega_n$,
For $\Omega_n^{\text{original}}(n_1,n_2,n_3\ldots n_k\ldots)$
  Find the coordinate $n_k(i,j,k)$ of control point n_k
  Put the control points with the coordinate $n_c[(i - D_{num})$: $(i + D_{num})$, $(j - D_{num})$: $(j + D_{num})$,$(k - D_{num})$: $(k + D_{num})]$ into Ω_n
End for

strategies can be used in the MG approach and the V-cycle strategy [54,64] is the most common one and used in this work. The V-cycle strategy discretizes the matrix $\boldsymbol{K}$ of equations $\boldsymbol{KU}=\boldsymbol{F}$ into L levels of resolution, denoted as $\boldsymbol{K}^{(1)}, \boldsymbol{K}^{(2)},\ldots, \boldsymbol{K}^{(L)}$, where the finest grid and coarsest grid are level 1 and level L, respectively. The algorithm implementation of multigrid V-cycle strategy is shown as Table 2:

In Algorithm 2, $\boldsymbol{P}$ is the prolongation operator, mapping the coarse grid solution to the fine grid. The transpose of the prolongation $\boldsymbol{P}^T$ as the restriction operator, maps the fine grid residual to the coarse grid. Fig. 8 shows a depiction of multigrid V-cycle strategy, in which $L = 3$.

The smoothing process, smooth $(K^{(l)},U^l,F^l)$, is typically based on a stationary iterative method such as Gauss-Seidel or Jacobi, aiming to reduce the oscillatory error in the solution. The smoothed low order components can be effectively approximated on a coarser grid since the smoother removes the high frequency components of the residua. In

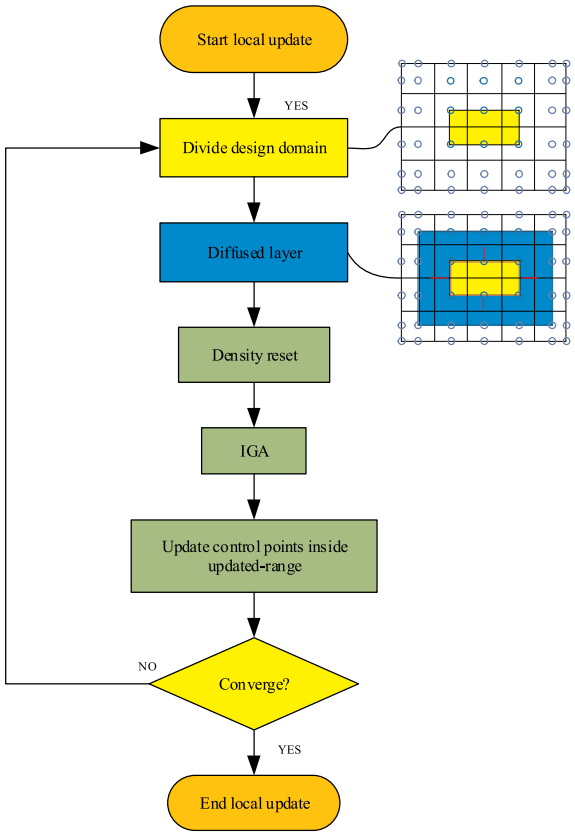


Fig. 10. The flowchart of the local-update strategy.

this work, we choose Jacobi as smoother, shown as follows:

$$\boldsymbol{U}^{(l)} = \boldsymbol{U}^{(l)} + \boldsymbol{S}^{-1}(\boldsymbol{F}^{(l)} - \boldsymbol{K}^{(l)}\boldsymbol{U}^{(l)}), \tag{24}$$

where $\boldsymbol{S}^{-1}$ is based on the damped Jacobi method, represented as follows:

$$\boldsymbol{S}^{-1} = \omega\boldsymbol{D}^{-1}, \tag{25}$$

ω is the damping factor, $\boldsymbol{D}$ is the diagonal of the matrix $\boldsymbol{K}^{(l)}$. More detail can be found in [64,65].

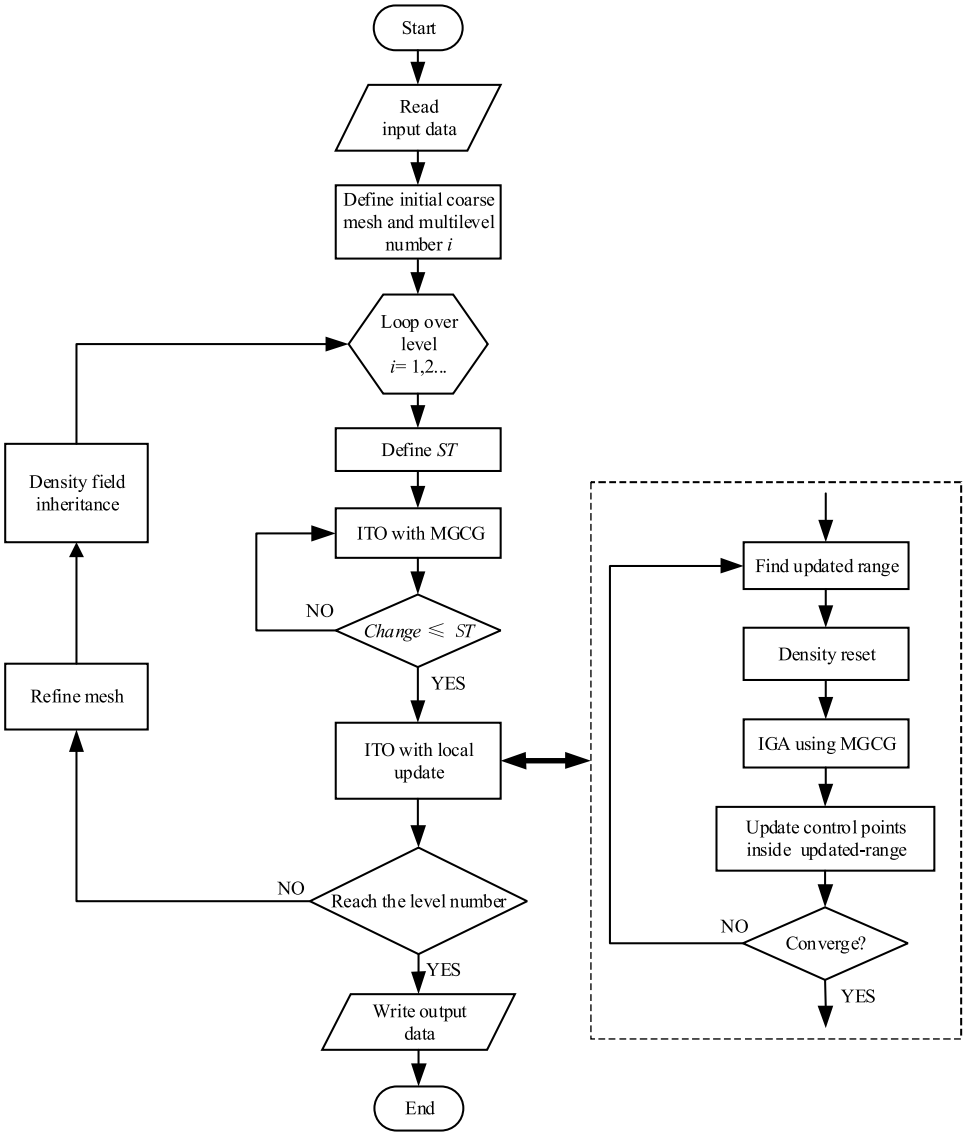


Fig. 11. The flowchart of HITO.

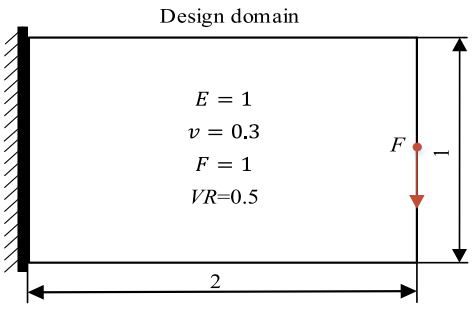


Fig. 12. The cantilever beam benchmark.

Fig. 13. The final structure of CITO.

Combined with PCG, MG joins in PCG as a preconditioner, replacing $\boldsymbol{z}_i = \boldsymbol{M}^{-1}\boldsymbol{r}_i$ by calling MG_V-cycle($\boldsymbol{r}_i$,$\boldsymbol{K}$), which makes MGCG greatly improve the efficiency of solving large linear equations.

3.3. . Local-update strategy

Local-update strategy, reducing the design variables in each iteration, is the last and the most important efficiency improvement in this work. In each iteration of conventional ITO, all design variables must be updated. Referred to [55], it will not bring significant loss in performance of solutions but save considerable computational cost by reducing some design variables. Compared to full design variables, reducing some design variables leads to fewer iterations. To reduce the computational cost, a local-update strategy is proposed to realize a high-efficiency ITO method.

The core of the local-update strategy is to identify the updated range. To keep ITO in stabilization and accuracy, the updated range should include both the solid region and their neighbor region. Therefore, the key of local-update strategy is to find such regions.

Since ITO-SIMP using the control point density as design variables, it is easy to define a criterion to divide the design domain into two subdomains by a threshold density value: (1) the subdomain with

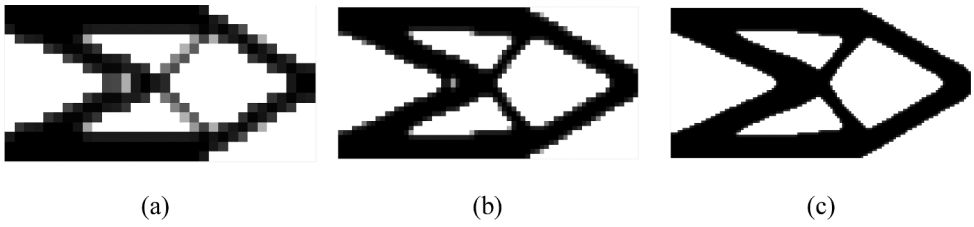


Fig. 14. The optimization results of each level: (a) 32 × 16, (b) 64 × 32, (c) 128 × 64.

Table 4
Efficiency comparison between CITO and ITO with multilevel mesh.

	CITO	Multilevel mesh
Compliance	64.72	64.05
Number of iterations	353	97+15+115
Computational time(s)	169.25	63.47

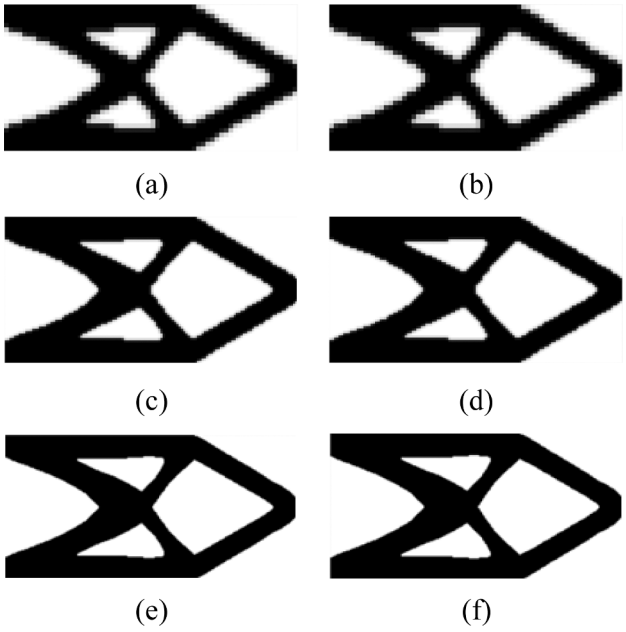


Fig. 15. The final structures of ITO with ICCG and MGCG in different meshes: (a) ITO with ICCG in 64 × 32 mesh, (b) ITO with MCCG in 64 × 32 mesh, (c) ITO with ICCG 128 × 64 mesh, (d) ITO with MCCG in 128 × 64 mesh, (e) ITO with ICCG in 256 × 128 mesh, and (f) ITO with MCCG in 256 × 128 mesh.

control point densities smaller than the threshold density and (2) the subdomain with control point densities larger than or equal to the threshold density. In terms of the threshold density, we can extract the boundaries that separate the subdomains. Only the control points inside the later subdomain are updated. The math description is as follows:

$$\forall \ \rho_n > \rho_t, \ \ n \in \Omega_n, \tag{26}$$

where ρ_t is threshold value of control point density, Ω_n is the set of updated control points, and n is the control point ID. The ρ_t is smaller, the updated range is bigger. If ρ_t is equal to 0, all control points will be updated. Vice versa, ρ_t is bigger, the updated range is smaller. If ρ_t is equal to 1, no control point will be updated.

Besides the threshold density value, another important parameter is the diffused layer number, which expands the neighbor region to the outside area. The other advantage of diffused layer is to improve optimization accuracy, with more design variables and lower efficiency. When the diffused layer number is zero, only the control points inside the red boundary of Fig. 9 are updated. If the diffused layer number is 1, the boundary will move 1 layer toward outside area, as shown in Fig. 9, larger diffused layer number means more control points will be included in the updated range. The mathematical description is as follows:

$$n \in \Omega_n \rightarrow \ \ n_c(i, j, k), \tag{27}$$

$$\forall \ \ n \in n_c[(i - D_{num}){:}\ (i + D_{num}),$$
$$(j - D_{num}){:}\ (j + D_{num}),\ (k - D_{num}){:}\ (k + D_{num})] \rightarrow \ \ n \in \Omega_n, \tag{28}$$

where Ω_n is the set of updated control points, n is the control points ID, $n_c(i, j, k)$ represents the coordinate of control points, and D_{num} is the diffused layer number. For the 2D case, the z-dimension is ignored. The diffused-layer algorithm implementation is listed in Table 3. It is worth noting that the index $[(i - D_{num}){:}\ (i + D_{num}), (j - D_{num}){:}\ (j + D_{num}), (k - D_{num}){:}\ (k + D_{num})]$ cannot exceed coordinate matrix dimension.

Furthermore, two approaches are proposed to improve the efficiency of local-update strategy. In conventional ITOs, an approximate optimal structure can be obtained in a few iterations but the optimization convergence requires much more iterations. When the structure stabilizes around a certain shape, it is a good opportunity to start the local-update strategy. Therefore, when to start the local-update strategy is of great importance in the ITO problem. The parameter, *change*, is utilized as measure of convergence, defined as

$$change = \|\boldsymbol{\rho}^{i+1} - \boldsymbol{\rho}^{i}\|_{\infty} \ , \tag{29}$$

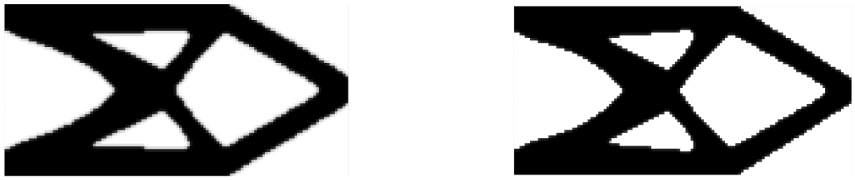
where ρ is the design variable vector, and i denotes the ith iteration. The value of *change* general trend to descend when the iteration goes toward convergence.

Therefore, we define a startup-parameter *ST*, defined as the threshold of *change*, to determine the moment of local-update strategy startup. When the ITO has obtained an approximate structure, i.e., the *change* is smaller than *ST*, local-update strategy starts to work.

When the volume fraction is small, the control points with small density occupy most of computational resource but have less influence on the accuracy of computation and design. The other disadvantage is the grey elements, which bring difficulties to generate a black-and-white result in ITO. Thus, it is helpful to reset the small density to the zero directly under certain conditions. The detail approach is: defining a small-density threshold ρ_s, the control point density below the threshold can be set to zero. This density processing approach has two significant benefits: one is decreasing grey elements while increase solid elements, and the other is accelerating the convergence rate. Since the

Table 5
Efficiency comparison between ITO with ICCG and MGCG in different meshes.

case	Compliance (ICCG)	Compliance (MCCG)	Computational time(s) per iteration (IGCG)	Computational time(s) per iteration (MGCG)
64 × 32	67.44	67.44	0.22	0.27
128 × 64	64.72	64.72	1.27	1.28
256 × 128	63.59	63.51	8.32	5.70



(a) (b)

Fig. 16. The optimization results of (a) CITO and (b) ITO with local-update strategy.

Table 6
Efficiency comparison between CITO and ITO with local-update strategy.

	CITO	Local-update strategy
Compliance	64.72	65.70
Number of iterations	353	128
Computational time(s)	169.25	68.36

volume fraction is constant, when the densities of small-density elements are set to zero, more material can be allocated to solid domain, which will make volume of solid domain perfectly match the volume fraction constraint and increase the accuracy of optimal result. To guarantee the optimal structure fulfill the volume constraint, the small-density elements reset only implements before density update process. It is worth noting that it will result in instable iterations if the small-density threshold is bigger than (1-*VF)*, where *VF* is volume fraction.

Based on the above discussion, the flowchart of local-update strategy is given in Fig. 10:

4. Algorithm implementation

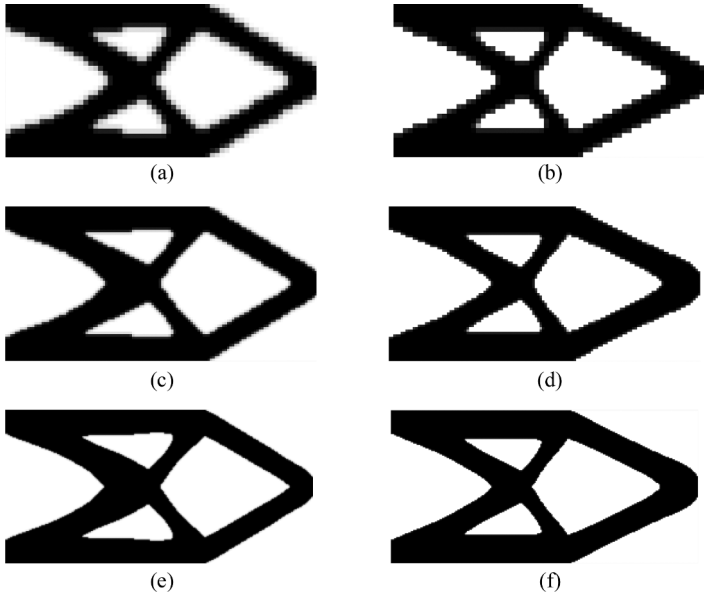
The high-efficiency isogeometric topology optimization (HITO) using Matlab (Natick, Massachusetts, USA) is developed in this work to solve the minimum compliance problem and compliant mechanism. Fig. 11 illustrates the flowchart of the HITO. The sensitivity filter is the NURBS filter proposed in this work.

Due to multilevel mesh approach is applied, the HITO process can be divided into several levels, in which the MGCG solves the $\boldsymbol{KU} = \boldsymbol{F}$ for $\boldsymbol{U}$ and local-update strategy is utilized. To reduce the computational cost furtherly, the loose convergence criterion can be used on the coarser mesh levels to reduce the convergence time. It is noted that the first ITO on initial coarse mesh should be able to obtain the approximate structure, therefore the initial mesh should not be too coarse. The parameters in local-update strategy of each mesh level can be different, how to choose the suitable parameter value will be discussed in Section 5.2. For simplicity, the HITO in this work use the same parameters in local-update strategy of every mesh level.

5. Numerical examples

In order to demonstrate the acceleration efficiency of HITO, three benchmark examples for minimum compliance design and a 3D compliant mechanism example are examined in this section. All examples are run on a desktop computer with CPU Intel core i7-6700 K of 4.00 GHz, RAM of 16.0GB, and software environment MATLAB.

In Section 5.1, we test a classical cantilever beam to compare the computation efficiency of HITO with that of a conventional ITO (CITO) in three aspects consisting of multilevel mesh, MGCG and local update strategy, and study the accelerated efficiency of HITO on different meshes. In Section 5.2, we discuss the effect of the local-update strategy parameters (ST and ρ_s) through MBB beam. In Section 5.3, a quarter annulus is utilized to demonstrate the direct design-to-analysis in IGA. Above 3 examples use the same convergence criterion (the difference of the design variables between adjacent iterations is smaller than 0.01, i.e., $change < 0.01$). In the Section 5.4, a 3D compliant mechanism is used to demonstrate the HITO also perform well in the 3D case.



(a) (b) (c) (d) (e) (f)

Fig. 17. The optimization results of CITO and HITO based on different meshes: (a) CITO with 64 × 32, (b) HITO with 64 × 32, (c) CITO with 128 × 64, (d) HITO with 128 × 64, (e) CITO with 256 × 128, and (f) HITO with 256 × 128.

Table 7
Efficiency comparison between CITO and HITO.

case	Compliance (CITO)	Compliance (HITO)	Computational time(s) (CITO)	Computational time(s) (HITO)	Efficiency value ($time_{CITO}/time_{HITO}$)
64 × 32	67.44	66.66	14.68	9.15	1.60
128 × 64	64.72	64.15	169.25	44.31	3.82
256 × 128	63.71	63.19	2048.00	406.51	5.04

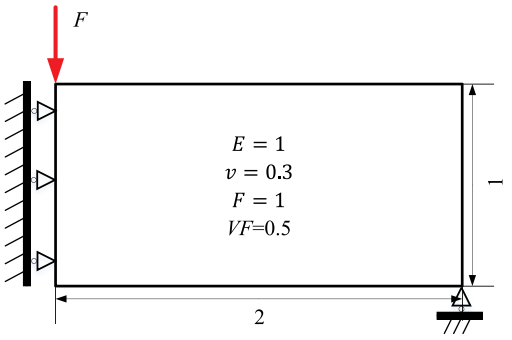


Fig. 18. The design domain and boundary conditions of half MBB beam.

5.1. . Cantilever beam

Fig. 12 illustrates the parameter setting of the 2D cantilever beam benchmark [2], in which the dimension is 2 × 1. An external load *F* is applied to the center of the right-end edge, and the left end of the cantilever beam is clamped.

Table 8
Efficiency comparison between CITO and local-update ITO with different *ST*.

	CITO	local-update ITO			
		ST = 0.05	*ST* = 0.10	*ST* = 0.15	Without *ST*
Compliance	87.07	87.33	87.96	Misconvergence	
Number of iterations	144	75	49		
Computational time(s)	25.78	10.81	7.78		

5.1.1. ITO with multilevel mesh

The design domain is discretized to a mesh of 128 × 64 quadratic NURBS elements, and the volume fraction (*VF*) of optimization is set to 0.5. The final structure of conventional ITO (CITO) is shown in Fig. 13.

For the cantilever beam with 128 × 64 elements, it is easy to be divided into 3 levels in multilevel mesh, the initial mesh is 32 × 16, the middle mesh is 64 × 32, and the final mesh is 128 × 64. The optimization results of each level are shown in Fig. 14.

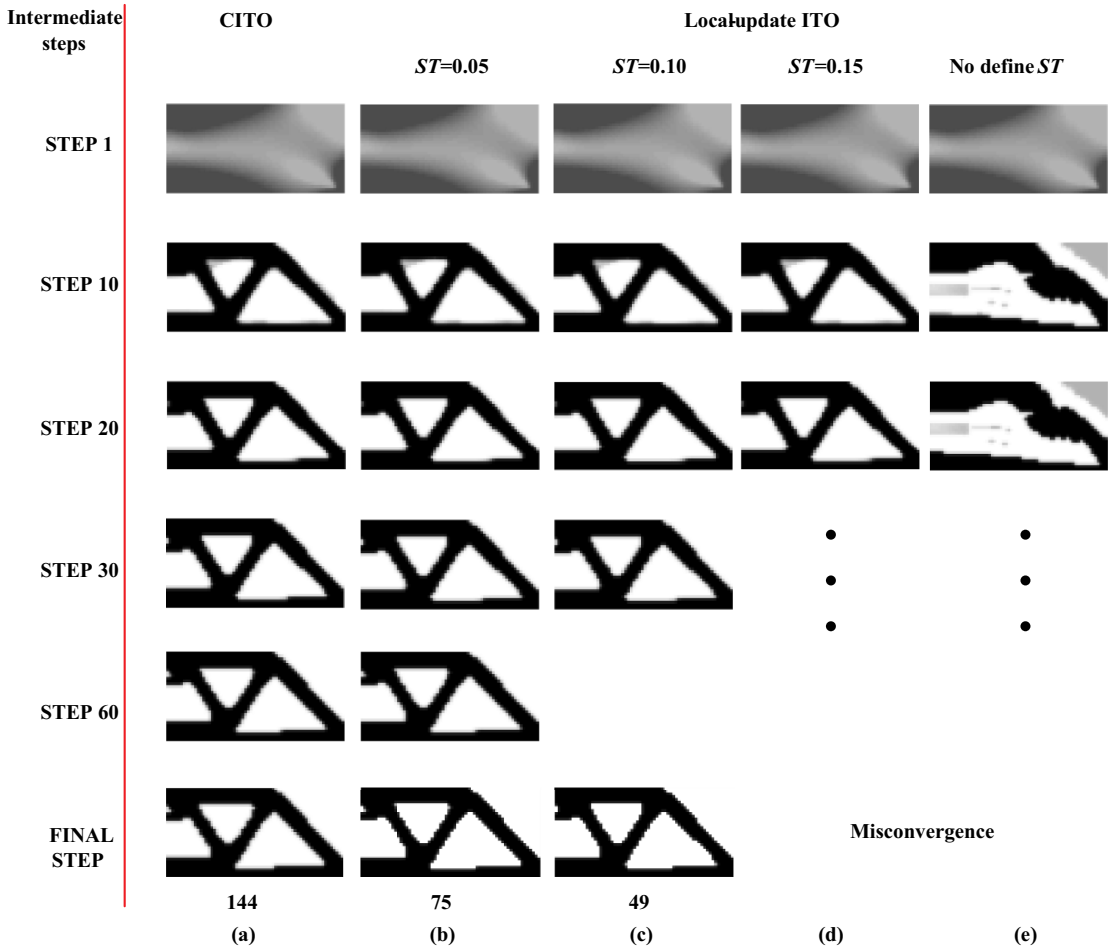


Fig. 19. The intermediate and final optimized structures: (a) for CITO and the rest are local-update ITO with different *ST*: (b) *ST* = 0.05, (c) *ST* = 0.10, (d) *ST* = 0.15, (e) no *ST*.

Fig. 20. The final optimized structures of CITO (a) and local-update ITO with different ρ_s: (b) $\rho_s = 0.1$, (c) $\rho_s = 0.2$, (d) $\rho_s = 0.3$.

Table 9
Efficiency comparison between CITO and local-update ITO with different ρ_s.

	CITO	local-update ITO			
		$\rho_s = 0.1$	$\rho_s = 0.2$	$\rho_s = 0.3$	$\rho_s > 0.4$
Compliance	87.07	87.75	87.82	87.96	Misconvergence
Number of iterations	144	70	57	49	
Computational time(s)	25.78	10.88	8.64	7.78	

Table 4 shows the efficiency comparision between ITO with multilevel mesh and CITO. It is obvious to find that the computational time of ITO with multilevel mesh is much less than that of CITO, demonstrating the advantage of multilevel mesh in efficiency improvement. For multilevel mesh, the iteration number of each mesh level from coarse to fine are 97, 15 and 115 respectively, and the final level with finest mesh occupies a large proportion of computational time. Through the iterations on the coarser mesh, the optimization on the finer mesh can convergence quickly, e.g., only 115 iterations for the finest level, which saves lots of computational time compared to the CITO with 353 iterations.

5.1.2. ITO with multigrid conjugate gradient method (MGCG)

Since MGCG is modified from PCG, this section takes the PCG method, incomplete Cholesky conjugate gradient method (ICCG) [66-68] to compare with MGCG. ICCG is a PCG method with the incomplete Cholesky precondition, which can solve large-scale system of linear equations efficiently. For the ITO, the design domain is discretized by three different mesh scales, which respectively consist of 64×32, 128×64, 256×128 quadrilateral elements, and the final structures of ITO with ICCG and MGCG in differrent meshes are shown in Fig. 15.

Table 5 reports that the ITO with MGCG costs less computational time in each iteration compared with ICCG when the mesh scale is large. For this example, it should be noted that ICCG may be more efficient for small-scale problems such as a mesh of 64×32 elements. With the mesh scale becoming larger, the computational time per iteration of ITO with MGCG is nearly equal to IGCG in the mesh of 128×64 elements, and much smaller than IGCG in the mesh of 256×128 elements, meaning the solving speed of MGCG is faster for such case, which demonstrates the advantages of MGCG for solving large-scale problems.

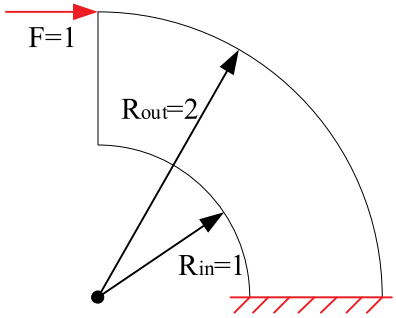


Fig. 22. The design domain and boundary conditions of quarter annulus.

5.1.3. ITO with local-update strategy

In the ITO with local-update strategy, the startup-parameter *ST* is 0.1, the threshold density ρ_t is 0.5, diffused layer number D_{num} is 4. The control point density below 0.2 are set to zero in each iteration. The design domain is discretized with 128×64 quadratic NURBS elements, and the volume fraction is 0.5. The final structure is presented in Fig. 16.

Table 6 represents the efficiency comparison between CITO and ITO with local-update strategy, which shows that the computational time of ITO with local-update strategy is less than half of that used in CITO, as well as a smaller iteration number, demonstrating the high-efficiency of the local-update strategy. Due to the small-density reset, the final result of local-update strategy has fewer grey elements and more solid elements. More discussion about local-update strategy can be found in Sect. 5.2.

5.1.4. HITO

To further discuss the acceleration efficiency of the method proposed in this work, we have modeled the cantilever beam with meshes of 64×32, 128×64, 256×128 quadratic NURBS elements, respectively. The volume fraction (*VF*) of optimization is set to 0.5. The

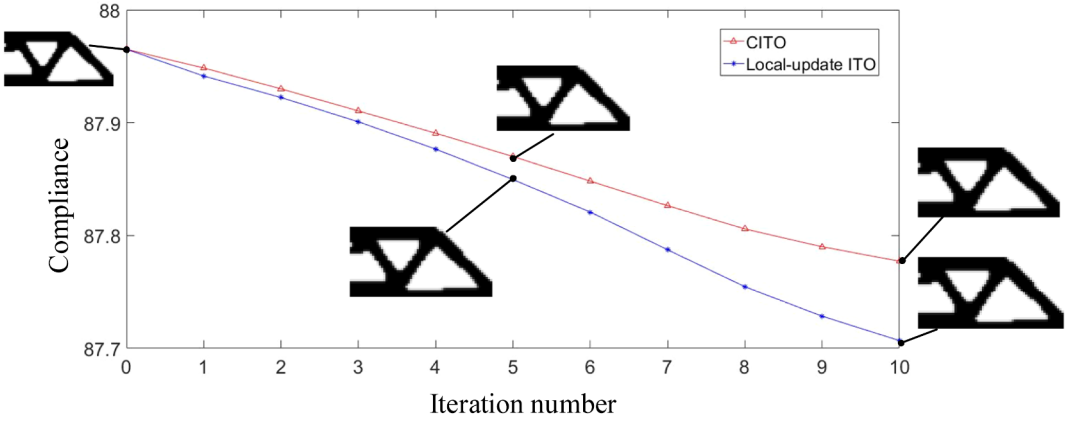


Fig. 21. The results comparison between CITO and local-update ITO at same iteration number.

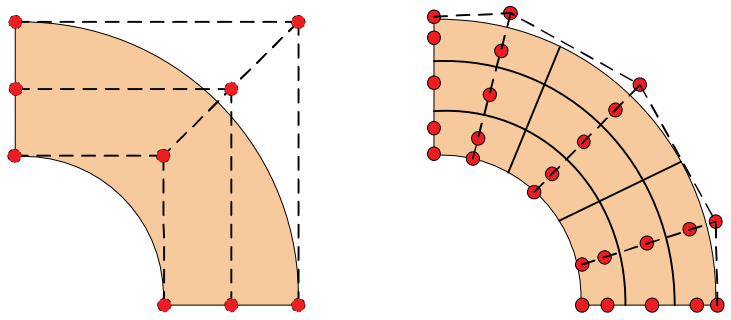


Fig. 23. The control points of quarter annulus with (a) 1 × 1 quadratic NURBS element and (b) 3 × 3 quadratic NURBS elements.

Fig. 24. The final structure of quarter annulus with (a) CITO and (b) HITO.

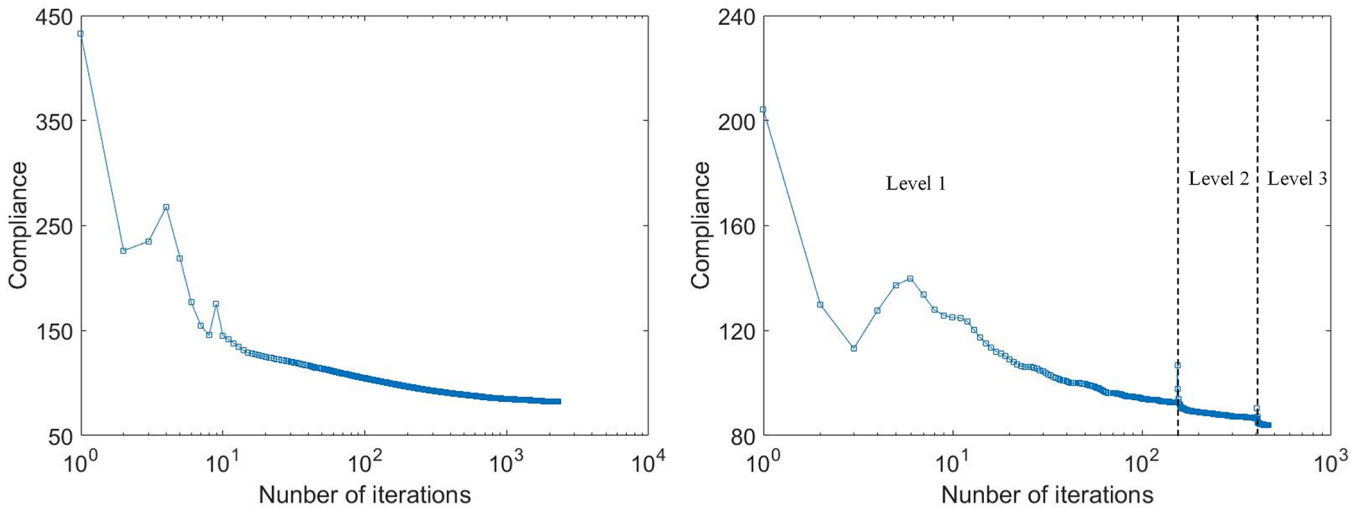


Fig. 25. The convergence history of quarter annulus with CITO (left) and HITO (right).

Table 10
Efficiency comparison between CITO and ITO with local-update strategy.

	CITO	HITO
Compliance	82.22	83.90
Number of iterations	2316	153 + 252 + 63
Computational time (s)	1346.60	90.02

comparison between CITO and HITO on difference mesh are shown in Fig. 17 and Table 7.

From Table 7, we find that the computational time of HITO is smaller than that of CITO for all cases with different scales. The efficiency value ($time_{CITO}/time_{HITO}$) from 1.60 to 5.04 represens the proposed HITO has reduced 37%~80% computational time compared to CITO, and demonstrates the higher efficiency of HITO. Compared with the CITO, HITO reduces more computational time when the mesh scale increases, which shows that HITO has better high-efficiency performance for large-scale problems.

5.2. MBB beam

Fig. 18 shows the half MBB beam, a classical benchmark problem in TO [10,14]. Due to the symmetry, only half MBB beam is modeled and symmetry boundary conditions are applied. This case study aims to discuss the effect of the parameters (*ST* and ρ_s) in local-update strategy.

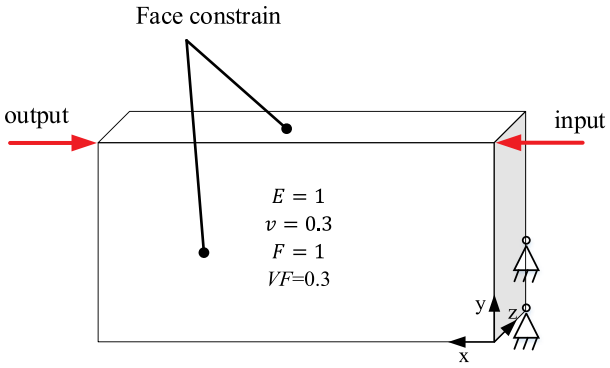


Fig. 26. The design domain and boundary conditions of the 3D compliant mechanism.

5.2.1. Startup-parameter

In ITO, it can be found that the value of *change* is related to the convergence stability: a small *change* value usually means a small stucture change. The startup-parameter *ST*, defined as the threshold of *change*, aims to start local-update strategy when the optimization obtains an approximate structure. Considering different values of *ST*, this section discusses the efficiency of local-update ITO for different cases and compares them with CITO. Local-update strategy is utilized in the MBB beam discretized into 64 × 32 elements, where the threshold value of control point density ρ_t is 0.5 and the diffused layer number D_{num} is 4. The control point density below 0.3 are set to zero in each iteration. Fig. 19 shows the intermediate and final optimized structures corresponding to different values of *ST*.

Table 8 lists the compliance values, the iteration number and computational time for the ITOs shown in Fig. 19, where we can conclude that a suitable *ST* is helpful to improve the efficiency of ITO. With *ST* increasing, the compliance has a little rise, since the earlier start momont will make the local-update start based ona higher-compliance intial structure. If the value of *ST* is too big, it will lead to instable iterations and the optimization cannot converge. When *ST* is not defined, meaning starting local-update strategy at beginning of the ITO, the optimization cannot obtain an approximate structure. If the value of *ST* is too small, the effeciency improvement is non-significant. Therefore, a suitble *ST* has positive effect on efficiency improvement, which plays an important role to reduce the computational time and the number of iterations.

5.2.2. . Small-density threshold

In Sect. 3.3, the small-density threshold ρ_s is proposed to further improve the convergence, which is used to reset the small densities to zero. This density processing approach has two significant benefits: one is decreasing grey elements while increasing solid elements, and the other is accelerating the convergence. Here we will discuss the influence of the small-density threshold on ITO with local-update strategy. Three different thresholds are tested for the half MBB beam and the final optimized structures are shown in Fig. 20, where the startup-parameter *ST* is 0.1, other parameters are the same as that in Section 5.2.1.

Table 11
Efficiency comparison between CITO and ITO in 3D compliant mechanism.

	CITO	HITO
Max displacement	−0.1276	−0.1310
Number of iterations	50	40 + 27
Computational time (s)	581	296

Table 9 shows that the number of iterations and computational time both decrease with the increasing ρ_s ($\rho_s \leq 0.3$), which demonstrates that ρ_s can reduce the computational cost, and the bigger ρ_s reduces more computational cost when ρ_s is in a suitable range. With ρ_s increasing, the value of compliance also has small rise, this is beacause the theoretical optimal density distribution may include the small density, when the ρ_s increseases, more small density are reset, the density distribution is more diverged from the theoretical optimum, the compliance has a little rise. Furthermore, the optimization may not converge when the small-density threshold ρ_s is too large, e.g., $\rho_s > 0.4$. The reasons for the misconvergence include: (1) lacking of design variables, and (2) the volume of elements reset to zero density exceeds the volume constraint. Therefore, ρ_s must be assigned in a suitable range which should be smaller than the volume constraint. Generally, we recommend ρ_s to be a value smaller than 0.8*(1-*VF*) (*VF* is the volume fraction).

5.2.3. . The comparison between CITO and local-update ITO at same iteration number

To further discuss the advantages of local-update strategy, the comparison between CITO and local-update ITO at same iteration number is presented.

In this case, local-update strategy is utilized in the MBB beam discretized into 64 × 32 elements, where the startup-parameter *ST* is 0.1, the threshold value of control point density ρ_t is 0.5 and the diffused layer number D_{num} is 4. The control point density below 0.2 are set to zero in each iteration. When the *change* is smaller than *ST*, local-update strategy starts, meanwhile defining current iteration number as 0. Using CITO and local-update ITO to iterate 10 times respectively, the results of two methods are plotted in Fig. 21.

From Fig. 21, the compliance of local-update ITO decreases more rapidly than CITO, demonstrating the previous method has the better performance. The reason for this is that the small density reset can allocate more density to the solid domain and reduce the grey elements, which also accelerates the convergency.

5.3. Quarter annulus

To demonstrate the advantage of the proposed HITO method, i.e., direct design-to-analysis, a quarter annulus example with curved boundaries is utilized, which is shown in Fig. 22. A concentrated force is horizontally loaded at left-top corner while the bottom edge is fixed.

The control points of quarter annulus with quadratic NURBS elements on differrent meshes are shown in Fig. 23, where the same

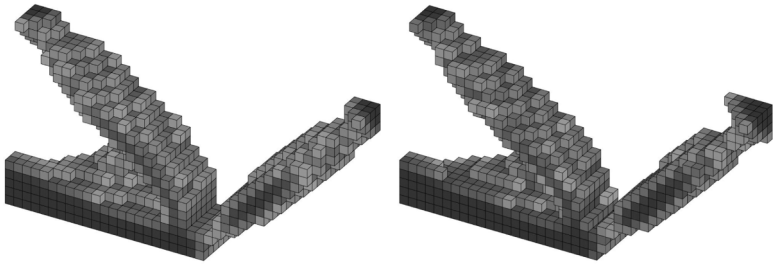


Fig. 27. The final structure of 3D compliant mechanism with CITO (left) and HITO (right).

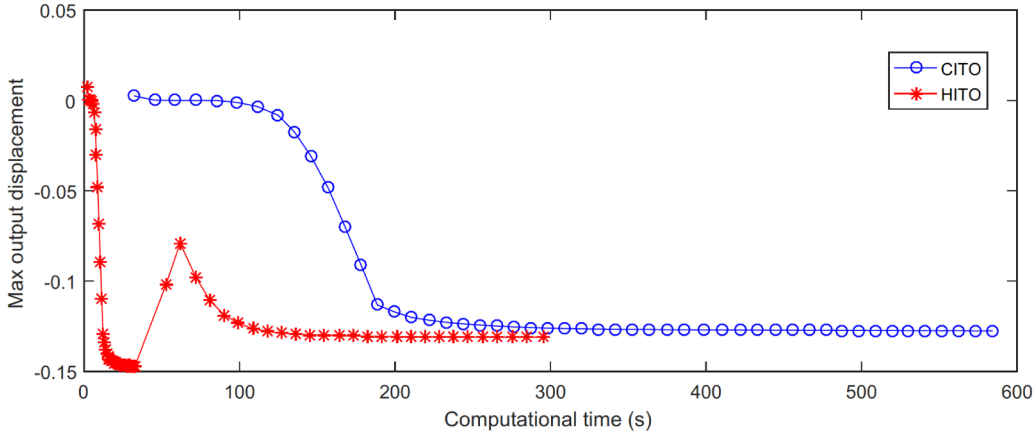


Fig. 28. The convergence history of 3D compliant mechanism with CITO and HITO. Each point on the curves represents a iteration result.

geometries illustrates that no discretization error exists in the IGA.

In this section, the quarter annulus is discretized into 128 × 64 quadratic NURBS elements, the volume fraction is set to 0.5. The parameters in HITO is as follows: the design domain is divided into three mesh scale levels as 32 × 16, 64 × 32 and 128 × 64. The startup-parameter ST is set to 0.1, the threshold value of control point density ρ_t is 0.5, and the diffused layer number D_{num} is 4. The control point densities below 0.2 are set to zero in each iteration. Fig. 24 shows the final structures of quarter annulus by CITO and HITO. It is noted that IGA directly uses the exact CAD model and no spatial discretization error exists, but the visulization here is based on the pixel mesh interpolated by the control points that will cause some boundaries looks non-smooth.

Fig. 25 presents the convergence history of quarter annulus with CITO and HITO, and the efficiency comparison is shown in Table 10. From Fig. 25, we can find that the conpliance of CITO and HITO both tend to decrease during the optimizaiton, and the conpliance of HITO has an oscillation when the mesh level changes. In Table 10, the iteration number of each mesh level in HITO from coarse to fine are 153, 252 and 63 respectively, which is much smaller than that of CITO. Since per iteration time of finest mesh level is much bigger than the coarser mesh levels and the finest mesh level of HITO only costs 63 iterations, the number of iterations of HITO is much less than 2316 of CITO. The proposed HITO can reduce 93% computational time compared with CITO, which demonsrate the high-efficiency of HITO. Note that this curved example is more difficult to converge that other aforementioned examples with regular design domain, especially for a high-standard convergence criterion as 0.01. Apart from the element size and element geometry are different in the curved example, this convergence difficulty may be also caused by the relationship between the control point densities and the element densities, for example, each element density is calculted by interpolating the densities of control points influencing the element as Eq. (14).

5.4. . 3D compliant mechanism

To further demostrate the high efficiency of HITO, a 3D case of compliant mechanism is used in this subsection. Compliant mechanism is a more challenging case, since the design goal is the maximum output displacement rather than minimum compliance, and the convergence of compliant mechanism is more difficult. For the 3d compliant mechanism case (i.e., the force inverter problem) shown in Fig. 26, the input load is defined as the positive horizontal direction while the output is negative. Both the side face and top face are imposed with symmetric constrains, i.e., nodes can only move within the plane.

In this section, the 3D compliant mechanism is discretized into 40 × 20 × 6 quadratic NURBS elements, the volume fraction is set to 0.3. The parameters in HITO is as follows: the design domain is divided into two mesh levels as 20 × 10 × 3 and 40 × 20 × 6. The startup-parameter ST is set to 0.1, the threshold value of control point density ρ_t is 0.5, and the diffused layer number D_{num} is 2. The control point densities below 0.3 are set to zero in each iteration. Since it is more difficult to converge for 3D compliant mechanism than above minimum compliance problems, to avoid a large number of iterations, a different convergence criterion is used as that in [69,70],

$$\frac{\sum_{i=1}^{N}(C_{k-i+1}-C_{k-N-i+1})}{\sum_{i=1}^{N}C_{k-i+1}}<\varepsilon, \tag{30}$$

where C is the objective function value, k is the number of the current iteration, ε is a preset convergence error set as 0.001, and N is an integer number set as 5.

The final structures of 3D compliant mechanism using CITO and HITO are presented in Fig. 27, and the efficiency comparison is listed in Table 11, which indicates the HITO has obtained a greater max-displacement and reduced 49% computational time than the CITO. The detail of convergence process can be found in Fig. 28. Compared to CITO, HITO can rapidly converge to an appropriate max-displacement. The iteration numbers of mesh levels from coarse to fine in HITO are 40 and 27 respectively. Since the coarse mesh has fewer degrees of freedom, meaning lack of accuracy, the max displacement reaches overlarge values during the optimization. When the mesh is refined, the intitial value, inherited from the old optimal result, is not accurate enough, the max-displacement of HITO has an oscillation. But the initial value is also closed to the high-accuracy final result, which makes it converge in a short time during the optimization on fine mesh. Apart from the compliance problem, the HITO also improves the efficiency of the 3D compliant mechanism, which further demostrates the validity and efficiency of the proposed HITO.

6. . Conclusions

This paper has proposed a new high-efficiency isogeometric topology optimization method (HITO), which includes three parts: multilevel mesh for mesh-scale reduction, MGCG (multigrid conjugate gradient method) for solving acceleration and local-update strategy for design variables reduction. Four benchmark examples are used to verify the proposed method. The high-efficiency of HITO and its three accelerating parts has been demonstrated in the first example, where the HITO has reduced 37%~80% computational time compared to the conventional ITO. Furthermore, as the mesh scale increasing, the efficiency improvement is more significant, representing that the HITO is especially suitable for large-scale problems. In the second example, we discuss the effect of the parameters (ST and ρ_s) in local-update strategy, which demonstrates that the suitable parameter choice in local-update

strategy is of importance. Then, a quarter annulus example is utilized to demonstrate the advantage of IGA, i.e., direct design-to-analysis, in which no spatial discretization error exists. In this example, HITO has reduced 93% computational time compared to CITO, demonstrating its high-efficiency for examples with curved boundaries. Finally, a 3D compliant mechanism is used to demonstrate the HITO has the universality for different problems. Although the current work is based on ITO-SIMP, the HITO can be extended to other ITO methods such as ITO-level set and ITO-MMC [2,7] or solve other problems [71] and combined with parallel computing techniques for further improving the efficiency [30,37].

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.

Acknowledgments

This work has been supported by National Natural Science Foundation of China (51705158), the Fundamental Research Funds for the Central Universities (2018MS45), and Open Funds of National Engineering Research Center of Near-Net-Shape Forming for Metallic Materials (2018005). These supports are gratefully acknowledged.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.advengsoft.2019.102733.

References

[1] Hughes TJ, Cottrell JA, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 2005;194:4135–95.
[2] Xie X, Wang S, Xu M, Wang Y. A new isogeometric topology optimization using moving morphable components based on R-functions and collocation schemes. Comput Methods Appl Mech Eng 2018;339:61–90.
[3] Benson DJ, Bazilevs Y, Hsu MC, Hughes TJR. Isogeometric shell analysis: the Reissner–Mindlin shell. Comput Methods Appl Mech Eng 2010;199:276–89.
[4] Liu H, Zhu X, Yang D. Isogeometric method based in-plane and out-of-plane free vibration analysis for Timoshenko curved beams. Struct Eng Mech 2016;59:503–26.
[5] Bazilevs Y, Calo VM, Hughes TJR, Zhang Y. Isogeometric fluid-structure interaction: theory, algorithms, and computations. Comput Mech 2008;43:3–37.
[6] Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y. Patient-specific isogeometric fluid-structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik 2000 left ventricular assist device. Comput Methods Appl Mech Eng 2009;198:3534–50.
[7] Wang Y, Benson DJ. Isogeometric analysis for parameterized LSM-based structural topology optimization. Comput Mech 2016;57:19–35.
[8] Seo Y-D, Kim H-J, Youn S-K. Isogeometric topology optimization using trimmed spline surfaces. Comput Methods Appl Mech Eng 2010;199:3270–96.
[9] De Luycker E, Benson DJ, Belytschko T, Bazilevs Y, Hsu MC. X-FEM in isogeometric analysis for linear fracture mechanics. Int J Numer Meth Eng 2011;87:541–65.
[10] Wang Y, Xu H, Pasini D. Multiscale isogeometric topology optimization for lattice materials. Comput Methods Appl Mech Eng 2017;316:568–85.
[11] Buffa A, Sangalli G, Vázquez R. Isogeometric analysis in electromagnetics: b-splines approximation. Comput Methods Appl Mech Eng 2010;199:1143–52.
[12] Cottrell JA, Hughes TJ, Bazilevs Y. Isogeometric analysis: toward integration of CAD and FEA. John Wiley & Sons; 2009.
[13] Bendsøe MP, Kikuchi N. Generating optimal topologies in structural design using a homogenization method. Comput Methods Appl Mech Eng 1988;71:197–224.
[14] Sigmund O. A 99 line topology optimization code written in Matlab. Struct Multidiscip Optim 2001;21:120–7.
[15] Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O. Efficient topology optimization in Matlab using 88 lines of code. Struct Multidiscip Optim 2011;43:1–16.
[16] Bendsøe MP, Sigmund O. Material interpolation schemes in topology optimization. Arch Appl Mech 1999;69:635–54.
[17] Zhou M, Rozvany G. The coc algorithm, part II: topological, geometrical and generalized shape optimization. Comput Methods Appl Mech Eng 1991;89:309–36.
[18] Xie YM, Steven GP. A simple evolutionary procedure for structural optimization. Comput Struct 1993;49:885–96.
[19] Huang X, Xie M. Evolutionary topology optimization of continuum structures: methods and applications. John Wiley & Sons; 2010.
[20] Allaire G, Jouve F, Toader AM. Structural optimization using sensitivity analysis and a level-set method. J Comput Phys 2004;194:363–93.
[21] Sethian JA, Wiegmann A. Structural boundary design via level set and immersed interface methods. J Comput Phys 2000;163:489–528.
[22] Wang MY, Wang X. Color" level sets: a multi-phase method for structural topology optimization with multiple materials. Comput Methods Appl Mech Eng 2004;193:469–96.
[23] Guo X, Zhang WS, Zhong WL. Doing topology optimization explicitly and geometrically-a new moving morphable components based framework. J Appl Mech ASME 2014;81(8):081009.
[24] Zhang W, Song J, Zhou J, Du Z, Zhu Y, Sun Z, et al. Topology optimization with multiple materials via moving morphable component (MMC) method. Int J Numer Meth Eng 2018;113:1653–75.
[25] Dedè L, Borden MJ, Hughes TJ. Isogeometric analysis for topology optimization with a phase field model. Arch Comput Method E 2012;19:427–65.
[26] Kang P, Youn S-K. Isogeometric topology optimization of shell structures using trimmed Nurbs surfaces. Finite Elem Anal Des 2016;120:18–40.
[27] Roodsarabi M, Khatibinia M, Sarafrazi S. Isogeometric topology optimization of structures using level set method incorporating sensitivity analysis. Iran Univ Sci Technol 2016;6:405–22.
[28] Hou W, Gai Y, Zhu X, Wang X, Zhao C, Xu L, et al. Explicit isogeometric topology optimization using moving morphable components. Comput Methods Appl Mech Eng 2017;326:694–712.
[29] Wang Y, Benson DJ. Geometrically constrained isogeometric parameterized level-set based topology optimization via trimmed elements. Front Mech Eng 2016;11:328–43.
[30] Xia Z, Wang Y, Wang Q, Mei C. GPU parallel strategy for parameterized LSM-based topology optimization using isogeometric analysis. Struct Multidiscip Optim 2017;56:413–34.
[31] Wang Y, Xu H, Pasini D. Multiscale isogeometric topology optimization for lattice materials. Comput Methods Appl Mech Eng 2016;316:568–85.
[32] Kazemi H, Tavakkoli S, Naderi R. Isogeometric topology optimization of structures considering weight minimization and local stress constraints. Iran Univ Sci Technol 2016;6:303–17.
[33] Sahithi N, Chandrasekhar K, Rao TM. A comparative study on evolutionary algorithms to perform isogeometric topology optimisation of continuum structures using parallel computing. J Aerosp Eng Technol 2018;8:51–8.
[34] Liu H, Yang D, Hao P, Zhu X. Isogeometric analysis based topology optimization design with global stress constraint. Comput Methods Appl Mech Eng 2018;342:625–52.
[35] Wang Y, Wang Z-P, Xia Z, Poh LH. Structural design optimization using isogeometric analysis: a comprehensive review. Comput Model Eng Sci 2018:455–507.
[36] Aage N, Andreassen E, Lazarov BS, Sigmund O. Giga-voxel computational morphogenesis for structural design. Nature 2017;550:84–6.
[37] Martínez-Frutos J, Herrero-Pérez D. GPU acceleration for evolutionary topology optimization of continuum structures using isosurfaces. Comput Struct 2017;182:119–36.
[38] Zegard T, Paulino GH. Toward GPU accelerated topology optimization on unstructured meshes. Struct Multidiscip Optim 2013;48:473–85.
[39] Aage N, Lazarov BS. Parallel framework for topology optimization using the method of moving asymptotes. Struct Multidiscip Optim 2013;47:493–505.
[40] Wang Y, Wang Q, Deng X, Xia Z, Yan J, Xu H. Graphics processing unit (GPU) accelerated fast multipole BEM with level-skip M2L for 3D elasticity problems. Adv Eng Softw 2015;82:105–18.
[41] Kim SY, Kim IY, Mechefske CK. A new efficient convergence criterion for reducing computational expense in topology optimization: reducible design variable method. Int J Numer Methods Eng 2012;90:752–83.
[42] Amir O, Sigmund O. On reducing computational effort in topology optimization: how far can we go? Struct Multidiscip Optim 2011;44:25–9.
[43] Liao Z, Zhang Y, Wang Y, Li W. A triple acceleration method for topology optimization. Struct Multidiscip Optim 2019;60:727–44.
[44] Lin CY, Chou JN. A two-stage approach for structural topology optimization. Adv Eng Softw 1999;30:261–71.
[45] Stainko R. An adaptive multilevel approach to the minimal compliance problem in topology optimization. Commun Numer Methods Eng 2006;22:109–18.
[46] Nguyen TH, Paulino GH, Song J, Le CH. A computational paradigm for multiresolution topology optimization (MTOP). Struct Multidiscip Optim 2010;41:525–39.
[47] Lieu QX, Lee J. A multi-resolution approach for multi-material topology optimization based on isogeometric analysis. Comput Methods Appl Mech Eng 2017;323:272–302.
[48] Davis TA. Direct methods for sparse linear systmes. Vol. 2. Siam; 2006.
[49] Saad Y. Iterative methods for sparse linear systems. Vol. 82. Siam; 2003.
[50] Benzi M. Preconditioning techniques for large linear systems: a survey. J Comput Phys 2002;182:418–77.
[51] Papadrakakis M, Tsompanakis Y, Hinton E, Sienz J. Advanced solution methods in topology optimization and shape sensitivity analysis. Eng Comput 1996;13:57–90.
[52] Amir O, Stolpe M, Sigmund O. Efficient use of iterative solvers in nested topology optimization. Struct Multidiscip Optim 2009;42:55–72.
[53] Liu K, Tovar A. An efficient 3D topology optimization code written in Matlab. Struct Multidiscip Optim 2014;50:1175–96.
[54] Amir O, Aage N, Lazarov BS. On multigrid-CG for efficient topology optimization. Struct Multidiscip Optim 2014;49:815–29.
[55] Guest JK, Genut LCS. Reducing dimensionality in topology optimization using adaptive design variable fields. Int J Numer Methods Eng 2010;81:1019–45.

[56] Yoo J, Lee I. Efficient density based topology optimization using dual-layer element and variable grouping method for large 3D applications. In: Schumacher A, Vietor T, Fiebig S, Bletzinger K-U, Maute K, editors. Advances in structural and multidisciplinary optimization. editorsCham: Springer International Publishing; 2018. p. 967–78.
[57] Qian XP. Topology optimization in B-spline space. Comput Methods Appl Mech Eng 2013;265:15–35.
[58] Piegl L, Tiller W. The nurbs book. Berlin Heidelberg: Springer; 1997.
[59] Boor CD. On calculating with B -splines. J Approx Theory 1972;6:50–62.
[60] Bruns TE, Tortorelli DA. Topology optimization of non-linear elastic structures and compliant mechanisms. Comput Methods Appl Mech Eng 2001;190:3443–59.
[61] Boggs PT, Tolle JW. Sequential quadratic programming. AcNum 1995;4:1–51.
[62] Svanberg K. The method of moving asymptotes—a new method for structural optimization. Int J Numer Meth Eng 1987;24:359–73.
[63] Tatebe O, Oyanagi Y. Efficient implementation of the multigrid preconditioned conjugate gradient method on distributed memory machines. Proceedings of the ACM/IEEE conference on Supercomputing. IEEE Computer Society Press; 1994. p. 194–203.
[64] McAdams A, Sifakis E, Teran J. A parallel multigrid poisson solver for fluids simulation on large grids. Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation: eurographics association. 2010. p. 65–74.
[65] Notay Y, Vassilevski PS. Recursive Krylov-based multigrid cycles. Numer Linear Algebr 2008;15:473–87.
[66] Kershaw DS. The incomplete Cholesky—conjugate gradient method for the iterative solution of systems of linear equations. J Comput Phys 1978;26:43–65.
[67] Gao J, Liang R, Wang J. Research on the conjugate gradient algorithm with a modified incomplete Cholesky preconditioner on GPU. J Parallel Distrib Comput 2014;74:2088–98.
[68] Gonzaga de Oliveira SL, Bernardes JAB, Chagas GO. An evaluation of reordering algorithms to reduce the computational cost of the incomplete Cholesky-conjugate gradient method. Comput Appl Math 2018;37:2965–3004.
[69] Huang X, Xie YM. Convergent and mesh-independent solutions for the bi-directional evolutionary structural optimization method. Finite Elem Anal Des 2007;43:1039–49.
[70] Wang Y, Arabnejad S, Tanzer M, Pasini D. Hip implant design with three-dimensional porous architecture of optimized graded density. ASME J Mech Des 2018;140:111406.
[71] Tran AV, Zhang X, Zhu B. The development of a new piezoresistive pressure sensor for low pressures. ITIE 2018;65:6487–96.