Struct Multidisc Optim (2015) 51:1159–1172
DOI 10.1007/s00158-014-1190-z

EDUCATIONAL ARTICLE

Matlab code for a level set-based topology optimization method using a reaction diffusion equation

Masaki Otomori · Takayuki Yamada · Kazuhiro Izui · Shinji Nishiwaki

Received: 10 March 2014 / Revised: 3 October 2014 / Accepted: 21 October 2014 / Published online: 19 November 2014

Abstract This paper presents a simple Matlab implementation for a level set-based topology optimization method in which the level set function is updated using a reaction diffusion equation, which is different from conventional level set-based approaches (Allaire et al. 2002, 2004; Wang et al. 2003) that use the Hamilton-Jacobi equation to update the level set function. With this method, the geometrical complexity of optimized configurations can be easily controlled by appropriately setting a regularization parameter. We explain the code in detail, and also the derivation of the topological derivative that is used in the level set-based topology optimization. Numerical results for stiffness maximization problems are provided to facilitate the reader's understanding. The presented code is intended for educational purposes only. This paper was inspired by previously published papers presenting Matlab code for a SIMP method (Sigmund 2001; Andreassen et al. 2011), a level set-based method (Challis 2010), and FreeFem++ code for a structural optimization method (Allaire and Pantz 2006). Readers can investigate results provided by these different methods and discover the prominent aspects of each particular method. The code presented here can be downloaded from http://www.osdel.me.kyoto-u.ac.jp/members/yamada/codes.html.

Keywords Education · Matlab code · Topology optimization · Level set method · Topological derivative · Reaction diffusion equation

M. Otomori · T. Yamada (✉) · K. Izui · S. Nishiwaki
Department of Mechanical Engineering and Science, Kyoto University, Kyotodaigaku-katsura, Nishikyo-ku, Kyoto 615–8540, Japan
e-mail: takayuki@me.kyoto-u.ac.jp

1 Introduction

This paper presents Matlab code for a level set-based topology optimization method incorporating a reaction diffusion equation for a compliance minimization problem. Concerning level set-based structural optimization methods, many have been proposed since the pioneering works by Allaire et al. (2002, 2004) and Wang et al. (2003). Programming code for a level set approach is freely available: Scilab code (Allaire et al. 2004) can be downloaded via http://www.cmap.polytechnique.fr/~allaire/levelset_en.html. Also, Challis (2010) presented Matlab code for a discrete level set approach, which is available in the manuscript. The Matlab code discussed in this paper can be used as an aid to understand the similarities and differences between this level set-based method and other topology optimization methods.

The level set method was first proposed by Osher and Sethian (1988) as a method to implicitly represent the evolution of interfaces. In such analyses, the evolution of interface boundaries is tracked by solving a so-called Hamilton-Jacobi equation, using an appropriate velocity normal to the interface of the moving boundary. In level set-based approaches, structural boundaries are represented by the zero iso-surface of the level set function and the level set function is defined so that the structural domain is represented wherever the level set function has a positive value. With respect to boundary representation methods, we classify the method presented here as a level set-based approach, which was developed by several of the authors (Yamada et al. 2010).

In this method, the level set function is updated by solving a reaction-diffusion equation based on the topological derivative of the objective functional. Therefore, this method allows topological changes that generate new boundaries during the optimization procedure and it does not require re-initialization of the level set function, which is typically required in Hamilton-Jacobi-based approaches to ensure accuracy when solving the Hamilton-Jacobi partial differential equation. In addition, our method enables qualitative control of the geometrical complexity of the optimized configurations by using appropriate values when setting a regularization parameter. In the following, we provide a rigorous derivation of the topological derivative for a mean compliance minimization problem. We note that although the term representing the effect of boundary conditions when a hole is created in the design domain was ignored in previous research (Yamada et al. 2010), this effect is now considered in the method presented here.

The rest of this paper is as follows. Section 2 describes the formulation of the level set-based topology optimization method using a reaction diffusion equation and the formulation of an optimization problem for a compliance minimization problem. Section 3 explains the Matlab code we use, and Section 4 provides numerical examples and an extension of the code. A conclusion is provided in Section 5, the Matlab code is provided in Appendix A, and the derivation of the topological derivative is provided in Appendix B.

2 Formulation

2.1 Topology optimization

For structural optimization, optimization problems are formulated using a domain Ω filled with a material domain, a void domain, and boundaries Γ, as follows:

$$\inf_{\Omega} \ F[\Omega] = \int_{\Omega} f_d(\boldsymbol{x}, \boldsymbol{x}, \nabla \boldsymbol{x}) d\Omega + \int_{\Gamma} f_b(\boldsymbol{x}, \boldsymbol{x}, \nabla \boldsymbol{x}) d\Gamma, \tag{1}$$

where f_d and f_b are arbitrary real functions defined for domain Ω and boundary Γ, respectively. $\boldsymbol{x}$ represents a point located in Ω and $\boldsymbol{x}$ represents the state variables.

In topology optimization, as Fig. 1 shows, the above optimization problem is replaced with a material distribution problem within a fixed design domain, D, using the characteristic function χ_Ω. The optimization problem is then defined as

$$\inf_{\chi_\Omega} F[\chi_\Omega] = \int_D f_d(\boldsymbol{x}, \chi_\Omega) \, d\Omega + \int_\Gamma f_b(\boldsymbol{x}, \chi_\Omega) \, d\Gamma, \tag{2}$$

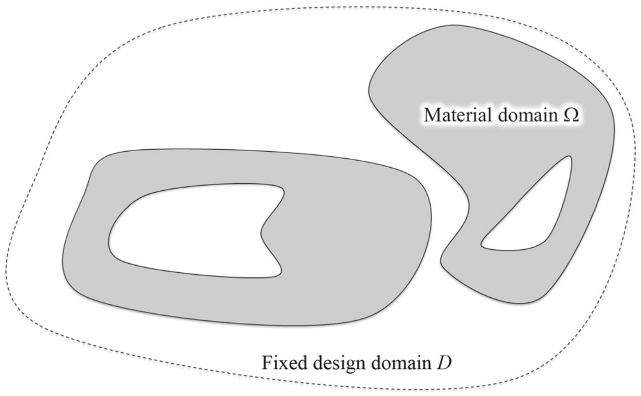

Fig. 1 Fixed design domain, D

where the χ_Ω is defined as

$$\chi_\Omega(\boldsymbol{x}) = \begin{cases} 1 \ \forall \boldsymbol{x} \in \Omega \\ 0 \ \forall \boldsymbol{x} \in D \setminus \Omega. \end{cases} \tag{3}$$

In the above optimization problem, topological changes such as an increase or decrease in the number of holes are allowed during the optimization procedure, in addition to changes in the shape represented by outer or inner boundaries. However, the continuity of the characteristic function is not guaranteed and the value of the characteristic function can be discontinuous at every point. That is, since the characteristic function χ_Ω is defined as a subset of a bounded Lebesgue space, L^∞, which only assures integrability, the obtained solutions can be discontinuous at every point in the fixed design domain. The ill-posed nature of the optimization problem is due to the lack of regularity of the admissible shapes.

In other words, because increasing the number of holes in a given design without changing its volume, by concomitantly decreasing their size, results in improved values of the objective functional, the solution of the optimization problem is rendered nonexistent. This is caused by the lack of closedness of the set of feasible designs. To overcome this obstacle, the optimization problem must be regularized so that a solution can exist. Homogenization-based approaches (Bendsøe and Kikuchi 1988; Allaire 2002) and fictitious density approaches such as the SIMP method (Bendsøe 1989) are two popular methods used to regularize optimization problems.

Homogenization-based approaches use homogenization theory to relax the design domain, that is, enlarge the set of possible designs. On the other hand, fictitious density approaches replace the characteristic function with a fictitious material whose elasticity tensor is assumed to be a continuous function of the material density. The SIMP (Solid Isotropic Material with Penalization) method is a type of fictitious density approach that uses a continuous fictitious isotropic material whose elasticity tensor is assumed to be a function of material density, penalized using an exponential parameter. In both the Homogenization-based approaches and fictitious density approaches, optimized configurations are represented as density distributions expressed using continuous values that range from 0 to 1. Therefore, the obtained optimized configurations often include grayscale areas where the density assumes an intermediate value between 0 and 1. Furthermore, the application of a boundary condition at the structural boundaries is problematic in these approaches because the structural boundaries are not clearly expressed.

To fundamentally overcome these problems, approaches based on a level set method were proposed (Allaire 2002, 2004; Wang et al. (Wang et al. 2003)), in which the boundaries of the optimal configuration are implicitly represented using a level set function. However, because these particular level set-based methods are based on a boundary advection concept, topological changes that generate new boundaries during the optimization procedure are not allowed, although the number of holes in an existing configuration can be decreased. We note that, for 3-dimensional problems, structural boundaries can easily merge to create holes in the fixed design domain, but this is different from the generation of holes inside the material domain, which is still disallowed. Therefore, although the above mechanism for the generation of new holes makes the method much more flexible, conventional level set-based approaches that use the Hamilton-Jacobi equation for advecting the shape are subject to limitations of topological changes even for 3-dimensional problems. As a result, the initial configuration settings greatly affect the obtained optimized configurations. To overcome this difficulty, Allaire et al. (2005) applied the bubble method (Eschenauer et al. 1994) to a level set-based structural optimization method in which boundaries are updated using the shape derivative, and holes are inserted based on the value of the topological derivative. In Allaire et al. (2005), hole nucleation based on the topological derivative is performed in several steps during the optimization procedure, as the level set function is updated using the shape derivative.

2.2 Level set-based topology optimization using a reaction diffusion equation

This subsection describes the level set-based topology optimization method applied in this paper (Yamada et al. 2010). In this method, the level set function is updated by solving a reaction-diffusion equation based on the topological derivative of the objective functional. Therefore, this method allows topological changes that generate new boundaries during the optimization procedure, and re-initialization of the level set function is not required.

As shown in Fig. 2, the structural boundaries in a level set-based topology optimization method are implicitly represented using the iso-surface of the level set function ϕ, as follows.

$$\begin{cases} 1 \geqslant \phi(\boldsymbol{x}) > 0 & \forall \boldsymbol{x} \in \Omega \setminus \partial\Omega \\ \phi(\boldsymbol{x}) = 0 & \forall \boldsymbol{x} \in \partial\Omega \\ 0 > \phi(\boldsymbol{x}) \geqslant -1 & \forall \boldsymbol{x} \in D \setminus \Omega . \end{cases} \tag{4}$$

The above level set function is used to represent the boundaries of the target structure; positive values represent the material domain, negative values represent the void domain, and zero represents the structural boundaries. The level set function has upper and lower limits imposed for the regularization term, used for regularizing the optimization problem, which will be explained in the next subsection. To avoid confusion concerning the sign of the level set function, we note that even though we define positive values of the level set function as corresponding to points in the material domain, as in (Wang et al. 2003; Yamada et al. 2010), an inverse definition where negative values correspond to points in the material domain, as in (Allaire 2002; Allaire et al. 2004; Challis 2010), is equally valid. We also note that the level set function used in the method presented here is not a signed distance function, such as is typically used in many level set-based approaches. The optimization problem that minimizes an objective functional F under an inequality constraint G is then formulated as follows, using the level set function ϕ defined above.

$$\inf_{\chi_\phi} \quad F[\chi_\phi] = \int_D f_d(\boldsymbol{x}, \chi_\phi) d\Omega + \int_\Gamma f_b(\boldsymbol{x}, \chi_\phi) d\Gamma \tag{5}$$

$$\text{subject to } G[\chi_\phi] = \int_D g(\boldsymbol{x}, \chi_\phi)\, d\Omega - G_{\max} \leqslant 0 , \tag{6}$$

where g is the density function and $G_{\max}$ is the upper limit value of the response $\int_D g(\boldsymbol{x}, \chi_\phi)\ d\Omega$. The characteristic function $\chi_\phi(\phi)$ is now defined as

$$\chi_\phi(\phi) = \begin{cases} 1 & \text{if } \phi \geqslant 0 \\ 0 & \text{if } \phi < 0 . \end{cases} \tag{7}$$

The above optimization problem is now replaced with an unconstrained optimization problem, using Lagrange's method of undetermined multipliers, as follows.

$$\inf_{\phi} \quad \bar{F}[\chi_\phi, \phi] = F + \lambda G , \tag{8}$$

where λ is the Lagrange multiplier. Based on the above formulation, the KKT (Karush-Kuhn-Tucker) conditions of this optimization problem are described as

$$\bar{F}' = 0, \quad \lambda G = 0, \quad \lambda \geqslant 0, \quad G \leqslant 0 , \tag{9}$$

where the notation $\bar{F}'$ represents the derivative of the Lagrangian $\bar{F}$.

Level set functions that satisfy the above KKT conditions are candidate solutions of the level set function that represent optimized configurations. However, it is nearly

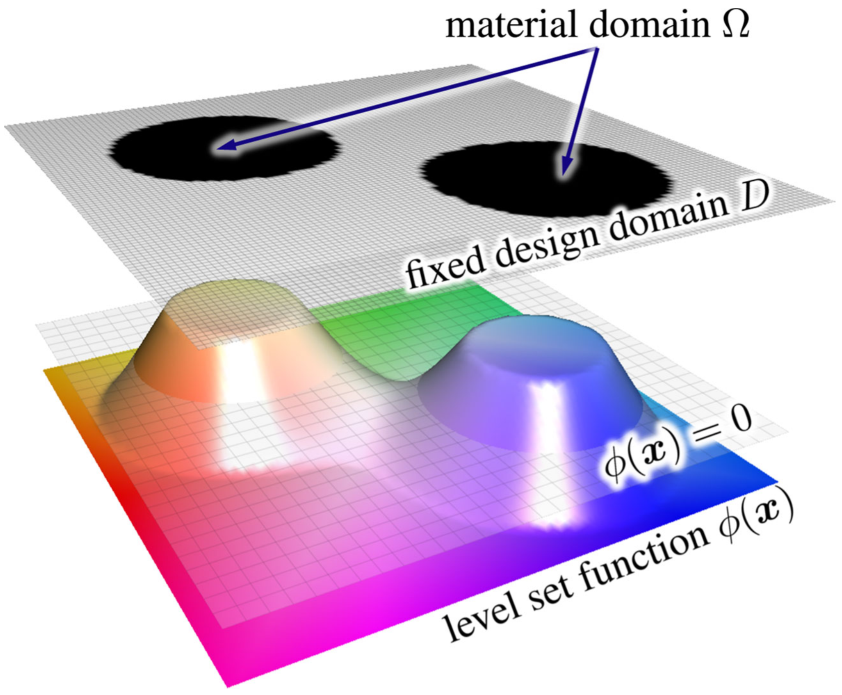
material domain Ω
fixed design domain D
$\phi(\boldsymbol{x}) = 0$
level set function $\phi(\boldsymbol{x})$

Fig. 2 Fixed design domain D and level set function ϕ

impossible to find such optimized solutions directly, so the optimization problem is replaced with a time evolution equation by introducing a fictitious time t. The level set function is then updated by solving this equation, and an optimized configuration is ultimately obtained, as explained below.

2.3 Time evolution equation

For the following formulation, which introduces fictitious time t, it is assumed that the variation of the level set function is proportional to the gradient of Lagrangian $\bar{F}$, as follows.

$$\frac{\partial \phi}{\partial t} = -K \bar{F}', \tag{10}$$

where $K > 0$ is a coefficient of proportionality. The optimization problem as formulated above is an ill-posed problem, so it is regularized by adding a regularization term, as follows.

$$\frac{\partial \phi}{\partial t} = -K \left(\bar{F}' - \tau \nabla^2 \phi \right), \tag{11}$$

The above equation is a reaction diffusion equation, and the diffusive term $\nabla^2 \phi$ ensures the smoothness of the level set function in the presented method. Furthermore, the value of the regularization parameter τ affects the degree of this diffusivity, with larger values of τ providing increased diffusivity for the level set function. Thus, an appropriately set value of τ can prevent the generation of structures that have excessive geometrical complexity, so that an optimized configuration that has a desired degree of geometrical simplicity can be obtained.

Upper and lower limits are imposed on the level set function so that the smoothing effect only operates on points that are close to the structural boundaries. Therefore, this regularization term functions implicitly as a kind of perimeter control. That is, the complexity of the optimized configuration can be controlled by adjusting the value of the regularization parameter τ. For relatively small values of τ, relatively complex optimized configurations are obtained, and the converse is true. The details are provided in the numerical examples. We note that adding the regularization term to (10) to obtain (11) makes it currently difficult to guarantee that the objective functional monotonically decreases, although the advection velocity is rigorously guaranteed to decrease in a descent direction in Allaire et al. (2004). We hope to address this issue in future research.

Setting appropriate boundary conditions for (11), the following equations are obtained.

$$\begin{cases} \dfrac{\partial \phi}{\partial t} = -K \left(\bar{F}' - \tau \nabla^2 \phi \right) & \text{in } D \\ \phi = 0 & \text{on } \partial D. \end{cases} \tag{12}$$

We note that although boundary ∂D is defined for values of the level set function that are equal to zero, this does not mean that ∂D represents structural boundaries. The use of this definition allows domains that are close to the boundary of the design domain to remain unaffected by boundary settings. Other boundary conditions might work fine. However, we use the Dirichlet boundary condition $\phi = 0$ on the boundary of the design domain for simplicity on implementation. The optimized configuration can now be obtained by solving the above time evolution problem. In this method, topological derivative $d_t\bar{F}$ is used for the $\bar{F}'$ term.

2.4 Optimization problem

Figure 3 shows the fixed design domain D and boundary conditions for a mean compliance minimization problem. The displacement is fixed at boundary Γ_u and traction t_i is applied at boundary Γ_t. The objective is to find the optimum layout of the design space that minimizes the mean compliance under a given volume constraint. The optimization problem is defined as

$$\inf_{\phi} \quad F = \int_{\Gamma} t_i u_i d\Gamma \tag{13}$$

$$\text{s.t.} \quad G = \int_{\Omega} d\Omega - V_{\max} \leqslant 0 \tag{14}$$

$$div\left(C_{ijkl}u_{k,l}\right) = 0 \quad \text{on} \quad \Omega \tag{15}$$

$$u_i = \bar{u}_i \qquad \text{in} \quad \Gamma_u \tag{16}$$

$$t_i = \bar{t}_i \qquad \text{in} \quad \Gamma_t\,, \tag{17}$$

where $V_{\max}$ is the upper limit of the volume constraint, C_{ijkl} is the elastic tensor, u_i is the displacement and $t_i = \sigma_{ij}n_j = C_{ijkl}u_{k,l}n_j$ is the traction. $\bar{u}_i$ and $\bar{t}_i$ are constant values that represent the given displacement and traction, respectively.

Using Lagrange's method of undetermined multipliers, the above optimization problem is replaced with an unconstrained optimization problem, using Lagrangian $\bar{F}$, and Lagrange multiplier $\tilde{u}_i$ and λ, as follows:

$$\bar{F} = \int_{\Gamma} t_i u_i d\Gamma + \int_{\Omega} \tilde{u}_i div\left(C_{ijkl}u_{k,l}\right) d\Omega + \lambda\left(\int_{\Omega} d\Omega - V_{\max}\right). \tag{18}$$

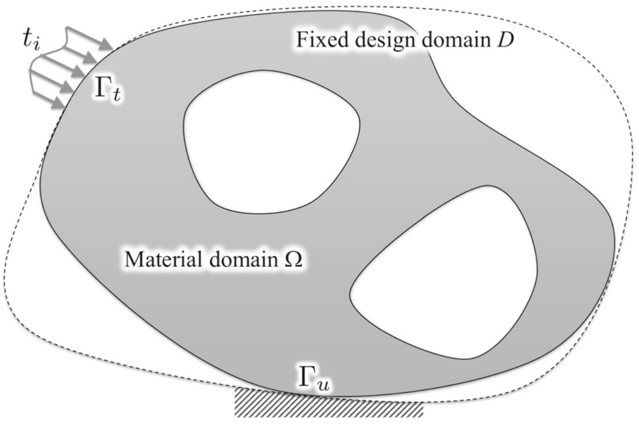

Fig. 3 Fixed design domain D and boundary conditions

The second term on the right-hand side in the above equation can be replaced using Green's formula, as follows:

$$\bar{F} = \int_{\Gamma} t_i u_i d\Gamma + \int_{\Gamma} \tilde{u}_i t_i d\Gamma - \int_{\Omega} \tilde{u}_{i,j} C_{ijkl} u_{k,l} d\Omega + \lambda\left(\int_{\Omega} d\Omega - V_{\max}\right). \tag{19}$$

2.5 Topological derivative

Figure 4 shows the concept of the topological derivative, which is a measure of the influence when a hole Ω_ε with radius ϵ is created at a certain point in the domain Ω. Γ_ϵ represents the boundary of the created hole. The topological derivative d_tF of objective functional F is defined as

$$d_tF := \lim_{\varepsilon \to 0} \frac{(F + \delta F) - F}{\text{meas}(\Omega \setminus \Omega_\varepsilon) - \text{meas}(\Omega)}\,. \tag{20}$$

The topological derivative for Lagrangian (19) is given as follows.

$$d_t\bar{F} = \lim_{\epsilon \to 0} \frac{\delta \bar{F}}{\frac{4\pi\epsilon^3}{3}} = \tilde{u}^0_{i,j} A_{ijkl} u^0_{k,l} - \lambda\,, \tag{21}$$

where the superscript 0 indicates the value without creating holes, and A_{ijkl} is defined as follows:

$$A_{ijkl} = \frac{3(1-\nu)}{2(1+\nu)(7-5\nu)}\left[\frac{-(1-14\nu+15\nu^2)E}{(1-2\nu)^2}\delta_{ij}\delta_{kl} + 5E(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})\right]. \tag{22}$$

The details for the derivation of topological derivative are provided in Appendix B.

3 Implementation

3.1 Reaction-diffusion equation

We now present a scheme for implementing the reaction-diffusion equation (Yamada et al. 2010). As discussed above, the level set function is updated using a reaction diffusion equation, and the diffusive term $\nabla^2\phi$ ensures that the optimization has sufficient smoothness and numerical stability.

First, we introduce a parameter C to normalize the sensitivities so that the value of τ can be chosen regardless of the

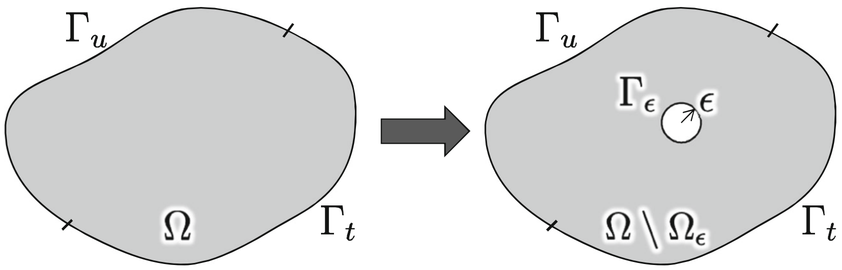

Fig. 4 Concept of the topological derivative

particular problem being solved. We then replace (12) with dimensionless equations, as follows.

$$\begin{cases} \dfrac{\partial \phi}{\partial t} = -\left(-Cd_t\bar{F} - \tau\nabla^2\phi\right) \text{ in } & D \\ \phi = 0 & \text{on } \partial D. \end{cases} \tag{23}$$

Here, K in (12) is set to 1. The effect of this term is explained in Yamada et al. (2010). We note that the $\bar{F}'$ term in (12) is replaced by $-Cd_t\bar{F}$. The opposite in sign is due to the relationship between the definitions of topological derivative and the sign of level set function. C is defined so that

$$C = \frac{\int_D \mathrm{d}\Omega}{\int_D \mid d_t F \mid \mathrm{d}\Omega}. \tag{24}$$

Using the finite difference approach, the equations in (23) are discretized in the time direction as follows:

$$\begin{cases} \dfrac{\phi(t+\Delta t)}{\Delta t} - \tau\nabla^2\phi(t+\Delta t) = Cd_t\bar{F} + \dfrac{\phi(t)}{\Delta t} \\ \phi = 0 \text{ on } \partial D, \end{cases} \tag{25}$$

where Δt is the step for fictitious time t. Using the finite element approach, the above equations are expressed in weak form as follows:

$$\begin{cases} \int_D \frac{\phi(t+\Delta t)}{\Delta t}\tilde{\phi} dD + \int_D \nabla^T\phi(t+\Delta t)\left(\tau\nabla\tilde{\phi}\right)dD \\ \quad = \int_D\left(Cd_t\bar{F} + \frac{\phi(t)}{\Delta t}\right)\tilde{\phi}dD \quad \text{for } \forall\tilde{\phi} \in \tilde{\Phi} \\ \phi = 0 \text{ on } \partial D, \end{cases} \tag{26}$$

where $\tilde{\Phi}$ is the functional space defined such that

$$\tilde{\Phi} = \left\{\phi(\boldsymbol{x}) | \phi(\boldsymbol{x}) \in H^1(D) \text{ with } \phi = 0 \text{ on } \partial D\right\}. \tag{27}$$

Discretizing (26) using the finite element method, we obtain the following:

$$\begin{cases} \mathbf{T}\,\Phi(t+\Delta t) = \mathbf{Y} \\ \phi = 0 \quad \text{on} \quad \partial D, \end{cases} \tag{28}$$

where $\Phi(t)$ is a vector that expresses the nodal value of the level set function at time t. $\mathbf{T}$ and $\mathbf{Y}$ are now described as

$$\mathbf{T} = \bigcup_{e=1}^{N} \int_{V_e} \left(\frac{1}{\Delta t}\mathbf{N}^T\mathbf{N} + \nabla^T\mathbf{N}\tau\nabla\mathbf{N}\right)dV_e \tag{29}$$

$$\mathbf{Y} = \bigcup_{e=1}^{N} \int_{V_e} \left(Cd_t\bar{F} + \frac{\phi(\boldsymbol{x},t)}{\Delta t}\right)\mathbf{N}dV_e, \tag{30}$$

where N is the number of elements and V_e is the volume of an element. $\bigcup_{e=1}^{N}$ represents the union set of the elements, where e is the element number. $\mathbf{N}$ is the interpolation function of the level set function.

After updating the level set function using (28), the level set function is replaced based on the following rule, so that the upper and lower limit constraints of the level set function in (4) are satisfied.

$$\text{if} \quad \|\phi\| > 1 \quad \text{then} \quad \phi = \text{sign}(\phi). \tag{31}$$

3.2 Volume constraint

The volume constraint is dealt with using the augmented Lagrangian method, in which the Lagrange multiplier λ, is updated according to the following scheme.

$$\lambda = \frac{\int_D d_t F d\Omega}{\int_D d\Omega} \exp^{\left[p\left(\frac{G}{G_{\max}}+d\right)\right]}, \tag{32}$$

where p and d are parameters that adjust the position of the curve. This scheme can be considered as a modified version of the augmented Lagrangian method, in which Lagrange multiplier λ is updated as $\lambda^{i+1} = \lambda^i + cG$, where c is a penalty parameter. We note that the Maclaurin expansion of an exponential function is expressed as $e^x = 1 + x + \frac{1}{2}x^2 + \cdots$, so the standard updating scheme can be considered as a first-order expansion of the above updating scheme when the volume constraint is active, with the value of $p\lambda^i$ used as the penalty parameter c.

When the constraint is sufficiently satisfied, that is, when the value of G is very small, the value of λ in (32) approaches 0. The sensitivity of the constraint functional then becomes relatively small compared with that of the objective functional, in the sensitivity expression $F' + \lambda G'$. In this way, the optimization is primarily affected by the value of the objective functional. On the other hand, when the constraint is far from being satisfied, the value of λ becomes very large, causing the sensitivity of the constraint functional to become relatively large compared with that of the objective functional. When this occurs, the optimization is primarily affected by the need to satisfy the constraint functional. Although other methods can be applied to satisfy the volume fraction, we use this scheme for its simplicity.

When the volume fraction of an initial guess is greater than the maximum allowable volume fraction $V_{\max}$, the volume constraint is relaxed according to the following formula, to stabilize the convergence.

$$G = \int_D \chi_\phi d\Omega - V_{\max} - (V_0 - V_{\max}) \max\left(0, 1 - \frac{i}{n_{\text{vol}}}\right) \leqslant 0, \tag{33}$$

where i is the current iteration number and V_0 is the volume fraction of the initial guess. The first term represents the volume of the configuration at the current iteration. The third term in the right-hand side of (33) is added to the primal volume constraint to relax the upper limit of the volume constraint so that the constraint is gradually tightened during n_{vol} iterations. After n_{vol} iterations, the constraint functional (33) represents the original volume constraint.

4 Matlab code

The simple Matlab code for the mean compliance minimization problem is provided in Appendix A. The program consists of four parts: parameter definitions (lines 3–13), finite element analysis preparation (lines 14–40), loads and boundary settings (lines 41–53), and the main loop (lines 54–88). Figure 5 shows the design domain and boundary conditions of the problem. Displacement is fixed at the left boundary and a downward force is applied at the center of the right boundary. The design domain is discretized using a rectangular mesh. The numbers of elements are nelx in the horizontal direction and nely in the vertical direction.

The Matlab code is evoked with the following call:

levelset88(nelx,nely,V_{max},tau),

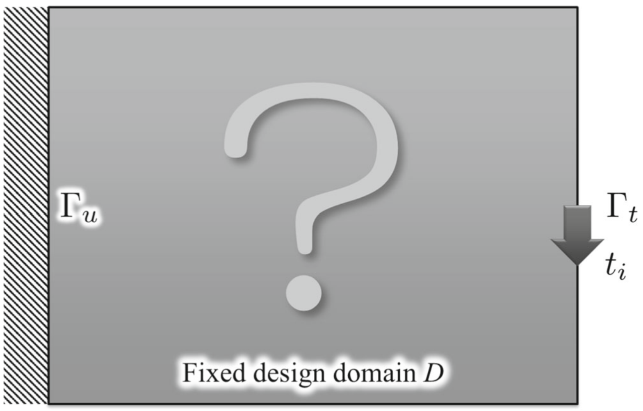

Fig. 5 Fixed design domain and boundary conditions of design model 1

where V_{max} is the upper limit of the volume fraction and tau is the regularization parameter τ.

4.1 Parameter definition: lines 3–13

Explanations for the various parameters are given in Table 1. E0, Emin, and nu are parameters used for analysis, Young's modulus in the material and void domains, and Poisson's ratio, respectively. nvol corresponds to the iteration number for the volume constraint, n_{vol} in (33). When the initial volume is greater than the upper limit of the volume constraint $V_{\max}$, the volume constraint is relaxed, and then it is gradually tightened during the iterations prescribed by nvol. dt, d, and p are optimization parameters. dt corresponds to the fictitious time step Δt in (29) and (30). d and p are parameters used in the updating scheme applied to Lagrange multiplier λ for the volume constraint in the augmented Lagrangian method formulated in (32). phi and str represent the level set function ϕ, and the material distribution, respectively.

4.2 Preparation of finite element analysis: lines 14–40

These lines of code define the preparation for solving the displacement field and reaction diffusion equation, and computing the topological derivative. This code follows the same procedure for computing the global stiffness matrix as that of Andreassen et al. (2011), where assembly of the global stiffness matrix is efficiently performed using the sparse function in Matlab. This procedure is advantageous from the standpoint of computation time because it avoids the use of for loops. Lines 16–19 define the components of the element stiffness matrix KE for the displacement field and these are assembled to create the global stiffness matrix K at each iteration.

Table 1 Parameter definition

parameter	meaning
nelx*	number of elements in x direction
nely*	number of elements in y direction
Vmax*	maximum allowable volume
tau*	regularization parameter
E0	Young's modulus in material domain
Emin	Young's modulus in void domain
nu	Poisson's ratio
nvol	iteration number for volume constraint
dt	step size for fictitious time t
d, p	parameter for augmented Lagrangian method

$*$: set according to the user's function call

Due to the similarity in the formulation of the topological derivative $\tilde{u}^0_{i,j} A_{ijkl} u^0_{k,l}$ defined in (22), and the mutual strain energy density $\tilde{u}_{i,j} K_{ijkl} u_{k,l}$ (Nishiwaki et al. 1998; Howell 2001), the topological derivative can be computed using the same procedure as that for computing the mutual strain energy density, by replacing Young's modulus E and Poisson's ratio ν as follows:

$$E \to \frac{4A_2^2}{A_1 + 2A_2}, \tag{34}$$

$$\nu \to \frac{A_1}{A_1 + 2A_2}, \tag{35}$$

where

$$A_1 = -\frac{3(1-\nu)\left(1 - 14\nu + 15\nu^2\right)}{2(1+\nu)(7-5\nu)(1-2\nu)^2} E, \tag{36}$$

$$A_2 = \frac{15E(1-\nu)}{2(1+\nu)(7-5\nu)}. \tag{37}$$

The detailed derivation of this relationship is provided in Appendix C. a1 and a2 in the code correspond to A_1 and A_2 in the above equation, respectively, and A corresponds to A_{ijkl}. We note that when the given displacement at boundary Γ_u is set to $\bar{u} = 0$ as in Fig. 5, the adjoint problem (46)–(48) is equivalent to the state problem (15)–(17). Therefore the optimization problem becomes self-adjoint problem and $\tilde{u}_{i,j} = u_{i,j}$.

nodenrs is a matrix that indicates the node number for all elements and edofVec is a vector representing the first DOF index for all elements. The i-th column of edofMat consists of the node number of the i-th element. iK and jK are index vectors used when assembling the global stiffness matrix using the element stiffness matricies. Details of this procedure are given in Andreassen et al. (2011). Next, lines 30 through 40 compute the global matrices $\mathbf{N}^T\mathbf{N}$, $\nabla\mathbf{N}^T\nabla\mathbf{N}$ for the reaction-diffusion equation. NN and NNdif respectively correspond to $\mathbf{N}^T\mathbf{N}$ and $\nabla\mathbf{N}^T\nabla\mathbf{N}$ defined in (29). Although the number of variables defined for each node is different for the displacement field (two variables: u, v) and the level set function (one variable: ϕ), the procedure for computing NN and NNdif is same as that for computing global stiffness matrix K.

4.3 Loads and boundary settings: lines 41–53

Lines 42–48 define the boundary conditions for the displacement field. Line 45 defines the load vector that is applied at the center of right-hand boundary. The length of the line where the load is applied is 1/16th that of the entire length of the right-hand boundary. Line 49 defines the stiffness matrix T shown in (29) for the reaction-diffusion equation. The load vector Y shown in (30) for the reaction-diffusion equation is defined in the main loop, since it includes the level set function that varies during optimization. Lines 49–53 define the boundary conditions for the reaction-diffusion equation.

4.4 Optimization loop: lines 54–88

The optimization loop consists of four parts: finite element analysis and sensitivity computation, convergence check, computation of Lagrange multiplier λ for the volume constraint, and the level set function update.

4.4.1 Finite element analysis and sensitivity computation

sK is a vector that is created by reshaping the element stiffness matrix, and global stiffness matrix K is constructed using the sparse function. SED and TD are the strain energy density and topological derivative, respectively. TD describes the value of the topological derivative in each element and these values are mapped onto nodes. TDN is the value of the topological derivative used on the nodes. In line 65 and 66, the objective functional, objective, and the volume of the current configuration, vol, are calculated. Lines 67–70 print the optimization results and plot the material distribution str.

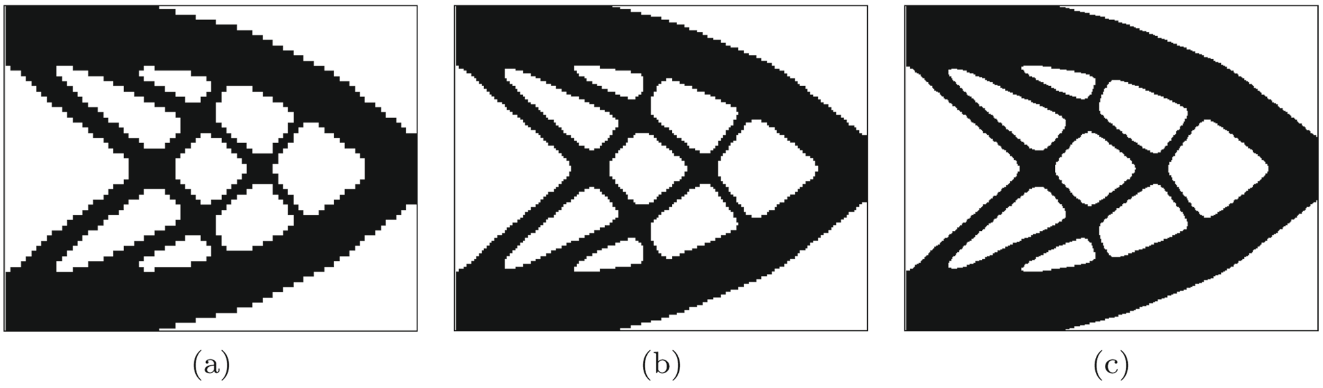

Fig. 6 Optimized configurations for design model 1 obtained using different meshes: **a** 80×64; **b** 160×128; **c** 320×256

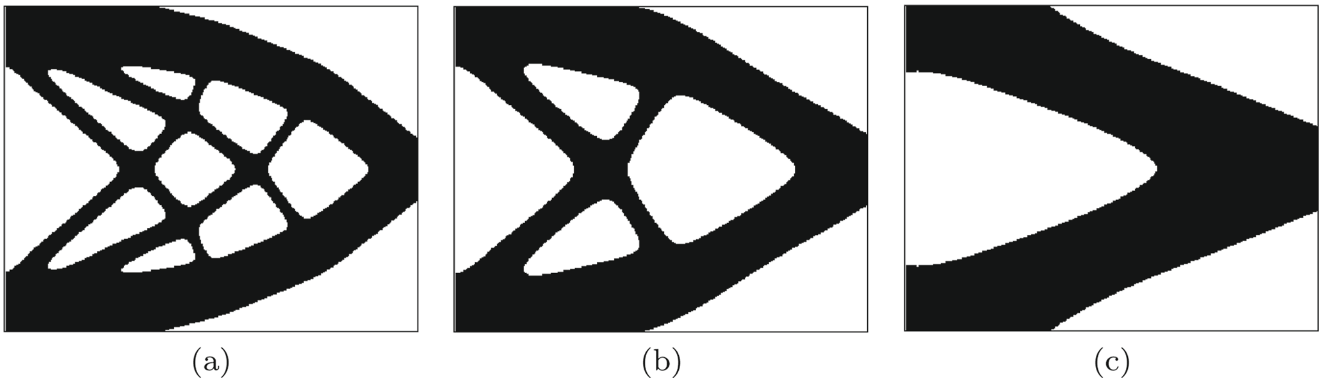

Fig. 7 Optimized configurations for design model 1 obtained using different regularization parameter τ settings: **a** 2×10^{-4}; **b** 6×10^{-4}; **c** 1×10^{-2}

4.4.2 Convergence check

Lines 72–75 are for the convergence check of the algorithm. The convergence check is performed after nvol iterations. The optimization terminates if both of the following conditions are satisfied: the volume is within 0.005 of the required value, $V_{\max}$, and the five previous values of the objective functional differ by less than 1 %.

4.4.3 Updating the Lagrange multiplier for the volume constraint

ex corresponds the second and third terms in (33), and indicates the relaxed upper limit of the volume constraint so that the constraint will be gradually tightened during nvol iterations. lambda corresponds to the Lagrange multiplier λ defined in (32). Line 79 computes the normalization parameter C defined in (24). And g2 in line 80 expresses the nodal topological derivative in vector notation.

4.4.4 Updating the level set function

Matrix Y is computed based on the Lagrange multiplier lambda (line 82) and the level set function phi obtained during the previous iteration. The level set function phi is then updated by solving the reaction-diffusion equation (line 83). The updated level set function may not satisfy the upper and lower limit constraints of the level set function. To ensure that these constraints are satisfied, the level set function is replaced based on the rule described in (31) (line 84 in the code). Nodal level set function phin is mapped onto the elements. An updated structure is obtained using the elemental level set function phie. The process then returns to the first step of the optimization loop.

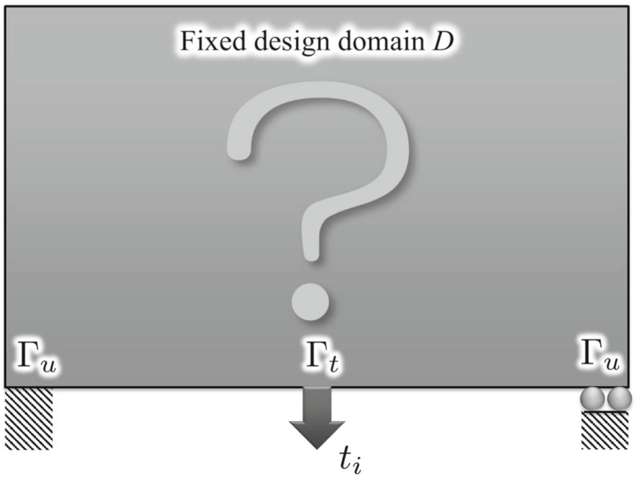

Fig. 8 Fixed design domain and boundary conditions of design model 2

5 Numerical examples

5.1 Mesh independency

First, we examine three cases whose degree of discretization is subject to the following mesh parameters: 80×64, 160×128, and 320×256. The regularization parameter τ is set to 2×10^{-4} and a volume fraction upper limit of 0.5 is used in all cases. The Matlab code calls for these three cases are respectively:

```
levelset88(80,64,0.5,2e-4),
levelset88(160,128,0.5,2e-4), and
levelset88(320,256,0.5,2e-4).
```

Figure 6 shows the optimized configurations for each case. The results indicate that the optimized configuration has minimal dependency on the mesh discretization.

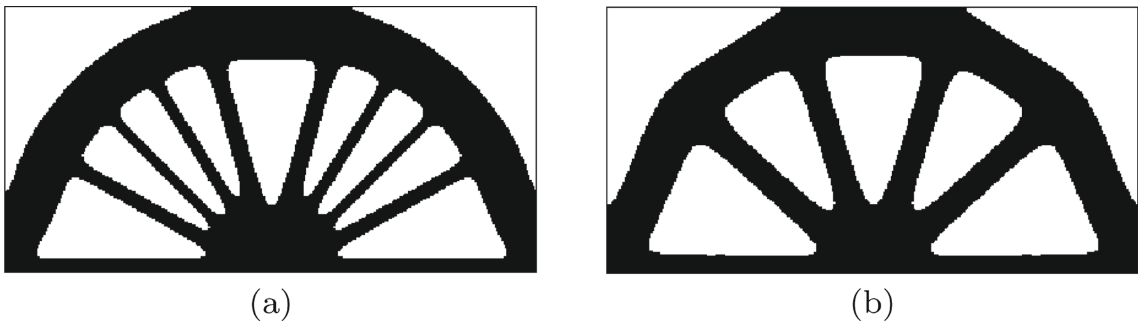

Fig. 9 Optimized configurations for design model 2 obtained using different values for the regularization parameter τ: **a** 2×10^{-4}; **b** 6×10^{-4}

5.2 Effect of the regularization parameter τ

Next, we examine three cases in which different degrees of diffusivity are imposed by setting parameter τ to the following values: $\tau = 2 \times 10^{-4}$, 6×10^{-4}, and 1×10^{-2}. A mesh discretization of 320×256 and a volume fraction upper limit of 0.5 is used in all cases. Figure 7 shows the optimized configurations for each case. The results show that relatively complex optimized configurations are obtained when τ is set to smaller values, and *vice versa*. We note that in stiffness maximization problems for cantilever structures, a structure with an infinite number of infinitely thin members is known to be optimal. Therefore, as the regularization parameter is set to smaller values, the optimized configuration becomes more complex and correspondingly better values of the objective functional are obtained. The results indicate that essentially any desired degree of geometrical complexity in the optimized configuration can be obtained, by setting the regularization parameter τ to an appropriate value.

5.3 Other boundary conditions

Last, we offer an additional example in which different boundary conditions are used. Figure 8 shows the analysis domain and boundary conditions. A load is applied at the bottom center of the analysis domain. Vertical and horizontal displacements at the bottom left corner of the analysis domain are fixed, as is the horizontal displacement at the bottom right corner of the analysis domain. The design domain is symmetric along the y-axis and a roller constraint is imposed on the boundary at the bottom right corner, so only the right half of the design domain is considered in the optimization problem.

To enable different load and boundary conditions for this optimization problem, lines 44, 45, 49 and 50 of the code are changed, as follows.

Line 44:
```
F(2*(nely+1):2*(nely+1):2*(nely+1)*
(round(nelx/32)+1),1) = 1;
```

Line 45:
```
fixeddofs = [1:2:2*(nely+1) 2*(nely+1)*
(nelx+1-round(nelx/32)):2*(nely+1):2*
(nely+1)*(nelx+1)];
```

line 49, 50:
```
fixeddofs_phi = [];
```

Figure 9 shows the optimization results obtained using two different settings for the regularization parameter τ. A mesh discretization of 120×120 (half model) and a volume fraction upper limit of 0.5 is used in both cases. We can confirm that the optimization obtained appropriate optimized configurations.

6 Conclusions

This paper presented Matlab code for a level set-based topology optimization method that uses a reaction diffusion equation to update the level set function. A rigorous derivation of the topological derivative for a compliance minimization problem was provided. Although the term representing the effect of boundary conditions when a hole is created in the design domain was ignored in previous research (Yamada et al. 2010), this effect was considered in the method presented here.

We hope that dissemination of this code will enable readers to more easily understand the operation of this topology optimization method and also allow them to compare results obtained when using different topology optimization methods. We believe that this code is very compact, comprehensible, and computationally efficient. It employs the loop vectorization and memory preallocation detailed in Andreassen et al. (2011).

Acknowledgments The authors would like to express their gratitude to Prof. Toshiro Matsumoto and Prof. Hiroshi Isakari for their helpful discussions concerning topological derivatives and encouragement of this work.

Appendix A: Matlab code

```
% Matlab code for topology optimization using a reaction diffusion equation
function [str,phi] = levelset88(nelx,nely,Vmax,tau)
%% Parameter definition
E0 = 1;
Emin = 1e-4;
nu = 0.3;
nvol = 100;
dt = 0.1;
d = -0.02;
p = 4;
phi = ones((nely+1)*(nelx+1),1);
str = ones(nely,nelx);
volInit = sum(str(:))/(nelx*nely);
%% Finite element analysis preparation
% For displacement field
A11 = [12 3 -6 -3; 3 12 3 0; -6 3 12 -3; -3 0 -3 12];
A12 = [-6 -3 0 3; -3 -6 -3 -6; 0 -3 -6 3; 3 -6 3 -6];
B11 = [-4 3 -2 9; 3 -4 -9 4; -2 -9 -4 -3; 9 4 -3 -4];
B12 = [ 2 -3 4 -9; -3 2 9 -2; 4 9 2 3; -9 -2 3 2];
KE = 1/(1-nu^2)/24*([A11 A12;A12' A11]+nu*[B11 B12;B12' B11]);
% For topological derivative
a1 = 3*(1-nu)/(2*(1+nu)*(7-5*nu))*(-(1-14*nu+15*nu^2)*E0)/(1-2*nu)^2;
a2 = 3*(1-nu)/(2*(1+nu)*(7-5*nu))*5*E0;
A = (a1+2*a2)/24*([A11 A12;A12' A11]+(a1/(a1+2*a2))*[B11 B12;B12' B11]);
nodenrs = reshape(1:(1+nelx)*(1+nely),1+nely,1+nelx);
edofVec = reshape(2*nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat = repmat(edofVec,1,8)+repmat([0 1 2*nely+[2 3 0 1] -2 -1],nelx*nely,1);
iK = reshape(kron(edofMat,ones(8,1))',64*nelx*nely,1);
jK = reshape(kron(edofMat,ones(1,8))',64*nelx*nely,1);
% For reaction diffusion equation
NNdif_e = 1/6*[ 4 -1 -2 -1;-1 4 -1 -2;-2 -1 4 -1;-1 -2 -1 4];
NN_e = 1/36*[ 4 2 1 2;2 4 2 1;1 2 4 2;2 1 2 4];
edofVec2= reshape(nodenrs(1:end-1,1:end-1)+1,nelx*nely,1);
edofMat2= repmat(edofVec2,1,4)+repmat([0 nely+1 nely -1],nelx*nely,1);
iN = reshape(kron(edofMat2,ones(4,1))',16*nelx*nely,1);
jN = reshape(kron(edofMat2,ones(1,4))',16*nelx*nely,1);
sNN = reshape(NN_e(:)*ones(1,nely*nelx),16*nelx*nely,1);
NN = sparse(iN,jN,sNN);
sNNdif = reshape(NNdif_e(:)*ones(1,nely*nelx),16*nelx*nely,1);
NNdif = sparse(iN,jN,sNNdif);
%% Loads and boundary settings
F = sparse(2*(nely+1)*(nelx+1),1);
U = zeros(2*(nely+1)*(nelx+1),1);
F((nely+1)*(nelx)*2+nely+2*(-round(nely/32)+1):2:(nely+1)*(nelx)*2+nely+2*(round(nely/32)+1),1) = 1;
fixeddofs = 1:2*(nely+1);
alldofs = 1:2*(nely+1)*(nelx+1);
freedofs = setdiff(alldofs,fixeddofs);
T = NN/dt + tau*(nely*nelx)*NNdif;
fixeddofs_phi = sort([1:nely+1 nely+2:nely+1:(nely+1)*(nelx) 2*(nely+1):nely+1:(nely+1)*(nelx) ...
   (nely+1)*nelx+1:(nely+1)*(nelx+1)]);
phi(fixeddofs_phi) = 0;
alldofs_phi = 1:(nely+1)*(nelx+1);
freedofs_phi = setdiff(alldofs_phi,fixeddofs_phi);
%% Main loop
for iterNum = 1:200
   % FE-analysis, calculate sensitivities
   sK = reshape(KE(:)*(Emin+str(:)'*(E0-Emin)),64*nelx*nely,1);
   K = sparse(iK,jK,sK);
   K = (K+K')/2;
   U(freedofs) = K(freedofs,freedofs) \ F(freedofs);
   SED = (Emin+str*(E0-Emin)).*reshape(sum((U(edofMat)*KE).*U(edofMat),2),nely,nelx);
   TD = (1e-4+str*(1-1e-4)).*reshape(sum((U(edofMat)*A).*U(edofMat),2),nely,nelx);
   td2=[TD(1,1) TD(1,:) TD(1,end); TD(:,1) TD TD(:,end) ; TD(end,1) TD(end,:) TD(end,end)];
   TDN = 0.25*(td2(1:end-1,1:end-1)+td2(2:end,1:end-1)+td2(1:end-1,2:end)+td2(2:end,2:end));
   objective(iterNum) = sum(SED(:));
   vol = sum(str(:))/(nelx*nely);
   % Print results
   disp(['It.: ' num2str(iterNum) ' Compl.: ' sprintf('%10.4e',objective(iterNum)/((nelx*nely)))...
      ' Vol.: '  sprintf('%6.2f' ,vol)])
   colormap(gray); imagesc(-str,[-1,0]); axis equal; axis tight; axis off; drawnow;
   % Check for convergence
   if iterNum>nvol && (abs(vol-Vmax)<0.005) && all(abs(objective(end)- ...
      objective(end-5:end-1))< 0.01*abs(objective(end)))
      return;
   end
```

```
    % Set augmented Lagrangian parameters
    ex = Vmax+(volInit-Vmax)*max(0,1-iterNum/nvol);
    lambda = sum(sum(TDN))/((nely+1)*(nelx+1))*exp(p*((vol-ex)/ex+d));
    C = 1/sum(abs(TDN(:)))*(nely*nelx);
    g2 = reshape(TDN,(nely+1)*(nelx+1),1);
    % Update level set function
    Y = NN*(C*(g2-lambda*ones(size(g2)))+phi/dt);
    phi(freedofs_phi,:) = T(freedofs_phi,freedofs_phi) \ Y(freedofs_phi,:);
    phi = min(1,max(-1,phi));
    phin = reshape(phi,nely+1,nelx+1);
    phie = 0.25*(phin(1:end-1,1:end-1)+phin(2:end,1:end-1)+phin(1:end-1,2:end)+phin(2:end,2:end));
    str(:,:) = (phie(:,:)>0);
end
```

Appendix B: Topological derivative derivation

The boundary value problem for a created hole can be described as

$$div\left(C_{ijkl}\left(u_{k,l}+\delta u_{k,l}\right)\right)=0 \quad \text{in} \quad \Omega\setminus\Omega_\varepsilon \tag{38}$$

$$u_i+\delta u_i \quad =\bar{u}_i \quad \text{on} \quad \Gamma_u \tag{39}$$

$$t_i+\delta t_i \quad =\bar{t}_i \quad \text{on} \quad \Gamma_t \tag{40}$$

$$t_i+\delta t_i \quad =0 \quad \text{on} \quad \Gamma_\epsilon\,. \tag{41}$$

We note that the boundary conditions expressed in (39) and (40) indicate that the displacement and traction after creating a hole must satisfy the original displacement and traction constraints, respectively. (41) represents the free surface on the boundary of the created hole.

The Lagrangian that includes the created hole is given as

$$\begin{aligned}\bar{F}+\delta\bar{F} &= \int_{\Gamma_u\cup\Gamma_t}(t_iu_i+t_i\delta u_i+\delta t_iu_i)d\Gamma \\ &+\int_{\Gamma_u\cup\Gamma_t}\tilde{u}_i(t_i+\delta t_i)d\Gamma+\int_{\Gamma_\varepsilon}\tilde{u}_i(t_i+\delta t_i)d\Gamma \\ &-\int_{\Omega\setminus\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}(u_{k,l}+\delta u_{k,l})d\Omega \\ &+\lambda\left(\int_{\Omega\setminus\Omega_\varepsilon}d\Omega-V_{\max}\right).\end{aligned} \tag{42}$$

When the boundary condition (41) is inserted into (42), the third term on the right-hand side becomes zero. Subtracting the Lagrangian (19) from the Lagrangian (42), the variation $\delta\bar{F}$ of the Lagrangian then becomes

$$\begin{aligned}\delta\bar{F} &= \int_{\Gamma_u\cup\Gamma_t}(t_i\delta u_i+\delta t_iu_i)d\Gamma+\int_{\Gamma_u\cup\Gamma_t}\tilde{u}_i\delta t_i\,d\Gamma \\ &-\int_{\Omega\setminus\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}\delta u_{k,l}\,d\Omega+\int_{\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}u_{k,l}\,d\Omega \\ &-\lambda\int_{\Omega_\varepsilon}d\Omega.\end{aligned} \tag{43}$$

The third term on the right-hand side in the above equation can be replaced using Green's formula, as follows:

$$\begin{aligned}\int_{\Omega\setminus\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}\delta u_{k,l}\,d\Omega &= \int_{\Gamma_u\cup\Gamma_t}\tilde{t}_i\delta u_i\,d\Gamma \\ +\int_{\Gamma_\varepsilon}C_{ijkl}\tilde{u}_{i,j}n_l\delta u_k\,d\Gamma &- \int_{\Omega\setminus\Omega_\varepsilon}div\left(C_{ijkl}\tilde{u}_{k,l}\right)\delta u_i\,d\Omega,\end{aligned} \tag{44}$$

where $\tilde{t}_i$ is the derivative of Lagrange multiplier $\tilde{u}_i$ in the normal direction and n_l is the normal vector at boundary Γ_ϵ. Substituting the above equation into (43) and considering the conditions, $\delta u_i=0$ on Γ_u and $\delta t_i=0$ on Γ_t, the variation of the Lagrangian is then given as

$$\begin{aligned}\delta\bar{F} &= \int_{\Omega\setminus\Omega_\varepsilon}div\left(C_{ijkl}\tilde{u}_{k,l}\right)\delta u_i\,d\Omega+\int_{\Gamma_t}(t_i-\tilde{t}_i)\delta u_i\,d\Gamma \\ &+\int_{\Gamma_u}(u_i+\tilde{u}_i)\delta t_i\,d\Gamma+\int_{\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}u_{k,l}\,d\Omega \\ &-\int_{\Gamma_\varepsilon}C_{ijkl}\tilde{u}_{i,j}n_l\delta u_k\,d\Gamma-\lambda\int_{\Omega_\varepsilon}d\Omega.\end{aligned} \tag{45}$$

The adjoint equation is now defined so that the integrals of the term that includes δu_i or δt_i are canceled out, as follows:

$$div\left(C_{ijkl}\tilde{u}_{k,l}\right)=0 \quad \text{in} \quad \Omega \tag{46}$$

$$\tilde{u}_i=-u_i \quad \text{on} \quad \Gamma_u \tag{47}$$

$$\tilde{t}_i=t_i \quad \text{on} \quad \Gamma_t\,. \tag{48}$$

The variation of the Lagrangian can be obtained as follows, using the adjoint variable obtained by solving the above adjoint field.

$$\begin{aligned}\delta\bar{F} &= \int_{\Omega_\varepsilon}\tilde{u}_{i,j}C_{ijkl}u_{k,l}\,d\Omega-\int_{\Gamma_\varepsilon}C_{ijkl}\tilde{u}_{i,j}n_l\delta u_k\,d\Gamma \\ &-\lambda\int_{\Omega_\varepsilon}d\Omega.\end{aligned} \tag{49}$$

The first term on the right-hand side of the above equation represents state variable u_i and adjoint variable $\tilde{u}_i$, which are known values. On the other hand, the value of δu_i in the second term is unknown. We note that this second term was ignored in previous research (Yamada et al. 2010), so that the method did not consider the effect of boundary condition Γ_ϵ that arises when a hole is created in the design domain.

By subtracting the boundary value problem (15)–(17) from the boundary value problem (38)-(41), we obtain the following boundary value problem by which the unknown value δu is governed.

$$div\left(C_{ijkl}\delta u_{k,l}\right)=0 \quad \text{in} \quad \Omega\setminus\Omega_\varepsilon \tag{50}$$
$$\delta u_i=0 \quad \text{on} \quad \Gamma_u \tag{51}$$
$$\delta t_i=0 \quad \text{on} \quad \Gamma_t \tag{52}$$
$$\delta t_i=-t_i \quad \text{on} \quad \Gamma_\epsilon\,. \tag{53}$$

In the above problem, since the radius of the hole ϵ is sufficiently small, the effect of boundaries Γ_u and Γ_t can be ignored for solving δu_i on Γ_ϵ. δu_i is then a solution of the following boundary value problem.

$$div\left(C_{ijkl}\delta u_{k,l}\right)=0 \quad \text{in} \quad \Omega\setminus\Omega_\varepsilon \tag{54}$$
$$\delta t_i=-\sigma^0_{ij}n_j+O(\epsilon) \quad \text{on} \quad \Gamma_\epsilon\,, \tag{55}$$

where the superscript 0 indicates the value without creating holes, and $\sigma_{ij}=C_{ijkl}u_{k,l}$. The solution of this boundary value problem in sphere Γ_ϵ is given (Guzina and Bonnet 2004; Lurie and Belyaev 2005) as:

$$\delta u_i=-\frac{\varepsilon}{\mu}\left(\frac{4-5\mu}{7-5\mu}\sigma^0_{ij}n_j-\frac{3-5\mu}{4(7-5\mu)}\sigma^0_{jj}n_j+O(\varepsilon)\right). \tag{56}$$

The variation of Lagrangian $\bar F$ is then given by:

$$\delta\bar F=\frac{4\pi\varepsilon^3}{3}\left\{\frac{3(1-\nu)}{2(1+\nu)(7-5\nu)}\left[\frac{-(1-14\nu+15\nu^2)E}{(1-2\nu)^2}\delta_{ij}\delta_{kl}+5E(\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk})\right]\tilde u^0_{i,j}u^0_{k,l}-\lambda\right\}, \tag{57}$$

where δ_{ij} is Kronecker's delta function. Finally, the topological derivative of the Lagrangian is given as

$$d_t\bar F=\lim_{\epsilon\to 0}\frac{\delta\bar F}{\frac{4\pi\epsilon^3}{3}}=\tilde u^0_{i,j}A_{ijkl}u^0_{k,l}-\lambda\,, \tag{58}$$

where A_{ijkl} is defined as follows:

$$A_{ijkl}=\frac{3(1-\nu)}{2(1+\nu)(7-5\nu)}\left[\frac{-(1-14\nu+15\nu^2)E}{(1-2\nu)^2}\delta_{ij}\delta_{kl}+5E(\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk})\right]. \tag{59}$$

Appendix C: Topological derivative implementation

First, for simplicity, we reformulate the tensor A_{ijkl} defined in (22) as follows:

$$A_{ijkl}:=A_1\delta_{ij}\delta_{kl}+A_2(\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk})\,. \tag{60}$$

Namely,

$$A_1=-\frac{3(1-\nu)\left(1-14\nu+15\nu^2\right)}{2(1+\nu)(7-5\nu)(1-2\nu)^2}E,\; A_2=\frac{15E(1-\nu)}{2(1+\nu)(7-5\nu)}\,. \tag{61}$$

The first term on the right-hand side in (21) can then be given as follows:

$$\begin{aligned}\tilde u_{i,j}A_{ijkl}u_{k,l} &= \tilde u_{i,j}\left\{A_1\delta_{ij}\delta_{kl}+A_2(\delta_{ik}\delta_{jl}+\delta_{il}\delta_{jk})\right\}u_{k,l} \qquad (62)\\ &= A_1(\tilde u_{1,1}+\tilde u_{2,2}+\tilde u_{3,3})(u_{1,1}+u_{2,2}+u_{3,3})\\ &\quad +A_2(\tilde u_{1,1}u_{1,1}+\tilde u_{1,2}u_{1,2}+\tilde u_{1,3}u_{1,3}\\ &\quad +\tilde u_{2,1}u_{2,1}+\tilde u_{2,2}u_{2,2}+\tilde u_{2,3}u_{2,3}+\tilde u_{3,1}u_{3,1}\\ &\quad +\tilde u_{3,2}u_{3,2}+\tilde u_{3,3}u_{3,3})\\ &\quad +A_2(\tilde u_{1,1}u_{1,1}+\tilde u_{1,2}u_{2,1}+\tilde u_{1,3}u_{3,1}\\ &\quad +\tilde u_{2,1}u_{1,2}+\tilde u_{2,2}u_{2,2}+\tilde u_{2,3}u_{3,2}+\tilde u_{3,1}u_{1,3}\\ &\quad +\tilde u_{3,2}u_{2,3}+\tilde u_{3,3}u_{3,3}) \qquad (63)\\ &= (A_1+2A_2)\tilde u_{1,1}u_{1,1}+(A_1+2A_2)\tilde u_{2,2}u_{2,2}\\ &\quad +(A_1+2A_2)\tilde u_{3,3}u_{3,3}\\ &\quad +A_1(\tilde u_{1,1}u_{2,2}+\tilde u_{2,2}u_{1,1}\\ &\quad +\tilde u_{2,2}u_{3,3}+\tilde u_{3,3}u_{2,2}+\tilde u_{3,3}u_{1,1}\\ &\quad +\tilde u_{1,1}u_{3,3})\\ &\quad +A_2[(\tilde u_{1,2}+\tilde u_{2,1})(u_{1,2}+u_{2,1})\\ &\quad +(\tilde u_{2,3}+\tilde u_{3,2})(u_{2,3}+u_{3,2})\\ &\quad +(\tilde u_{3,1}+\tilde u_{1,3})(u_{3,1}+u_{1,3})]. \qquad (64)\end{aligned}$$

In the above formulation, we use $u^0_{i,j}$ instead of $u_{i,j}$ for simplicity. Now, using the following definition of strains, $\epsilon_{ii}=u_{i,i}$, and $\tau_{ij}=u_{i,j}+u_{j,i}$, the above equation is transformed as follows:

$$\begin{aligned}\tilde u_{i,j}A_{ijkl}u_{k,l} &= (A_1+2A_2)\tilde\epsilon_{11}\epsilon_{11}+(A_1+2A_2)\tilde\epsilon_{22}\epsilon_{22}+(A_1+2A_2)\tilde\epsilon_{33}\epsilon_{33}\\ &\quad +A_1(\tilde\epsilon_{11}\epsilon_{22}+\tilde\epsilon_{22}\epsilon_{11}+\tilde\epsilon_{22}\epsilon_{33}+\tilde\epsilon_{33}\epsilon_{22}+\tilde\epsilon_{33}\epsilon_{11}+\tilde\epsilon_{11}\epsilon_{33})\\ &\quad +A_2(\tilde\tau_{12}\tau_{12}+\tilde\tau_{23}\tau_{23}+\tilde\tau_{31}\tau_{31}) \qquad (65)\\ &= \begin{bmatrix}\tilde\epsilon_{11} & \tilde\epsilon_{22} & \tilde\epsilon_{33} & \tilde\tau_{12} & \tilde\tau_{23} & \tilde\tau_{31}\end{bmatrix}\\ &\quad\times\begin{bmatrix}A_1+2A_2 & A_1 & A_1 & 0 & 0 & 0\\ A_1 & A_1+2A_2 & A_1 & 0 & 0 & 0\\ A_1 & A_1 & A_1+2A_2 & 0 & 0 & 0\\ 0 & 0 & 0 & A_2 & 0 & 0\\ 0 & 0 & 0 & 0 & A_2 & 0\\ 0 & 0 & 0 & 0 & 0 & A_2\end{bmatrix}\begin{bmatrix}\epsilon_{11}\\ \epsilon_{22}\\ \epsilon_{33}\\ \tau_{12}\\ \tau_{23}\\ \tau_{31}\end{bmatrix}.\end{aligned} \tag{66}$$

Similarly, for the plane stress problem, we have

$$\begin{aligned}\tilde u_{i,j}A_{ijkl}u_{k,l} &= \begin{bmatrix}\tilde\epsilon_{11}\tilde\epsilon_{22} & \tilde\tau_{12}\end{bmatrix}\begin{bmatrix}A_1+2A_2 & A_1 & 0\\ A_1 & A_1+2A_2 & 0\\ 0 & 0 & A_2\;0\end{bmatrix}\begin{bmatrix}\epsilon_{11}\\ \epsilon_{22}\\ \tau_{12}\end{bmatrix}\\ &= (A_1+2A_2)\begin{bmatrix}\tilde\epsilon_{11} & \tilde\epsilon_{22} & \tilde\tau_{12}\end{bmatrix}\begin{bmatrix}1 & c & 0\\ c & 1 & 0\\ 0 & 0 & \frac{1-c}{2}\end{bmatrix}\begin{bmatrix}\epsilon_{11}\\ \epsilon_{22}\\ \tau_{12}\end{bmatrix},\end{aligned} \tag{67}$$

where $c=\frac{A_1}{A_1+2A_2}$. Compare this with the following formulation of the mutual strain energy density:

$$\tilde u_{i,j}E_{ijkl}u_{k,l}=\frac{E}{(1-\nu)^2}\begin{bmatrix}\tilde\epsilon_{11} & \tilde\epsilon_{22} & \tilde\tau_{12}\end{bmatrix}\begin{bmatrix}1 & \nu & 0\\ \nu & 1 & 0\\ 0 & 0 & \frac{1-\nu}{2}\end{bmatrix}\begin{bmatrix}\epsilon_{11}\\ \epsilon_{22}\\ \tau_{12}\end{bmatrix}. \tag{68}$$

The topological derivative can be therefore computed by substituting A_1, A_2 into E, ν in the procedure for computing the mutual strain energy density, as follows:

$$E \to (A_1 + 2A_2)(1-c)^2, \ \nu \to c. \tag{69}$$

References

Allaire G (2002) Shape optimization by the homogenization method. Vol. 146 of applied mathematical sciences. Springer, New York

Allaire G, Pantz O (2006) Structural optimization with FreeFem++. Struct Multidiscipl Optim 32:173–181

Allaire G, Jouve F, Toader AM (2002) A level-set method for shape optimization. C R Acad Sci Paris Ser I 334:1125–1130

Allaire G, Jouve F, Toader AM (2004) Structural optimization using sensitivity analysis and a level-set method. J Comp Phys 194:363–393

Allaire G, de Gournay F, Jouve F, Toader AM (2005) Structural optimization using topological and shape sensitivity via a level set method. Control Cybern 34:59–80

Andreassen E, Clausen A, Schevenels M, Lazarov BS, Sigmund O (2011) Efficient topology optimization in MATLAB using 88 lines of code. Struct Multidiscipl Optim 43:1–16

Bendsøe MP, Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method. Comput Methods Appl Mech Eng 71:197–224

Bendsøe MP (1989) Optimal shape design as a material distribution problem. Struct Multidiscipl Optim 1:193–202

Challis VJ (2010) A discrete level-set topology optimization code written in Matlab. Struct Multidisc Optim 41:453–464

Eschenauer HA, Kobelev VV, Schumacher A (1994) Bubble method for topology and shape optimization of structures. Struct Multidisc Optim 8:42–51

Howell HH (2001) Compliant mechanisms. Wiley

Guzina BB, Bonnet M (2004) Topological derivative for the inverse scattering of elastic waves. Q J Mech Appl Math 57(2):161–179

Lurie AI, Belyaev A (2005) Theory of elasticity. Springer, New York

Nishiwaki S, Frecker MI, Min S, Kikuchi N (1998) Topology optimization of compliant mechanisms using the homogenization method. Int J Numer Methods Eng 535–559

Osher SJ, Sethian JA (1988) Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. J Comput Phys 79:12–49

Sigmund O (2001) A 99 line topology optimization code written in Matlab. Struct Multidisc Optim 21:120–127

Wang MY, Wang X, Guo D (2003) A level set method for structural topology optimization. Comput Methods Appl Mech Eng 192:227–246

Yamada T, Izui K, Nishiwaki S, Takezawa A (2010) A topology optimization method based on the level set method incorporating a fictitious interface energy. Comput Methods Appl Mech Eng 199:2876–2891