AN INTRODUCTION TO OPTIMAL DESIGN: HANDS-ON SESSION

C. DAPOGNY[1]

[1] *CNRS & Laboratoire Jacques-Louis Lions, Sorbonne Université, Paris, France*

Abstract. This note accompanies the hands-on session of a short course about optimal design, taught at the occasion of the CIMPA school "Contrôle et optimisation de forme : théorie et applications", held at Assane Seck University Ziguinchor, from 04/13/2026 to 04/24/2026.

Contents

The purpose of this practical session is twofold. At first, it aims to introduce the participants to the numerical resolution of boundary value problems with `FreeFem` [7] – an open-source environment for solving Partial Differential Equations (PDE) with the Finite Element Method from the input of their variational formulation via a syntax close to that of `C++` . The second objective of this session is to guide the participants through the implementation of a model density-based strategy for dealing with a shape and topology optimization problem: the famous Solid Isotropic Material with Penalization (SIMP) method.

The exercise session is organized as follows: the participants are encouraged to start with the two exercises in Section 1, in which the physical settings of interest in this course are introduced, namely those of the Laplace equation and the linearized elasticity system. The main features and commands of `FreeFem++` are presented in the meantime; for the (many!) subsequent possibilities offered by this environment, the participants will be referred to the exhaustive, albeit comprehensive online documentation, available at the following address:

`https://doc.freefem.org/documentation/index.html`

Throughout the following sections, more advanced theoretical or practical problems - tagged with a * - are left as food for thought to the reader. Their solutions are not required for the correct understanding of the targeted notions.

The present notes, as well as tentative corrections to most of the theoretical and practical questions they contain, may be downloaded (in an extended version) from the following `GitHub` repository:

`https://github.com/dapogny/GDR-MOA-Course`

The participants may also find more complete reviews of background material and tutorials about numerical methods at the following webpage of an online numerical tours project:

`https://dapogny.github.io/sctuto/index.html`

1. Getting started with FreeFem

This first section is a short introduction to the basic features of FreeFem. Participants who are already familiar with this environment may skim fast through this preliminary part.

1.1. A worked example: resolution of the Laplace equation

Our first contact with FreeFem arises in the context of the Laplace equation, which is a basic physical model for the stationary distribution of the temperature within a region, or for the voltage potential inside an electrically conductive medium.

Let D be the L-shaped 2d domain depicted on Fig. 1 (left); D is filled with a material with thermal conductivity γ; a heat source $f \in L^2(D)$ is acting inside D and its boundary ∂D is maintained at temperature 0. In this situation, the temperature field $u : D \to \mathbb{R}$ is the solution to the following Laplace equation:

$$\begin{cases} -\mathrm{div}(\gamma \nabla u) = f & \text{in } D, \\ u = 0 & \text{on } \partial D. \end{cases} \tag{1.1}$$

The numerical resolution of (1.1) using FreeFem relies on the Finite Element Method, which we now briefly summarize, referring to classical monographs such as [2] for more exhaustive presentations.

The starting point of this method is the associated *variational formulation* to (1.1): u is sought in the functional space $V := H^1_0(D)$ of functions in $H^1(D)$ whose trace vanishes on ∂D; it is the unique function in this space that satisfies:

$$\forall v \in V,\ a(u,v) = \int_D f v\, dx, \text{ where } a(u,v) := \int_D \gamma \nabla u \cdot \nabla v\, dx. \tag{1.2}$$

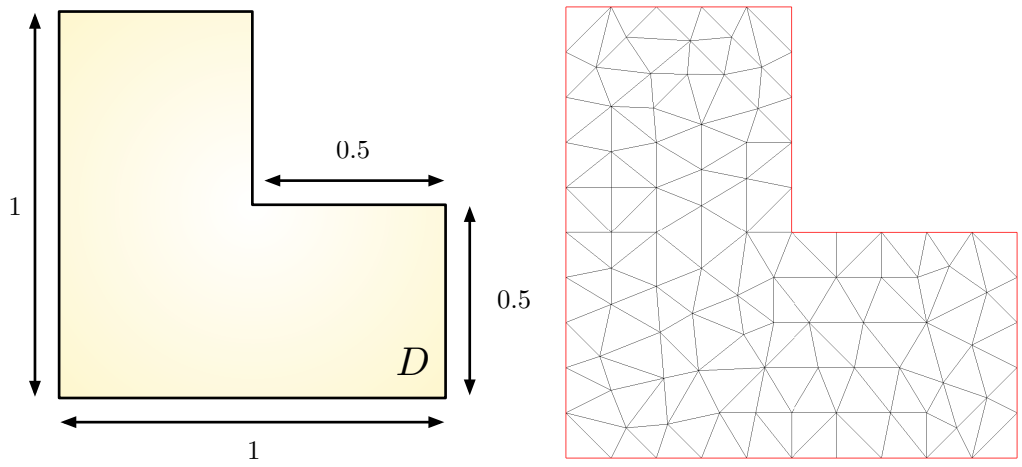

Figure 1. *(Left) The L-shaped domain D considered in Section 1.1 and (right) an associated triangular mesh generated using FreeFem.*

In practice, D is discretized by means of a computational mesh $\mathcal{T}$ (for instance made of triangles); see Fig. 1 (right). Such a mesh may be generated in FreeFem from the input of the boundary ∂D as a set of parametrized curves, according to the syntax outlined in Listing 1.

```
/* Declaration of the boundary curves of the domain */
border left(t=0,1.0){ x=0.0 ; y=1.0-t; label=1;};
border bot(t=0,1.0){ x=t ; y=0.0; label=1;};
border right(t=0,0.5){ x=1.0 ; y=t; label=1;};
border ang1(t=0,0.5){ x=1.0-t ; y=0.5; label=1;};
border ang2(t=0,0.5){ x=0.5 ; y=0.5+t; label=1;};
border top(t=0,0.5){ x=0.5-t ; y=1.0; label=1;};
```

```
/* Build mesh, display and save as a .mesh file */
mesh Th = buildmesh(left(10)+bot(10)+right(5)+ang1(5)+ang2(5)+top(5));
plot(Th,wait=1);
savemesh(Th,"Lshape.mesh");
```

LISTING 1. *Generation of a triangular mesh for the L-shaped domain of Section 1.1.*

Thence, the (continuous) variational formulation (1.2) is discretized on the mesh $\mathcal{T}$: a finite element space $V_h \subset V$ is chosen, which is indexed by the size h of the mesh $\mathcal{T}$. For instance V_h may be chosen as the set of $\mathbb{P}_1$ Lagrange finite element functions on $\mathcal{T}$, that is:

$$V_h = \left\{ u \in H_0^1(D),\ \forall T \in \mathcal{T},\ u|_T \text{ is a bivariate first-order polynomial function} \right\}. \tag{1.3}$$

Alternatively, V_h may be the set of $\mathbb{P}_2$ Lagrange finite element functions on $\mathcal{T}$:

$$V_h := \left\{ u \in H_0^1(D),\ \forall T \in \mathcal{T},\ u|_T \text{ is a bivariate second-order polynomial function} \right\}. \tag{1.4}$$

The discrete counterpart of (1.2) reads:

$$\text{Search for } u_h \in V_h \text{ s.t. } \forall v_h \in V_h,\ a(u_h, v_h) = \int_D f v_h \, dx. \tag{1.5}$$

Introducing a basis $\{\varphi_i\}_{i=1,\dots,N_h}$ of V_h, this rewrites as the linear system:

$$AU = b \tag{1.6}$$

where the unknown vector $U = (u_{h,1}, ..., u_{h,N_h})^T \in \mathbb{R}^{N_h}$ gathers the coordinates of u_h in the basis $\{\varphi_i\}$:

$$u_h = \sum_{i=1}^{N_h} u_{h,i} \varphi_i, \tag{1.7}$$

and A (resp. b) is the $N_h \times N_h$ matrix (resp the vector of size N_h) whose entries are given by:

$$\forall i, j = 1, ..., N_h,\ A_{ij} = a(\varphi_j, \varphi_i), \text{ and } b_i = \int_D f \varphi_i \, dx.$$

In `FreeFem`, finite element spaces (and functions) of the form (1.3) and (1.4) are declared as in Listing 2.

```
/* Definition of finite element spaces and functions */
fespace Vh(Th,P1); // or P2
Vh u,v;

/* Other parameters */
real gamma = 1.0;

/* Source term */
func real f() {
  return(1.0);
}
```

LISTING 2. *Definition of finite element spaces and functions in FreeFem.*

Finally, the variational formulation (1.5) is programmed and solved in `FreeFem` according to the self-explanatory syntax of Listing 3.

```
/* Stationary heat equation (solver = Conjugate Gradient) */
problem laplace(u,v,solver=CG) = int2d(Th)(gamma*(dx(u)*dx(v)+dy(u)*dy(v)))
                               - int2d(Th)(f()*v)
                               + on(1,u=0.0);  // Homogeneous Dirichlet BC

/* Resolution of the problem */
```

```
laplace;

/* Display of the result */
plot(Th,u,fill=1);
```

LISTING 3. *Resolution of the Laplace equation (1.5) in* FreeFem.

Question 1.1.1. Implement the above listings. You may want to try out different sets of physical parameters, and in particular:

- To modify the shape of the computational domain D;
- To select different mesh sizes;
- To choose different values for the conductivity γ and for the heat source f (e.g. space-dependent ones).

Unfortunately, the above numerical resolution yields disappointing results, as depicted on Fig. 2 (left). This is partly due to the fact that the solution u to (1.1) presents sharp variations near the reentrant corner of D: mathematically, this function is not much more regular than just $H^1(D)$ in there (in particular, it is not of class $\mathcal{C}^2$). The chosen coarse mesh used for the numerical resolution (see Fig. 1, (right)) is obviously not tailored to capture such fast variations. On the other hand, it is generally not desirable to work out with a uniformly very fine mesh (i.e. one composed of a lot of triangles) since the size of the linear system (1.6) would increase dramatically, and so would the CPU cost.

One remedy to this concern consists in *adapting* the computational mesh, so that it is selectively refined where needed (i.e. at the regions where the computed solution u_h presents large variations).

Question 1.1.2. Implement the procedure of Listing 4 to achieve this feature.

The result is that presented on Fig. 2 (right).

```
/* Multiple resolutions of the Laplace equation,
                intertwined with mesh adaptation steps */
for (int it=0; it<maxit; it++) {
  laplace;
  Th = adaptmesh(Th,u,err=eps);
  plot(Th,u,fill=1);
  eps *= 0.5;
}
```

LISTING 4. *Iterative resolution of (1.5) using mesh adaptation.*

Remark 1.1. *There is one way to vizualize the results of computations which offers more flexibility than the standard* `plot` *command from* FreeFem. *This uses the software* `medit`, *which is integrated into* FreeFem *under the name* `ffmedit`. *A documentation of this software may be consulted at the address:*

`https://www.ljll.math.upmc.fr/frey/logiciels/Docmedit.dir/index.html`

In particular, `medit` *makes it possible to save pictures from your results, etc.*

1.2. The linearized elasticity system

The second physical context of interest in this course is that of *linearized elasticity*, a system of equations which is widely used to model the behavior of mechanical structures (e.g. beams, struts, etc.) undergoing small deformations when external stresses are at play.

In this section, the structure Ω of interest is a two-dimensional bridge, as depicted on Fig. 3 (left): Ω is clamped on a region Γ_D of its boundary $\partial\Omega$, and surface loads $g = (0, -0.01)$ are applied on a disjoint subset $\Gamma_N \subset \partial\Omega$, modelling the impact of pedestrians or cars on the upper deck of the bridge. The remaining part $\Gamma = \partial\Omega \setminus (\overline{\Gamma_D} \cup \overline{\Gamma_N})$ is traction-free.

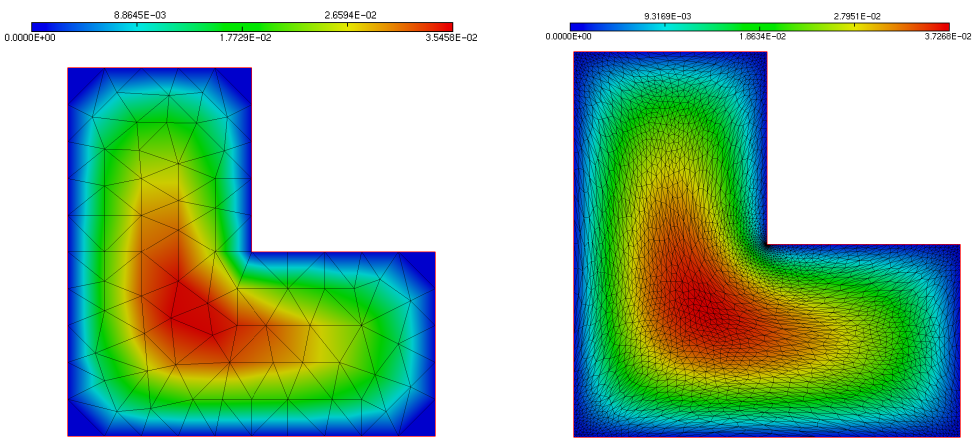

FIGURE 2. *Temperature field u obtained in the resolution of (1.5) using* `FreeFem` *with $\mathbb{P}_1$ elements (left) on the mesh of Fig. 1, (right) on an iteratively adapted mesh.*

The considered bridge Ω is filled with a linearly elastic material characterized by its Hooke's tensor A:

$$\text{For all symmetric } 2\times 2 \text{ matrix} e,\ \ Ae = 2\mu e + \lambda \mathrm{tr}(e)\mathrm{I}; \tag{1.8}$$

in the above equation, λ and μ are the Lamé coefficients, characterizing the elastic behavior of the material. In the present $2d$ plane stress situation, these are related to the more physical quantities E (the Young's modulus, appraising the resistance of the material to traction stresses) and ν (the Poisson's ratio, accounting for its resistance to shear) via the relations:

$$\lambda = \frac{E}{2(1+\nu)},\ \text{and}\ \mu = \frac{\nu E}{2(1+\nu)(1-\nu)}.$$

For the applications of this exercise, the following non dimensional values will be used:

$$E = 1 \text{ and } \nu = \frac{1}{3}. \tag{1.9}$$

The displacement $u = (u_1, u_2) : \Omega \to \mathbb{R}^2$ of the structure in these circumstances belongs to the functional space

$$H^1_{\Gamma_D}(\Omega)^2,\ \text{where } H^1_{\Gamma_D}(\Omega) := \left\{ u \in H^1(\Omega)^2,\ \ u = 0 \text{ on } \Gamma_D \right\}, \tag{1.10}$$

and it is the unique solution in the latter to the linearized elasticity system:

$$\begin{cases} -\mathrm{div}(Ae(u)) = 0 & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ Ae(u)n = g & \text{on } \Gamma_N, \\ Ae(u)n = 0 & \text{on } \Gamma, \end{cases} \tag{1.11}$$

where

$$e(u) = \frac{1}{2}(\nabla u + \nabla u^T),\ \ (e(u))_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right),\ \ i,j = 1,2, \tag{1.12}$$

is the strain tensor (i.e a matrix with size 2×2) associated to the displacement u, and $\mathrm{div}(\cdot)$ is the row-wise application of the divergence operator:

$$\text{For all differentiable matrix-valued mapping } M = (M_{ij})_{i,j=1,\dots,d} : \Omega \to R^{d\times d},\quad \mathrm{div}(M)_i = \sum_{j=1}^{d} \frac{\partial M_{ij}}{\partial x_j}.$$

1.2.1. *Solving the linearized elasticity system using* `FreeFem`

We first focus on the numerical resolution of (1.11) using `FreeFem` on the geometry Ω of Fig. 3 (left).

Question 1.2.1. Show that the the variational formulation of (1.11) reads:

$$\text{Search for } u \in H^1_{\Gamma_D}(\Omega)^2 \text{ s.t. } \quad \forall v \in H^1_{\Gamma_D}(\Omega)^2, \quad \int_\Omega Ae(u) : e(v)\,\mathrm{d}x = \int_{\Gamma_N} g \cdot v\,\mathrm{d}s,$$

where $M : N = \sum_{i,j=1}^{d} M_{ij}N_{ij}$ is the Frobenius inner product between $d \times d$ matrices $M = (M_{ij})$, $N = (N_{ij})$.

Question 1.2.2. Implement this formulation in `FreeFem`; in order to create the geometry of Ω in `FreeFem`, you may use the supplied `creabridge1.edp` file.

Question 1.2.3. Visualize the displacement of the shape. To achieve this, you may use the `movemesh` command, as is exemplified in Listing 5; see Fig. 3 (right) for the result.

```
/* Move each vertex (x,y) of Th to (x+ux(x,y),y+uy(x,y)) */
Thn = movemesh(Th,[x+ux,y+uy]);
plot(Thn);
savemesh(Thn,"brn.mesh");
```

LISTING 5. *Sample use of the* `movemesh` *command.*

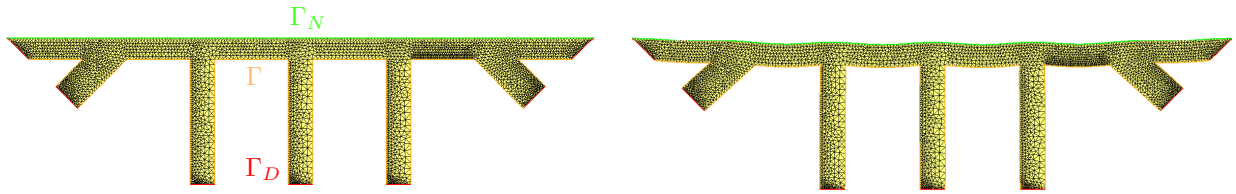

FIGURE 3. *(Left) Geometry and mesh of the bridge considered in Section 1.2; (right) deformed configuration of the bridge in the situation of (1.11).*

1.2.2.* *An eigenvalue problem*

We now turn to another key aspect in structural mechanics, namely the analysis of vibration modes, or *eigenmodes*. From the technical viewpoint, this subject is a fair excuse to get familiar with the handling of finite element matrices in `FreeFem`.

Mathematically, a real value $\lambda \in \mathbb{R}$ (resp. a $H^1_{\Gamma_D}(\Omega)^2$ function u which is not identically 0) is an eigenvalue (resp. an associated eigenfunction) for the linearized elasticity system (1.11) if:

$$\begin{cases} -\mathrm{div}(Ae(u)) = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ Ae(u)n = 0 & \text{on } \Gamma_N, \\ Ae(u)n = 0 & \text{on } \Gamma. \end{cases} \tag{1.13}$$

It is a classical, albeit non trivial result from spectral theory (see for instance [2], Chap. 7) that these eigenvalues form a sequence λ_n of positive, non decreasing real numbers going to $+\infty$ as $n \to \infty$. The corresponding eigenfunctions u_n are usually normalized by the condition $\int_\Omega u_n^2\,dx = 1$.

From the physical point of view, eigenvalues represent those values of the frequency at which imposed time-harmonic stresses may entail a dramatically large response of the structure, thus jeopardizing its integrity; such *resonance* phenomena have been responsible for the collapse of multiple buildings in the past, and notably of that of the Tacoma bridge; see for instance [1].

Question 1.2.4. Write down the (continuous) variational formulation for (1.13).

Question 1.2.5. Using the notations of Section 1.1, the discretized version of this formulation relies on the choice of an adequate finite element space V_h. Write this discretized version under the form:

$$\text{Search for } \lambda_h \in \mathbb{R} \text{ s.t. } AU = \lambda_h M, \tag{1.14}$$

where the unknown vector U is given by (1.7), and A and M are two $N_h \times N_h$ matrices which are respectively called *stiffness* and *mass* matrices.

Question 1.2.6. Implement the numerical calculation of the first (i.e. the lowest) eigenpairs (λ_1, u_1), (λ_2, u_2),... of the system (1.13) in `FreeFem`.

[Hint: the calculation of the eigenvalues of partial differential operators in ***FreeFem*** *relies on a discretized variational formulation such as that established in Question 1.2.4 and 1.2.5; this involves the assembly of the above stiffness and mass matrices A and M; see the sketch in Listing 6 for useful sample commands and Fig. 4 for a look at the result.]*

```
real sigma = 0.0; // shift value, around which eigenvalues are calculated
int nev = 5; // number of computed eigenpairs
real[int] ev(nev);
Vh2[int] [eVx,eVy](nev);
int ier;

/* Bilinear form associated to the linearized elasticity system,
        shifted by sigma*bilinear form for the rhs of the eigenproblem */
varf elas([ux,uy],[vx,vy],solver=Crout) =  int2d(Th)( 2.0*mu*
   (dx(ux)*dx(vx) + 0.5*(dy(ux)+dx(uy))*(dy(vx)+dx(vy)) + dy(uy)*dy(vy))
                        + lambda*(dx(ux)+dy(uy))*(dx(vx)+dy(vy))
                        - sigma*(ux*vx+uy*vy))
          + on(1,ux=0.0,uy=0.0);

/* Bilinear form associated to the rhs of the eigenvalue problem */
varf b([ux,uy],[vx,vy]) = int2d(Th)(ux*vx+uy*vy);

/* Matrices associated to both biinear forms */
matrix A = elas(Vh2,Vh2,solver=Crout,factorize=1);
matrix B = elas(Vh2,Vh2,solver=Crout,factorize=1);

/* Calculation of eigenvalues and eigenvectors ;
   Beware of the misleading syntax:
   passing the first component of eV will store all its components */
ier = EigenValue(A,B,sym=true,sigma=sigma,value=ev,
                                        vector=eVx,tol=1.e-10,maxit=0,ncv=0);
```

Listing 6. *Sample resolution of the eigenvalue problem (1.13) using* ***FreeFem****.*

2. Topology optimization using density-based methods

This second section focuses on one implementation of density-based methods for topology optimization.

2.1. Generalities about the implementation of density-based methods

We start with a few generalities about the implementation of density-based topology optimization methods, in the context of linearized elasticity described in Section 1.2: we aim to minimize a function $J(\Omega)$, depending on the shape of an elastic structure Ω via the displacement u_Ω, solution to (1.11).

(1) A fixed computational domain D is considered, which is meshed once and for all.

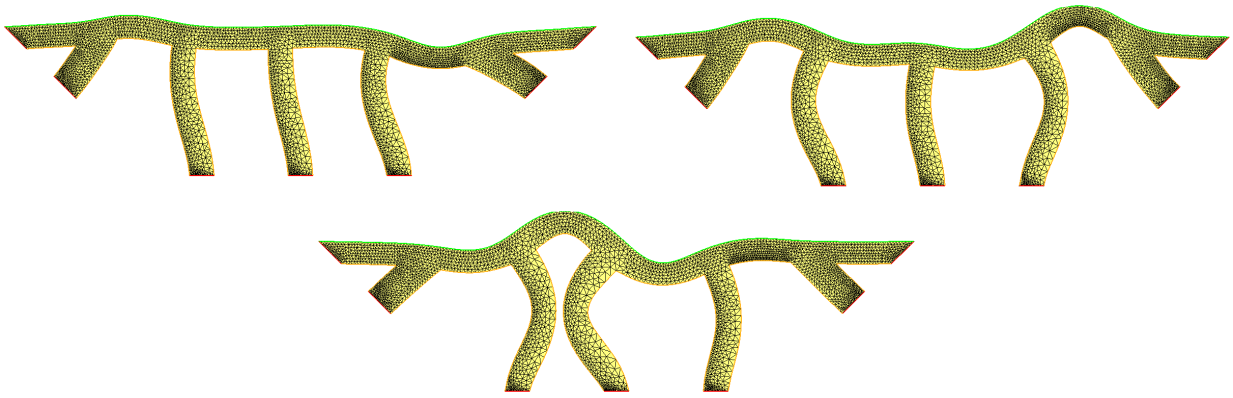

FIGURE 4. *(From left to right, top to bottom) Deformed configurations associated to the first three eigenmodes u_1, u_2, u_3 of the bridge considered in Section 1.2.*

(2) Instead of the "exact" linearized elasticity system (1.11) for u_Ω, we consider the "ersatz material approximation" posed on D, where the "void" region $D \setminus \overline{\Omega}$ is replaced by a "very soft" material:

$$\left\{ \begin{array}{rl} -\mathrm{div}(A_{\Omega,\varepsilon} e(u_{\Omega,\varepsilon})) = 0 & \text{in } D, \\ A_{\Omega,\varepsilon} e(u_{\Omega,\varepsilon}) n = g & \text{on } \Gamma_N, \\ A_{\Omega,\varepsilon} e(u_{\Omega,\varepsilon}) n = 0 & \text{on } \Gamma, \\ u_{\Omega,\varepsilon} = 0 & \text{on } \Gamma_D. \end{array} \right. \tag{2.1}$$

Here, the tensor $A_{\Omega,\varepsilon}$ is defined by:

$$A_{\Omega,\varepsilon} = \chi_\Omega A + (1 - \chi_\Omega)\varepsilon A, \tag{2.2}$$

involving the characteristic function χ_Ω of Ω:

$$\forall x \in D, \ \ \chi_\Omega(x) = \left\{ \begin{array}{ll} 1 & \text{if } x \in \Omega, \\ 0 & \text{if } x \in D \setminus \overline{\Omega}, \end{array} \right.$$

and the "ersatz" parameter $\varepsilon \ll 1$.

(3) Density-based methods give a meaning to the quantities $A_{\Omega,\varepsilon}$ and $u_{\Omega,\varepsilon}$ in (2.1) and (2.2) in the more general situation where χ_Ω is replaced by a *density function* $h : D \to [0,1]$ - i.e. h may take "grayscale" values between "black" (1, indicating the presence of material) and "white" (0, indicating ersatz material). To this end, one possibility is to trade (2.2) for the tensor:

$$A_{h,\varepsilon} := \zeta(h) A + (1 - \zeta(h))\varepsilon A, \tag{2.3}$$

where $\zeta : \mathbb{R} \to \mathbb{R}$ is an *interpolation function* which endows material properties to intermediate values of the density. The most common choice in the practice of the SIMP method is a power law

$$\zeta(h) = h^p \ \text{ (typically, } p = 3),$$

which has the effect of penalizing the presence of grayscale values for h.

One then defines $u_{h,\varepsilon}$ as the solution to (2.1) where the tensor $A_{\Omega,\varepsilon}$ is replaced by $A_{h,\varepsilon}$. The objective function of the domain $J(\Omega)$ is then given a density-based counterpart $J(h)$.

(4) For various purposes, it is often desirable to *filter* the density h before it is used into the interpolation scheme (2.3). The following filters will be considered during this exercise session, see [10] for other possibilities:

- *Smoothing filter* [4]*:* h is replaced by $L_\alpha h = q$, the solution in $H^1(D)$ to the following equation:

$$\left\{ \begin{array}{rl} -\alpha^2 \Delta q + q = h & \text{in } D, \\ \frac{\partial q}{\partial n} = 0 & \text{on } \partial D, \end{array} \right.$$

for a small parameter $\alpha > 0$, of the order of the mesh size. This regularizing practice ensures that the effective density $q = L_\alpha h$ does not show too steep variations between the values 0 and 1, which could cause numerical instabilities.

- *Heaviside filter:* h is replaced in (2.3) by $H_\beta h$, where

$$H_\beta h = 1 - e^{-\beta h} + e^{-\beta} h.$$

Intuitively, H_β gets closer and closer to the Heaviside function as $\beta \to \infty$; thus, the intermediate values of the effective density will be attracted to either 0 or 1; see [6]. Variants of the above filter can be considered, that are not so strongly biased towards the value 1, see e.g. [12]:

$$\widetilde{H_{\beta,\eta}} h = \frac{\tanh(\beta\eta) + \tanh(\beta(h-\eta))}{\tanh(\beta\eta) + \tanh(\beta(1-\eta))},$$

where $\eta \in (0,1)$ tunes the bias between 0 and 1 of the filter as $\beta \to \infty$.

Of course, both types of filters may be combined (i.e. composed); it is important to note that the use of filters implies that the derivatives of the mappings $h \mapsto L_\alpha h$ and $h \mapsto H_\beta h$ (or $h \mapsto \widetilde{H_{\beta,\eta}} h$) will appear in the derivative of the optimized criterion.

2.2. **Topology optimization of a heat lens**

The purpose of this example is to optimize the shape of a *heat lens* - a device meant to insulate a user-defined region from an incoming heat flux. This test-case is inspired from the thermal lens problem discussed in [5], Example 5.4.2.

The 2d setting is that depicted on Fig. 5. The physics at play when the domain is occupied with a density h is described by the following steady-state heat conduction equation:

$$\left\{ \begin{array}{ll} -\mathrm{div}(\zeta(h)\nabla u_h) = 0 & \text{in } D, \\ \zeta(h)\frac{\partial u_h}{\partial n} = g_{\text{in}} & \text{on } \Gamma_{\text{in}}, \\ \zeta(h)\frac{\partial u_h}{\partial n} = g_{\text{out}} & \text{on } \Gamma_{\text{out}}, \\ \zeta(h)\frac{\partial u_h}{\partial n} = 0 & \text{on } \partial D \setminus (\Gamma_{\text{in}} \cup \Gamma_{\text{out}}), \end{array} \right. \tag{2.4}$$

where the incoming and outgoing fluxes equal respectively $g_{\text{in}} = -1.0$ and $g_{\text{out}} = 1.0$, and $\zeta(h)$ interpolates between a poor conductor with conductivity $\gamma_0 = 1$ (the phase where $h(x) = 0$) and a good conductor with conductivity $\gamma_1 = 10$ (the phase where $h(x) = 1$).

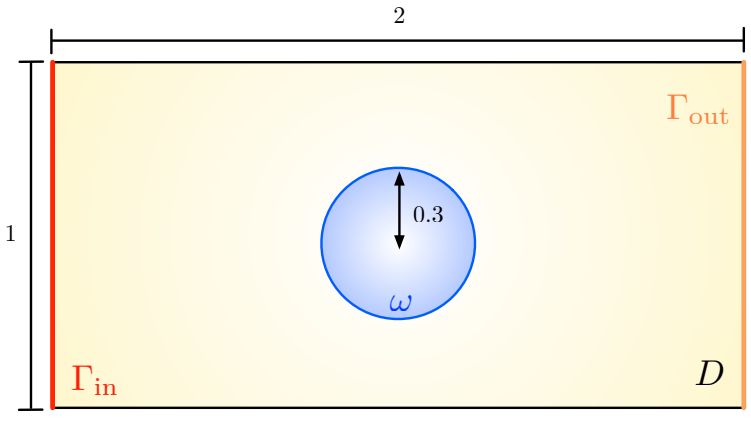

FIGURE 5. *Setting of the thermal lens test-case of Section 2.2.*

By acting on the repartition h between weak and good conductors within D, we aim to nullify the horizontal heat flux $\gamma_0 \nabla u_h \cdot e_1$ inside a fixed region $\omega \Subset D$ occupied with the weak conductor. In applications, good conductors are more expensive than weak ones, and so we add a constraint on the volume $\mathrm{Vol}(h) =$

$\int_D h\,dx$ of used good conductor. The considered topology optimization problem then reads:

$$\min_h \{F(h) + \ell \mathrm{Vol}(h)\}\,, \text{ where } F(h) = \int_D \chi_\omega \gamma_0^2 \left|\frac{\partial u_h}{\partial x_1}\right|^2 dx, \tag{2.5}$$

and ℓ is a fixed weight; e.g. $\ell = 10$ for this example.

Question 2.2.1. Write down the variational formulation for (2.4), and implement it, for instance in the case where the density h^0 identically takes the value 0.5, except inside ω where it is set to 0.

[Hint: Take care of the fact that only homogeneous Neumann boundary conditions are applied in (2.4), so that the field u_h is defined up to a constant. How do you take this fact into account in `FreeFem`*?]*

Question 2.2.2. Calculate the derivative of the objective function $F(h)$ given in (2.5), for instance by using the method of Céa.

Question 2.2.3. Implement a basic, steepest-descent density-based topology optimization algorithm to solve the problem (2.5). The device of your algorithm notably implies several choices as regards:

- The interpolation profile $\zeta(h)$ between the conductivity values γ_0 and γ_1 of the poor and good conductors, respectively;
- The use of one or several of the density filters described in Section 2.1;
- The use of a change of inner products to infer a descent direction from the derivative calculated in Question 2.2.2 (see the course);
- The particular gradient-based optimization algorithm based on the result of Question 2.2.2 (use of a line search procedure for finding a good descent step, etc.).

An illustration of the result is provided in Fig. 6.

Question* 2.2.4. Of great physical interest for applications are also devices achieving the opposite physical behavior to that of thermal lenses, namely so-called *field concentrators*, whose aim is to *maximize* the horizontal heat flux $F(h)$ through the region ω, while still imposing a constraint on the volume of the phase filled with good conductor.

Adapt the previous questions to this new setting.

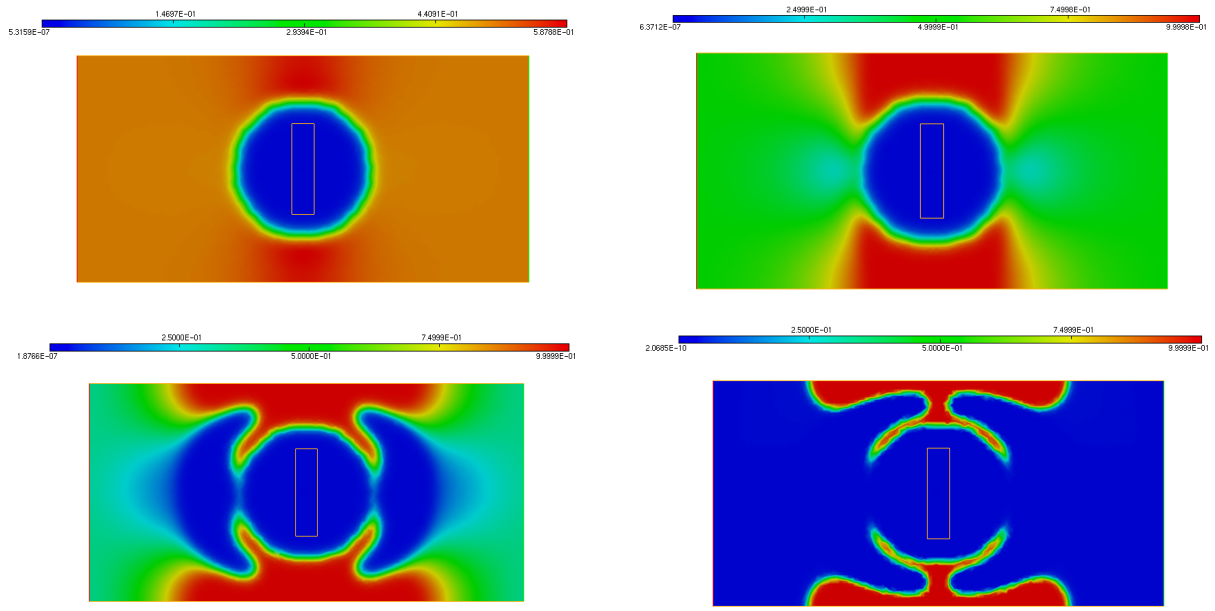

FIGURE 6. *Iterations* 0, 10, 75 *and* 200 *of the heat lens optimization test-case of Section 2.2.*

Question* 2.2.5. In the practice of the SIMP method, people generally use different (still gradient-based) optimization algorithms from the steepest-descent strategy advocated in Question 2.2.3. Here are two examples of popular strategies that you may try to implement:

- The Optimality Criteria (OC) method, which allows to impose exactly a desired volume constraint (instead of a mere penalization of the objective in (2.5)) by relying on the optimality conditions of constrained optimization programs; see for instance [3] §1.2.2 about this strategy.
- The Method of Moving Asymptotes (MMA) [11], a method from constrained optimization relying on convex approximations of the objective and constraint functions which is especially tailored to density-based topology optimization; see also [3] §1.2.3.

You may also consider to try out different optimization algorithms such as the nonlinear conjugate gradient method; see [8], Section 8.1 about the implementation of such algorithms in `FreeFem`

2.3. The classical cantilever benchmark in linearized elasticity

We now turn to the perhaps most classical example in shape and topology optimization, namely the so-called *cantilever* example. The physical setting is illustrated in Fig. 7: a beam occupies a rectangular hold-all domain D with size 2×1; D is clamped on its left-hand side Γ_D and vertical traction loads $g = (0, -1)$ are applied on a part Γ_N of its right-hand side. Body forces within D are neglected.

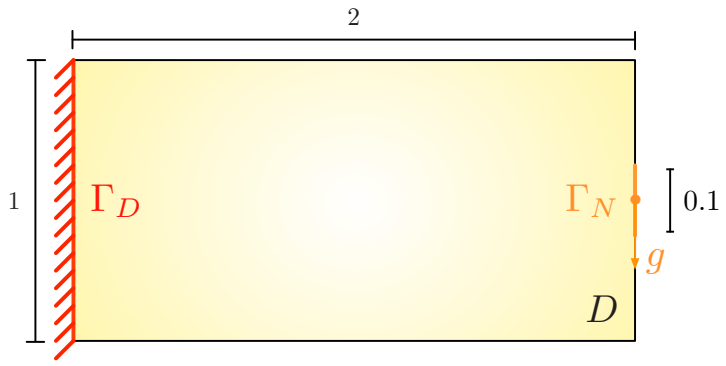

FIGURE 7. *Setting of the cantilever test case of Section 2.3.*

In the context of density-based topology optimization, the problem at stake consists in searching for the repartition of an elastic material with Hooke's tensor A defined by (1.8) and (1.9), and of the corresponding ersatz material εA, with $\varepsilon = 1e^{-3}$ within D, as described in Section 2.1.

Question 2.3.1. Write down the linearized elasticity system for the displacement $u_h \in H^1_{\Gamma_D}(D)^2$ characterizing the behavior of D when the repartition between both phases is dictated by the density function $h : D \to (0, 1)$. Write down the corresponding variational formulation for u_h.

Our aim is to minimize the *compliance* of the structure; the latter may be equivalently understood as the elastic energy stored in the structure D, or as the work of external loads acting on D:

$$C(h) = \int_D A_h e(u_h) : e(u_h)\, dx = \int_{\Gamma_N} g \cdot u_h\, ds.$$

A constraint on the volume $\mathrm{Vol}(h)$ of the material phase is added, so that we solve the problem

$$\min_h \{C(h) + \ell \mathrm{Vol}(h)\} \tag{2.6}$$

for a fixed Lagrange multiplier ℓ (in the application of this section, you may use $\ell = 10$).

Question 2.3.2. Calculate the derivative of the mapping $h \mapsto C(h)$, for instance by using the formal method of Céa.

Question 2.3.3. Implement a steepest-descent algorithm for the numerical resolution of (2.6), in the spirit of that of Question 2.2.3.

Again, you may be interested in trying out different mesh sizes, different filters, and different initial designs. What do you observe?

The formulation (2.6) of the considered optimization problem is a little unsatisfactory, since it does not account for a constraint on the volume of used material, so to speak, but rather for a penalization of the minimized objective (the compliance) by this volume constraint. In particular, a little tuning of the Lagrange multiplier ℓ is in order so as to "guess" which value will yield an optimized design with the desired volume.

Question 2.3.4. The purpose of this question is to implement an augmented Lagrangian algorithm (see e.g. [9], Chap. 17) dedicated to the resolution of the equality constrained optimization program

$$\min_h C(h) \text{ s.t. } \mathrm{Vol}(h) = V_T, \tag{2.7}$$

where V_T is a user-defined desired volume target.

Roughly speaking, the augmented Lagrangian algorithm transforms the constrained optimization problem (2.7) into a sequence of unconstrained problems of the form (2.6), indexed with the superscript ^{n}, in which the weight parameter is consistently updated.

More precisely, we define the augmented Lagrangian functional $\mathcal{L}(h, \ell, b)$ by:

$$\mathcal{L}(h, \ell, b) = J(h) - \ell(\mathrm{Vol}(h) - V_T) + \frac{b}{2}(\mathrm{Vol}(h) - V_T)^2;$$

intuitively, ℓ is intended as a closer and closer estimate of the Lagrange multiplier for the equality constraint in the optimality conditions of (2.7) while b is a penalty parameter, initialized with a "low" value, increasing so as to impose steadily this constraint in (2.7).

The augmented Lagrangian algorithm then alternates between the update of the coefficients ℓ^n and b^n and the search for a minimizer h^n to $h \mapsto \mathcal{L}(h, \ell^n, b^n)$ for fixed values ℓ^n and b^n. In practice, since the latter search is computationally expensive, the following practical version is used:

- **Initialization:** Start with an initial design h^0 and initial values ℓ^0 and b^0.
- **For** $n = 0, \ldots$ **until convergence:**
  - Find a descent direction $\widehat{h}^n$ for $h \mapsto \mathcal{L}(h, \ell^n, b^n)$;
  - Find a small enough time-step τ^n so that $\mathcal{L}(h^n + \tau^n \widehat{h}^n, \ell^n, b^n) < \mathcal{L}(h^n, \ell^n, b^n)$ and set $h^{n+1} = h^n + \tau^n \widehat{h}^n$;
  - Update the coefficients ℓ^n and b^n according to the rule:

$$\ell^{n+1} = \ell^n - b^n(\mathrm{Vol}(h^n) - V_T), \text{ and } b^{n+1} = \begin{cases} \alpha b^n & \text{if } b^n < b_{\max}, \\ b^n & \text{otherwise,} \end{cases}$$

    where $\alpha > 1$ and $b_{\max}$ are user-defined parameters.

Implement the augmented Lagrangian algorithm in the context of the cantilever test-case.

See Fig. 8 for an illustration of the result.

2.4. Topology optimization of a compliant mechanism

One very important application field of topology optimization is the design of efficient *actuators*, or *compliant mechanisms*: these are mechanisms whose displacement complies with a prescribed behavior under application of a given input force.

In this section, we consider the optimal design of a displacement inverter, as illustrated on Fig. 9. In a square-shaped hold-all domain D, the considered structures $\Omega \subset D$ are clamped near the upper and lower regions Γ_D of their left-hand side, and a known force $g = (100, 0)$ is applied on the middle Γ_N of their left-hand side. A region Γ_S of their right-hand side is connected to a spring with stiffness constant $k_s = 0.1$, which opposes the motion of the latter part. In this setting, the displacement $u_\Omega \in H^1_{\Gamma_D}(\Omega)^2$ of a shape

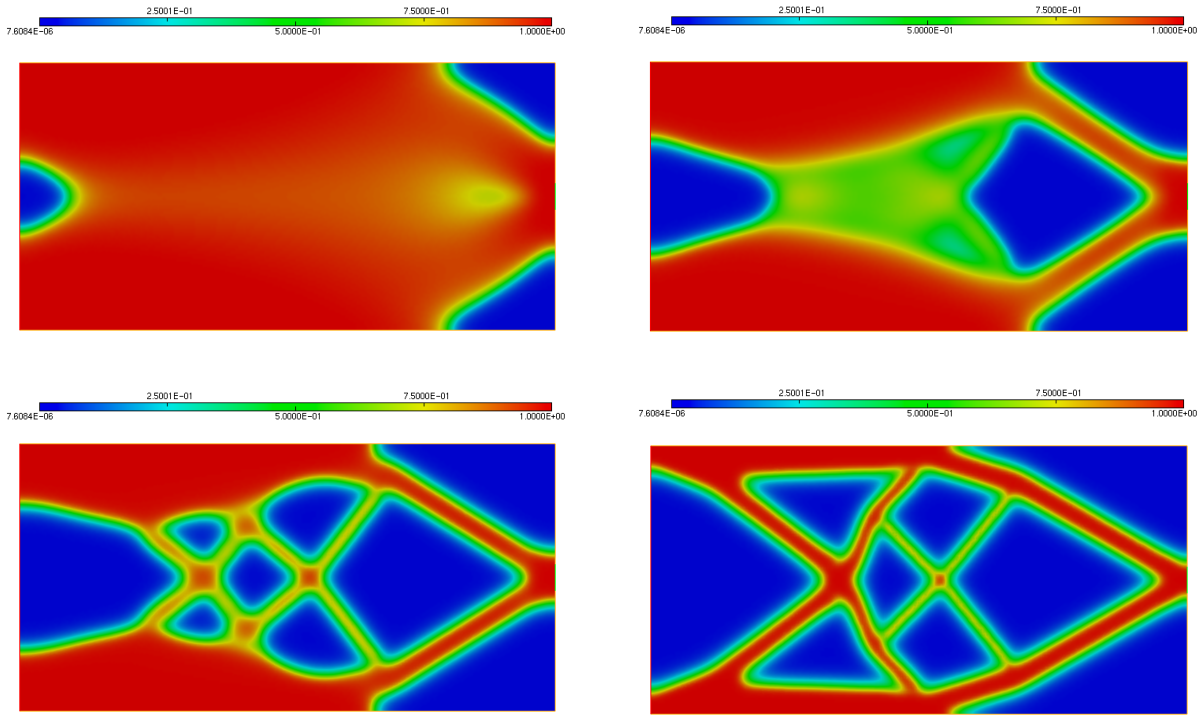

FIGURE 8. *(From left to right, top to bottom) Iterations* 20, 35, 50 *and* 200 *in the topology optimization example of the cantilever beam of Section 2.3.*

$\Omega \subset D$ satisfies the following linearized elasticity system:

$$\begin{cases} -\mathrm{div}(Ae(u)) = 0 & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ Ae(u)n = g & \text{on } \Gamma_N, \\ Ae(u)n + k_s u = 0 & \text{on } \Gamma_S, \\ Ae(u)n = 0 & \text{on } \Gamma. \end{cases} \tag{2.8}$$

We wish to maximize the negative horizontal displacement of Ω on the output area Γ_S, while minimizing the positive horizontal displacement in the input region Γ_N. To achieve this goal, we minimize the functional $A(\Omega)$ defined by:

$$A(\Omega) = \alpha_{\text{in}} \int_{\Gamma_N} u_\Omega \cdot e_1 \, \mathrm{d}s + \alpha_{\text{out}} \int_{\Gamma_S} u_\Omega \cdot e_1 \, \mathrm{d}s,$$

under a constraint on the volume of shapes. Here α_{in} and α_{out} are two weighting constants (for instance, $\alpha_{\text{in}} = 1$ and $\alpha_{\text{out}} = 2$).

Question 2.4.1. Write down the density-based formulation corresponding to the shape optimization problem

$$\min_{\Omega} A(\Omega) \text{ s.t. } \mathrm{Vol}(\Omega) = V_T,$$

where V_T is a user-defined target volume.

Question 2.4.2. Calculate the derivative of the objective function $A(h)$ involved in this density-based topology optimization problem, for instance by using the method of Céa.

Question 2.4.3. Implement a steepest-descent algorithm for the resolution of this topology optimization problem, taking stock of the knowledge obtained from the work in Sections 2.2 and 2.3.

An illustration of the result is provided in Figs. 10 and 11.

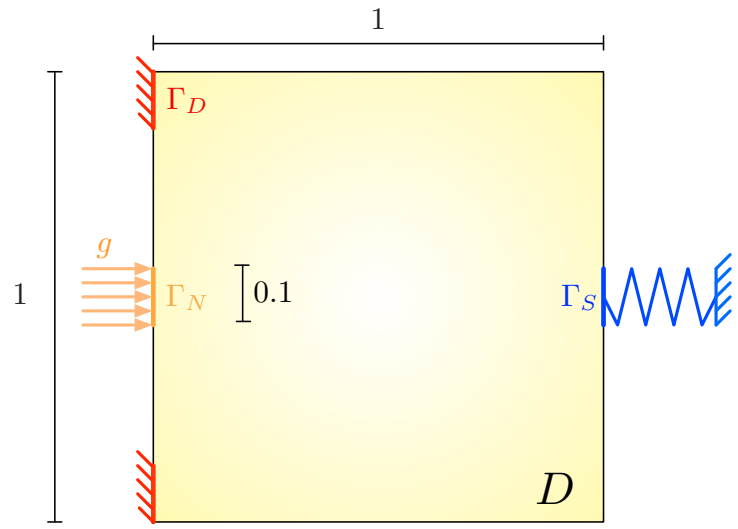

FIGURE 9. *Mechanical setting of the displacement inverter mechanism test-case of Section 2.4.*

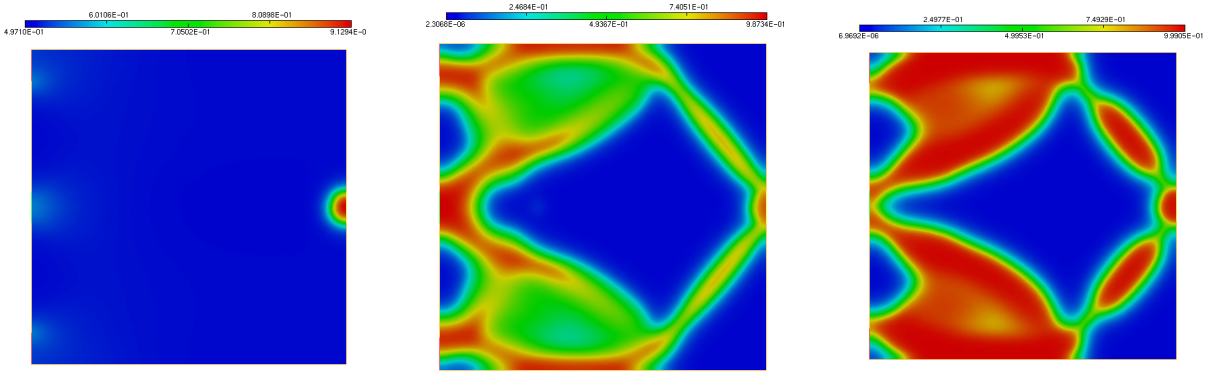

FIGURE 10. *(From left to right) Iterations* 0, 50 *and* 200 *in the topology optimization example of the inverter mechanism of Section 2.4.*

REFERENCES

[1] *Tacoma bridge*. https://fr.wikipedia.org/wiki/Pont_du_detroit_de_Tacoma_(1940). Accessed: 2018-08-27.

[2] G. ALLAIRE, *Analyse numérique et optimisation: Une introduction à la modélisation mathématique et à la simulation numérique*, Editions Ecole Polytechnique, 2005.

[3] M. P. BENDSOE AND O. SIGMUND, *Topology optimization: theory, methods, and applications*, Springer Science & Business Media, 2013.

[4] B. BOURDIN, *Filters in topology optimization*, International journal for numerical methods in engineering, 50 (2001), pp. 2143–2158.

[5] A. CHERKAEV, *Variational methods for structural optimization*, vol. 140, Springer Science & Business Media, 2012.

[6] J. K. GUEST, J. H. PRÉVOST, AND T. BELYTSCHKO, *Achieving minimum length scale in topology optimization using nodal design variables and projection functions*, International journal for numerical methods in engineering, 61 (2004), pp. 238–254.

[7] F. HECHT, *New development in freefem++*, Journal of numerical mathematics, 20 (2012), pp. 251–266.

[8] F. HECHT, O. PIRONNEAU, A. LE HYARIC, AND K. OHTSUKA, *Freefem++ manual*, 2005.

[9] J. NOCEDAL AND S. J. WRIGHT, *Numerical optimization 2nd*, Springer, 2006.

[10] O. SIGMUND AND K. MAUTE, *Topology optimization approaches*, Structural and Multidisciplinary Optimization, 48 (2013), pp. 1031–1055.

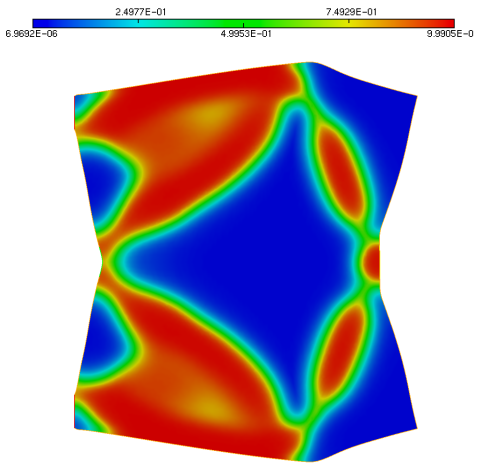

FIGURE 11. *Deformed configuration of the optimized inverter mechanism of Section 2.4.*

[11] K. SVANBERG, *The method of moving asymptotesÑa new method for structural optimization*, International journal for numerical methods in engineering, 24 (1987), pp. 359–373.

[12] F. WANG, B. S. LAZAROV, AND O. SIGMUND, *On projection methods, convergence and robust formulations in topology optimization*, Structural and Multidisciplinary Optimization, 43 (2011), pp. 767–784.