



# A triple acceleration method for topology optimization

Zhongyuan Liao<sup>1</sup> · Yu Zhang<sup>1</sup> · Yingjun Wang<sup>1</sup> · Weihua Li<sup>2</sup>

Received: 30 October 2018 / Revised: 3 January 2019 / Accepted: 13 February 2019 / Published online: 8 March 2019  
© Springer-Verlag GmbH Germany, part of Springer Nature 2019

## Abstract

This paper presents a triple acceleration method (TAM) for the topology optimization (TO), which consists of three parts: multilevel mesh, initial-value-based preconditioned conjugate-gradient (PCG) method, and local-update strategy. The TAM accelerates TO in three aspects including reducing mesh scale, accelerating solving equations, and decreasing the number of updated elements. Three benchmark examples are presented to evaluate proposed method, and the result shows that the proposed TAM successfully reduces 35–80% computational time with faster convergence compared to the conventional TO while the consistent optimization results are obtained. Furthermore, the TAM is able to achieve a higher speedup for large-scale problems, especially for the 3D TOs, which demonstrates that the TAM is an effective method for accelerating large-scale TO problems.

**Keywords** Topology optimization · Triple acceleration · Multilevel mesh · Preconditioned conjugate-gradient method · Local update

## 1 Introduction

Topology optimization (TO) is a mathematical method aiming at finding the optimal material distribution subjected to some constraints within a given domain. From the pioneering work of Bendsoe and Kikuchi (1988), TO has become one of the most important optimization methods in structural optimization, and a series of impressive TO methods have emerged in the past 30 years, such as homogenization method (Bendsoe and Kikuchi 1988), solid isotropic material with penalization (SIMP) approach (Bendsoe 1989; Sigmund 2001), evolutionary approach (Xie and Steven 1993), level set method (Allaire

et al. 2004; Deng et al. 2016; Mei and Wang 2004), and moving morphable component (MMC) method (Guo et al. 2014; Zhang et al. 2018). TO has been successfully applied in various academic fields, such as solid mechanics (Bendsee 1983; Lindgaard and Dahl 2013; Madah and Amir 2017), fluid mechanics (Gersborg-Hansen et al. 2005; Norgaard et al. 2016), heat transfer (Dirker and Meyer 2013; Gersborg-Hansen et al. 2006), acoustics (Christiansen et al. 2015; Desai et al. 2018) and electromagnetics (Zhou et al. 2010), and engineering fields, such as aircraft (Zhu et al. 2016), additive manufacturing (Langelaar 2016; Langelaar 2017), composite materials (Cheng and Chen 2015), and design in photonics (Li et al. 2016a; Li et al. 2016b).

In actual engineering problems, the computational scales of TOs are usually large (Aage et al. 2017; Alexandersen et al. 2016; Evgrafov et al. 2008), and will be larger and larger in the future since the problems are becoming more complex with more details. Therefore, pursuing higher computational efficiency is the eternal objective of TOs. In general, there are two types of methods to improve the computational efficiency: one is the algorithm improvement (Amir and Sigmund 2011; Kim et al. 2012), and the other is parallel computing (Aage and Lazarov 2013; Martínez-Frutos and Herrero-Pérez 2017; Wang et al. 2015b; Xia et al. 2017; Zegard and Paulino 2013). Apparently, the former one is capable to fundamentally accelerate the TO and is worthier to study. Several aspects can be studied to improve the efficiency of the TO, and three representative aspects are convergence acceleration, solution acceleration, and design variable reduction.

---

Responsible Editor: Emilio Carlos Nelli Silva

✉ Yingjun Wang  
wangyj84@scut.edu.cn

<sup>1</sup> National Engineering Research Center of Novel Equipment for Polymer Processing, Key Laboratory of Polymer Processing Engineering (South China University of Technology), Ministry of Education, Guangdong Provincial Key Laboratory of Technique and Equipment for Macromolecular Advanced Manufacturing, School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510641, China

<sup>2</sup> National Engineering Research Center of Near-Net-Shape Forming for Metallic Materials, School of Mechanical and Automotive Engineering, South China University of Technology, Guangzhou 510641, China

Fig. 1 density field inheritance

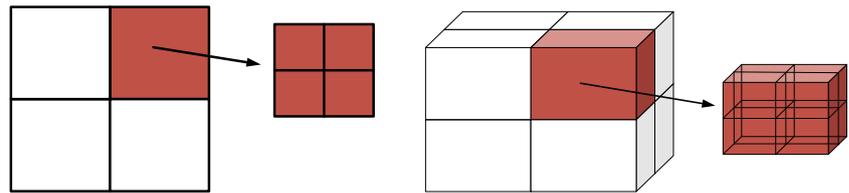
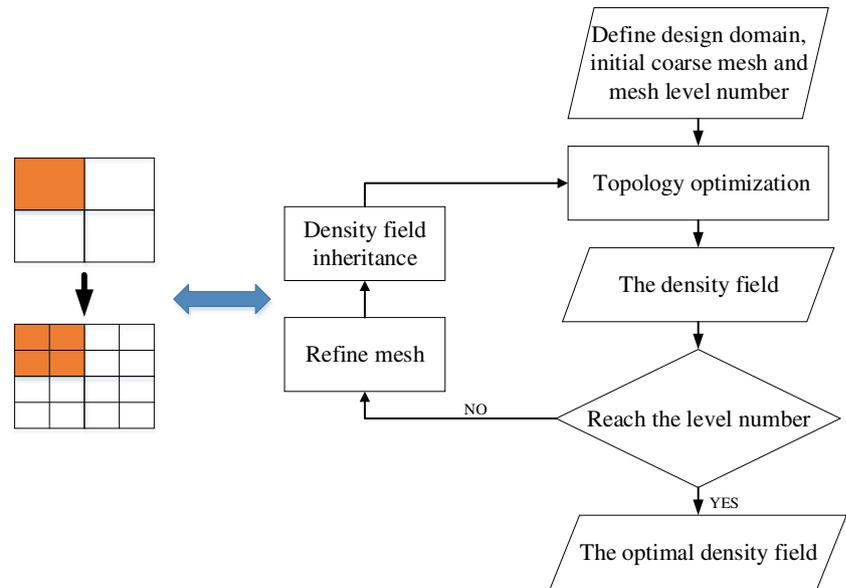


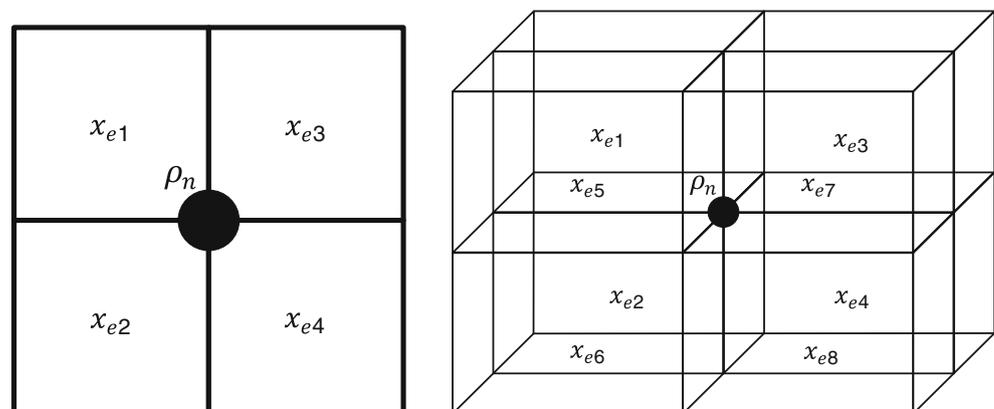
Fig. 2 The flowchart of multilevel mesh algorithm



In order to accelerate the convergence, Lin and Chou (1999) presented a two-stage approach for TO, the optimal result of the first stage constructed by big-sized elements is used as an initial topology of the second stage whose element size is smaller, but only two different element sizes are considered in the two-stage approach. Kim and Yoon (2000) proposed multiresolution multiscale topology optimization (MTOP) where the design optimization was performed progressively from low to high resolution. Nguyen et al. (2010) used multiresolution topology optimization (MTOP) scheme

to obtain high-resolution designs with relatively low computational cost (i.e., a coarse mesh used for finite element analysis (FEA)), which implemented in the TopOpt app (Aage et al. 2013). Recently, Lieu and Lee (2017a, 2017b) proposed a MTOP using isogeometric analysis (IGA) and extended it to multimaterial TO. Although above MTOP approaches can greatly reduce the degrees of freedom (DOFs) in the FEA or IGA by using a coarse mesh with fewer elements, but the accuracy of FEA or IGA is also sacrificed due to the coarse mesh. For the strategy of refining mesh, Stainko (2006)

Fig. 3 The nodal density mapping relation



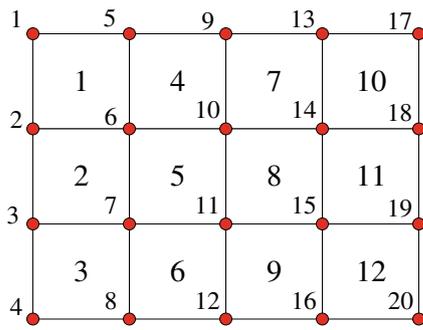


Fig. 4 The 2D design domain with 12 elements

presented a multilevel scheme adaptively refined only in the interface between solid and void. More researches of refining mesh can be found in (Lambe and Czekanski 2018; Panesar et al. 2017; Wang et al. 2013). Above refining approaches refined mesh on the interface between solid and void, which increase the complexity of TO. When the volume fraction is small, there are still many small-density and void elements to be updated, which occupy major computational resource but have fewer effects on the accuracy of computation and design.

In TOs, it is required to solve finite element (FE) equations in each iteration, and a fast solution method for the FE equations can definitely accelerate the TOs. There are two types of equation solution methods, direct methods (Davis 2006) and iterative methods (Saad 2003). For large-scale FE-based TO problems, iterative methods have better performance than direct methods in computational time and memory requirements (Benzi 2002). The key to speed up the iterative method is how to choose a suitable iterative method and its preconditioner. Augarde et al. (2006) chose a preconditioner with element displacement components to improve convergence. Wang et al. (2007) proposed a preconditioned Krylov subspace methods with recycling, which gave a fast iteration solver for large three-dimensional TO. Many iterative methods have been introduced into TO problems, one of which is PCG (preconditioned conjugate-gradient method) (Papadrakakis et al. 1996). Amir et al. (2009) used a standard PCG solver to terminate the iterative solution of the nested equations in a short time. Furthermore,

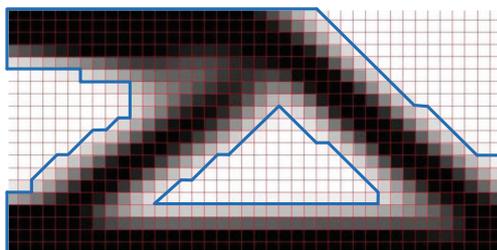


Fig. 5 The updated range in cantilever beam (40 × 20)

Table 1 Algorithm 1

Algorithm 1 Threshold value of nodal density

Choose a threshold value of nodal density  $\rho_c$   
 Compare each nodal density with  $\rho_c$   
 If  $\rho_n > \rho_c$  put the encircled elements of the nodes (according to the mapping relation between nodes and elements) into the updated range.

Amir et al. (2014) combined PCG and multigrid into multigrid preconditioned conjugate gradients (MGCG), aiming to accelerate the solution speed of the linear system in TO. Liu and Tovar (2014) used the PCG in MATLAB to write efficient 3D TO code. More PCG applications can be found in (Amir 2015; Amir and Sigmund 2011). Most improved methods for PCG are accelerated by preconditioner, other factors for acceleration were less considered.

For design variable reduction, Guest and Genut (2010) found that reducing some design variables was expected to fewer iterations than using full design variable fields. Kim et al. (2012) proposed a reducible design variable method (RDVM), which reduced some quickly converge design variables from iterations to save computational expense. Yoo and Lee (2018) proposed a variable grouping method which can reduce variables according to histogram of multiplication of sensitivity and density variable. For design variable reduction, there are few researches meanwhile lack of concise and effective methods.

Although many researches about acceleration algorithm exist, there are still lots of work can be done to further improve current researches. Furthermore, acceleration methods are mostly focused on one aspect, accelerating effect in multiple aspects is seldom seen. In this paper, we will propose a triple acceleration method for TO based on three parts: multilevel mesh, initial-value-based PCG, and local-update strategy, which accelerates TO in three aspects: convergence acceleration, solution acceleration, and design variable reduction. In the reminder of this paper, Sect. 2 introduces the basic theories of TO and the mathematical model of PCG. Section 3 presents a detailed description about the proposed acceleration method. Section 4 presents the process of algorithm implementation.

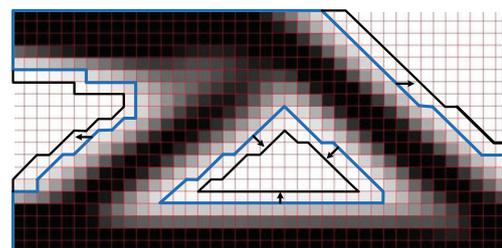


Fig. 6 The boundary moves 1 layer toward outside area (diffused layer number is 1)

**Table 2** Algorithm 2

Algorithm 2 Diffused layer

```

Choose a diffused layer number  $D_{num}$ 
For 1:  $D_{num}$ 
    Assign all nodal density inside updated range with 1. Compare each nodal density with 1.
    If  $\rho_n = 1$ , put the encircled elements of the node (according to the mapping relation between nodes and elements) into the new updated range
end
    
```

Section 5 discusses the acceleration effect of the proposed method by benchmark examples for both 2D and 3D problems, and the conclusions are drawn in Sect. 6.

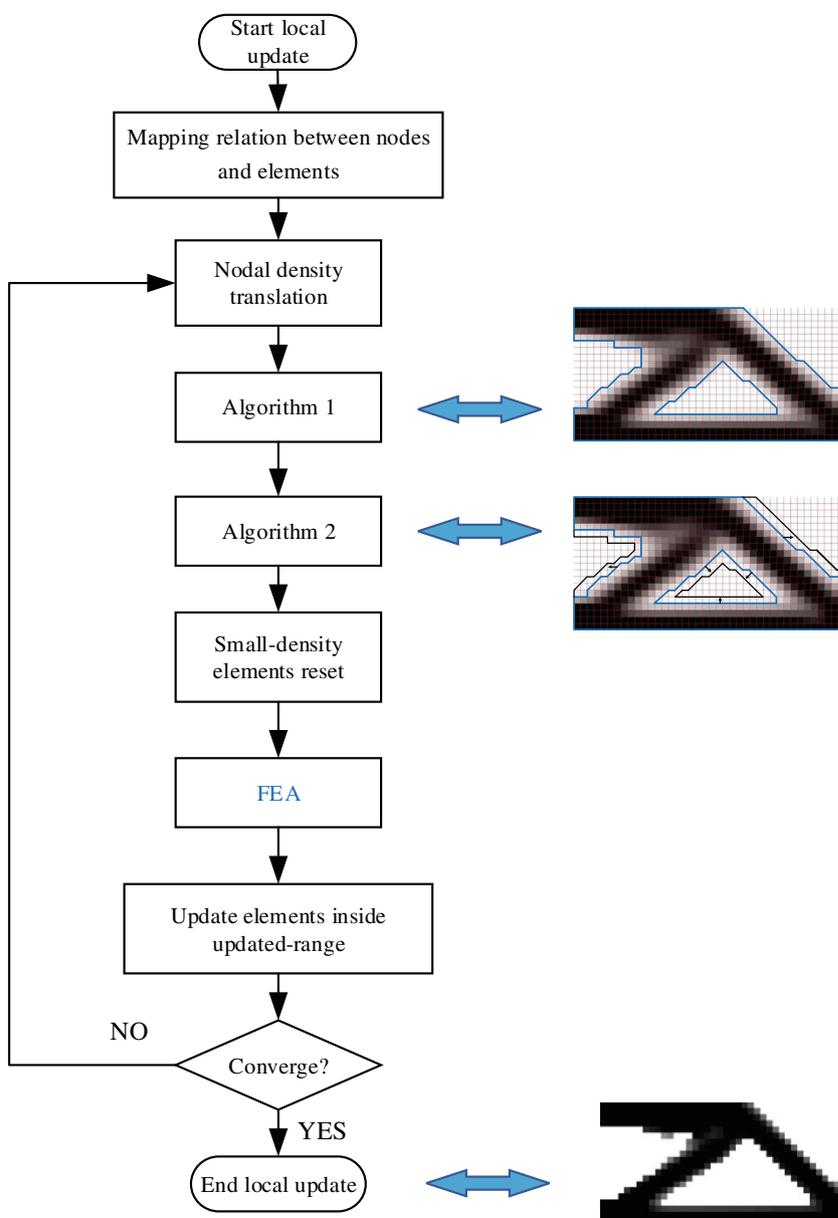
## 2 Theoretical basis

The theoretical bases of this paper include SIMP-based TO and PCG, which will be summarized in the following sections. For more detail discussion on SIMP-based TO and PCG, we refer readers to (Amir et al. 2009; Andreassen et al. 2011; Barrett 1995; Liu and Tovar 2014; Sigmund 2001).

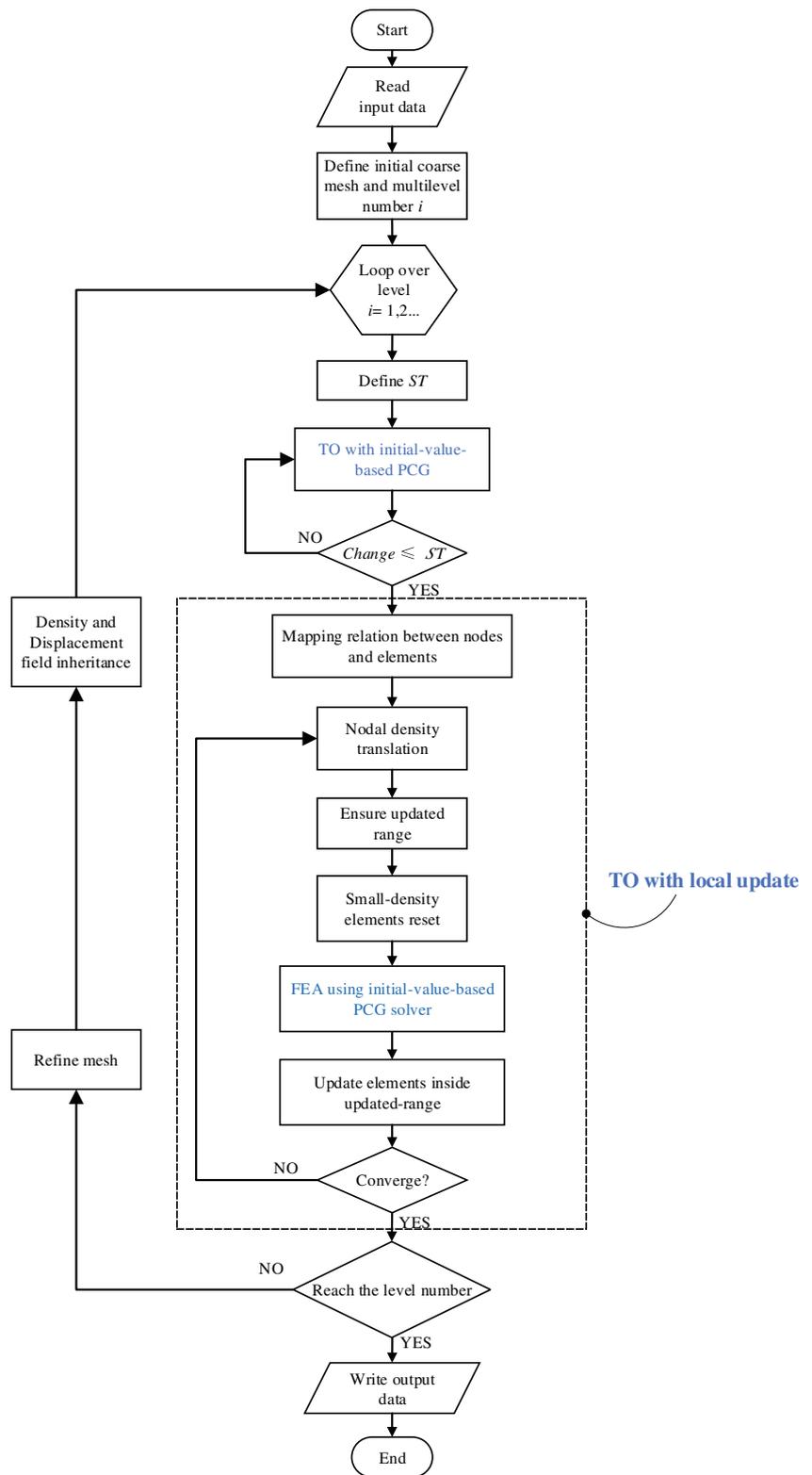
### 2.1 SIMP-based TO

SIMP is a simple and efficient TO method. Since Sigmund (2001) proposed the 99-line MATLAB codes for TO; the SIMP became more and more widespread in the world (Andreassen et al. 2011). For the SIMP, the design domain

**Fig. 7** The flowchart of the local-update strategy



**Fig. 8** The flowchart of the TO using TAM



is divided into finite elements, and the element density  $x_e$  determines its Young's modulus  $E_e$ :

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

where  $E_0$  is the Young's modulus of a fully solid element;  $E_{min}$  is a very small Young's modulus assigned to void elements aiming to prevent the stiffness matrix from becoming singular, and  $p$  is a penalization factor (typically  $p = 3$ ).

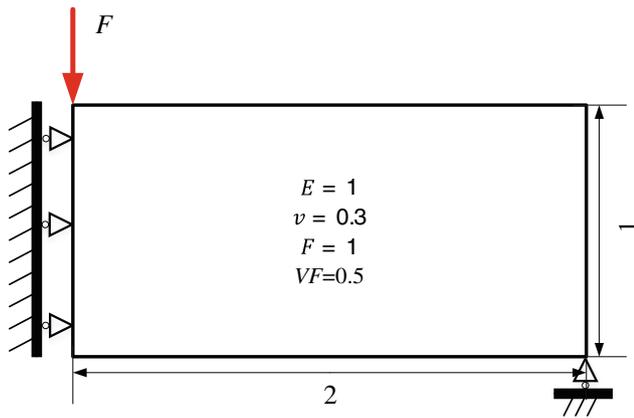


Fig. 9 The design domain and boundary conditions of half MBB beam

The mathematical formulation of the compliance optimization problem is shown as follows:

$$\begin{aligned}
 &\text{Find } \mathbf{x} = [x_1, x_2, \dots, x_e, \dots, x_N]^T, \\
 &\text{Min } c(\mathbf{x}) = \mathbf{U}^T \mathbf{K} \mathbf{U} = \sum_{e=1}^N E_e(x_e) \mathbf{u}_e^T \mathbf{k}_e \mathbf{u}_e, \\
 &\text{st } \begin{cases} \frac{V(\mathbf{x})}{V_0} = VF \\ \mathbf{K} \mathbf{U} = \mathbf{F} \\ 0 \leq x_e \leq 1 \end{cases}, \quad (2)
 \end{aligned}$$

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

To solve the optimization problem, a heuristic updating scheme called optimality criteria (OC) (Wang et al. 2017) is as follows:

$$x_e^{\text{new}} = \begin{cases} \max(x_{\min}, x_e - m), & \text{if } x_e B_e^\eta \leq \max(x_{\min}, x_e - m) \\ x_e B_e^\eta, & \text{if } \max(x_{\min}, x_e - m) \leq x_e B_e^\eta \leq \min(1, x_e + m) \\ \min(1, x_e + m), & \text{if } \min(1, x_e + m) \leq x_e B_e^\eta \end{cases}, \quad (3)$$

where  $m$  is the positive move limit;  $\eta$  is a numerical damping coefficient, and  $B_e$  can be obtained from the optimality condition as:

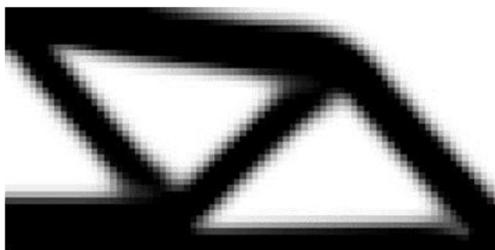


Fig. 10 The optimization result of conventional TO

$$B_e = \frac{-\frac{\partial c}{\partial x_e}}{\lambda \frac{\partial V}{\partial x_e}} \quad (4)$$

where  $\lambda$  is the Lagrangian multiplier which can be obtained by bi-sectioning algorithm.

The sensitivities of the objective function  $c$  and the material volume  $V$  with respect to the element density  $x_e$  are found as:

$$\frac{\partial c}{\partial x_e} = -p(x_e)^{p-1} \mathbf{u}_e^T \mathbf{k}_e \mathbf{u}_e \quad (5)$$

$$\frac{\partial V}{\partial x_e} = 1 \quad (6)$$

where Eq. (6) is established under the condition that each element has unit volume.

To avoid the drawbacks of TO including checkboard, local minima, and mesh dependency, we use the density filter in this work, and more details can be found in the work of Bruns and Tortorelli (Bruns and Tortorelli 2001).

### 2.2 Preconditioned conjugate-gradient method

PCG, improved from conjugate-gradient (CG) method, is an iterative method for solving a system of linear algebraic equations, in which a preconditioner for PCG can be used to accelerate the convergence. Applying the PCG to solve the following equations

$$\mathbf{A} \mathbf{x} = \mathbf{b} \quad (7)$$

in which the residuals of each iteration are:

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A} \mathbf{x}_i \quad (8)$$

where the coefficient matrix  $\mathbf{A}$  must be symmetric, positive-definite, and sparse.

Improved from the CG, PCG adds a preconditioning matrix  $\mathbf{M}$  to accelerate the solution as follows:

$$\mathbf{M}^{-1} \mathbf{A} \mathbf{x} = \mathbf{M}^{-1} \mathbf{b} \quad (9)$$

and the recursion formula of residuals is:

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha_{i-1} \mathbf{A} \mathbf{p}_{i-1} \quad (10)$$

where the search direction vector  $\mathbf{p}_i$  and the multiplier  $\alpha_i$  are expressed as:

$$\mathbf{p}_i = \mathbf{z}_i + \beta_i \mathbf{p}_{i-1} \quad (11)$$

$$\alpha_i = \frac{\mathbf{r}_i^T \mathbf{z}_i}{\mathbf{p}_i^T \mathbf{A} \mathbf{p}_i} \quad (12)$$

and

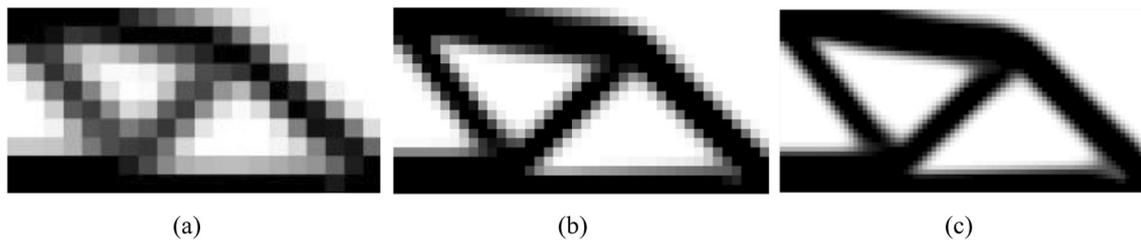


Fig. 11 The optimization results of each level. a  $20 \times 10$ . b  $40 \times 20$ . c  $80 \times 40$

$$z_i = M^{-1}r_i \tag{13}$$

$$\beta_i = \frac{r_i^T z_i}{r_{i-1}^T z_{i-1}}. \tag{14}$$

The vector  $x_i$  is updated in each iteration by a multiplier  $\alpha_i$  of the search direction vector  $p_i$ :

$$x_{i+1} = x_i + \alpha_i p_i. \tag{15}$$

When the residual vector  $r_i$  reaches a predefined small level, the iteration stops. Compared with the traditional method of solving linear algebraic equations with a direct method, PCG is more suitable for large-scale problems and is widespread used as a iterative solver in FE solution (Amir et al. 2009).

### 3 Triple acceleration algorithm model

To accelerate the TO, three acceleration schemes including multilevel mesh, initial-value-based PCG and local update will be proposed in this section from different aspects.

#### 3.1 Multilevel mesh

To reduce the computational cost, a multilevel mesh acceleration algorithm is proposed, which first solves the TO in a coarser mesh and then maps the optimized result to a finer mesh as the initial design. During the SIMP TO, each element density is a design variable, and the number of elements has an important influence on the solution process, i.e., more elements costs more solving time. On the other hand, if the mesh is too coarse, it is hard to get the precise boundary and the

Table 3 Efficiency comparison between Top88 and TO with multilevel mesh

	Top88	Multilevel mesh
Compliance	84.84	85.25
Iteration number	61	48 + 25 + 18
Computational time (s)	3.22	2.67

optimal density distribution field. Multilevel mesh algorithm provides a better choice to balance coarse and fine meshes.

The multilevel mesh method divides the TO process into several levels, and the mesh size varies from coarse to fine. First, the TO process is implemented on the coarse mesh, which has the fewer elements. The optimal result of the coarse mesh can be quickly obtained by a fast convergence. After then, the mesh is refined, and the density field of the coarse mesh is mapped to the refined mesh as the initial density field. Through such density mapping, the TO converges much faster so that the number of iterations on the refined mesh can be reduced, which can successfully save the computational time since the refined mesh requires much more computational cost in each iteration. The above procedure can be repeated several times for multiple levels in terms of the practical requirement.

The core of the multilevel mesh method is the density field inheritance between adjacent levels. In this work, we use a bisection method to refine the mesh. When the mesh refinement is applied, one element will be refined into four (2D) or eight (3D) subelements, and Fig. 1 shows the density field inheritance in 2D and 3D mesh.

The flowchart of multilevel mesh algorithm is shown in Fig. 2.

Although two levels and coincident regular grids are taking as examples in Figs. 1 and 2, it should be noted that the proposed multilevel approach can be generalized for any mesh level and non-coincident regular grids (an interpolation method is required to transmit the density fields between the non-coincident grids).

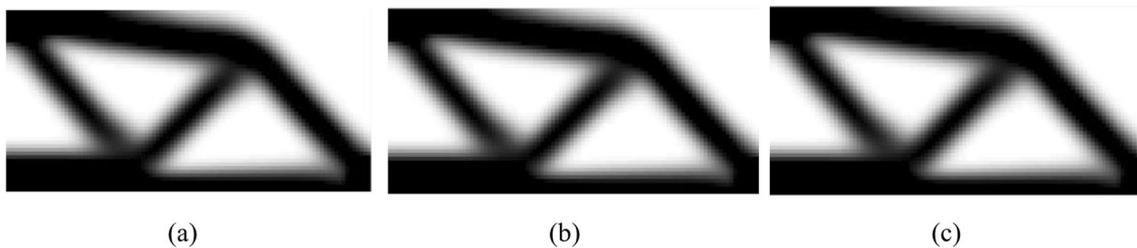
#### 3.2 Initial-value-based PCG

To solve the FE problem in TO, the PCG (preconditioned conjugate-gradient method) is applied in this work to calculate the displacement vector  $U$  in the FE equations  $KU = F$ , and a preconditioner  $M$  is used to accelerate the solution of the equations.

The convergence rate of PCG is given by (Amir et al. 2014)

$$\|U - U_i\|_{K^{-1}} \leq \|U - U_0\|_{K^{-1}} \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^i \tag{16}$$

where  $i$  is the iteration number;  $\kappa$  is the condition number of matrix  $K$ . When  $\kappa \gg 1$ , the convergence will be very slow.



**Fig. 12** The optimization results of TOs using PCG under different conditions. **a** No preconditioner and  $U_{pre}$ . **b** Only preconditioner. **c** Both preconditioner and  $U_{pre}$

Therefore, a preconditioning matrix  $M$  is applied in the iterations, which makes  $\kappa(M^{-1}K) \ll \kappa(K)$ . In PCG method, the effective preconditioner  $M$  should make the  $\kappa(M^{-1}K)$  close to 1, which can accelerate the convergence considerably.

In order to obtain a good preconditioner  $M$ , many methods have been proposed, such as incomplete Cholesky factorization, diagonal scaling, or factorized sparse approximate inverses (FSAI). More details can be found in (Barrett 1995). In this paper, we choose incomplete Cholesky factorization as the preconditioner,  $M^{-1}$  can be factorized as follows:

$$M^{-1} = LL^T \tag{17}$$

where  $L$  is a lower triangular matrix. The FE equations  $KU = F$  is rewritten as

$$\hat{K}\hat{U} = \hat{F} \tag{18}$$

where

$$\begin{aligned} \hat{K} &= L^T K L, \\ \hat{U} &= L^{-1} U, \\ \hat{F} &= L^T F \end{aligned} \tag{19}$$

By using incomplete Cholesky factorization, it is successful to make the  $\kappa(\hat{K}) \ll \kappa(K)$ . The actual acceleration effect can be found in Sect. 5.1.2.

An initial value may accelerate the solution of PCG. In the TO, the optimized results are similar between two adjacent iterations, especially for the later iterations of the solution.

**Table 4** Efficiency comparison of PCG with different conditions

	No preconditioner and $U_{pre}$	Only preconditioner	Both preconditioner and $U_{pre}$
Compliance	93.80	93.80	93.95
Iteration number of TO	43	43	45
Average iteration number of PCG	8344	7	4
Computational time (s)	82.22	5.30	5.09

Therefore, we define the previous iteration result, named  $U_{pre}$ , as an initial value. In current iteration,  $U_{pre}$  is close to the unknown vector  $U$ , which helps the PCG easier to find the solution and save solution time.

To integrate with multilevel mesh, displacement ( $U$ ) inheritance should be also applied between the adjacent two levels. The  $U$  can be calculated by such interpolation strategy:

$$U_i^n = \sum_{j=1}^k N_j U_j^{n-1} \tag{20}$$

where  $U_i^n$  represents the new displacement field of an element in the refinement mesh;  $n$  is the number of mesh levels;  $i$  represents the nodal ID of the refined mesh;  $N_j$  is the value of shape function of the coarser mesh at the  $i$ th node on the refined mesh;  $j$  represents the nodal ID of the coarser mesh;  $U_j^{n-1}$  is the displacement field of the coarser mesh. With the interpolation method, the PCG is successfully applied between the adjacent two levels, further accelerating the TO.

### 3.3 Local-update strategy

The last and the most important acceleration proposed in this work is the local-update strategy, which reduces the computational scale in the TO iterations. During the conventional TO, all design variables must be updated in each iteration. According to Guest and Genut (2010), reducing some design variables will not significantly influence the performance of solutions, but save significant computational cost. Reducing some design variables is expected to achieve fewer iterations than using all design variables. Enlightened by this research, a local-update strategy is proposed and realizes a high-efficiency TO method.

The most important part of the local-update strategy is how to identify the updated range. To guarantee the stabilization and accuracy of TO, the updated range should cover the solid region and the boundary. Therefore, the key of local-update strategy is finding the boundary and the solid region.

It is a serious drawback that the traditional element-based TO is difficult to obtain exact boundary. To identify the boundary precisely, the nodal density is used to replace the element density. For a node, shared by four elements (2D) or eight elements (3D) as Fig. 3, its density  $\rho_n$  is calculated by

**Fig. 13** The optimization results. **a** Top88. **b** Local-update strategy



$$\rho_n = \frac{\sum_{j=1}^N x_{ej}}{N} \tag{21}$$

where  $N$  is the number of encircled elements (four for 2D, eight for 3D, particularly at the corner or boundary, the number of encircled elements is only 1,2 for 2D or 1,2,4 for 3D), and  $j$  represents the  $j$ th encircled element. It is noted that the nodal density is also a relative density, i.e.,  $\rho_n \in [0, 1]$ .

In order to map element density to nodal density, we need to define the mapping relation. Take a coarse mesh with 12 elements as an example, where the design domain is rectangular and discretized with square elements. Both elements and nodes are numbered column-wise from left to right, shown in Fig. 4.

There are exact IDs for each node and element, so the mapping relation between nodes and elements of the 12 elements is formulated as

$$\begin{aligned} \text{ele} \rightarrow \text{node} &= \begin{bmatrix} 1 & 2 & 5 & 6 \\ 2 & 3 & 6 & 7 \\ 3 & 4 & 7 & 8 \\ 5 & 6 & 9 & 10 \\ \vdots & \vdots & \vdots & \vdots \\ 15 & 16 & 19 & 20 \end{bmatrix} \begin{matrix} \leftarrow \text{element 1} \\ \leftarrow \text{element 2} \\ \leftarrow \text{element 3} \\ \leftarrow \text{element 4} \\ \vdots \\ \leftarrow \text{element 12} \end{matrix} \\ \\ \text{node} \rightarrow \text{ele} &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 3 & 0 & 0 \\ 3 & 0 & 0 & 0 \\ 1 & 4 & 0 & 0 \\ 1 & 2 & 4 & 5 \\ 2 & 3 & 5 & 6 \\ \vdots & \vdots & \vdots & \vdots \\ 12 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} \leftarrow \text{node 1} \\ \leftarrow \text{node 2} \\ \leftarrow \text{node 3} \\ \leftarrow \text{node 4} \\ \leftarrow \text{node 5} \\ \leftarrow \text{node 6} \\ \leftarrow \text{node 7} \\ \vdots \\ \leftarrow \text{node 20} \end{matrix} \end{aligned}$$

**Table 5** Efficiency comparison between Top88 and TO with local-update strategy

	Top88	Local-update
Compliance	84.85	79.03
Iteration number	61	32
Computational time (s)	3.22	1.91

After obtaining nodal densities, it is easy to define a criterion to divide the design domain into two subdomains by a threshold density value: (1) the subdomain with nodal density larger than or equal to the threshold density and (2) the subdomain with nodal density smaller than the threshold density. Only the elements inside the former subdomain are updated. In terms of the position coordinates of nodes, we can extract the boundaries that separate the subdomains. Figure 5 shows one of the iterative results during TO for cantilever beam with  $40 \times 20$  mesh scale, in which the nodes with the approximate threshold density form a closed area (inside the blue boundary), only the elements inside this domain will be updated. The mathematical description is as follows:

$$\forall \rho_n > \rho_c, n \rightarrow e \in \Omega_e \tag{22}$$

where  $\rho_c$  is threshold value of nodal density;  $\Omega_e$  is the set of updated elements;  $n$  is the node ID, and  $e$  is the element ID.  $n \rightarrow e$  represents the elements mapped from the nodes whose nodal densities are bigger than  $\rho_c$ , and these elements are assigned to  $\Omega_e$ . The algorithm implementation is explained in Algorithm 1 (see Table 1).

The threshold value is smaller, the updated range is bigger. If the threshold value is 0, all elements will be updated. Vice versa, the threshold value is bigger, the updated range is smaller. If the threshold value is 1, no element will be updated.

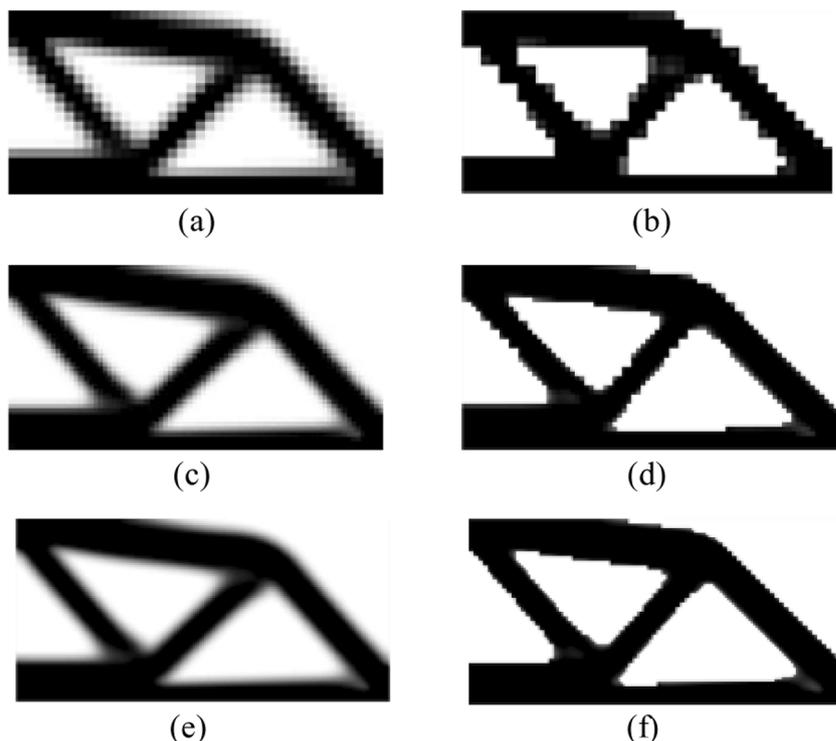
Another important parameter is the diffused layer number, which moves the boundary to the outside area. The aim of diffused layer number is to improve optimization accuracy with more design variables. A positive effect is to decrease the isolated gray elements (see in Sect. 5.2) outside the updated range. When the diffused layer number is 0, only the elements inside the blue boundary are updated. If the diffused layer number is 1, the boundary will move 1 layer toward outside area, as shown in Fig. 6. Larger diffused layer number means more elements will be included in the updated range. The mathematical description of one-layer moving is follows:

$$e \in \Omega_e \rightarrow n, \rho_n = 1 \tag{23}$$

$$\forall \rho_n = 1, n \rightarrow e \in \Omega_e \tag{24}$$

where  $\Omega_e$  is the set of updated elements;  $n$  is the node ID, and  $e$  is the element ID. Equation (23) represents the nodal density inside the updated range is assigned with 1. The elements, mapped from the nodes whose nodal density equal to 1,

**Fig. 14** The optimization results of Top88 and TO-TAM for different meshes. **a** Top88 with  $40 \times 20$ . **b** TO-TAM with  $40 \times 20$ . **c** Top88 with  $80 \times 40$ . **d** TO-TAM with  $80 \times 40$ . **e** Top88 with  $160 \times 80$ . **f** TO-TAM with  $160 \times 80$



should be assigned to  $\Omega_e$ . Repeating such process can move  $D_{num}$  layers. The diffused-layer algorithm is listed in Algorithm 2 (see Table 2).

Besides, how to choose  $D_{num}$  is influenced by the threshold value of nodal density  $\rho_c$ . Since  $D_{num}$  and  $\rho_c$  determine the updated range; if the  $\rho_c$  is too large, the  $D_{num}$  should increase to ensure a suitable updated range.

Furthermore, to improve the convergence rate of local-update strategy, two improving approaches are proposed. In traditional TOs, the optimization process always generates approximate structure in fewer iterations but takes a long history to converge. When the structure stabilizes around a certain shape, the local-update strategy can be adopted to accelerate the TO. Therefore, the moment when to start the local-update strategy plays an important role in the TO. By studying the TO, it can be found that the change value (the relative difference of the design variables between adjacent iterations) is smaller, and the structure change is smaller. Therefore, we define a startup-parameter  $ST$  for the moment of local-update strategy startup.  $ST$  is defined as the threshold of change value. When the change value is smaller than  $ST$ , the TO usually generates an approximate structure, after then local-update strategy is applied to achieve the final result. Combined with multilevel mesh, it can be easy to gain the approximate structure from the coarser mesh level; therefore, local-update strategy can start in a short time and furtherly reduce the number of iterations. Section 5.2.1 will explain how to choose the  $ST$  in detail.

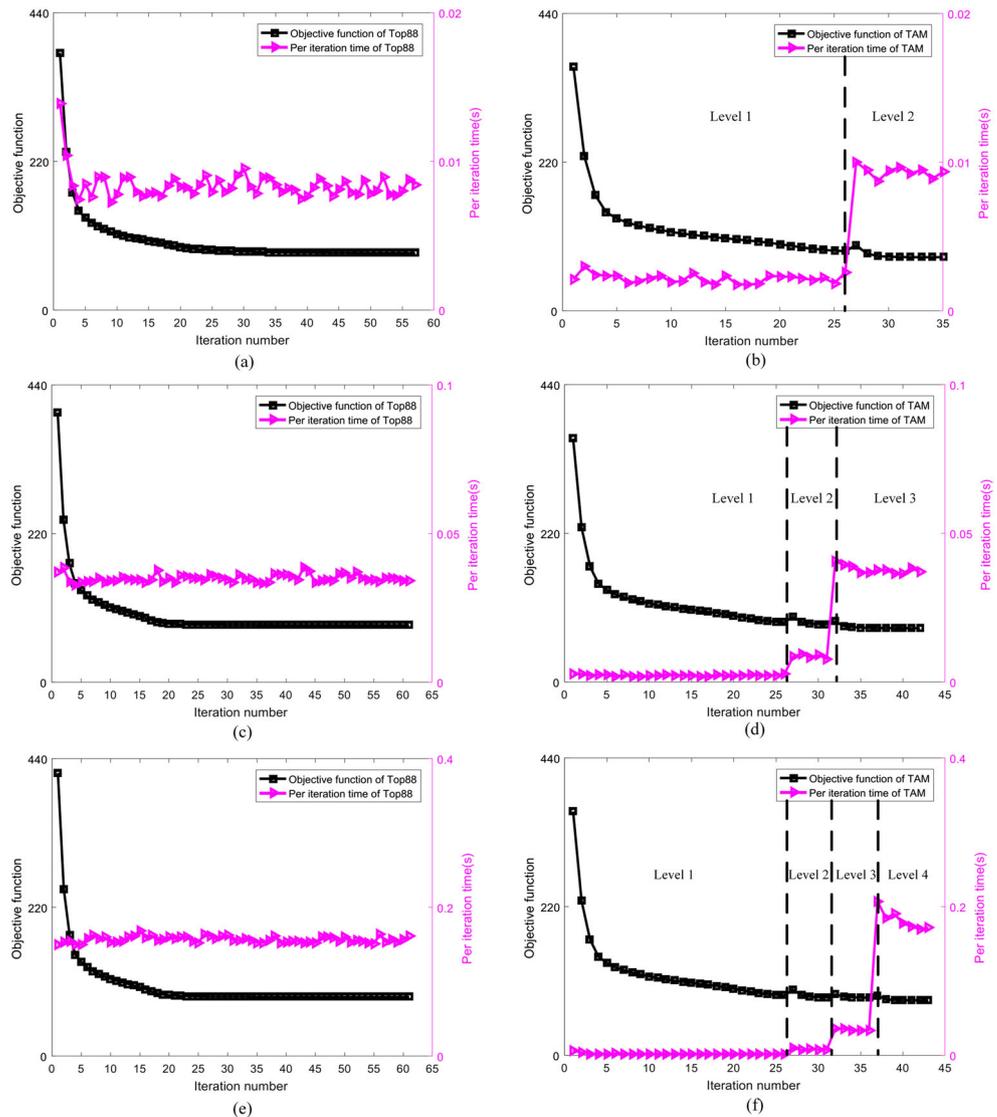
The elements with small density occupy most of computational resource when the volume fraction is small, but such elements have a small effect on the accuracy of computation and design. The other negative effect is from the gray elements, which cause TO difficult to obtain a black-and-white result. Therefore, the densities of small-density elements are directly set to the zero under certain conditions. The detail approach is defining a small-density threshold; the element density below the threshold will be set to zero. This density processing approach has two significant benefits: (1) accelerating the convergence rate, and (2) decreasing gray elements and increasing the fully solid elements. To ensure the optimal result meets the volume constraint, this process should perform before element density update process. It is worth noting that instable iterations will occur if the small-density threshold is bigger than volume fraction ( $VF$ ). Finally, the flowchart of local-update strategy is given in Fig. 7.

#### 4 Algorithm implementation

The triple acceleration algorithm (TAM) is developed based on MATLAB (Natick, MA, USA) in this work to solve the minimum compliance problem.

Figure 8 illustrates the flowchart of the TO using TAM (TO-TAM). In each mesh level, the initial-value-based PCG algorithm solves the  $KU = F$  for  $U$ , and local-update strategy is utilized. The convergence criterion is the relative difference of the design variables between adjacent iterations. In order to

**Fig. 15** The convergence history of Top88 and TO-TAM for different meshes. **a** Top88 with  $40 \times 20$ . **b** TO-TAM with  $40 \times 20$ . **c** Top88 with  $80 \times 40$ . **d** TO-TAM with  $80 \times 40$ . **e** Top88 with  $160 \times 80$ . **f** TO-TAM with  $160 \times 80$



accelerate the convergence rate, the coarser mesh level should use a loose convergence criterion.

### 5 Numerical examples

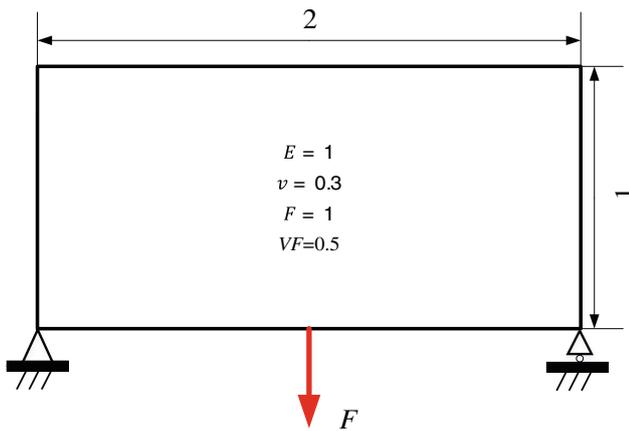
Three benchmark examples for minimum compliance design are examined in this section to demonstrate the acceleration

efficiency of triple acceleration algorithm. All examples are run on a desktop computer with CPU Intel core i7-6700 K of 4.00GHz, RAM of 16.0GB, and software environment MATLAB.

In Sect. 5.1, we examine a classical MBB beam to compare the computation efficiency of three parts of TAM versus the conventional TO (Andreassen et al. 2011), and study the accelerated efficiency of TAM on different meshes. In Sect. 5.2,

**Table 6** Efficiency comparison between Top88 and TO-TAM

Case	Compliance (Top88)	Compliance (TO-TAM)	Computational time (s) (Top88)	Computational time (s) (TO-TAM)	Speedup (TAM/Top88) $\frac{\text{time}_{\text{Top88}}}{\text{time}_{\text{TAM}}}$
$40 \times 20$	84.90	80.26	1.40	0.90	1.55
$80 \times 40$	84.85	79.93	3.22	1.61	2.00
$160 \times 80$	87.40	81.24	10.75	5.31	2.02



**Fig. 16** The design domain and boundary conditions of Michell type structure

we discuss how to choose the local-update strategy parameters ( $ST$  and  $D_{num}$ ) through the Michell type structure. Finally, Sect. 5.3 studies a larger-scale 3D cantilever beam to show the advantages of TAM for 3D cases.

## 5.1 MBB beam

Figure 9 shows the half MBB beam, a benchmark problem of TO (Andreassen et al. 2011; Sigmund 2001; Sigmund 2007). Due to the symmetry, only half MBB

**Table 7** Efficiency comparison between Top88 and local-update TO with different  $ST$

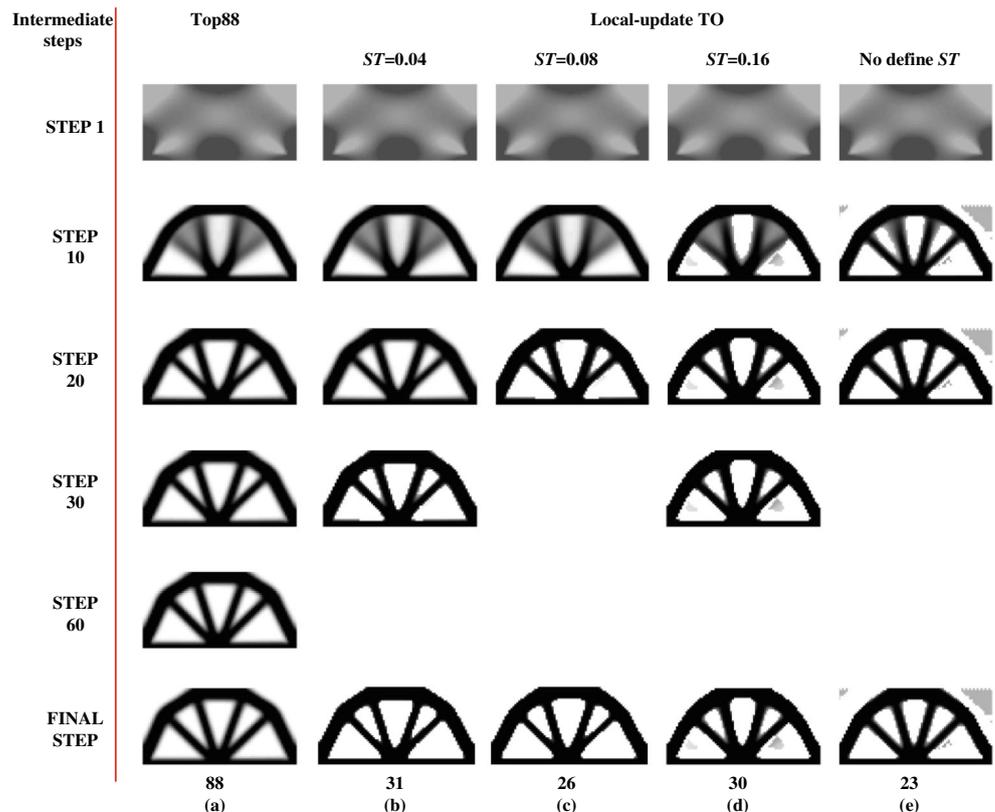
	Top88	Local-update TO			
		$ST=0.04$	$ST=0.08$	$ST=0.16$	Without $ST$
Compliance	15.55	14.34	14.39	14.41	14.66
Iteration number	88	31	26	30	23
Computational time (s)	4.46	1.82	1.69	2.00	1.54

beam is modeled and symmetry boundary conditions are utilized. This case study aims to compare the computation efficiency between three parts of TAM and the conventional TO, and research accelerated efficiency of TAM on different meshes.

### 5.1.1 Acceleration using multilevel mesh

The design domain is discretized with  $80 \times 40$  linear quadrilateral elements, and the volume fraction ( $VF$ ) of optimization is set to 0.5. Figure 10 shows the optimization result of conventional TO using the classical efficient 88-line TO code Top88 (Andreassen et al. 2011).

**Fig. 17** The intermediate and final optimized structures: (a) for Top88 and the rest are local-update TO with different  $ST$ : (b)  $ST=0.04$ , (c)  $ST=0.08$ , (d)  $ST=0.16$ , (e) no  $ST$



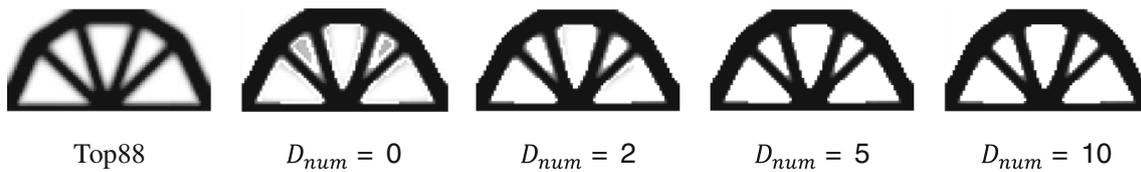


Fig. 18 The final optimized structures: left for Top88, the rest are local-update TO with different  $D_{num}$

For the  $80 \times 40$  half MBB beam, it is easy to be divided into three levels; the initial mesh is  $20 \times 10$ , the middle mesh is  $40 \times 20$ ; the final mesh is  $80 \times 40$ . The optimization results of each level are shown in Fig. 11.

Table 3 gives efficiency comparison between Top88 and TO with multilevel mesh algorithm. The iteration numbers corresponding to different levels from coarse to fine are 48, 25, and 18, i.e., 48 iterations for mesh level 1, 25 iterations for mesh level 2, and 18 iterations for mesh level 3 (the most refined mesh level). As expected, the computational time of the TO using multilevel mesh is less than that of Top88, demonstrating the advantage of multilevel mesh in acceleration efficiency. Due to the density field inheritance, it can be rapid to obtain an approximate result on the coarse mesh so that the optimization on the refined mesh can skip the lengthy iterations with fast convergence, where the iteration number of the final level is only 18. This result demonstrates that the multi-level mesh scheme has a great potential in TO acceleration.

### 5.1.2 Acceleration using initial-value-based PCG

Preconditioned conjugate-gradient method (PCG), an iterative solver, is usually used to solve large sparse linear equations. Therefore, in this section, the MBB beam is divided into  $100 \times 50$  elements to discuss the accelerated effect of the preconditioner and initial value  $U_{pre}$ . Figure 12 shows the final structures of TOs by using PCG under different conditions.

Table 4 shows the compliance values iteration number, average iteration number of PCG and computational time for the corresponding TOs in Fig. 12, which demonstrates that the preconditioner is significant for PCG with reducing over 90% computational time. Furthermore, the accelerated efficiency also has been improved by using the  $U_{pre}$ , with the lowest average iteration number of PCG and the minimum

Table 8 Efficiency comparison between Top88 and Local-update TO with difference  $D_{num}$

	Top88	Local-update TO			
		$D_{num} = 0$	$D_{num} = 2$	$D_{num} = 5$	$D_{num} = 10$
Compliance	15.55	14.69	14.40	14.36	14.34
Iteration number	88	30	26	28	27
Computational time (s)	4.46	1.54	1.66	1.69	2.04

computational time. Therefore, both of the preconditioner and initial value  $U_{pre}$  have positive effect on the TO acceleration.

### 5.1.3 Acceleration using local-update strategy

In the local-update strategy, startup-parameter  $ST$  is 0.04; the threshold value of nodal density  $\rho_c$  is 0.5; diffused layer number  $D_{num}$  is 8. The element density below 0.4 is set to zero in the iterations. The design domain is discretized with  $80 \times 40$  square elements, and the volume fraction is 0.5. The final structure is presented in Fig. 13.

Table 5 shows the the efficiency comparison between the TO with local-update strategy and Top88. Compared to Top88, the TO with local-update strategy costs nearly half iterations and less computational time, demonstrating local-update strategy as an excellent accelerating approach. Since the densities of small-density elements are reset to zero, the final optimal result contains less gray elements and more fully solid elements that will result in a clearer distinction between black and white structure. Furthermore, the compliance of local-update TO can be smaller. It is noted that the small-density reset just performs before density update, so the volume fraction keeps accuracy in optimization process. The optimal result meets the volume constraint with a lower compliance making the structure more accurate. More detail about local-update strategy will be given in Sect. 5.2.

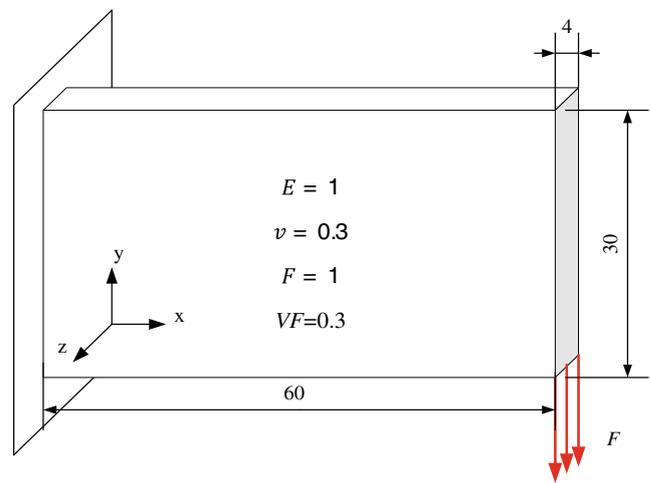
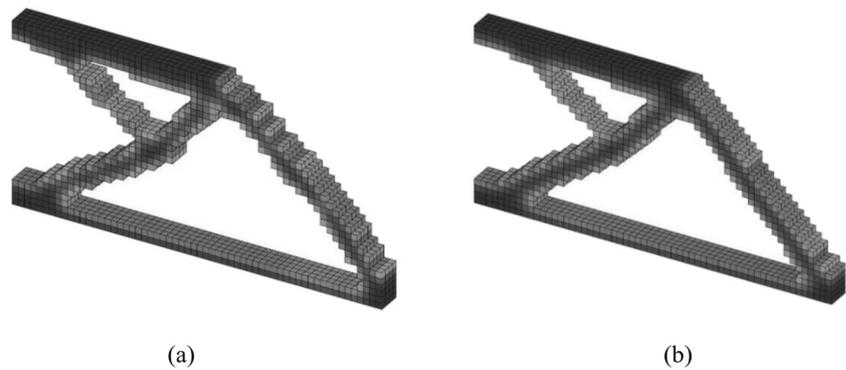


Fig. 19 The design domain and boundary conditions of 3D cantilever beam

**Fig. 20** The final structure optimized by **a** top3d and **b** TO-TAM



### 5.1.4 Acceleration using triple acceleration method

To further discuss the acceleration efficiency of the method proposed in this work, we have modeled the MBB beam with meshes of  $40 \times 20$ ,  $80 \times 40$ ,  $160 \times 80$  linear quadrilateral elements, respectively. The volume fraction ( $VF$ ) of optimization is set to 0.5. The comparison between the TO with triple acceleration method (TO-TAM) and Top88 on difference mesh are shown in Fig. 14.

Figure 15 presents the convergence history of Top88 and TO-TAM on different meshes and the efficiency comparison is given in Table 6. Since the initial element density is equal to  $VF$ , the volume fraction in optimization process is always equal to 0.5. From Fig. 15, the values of the objective function, i.e., compliance, decrease generally with the iteration number increasing in all cases, only have a small oscillation when the mesh level changes in the TO-TAM. Due to the multilevel mesh, per iteration time of coarser mesh levels is much smaller than that of the finest mesh level, while the per iteration time of the finest mesh level is closed to that of Top88 with the same mesh scale. Since the approximate structure can be inherited from coarser mesh levels by the multilevel mesh approach, the local-update strategy can start quickly and converge fast to further decrease the number of iterations. Besides, due to the usage of initial-value-based PCG, the speed of solving FE equations can be also accelerated in each iteration. From Table 6, the speedup value ranges from 1.55 to 2.02, meaning the TO-TAM has reduced 35–51% computational time compared to Top88, demonstrating the higher efficiency of TO-TAM. For the large-scale mesh, TO-TAM can achieve better accelerated effect. Since the densities of small-density elements are reset to zero in the local-update strategy,

smaller compliance values are obtained by the TO-TAM compared to the Top88, which further demonstrates accuracy of the TO-TAM.

## 5.2 Michell type structure

In this section, we discuss how to choose the parameters of local-update strategy, and the Michell type structure (Luo et al. 2009; Wang et al. 2017) is used here as an example. The design domain and boundary conditions are presented in Fig. 16. There are a fixed constraint at the bottom-left corner and a roller constraint at the bottom-right corner, as well as a mid-span point load.

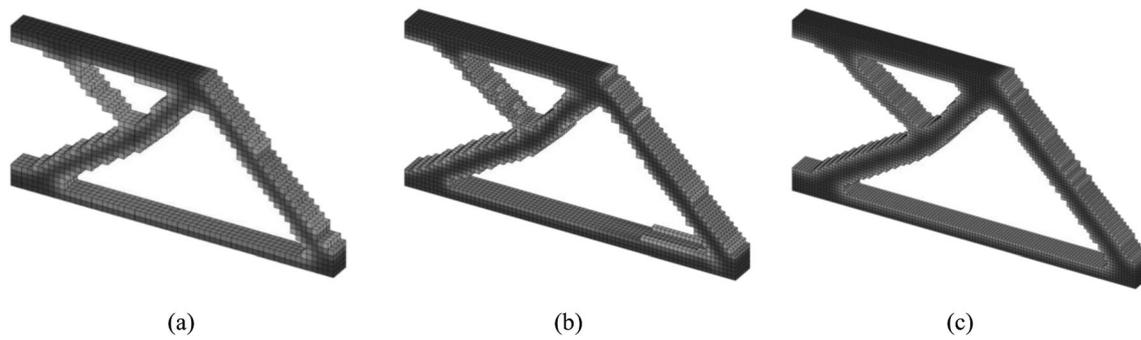
### 5.2.1 Startup parameter

By studying the conventional TO using the classical top88 code (Andreassen et al. 2011), there is a connection between the structure and change value: the change value is smaller and the structure change is smaller. When the structure stabilizes around a certain shape, the local-update strategy is applied. For the Michell type structure, we discuss various values of  $ST$ , defined as threshold of change value, and the local-update strategy is utilized when the change value is smaller than  $ST$ . The final structures of local-update strategy under different  $ST$  are presented in Fig. 17, among which no  $ST$  represents the TO starts directly with the local-update strategy. The Michell type structure is discretized into  $80 \times 40$  elements; the threshold value of nodal density  $\rho_c$  is 0.5, and the diffused layer number  $D_{num}$  is 8. The element density below 0.4 is set to zero in the iterations.

From Table 7, we can find that startup-parameter  $ST$  is better to chose nearby 0.08 for this benchmark problem. If  $ST$  is too big, the final structure will have isolated gray elements; if  $ST$  is too small, the acceleration efficiency is low. In addition, Table 7 demonstrates that starting TO process without defining  $ST$  has bad performance leading to lots of isolated gray elements, and the reason for this is that the TO starting directly with local-update strategy is not based on an approximate structure as that of the TO with a startup-paramter. In

**Table 9** Efficiency comparison between top3d and TO-TAM

	top3d	TO-TAM
Compliance	1139.48	935.84
Iteration number	113	45 + 32
Computational time (s)	534.25	108.35



**Fig. 21** The results of 3D TO-TAM for different meshes of **a**  $60 \times 30 \times 4$ , **b**  $80 \times 40 \times 6$ , and **c**  $120 \times 60 \times 8$

general, a suitable interval of  $ST$  (around 0.1) can obtain a high-speed and accurate TO result.

### 5.2.2 Diffused layer number

The another parameter to be discussed is diffused layer number  $D_{num}$ . During the local-update strategy, one problem is the isolated gray elements. In the local-update iterations, these isolated elements are outside the updated range, so these gray elements will not change and become isolated from the update-range. In order to solve this problem, the diffused layer is applied. For the  $80 \times 40$  Michell type structure, the final structures of the TOs with different diffused layer number  $D_{num}$  are shown in Fig. 18. The startup-parameter  $ST$  is set to 0.12; other parameters are the same as that in Sect. 5.2.1.

Table 8 lists the compliance values, iteration number, and computational time for the TOs illustrated in Fig. 18. We observe, as expected, small  $D_{num}$  leads to isolated gray elements and large  $D_{num}$  leads to more computational time. As  $D_{num}$  increases, the compliance decreases, also meaning the improvement of accuracy. For this benchmark problem,  $D_{num}$  should be bigger than 5. It should be noted the optimal  $D_{num}$  may be different for different cases, e.g.,  $D_{num}$  can be set to 2 for the 3D example in Sect. 5.3.

In this section, we find that both startup-parameter  $ST$  and diffused layer number  $D_{num}$  are of great importance in computational cost and accuracy for the local-update strategy. And for different TO problems, the parameters should be defined based on the actual conditions.

**Table 10** Comparison of 3D TO-TAM for different meshes

Case	Computational time (s)	Iteration number	Maximum memory allocation (MB)
$60 \times 30 \times 4$	108.35	45 + 32	873
$80 \times 40 \times 6$	492.44	53 + 45	1731
$120 \times 60 \times 8$	1685.32	45 + 32 + 37	4927

### 5.3 3D cantilever beam

Figure 19 shows an example of 3D cantilever beam (Liu and Tovar 2014), which is fully constrained in one end, and a unit distributed vertical load is applied downwards on lower free edge. The 3D cantilever beam is discretized into  $60 \times 30 \times 4$  linear hexahedron elements; the prescribed volume fraction  $VF$  is set to 0.3. By using the TAM method, 3D cantilever beam is divided into two levels, the coarser mesh is  $30 \times 15 \times 2$ ; the final mesh is  $60 \times 30 \times 4$ . Startup parameter  $ST$  is defined as 0.12; convergence criterion is 0.01; the threshold value of nodal density  $\rho_c$  is 0.5. Since the width is only equal to 4,  $D_{num}$  is suitably set to 2. The element density below 0.5 is set to zero. Finally, the final structure optimized by the TO-TAM is presented in Fig. 20b, as well as the final structure of conventional TO (using the classical top3d code by Liu and Tovar (2014)) in Fig. 20a.

Table 9 presents a clear comparison between TO-TAM and top3d. The iteration numbers of TO-TAM for different mesh levels from coarse to fine are 45 and 32, respectively, and TO-TAM is much faster than top3d (only costs 1/5 time of top3d). The compliance of TO-TAM is also less than top3d, which indicates that the optimal result of TAM is more accurate. This demonstrates that the use of TAM can also achieve a high-efficient 3D TO, thereby demonstrating the advantages of TAM for 3D cases.

To further discuss the capability of TAM for 3D cases, we test the 3D TO-TAM for three more refined cases with meshes of  $60 \times 30 \times 4$ ,  $80 \times 40 \times 6$  and  $120 \times 60 \times 8$  linear hexahedron elements, respectively, and the numbers of mesh levels for these cases are 2, 2, and 3.

The optimized results are shown in Fig. 21, and the comparison for the different cases is given in Table 10. Definitely, the computational time and maximum memory allocation increase with the mesh scale increasing. For a large-scale problem, it should be noted that the mesh can be divided into more than two levels as long as the TO on initial mesh level can obtain the approximate structure, e.g., three levels are used for the case with mesh of  $120 \times 60 \times 8$  elements. All the three cases successfully obtain consistent optimized results (see Fig. 21), and the iteration numbers for different mesh levels

from coarse to fine are 45 and 32 ( $60 \times 30 \times 4$  case); 53 and 45 ( $80 \times 40 \times 6$  case); and 45, 32, and 37 ( $120 \times 60 \times 8$  case), respectively, which demonstrates the significant convergence of the TO-TAM.

## 6 Conclusions

This paper has proposed a triple acceleration method (TAM) for topology optimization (TO), which includes three parts: multilevel mesh, initial-value-based PCG, and local-update strategy. Three benchmark examples have been tested to verify the proposed method. The first example compares the acceleration efficiency between three parts of TAM and the classical TO code Top88, and all the three parts can accelerate the TO; then, we examine TAM in different scales of mesh, and the results show that the TO using TAM has reduced 35–51% computational time compared to Top88, especially better for large-scale problems. The second example has illustrated the importance of parameters choice for the local-update strategy. The third example demonstrates that the TAM also has excellent performance for accelerating 3D TO problems, performs better than 2D TO problems, which reduces 80% computational time. All examples demonstrate that the proposed TAM provides an excellent acceleration for TOs. Although the current work takes SIMP as example, the TAM will be extended to other types of TO methods such as level-set-based TO and MMC-based TO (Wang and Benson 2016a; Xie et al. 2018) in the future, and combined with advanced methods such as isogeometric analysis (Wang and Benson 2016b; Wang et al. 2015a) and GPU parallel computing (Martínez-Frutos and Herrero-Pérez 2017; Xia et al. 2017).

**Replication of results** The proposed method is based on classical TO method SIMP, the implementation of which has been introduced detailly in this paper, making it easy to reproduce results. Besides, the proposed method is a part of a new project that forbids opening the source code.

**Funding information** 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).

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

## References

- Aage N, Andreassen E, Lazarov BS, Sigmund O (2017) Giga-voxel computational morphogenesis for structural design. *Nature*. 550:84–86. <https://doi.org/10.1038/nature23911>
- Aage N, Lazarov BS (2013) Parallel framework for topology optimization using the method of moving asymptotes. *Struct Multidiscip O* 47:493–505. <https://doi.org/10.1007/s00158-012-0869-2>
- Aage N, Nobel-Jørgensen M, Andreassen CS, Sigmund O (2013) Interactive topology optimization on hand-held devices. *Struct Multidiscip O* 47:1–6. <https://doi.org/10.1007/s00158-012-0827-z>
- Alexandersen J, Sigmund O, Aage N (2016) Large scale three-dimensional topology optimisation of heat sinks cooled by natural convection. *Int J Heat Mass Tran* 100:876–891. <https://doi.org/10.1016/j.ijheatmasstransfer.2016.05.013>
- Allaire G, Jouve F, Toader AM (2004) Structural optimization using sensitivity analysis and a level-set method. *J Comput Phys* 194: 363–393. <https://doi.org/10.1016/j.jcp.2003.09.032>
- Amir O (2015) Revisiting approximate reanalysis in topology optimization: on the advantages of recycled preconditioning in a minimum weight procedure. *Struct Multidiscip O* 51:41–57. <https://doi.org/10.1007/s00158-014-1098-7>
- Amir O, Aage N, Lazarov BS (2014) On multigrid-CG for efficient topology optimization. *Struct Multidiscip O*. 49:815–829. <https://doi.org/10.1007/s00158-013-1015-5>
- Amir O, Sigmund O (2011) On reducing computational effort in topology optimization: how far can we go? *Struct Multidiscip O*. 44:25–29. <https://doi.org/10.1007/s00158-010-0586-7>
- Amir O, Stolpe M, Sigmund O (2009) Efficient use of iterative solvers in nested topology optimization. *Struct Multidiscip O*. 42:55–72
- Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O (2011) Efficient topology optimization in MATLAB using 88 lines of code. *Struct Multidiscip O* 43:1–16. <https://doi.org/10.1007/s00158-010-0594-7>
- Augarde CE, Ramage A, Staudacher J (2006) An element-based displacement preconditioner for linear elasticity problems. *Comput Struct* 84:2306–2315. <https://doi.org/10.1016/j.compstruc.2006.08.057>
- Barrett R (1995) Templates for solution of linear systems: building blocks for iterative methods. Siam
- Bendsøe MP (1989) Optimal shape design as a material distribution problem. *Structural Optimization* 1:193–202
- Bendsøe MP, Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method. *Comput Methods Appl Mech Eng* 71:197–224. [https://doi.org/10.1016/0045-7825\(88\)90086-2](https://doi.org/10.1016/0045-7825(88)90086-2)
- Bendsee MP (1983) On obtaining a solution to optimization problems for solid, elastic plates by restriction of the design space\*. *Journal of Structural Mechanics* 11:501–521
- Benzi M (2002) Preconditioning techniques for large linear systems: a survey. *J Comput Phys* 182:418–477. <https://doi.org/10.1006/jcph.2002.7176>
- Bruns TE, Tortorelli DA (2001) Topology optimization of non-linear elastic structures and compliant mechanisms. *Comput Meth Appl Mech Eng* 190:3443–3459. [https://doi.org/10.1016/S0045-7825\(00\)00278-4](https://doi.org/10.1016/S0045-7825(00)00278-4)
- Cheng CH, Chen YF (2015) Topology optimization of conduction path in laminated metals composite materials. *Int J Therm Sci* 96:183–190. <https://doi.org/10.1016/j.ijthermalsci.2015.05.005>
- Christiansen RE, Lazarov BS, Jensen JS, Sigmund O (2015) Creating geometrically robust designs for highly sensitive problems using topology optimization: acoustic cavity design. *Struct Multidiscip O* 52:737–754
- Davis TA (2006). *Direct methods for sparse linear Systems*, Siam
- Deng X, Wang Y, Yan J, Liu T, Wang S (2016) Topology optimization of total femur structure: application of parameterized level set method under geometric constraints. *J Mech Des* 138:011402
- Desai J, Faure A, Michailidis G, Parry G, Estevez R (2018) Topology optimization in acoustics and elasto-acoustics via a level-set method. *J Sound Vib* 420:73–103. <https://doi.org/10.1016/j.jsv.2018.01.032>
- Dirker J, Meyer JP (2013) Topology optimization for an internal heat-conduction cooling scheme in a square domain for high heat flux applications. *J Heat Trans-T ASME* 135:111010. <https://doi.org/10.1115/1.4024615>

- Evgrafov A, Rupp CJ, Maute K, Dunn ML (2008) Large-scale parallel topology optimization using a dual-primal substructuring solver. *Struct Multidiscip O* 36:329–345. <https://doi.org/10.1007/s00158-007-0190-7>
- Gersborg-Hansen A, Bendsoe MP, Sigmund O (2006) Topology optimization of heat conduction problems using the finite volume method. *Struct Multidiscip O* 31:251–259. <https://doi.org/10.1007/s00158-005-0584-3>
- Gersborg-Hansen A, Sigmund O, Haber RB (2005) Topology optimization of channel flow problems. *Struct Multidiscip O* 30:181–192. <https://doi.org/10.1007/s00158-004-0508-7>
- Guest JK, Genut LCS (2010) Reducing dimensionality in topology optimization using adaptive design variable fields. *Int J Numer Methods Eng* 81:1019–1045
- Guo X, Zhang WS, Zhong WL (2014). Doing topology optimization explicitly and geometrically—a new moving morphable components based framework. *J Appl Mech-T ASME*. 81. doi:<https://doi.org/10.1115/1.4027609>
- Kim SY, Kim IY, Mechefske CK (2012) A new efficient convergence criterion for reducing computational expense in topology optimization: reducible design variable method. *Int J Numer Methods Eng* 90:752–783. <https://doi.org/10.1002/nme.3343>
- Kim YY, Yoon GH (2000) Multi-resolution multi-scale topology optimization—a new paradigm. *Int J Solids Struct* 37:5529–5559
- Lambe AB, Czekanski A (2018) Topology optimization using a continuous density field and adaptive mesh refinement. *Int J Numer Methods Eng* 113:357–373. <https://doi.org/10.1002/nme.5617>
- Langelaar M (2016) Topology optimization of 3D self-supporting structures for additive manufacturing. *Additive Manufacturing* 12:60–70. <https://doi.org/10.1016/j.addma.2016.06.010>
- Langelaar M (2017) An additive manufacturing filter for topology optimization of print-ready designs. *Struct Multidiscip O* 55:871–883. <https://doi.org/10.1007/s00158-016-1522-2>
- Li YF, Huang X, Zhou S (2016a) Topological design of cellular phononic band gap crystals. *Materials (Basel)* 9:186. <https://doi.org/10.3390/ma9030186>
- Li YF, Huang XD, Meng F, Zhou SW (2016b) Evolutionary topological design for phononic band gap crystals. *Struct Multidiscip O* 54:595–617. <https://doi.org/10.1007/s00158-016-1424-3>
- Lieu QX, Lee J (2017a) A multi-resolution approach for multi-material topology optimization based on isogeometric analysis. *Comput Methods Appl Mech Eng* 323:272–302
- Lieu QX, Lee J (2017b) Multiresolution topology optimization using isogeometric analysis. *Int J Numer Methods Eng* 112:2025–2047. <https://doi.org/10.1002/nme.5593>
- Lin CY, Chou JN (1999) A two-stage approach for structural topology optimization. *Adv Eng Softw* 30:261–271. [https://doi.org/10.1016/S0965-9978\(98\)00081-7](https://doi.org/10.1016/S0965-9978(98)00081-7)
- Lindgaard E, Dahl J (2013) On compliance and buckling objective functions in topology optimization of snap-through problems. *Struct Multidiscip O* 47:409–421. <https://doi.org/10.1007/s00158-012-0832-2>
- Liu K, Tovar A (2014) An efficient 3D topology optimization code written in Matlab. *Struct Multidiscip O* 50:1175–1196. <https://doi.org/10.1007/s00158-014-1107-x>
- Luo Z, Tong LY, Kang Z (2009) A level set method for structural shape and topology optimization using radial basis functions. *Comput Struct* 87:425–434. <https://doi.org/10.1016/j.compstruc.2009.01.008>
- Madah H, Amir O (2017) Truss optimization with buckling considerations using geometrically nonlinear beam modeling. *Comput Struct* 192:233–247. <https://doi.org/10.1016/j.compstruc.2017.07.023>
- Martínez-Frutos J, Herrero-Pérez D (2017) GPU acceleration for evolutionary topology optimization of continuum structures using isosurfaces. *Comput Struct* 182:119–136. <https://doi.org/10.1016/j.compstruc.2016.10.018>
- Mei YL, Wang XM (2004) A level set method for structural topology optimization and its applications. *Adv Eng Softw* 35:415–441. <https://doi.org/10.1016/j.advengsoft.2004.06.004>
- Nguyen TH, Paulino GH, Song J, Le CH (2010) A computational paradigm for multiresolution topology optimization (MTOP). *Struct Multidiscip O*. 41:525–539. <https://doi.org/10.1007/s00158-009-0443-8>
- Norgaard S, Sigmund O, Lazarov B (2016) Topology optimization of unsteady flow problems using the lattice Boltzmann method. *J Comput Phys* 307:291–307
- Panesar A, Brackett D, Ashcroft I, Wildman R, Hague R (2017) Hierarchical remeshing strategies with mesh mapping for topology optimisation. *Int J Numer Methods Eng* 111:676–700. <https://doi.org/10.1002/nme.5488>
- Papadarakakis M, Tsompanakis Y, Hinton E, Sienz J (1996) Advanced solution methods in topology optimization and shape sensitivity analysis. *Eng Comput* 13:57–90
- Saad Y (2003) Iterative method for sparse linear systems. Siam
- Sigmund O (2001). A 99 line topology optimization code written in Matlab. *Struct Multidiscip O*. 21:120–127. doi:DOI <https://doi.org/10.1007/s001580050176>
- Sigmund O (2007) Morphology-based black and white filters for topology optimization. *Struct Multidiscip O*. 33:401–424. <https://doi.org/10.1007/s00158-006-0087-x>
- Stainko R (2006) An adaptive multilevel approach to the minimal compliance problem in topology optimization. *Commun Numer Meth En* 22:109–118. <https://doi.org/10.1002/cnm.800>
- Wang S, de Sturler E, Paulino GH (2007) Large-scale topology optimization using preconditioned Krylov subspace methods with recycling. *Int J Numer Methods Eng* 69:2441–2468. <https://doi.org/10.1002/nme.1798>
- Wang Y, Benson DJ (2016a) Geometrically constrained isogeometric parameterized level-set based topology optimization via trimmed elements. *Front Mech Eng* 11:328–343
- Wang Y, Benson DJ (2016b) Isogeometric analysis for parameterized LSM-based structural topology optimization. *Comput Mech* 57:19–35
- Wang Y, Benson DJ, Nagy AP (2015a) A multi-patch nonsingular isogeometric boundary element method using trimmed elements. *Comput Mech* 56:173–191
- Wang Y, Wang Q, Deng X, Xia Z, Yan J, Xu H (2015b) Graphics processing unit (GPU) accelerated fast multipole BEM with level-skip M2L for 3D elasticity problems. *Adv Eng Softw* 82:105–118
- Wang Y, Xu H, Pasini D (2017) Multiscale isogeometric topology optimization for lattice materials. *Comput Methods Appl Mech Eng* 316:568–585
- Wang YQ, Kang Z, He QZ (2013) An adaptive refinement approach for topology optimization based on separated density field description. *Comput Struct* 117:10–22
- Xia Z, Wang Y, Wang Q, Mei C (2017) GPU parallel strategy for parameterized LSM-based topology optimization using isogeometric analysis. *Struct Multidiscip O* 56:413–434
- Xie X, Wang S, Xu M, Wang Y (2018) A new isogeometric topology optimization using moving morphable components based on R-functions and collocation schemes. *Comput Methods Appl Mech Eng* 339:61–90
- Xie YM, Steven GP (1993) A simple evolutionary procedure for structural optimization. *Comput Struct* 49:885–896
- Yoo J, Lee I (2018) 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 (eds) *Advances in Structural and Multidisciplinary Optimization*, Cham, 2018. Springer International Publishing, pp 967–978
- Zegard T, Paulino GH (2013) Toward GPU accelerated topology optimization on unstructured meshes. *Struct Multidiscip O* 48:473–485. <https://doi.org/10.1007/s00158-013-0920-y>

- Zhang W, Song J, Zhou J, Du Z, Zhu Y, Sun Z, Guo X (2018) Topology optimization with multiple materials via moving morphable component (MMC) method. *Int J Numer Methods Eng* 113:1653–1675. <https://doi.org/10.1002/nme.5714>
- Zhou SW, Li W, Li Q (2010) Level-set based topology optimization for electromagnetic dipole antenna design. *J Comput Phys* 229:6915–6930. <https://doi.org/10.1016/j.jcp.2010.05.030>
- Zhu J-H, Zhang W-H, Xia L (2016) Topology optimization in aircraft and aerospace structures design. *Arch Comput Method Eng* 23:595–622. <https://doi.org/10.1007/s11831-015-9151-2>