ACHIEVING MINIMUM LENGTH SCALE AND DESIGN CONSTRAINTS IN TOPOLOGY OPTIMIZATION: A NEW APPROACH

BY

CHAU HOAI LE

B.E., Hanoi National Institute of Civil Engineering, 1997

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Civil and Environmental Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2006

Urbana, Illinois

Acknowledgements

I would like to express my deep gratitude to my advisor Professor G.H. Paulino, who has provided great support, guidance and encouragement during my study. Especially, he has passed much energy and motivation for me to overcome challenges in the work. His comments and suggestions helped improve to a great extent the content of this thesis.

I would like to thank all of my colleagues for their help during my study. It is a great pleasure for me to work with my research group members, S.H. Song, K. Park, Z. Zhang, A. Sutradhar, S. Wang, B. Shen, E. Dave, H.Y. Yin, M. Gonzalez, T. Nguyen, and C. Talischi. Especially, C. Talischi has proof read the thesis. I will treasure the time we shared together. I am grateful for the help of professors and staff in the Civil and Environmental Engineering Department at the University of Illinois. They have created a professional and warm environment.

I gratefully acknowledge the financial support from the Vietnam Education Foundation through the VEF fellowship. I also gratefully acknowledge the financial support from the Center for Process Simulation and Design (CPSD) at the University of Illinois via grant NSF DMR 01-21695.

Finally, I am indebted to my wife Giang, and my kids Binh and Minh, for their encouragement and sacrifice for my graduate study.

Abstract

The topology optimization problem is generally an ill-posed problem, where the solution is not unique. Because of that, implementation of topology optimization faces complications such as mesh-dependency of the solution, and numerical instabilities. Also, the results obtained by topology optimization are not usually fabrication friendly due to the fine and arbitrary patterns. An effective method that addresses both above mentioned issues consists of imposing a minimum length scale to the resulting structural members. Several approaches to achieve minimum length scale in topology optimization have been proposed in the literature. However, while some problems are solved, others are created or remain unsolved; and the search for better methods continues. In this thesis, we review several prominent approaches and propose a new approach to achieve minimum length scale. We discuss the potential of the new approach for obtaining other fabrication and design constraints, in addition to the minimum length scale constraint. Thus the new approach has the potential of improving the quality of topology optimization in various engineering applications where design constraints must be placed.

Table of Contents

List of Figures

Chapter 1 : Introduction

Topology optimization is a relatively new branch of structural optimization. Unlike shape and size optimization, topology optimization parameterizes structures using material points in a way similar to defining an image by pixels. In a continuum setting, the feasible design space includes all configurations (i.e. shape, size and connectivity) in a given domain [1, 2].

In this work, we propose a general method for restricting directly the topological design space and eliminating undesirable patterns from the solutions. The task is achieved by introducing additional layers of design variables. The design space is constructed based on design variables using new projection schemes.

1.1 Background and literature review

The desired result of topology optimization is the domain containing either material regions or void regions. However, it is mathematically difficult to work with integer variables, thus relaxation is usually applied. By allowing intermediate material density between 0 and 1, objective and constraint functions become continuous and differentiable. To obtain integer final solution, intermediate density is penalized (with the same volume, it leads to reduced stiffness for intermediate material). At the early stage, the homogenization method [3, 4, 5] was used to derive the stiffness of intermediate density based on certain configurations of microstructures. However, final solutions are not supposed to contain porous microstructures. Thus, the derivation of the stiffness

of intermediate material density based on the homogenization approach only serves as a mean of penalization. Later on, the Solid Isotropic Material with Penalization (SIMP) approach was proposed [6, 7] as a more straightforward way to interpolate the stiffness of intermediate density. The SIMP material model was originally thought of as a fictitious material model but, later on, it was proved [8] that there exist microstructures corresponding to the stiffness derived by the SIMP model if the penalization parameter is sufficiently large.

It is generally known that continuum topology optimization is an ill-posed problem, for which a unique solution does not exist. Finer patterns of the topology usually yield more optimized solution, thus the solution tends to be in the form of microstructures. When finite element discretization is used for the topology and state variable, the ill-posedness of the problem manifests in the form of mesh-dependency of the solution. Although a unique topology may be obtained for each specific discretization, qualitatively different topologies are obtained for different mesh resolutions. In order to achieve uniqueness of the solution, the design space must be limited by imposing additional restrictions on the final topology. Checkerboard like numerical instability [9, 10] is under the same category. A significant amount of literature has been dedicated to solving this problem and numerous methods have been proposed.

The methods to solve the aforementioned complications of topology optimization fall into three categories. The first category applies filters. The second category reduces the design space directly. The third category imposes additional constraints to the optimization problem. Existing approaches include filters [17] of density [18,] or sensitivities [20, 10, 21]; restriction of the design space using wavelet [24, 25]; The use of implicit functions and regularization [26], or nodal design variables and projection functions [27] to restrict design space; the perimeter control implemented by Haber *et al.* [11] (based on the theory by Ambrosio and Buttazzo [12]), and further studied by Peterson [13]; gradient constraints, either global or local [14, 15, 16]; monotonicity based length scale [22, 23].

By choosing suitable design variables and projection schemes, we may restrict the design space as desired. Based on this concept, a new method is proposed herein. It is applied to solve two important problems of topology optimization. One is the problem related to checkerboard problem, non-uniqueness of the 0-1 solution, mesh dependency of the solution and obtaining minimum length scale. The other is the problem of imposing manufacturing constraints so that the solution is meaningful and economical for fabrication. In this work, we will cover the first application in detail and give an introduction to the second application.

1.2 Elements for topology optimization

In general, one may use two interpolations in a topology optimization problem. One is used to approximate the displacement field, and the other is used to approximate the material density field. Thus, appropriate notation for elements for topology optimization is needed [33]. The quadrilateral element (Q4) implemented with the element based approach is denoted as Q4/U, where the Q4 is the element used to approximate the displacement field and U refers to uniform material density inside each element. The Q4 element used with the CAMD approach is denoted as Q4/Q4, meaning Q4 element is used for both the displacement field and the material density field. Similarly, Q8/U, Q8/Q4, and so on are possible elements in 2D. In 3D, similar notation can be used. For example, B8/U denotes the brick element implemented with the element based approach; B8/B8 element denotes the brick element implemented with the CAMD approach where B8 is used to approximate both the displacement field and material density field. Figure 1 shows examples of elements for topology optimization.

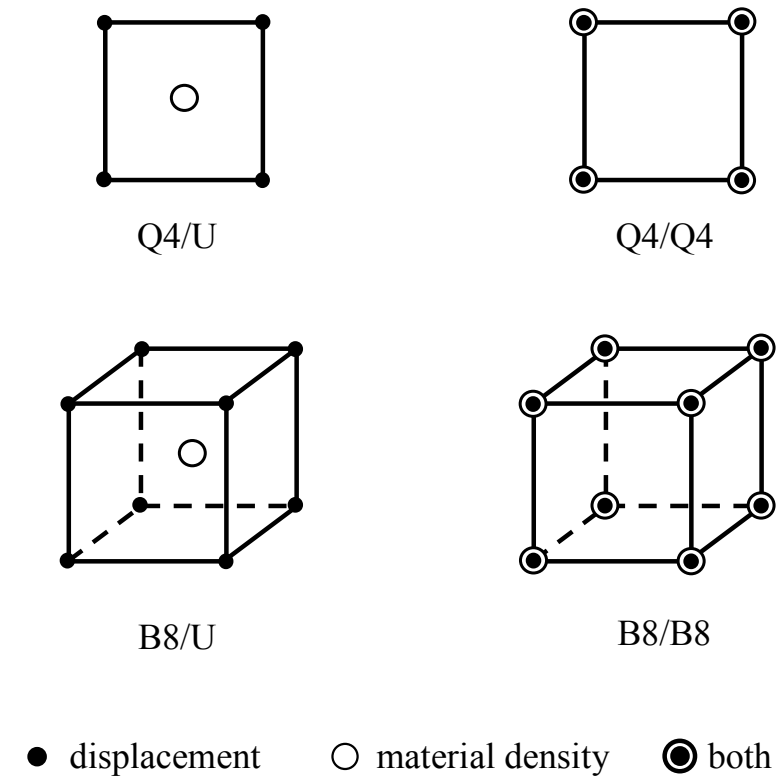


Figure 1 : Examples of elements for topology optimization.

1.3 Element-based problem description

The problem to be addressed is stated in the finite element discretization setting, e.g. using elements such as Q4/U or B8/U (Figure 1). The objective is to minimize the mean compliance, or in other words, to maximize the stiffness of the structure. The constraint is the total volume of material used in the structure. The material density in each element is assumed to be constant and is represented by one design variable (ρ_e). This approach is called the element-based approach. For the relaxed problem, the SIMP [3, 8, 7, 29] model is assumed, where the stiffness of intermediate density is proportional to the density raised to a penalization power p. The problem statement is as follows:

$$
\begin{aligned}
\min_{\rho_e} \quad & \theta(\mathbf{u}, \rho_e) \\
\text{s.t.} \quad & \mathbf{K}\left(\mathbf{E}_e\right)\mathbf{u} = \mathbf{f} \\
& \mathbf{E}_e = \rho_e^p \mathbf{E}_o \\
& \sum_{e=1}^{n} \rho_e v_e \le V_o \\
& 0 \le \rho_{\min} \le \rho_e \le 1 \qquad e=1..n
\end{aligned}
\tag{1}
$$

where θ is the objective function to be minimized; n is the number of elements in the domain; $\mathbf{E}_e$ is the elastic tensor corresponding to intermediate densities; $\mathbf{E}_o$ is the elastic tensor of the base material (i.e. with density equal to 1); $\mathbf{K}$ is the stiffness matrix which is a function of the material properties, i.e. $\mathbf{K} \equiv \mathbf{K}\left(\mathbf{E}_e\right)$; V_o is the upper bound constraint on the total volume of the structure; and $\rho_{\min}$ is a small value to avoid singularity of the stiffness matrix.

The objective function θ should be dependent on displacement, otherwise the problem becomes trivial and needs no finite element analysis (FEA). In case of the minimum compliance problem, the objective function is calculated as follows:

$$
\theta(\mathbf{u}, \rho) = \mathbf{u}^T \mathbf{K} \mathbf{u} = \sum_{e=1}^{n} \left(\rho_e\right)^p \mathbf{u}_e^T \mathbf{K}_o \mathbf{u}_e \tag{2}
$$

Instead of the SIMP model, an alternative is to use the homogenization approach [4, 5] with certain microstructure patterns to interpolate the stiffness of intermediate density. As intermediate densities are not preferred in the final solution, homogenization only serves as a mean to relax the integer problem.

1.4 Continuous approximation of material distribution (CAMD)

So far, the most common formulation for topology optimization is the so-called element-based approach, where the material density field is parameterized by giving one material density value to each finite element. However, material

density is naturally a field where each point in the domain can assume a value. There have been attempts to solve topology optimization problems by considering the material density as a field and solving for displacement field and material density field simultaneously [32]. Matsui and Terada [28], and Rahmatalla and Swan [33] proposed to approximate the material density field the same way we do for the displacement field in finite element analysis. Material density field inside of a finite element is interpolated using nodal values and shape functions. The determination of element matrices follows the graded finite element formulation, which can be found in the work by Kim and Paulino [34]. The approach is called the Continuous Approximation of Material Distribution (CAMD) approach.

According to the CAMD approach, the topology optimization problem is parameterized using material densities at nodes. During the optimization process, nodal densities are updated and optimized. The topology of the final structure is represented by the material density field, which is interpolated using nodal densities and shape functions.

Matsui and Terada [28], and Rahmatalla and Swan [33] used the standard Q4 element and bilinear shape functions for the material density field (i.e. the Q4/Q4 element of Figure 1.) Higher order interpolation for the displacement field usually yields more accurate and well defined results. However, it is not recommended to use higher order interpolation for the material density field to avoid spurious material density values, such as negative values or values greater than 1.

1.4.1 Problem statement

In this section, the topology problem statement is described in a finite element setting. We seek to minimize an objective function θ which is typically dependent on displacement field u and material density. The objective function

should be dependent at least on the displacement field, otherwise the problem becomes trivial and no iteration process is required. Each optimization step will update the design variables, which is the material density at nodes ρ_n such that the objective function is improved. Material density inside each finite element is interpolated from nodal densities of nodes (e.g. 4 nodes for the Q4 element) using bilinear shape functions. Elasticity tensor of the intermediate density is interpolated using a material interpolation model, such as the Solid Isotropic Material with Penalization (SIMP), using Hashin-Strikman bounds [35, 36], or using homogenization of certain microstructure pattern [4, 5]. The displacement field must satisfy the equilibrium condition. In addition, the solution must satisfy constraints, such as the bounds on material density $(0 \le \theta \le 1)$ or limit on total volume fraction $V \le V_o$.

The following CAMD problem statement adopts the SIMP model:

$$\begin{aligned} \min_{\boldsymbol{\Psi}} \quad & \theta(\mathbf{u}, \rho) \\ \text{s.t.} \quad & \rho = \sum_{i=1}^{N} N_i \Psi_i \\ & \mathbf{E} = \rho^p \mathbf{E}_o \\ & \mathbf{K}(\mathbf{E})\mathbf{u} = \mathbf{f} \\ & g_j(\mathbf{u}, \rho) \le 0 \quad j = 1..m \end{aligned} \tag{3}$$

where θ is the objective function to be minimized; Ψ_i are densities at node i; ρ is the material density field (inside elements); N is the number of nodes of each element; N_i is the shape function associated with node i; $\mathbf{E}$ is the elasticity tensor of materials with intermediate density; $\mathbf{E}_o$ is the elasticity tensor of solid material (with $\rho = 1$); $\mathbf{K} \equiv \mathbf{K}(\mathbf{E})$ is the stiffness matrix; and m is number of general constraints.

According to the graded finite element formulation by Kim and Paulino [34], the element stiffness matrices are obtained using Gauss quadrature. Material densities are evaluated at Gauss points by interpolation using shape functions. In addition to the standard topology optimization steps, the implementation of the

CAMD approach includes calculation of material density at Gauss points based on nodal values and shape functions; and calculation of sensitivities with respect to nodal densities based on sensitivities with respect to densities at Gauss points.

1.4.2 Sensitivities with respect to nodal densities

Sensitivities of a function f, which is either an objective function or a constraint function, with respect to nodal densities is calculated by the chain-rule,

$$\frac{\partial f}{\partial \rho_i} = \int_{\Omega} \frac{\partial f}{\partial \rho} \frac{\partial \rho}{\partial \rho_i} d\Omega \tag{4}$$

where $\partial f / \partial \rho$ at a point is defined as the gradient of f with respect to change in ρ in a unit domain (e.g. unit area for 2D and unit volume for 3D) at that point. The sensitivity $\partial f / \partial \rho$ is calculated using a traditional method, which is usually adjoint method [37]. For a point inside a finite element having node i, $\partial \rho / \partial \rho_i$ is given as follows.

$$\frac{\partial \rho}{\partial \rho_i} = N_i \tag{5}$$

Thus, the finite element form of the sensitivities expression is

$$\frac{\partial f}{\partial \rho_i} = \sum_{S_i} \int_{\Omega_e} \frac{\partial f}{\partial \rho} N_i^e d\Omega_e \tag{6}$$

where S_i is the set of all elements sharing node i; Ω_e is the sub-domain inside element e; and N_i^e is the shape function corresponding to node i of element e. Usually, the expression for sensitivities with respect to nodal densities is evaluated using Gauss quadrature.

1.4.3 Numerical instabilities with CAMD approach

Rahmatalla and Swan [33] have reported "islanding" and "layering" phenomena with the implementation of CAMD. This type of numerical instabilities resembles the well-known checkerboard problem in the element-based implementation. The reason for these patterns to appear in final topological results is that these patterns overestimate the stiffness making them more advantageous than the homogeneous distribution with the same material density when penalization $q>1$ is applied. In other words, they avoid the penalization to some extent. Figure 2 shows the three patterns and Figure 3 provides quantitative comparison of the stiffness of those patterns together with the homogeneous distribution. The homogenized stiffness is calculated as

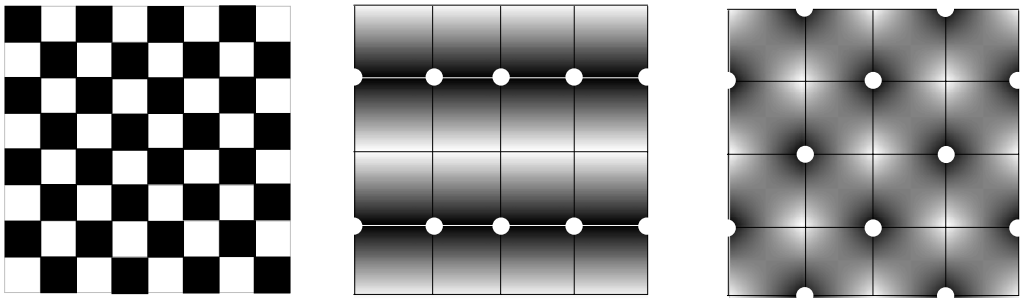
$$C_{ijkl}^{H}=\frac{1}{Y}\int_{Y}\left(C_{ijkl}-C_{ijpq}\frac{\partial\chi_{p}^{kl}}{\partial y_{q}}\right)dY \tag{7}$$

where C_{ijkl}^{H} are stiffness coefficients at a point in the domain; C_{ijkl}^{H} are homogenized stiffness coefficients; Y is the periodic domain; $\chi_{1}^{kl},\chi_{2}^{kl}$ are characteristic displacements obtained by solving the following microscopic equation [5]

$$\int_{Y}C_{ijpq}\frac{\partial\chi_{p}^{kl}}{\partial y_{q}}\frac{\partial v_{i}}{\partial y_{j}}dY=\int_{Y}C_{ijkl}\frac{\partial\chi_{p}^{kl}}{\partial y_{q}}\frac{\partial v_{i}}{\partial y_{j}}dY \quad \text{for all} \quad v=(v_{1},v_{2})\in V_{h} \tag{8}$$

where V_{h} is the set of all Y-periodic functions. Due to the geometry of the patterns and the use of the Q4 element for discretization, the characteristic displacements vanish. The expression for the stiffness coefficients is reduced as follows:

$$C_{ijkl}^{H}=\frac{1}{Y}\int_{Y}C_{ijkl}dY \tag{9}$$



(a) Checkerboard (b) Layering (c) Islanding

Figure 2 : Checkerboard, "layering", and "islanding" patterns in 2D.

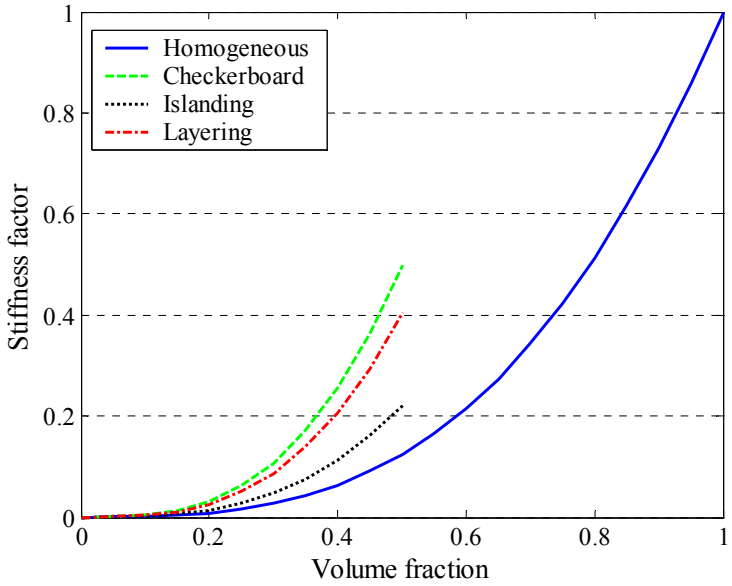


Figure 3 : Homogenized stiffness of different checkerboard-like patterns with intermediate volume fraction.

For the checkerboard pattern in Figure 2 (a), the homogenized stiffness coefficients are

$$C_{ijkl}^{H} = \frac{1}{2}(2\rho)^{p} C_{ijkl}^{o} \text{ with } \rho = [0, 0.5] \tag{10}$$

where C_{ijkl}^{o} are the stiffness coefficients of the solid material ($\rho = 1$); p is the penalization parameter. For the "layering" and "islanding" patterns given in

Figure 2 (b) and (c), Gauss quadrature can be used to evaluate the integrals as follows:

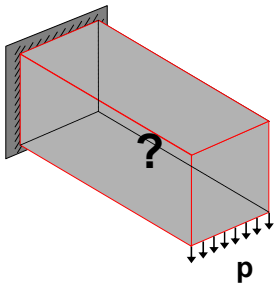
$$C_{ijkl}^{H} = C_{ijkl}^{o} \sum_{g=1}^{4} \left[\sum_{n=1}^{4} \rho_n N_n \left(y_{1g}, y_{2g} \right) \right]^{p} \tag{11}$$

where $\rho_n = 2\rho$ at half of the nodes and $\rho_n = 0$ at the others; $\rho = [0, 0.5]$ as in the case of checkerboard pattern. The factor of the homogenized stiffness coefficients to the original ones are illustrated in Figure 3. As shown, the checkerboard patterns over estimate the stiffness the most. "Layering" and "islanding" patterns also overestimate the stiffness but to a smaller extent. That explains why CAMD approach is less susceptible to numerical instabilities.

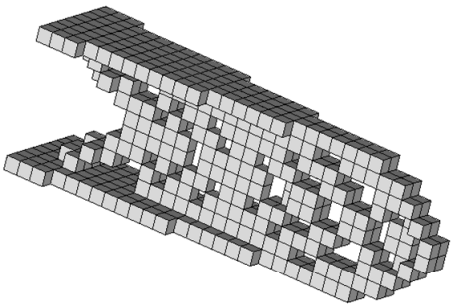
The implementation of the CAMD approach without filtering technique in 3D also shows isolated layers, bars and islands. These patterns resemble the "layering" and "islanding" patterns in 2D. Figure 4 shows the results obtained with the element based approach and CAMD approach without using filtering technique. Although the patterns of checkerboard, "layering" and "islanding" look different, they are all numerical instabilities [10] caused by the error in the finite element discretization.

1.5 Thesis Organization

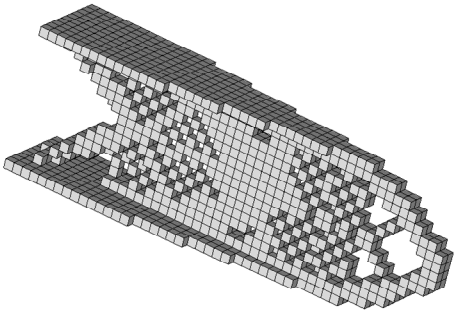
We present a new scheme to achieve minimum length scale to the topology. This projection scheme does not involve any restriction on the gradient of material density and does not produce the fading effect that is seen with the filtering technique and nodal projection approach proposed by Guest *et al.* [27]. The new scheme is different from the nodal projection approach [27] in the projection functions from the nodal design variables space onto element densities space. The new scheme does not cause gray areas around structural members. We also propose the application of the new scheme on top of other approaches such as the element-based, CAMD [28], and even the nodal projection approach.



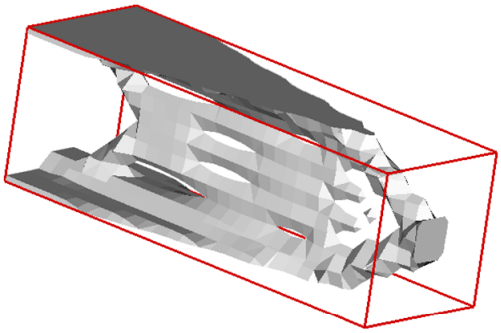
(a) Domain configuration



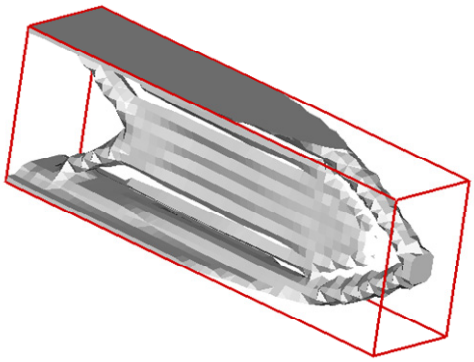
(b) 30x10x10 B8/U elements



(c) 60x20x20 B8/U elements



(d) 30x10x10 B8/B8 elements



(e) 60x20x20 B8/B8 elements

Figure 4 : Numerical instabilities (checkerboard-like) in 3D for element based and CAMD approaches.

We introduce additional layers of variables on top of the existing layer of element (or nodal) densities of each approach. The new projection scheme is employed for the projection of the design variable layer onto the element (or nodal) density layer. The projection of the second "layer" of nodal variables onto the element density depends on the original projection function of each approach. With the CAMD approach, for example, the latter projection is the interpolation of material density based on the nodal values using shape functions. It provides smoothening effect to enhance convergence toward global optima. The first projection imposes the minimum length scale.

The thesis is organized as follows. First we introduce the topology optimization problem and review the most prominent existing approaches to solving complications of topology optimization. Then we present in detail the new scheme to achieve minimum length scale. The proposed scheme naturally imposes a minimum length scale and solves the ill-posedness of the topology optimization problem (i.e. the non-uniqueness of the solution in the continuum setting and mesh-dependency problem in the discrete setting) without adding any constraints to the original optimization problem or using any filtering technique. Afterwards, we discuss the generalization of the proposed method and an extension to achieve design constraints with the design of stiffened plates. Finally, conclusions are inferred and suggestions for future work are provided.

Chapter 2 : On existing methods to address minimum length scale

Various methods have been proposed in the literature to address the aforementioned complications of topology optimization. As discussed in Chapter 1, the methods fall into three categories.

- Filtering (of densities or sensitivities)
- Reduction of the design space
- Imposition of additional constraints

In this chapter, we review four methods which are representative of the three categories. Those methods are: filter of sensitivities (first category); the nodal projection (second category); perimeter constraint (third category); and monotonicity based length scale (third category). Then we discuss other approaches and some key points for successful methods of topology optimization.

2.1 Filtering technique

Filtering technique includes both filtering of densities and filtering of sensitivities. Filter usually prevents sharp change of material densities from 0 to 1, therefore it prevents thin structural members. By specifying the filter radius, which is independent of element size, the problem of mesh-dependency of the solutions is solved. Also, checkerboard problem is eliminated by means of filtering because checkerboard patterns are patterns with the thinnest structural members. Many filters have been proposed in the literature [10, 17, 18, 19, 20, 21], however, the most used one so far is the filter of sensitivities by Sigmund [10, 20, 21].

The filter of sensitivities is applied to modify the actual sensitivities in each optimization step. The modified sensitivities are then used in gradient based optimization update. The flowchart illustrating the optimization process and the application of the filter is provided in Figure 5.

The modification of the sensitivities follows:

$$\frac{\widehat{\partial\theta}}{\partial\rho_e} = \frac{1}{\rho_e \sum_{f=1}^{N} w_f} \sum_{f=1}^{N} w_f \rho_f \frac{\partial\theta}{\partial\rho_f} \tag{12}$$

where w_f is the weight factor defined by

$$w_f = \begin{cases} r_{\min} - r(e,f) & \text{if element } f \text{ lies within the filter circle centered at "}e\text{"} \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

in which $r(e,f)$ is the distance between the center of gravity of element "e" and element "f". Moreover, f denotes elements which fall within the filter circle centered at element "e" with radius $r_{\min}$, and f is defined as

$$\left\{ f \in N \mid r(e,f) \le r_{\min} \right\} \tag{14}$$

Figure 6 shows the filter circle and related entities.

The filter of sensitivities does not work exactly as the common low-pass filter used in image processing. It favors low density elements around high density ones instead of merely smoothening the sensitivities distribution. Figure 7 shows the densities distribution, the original sensitivities and the filtered sensitivities.

The disadvantage of the filter is that the results obtained usually contain elements of intermediate densities or gray areas around structural members, as depicted by Figure 7(a). These gray elements are not preferred in case a 0-1 structure is sought.

Another disadvantage of the filter of sensitivities is that the solution obtained does not satisfy the optimality criteria. However, the solution is practically close enough to the optimum solution in the engineering sense. The likely reason for the filter of sensitivities to be so effective and favorable is that it enhances the convergence to global optima, especially for minimum compliance problems,

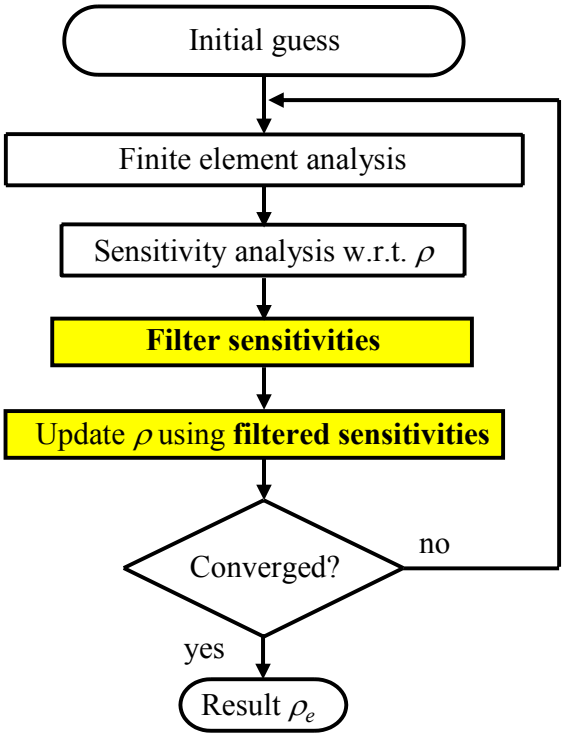


Figure 5 : Flowchart of topology optimization using filter of sensitivities.

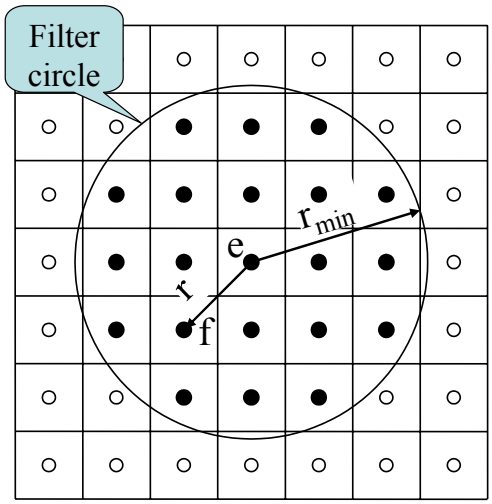


Figure 6 : The filter circle.

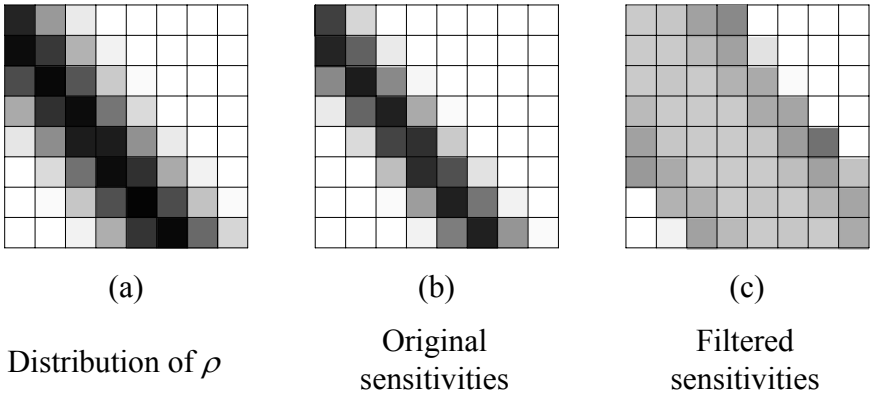


Figure 7 : Effect of the sensitivities filter. The above illustrations were obtained from a computer simulation.

besides the effects common to all other filters. With the filter of sensitivities, structural members, after they are formed, can easily change their locations during the optimization process. In other words, structural members are not trapped in local minima after they are formed. We will not go further into this discussion because the global convergence itself is a major topic of research and is not the main focus of this thesis.

2.2 Nodal design variables and projection functions

Guest *et al.* [27] proposed an approach which implicitly prevents sharp change of material density from 0 to 1. The approach ensures that the change of density from 0 to 1 occur over a minimum length. Gray areas around structural members can not be narrower than a minimum length, which is specified by the user. Thus, checkerboard patterns are eliminated and structural members are thought of as having minimum length scale by converting gray areas into equivalent solid ones.

The essence of the approach lies in the introduction of design variables located at nodes and projection functions to obtain element densities from nodal variables. Density of an element is calculated as the weighted average of the nodal design variables in its neighborhood, which are determined by the minimum length scale parameter.

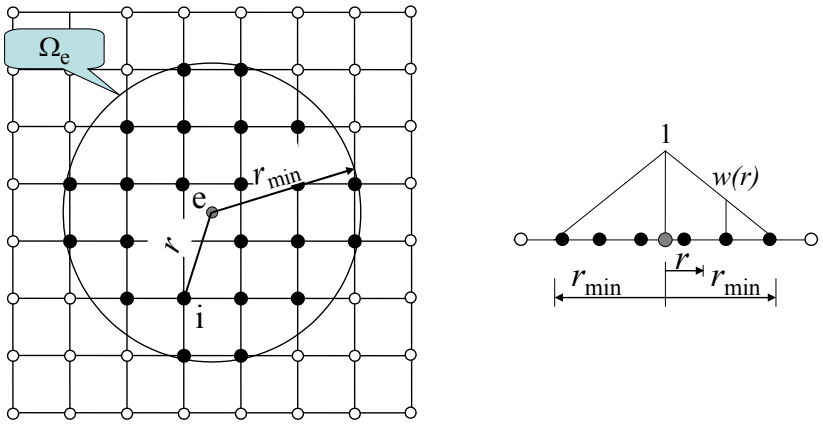
Let d_n denote all design variables associated with nodes; and let ρ_e be the value of material density at element "e". Assume that the change of material density is required to occur over a minimum length of $r_{\min}$. We obtain ρ_e from d_n as

$$\rho_e = f(d_n) \tag{15}$$

where f is the projection function. A linear projection is defined as

$$\rho_e = f(d_n) = \frac{\sum_{i \in S_e} d_i w(x_i - x_e)}{\sum_{j \in S_e} w(x_i - x_e)} \tag{16}$$

where d_i is the design variable located at node i, and S_e is the set of nodes in the domain of influence of element e (Ω^e), which is the circle of radius $r_{\min}$ and center at element "e".



(a) Domain of influence (b) Weight function

Figure 8 : Domain of influence and weight function for the nodal projection approach.

The weight function w is defined as

$$w(x_j - x_e) = \begin{cases} \dfrac{r_{\min} - r_{ie}}{r_{\min}} & \text{if } \mathrm{x}_i \in \Omega_e \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

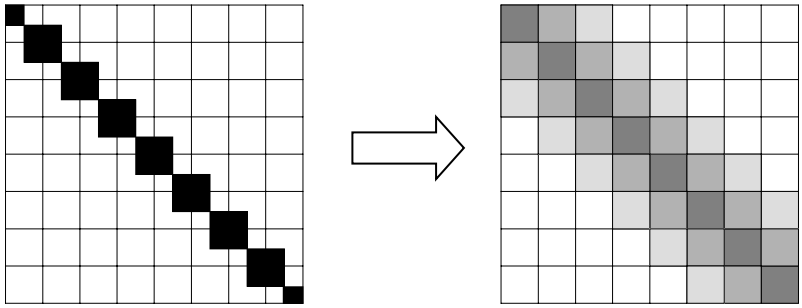
where r_{ie} is the distance between nodes "i" and element "e", i.e.

$$r_{ie} = \left\| x_i - x_e \right\| \tag{18}$$

The topology optimization problem definition is revised accordingly:

$$\begin{aligned} \min_{d_n} \quad & \theta(\mathbf{u}, \rho_e) \\ \text{s.t.} \quad & \rho_e = f\left(d_n\right) \\ & \mathbf{K}\left(\rho_e\right)\mathbf{u} = \mathbf{f} \\ & g_i(\mathbf{u}, \rho_e) \le 0 \quad i = 1..m \end{aligned} \tag{19}$$

where θ is the objective function, g_i are constraint functions, and m is the number of constraints.



(a) Distribution of d_{n} (b) Distribution of ρ_{e}

Figure 9 : Illustration for the linear projection scheme.

Sensitivities with respect to nodal design variables are obtained based on those with respect to element densities using chain-rule:

$$\frac{\partial(\bullet)}{\partial d_i} = \sum_{e \in \Omega} \frac{\partial(\bullet)}{\partial \rho_e} \frac{\partial \rho_e}{\partial d_i} \tag{20}$$

in which, Ω is the entire domain, but $\partial \rho_e / \partial d_i$ is non-zero only at nodes "e" whose influence domain (Ω_e) contains node i. The sensitivity $\partial \rho_e / \partial d_i$ is given by

$$\frac{\partial \rho_e}{\partial d_i} = \frac{w(\mathbf{x}_i - \mathbf{x}_e)}{\sum_{k \in \Omega^e} w(\mathbf{x}_k - \mathbf{x}_e)} \tag{21}$$

As usual, the sensitivity $\partial(\bullet)/\partial\rho_e$ is obtained using a traditional method such as the adjoint method. Figure 9 illustrates how the projection works.

During the optimization process, only nodal design variables are updated by the optimization update module. The structure is determined by element densities which are obtained from nodal design variables and the projection function. In other words, the optimization module does not update the topology directly as the traditional approach does. Instead, it updates the topology indirectly through the nodal design variables. The flowchart of the optimization process is illustrated in Figure 10.

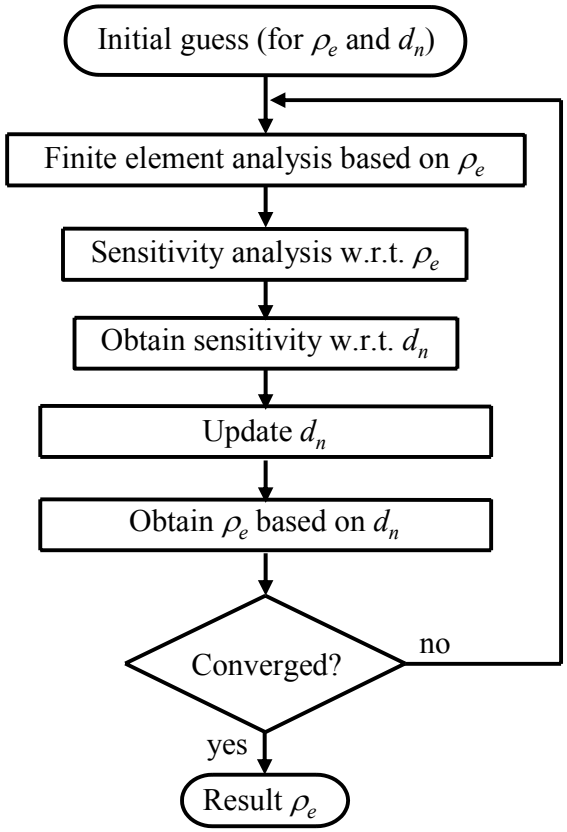


Figure 10 : Flowchart for the implementation of the nodal projection approach.

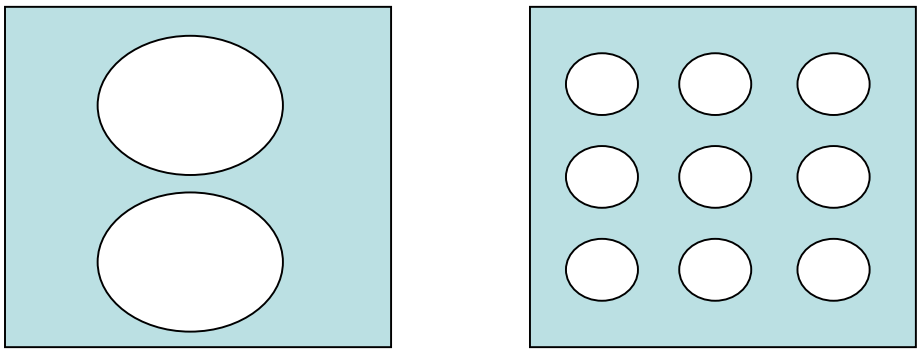
The common effect of the filter of sensitivities and the nodal projection is the fading or gray areas around structural members. The nodal projection approach is mathematically consistent and provides a restriction on the gradient of material density distribution while the filter of densities does not. However, the nodal projection approach may not induce the global convergence as well as the filter of densities does. Guest *et al.* [27] also proposed a non-linear projection to mitigate the fading effect. However, the proposed exponential expression with the power of order 25 makes the optimization process ill-conditioned and may even jeopardize the results.

It is worth noting that the slope constrained topology optimization by Petersson and Sigmund [14] may have the same effect as the nodal projection approach. By adding a constraint on maximum slope of every point in the domain based on a length scale, a well-posed problem is achieved. The checkerboard problem is eliminated and mesh-independent solution can be achieved. However, it also introduces gray areas around structural member, which is an undesirable side effect.

2.3 Perimeter constraint

Haber *et al.* [11] implemented an upper bound constraint on the total perimeter of the structure to solve the problem of non-uniqueness of solutions in topology optimization. The theoretical background for the method is developed by Ambrosio and Buttazzo [12]. Petersson [13] did further study on the approach. The idea is that structures perforated with more number of small holes will have larger total perimeters than those with large holes and less number of holes (see Figure 11). Therefore, an upper bound constraint on the total perimeter of the solution prevents the creation of large number of holes in the solution. In other words, the constraint specifically excludes finest patterns of structural members from the solution. Note that extremely fine patterns usually exist in optimal

solutions of topology optimization problems, and are the root of the complications such as non-uniqueness of the solution, checkerboard problem and mesh dependency of the solution. Exclusion of those patterns from the design space guarantees existence of solution.



(a) Structure with two large holes (b) Structure with many small holes

Figure 11 : Structures with larger holes (a) have fewer perimeters than structures with smaller holes (b) when both have the same volume.

The topology optimization problem statement corresponding to perimeter constraint approach is revised as follows:

$$\begin{aligned} \min_{\rho_e} \quad & \theta(\mathbf{u}, \rho_e) \\ \text{s.t.} \quad & \mathbf{K}\left(\mathbf{E}_e\right)\mathbf{u} = \mathbf{f} \\ & \mathbf{E}_e = \rho_e^p \mathbf{E}_o \\ & \sum_{e=1}^{n} \rho_e v_e \le V_o \\ & 0 \le \rho_e \le 1 \qquad e=1..n \\ & P \le P^* \end{aligned} \tag{22}$$

where θ is the objective function to be minimized; n is the number of elements in the domain; $\mathbf{E}_e$ is the elastic tensor corresponding to intermediate densities; $\mathbf{E}_o$ is the elastic tensor of the base material (i.e. with normalized density equal to 1); $\mathbf{K} \equiv \mathbf{K}\left(\mathbf{E}_e\right)$ is the stiffness matrix; V_o is the upper bound constraint on the

total volume of the structure; $\rho_{\min}$ is a small value to avoid singularity of the stiffness matrix; P is the total perimeter of the structure and P^* is a designer-specified value corresponding to the upper bound constraint on the perimeter.

In a binary problem setting, the total perimeter P is ready to be calculated. However, it is not practical to solve the topology optimization in the binary setting. Often, relaxed problem, where intermediate material density is allowed, is employed. With continuum formulation, the perimeter can be defined as

$$P \equiv \int_{\Omega \setminus \Gamma_j} \left|\nabla \rho\right| d\Omega + \int_{\Gamma_j} \left|\left\langle \rho \right\rangle\right| d\Gamma \tag{23}$$

where $\Omega \setminus \Gamma_j$ denotes the sub-domain obtained by subtracting Γ_j (the sub-domain where there is jump in material density) from Ω (the whole design domain); $\nabla \rho$ is the gradient of material density; and $\left\langle \rho \right\rangle$ is the jump in material density across the domain Γ_j.

In order to circumvent numerical problems with the non-differentiability of the absolute function, regularization function can be used. Refer to Haber's work [11] for further details of regularized functions.

In case the element-based approach is used, material density within each element is treated as constant. The first term in the expression of the total perimeter (P) equals zero. The expression of P may be re-written as

$$P \equiv \sum_{k=1}^{K} \ell_k \left|\left\langle \rho \right\rangle_k\right| \tag{24}$$

where K is the number of element interface; ℓ_k is the length of interface k; and $\left\langle \rho \right\rangle_k$ is the jump in material density at interface k.

Expression (24) is ready to be implemented with the finite element method used to solve the displacement field. The regularized expression for total perimeter (P) is as follows,

$$P = \sum_{k=1}^{K} \ell_k \left(\sqrt{\left\langle \rho \right\rangle_k^2 + \varepsilon^2} - \varepsilon \right) \tag{25}$$

where ε is a smoothing parameter, which should be chosen relative to the element size.

Although the perimeter constraint imposes dimensional parameters to the solution, it does not prevent local thinning of structural members [27]. The designer can control the number of holes but not a minimum member size, which is more favorable. In other words, the perimeter control does not provide an explicit length scale as in [22]. The perimeter of a structure may not be significant in the engineering point of view, unless in special cases, where one is concerned with painting areas, for example. Also, it is not trivial to determine the perimeter upper bound, especially for 3D structures, for a meaningful design.

Complication in the implementation is also a drawback for the method. With the finite element implementation, a structural member with zigzagged boundary has much more perimeter than the one with straight boundary. Thus the solution is biased towards straight members. Also, it is observed that corners are always rounded because rounded corners have smaller perimeter. Note that the optimum structures (obtained with other approaches) do not usually have rounded corners. Rounded corners are indeed a side effect of the approach.

2.4 Monotonicity based length scale

Poulsen [22, 23, 24] proposed an approach to prevent the local thinning of structural members by checking the material distribution around each and every point (elements) in the domain. If the gradient of the material distribution is monotonic within a circle of specific size around each point (element) in the domain, then structural members that are thinner than the diameter of the circle will not be generated. The idea is like passing a "looking glass" (see Figure 12) around the domain to detect the non-monotonic distribution of material and eliminate it by imposing a constraint to the optimization problem.

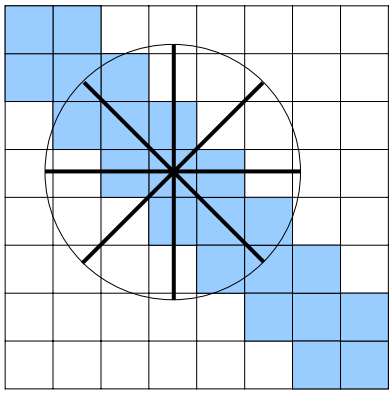


Figure 12 : The "looking glass" and 4 test directions. The gradient of material distribution is non-monotonic in the horizontal, vertical and two diagonal directions, indicating that the member violates the minimum length scale.

The monotonicity is checked in 4 directions at each point. Those directions include horizontal, vertical and two diagonal directions. The monotonicity is indicated by an indicator function that takes zero for monotonic distribution and a positive number for non-monotonic distribution. The global indicator function is obtained by summing up the indicator function at all points (elements) in the domain. Thus, we can see that if the minimum length scale is not violated at any point (element) in the domain, the global indicator function assumes zero. If there exist points (elements) where the length scale is violated in the domain, the global indicator function is greater than zero. The more the minimum length scale is violated, the larger (and positive) the global indicator function will be. Thus, by constraining the global indicator function to be zero, we can ensure that the minimum length scale is achieved. To circumvent numerical problems, the global indicator function is constrained to be smaller than a small positive number instead of zero.

The topology problem statement using the monotonicity based length scale approach is defined as follows:

$$
\begin{aligned}
\min_{\rho_e} \quad & \theta(\mathbf{u}, \rho_e) \\
\text{s.t.} \quad & \mathbf{K}(\mathbf{E}_e)\mathbf{u} = \mathbf{f} \\
& \mathbf{E}_e = \rho_e^p \mathbf{E}_o \\
& \sum_{e=1}^{n} \rho_e v_e \le V_o \\
& 0 \le \rho_e \le 1 \qquad e=1..n \\
& L \le \delta
\end{aligned}
\tag{26}
$$

where L denotes the global indicator function; and δ denotes a small positive number replacing zero to avoid numerical problems. The remaining notation in the above statement is the same as in the perimeter constraint method – see Equation (22).

The indicator function is defined as

$$
L = \sum_{e \in \Omega} \left(\sum_{v \in \Lambda} M_e(v) \right)^p \tag{27}
$$

where Ω denotes the design domain; Λ denotes the search direction space, which usually contains 4 directions (refer to Figure 12); v denotes the vector containing elements in each search direction (see Figure 13); p denotes an implementation parameter tuned to enhance global convergence; M_e denotes the local indicator function corresponding to element "e". The function $M_e(v)$ is zero if the change in densities of vector v is monotonic, and greater than zero if the change is non-monotonic. It is expressed by

$$
M(v) = \sum_{i=1}^{n-1} A(\rho_{i+1} - \rho_i) - A(\rho_n - \rho_1) \tag{28}
$$

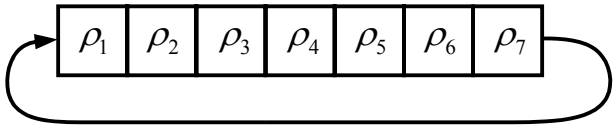
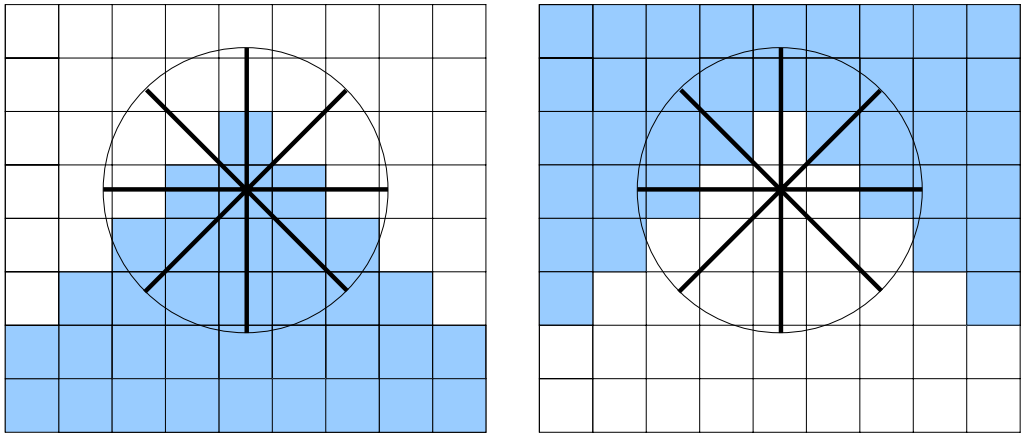


Figure 13 : Densities in vector v and cyclic manner of the calculation of the local indicator function.

The function $A(x)$ is a regularization for the absolute value of the jump in densities between element i and element $i+1$, i.e.

$$A(x) = \sqrt{x^2 + \varepsilon^2} - \varepsilon \tag{29}$$

where ε is a small positive number.



(a) Acute corner (b) Obtuse corner

Figure 14 : Monotonicity of the gradient of the material distribution is violated at corners, although corners do not violate the minimum member size.

The known problem with this approach, as the author also pointed out, is the problem with corners. Corners, either with acute or obtuse angle, violate the monotonicity of the gradient of material distribution, although corners do not violate the minimum member size. Figure 14 shows that there is a direction at corners where the gradient of material distribution is non-monotonic.

Because corners violate the monotonicity constraint, the method struggles to reduce the indicator function at corner locations. The result is that we have a mess in the corner regions. Figure 15, which is taken from [22], shows the results obtained using the monotonicity based length scale applied to the Messerschmitt-Bolkow-Blohm (MBB) beam. This problem is also solved using the new approach. The minimum member size is 20% of the beam height. The result is

shown in Figure 16 for comparison. Notice that the problems at the corners are alleviated with the new approach.

Figure 15 : Result extracted from reference [22], which illustrates that the monotonicity based length scale method leads to problems at corners.

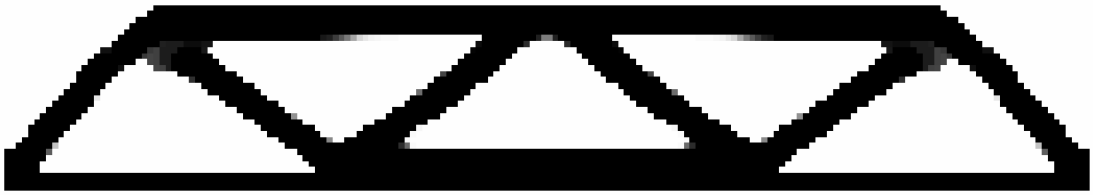


Figure 16 : Beam example using the new approach: minimum member size is 20% of the beam height; volume fraction is 45%; Poisson's ratio is 0.3; and penalization is 3 without continuation. The discretization (180x30 Q4 elements) is the same as the one in Figure 15.

2.5 Discussion of approaches

There are other methods in the literature, which deal with ill-posedness, mesh-dependence, checkerboard problem, and minimum length scale. For instance, Poulsen [23] proposed the use of wavelet space to parameterize the design space and to eliminate corner contacts and fine structural members from the solutions. Zhou and Rozvany [6], and Pomezanski [30] proposed and discussed an extended SIMP algorithm with direct corner contact control (CO-SIMP), in which a corner contact function (CCF) is used to penalize corner contacts and, as claimed by the authors, to correct the finite element method (FEM) errors associated with direct corner contacts. Jang *et al.* [31] proposed the use of non-conforming quadrilateral

elements to achieve checkerboard-free topology optimization. The approach does alleviate checkerboard problems but does not incorporate any length scale to the solution. Also, the quality of an element used for topology optimization must be assessed based on the overall performance to generate optimum, high-fidelity and manufacturing friendly structures. The non-conforming elements must be assessed for the aforementioned quality to be claimed as good elements for topology optimization.

Through the study of various approaches to well-posed, checkerboard-free, mesh-independent topology optimization solutions, we came to the following essential points for a successful approach:

- Limit the design space by excluding fine structural patterns from the admissible design space
- Achieve minimum length scale (i.e. minimum member size) for the solution

It is noticed that approaches that achieve the above two points by imposing additional constraint to the optimization problem statement are not advantageous in terms of implementation and achieving global optimum. Those approaches include perimeter constraint and monotonicity based length scale. Adding constraints, especially those related to local quantities, usually causes harm to the convergence of the solution to global optimum. That is the reason why those approaches are relatively difficult to implement. Approaches that achieve the two aforementioned points naturally without imposing additional constraints to the optimization problems, such as the filter of sensitivities and the nodal projection approaches, are more advantageous in terms of convergence to global optimum.

While achieving the aforementioned two essential points, existing approaches introduce undesirable side effects, such as gray areas, irregular corners, difficulty in obtaining global convergence, etc. In this work, we look at the essential points more directly and propose a direct approach to the points with minimum side

effects. The new approach will be implemented on top of the traditional element-based, nodal projection, and CAMD formulations.

Chapter 3 : A new scheme to achieve minimum length scale

A natural conclusion from the previous chapter is that restricting the feasible design space by excluding fine structural patterns is the essence of the solution to problems relating to non-uniqueness, mesh dependency, checkerboard patterns, and minimum length scale. In this chapter, we propose an approach to achieve minimum length scale for structural members without imposing any additional constraint and without inducing gray areas around structural members. Our proposed approach ensures directly that all structural members smaller than a designer-specified size are eliminated from the admissible design space. All problems as mentioned above are solved without any side effect to the final topological results.

3.1 Overview

The minimum length scale for topological results is obtained by introducing an additional design variable layer to the topology optimization problem and using the maximum function to project the intermediate design variable onto the material density space. The projection function can be understood as a rule to construct the structure based on intermediate design variables. The projection function will ensure that undesirable patterns are not feasible in the design domain. No constraints need to be imposed on the design variables, yet part of the material density distribution is eliminated from the design space. The part of design space that is eliminated corresponds exactly to the finest structural patterns, which is the root cause of many problems in topology optimization.

Figure 17 shows the idea of the projection and Figure 18 shows schematically the result of the projection.

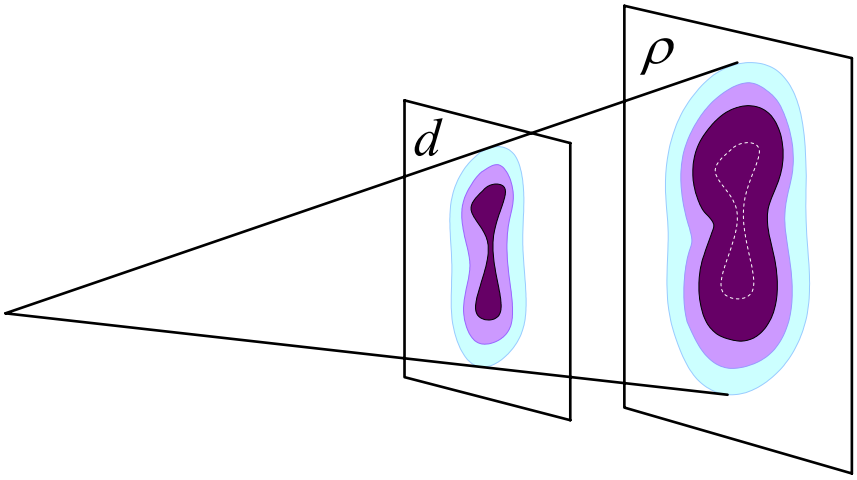
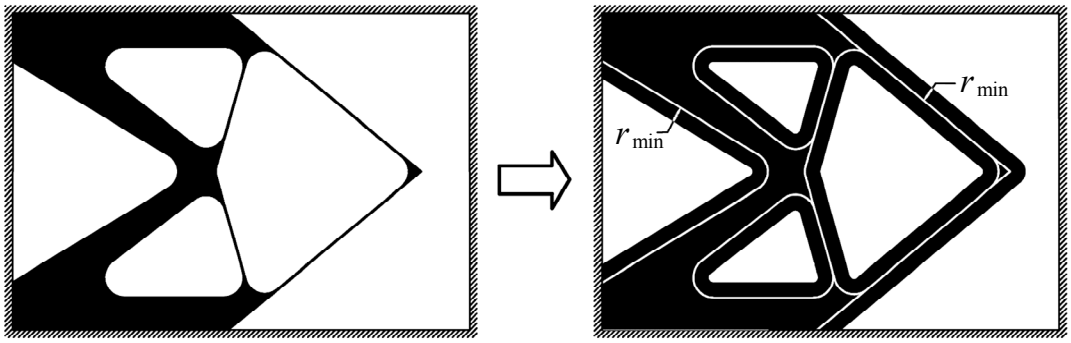


Figure 17 : Projecting design variable layer (d) onto density layer (ρ).



(a) Distribution of design variable (b) Distribution of material density

Figure 18 : Result of the projection: schematic illustration.

During the topology optimization process, sensitivities of objective and constraint functions with respect to intermediate design variables are calculated based on those with respect to material densities. The optimization module will update intermediate design variables based on the sensitivities information. The structure is obtained by projecting intermediate design variables onto material densities layer. The finite element analysis used to obtain the response field (e.g. displacement) is based only on the projected structure.

3.2 Problem statement

The new scheme to achieve minimum length scale will be applied on top of three existing approaches:

- element based
- continuous approximation of material distribution (CAMD)
- nodal projection

The topology optimization problem statements, corresponding to each approach, are described in this section. This illustrates the generality of the proposed scheme.

3.2.1 The new scheme applied to the element-based approach

We seek to minimize an objective function θ which is typically dependent on displacement and material density. The objective function should be dependent at least on the displacement, otherwise the problem becomes trivial and no iteration process is required. The objective function is minimized by varying a set of design variables (d). Material density is assumed constant inside each element. Element densities (ρ) are obtained by projecting the design variables using a specified projection rule (defined by the function f). The minimization is subjected to constraints, which can be equality or inequality constraints. Typical of those constraints are the box constraint on element densities ($0 \le \rho \le 1$) and total volume constraint ($V \le V_o$). The objective and constraint functions are calculated based on element densities. The displacement field is implicitly defined by the equilibrium equation. The problem statement based on the finite element discretization and the SIMP material interpolation model is given by:

$$
\begin{aligned}
\min_{\mathrm{d}} \quad & \theta(\mathbf{u},\rho) \\
\text{s.t.} \quad & \rho = f(\mathrm{d}) \\
& \mathbf{E}_e = \rho_e{}^p \mathbf{E}_o \\
& \left(\sum_{e=1}^{n} \mathbf{K}_e\left(\mathbf{E}_e\right) \right) \mathbf{u} = \mathbf{f} \\
& g_j(\mathbf{u},\rho) \le 0 \quad j = 1..m
\end{aligned}
\tag{30}
$$

where θ is the objective function to be minimized; d is the vector containing design variables; ρ is the vector containing element material densities; ρ_e is the material density of element e; n is the number of element in the domain; $\mathbf{E}_e$ is the elasticity tensor of materials with intermediate density; $\mathbf{E}_o$ is the elasticity tensor of solid material (with $\rho = 1$); m is number of general constraints; $\mathbf{K}_e \equiv \mathbf{K}_e\left(\mathbf{E}_e\right)$ is the stiffness matrix; the summation $(1..n)$ denotes the assembly of element stiffness matrix; $\mathbf{u}$ is the displacement vector; and $\mathbf{f}$ is the load vector. Details of the projection function f will be provided in Section 3.3.

3.2.2 The new scheme applied to the CAMD approach

The CAMD approach [28, 33] does not possess a minimum length scale and naturally possesses a certain level of smoothening effect. To achieve the minimum length scale for the CAMD approach, we introduce another layer of nodal variables on top of the existing nodal variables. The variables in the new layer are used as design variables, which are updated by the iterative optimization process. Starting from the design variables, we use the projection function to obtain nodal densities, which are then used to calculate element stiffness matrices according to the CAMD approach.

We seek to minimize an objective function θ which is typically dependent on displacement and material density. The objective function is minimized by varying a set of design variables (d). Nodal densities (Ψ) are obtained by projecting the design variables using a specified projection rule (defined by the

function f). Material density inside each element (ρ) is obtained from nodal densities (Ψ) using shape functions. Element matrices are calculated using the graded finite element formulation by Kim and Paulino [34]. The minimization is subjected to constraints, which can be either equality or inequality constraints. Typical of those constraints are the box constraint on element densities ($0 \le \Psi \le 1$), and total volume constraint ($V \le V_o$). The objective and constraint functions are calculated based on nodal densities (Ψ) using graded finite element formulation [34]. Displacement field is implicitly defined by the equilibrium equation. The problem statement based on the finite element discretization and the SIMP material interpolation model is given by:

$$\begin{aligned} \min_{\mathbf{d}} \quad & \theta(\mathbf{u},\rho) \\ \text{s.t.} \quad & \mathbf{\Psi} = f(\mathbf{d}) \\ & \rho = \sum_{i=1}^{N} N_i \Psi_i \\ & \mathbf{E} = \rho^p \mathbf{E}_o \\ & \mathbf{K}(\mathbf{E})\mathbf{u} = \mathbf{f} \\ & g_j(\mathbf{u},\rho) \le 0 \quad j = 1..m \end{aligned} \tag{31}$$

where θ is the objective function we wish to minimize; $\mathbf{d}$ is the vector containing design variables; $\mathbf{\Psi}$ is the vector containing nodal densities; ρ is the material density field (inside elements); N is the number of nodes of each element; N_i is the shape function associated with node i; $\mathbf{E}$ is the elasticity tensor of materials with intermediate density; $\mathbf{E}_o$ is the elasticity tensor of solid material (with $\rho = 1$); $\mathbf{K} \equiv \mathbf{K}(\mathbf{E})$ is the stiffness matrix; $\mathbf{u}$ is the displacement vector; $\mathbf{f}$ is the load vector.; and m is number of general constraints.

According to the graded finite element formulation [34], the element stiffness matrix is obtained using Gauss quadrature. Material densities are evaluated at Gauss points by interpolation using shape functions. The implementation of the CAMD approach includes the calculation of material density at Gauss points based on nodal values and shape functions, and the calculation of sensitivities

with respect to nodal densities based on sensitivities with respect to densities at Gauss points.

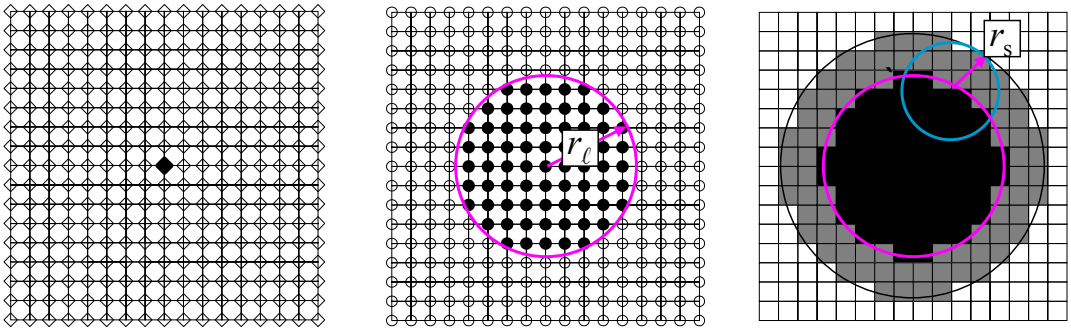
3.2.3 The new scheme applied to the nodal projection approach

The nodal projection approach by Guest *et al.* [27] also provides a smoothening effect because it possesses restriction on the gradient of material density. Moreover, the nodal projection approach itself imposes a minimum length scale. However, it induces "gray" areas of material, which is not practically meaningful. In this section, the new scheme is implemented on top of the nodal projection approach. The implementation takes advantage of the smoothening effect of the nodal projection approach, and the minimum length scale of the new projection scheme. Thus, it allows separate control over two effects: length scale and smoothening.

The nodal projection approach is similar to the CAMD approach in that they both employ nodal variables as design variables. The differences are in the assumption of the material distribution and the projection/interpolation of material density inside each element. The CAMD approach assumes continuous material distribution inside each element and uses shape functions for interpolation. The nodal projection approach assumes constant material density for each element and uses projection functions to obtain element material density.

The implementation of the new projection scheme with nodal projection approach is similar to its implementation for CAMD approach. Notice, however, that there are two values of length scale involved in this process (see Figure 19): one is used by the original nodal projection approach itself (let us denote it by r_s); the other is used by the new projection scheme (let us denote it by r_ℓ). The parameter r_s serves the purpose of smoothening the optimization problem. The parameter r_ℓ serves as a minimum length scale. Larger r_s provides more

smoothening but also causes greater fading effect. Continuation of r_ℓ, with larger value at the beginning of the optimization process and minimum value at the end of the process when the solution has sufficiently converged is a good option.



(a) Design variables (b) Nodal densities (c) Element densities

Figure 19 : Illustration of length scales (r_ℓ and r_s). One "black" design variable creates a circular region of "black" nodal densities, which then create "gray" and "black" regions of element density.

We use a relatively small value for r_s (r_s =1) and the linear projection for the numerical example provided in this paper. With that setting, the projection from nodal densities to element densities becomes very simple: the material density of each element is simply the average of the four nodal values of the nodal densities. In this case, the final topology can be represented directly by the nodal densities so that the fading effect problem is circumvented.

The problem statement is described as follows:

$$\begin{aligned} \min_{\mathbf{d}} \quad & \theta(\mathbf{u},\rho) \\ \text{s.t.} \quad & \boldsymbol{\Psi} = f_\ell(\mathbf{d}) \\ & \boldsymbol{\rho} = f_s(\boldsymbol{\Psi}) \\ & \mathbf{E}_e = \rho_e^p \mathbf{E}_o \\ & \left(\sum_{e=1}^{n} \mathbf{K}_e(\mathbf{E}_e)\right)\mathbf{u} = \mathbf{f} \\ & g_j(\mathbf{u},\rho) \le 0 \quad j = 1..m \end{aligned} \tag{32}$$

where θ is the objective function we wish to minimize; $\mathbf{d}$ is the vector containing design variables; $\mathbf{\Psi}$ is the vector containing nodal densities; ρ is element material densities; f_ℓ is the projection function used for the new scheme; f_s is the projection function used for the nodal projection approach; $\mathbf{E}$ is the elasticity tensor of materials with intermediate density; $\mathbf{E}_o$ is the elasticity tensor of solid material; $\mathbf{K}_e \equiv \mathbf{K}_e(\mathbf{E}_e)$ is the stiffness matrix; the summation (1..n) denotes the assembly of element stiffness matrices; $\mathbf{u}$ is the displacement vector; $\mathbf{f}$ is the load vector; and m is number of general constraints.

3.3 The projection function

In the previous section, we have mentioned about the projection function to obtain densities from design variables. For each implementation, the projection function is given as follows:

- element-based: $\boldsymbol{\rho} = f(\mathbf{d})$
- CAMD : $\mathbf{\Psi} = f(\mathbf{d})$ (33)
- nodal projection: $\mathbf{\Psi} = f_\ell(\mathbf{d})$

In the above, $\mathbf{d}$ denotes the vector of design variables; $\boldsymbol{\rho}$ and $\mathbf{\Psi}$ denote element densities and nodal densities, respectively.

In this section, we describe the projection function for the first case: the projection between design variables and element densities. The other two cases are similar. Also, for the sake of visualization, the description is in a two-dimensional context. However, the approach is valid in a three-dimensional context as well. The projection employs the maximum function:

$$\rho_i = \max_{j \in \Omega_i}(d_j) \tag{34}$$

where ρ_i is the material density of element (or node) i, assumed to be located at the centroid of the element (or at the node); d_j is the design variable located at

the centroid of element (or node) j; and Ω_i is the sub-domain corresponding to element (or node) i. More explanation for the notation follows.

The arrangements of design variables and element densities (i.e. their locations) are coincident. At the centroid of each element (or at each node) there is a material density and a design variable. In Figure 21, element (or nodal) densities and design variables are shown in two separate drawings for clarity. However, those two drawings should be understood as superposed. The two can take different values. The relation between the element (or nodal) densities and design variables are determined only by the projection.

The sub-domain Ω_i corresponding to element (or node) i is defined as a circle with its center located at the centroid of element (or at the node) i and radius equal $r_{\min}$. For a discretized problem, Ω_i are the set of nodes j defined by Figure 20 and:

$$j \in \Omega_i \quad \text{if} \quad r \equiv \left(r_i - r_j\right) < r_{\min} \tag{35}$$

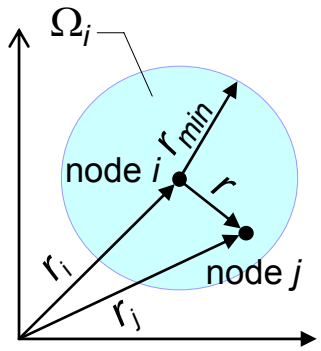


Figure 20 : Sub-domain corresponding to point i (either element or node).

Note that $r_{\min}$ is a designer-defined parameter, which determines the minimum length scale of the topological solution. Physically, $r_{\min}$ is half of the minimum size of structural members in the optimized structure. The parameter j is any element (or node) that lies within Ω_i. Figure 21 illustrates the sub-domain and related entities.

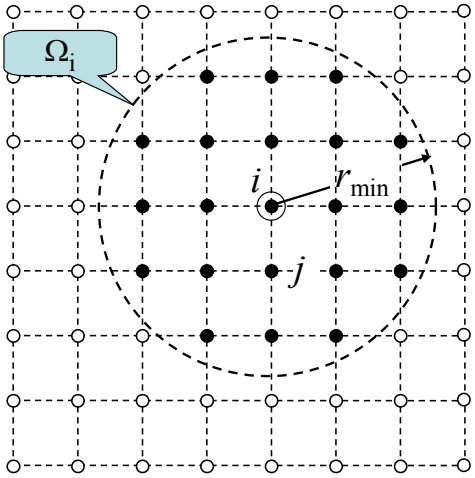
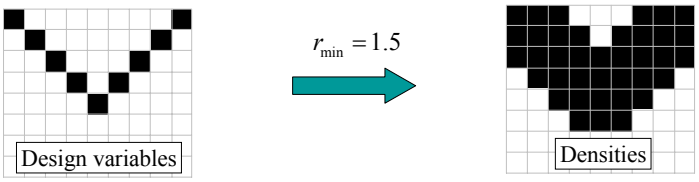
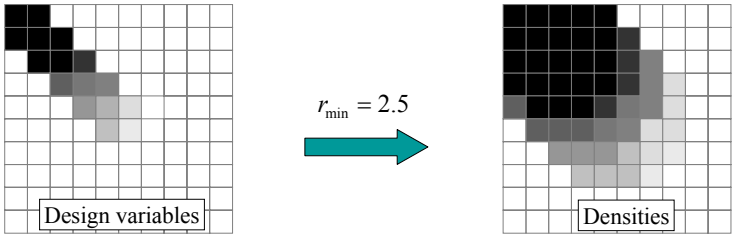


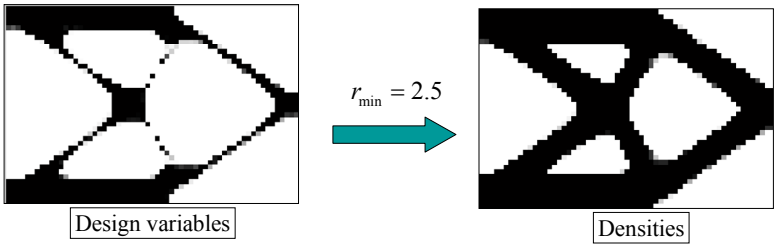
Figure 21 : Sub-domain of node (or element) *i*.



(a) Example with integer design variables



(b) Example with "gray" design variables



(c) Results obtained from actual simulation

Figure 22 : Results of the new projection scheme.

Figure 22 shows the results of the projection to illustrate how it actually works. Figure 22(a) shows the result of the projection with the minimum member size obtained. Observe that the resulting structural member (densities) cannot be thinner because the design variables cannot shrink more. Figure 22(b) shows how gray areas in the design variable layer are projected to density layer. Figure 22(c) shows a result taken directly from actual simulation.

3.4 Mapping between design variables and material densities

To facilitate the implementation of the new approach, we provide in this section an explanation of the new projection scheme from a different perspective. The projection using the maximum function described in previous sections is essentially a mapping between design variables and element (or nodal) densities. Each element (or nodal) density is equal to one design variable, in other words, mapped to one design variable. One design variable may control the value of one or more than one element (or nodal) densities. The forward mapping is defined as the mapping from design variables to element (or nodal) densities. The inverse mapping is the one from element (or nodal) densities to design variables. Figure 23 illustrates both the forward and the inverse mapping.

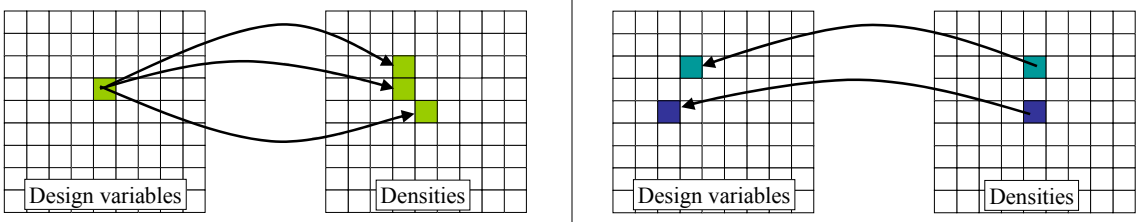


Figure 23 : Forward and inverse mapping.

The forward and inverse mapping are sufficient to calculate element (or nodal) densities from design variables, and calculate sensitivities with respect to design variables from sensitivities with respect to element (or nodal) densities. The calculation of element (or nodal) densities from design variables is formally known as the projection of design variable layer onto the element (or nodal) density layer. The calculation of sensitivities with respect to design variable from element (or nodal) densities may be understood as the inverse of the projection.

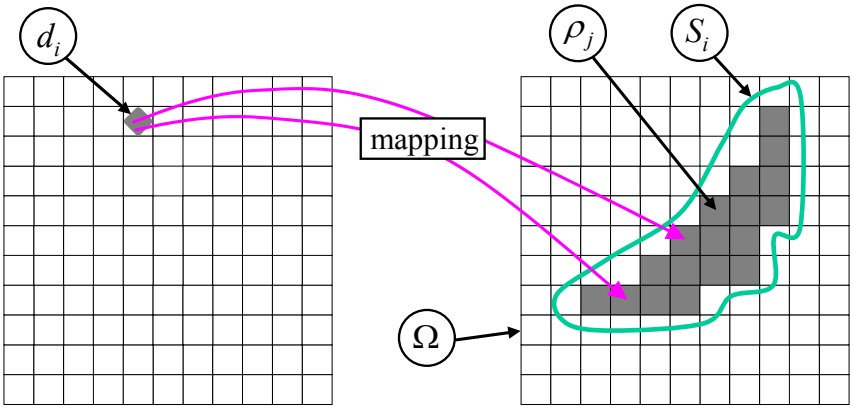
The mapping changes during the optimization process. The change occurs between consecutive iteration steps when design variables are updated at the end of each iteration step. It is obvious that during one iteration step, neither the design variables are changed, nor the mapping. Therefore the projection is consistent inside each iteration step. After the update of design variables, the mapping is refreshed, the structure is re-analyzed, objective and constraint functions are re-calculated, and sensitivities are re-evaluated.

The mapping facilitates the calculation of sensitivities with respect to design variables. In general, the sensitivities with respect to element (or nodal) densities are calculated using traditional methods, such as adjoint method. The sensitivities with respect to design variables are calculated from the sensitivities with respect to element (or nodal) densities using chain rule. The mapping serves to bridge the two classes of sensitivities.

3.5 Sensitivity analysis

The calculation of sensitivities with respect to design variables based on sensitivities with respect to element (or nodal) densities is provided. The calculation of sensitivities with respect to element (or nodal) densities for element based approach using adjoint method can be found in Bendsoe and Sigmund's book [38]; for the CAMD approach, see Matsui and Terada [28]; and for the nodal projection approach, see Guest *et al.* [27].

Suppose we need to calculate the sensitivities of a function Θ, which may be an objective or constraint function, with respect to design variables. The function Θ can be considered as a function of element (or nodal) densities ρ, which are again a function of design variables d, i.e. $\Theta \equiv \Theta\left(\rho(d)\right)$. Let us look at the sensitivity with respect to a specific design variable d_i. We will determine the variation of design variable (δd_i), the variation of element (or nodal) densities ($\delta\rho$), and then the variation of the function $\delta\Theta$. In the previous section, we have discussed about the mapping between design variables and element (or nodal) densities. Let us assume that the mapping is known and S_i is the set of element (or nodal) densities, which are mapped to the design variable d_i. See Figure 24 for an illustration.



(a) Design variables (b) Element (or nodal) densities

Figure 24 : Relating design variables (d) and element (or nodal) densities (ρ).

The variation of d_i causes the variation of a number of element (or nodal) densities, which belong to the influence set S_i as follows:

$$\delta\rho_j\left(d_i, \delta d_i\right) = \begin{cases} \delta d_i & \text{for} \quad j \in S_i \\ 0 & \text{otherwise} \end{cases} \tag{36}$$

The variation of the function Θ is calculated as

$$\delta\Theta\left(\rho\left(d_i\right),\delta\rho\left(d_i,\delta d_i\right)\right)=\sum_{k\in\Omega}\frac{\partial\Theta}{\partial\rho_k}\delta\rho_k=\sum_{j\in S_i}\frac{\partial\Theta}{\partial\rho_j}\delta d_i=\left(\sum_{j\in S_i}\frac{\partial\Theta}{\partial\rho_j}\right)\delta d_i \tag{37}$$

Thus, the sensitivity of the function Θ with respect to design variable d_i is expressed as

$$\frac{\partial\Theta}{\partial d_i}=\sum_{j\in S_i}\frac{\partial\Theta}{\partial\rho_j} \tag{38}$$

where $\partial\Theta/\partial\rho_j$ are sensitivities with respect to element (or nodal) densities. The calculation of those depends on the type of function and type of problem, such as minimum compliance, compliant mechanism, inverse homogenization, and so on. The details of those calculations are not included in this thesis but can be easily found in published literature. See, for example, reference [38] for sensitivities with respect to element densities in element based approach, reference [28] for sensitivities with respect to nodal densities in the CAMD approach, and reference [27] for sensitivities with respect to nodal densities in the nodal projection approach.

The influence set S_i is built for each iteration step while doing the projection. For each element (or nodal) density ρ_i, the maximum design variable is sought among the sub-domains Ω_i. Then a mapping is created to link the element (or nodal) density and the maximum design variable.

It is worth discussing an important case that arises during the sensitivity analysis: the case when the influence set S_i is empty. In this case, the design variable d_i does not control any element (or nodal) densities. Therefore changes in that design variable does not lead to any change in element (or nodal) densities. Mathematically, that means the sensitivity with respect to the design variable d_i vanishes in the current iteration step. However, empirically, the simulation works better if we treat the sensitivities with respect to the design variable d_i as unknown for this case. The treatment of zero sensitivities and unknown sensitivities can be the same or different depending on the type of

problem and optimization update scheme. In general, zero sensitivities do not lead to zero updatee for the corresponding design variablee. On the other hand, unknown sensitivities lead to unchanged (zero update) design variables for the current optimization step.

3.6 Discussion

The essence of the new projection scheme lies in its simplicity and effectiveness. The new scheme excludes from the design space structural members which are finer than a designer-specified minimum length scale parameter. The outcome is a natural restriction of the feasible design space with only undesirable solutions eliminated. The new scheme avoids patterns that cause problems in topology optimization, such as non-uniqueness of the solution, mesh-dependency, numerical instabilities, and manufacturing unfriendliness. Compared to other approaches, the exclusion of undesirable design space is done in the most direct manner. Thus there are no pronounced side effects commonly seen in other approaches, such as gray areas, rounded and messy corners, and directional bias.

Chapter 4 : Numerical implementation of the minimum length scale scheme

The new projection scheme is general and applicable to various topology optimization problems. To illustrate, the implementation for the minimum compliance problem is provided. We seek to minimize the compliance (or maximize the stiffness) with a constraint on total volume of the structure. The compliance is defined as

$$c = \mathbf{u}^T \mathbf{f} \tag{39}$$

And the volume constraint is

$$V = \int_\Omega \rho dV \leq V_o \tag{40}$$

This section contains pseudo-codes and flowcharts for the implementation of the new scheme on top of three approaches: element-based, CAMD, and nodal projection.

4.1 Element-based approach

The basic implementation for the element-based approach is adopted from Sigmund's educational paper [20]. The modification for the new scheme is provided below. It is further illustrated with the flowcharts given in Figure 25 and Figure 26, and the pseudo-code 1 through 4.

Figure 25 illustrates the main flowchart for implementing the new projection scheme with the element-based approach. At the beginning of the program, an array containing information of the sub-domains Ω_i for every element is built.

Each element will be assigned a list of elements that lie within the circle defining Ω. The sub-domain information kept for the entire program cycle. It is used for the projection from the design variable layer onto element density layer (Figure 22), which is essentially the search for the maximum design variable that controls each element density.

The initial guess for design variables is chosen to be the same as the initial guess for element densities. The map between design variables and element densities is initialized so that the design variable at any element is mapped to the element density at that element.

After the calculation of sensitivities with respect to element densities, a summation process is carried out to calculate sensitivities with respect to design variables. This is essentially a loop over all elements and at each cycle. Sensitivity with respect to the element density is added to the sensitivity with respect to the design variable, which is mapped to that element. Note that the mapping has been built earlier (see Section 3.4).

The Optimality Criteria Update is employed as the optimizer (see Figure 26). The inner loop to find the Lagrangian multiplier is modified. At each of the inner iteration steps, the design variables are updated according to the current value of the Lagrangian multiplier. Element densities are obtained by the new projection, which is essentially the search for maximum design variable within the sub-domain of each element. Then total volume is calculated and compared to the upper bound constraint. A Newton bisectional scheme is followed to modify the Lagrangian multiplier until the volume constraint is satisfied. At each of the inner iteration steps, the result of the projection mapping is recorded in an array. The mapping at the final step of the inner iteration loop is the mapping for the calculation of sensitivities for the next step. Other optimizers, such as the MMA [39], need to be modified to output the mapping. The flowchart of the main program is given in Figure 25, and the flowchart of the Optimality Criteria Update sub-function is given in Figure 26.

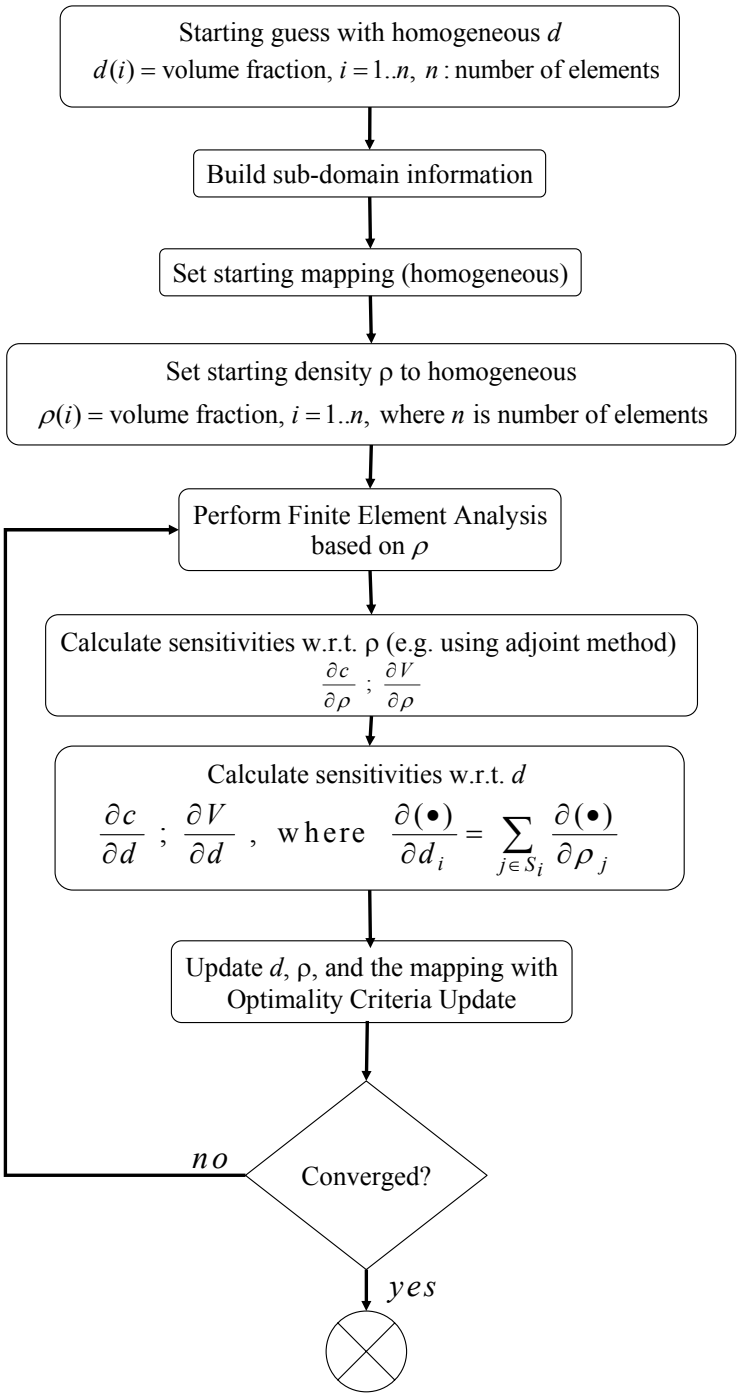


Figure 25 : Flowchart for implementing the new projection scheme with the element-based approach: main program. The Optimality Criteria (OC) Update is given in Figure 26.

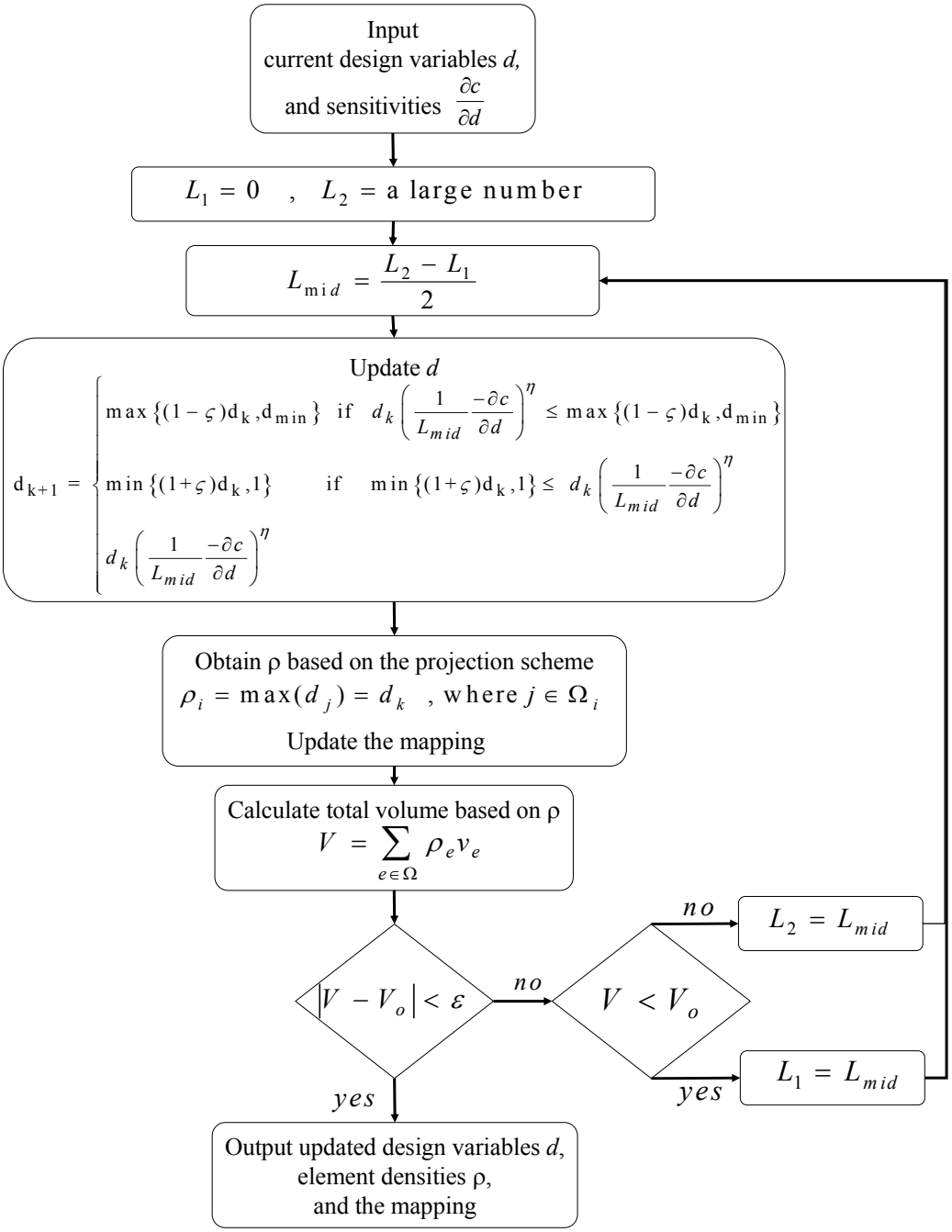


Figure 26 : Flowchart for implementing the new projection scheme with the element-based approach: Optimality Criteria (OC) Update.

The pseudo-code for the implementation with the element-based approach is provided below.

Pseudo-code 1: Main function

```
Notes:
Input parameters
nelx: number of elements in x direction
nely: number of elements in y direction
V_o: averaged volume fraction
p: penalization parameter
r_min: length scale parameter (half of the minimum member width)

Selected internal variables
xd: design variables
xe: element densities
dc_xd: Sensitivities of the compliance w.r.t. design variables
dc_xe: Sensitivities of the compliance w.r.t. element densities
change: maximum change of element densities between two iterations
enum: element id
ne: number of elements belonging to sub-domain of each element
elistx: list of column numbers of element sub-domain
elisty: list of row numbers of element sub-domain
mapx: list of col number of the design variable mapped to element
mapy: list of row number of the design variable mapped to element
KE: element stiffness matrix of solid material
c: objective function (which is the compliance in this case)
Ue: element displacement, vector
```

```
01: xd ← V_o ; xe ← V_o
02: dc_xe ← 0 ; dc_xe ← 0
    << Build the list of adjacent elements >>
03: FOR all elements in the domain (having row ely and column elx) DO
04:    enum ← nelx(elx-1) + ely
05:    FOR all elements (having row nly and column nlx) DO
06:       r ← sqrt((elx - nlx)^2 + (ely - nly)^2)
07:       IF r ≤ r_min THEN
08:          ne(enum,1) ← ne(enum) + 1
09:          elistx(enum,ne(enum)) ← nlx
10:          elisty(enum,ne(enum)) ← nly
```

```
11:       END IF
12:    END FOR
13: END FOR
    << Initialize the map from element densities to design variables >>
14: FOR all elements in the domain (having row ely and column elx) DO
15:    mapy(ely,elx) ← ely
16:    mapx(ely,elx) ← elx
17: END FOR
18: CALL (lk) sub-function to compute (KE)
    << Start iteration >>
19: WHILE change > 0.01 DO
20:    loop ← loop + 1
21:    CALL sub-function (FE) with (nelx,nely,xe,penal) to compute (U)
22:    << Calculate compliance and sens w.r.t. elem densities >>
23:    c ← 0
24:    FOR all elements in the domain (having row ely and column elx) DO
25:       Extract (Ue) from (U)
26:       c ← c + xe(ely,elx)^p(Ue)'(KE)(Ue)
27:       dc_xe(ely,elx) ← -(p)xe(ely,elx)^(p-1)(Ue)'(KE)(Ue)
28:    END FOR
       << Calculate sensitivities w.r.t. design variables >>
29:    tempdc ← 0 ; nan ← 0
30:    FOR all elements in the domain (having row ely and column elx) DO
31:       tempdc(mapy(ely,elx),mapx(ely,elx)) ← dc_xe(ely,elx)
32:       nan(mapy(ely,elx),mapx(ely,elx)) ←
          nan(mapy(ely,elx),mapx(ely,elx)) + 1
33:    END FOR
       << Normalize the sens by dividing by the influenced volume >>
34:    FOR all elements in the domain (having row ely and column elx) DO
35:          IF nan(ely,elx) > 0 THEN
36:             dc_xd(ely,elx) ← tempdc(ely,elx)/nan(ely,elx);
37:          ELSE
38:             dc_xd(ely,elx) ← dc_xe(ely,elx);
39:          END IF
40:    END FOR
41:    xdold ← xd ; xeold ← xe
42:    CALL (OC) sub-function with
43:         (nelx,nely,xd,volfrac,dc_xd,ne,elistx,elisty)
44:         to compute (xd,xe,mapy,mapx)
45:    Output (loop, c, change, total volume)
46:    Plot (xe) for visualization
47: END WHILE
```

Pseudo-code 2: Sub-function OC

Notes:
Input variables:
nelx: number of element in x direction
nely: number of element in y direction
xd: current design variables
V_o: volume fraction
dc_xd: sensitivities w.r.t. design variables
ne: number of elements belonging to subdomain of each element
elistx: list of column numbers of element sub-domain
elisty: list of row numbers of element sub-domain

Output variables:
xdnew: updated design variables
xenew: updated element densities
mapx: list of col number of the design variable mapped to element
mapy: list of row number of the design variable mapped to element

Selected local variables:
maxnodey: row number of the maximum design variables in range
maxnodex: col number of the maximum design variables in range
enum: element id
l1, *l2*, *lmid*: temporary values of the Lagrangian multipliers used by the bi-sectioning algorithim

01: $l1 \leftarrow 0$
02: $l2 \leftarrow 100000$
03: $move \leftarrow 0.2$

04: **WHILE** $l2\text{-}l1 > 1e\text{-}4$ **DO**
05: $mapy \leftarrow 0$; $mapx \leftarrow 0$
06: $lmid \leftarrow 0.5(l2 + l1)$
07: $$xdnew \leftarrow \begin{cases} \max\left\{xd - move, 0.001\right\} & \text{if } xd\left(\frac{-dc\_xd}{lmid}\right)^{1/4} \leq \max\left\{xd - move, 0.001\right\} \\ \min\left\{xd - move, 0.001\right\} & \text{if } \min\left\{xd - move, 0.001\right\} \leq xd\left(\frac{-dc\_xd}{lmid}\right)^{1/4} \\ xd\left(\frac{-dc\_xd}{lmid}\right)^{1/4} & \text{otherwise} \end{cases}$$

```
08:    xnew ← 0
09:    FOR all elements in the domain (having row ely and column elx) DO
10:       enum ← nely(elx-1)+ely
11:       FOR all elements (i) in the sub-domain DO
12:          IF xenew(ely,elx)<xdnew(elisty(enum,i),elistx(enum,i)) THEN
13:             xenew(ely,elx) ← xdnew(elisty(enum,i),elistx(enum,i))
14:             maxnodey ← elisty(enum,i)
15:             maxnodex ← elistx(enum,i)
16:          END IF
17:       END FOR
18:       mapy(ely,elx) ← maxnodey
19:       mapx(ely,elx) ← maxnodex
20:    END FOR
21:    IF total volume > V_o THEN
22:       l1 = lmid
23:    ELSE
24:       l2 = lmid
25:    END IF
26: END WHILE
```

Pseudo-code 3: Sub-function FE

```
Notes:
Input variables:
nelx: number of element in x direction
nely: number of element in y direction
x: element densities
p: penalization parameter

Output variables:
U: Displacement vector

Selected local variables
KE: element stiffness matrix of solid material
K: global stiffness matrix
F: global load vector
edof: element degrees of freedom
fixeddofs: prescribed degrees of freedom
alldofs: all degrees of freedom
freedofs: free degrees of freedom
```

```
01: CALL sub-function (lk) to compute (KE)
02: K ← 0
03: F ← 0
04: FOR all elements in the domain (having row ely and column elx) DO
05:    Build element degrees of freedom table
06:    Assemble (K) using (xe(ely,elx))^P(KE) and element dofs table
07: END FOR
    << Define loads and supports >>
08: F(2(nely+1)nelx+nely/2+1)) ← 1
09: fixeddofs ← 1 : 2(nely+1)
10: alldofs ← 1 : 2(nely+1)(nelx+1)
11: freedofs ← alldofs ∩ fixeddofs
12: Extract Kff corresponding to freedofs
13: Extract Fff corresponding to freedofs
14: Solve linear system (Kff)(Uff) = Kff for Uff
15: U(fixeddofs) ← 0
```

Pseudo-code 4: Sub-function lk

```
Notes:
Input variables: none
Output variables: KE

Selected local variables:
E: Young's modulus
nu: Poison's ratio
k: vector of temporary values

01: E ← 1
02: nu ← 0.3
03: k ← [1/2-nu/6, 1/8+nu/8, -1/4-nu/12, -1/8+3*nu/8, -1/4+nu/12,
        -1/8-  nu/8, nu/6, 1/8-3*nu/8]
04: KE ← E/(1-nu^2)[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
                     k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
                     k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
                     k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
                     k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
                     k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
                     k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
                     k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)]
```

4.2 CAMD approach

The basic implementation for the CAMD approach is adopted from Matsui and Terada [28]. The modification for the new scheme is similar to the modification done with the element-based approach; however, the element densities are replaced by the nodal densities.

The difference between the implementation with the element-based and CAMD approach is as follows. For the element-based approach, material density is assumed constant in side each element. Element matrices of intermediate densities are proportional to that of the basic material by a factor of ρ^p, where ρ is the element density and p is the penalization factor. For the CAMD approach, material density is assumed to be continuously varying inside each element and is interpolated using nodal values and shape functions. Element matrices must be integrated for each element using, for example, the graded finite element formulation by Kim and Paulino [34]. The output topology of the CAMD approach is the continuous material density field which is interpolated from nodal values instead of solid and void element as in the case of the element-based approach.

The flowchart for CAMD approach is the same as that of the element-based approach because all the differences are hidden in the finite element analysis and the calculation of sensitivities with respect to ρ. Details of those calculations can be found in reference [28].

4.3 The nodal projection approach

The basic implementation for the nodal projection approach is adopted from the paper by Guess *et al.* [27]. See Section 2.2 for more information of the basic nodal

projection approach. In the Matlab code implementation, the inner projection of the original nodal projection approach is simplified by fixing the smoothening parameter r_s to 1. The implementation described in this section is for a general case when both the length scale parameter r_ℓ and smoothening parameter r_s are user-defined. See Figure 24 for more information of r_ℓ and r_s. The explanation of the implementation is further illustrated with the flowcharts given in Figure 27 and Figure 28.

Essentially, two consecutive projection schemes are carried out in order to obtain element densities from design variables. The first is the new scheme which uses the maximum function. The second is the original linear projection by Guess *et al.* [28] using the weighted average function. The first projection helps achieving minimum member size and the second helps smoothening the optimization process.

At the beginning of the program, two arrays containing information of the sub-domains Ω_i^ℓ and Ω_i^s for every element is built. Each element will be assigned a list of elements that lie within the circle defining Ω_i^ℓ and Ω_i^s. The information of sub-domains is kept for the entire program cycle. It is used for the projection from the design variable layer onto nodal density layer (which is essentially the search for the maximum design variable that control each element density), and from the nodal densities layer to element density layer (which is the weighted average smoothening process.)

The initial guess for design variables is chosen to be the same as the initial guess for nodal densities, and initial element densities are calculated from initial nodal densities. The map between design variables and nodal densities is initialized so that the design variable any node is mapped to the nodal density at that node.

After the calculation of sensitivities with respect to element densities, sensitivities with respect to nodal densities are calculated based on the inverse of the weighted averaging function. Then a summation process is carried out to

calculate sensitivities with respect to design variables from sensitivities with respect to nodal densities. This is essentially a loop over all nodes and at each cycle. For all nodes, sensitivity with respect to each nodal density is added to the sensitivity with respect to the design variable, which is mapped to that nodal density. Note that the mapping has been built earlier.

A modified version of the Optimality Criteria Update [20] is employed as the optimizer (see Figure 28). As before, the inner loop to find the Lagrangian multiplier is modified. At each of the inner iteration steps, the design variable is updated according to the current value of the Lagrangian multiplier. Nodal densities are obtained by the projection, which is essentially the search for maximum design variable within the sub-domain of each node. Element densities are calculated based on the weighted averaging function. Element densities are summed up to obtain the total volume, which are then compared to the upper bound constraint. A Newton bisectional scheme is followed to modify the Lagrangian multiplier until the volume constraint is satisfied. At each of the inner iteration step, the result of the projection mapping is recorded in an array. The mapping at the final step of the inner iteration loop is the one used for the calculation of sensitivities for the next step. The flowchart of the main program is given in Figure 27, and the flowchart of the Optimality Criteria Update sub-function is given in Figure 28.

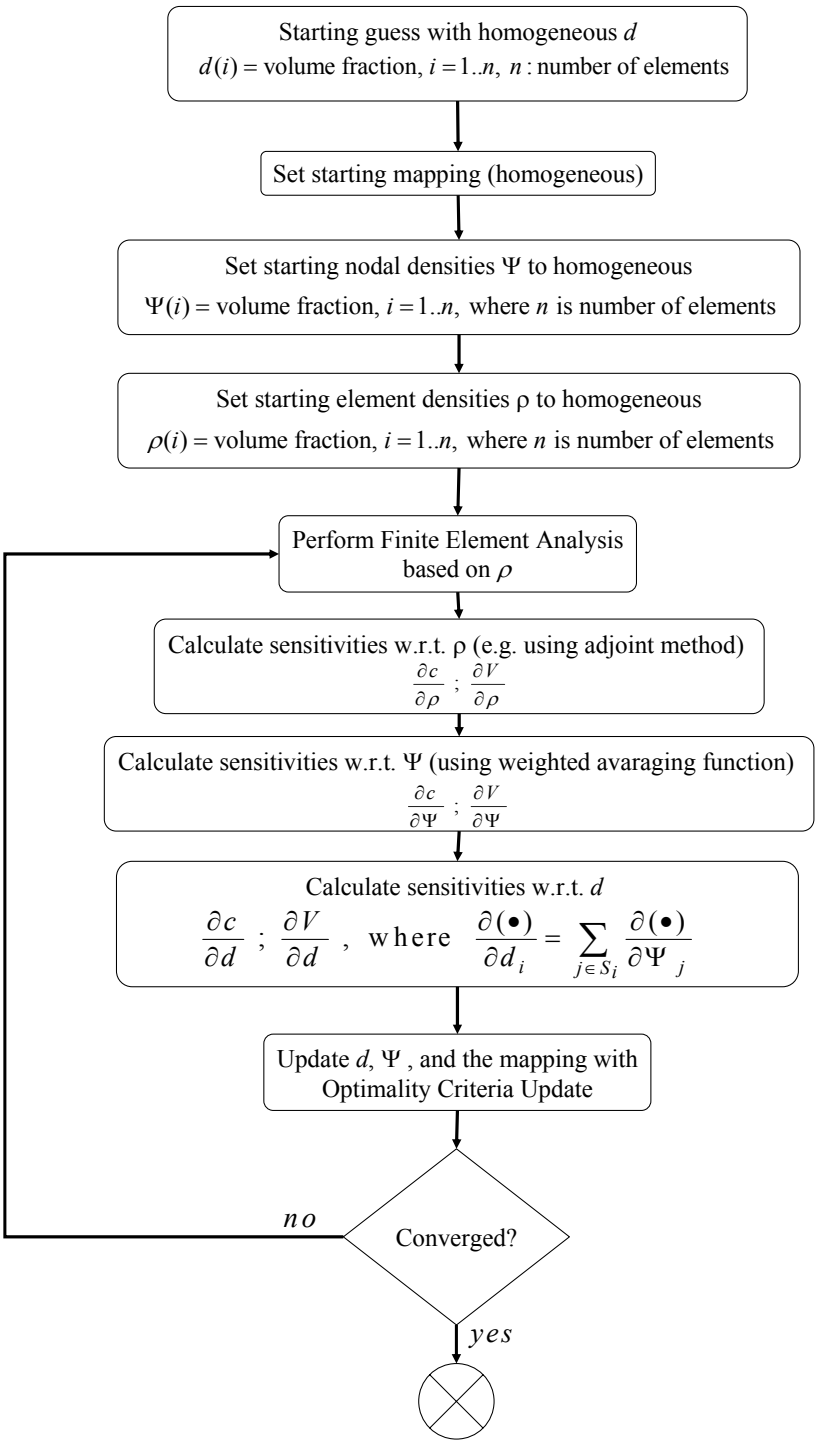


Figure 27 : Flowchart for implementing the new projection scheme with the nodal projection approach: main program. The Optimality Criteria (OC) Update is given in Figure 28.

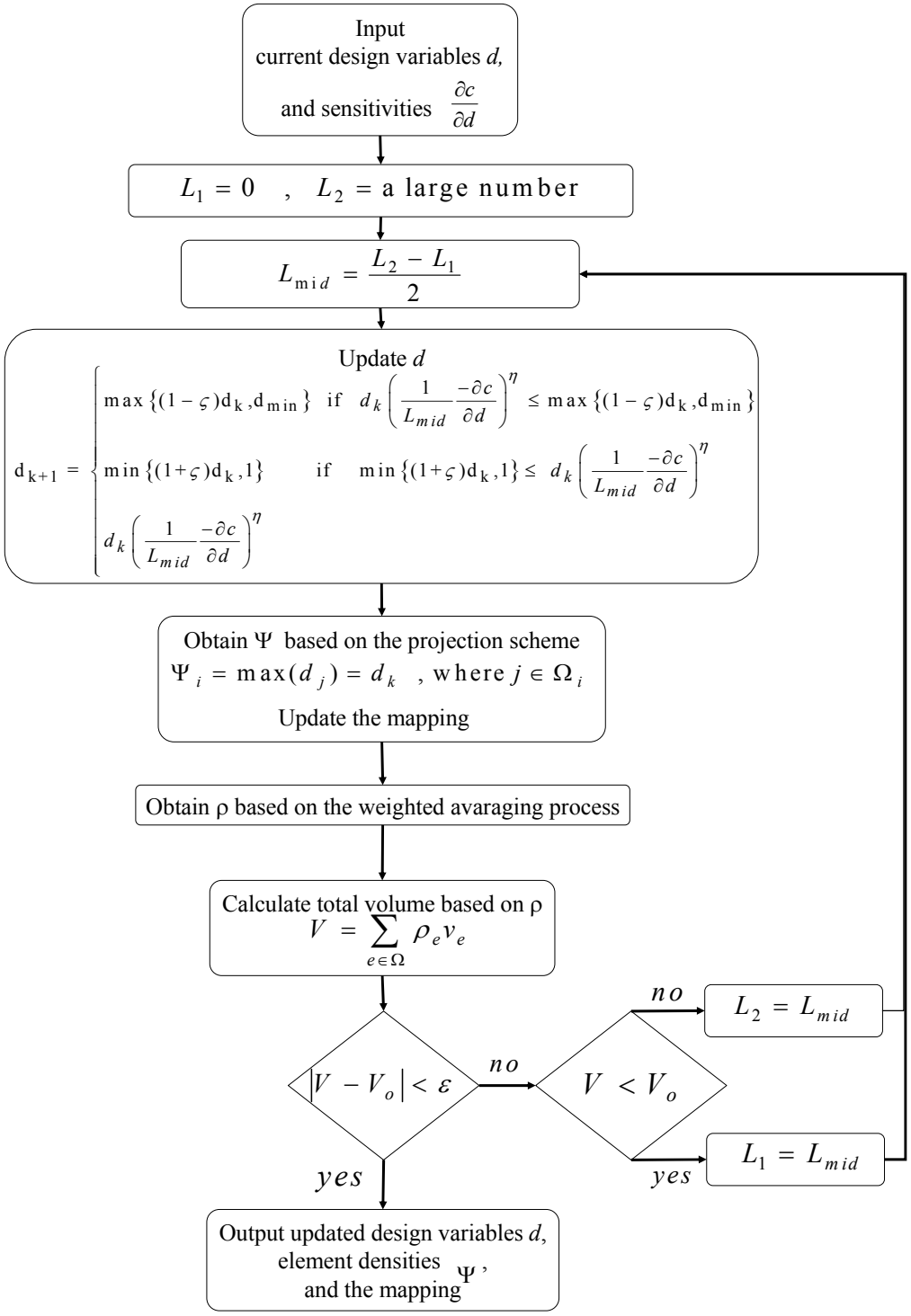


Figure 28 : Flowchart for implementing the new projection scheme with the nodal projection approach: Optimality Criteria (OC) Update.

Chapter 5 : Numerical Examples

For numerical verification, the traditional minimum compliance problem is solved for a cantilever beam. The SIMP model is used for the interpolation of stiffness tensor of intermediate material density. A penalization factor of p=3 is used without continuation. Poisson's ratio is 0.3 for the solid material. Total volume fraction is 50%. In all examples, the finite element mesh resolution is 80x50 elements.

The example below is similar to the one in Figure 5(b) of the paper by Guest *et al.* [27]. The configuration of the beam is illustrated in Figure 29. The problem is solved using each of the three implementations with minimum length scale scheme: element-based, CAMD, and nodal projection. The results are compared with those obtained by the original approaches.

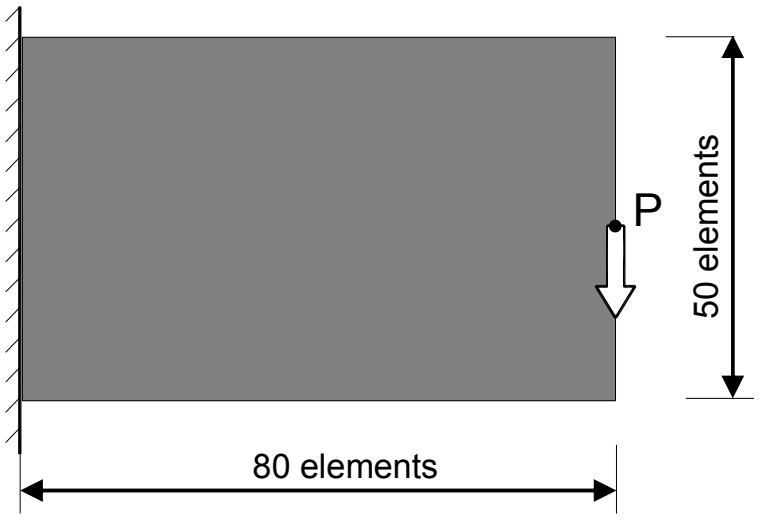
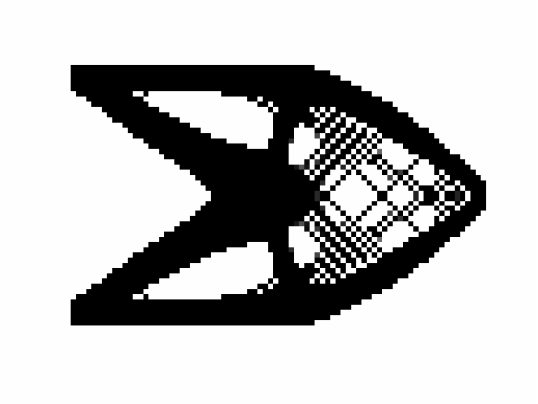


Figure 29 : Configuration of the cantilever beam problem.

5.1 Results obtained using the element-based approach

Figure 30 compares the topologies obtained by using the traditional element-based implementation: the result of Figure 30(a) is obtained without applying the sensitivity filtering technique by Sigmund [20]; the result of Figure 30(b) is obtained using of sensitivity filtering technique with a filtering radius equal 2.5 times element size; the result of Figure 30(c) is obtained using the new scheme to impose a minimum length scale. The length scale parameter is also set to 2.5 times element size. Note that a filtering radius of 2.5 is not exactly equivalent to the length scale parameter of 2.5 imposed by the new scheme. The reason is that the structural members obtained using the filtering technique contain "gray" edges, while the structural members obtained by imposing a minimum length scale with the new scheme are essentially solid. It is clear that without applying filtering or imposing minimum length scale, numerical instabilities (e.g. checkerboard patterns) appear. The filtering technique eliminates checkerboard patterns but also induce gray areas around the edges of structural members. The new scheme also solves the checkerboard problem, guarantees the exact minimum length scale, and does not cause the fading effect as in the case of the filtering technique.



(a) Topology obtained without minimum length scale and without filtering

(b) Topology obtained by applying the filter of sensitivities with $r_{\min} = 2.5$

(c) Topology obtained using the minimum length scale scheme with $r_{\min} = 2.5$

Figure 30 : Results obtained with the element-based approach: mesh size is 80x50 elements; volume fraction is 50%; and Poison's ratio is 0.3.

5.2 Results obtained using CAMD approach

The result obtained by means of the original version of the CAMD approach by Matsui and Terada [28] is compared to the result obtained using the new scheme on top of the CAMD approach (Figure 31). A minimum length scale parameter $r_{\min} = 2.5$ times element size is applied. The topology obtained by the original CAMD without imposing minimum length scale includes very thin structural members and a minor symptom of "layering" phenomenon (which is equivalent to checkerboard instabilities in the element-based approach). In contrast, the topology obtained with the new scheme comprises only structural members that satisfy minimum length scale. Notice the minimum length scale is strictly achieved without any additional fading effect.

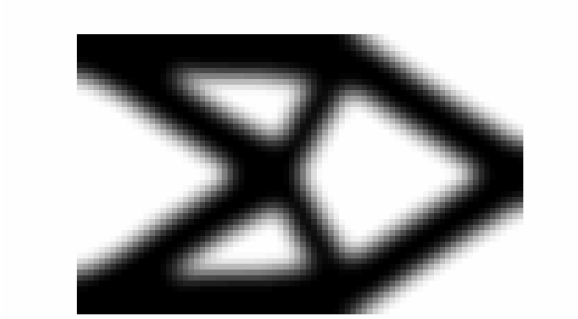
(a) Topology obtained by the original CAMD

(b) Topology obtained using the new scheme with $r_{\min} = 2.5$

Figure 31 : Results obtained with the CAMD approach: mesh size is 80x50 elements; volume fraction is 50%; and Poison's ratio is 0.3.

5.3 Results obtained using nodal projection approach

The result obtained by means of the nodal projection by Guest *et al.* [27] is compared to the result obtained using the new projection scheme on top of the original nodal projection approach with the linear projection (see Figure 32). For the implementation of the new projection scheme, a minimum length scale parameter $r_{\min} = 2.5$ times element size is applied. For the original nodal projection approach, a minimum length scale $r_{\min} = 4.0$ is applied. Note that the minimum length scale in the original nodal projection approach is defined on gray



(a) Topology obtained using the original nodal projection with $r_{\min} = 4.0$

(b) Topology obtained using the new scheme with $r_s = 1$ and $r_\ell = 2.5$

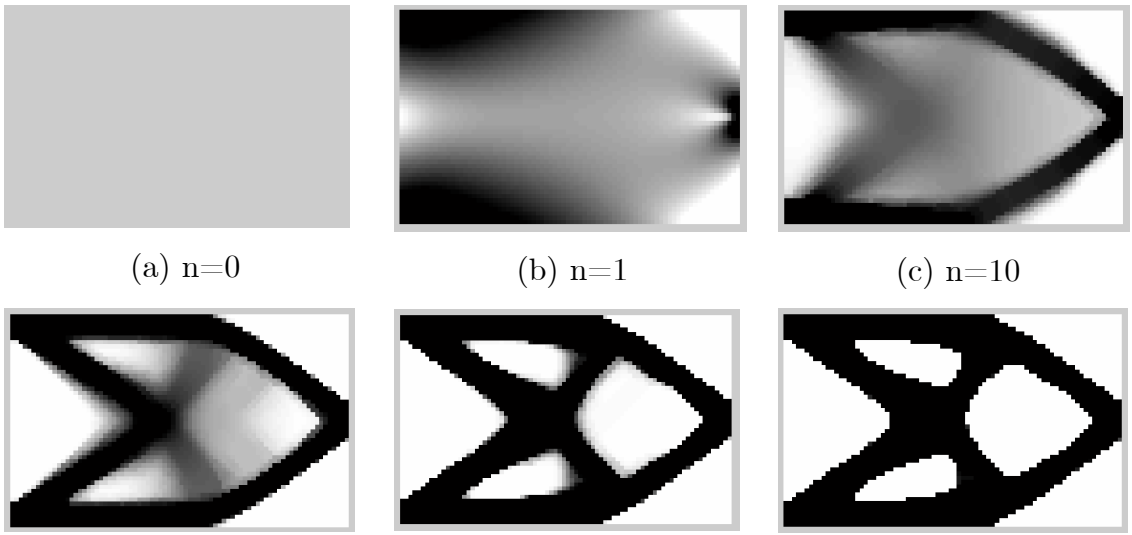
Figure 32 : Results obtained with the nodal projection approach: mesh size is 80x50 elements; volume fraction is 50%; and Poison's ratio is 0.3.

elements. Empirically, we observe that that $r_{\min} = 4$ including gray elements is equivalent to $r_{\min} = 2.5$ on totally solid members. This assumption is validated by the fact that if $r_{\min} = 2.5$ is applied for the original nodal projection approach, a different topology with finer structural members will result. It is clear from the result that the new scheme generates a sharp topology which strictly satisfies the minimum length scale, while the result obtained by the original nodal projection approach contains much undesirable gray areas.

5.4 Design evolution

In order to provide the reader with more information about the convergence process with the new projection scheme, the design evolution is given in Figure 33, Figure 34, and Figure 35. Note that no continuation or filtering technique is involved in the optimization process. The penalization parameter $p = 3$ and the length scale factor $r_{\min} = 2.5$ are set from the beginning and remain constant throughout the optimization process. Poisson's ratio is 0.3 for the solid material. Total volume fraction is 50%. In all examples, the finite element mesh resolution is 80x50 elements. In Figure 33 and Figure 34, notice that smoothly changed topologies are generated in intermediate steps, and sharp topologies are obtained at the end.

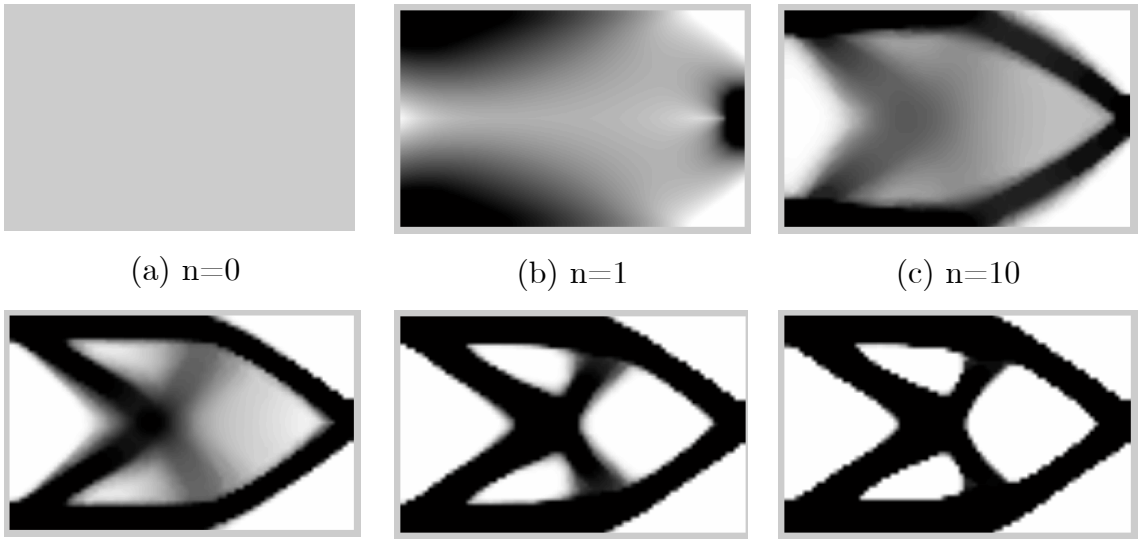
The design evolution for both design variable and topology are shown in Figure 35 considering the new minimum length scale scheme on top of the nodal projection approach. In order to visualize the distribution of design variables, they are treated as density of an area of element size located at the design variable location.



(a) n=0 (b) n=1 (c) n=10

(d) n=20 (e) n=30 (f) final step

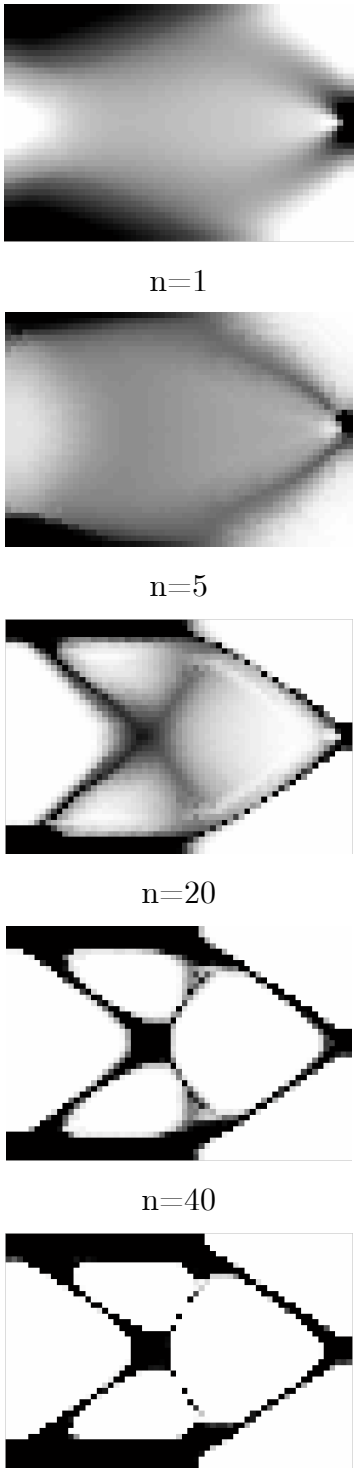
Figure 33 : Design evolutions with the new scheme applied to element based approach (n is the iteration step.)



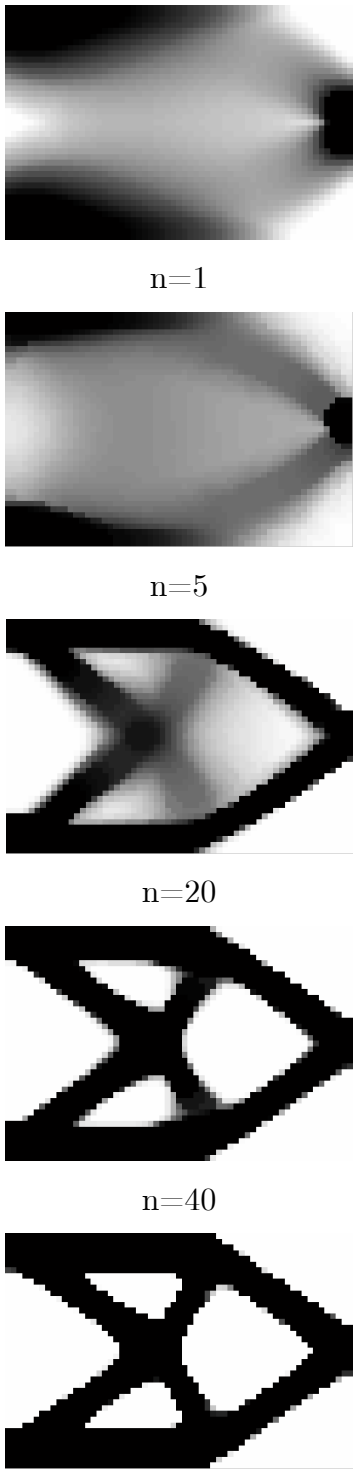
(a) n=0 (b) n=1 (c) n=10

(d) n=20 (e) n=30 (f) n=50

Figure 34 : Design evolutions with the new scheme applied to CAMD approach (n is the iteration step.)



(a) Distribution of design variables

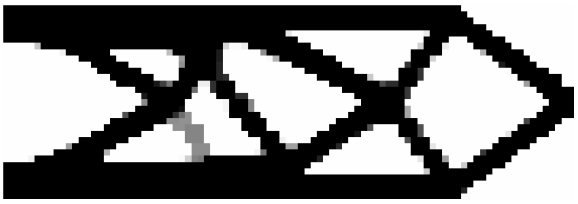


(b) Distribution of element densities

Figure 35 : Design evolutions with the new scheme applied to nodal projection approach (n is the iteration step.)

5.5 Limitation

In case coarse meshes are used to solve sensitive problems, we observe the existence of patches of intermediate density in the final topology. This phenomenon is distinguishable from the fading effect observed with filtering and other techniques. In this case, the optimization is struggling to satisfy the volume constraint while keeping the minimum length scale for structural members. However, the problem vanishes when mesh resolution is sufficiently high. Refer to Figure 36 for more information. This issue is illustrated by Figure 36.



(a) Cantilever beam is solved with 60x20 elements and $r_{\min} = 1.5$: gray structural members are observed.

(b) Cantilever beam is solved with 120x40 elements and $r_{\min} = 3.0$: gray structural members disappear.

Figure 36 : Influence of mesh discretization on the minimum length scale scheme: (a) gray members appear when coarse mesh is used for sensitive problems; (b) finer mesh solves the problem.

Chapter 6 : Exploring additional layers of design variables

The concept of using additional layers of design variables to achieve desired restriction on design space can successfully exclude fine patterns from the feasible design space. The new scheme solves many complications in topology optimization and help achieving the minimum length scale, one of the important manufacturing constraints. However, the concept can be generalized to be the construction of feasible design space from additional design variable layers. It has the potential to help achieving various other important and interesting constraints, such as manufacturing constraints.

6.1 Constructing the feasible design space from layers of design variables

The main part of this thesis describes the new scheme, which introduces an additional layer of design variables and a projection using the maximum function to achieve the minimum length scale for the topology optimization. From a broader point of view, what we do in the new scheme is to construct the feasible design space from an additional layer of design variables. The way we construct the feasible design space using the maximum function excludes undesirable fine patterns.

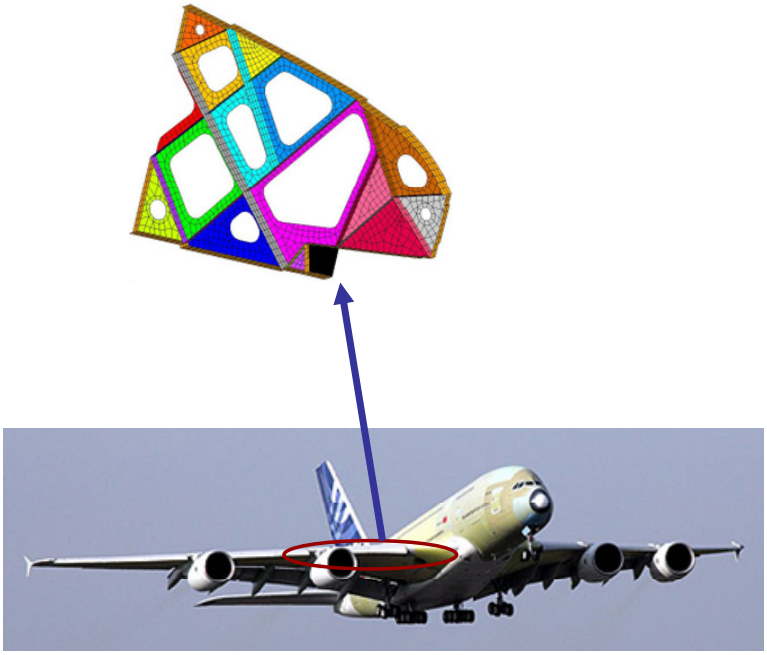
Design space for topology optimization has a unique characteristic, which is not present in classical structural optimization. Element densities do not only have values but also locations. They form a physical space (1D, 2D or 3D) that

can be manipulated. We can construct a feasible design space that does not contain the topological layouts that should not to appear in the solution. The feasible design space should contain all possible layouts other than undesired ones so that the solution will be the true optimum solution. We may use as many additional design variable layers as necessary. Each design variable layer may assume a different domain. Through appropriate choices of domains for design variable layers and the constructing rule (i.e. the projection), one can construct a feasible design space suitable for various purposes. This idea is explored in this Chapter.

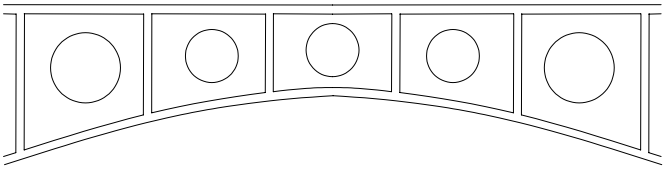
6.2 Design of stiffened plates

Stiffened plates have many applications in the engineering world. Figure 37 shows the applications of stiffened plates to two extreme cases: a high-tech structure of the airframe of a modern airplane; and a massive structure of a civil engineering bridge. To ensure that the results obtained by topology optimization are in the form of stiffened plates is a formidable task. That is why the state-of-the-art application of topology optimization [40] for this type of structure is limited to the two-dimensional imitation of the rib patterns. A real three-dimensional simulation may be achieved using the general concept of constructing feasible design space from additional layers of design variables and projection schemes.

In the stiffener problem, the domain is 3D. The feasible design space is constructed from two layers of design variables named d^w and d^s. The domains of the two layers of design variables are 2D areas corresponding to the mid-plane of the original design domain. The 3D structure in the form of stiffened plates is constructed by the following projection rules, which is defined as follows:



(a) Aircraft frame



(b) Bridge girder

Figure 37 : Applications of stiffened plates (photo in (a) was taken from Altair Engineering.)

$$\rho_i^w = \max_{j \in \Omega_i}(d_j^w, d_j^s)$$
$$\rho_i^s = \max_{j \in \Omega_i}(d_j^s) \tag{41}$$

where ρ_i^w is the density of the web element corresponding to location i of the cross section; ρ_i^s is the density of all stiffener elements (i.e. elements outside of the web) corresponding to location i of the cross section; Ω_i is the subdomain of element i of the cross section, which is essentially a circle with radius $r_{\min}$ (refer to previous sections for more explanations.)

Figure 38 shows schematically the topology constructed using the rule defined in Equation (41) with $r_{\min} = 2.5$ times element size. Notice that the resulting structure has the form of a stiffened plate.

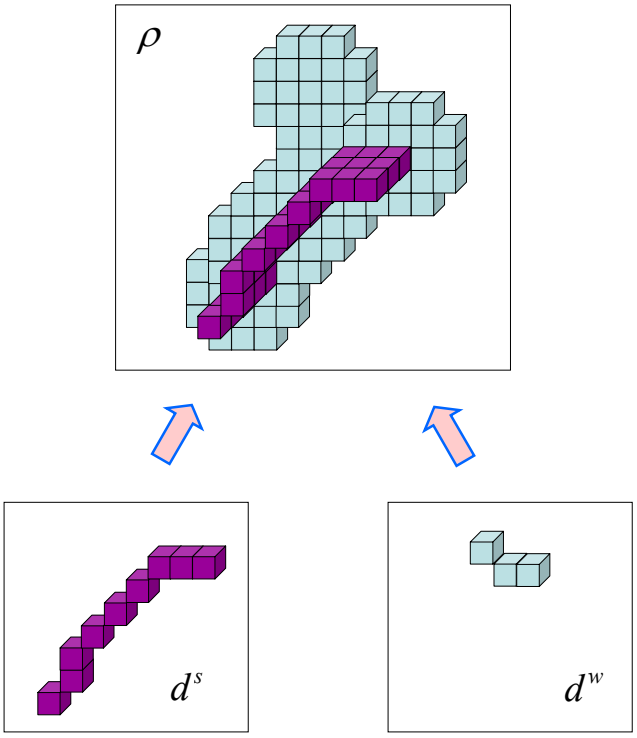
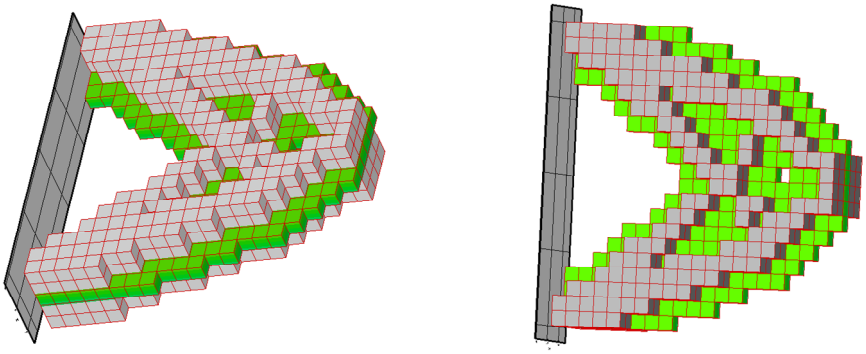
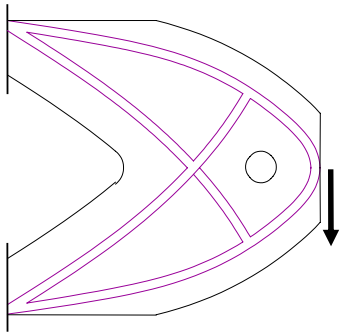


Figure 38 : Constructing the topology of a stiffened plate from two design variable layers.

A preliminary result for the stiffened plate problem is shown in Figure 39 (a). The green elements represent the web and the gray elements represent the stiffeners. Figure 39 (b) shows the interpretation of the result.



(a) Original result



(b) Interpretation of the result

Figure 39 : Preliminary result of the stiffened plate problem: mesh size is 20x20x5; volume fraction is 50%; the thickness of the web (green) is one element; and stiffeners (gray) height is 2 elements.

Chapter 7 : Summary and future work

Complications of topology optimization problems include non-existence, mesh-dependency, and checkerboard instabilities of its solutions. An extensive literature has been dedicated to the study and approaches to address those complications. Moreover, one of the remaining obstacles to topology optimization is manufacturability of its solutions. In this thesis, we review various approaches to solve complications of topology optimization and to achieve manufacturability of the results. Then we present a new approach, which is shown to be more advantageous than existing approaches in achieving minimum length scale and solving complications of topology optimization problems. We also discuss the generalization and extension of the approach to show that it has the potential of improving the quality of topology optimization in various applications where design constraints must be placed. This chapter summarizes the content of the work and provides suggestions for future extensions.

7.1 Summary

This work first provides background information about topology optimization and the complications of the problem. Formulation of the minimum compliance problem is given with two basic design parameterization approaches: the traditional element based; and the continuous approximation of material distribution (CAMD). Then existing methods to solve complications and achieve minimum length scale are reviewed in moderate details. Four methods are

discussed: filter of sensitivities [10, 20, 21], nodal design variables and projection functions [27], perimeter constraint [11, 12, 13], and monotonicity-based length scale [22, 23, 24]. A discussion about the essential points for a successful approach is also provided.

Based on the understanding of the aforementioned essential points, a new scheme is developed to achieve minimum length scale in topology optimization. The new scheme employs the idea of constructing the feasible design space using an additional layer of design variable and the maximum projection function. The projection scheme ensures that all structural members in the solution satisfy a minimum member size, while not inducing undesirable side effects commonly observed in other approaches. The new scheme is implemented on top of three basic implementations: the traditional element based; the CAMD; and the nodal projection.

The idea of constructing feasible design space is also generalized to the construction of feasible design space with more than one layer of design variables to achieve fabrication and design constraints other than the minimum member size. A projection scheme to achieve stiffened plate patterns is given to demonstrate the idea the potential of the approach in practical applications. Further unexplored areas are discussed in the following section.

7.2 Suggestions for future work

Two main areas for future works are:

- development of tailored schemes to achieve various types of design and fabrication constraints
- alternative smoothening techniques to augment the maximum projection scheme and enhance convergence to global optima.

In the first area of extension, there are many design and fabrication constraints that arise from practical applications. We have addressed two of them: the minimum length scale; and the stiffener pattern. We also have an

ongoing work, which employs the projection scheme to constrain the maximum material gradient in the design of functionally graded microstructures (material design). Other fabrication constraints include the following:

- Casting constraint such as mold removing direction, and extrusion
- Repetitive patterns, for example, to ensure the similarity of products
- Maximum member size
- Desired patterns to account for other design aspects

The method to achieve those constraints may include the choice of appropriate design variable spaces, the combination of those spaces using the maximum projection, and the exploration of other constructing functions.

The second extension work is intended to improve the performance of the new projection scheme. The desired integer (0-1) solution for topology optimization makes the problem very sensitive to local optima. Once the densities are updated to either zero or one, they are usually trapped to those values and prevent the structure to evolve to global optima. A method, such as the filtering of sensitivities, provides smoothening effects besides the minimum length scale. However, the new projection scheme proposed in this work does not provide any smoothening effect during the iterative solution process. Thus, for problems where the topology changes too much during the optimization evolution, the solution may be far from global optimum. The smoothening effect may be in the form of mathematical regularization or any scheme that facilitates the change in location of structural members after they are formed during the design evolution.

Appendices

Matlab codes of the implementation with the element-based, CAMD and nodal projection approach.

Appendix A

Matlab code for the implementation with element-based approach

```
% Topology Optimization Program %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   With minimum length scale achieved by the new method (maximum projection)
%   Implemented with element based approach
%   Usage: file_name(nelx,nely,volfrac,penal1,penal2,rmin)
%   nelx: number of elements in x direction
%   nely: number of elements in y direction
%   volfrac: averaged volume fraction
%   penal: penalization parameter
%   rmin: lentgh scale parameter (half of the minimum member width)
%
% Selected List of variables:%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   xd: design variables, matrix nely rows and nelx columns
%   xe: element densities, matrix nely rows and nelx columns
%   dc_xd: Sensitivities of the compiance w.r.t. design variables
%   dc_xe: Sensitivities of the compiance w.r.t. element densities
%   loop: iteration number
%   change: maximum change of element densities between two consecutive steps
%   ne: number of elements belonging to subdomain of each element, vector nelx
%       *nely rows
%   elemlistx: list of column numbers of elements belonging to subdomain of
%      each element, matrix nelx*nely rows, number of columns depends on rmin
%   elemlisty: list of row numbers of elements belonging to subdomain of each
%           element, matrix nelx*nely rows, number of columns depends on rmin
%   mapx: list of column number of the design variable corresponding to each
%         element density, matrix nely rows and nelx column
%   mapy: list of row number of the design variable corresponding to each
%         element density, matrix nely rows and nely column
%   KE: element stiffness matrix of solid material
%   c: objective function which is the compliance in this case
%   Ue: element displacement, vector
%
% List of sub-functions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
033 %   FE: sub-function to caryout finite element analysis, returns displacement
034 %   lk: sub-function to calculate element stiffness matrix of solid material
035 %   OC: sub-function to update design variable and calculate material densities
036 %       from design
037 %       variables, this is a revised version of the OC from Sigmund's code
038 %
039 function lengthscale_element_based(nelx,nely,volfrac,penal,rmin);
040 % Initialize %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
041 xd(1:nely,1:nelx) = volfrac;
042 xe(1:nely,1:nelx) = volfrac;
043 dc_xd=zeros(nely,nelx);
044 dc_xe=zeros(nely,nelx);
045 loop = 0;
046 change = 1.;
047 % Build the list of adjacent elements (with radius rmin)%%%%%%%%%%%%%%%%%%%%%%
048 ne=zeros(nely*nelx,1);
049 elemlistx=zeros(nely*nelx,(2*round(rmin)+1)^2);
050 elemlisty=zeros(nely*nelx,(2*round(rmin)+1)^2);
051 for ely=1:nely
052    for elx=1:nelx
053       % Search a rectangular area only %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
054       elemnumber=nely*(elx-1)+ely;
055       for nly=max((ely-floor(rmin)),1):min((ely+floor(rmin)),nely)
056          for nlx=max((elx-floor(rmin)),1):min((elx+floor(rmin)),nelx)
057             r=sqrt((elx-nlx)^2+(ely-nly)^2);
058             if r<=rmin
059                ne(elemnumber,1)=ne(elemnumber,1)+1;
060                elemlistx(elemnumber,ne(elemnumber,1))=nlx;
061                elemlisty(elemnumber,ne(elemnumber,1))=nly;
062             end
063          end
064       end
065    end
066 end
067 % Initialize the map from element densities to design variables %%%%%%%%%%%%%%%
068 for ely=1:nely
069    for elx=1:nelx
070       mapy(ely,elx)=ely;
071       mapx(ely,elx)=elx;
072    end
073 end
074 % Load element stiffness matrix for solid material %%%%%%%%%%%%%%%%%%%%%%%%%%%
075 [KE] = lk;
076 %
077 % Start iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
078 while change > 0.01
079    loop = loop + 1;
080    % FE- analysis %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
081    [U]=FE(nelx,nely,xe,penal);
082    % Calculate compliance and sensitivities w.r.t. element densities %%%%%%%%%%
083    c = 0.;
084    for ely = 1:nely
085       for elx = 1:nelx
086          n1 = (nely+1)*(elx-1)+ely;
```

```
087           n2 = (nely+1)* elx   +ely;
088           Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*n1+1;2*n1+2],1);
089           c= c + xe(ely,elx)^penal*Ue'*KE*Ue;
090           dc_xe(ely,elx)=penal*xe(ely,elx)^(penal-1)*Ue'*KE*Ue;
091        end
092     end
093     % Calculate sensitivities w.r.t. design variables %%%%%%%%%%%%%%%%%%%%%%%%%%
094     tempdc=zeros(nely,nelx);
095     nan=zeros(nely,nelx);
096     for ely = 1:nely
097        for elx = 1:nelx
098           tempdc(mapy(ely,elx),mapx(ely,elx))=tempdc(mapy(ely,elx), ...
099              mapx(ely,elx))-dc_xe(ely,elx);
100           nan(mapy(ely,elx),mapx(ely,elx))=nan(mapy(ely,elx),mapx(ely,elx))+1;
101        end
102     end
103     % Normalize the sensitivities by dividing by the influenced volume %%%%%%%%
104     for ely = 1:nely
105        for elx = 1:nelx
106           if nan(ely,elx)>0
107              dc_xd(ely,elx)=tempdc(ely,elx)/nan(ely,elx);
108           else
109              dc_xd(ely,elx)=-dc_xe(ely,elx);
110           end
111        end
112     end
113     % Update design variables with the revised Optimality Criteria sub-function
114     xdold=xd; xeold=xe;
115     [xd,xe,mapy,mapx]  = OC(nelx,nely,xd,volfrac,dc_xd,ne,elemlistx,elemlisty);
116     % Print results %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
117     change = max(max(abs(xd-xdold)));
118     disp([' It.: ' sprintf('%4i',loop) ' Obj.: ' sprintf('%10.4f',c) ...
119           ' Vol.: ' sprintf('%6.3f',sum(sum(xe))/(nelx*nely)) ...
120           ' ch.: ' sprintf('%6.3f',change )])
121     % Plot element densities %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
122     colormap(gray); imagesc(-xe); axis equal; axis tight; axis off;pause(1e-6);
123     % Recording pictures for animations if required
124     %  picturename=strcat('temp',sprintf('%0i',loop),'.tif');
125     %  pic= getframe(gcf);
126     %  imwrite(pic.cdata, picturename);
127  end
128  %
129  % SUB FUNCTIONS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
130  %%%%%%%%%% OPTIMALITY CRITERIA UPDATE - revised %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
131  % Input variables:
132  %   nelx: number of element in x direction
133  %   nely: number of element in y direction
134  %   xd: current design variables, matrix nely rows and nelx columns
135  %   volfrac: volume fraction
136  %   dc_xd: sensitivities w.r.t. design variables
137  %   ne: number of elements belonging to subdomain of each element, vector
138  %       nelx*nely rows
139  %   elemlistx: list of column numbers of elements belonging to subdomain of
140  %       each element, matrix nelx*nely rows, number of columns depends on rmin
```

```
141 %   elemlisty: list of row numbers of elements belonging to subdomain of each
142 %              element, matrix nelx*nely
143 % Output variables:
144 %   xdnew: updated design variables, matrix nely rows and nelx columns
145 %   xenew: updated element densities, matrix nely rows and nelx columns
146 %   mapx: updated list of column number of the design variable corresponding to
147 %         each element density, matrix nely rows and nelx column
148 %   mapy: updated list of row number of the design variable corresponding to
149 %         each element density, matrix nely rows and nelx column
150 % List of selected local variables
151 %   maxnodey: store the row number of the maximum design variables in range
152 %             while doing the search for maximum design variable for an
153 %             element density
154 %   maxnodex: store the column number of the maximum design variables in
155 %             range while doing the search for maximum design variable for
156 %             an element density
157 %   l1, l2, lmid: temporary values of the Lagrangian multipliers used by the
158 %                 bi-sectioning algorithim
159 function [xdnew,xenew,mapy,mapx]= ...
160    OC(nelx,nely,xd,volfrac,dc_xd,ne, elemlistx,elemlisty)
161 l1 = 0; l2 = 100000; move = 0.2;
162 while (l2-l1 > 1e-4)
163    mapy=zeros(nely,nelx);
164    mapx=zeros(nely,nelx);
165    lmid = 0.5*(l2+l1);
166    xdnew = ...
167       max(0.001,max(xd-move,min(1.,min(xd+move,xd.*(-dc_xd./lmid).^0.25))));
168    volume=0;
169    xenew=zeros(nely,nelx);
170    % loop all elements in the design domain
171    for elx=1:nelx
172       for ely=1:nely
173          elemnumber=nely*(elx-1)+ely;
174          % loop all element in the sub-domain of each element
175          for i=1:ne(elemnumber)
176             if xenew(ely,elx) < ...
177                (xdnew(elemlisty(elemnumber,i), elemlistx(elemnumber,i))-1.e-5)
178                xenew(ely,elx) = ...
179                xdnew(elemlisty(elemnumber,i), elemlistx(elemnumber,i));
180                maxnodey=elemlisty(elemnumber,i);
181                maxnodex=elemlistx(elemnumber,i);
182             end
183          end
184          mapy(ely,elx)=maxnodey; mapx(ely,elx)=maxnodex;
185       end
186    end
187    if sum(sum(xenew)) - volfrac*nelx*nely > 0
188       l1 = lmid;
189    else
190       l2 = lmid;
191    end
192 end
193 %
194 % FE-ANALYSIS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Input variables:
%   nelx: number of element in x direction
%   nely: number of element in y direction
%   x: element densities, matrix nely rows and nelx columns
%   penal: penalization parameter
% Output variables:
%   U: Displacement vector
% List of selected local variables
%   KE: element stiffness matrix of solid material
%   K: global stiffness matrix
%   F: global load vector
%   edof: element degrees of freedom
%   fixeddofs: prescribed dofs, vector
%   alldofs: all dofs, vector
%   freedofs: free dofs, vector
function [U]=FE(nelx,nely,x,penal)
[KE] = lk;
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U = zeros(2*(nely+1)*(nelx+1),1);
for elx = 1:nelx
   for ely = 1:nely
      n1 = (nely+1)*(elx-1)+ely;
      n2 = (nely+1)* elx   +ely;
      edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1; 2*n2+2; 2*n1+1; 2*n1+2];
      K(edof,edof) = K(edof,edof) + x(ely,elx)^penal*KE;
   end
end
% Define loads and supports %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Load and bc condition is for a cantilever beam fixed at left end
% Subject to a vertical load at the middle of the right edge
F(((nely+1)*nelx+nely/2+1)*2,1) = 1;
fixeddofs   = [1:2*(nely+1),1];
alldofs     = [1:2*(nely+1)*(nelx+1)];
freedofs    = setdiff(alldofs,fixeddofs);
% Solve the linear system %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
%
% ELEMENT STIFFNESS MATRIX %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [KE]=lk
E = 1.;
nu = 0.3;
k=[ 1/2-nu/6   1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
      -1/4+nu/12 -1/8-nu/8  nu/6       1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
   k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
   k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
   k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
   k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
   k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
   k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
   k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
```

Appendix B

Matlab code for the implementation with CAMD approach

```
% Topology Optimization Program %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   With minimum length scale achieved by the new method (maximum projection)
%   Implemented with the CAMD approach
%   Usage: file_name(nelx,nely,volfrac,penal1,penal2,rmin)
%   nelx: number of elements in x direction
%   nely: number of elements in y direction
%   volfrac: averaged volume fraction
%   penal: starting penalization parameter
%   rmin: lentgh scale parameter (half of the minimum member width)
%
% Selected List of variables:%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   ndx: number of nodes in x direction
%   ndy: number of node in y direction
%   xd: design variables, matrix ndy rows and ndx columns
%   xn: nodal densities, matrix ndy rows and ndx columns
%   dv: derivative of total volume w.r.t nodal densities
%   dc_xd: Sensitivities of the compiance w.r.t. design variables
%   dc_xn: Sensitivities of the compiance w.r.t. nodal densities
%   loop: iteration number
%   change: maximum change of element densities between two consecutive steps
%   nn: number of nodes belonging to subdomain of each node, vector ndx
%       , ndy rows
%   nodelistx: list of column numbers of nodes belonging to subdomain of
%             each node, matrix ndx*ndy rows, number of columns depends on rmin
%   nodelisty: list of row numbers of nodes belonging to subdomain of each
%           node, matrix ndx*ndy rows, number of columns depends on rmin
%   mapx: list of column number of the design variable corresponding to each
%         nodal density, matrix nely rows and ndx column
%   mapy: list of row number of the design variable corresponding to each
%         element density, matrix nely rows and ndy column
%   a, b, c, and d: values of shape functions at four Gauss points
%   K1, K2, K3, and K4: Integrand B'DB at four Gauss points for solid material
%   dKEr1, dKEr2, dKEr3, dKEr4: Derivative of the integrand B'DB w.r.t nodal
%         densities evaluated at four Gauss points
%   c: objective function which is the compliance in this case
%   Ue: element displacement, vector
%
% List of sub-functions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   FE: sub-function to caryout finite element analysis, returns displacement
%   lk: sub-function to calculate element stiffness matrix of solid material
%   OC: sub-function to update design variable and calculate material densities
%       from design
%       variables, this is a revised version of the OC from Sigmund's code
```

```
044
045 function lengthscale_CAMD(nelx,nely,volfrac,penal,rmin);
046 % Initialize %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
047 xd(1:(nely+1),1:(nelx+1)) = volfrac;
048 xn(1:nely+1,1:nelx+1) = volfrac;
049 dc_xd=zeros(nely+1,nelx+1);
050 dc_xn=zeros(nely+1,nelx+1);
051 dv=zeros((nely+1),(nelx+1));
052 loop = 0;
053 change = 1.;
054 % Build the list of adjacent nodes (with radius rmin)%%%%%%%%%%%%%%%%%%%%%%%%%
055 nn=zeros((nely+1)*(nelx+1),1);
056 nodelistx=zeros((nely+1)*(nelx+1),(2*round(rmin)+1)^2);
057 nodelisty=zeros((nely+1)*(nelx+1),(2*round(rmin)+1)^2);
058 ndy=nely+1;
059 ndx=nelx+1;
060 for cny=1:ndy
061    for cnx=1:ndx
062       %search a rectangular area only %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
063       nodenumber=ndy*(cnx-1)+cny;
064       for nly=max((cny-floor(rmin)),1):min((cny+floor(rmin)),ndy)
065          for nlx=max((cnx-floor(rmin)),1):min((cnx+floor(rmin)),ndx)
066             r=sqrt((cnx-nlx)^2+(cny-nly)^2);
067             if r<=rmin
068                nn(nodenumber,1)=nn(nodenumber,1)+1;
069                nodelistx(nodenumber,nn(nodenumber,1))=nlx;
070                nodelisty(nodenumber,nn(nodenumber,1))=nly;
071             end
072          end
073       end
074    end
075 end
076 % Initialize the map from nodal densities to design variables %%%%%%%%%%%%%%%%
077 for cny=1:ndy
078    for cnx=1:ndx
079       mapy(cny,cnx)=cny;
080       mapx(cny,cnx)=cnx;
081    end
082 end
083 % Sensitivities of constraint function V with respect to nodal densities %%%%%
084 for ely = 1:nely
085    for elx = 1:nelx
086       yn1 = ely; xn1=elx;
087       yn2 = ely; xn2=elx+1;
088       yn3 = ely+1; xn3=elx+1;
089       yn4 = ely+1; xn4=elx;
090       dv(yn1,xn1)=dv(yn1,xn1)+0.25;
091       dv(yn2,xn2)=dv(yn2,xn2)+0.25;
092       dv(yn3,xn3)=dv(yn3,xn3)+0.25;
093       dv(yn4,xn4)=dv(yn4,xn4)+0.25;
094    end
095 end
096 % Calculate B'*D*B at four Gauss points %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
097 [K1,K2,K3,K4] = lk;
```

```
098 % Values of shape functions
099 a=1/4*(1+1/sqrt(3))^2;
100 b=1/6;
101 c=1/4*(1-1/sqrt(3))^2;
102 d=1/6;
103 %
104 % Start iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
105 while (change > 0.01) | (penal <= penal2)
106    loop = loop + 1;
107    % FE- analysis %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
108    [U,K]=FE(nelx,nely,xn,penal);
109    % Calculate compliance and sensitivities w.r.t. nodal densities %%%%%%%%%%%%
110    dc_xn=zeros(nely+1,nelx+1);
111    c=0;
112    for ely = 1:nely
113       for elx = 1:nelx
114          yn1 = ely; xn1=elx;
115          yn2 = ely; xn2=elx+1;
116          yn3 = ely+1; xn3=elx+1;
117          yn4 = ely+1; xn4=elx;
118          KE=K1*([a d c b]* ...
119             [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal+ ...
120             K2*([b a d c]* ...
121             [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal+ ...
122             K3*([c b a d]* ...
123             [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal + ...
124             K4*([d c b a]* ...
125             [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal;
126          dKEr1=K1*([a d c b]*[xn(yn1,xn1); ...
127                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*a + ...
128             K2*([b a d c]*[xn(yn1,xn1); ...
129                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*b + ...
130             K3*([c b a d]*[xn(yn1,xn1); ...
131                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*c + ...
132             K4*([d c b a]*[xn(yn1,xn1); ...
133                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*d;
134          dKEr2=K1*([a d c b]*[xn(yn1,xn1); ...
135                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*d + ...
136             K2*([b a d c]*[xn(yn1,xn1); ...
137                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*a + ...
138             K3*([c b a d]*[xn(yn1,xn1); ...
139                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*b + ...
140             K4*([d c b a]*[xn(yn1,xn1); ...
141                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*c;
142          dKEr3=K1*([a d c b]*[xn(yn1,xn1); ...
143                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*c + ...
144             K2*([b a d c]*[xn(yn1,xn1); ...
145                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*d + ...
146             K3*([c b a d]*[xn(yn1,xn1); ...
147                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*a + ...
148             K4*([d c b a]*[xn(yn1,xn1); ...
149                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*b;
150          dKEr4=K1*([a d c b]*[xn(yn1,xn1); ...
151                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*b + ...
```

```
152             K2*([b a d c]*[xn(yn1,xn1); ...
153                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*c + ...
154             K3*([c b a d]*[xn(yn1,xn1); ...
155                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*d + ...
156             K4*([d c b a]*[xn(yn1,xn1); ...
157                xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^(penal-1)*penal*a;
158          n1 = (nely+1)*(elx-1)+ely;
159          n2 = (nely+1)* elx   +ely;
160          Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*n1+1;2*n1+2],1);
161          c= c- Ue'*KE*Ue;
162          dc_xn(yn1,xn1)=dc_xn(yn1,xn1)-Ue'*dKEr1*Ue;
163          dc_xn(yn2,xn2)=dc_xn(yn2,xn2)-Ue'*dKEr2*Ue;
164          dc_xn(yn3,xn3)=dc_xn(yn3,xn3)-Ue'*dKEr3*Ue;
165          dc_xn(yn4,xn4)=dc_xn(yn4,xn4)-Ue'*dKEr4*Ue;
166       end
167    end
168    % Normalize the sensitivities w.r.t. nodal densities %%%%%%%%%%%%%%%%%%%%%%
169    dc_xn=dc_xn./dv;
170    % Calculate sensitivities w.r.t. design variables %%%%%%%%%%%%%%%%%%%%%%%%
171    tempdc=zeros(nely+1,nelx+1);
172    nan=zeros(nely+1,nelx+1);
173    for cny = 1:ndy
174       for cnx = 1:ndx
175          tempdc(mapy(cny,cnx),mapx(cny,cnx))= ...
176             tempdc(mapy(cny,cnx),mapx(cny,cnx)) -dc_xn(cny,cnx);
177          nan(mapy(cny,cnx),mapx(cny,cnx))= ...
178             nan(mapy(cny,cnx),mapx(cny,cnx))+1;
179       end
180    end
181    % Normalize the sensitivities w.r.t. design variables %%%%%%%%%%%%%%%%%%%%%
182    for cny = 1:ndy
183       for cnx = 1:ndx
184          if nan(cny,cnx)>0
185             dc_xd(cny,cnx)=-tempdc(cny,cnx)/nan(cny,cnx);
186          else
187             dc_xd(cny,cnx)=dc_xn(cny,cnx);
188          end
189       end
190    end
191    % Update design variables with the revised Optimality Criteria sub-function
192    xdold=xd;xnold=xn;
193    [xd,xn,mapy,mapx] = ...
194       OC(ndx,ndy,xd,volfrac,dc_xd,dv,nn,nodelistx,nodelisty);
195    %Calculate xe (densities inside element) based on xn (nodal volume fraction
196    %according to two illustration scheme: avaraged and continuous
197    nofpixel=min(floor(700/nelx),floor(400/nely));
198    xecontinuous=zeros(nofpixel*nely,nofpixel*nelx);
199    for ely=1:nely
200       for elx=1:nelx
201          yn1 = ely; xn1=elx;
202          yn2 = ely; xn2=elx+1;
203          yn3 = ely+1; xn3=elx+1;
204          yn4 = ely+1; xn4=elx;
205          xe(ely,elx)=(xn(yn1,xn1)+xn(yn2,xn2)+xn(yn3,xn3)+xn(yn4,xn4))/4;
```

```
206           for i=1:nofpixel
207              for j=1:nofpixel
208                 tx=(j-nofpixel/2)/nofpixel*2; ty=(nofpixel/2-i)/nofpixel*2;
209                 xecontinuous((ely-1)*nofpixel+i,(elx-1)*nofpixel+j)= ...
210                    xn(yn1,xn1)*1/4*(1-tx)*(1+ty) + ...
211                    xn(yn2,xn2)*1/4*(1+tx)*(1+ty) + ...
212                    xn(yn3,xn3)*1/4*(1+tx)*(1-ty) + ...
213                    xn(yn4,xn4)*1/4*(1-tx)*(1-ty);
214              end
215           end
216        end
217     end
218     % Print results %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
219     change = max(max(abs(xd-xdold)));
220     disp([' It.: ' sprintf('%4i',loop) ' Obj.: ' sprintf('%10.4f',-c) ...
221           ' Vol.: ' sprintf('%6.3f',sum(sum(xe))/(nelx*nely)) ...
222           ' ch.: ' sprintf('%6.3f',change )])
223     % Plot element densities %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
224     colormap(gray);
225     %imagesc(-xe); axis equal; axis tight; axis off;pause(1e-6);
226     imagesc(-xecontinuous); axis equal; axis tight; axis off;pause(1e-6);
227  end
228  %
229  % SUB FUNCTIONS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
230  %%%%%%%%%% OPTIMALITY CRITERIA UPDATE - revised %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
231  % Input variables:
232  %   ndx: number of nodes in x direction
233  %   ndy: number of nodes in y direction
234  %   xd: current design variables, matrix ndy rows and ndx columns
235  %   volfrac: volume fraction
236  %   dc_xd: sensitivities w.r.t. design variables, matrix ndy rows and ndx cols
237  %   dv: derivative of total volume w.r.t nodal densities
238  %   nn: number of nodes belonging to subdomain of each node, vector
239  %       ndx*ndy rows
240  %   nodelistx: list of column numbers of nodes belonging to subdomain of
241  %       each node, matrix ndx*ndy rows, number of columns depends on rmin
242  %   nodelisty: list of row numbers of nodes belonging to subdomain of each
243  %              node, matrix ndx*ndy
244  % Output variables:
245  %   xdnew: updated design variables, matrix ndy rows and ndx columns
246  %   xnnew: updated nodal densities, matrix ndy rows and ndx columns
247  %   mapx: updated list of column number of the design variable corresponding to
248  %         each nodal density, matrix ndy rows and ndx column
249  %   mapy: updated list of row number of the design variable corresponding to
250  %         each nodal density, matrix ndy rows and ndx column
251  % List of selected local variables
252  %   maxnodey: store the row number of the maximum design variables in range
253  %             while doing the search for maximum design variable for an
254  %             nodal density
255  %   maxnodex: store the column number of the maximum design variables in
256  %             range while doing the search for maximum design variable for
257  %             an nodal density
258  %   l1, l2, lmid: temporary values of the Lagrangian multipliers used by the
259  %                 bi-sectioning algorithim
```

```
function [xdnew,xnnew,mapy,mapx]= ...
   OC(ndx,ndy,xd,volfrac,dc_xd,dv,nn,nodelistx,nodelisty)
l1 = 0; l2 = 100000; move = 0.2;
while (l2-l1 > 1e-4)
   mapy=zeros(ndy,ndx);
   mapx=zeros(ndy,ndx);
   lmid = 0.5*(l2+l1);
   xdnew = max(0.001,max(xd-move,min(1.,min(xd+move,xd.*(-dc_xd./lmid).^0.25))));
   xnnew=zeros(ndy,ndx);
   % loop all nodes in the design domain %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   for cnx=1:ndx
      for cny=1:ndy
         nodenum=ndy*(cnx-1)+cny;
         % loop all nodes in the sub-domain of each node %%%%%%%%%%%%%%%%%%%%%
         for i=1:nn(nodenum)
            if xnnew(cny,cnx)<xdnew(nodelisty(nodenum,i),nodelistx(nodenum,i))
               xnnew(cny,cnx)=xdnew(nodelisty(nodenum,i),nodelistx(nodenum,i));
               maxnodey=nodelisty(nodenum,i); maxnodex=nodelistx(nodenum,i);
            end
         end
         mapy(cny,cnx)=maxnodey; mapx(cny,cnx)=maxnodex;
      end
   end
   if sum(sum(xnnew.*dv)) - volfrac*(ndx-1)*(ndy-1) > 0
      l1 = lmid;
   else
      l2 = lmid;
   end
end
%
% FE-ANALYSIS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Input variables:
%   nelx: number of element in x direction
%   nely: number of element in y direction
%   x: element densities, matrix nely rows and nelx columns
%   penal: penalization parameter
% Output variables:
%   U: Displacement vector
% List of selected local variables
%   a, b, c, and d: values of shape functions at four Gauss points
%   K1, K2, K3, and K4: Integrand B'DB at four Gauss points for solid material
%   K: global stiffness matrix
%   F: global load vector
%   edof: element degreed of freedom
%   fixeddofs: prescribed dofs, vector
%   alldofs: all dofs, vector
%   freedofs: free dofs, vector
function [U,K]=FE(nelx,nely,xn,penal)
[K1,K2,K3,K4] = lk;
a=1/4*(1+1/sqrt(3))^2; b=1/6; c=1/4*(1-1/sqrt(3))^2; d=1/6;
K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
F = sparse(2*(nely+1)*(nelx+1),1); U = zeros(2*(nely+1)*(nelx+1),1);
for elx = 1:nelx
   for ely = 1:nely
```

```
      yn1 = ely; xn1=elx;
      yn2 = ely; xn2=elx+1;
      yn3 = ely+1; xn3=elx+1;
      yn4 = ely+1; xn4=elx;
      KE=K1*([a d c b]* ...
         [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal + ...
         K2*([b a d c]* ...
         [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal + ...
         K3*([c b a d]* ...
         [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal + ...
         K4*([d c b a]* ...
         [xn(yn1,xn1);xn(yn2,xn2);xn(yn3,xn3);xn(yn4,xn4)])^penal;
      n1 = (nely+1)*(elx-1)+ely;
      n2 = (nely+1)* elx   +ely;
      edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1; 2*n2+2; 2*n1+1; 2*n1+2];
      K(edof,edof) = K(edof,edof) + KE;
   end
end
% Define loads and supports %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Load and bc condition is for a cantilever beam fixed at left end
% Subject to a vertical load at the middle of the right edge
F(((nely+1)*nelx+nely/2+1)*2,1) = 1;
fixeddofs   = [1:2*(nely+1),1];
alldofs     = [1:2*(nely+1)*(nelx+1)];
freedofs    = setdiff(alldofs,fixeddofs);
% Solve the linear system
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
%
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [K1,K2,K3,K4]=lk
E = 1.;
nu = 0.3;
% In this place are expressions for K1, K2, K3, and K4, which are the B'*D*B
% at GPs. The Matlab script below to generates the matrices
```

Matlab script to generate K1, K2, K3, and K4 used for the CAMD code

```
% This script computes symbolically K1, K2, K3 & K4 used in the CAMD code %%%%
syms K K1 K2 K3 K4 B x n ro1 ro2 ro3 ro4 nu p
%shape functions
N1=1/4*(1-x)*(1+n);
N2=1/4*(1+x)*(1+n);
N3=1/4*(1+x)*(1-n);
N4=1/4*(1-x)*(1-n);
B=[diff(N1,x) 0 diff(N2,x) 0 diff(N3,x) 0 diff(N4,x) 0;
   0 diff(N1,n) 0 diff(N2,n) 0 diff(N3,n) 0 diff(N4,n);
   diff(N1,n) diff(N1,x) diff(N2,n) diff(N2,x) diff(N3,n) ...
      diff(N3,x) diff(N4,n) diff(N4,x)];
E=[1 nu 0;
```

```
013          nu 1 0;
014          0 0 0.5*(1-nu)];
015  % Ks will be multiplied by Elastic/(1-nu^2) in the CAMD code %%%%%%%%%%%%%%%%%%
016  x=-1/1.7321;n=1/1.7321;
017  K1=eval(B'*E*B);
018  x=1/1.7321;n=1/1.7321;
019  K2=eval(B'*E*B);
020  x=1/1.7321;n=-1/1.7321;
021  K3=eval(B'*E*B);
022  x=-1/1.7321;n=-1/1.7321;
023  K4=eval(B'*E*B);
024  %Output to screen
025  K1, K2, K3, K4
```

Appendix C

Matlab code for the implementation with nodal projection approach

```
% Topology Optimization Program %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   With minimum length scale achieved by the new method (maximum projection)
%   Implemented with nodal projection approach
%   Usage: file_name(nelx,nely,volfrac,penal1,penal2,rmin)
%   nelx: number of elements in x direction
%   nely: number of elements in y direction
%   volfrac: averaged volume fraction
%   penal: penalization parameter
%   rmin: lentgh scale parameter (half of the minimum member width)
%
% Selected List of variables:%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   ndx: number of nodes in x direction
%   ndy: number of nodes in y direction
%   xd: design variables, matrix ndy rows and ndx columns
%   xn: nodal densities, matrix ndy rows and ndx columns
%   rhoelement: density of an element, scalar
%   dc_xd: Sensitivities of the compiance w.r.t. design variables
%   dc_xn: Sensitivities of the compiance w.r.t. nodal densities
%   loop: iteration number
%   change: maximum change of element densities between two consecutive steps
%   nn: number of elements belonging to subdomain of each node, vector ndx
%       *ndy rows
%   nodelistx: list of column numbers of nodes belonging to subdomain of
%        node element, matrix ndx*ndy rows, number of columns depends on rmin
%   nodelisty: list of row numbers of nodes belonging to subdomain of each
%           node, matrix ndx*ndy rows, number of columns depends on rmin
%   mapx: list of column number of the design variable corresponding to each
%         nodal density, matrix ndy rows and ndx column
%   mapy: list of row number of the design variable corresponding to each
%         nodal density, matrix ndy rows and ndy column
%   KE: element stiffness matrix of solid material
%   c: objective function which is the compliance in this case
%   Ue: element displacement, vector
%
% List of sub-functions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   FE: sub-function to caryout finite element analysis, returns displacement
%   lk: sub-function to calculate element stiffness matrix of solid material
%   OC: sub-function to update design variable and calculate material densities
%       from design
%       variables, this is a revised version of the OC from Sigmund's code
%
function lengthscale_nodal_approach(nelx,nely,volfrac,penal,rmin);
% Initialize %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xd(1:(nely+1),1:(nelx+1)) = volfrac;
xn(1:nely+1,1:nelx+1) = volfrac;
```

```
046  dc_xd=zeros(nely+1,nelx+1);
047  dc_xn=zeros(nely+1,nelx+1);
048  dv=zeros((nely+1),(nelx+1));
049  loop = 0;
050  change = 1.;
051  % Build the list of adjacent nodess (with radius rmin)%%%%%%%%%%%%%%%%%%%%%%%%
052  nn=zeros((nely+1)*(nelx+1),1);
053  nodelistx=zeros((nely+1)*(nelx+1),(2*round(rmin)+1)^2);
054  nodelisty=zeros((nely+1)*(nelx+1),(2*round(rmin)+1)^2);
055  ndy=nely+1;
056  ndx=nelx+1;
057  for cny=1:ndy
058     for cnx=1:ndx
059        % Search a rectangular area only %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
060        nodenumber=ndy*(cnx-1)+cny;
061        for nly=max((cny-floor(rmin)),1):min((cny+floor(rmin)),ndy)
062           for nlx=max((cnx-floor(rmin)),1):min((cnx+floor(rmin)),ndx)
063              r=sqrt((cnx-nlx)^2+(cny-nly)^2);
064              if r<=rmin
065                 nn(nodenumber,1)=nn(nodenumber,1)+1;
066                 nodelistx(nodenumber,nn(nodenumber,1))=nlx;
067                 nodelisty(nodenumber,nn(nodenumber,1))=nly;
068              end
069           end
070        end
071     end
072  end
073  % Initialize the map from element densities to design variables %%%%%%%%%%%%%%
074  for cny=1:ndy
075     for cnx=1:ndx
076        mapy(cny,cnx)=cny;
077        mapx(cny,cnx)=cnx;
078     end
079  end
080  % Sensitivities of constraint function V with respect to nodal densities %%%%%
081  for ely = 1:nely
082     for elx = 1:nelx
083        yn1 = ely; xn1=elx;
084        yn2 = ely; xn2=elx+1;
085        yn3 = ely+1; xn3=elx+1;
086        yn4 = ely+1; xn4=elx;
087        dv(yn1,xn1)=dv(yn1,xn1)+0.25;
088        dv(yn2,xn2)=dv(yn2,xn2)+0.25;
089        dv(yn3,xn3)=dv(yn3,xn3)+0.25;
090        dv(yn4,xn4)=dv(yn4,xn4)+0.25;
091     end
092  end
093  % Load element stiffness matrix for solid material %%%%%%%%%%%%%%%%%%%%%%%%%%
094  [KE] = lk;
095  %
096  % Start iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
097  while (change > 0.01) | (penal <= penal2)
098     loop = loop + 1;
099     % FE- analysis %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
100     [U]=FE(nelx,nely,xn,penal);
101     % Calculate compliance and sensitivities w.r.t. element densities %%%%%%%%%%
102     dc_xn=zeros(ndy,ndx);
103     obj=0;
104     for ely = 1:nely
105        for elx = 1:nelx
106           yn1 = ely; xn1=elx;
107           yn2 = ely; xn2=elx+1;
108           yn3 = ely+1; xn3=elx+1;
109           yn4 = ely+1; xn4=elx;
110           rhoelement=(xn(yn1,xn1)+xn(yn2,xn2)+xn(yn3,xn3)+xn(yn4,xn4))/4;
111           n1 = (nely+1)*(elx-1)+ely;
112           n2 = (nely+1)* elx   +ely;
113           Ue = U([2*n1-1;2*n1; 2*n2-1;2*n2; 2*n2+1;2*n2+2; 2*n1+1;2*n1+2],1);
114           obj= obj-rhoelement^penal*Ue'*KE*Ue;
115           dc_xn(yn1,xn1)=dc_xn(yn1,xn1)-penal*rhoelement^(penal-1)*Ue'*KE*Ue;
116           dc_xn(yn2,xn2)=dc_xn(yn2,xn2)-penal*rhoelement^(penal-1)*Ue'*KE*Ue;
117           dc_xn(yn3,xn3)=dc_xn(yn3,xn3)-penal*rhoelement^(penal-1)*Ue'*KE*Ue;
118           dc_xn(yn4,xn4)=dc_xn(yn4,xn4)-penal*rhoelement^(penal-1)*Ue'*KE*Ue;
119        end
120     end
121     % Normalize the sensitivities w.r.t. nodal densities %%%%%%%%%%%%%%%%%%%%%%
122     dc_xn=dc_xn./dv;
123     % Calculate sensitivities w.r.t. design variables %%%%%%%%%%%%%%%%%%%%%%%%
124     tempdc=zeros(nely+1,nelx+1);
125     nan=zeros(nely+1,nelx+1);
126     for cny = 1:ndy
127        for cnx = 1:ndx
128           tempdc(mapy(cny,cnx),mapx(cny,cnx))= ...
129              tempdc(mapy(cny,cnx),mapx(cny,cnx))- ...
130              dc_xn(cny,cnx);
131           nan(mapy(cny,cnx),mapx(cny,cnx))= ...
132              nan(mapy(cny,cnx),mapx(cny,cnx))+1;
133        end
134     end
135     % Normalize the sensitivities by dividing by the influenced volume %%%%%%%%
136     for cny = 1:ndy
137        for cnx = 1:ndx
138           if nan(cny,cnx)>0
139              dc(cny,cnx)=-tempdc(cny,cnx)/nan(cny,cnx);
140           else
141              dc(cny,cnx)=dc_xn(cny,cnx);
142           end
143        end
144     end
145     % Update design variables with the revised Optimality Criteria sub-function
146     xdold=xd;xnold=xn;
147     [xd,xn,mapy,mapx] = ...
148        OC(ndx,ndy,xd,volfrac,dc,dv,nn,nodelistx,nodelisty);
149     %Calculate xe based on xn (nodal volume fraction)
150     %according to two illustration scheme: avaraged and continuous
151     nofpixel=min(floor(700/nelx),floor(400/nely));
152     for ely=1:nely
153        for elx=1:nelx
```

```
154           yn1 = ely; xn1=elx;
155           yn2 = ely; xn2=elx+1;
156           yn3 = ely+1; xn3=elx+1;
157           yn4 = ely+1; xn4=elx;
158           xe(ely,elx)=(xn(yn1,xn1)+xn(yn2,xn2)+xn(yn3,xn3)+xn(yn4,xn4))/4;
159        end
160     end
161     % Print results %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
162     change = max(max(abs(xd-xdold)));
163     disp([' It.: ' sprintf('%4i',loop) ' Obj.: ' sprintf('%10.4f',-obj) ...
164           ' Vol.: ' sprintf('%6.3f',sum(sum(xe))/(nelx*nely)) ...
165           ' ch.: ' sprintf('%6.3f',change )])
166     % Plot nodal densities %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
167     colormap(gray); imagesc(-xn); axis equal; axis tight; axis off;pause(1e-6);
168  end
169  %
170  % SUB FUNCTIONS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
171  %%%%%%%%%% OPTIMALITY CRITERIA UPDATE - revised %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
172  % Input variables:
173  %   ndx: number of nodes in x direction
174  %   ndy: number of nodes in y direction
175  %   xd: current design variables, matrix ndy rows and ndx columns
176  %   volfrac: volume fraction
177  %   dc_xd: sensitivities w.r.t. design variables, matrix ndy rows and ndx cols
178  %   dv: derivative of total volume w.r.t nodal densities
179  %   nn: number of nodes belonging to subdomain of each node, vector
180  %       ndx*ndy rows
181  %   nodelistx: list of column numbers of nodes belonging to subdomain of
182  %       each node, matrix ndx*ndy rows, number of columns depends on rmin
183  %   nodelisty: list of row numbers of nodes belonging to subdomain of each
184  %              node, matrix ndx*ndy
185  % Output variables:
186  %   xdnew: updated design variables, matrix ndy rows and ndx columns
187  %   xnnew: updated nodal densities, matrix ndy rows and ndx columns
188  %   mapx: updated list of column number of the design variable corresponding to
189  %         each nodal density, matrix ndy rows and ndx column
190  %   mapy: updated list of row number of the design variable corresponding to
191  %         each nodal density, matrix ndy rows and ndx column
192  % List of selected local variables
193  %   maxnodey: store the row number of the maximum design variables in range
194  %             while doing the search for maximum design variable for an
195  %             nodal density
196  %   maxnodex: store the column number of the maximum design variables in
197  %             range while doing the search for maximum design variable for
198  %             an nodal density
199  %   l1, l2, lmid: temporary values of the Lagrangian multipliers used by the
200  %                 bi-sectioning algorithim
201  function [xdnew,xnnew,mapy,mapx]= ...
202     OC(ndx,ndy,xd,volfrac,dc_xd,dv,nn,nodelistx,nodelisty)
203  l1 = 0; l2 = 100000; move = 0.2;
204  while (l2-l1 > 1e-4)
205     mapy=zeros(ndy,ndx);
206     mapx=zeros(ndy,ndx);
207     lmid = 0.5*(l2+l1);
```

```
208     xdnew = max(0.001,max(xd-move,min(1.,min(xd+move,xd.*(-dc_xd./lmid).^0.25))));
209     volume=0;
210     xnnew=zeros(ndy,ndx);
211     % loop all nodes in the design domain %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
212     for cnx=1:ndx
213        for cny=1:ndy
214           nodenumber=ndy*(cnx-1)+cny;
215           % loop all nodes in the sub-domain of each node %%%%%%%%%%%%%%%%%%%%
216           for i=1:nn(nodenumber)
217              if xnnew(cny,cnx) < ...
218                    xdnew(nodelisty(nodenumber,i),nodelistx(nodenumber,i))
219                 xnnew(cny,cnx) = ...
220                    xdnew(nodelisty(nodenumber,i),nodelistx(nodenumber,i));
221                 maxnodey=nodelisty(nodenumber,i);maxnodex=nodelistx(nodenumber,i);
222              end
223           end
224           mapy(cny,cnx)=maxnodey; mapx(cny,cnx)=maxnodex;
225        end
226     end
227     if sum(sum(xnnew.*dv)) - volfrac*(ndx-1)*(ndy-1) > 0
228        l1 = lmid;
229     else
230        l2 = lmid;
231     end
232  end
233  %
234  %%%%%%%%% FE-ANALYSIS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
235  % Input variables:
236  %   nelx: number of element in x direction
237  %   nely: number of element in y direction
238  %   xn: nodal densities, matrix ndy rows and ndx columns
239  %   rhoelement: density of an element, scalar
240  %   penal: penalization parameter
241  % Output variables:
242  %   U: Displacement vector
243  % List of selected local variables
244  %   KE: element stiffness matrix of solid material
245  %   K: global stiffness matrix
246  %   F: global load vector
247  %   edof: element degrees of freedom
248  %   fixeddofs: prescribed dofs, vector
249  %   alldofs: all dofs, vector
250  %   freedofs: free dofs, vector
251  function [U]=FE(nelx,nely,xn,penal)
252  [KE] = lk;
253  a=1/4*(1+1/sqrt(3))^2; b=1/6; c=1/4*(1-1/sqrt(3))^2; d=1/6;
254  K = sparse(2*(nelx+1)*(nely+1), 2*(nelx+1)*(nely+1));
255  F = sparse(2*(nely+1)*(nelx+1),1); U = zeros(2*(nely+1)*(nelx+1),1);
256  for elx = 1:nelx
257     for ely = 1:nely
258        yn1 = ely; xn1=elx;
259        yn2 = ely; xn2=elx+1;
260        yn3 = ely+1; xn3=elx+1;
261        yn4 = ely+1; xn4=elx;
```

```
      rhoelement=(xn(yn1,xn1)+xn(yn2,xn2)+xn(yn3,xn3)+xn(yn4,xn4))/4;
      n1 = (nely+1)*(elx-1)+ely;
      n2 = (nely+1)* elx   +ely;
      edof = [2*n1-1; 2*n1; 2*n2-1; 2*n2; 2*n2+1; 2*n2+2; 2*n1+1; 2*n1+2];
      K(edof,edof) = K(edof,edof) + rhoelement^penal*KE;
   end
end
% Define loads and supports %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Load and bc condition is for a cantilever beam fixed at left end
% Subject to a vertical load at the middle of the right edge
F(((nely+1)*nelx+nely/2+1)*2,1) = 1;
fixeddofs   = [1:2*(nely+1),1];
alldofs     = [1:2*(nely+1)*(nelx+1)];
freedofs    = setdiff(alldofs,fixeddofs);
% Solve the linear system %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
U(freedofs,:) = K(freedofs,freedofs) \ F(freedofs,:);
U(fixeddofs,:)= 0;
%
%%%%%%%%%% ELEMENT STIFFNESS MATRIX %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [KE]=lk
E = 1.;
nu = 0.3;
k=[ 1/2-nu/6   1/8+nu/8 -1/4-nu/12 -1/8+3*nu/8 ...
      -1/4+nu/12 -1/8-nu/8  nu/6       1/8-3*nu/8];
KE = E/(1-nu^2)*[ k(1) k(2) k(3) k(4) k(5) k(6) k(7) k(8)
   k(2) k(1) k(8) k(7) k(6) k(5) k(4) k(3)
   k(3) k(8) k(1) k(6) k(7) k(4) k(5) k(2)
   k(4) k(7) k(6) k(1) k(8) k(3) k(2) k(5)
   k(5) k(6) k(7) k(8) k(1) k(2) k(3) k(4)
   k(6) k(5) k(4) k(3) k(2) k(1) k(8) k(7)
   k(7) k(4) k(5) k(2) k(3) k(8) k(1) k(6)
   k(8) k(3) k(2) k(5) k(4) k(7) k(6) k(1)];
```

References

[1] Bendsoe, M.P. 1989: Optimal shape design as a material distribution problem, *Structural Optimization*, **1**, 193–202.

[2] Rozvany GIN. 2001: Aims, scope, methods, history and unified terminology of computer-aided topology optimization in structural mechanics, *Structural and Multidisciplinary Optimization*, **21**, 90–108.

[3] Rozvany, G.I.N., Zhou, M., and Birker, T. 1992: Generalized shape optimization without homogenization, *Structural Optimization*, **4**, 250–254.

[4] Bendsoe, M.P., and Kikuchi, N, 1988: Generating optimal topologies in structural design using a homogenization method, *Computer Methods in Applied Mechanics and Engineering*, **71**, 197–224.

[5] Suzuki, K., and Kikuchi, N. 1991: A homogenization method for shape and topology optimization, *Computer Methods in Applied Mechanics and Engineering*, **93**, 291–318.

[6] Zhou, M., and Rozvany, G. I. N. 1991: The COC Algorithm, Part II: Topological geometrical and generalized shape optimization, *Computer Methods in Applied Mechanics and Engineering*, **98**, 309–336.

[7] Mlejnik, H. P., and Schirrmacher, R. 1993: An engineering approach to optimal material distribution and shape finding, *Computer Methods in Applied Mechanics and Engineering*, **106**, 1–26.

[8] Bendsoe, M.P., and Sigmund, O. 1999: Material interpolation schemes in topology optimization, *Archive of Applied Mechanics*, **69**, 635–654.

[9] Diaz, A., and Sigmund O. 1995: Checkerboard patterns in layout optimization, *Structural Optimization*, **10**, 40–45.

[10] Sigmund, O., and Petersson, J. 1998: Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima, *Structural Optimization*, **16**, 68 –75.

[11] Haber, R.B., Jog, C.S., and Bensoe, M.P. 1996: A new approach to variable-topology shape design using a constraint on perimeter, *Structural Optimization*, **11**, 1–12.

[12] Ambrosio, L., and Buttazzo, G. 1993: An optimal design problem with perimeter penalization, *Calculus of Variations and Partial Differential Equations*, **1**, 55–69.

[13] Petersson, J. 1999: Some convergence results in perimeter-controlled topology optimization, *Computer Methods in Applied Mechanics and Engineering*, **171**, 123–140.

[14] Petersson, J. and Sigmund, O. 1998: Slope constrained topology optimization, *International Journal for Numerical Methods in Engineering*, **41**, 1417–1434.

[15] Borrwall, T. 2001: Topology optimization of elastic continua using restriction, *Archives of Computational Methods in Engineering*, **8**, 351–385.

[16] Zhou, M., Shyy, Y.K., and Thomas, H.L. 2001: Checkerboard and minimum member size control in topology optimization, *Structural and Multidisciplinary Optimization*, **21**, 152–158.

[17] Bourdin, B. 2001: Filters in topology optimization, *International Journal for Numerical Methods in Engineering*, **50**, 2143–2158.

[18] Bruns, T.E., and Tortorelli D.A. 2001: Topology optimization non-linear elastic structures and compliant mechanisms, *Computer Methods in Applied Mechanics and Engineering*, **190**, 3443–3459.

[19] Wang, M., and Wang S. 2005: Bilateral filtering for structural topology optimization, *International Journal for Numerical Methods in Engineering*, **63**, 1911–1938.

[20] Sigmund, O. 2001: A 99 line topology optimization code written in Matlab, *Structural and Multidisciplinary Optimization*, **21**, 120 –127.

[21] Sigmund, O. 1994: Design of Material Structures Using Topology Optimization, Ph.D. Thesis, *Department of Solid Mechanics, Technical University of Denmark*, 1994.

[22] Poulsen, A. 2003: A new scheme for imposing a minimum length scale in topology optimization, *International Journal for Numerical Methods in Engineering*, **57**, 741–760.

[23] Poulsen, A. 2002: A simple scheme to prevent checkerboard patterns and one-node connected hinges in topology optimization, *Structural and Multidisciplinary Optimization*, **24**, 396–399.

[24] Poulsen A., 2002: Topology optimization in wavelet space, *International Journal for Numerical Methods in Engineering*, **53**, 567–582.

[25] Kim, Y.Y., and Yoon, GH. 2000: Multi-resolution multi-scale topology optimization—a new paradigm, *International Journal of Solids and Structures*, **37**, 5529–5559.

[26] Belytschko, T., Xiao, S.P., and Parimi, C. 2003: Topology optimization with implicit functions and regularization, *International Journal for Numerical Methods in Engineering*, **57**, 1177–1196.

[27] Guest, J.K., Prevost, J.H., and Belytschko T. 2004: Achieving minimum length scale in topology optimization using nodal design variables and projection functions, *International Journal for Numerical Methods in Engineering*, **61**, 238-254.

[28] Matsui, K., and Terada, K. 2004: Continuous approximation of material distribution for topology optimization, *International Journal for Numerical Methods in Engineering*, **59**, 1925–1944.

[29] Bruns, T.E. 2005: A re-evaluation of the SIMP method with filtering and an alternative formulation for solid–void topology optimization, *Structural and Multidisciplinary Optimization*, **30**, 428–436.

[30] Pomezanski, V., Querin, O.M., and Rozvany, G.I.N. 2005: CO-SIMP: extended SIMP algorithm with direct COrner Contact COntrol, *Structural and Multidisciplinary Optimization*, **30**, 164–168.

[31] Jang, G., Jeong, H., Kim, Y., Sheen, D., Park, C., and Kim, M. 2003: Checkerboard-free topology optimization using non-conforming finite elements, *International Journal for Numerical Methods in Engineering*, **57**, 1717–1735.

[32] Jog, C.S., and Haber R.B. 1996: Stability of finite element models for distributed-parameter optimization and topology design, *Computer Methods in Applied Mechanics and Engineering*, **130**, 203–226.

[33] Rahmatalla, S.F., and Swan, C.C., 2004: A Q4/Q4 continuum structural topology optimization Implementation, *Structural and Multidisciplinary Optimization*, **27**, 130–135.

[34] Kim, J.H., and Paulino, G.H. 2002: Isoparametric graded finite elements for nonhomogeneous isotropic and orthotropic materials, *ASME Journal of Applied Mechanics*, **69**, 502–514.

[35] Hashin, Z., and Shtrikman, S. 1963: A variational approach to the theory of the elastic behaviour of multiphase materials, *Journal of the Mechanics and Physics of Solids*, **11**, 127-140.

[36] Paulino, G.H., and Silva, E.C.N. 2005: Design of functionally graded structures using topology optimization, *Materials Science Forum*, **492-493**, 435–440.

[37] Kirsch, U. 1993: Structural optimization, fundamentals and applications (Springer, New York).

[38] Bendsoe, M.P., and Sigmund, O. 2003: Topology Optimization: Theory, Methods and Applications (Springer, New York).

[39] Svanberg, K. 1987: The method of moving asymptotes – a new method for structural optimization, *International Journal for Numerical Methods in Engineering*, **24**, 359–373.

[40] Krog, L., and Tucker, A. 2004: Topology optimization of aircraft wing box ribs, *10^{th} AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*.