Structural Optimization with FreeFem++

Grégoire Allaire, Olivier Pantz

▶ **To cite this version:**

Grégoire Allaire, Olivier Pantz. Structural Optimization with FreeFem++. Structural and Multidisciplinary Optimization, 2006, 32, pp.173-181. hal-01089081

HAL Id: hal-01089081
https://hal.science/hal-01089081

Submitted on 1 Dec 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ECOLE POLYTECHNIQUE

CENTRE DE MATHÉMATIQUES APPLIQUÉES

UMR CNRS 7641

91128 PALAISEAU CEDEX (FRANCE). Tél: 01 69 33 41 50. Fax: 01 69 33 30 11
http://www.cmap.polytechnique.fr/

Structural Optimization with FreeFem++

G. Allaire, O. Pantz

R.I. 586 *Novembre 2005*

Structural Optimization with FreeFem++

G. Allaire, O. Pantz

Abstract The aim of this paper is to show that relatively small, simple and efficient shape optimization routines can be written using the free finite element software FreeFem++. This is illustrated by the implementation of two classical methods: the boundary variation method and the homogenization one. Even though these routines are simple enough so that their implementation can be assigned (partially or totally) as homework to graduate students, they yield results accurate enough to be useful tools for engineers or researchers.

1 Introduction

It is worthless to emphasize that a course on structural optimization must be illustrated by several examples of the methods introduced. Better is to let the students use structural optimization softwares or (even better) to let them write their own. Nevertheless, it is not realistic to ask a student to implement all parts of such algorithms as mesh generation or the finite elements method (for unstructured mesh). On the other hand, using a well-established finite element software, the difficulty is usually moved to the optimization loop (including the adjoint analysis). The goal of this paper is to exhibit (at least) one good compromise between easy optimization and easy finite element analysis, which makes possible for graduate students to develop their own structural optimization programs. Indeed, as shown in the following, relatively small (about 300 lines) and simple structural optimization routines can be written using FreeFem++, a free and user friendly 2-d finite element software (Hecht et al. 2005). Depending on the available time, it is possible to give the students most of the script and to let them complete some missing parts, as the variational formulations associated to the state and adjoint equations. Our experience is based on a graduate course taught at Ecole Polytechnique (Allaire 2005).

We have implemented two classical structural optimization methods: Hadamard method for geometric optimization (Pironneau 1984), (Sokołowski and Zolesio 1992) and the homogenization method for topology optimization (Allaire 2001), (Bendsoe and Sigmund 2003). The paper is divided in two independent parts, each one being devoted to one method. For each part, after recalling the principles of the method (we refer to the above quoted textbooks for a more complete presentation), we describe briefly its implementation in FreeFem++. The free software FreeFem++ requires as input a script which describes the geometry of the mesh, the variational formulation of the problem and the optimization loop. These ingredients have to be provided by the user, all other aspects of finite elements being automatically managed by FreeFem++, including mesh generation, adaptation and deformation, assembling the rigidity matrix, solving the linear system, displaying the result, etc. This is different in spirit of other softwares like Matlab (see (Sigmund 2001) for an application in structural optimization). Our routines (i.e. FreeFem++ scripts, tested with the 1.47 version) are freely available on the web page
http://www. cmap.polytechnique.fr/~optopo
so that anybody can reproduce our illustrative numerical examples (and get the inspiration to create new ones).

the date of receipt and acceptance should be inserted later

G. Allaire, O. Pantz

CMAP, Ecole Polytechnique, 91128 Palaiseau, France

2 Boundary variations or geometric optimization

2.1 The gradient method

The gradient method is probably the simplest tool of optimization but it may become tricky when applied to shapes, so we indulge ourselves in giving some details. Let F be a map from an Hilbert space X into $\mathbb{R}$. The gradient method amounts to build a sequence of elements $(x_n)_{n\geq 0} \in X$ by

$$x_{n+1} = x_n - h_n d_n,$$

where $h_n \in \mathbb{R}^+$ is a small positive step and d_n is the descent direction defined by

$$(d_n, y)_X =< F'(x_n), y >_{X^*,X} \text{ for any } y \in X.$$

Usually, the identification between X and its dual X^* under the scalar product $(.,.)_X$ is understood. If we do so, d_n is nothing more but the gradient of F, $F'(x_n)$. If $F'(x_n)$ is not equal to zero, then for h_n small enough, $F(x_{n+1}) < F(x_n)$. The algorithm is initialized with any element $x_0 \in X$, and if F is strongly convex, x_n converges toward the optimal solution x_* of the problem

$$F(x_*) = \min_{x \in X} F(x).$$

In structural optimization, the search set X is no more a Hilbert space but a subset of the open sets of $\mathbb{R}^N$ (where usually, $N = 2$ or 3) and has neither straightforward differentiable nor Hilbert structure. Nevertheless, the gradient method can successfully be applied. To this end, we need to define variations of open sets, and endow the set of variations with an Hilbert (or at least Banach) structure.

2.2 Variations of an open set

Let $J(\tilde{\Omega})$ be a real valued function defined for any open set $\tilde{\Omega}$ of $\mathbb{R}^N$. Let Ω be a regular open set of $\mathbb{R}^N$. Given a map θ from Ω into $\mathbb{R}^N$, we set

$$\Omega(\theta) = (\mathrm{Id} + \theta)(\Omega) \equiv \{x + \theta(x) \text{ s.t. } x \in \Omega\}.$$

For small vector field θ, the open set $\Omega(\theta)$ are one-to-one perturbations of the initial set Ω. If the map $F_\Omega : \theta \mapsto J(\Omega(\theta))$ is differentiable, we define the shape derivative

$$< J'(\Omega), \theta >=< F'_\Omega, \theta > .$$

By Hadamard structure theorem, it is known that the shape derivative is carried only on the boundary of the shape, i.e.

$$< J'(\Omega), \theta >= \int_{\partial\Omega} j(\Omega)\, \theta \cdot n \, ds. \tag{1}$$

2.3 The boundary variation algorithm

In order to apply the gradient method to shape optimization, it remains to associate to a given shape derivative $J'(\Omega)$ a direction of slope d. To this end, it suffices to endow the space of vector fields from Ω into $\mathbb{R}^N$ with an Hilbert structure, for instance $H^1(\Omega)^N$. In this case, the descent direction is the unique element $d \in H^1(\Omega)^N$ such that for every $\theta \in H^1(\Omega)^N$,

$$\int_\Omega (\nabla d \cdot \nabla \theta + d \cdot \theta)\, dx =< J'(\Omega), \theta > . \tag{2}$$

Computing d as the solution of (2) can also be interpreted as a regularization of the gradient.

Remark 1 The choice of $H^1(\Omega)^N$ as space of variations is more dictated by technical considerations (it is easy to solve (2) with `FreeFem++`) rather than theoretical ones. Many choices could be made and it is not obvious to find the one which provides the better rate of convergence. Moreover, it is possible to use a more general framework and to define the direction d as the minimizer (in some Banach space) of a functional of the form $\frac{1}{2} I(d) - < J'(\Omega), \theta >$, where I is a positive functional. The present case corresponds to the choice $I(d) = \|d\|^2_{H^1(\Omega)}$.

Remark 2 Hadamard structure theorem tells us that the directional derivative of J depends only on the value of the normal component $\theta \cdot n$ on $\partial\Omega$ (see formula (1)). Thus, one can replace the space $H^1(\Omega)^N$ in (2) by $H^1(\partial\Omega)^N$ with the corresponding change of scalar product and the new descent direction is the solution of

$$\int_{\partial\Omega} d' \cdot \theta' + d \cdot \theta \, ds =< J'(\Omega), \theta >, \text{ for every } \theta \in H^1(\partial\Omega)^N,$$

where $'$ denotes the surface gradient. Nevertheless, it is more convenient to use (2) because it is simpler to solve, and it yields a natural extension of the mesh deformation on the whole domain Ω.

The resulting algorithm can be summarized as follows:

1. Choose an initial shape Ω_0.
2. Iteration until convergence for $n \geq 0$,
   (a) Compute d_n solution of the problem (2) with $\Omega = \Omega_n$.
   (b) Set $\Omega_{n+1} = (\mathrm{Id} - h_n d_n)(\Omega_n)$, where h_n is a small positive real.

2.4 Algorithmic details of the method

In the following, several algorithmic details are discussed which make the method truly efficient and effective.

2.4.1 Regularization of the mesh

It is well known that numerical shape optimization may yield optimal designs with oscillating boundaries (i.e. having peaks and wells of length scale of the order of the mesh size). To avoid this problem, one can add a perimeter penalization to the cost function J, but the resulting solution will depend on the weight of the penalization. We prefer to use a regularization procedure which explicitly smoothes the shape at each iteration. To this end, we introduce two different meshes of Ω. At each

iteration, a fine mesh $\mathcal{S}_h$ is used to perform the finite element analysis and compute the descent direction d. We extract a coarser mesh $\mathcal{T}_h$ from $\mathcal{S}_h$, the nodes of which are moved in the direction $-d$ defined by (2). Finally, a new fine mesh $\mathcal{S}_h$ is deduced from the displaced $\mathcal{T}_h$ by mesh adaptation.

2.4.2 Regularization of the shape gradient

The displacement usually presents singularities at the corners of the shape or at the changes of boundary conditions type. In such cases, the formula (5) is not correct anymore, as the right member of the formula is not well defined. From the numerical point of view, this leads to strong oscillations of the shape near its corners and can produce unresolved meshing errors. To bypass this problem, we arbitrarily set the shape gradient to zero near the corners of the shape.

2.4.3 Volume constraint

Many structural optimization problems feature a volume constraint on the admissible open sets. This constraint is imposed by introducing a Lagrange multiplier ℓ in the formulation. More precisely, the descent direction is now computed from the shape derivative of the Lagrangian $J'(\Omega) + \ell V'(\Omega)$, where $V(\Omega) = |\Omega|$ denotes the volume of the shape Ω. The value of the Lagrange multiplier is refreshed at each iteration so the shape satisfies the volume constraint when the algorithm converges. Since moving the mesh is relatively costly, we do not enforce exactly the volume constraint before convergence. Instead, we increase the Lagrange multiplier if the current volume of the shape is greater than the target volume and we decrease it otherwise. Nevertheless, this could lead to oscillations of the shape's volume. Thus, we relax it with the value of the Lagrange multiplier computed by assuming that the optimality condition is satisfied, namely $J'(\Omega) + \ell V'(\Omega) = 0$, at least in the average sense on the boundary $\partial\Omega$, i.e. $\ell = -\int_{\partial\Omega} j(\Omega)\, ds / \int_{\partial\Omega} ds$ with the notations of (1). More precisely, the Lagrange multiplier is updated at each iteration by

$$\ell^{n+1} = (\ell^n + \ell)/2 + \epsilon_\ell (V - V_0),$$

where ϵ_ℓ is a small enough positive real number.

2.4.4 Stepping algorithm

The choice of the descent step h_n is not an easy task. Too big, the algorithm is unstable, too small the rate of convergence is insignificant. In order to refresh h_n, we compare at each iteration the current descent direction d_n with the previous one d_{n-1}. If the scalar product $(d_n, d_{n-1})_{H^1(\Omega)^N}$ is negative, we suspect that the algorithm is becoming unstable. In this case, we reduce the step and go backward: the next iteration is initialized with the previous shape Ω_{n-1}. On the other hand, if d_n and d_{n-1} are very close, the step h_n is increased. The step h_n is also decreased if reversed triangles are detected when the mesh is updated (see section 2.6.3).

2.4.5 Stopping criterion

A typical convergence criterion for stopping the optimization loop would be to check that the shape derivative $J'(\Omega)$ is small enough in some appropriate norm. However, since we use continuous gradients (and not discrete ones), it is hopeless to expect very small gradient norm because of numerical discretization errors. There is a more serious obstacle to using a strict convergence criterion which is linked to the inability of changing the topology. Indeed, it happens quite frequently that two different parts of the shape boundary tend to merge. In such a case, the descent step is decreased in order to avoid reversed triangles after moving the mesh (see Section 2.6.3). It can become almost zero even if the shape derivative $J'(\Omega)$ is large and the optimal shape is not reached. Consequently, we do not use any convergence criterion to stop the algorithm. We instead fix the number of iterations at the beginning of the algorithm. If it is too small, we can always restart it with the previous final shape as initial guess.

2.5 Numerical Applications for the boundary variation method

The above algorithm has been successfully tested in $N = 2$ space dimensions for different kinds of structural optimization problems, in particular on the classical cantilever and grip optimization.

2.5.1 Compliance optimization

Let Ω be the reference configuration of a linear isotropic elastic body. We assume that Ω is fixed on Γ_D, submitted to surface forces g on Γ_N and free on Γ_{opt}, where $\partial\Omega = \Gamma_D \cup \Gamma_N \cup \Gamma_{opt}$. The displacement of the structure $u(\Omega)$ is the solution of the linear elasticity system

$$\begin{cases} \mathrm{div}(Ae(u(\Omega))) = 0 \text{ in } \Omega, \\ (Ae(u(\Omega))n = g \quad \text{on } \Gamma_N, \\ (Ae(u(\Omega))n = 0 \quad \text{on } \Gamma_{opt}, \\ u(\Omega) = 0 \quad \text{on } \Gamma_D, \end{cases} \tag{3}$$

where $e(u) = (\nabla u + \nabla u^T)/2$ is the strain tensor, and n is the outward normal to the boundary, A is the Hooke's law or elasticity tensor defined by

$$A\xi = 2\mu\xi + \lambda(\mathrm{Tr}\,\xi)\,\mathrm{Id} \tag{4}$$

with Lamé moduli λ and μ. The variational formulation of (3), which has to be implemented in the `FreeFem++` software (see section 2.6.2), is

$$\int_\Omega (2\mu e(u(\Omega)) \cdot e(q) + \lambda\,\mathrm{div}\,u(\Omega)\,\mathrm{div}\,q)\,dx = \int_{\Gamma_N} g \cdot q\,ds$$

for every $q \in H^1(\Omega)^2$ such that $q = 0$ on Γ_D. We consider the compliance minimization problem

$$\min_\Omega c(\Omega) = \int_{\Gamma_N} g \cdot u(\Omega)\,ds,$$

over all open sets Ω such that $\Gamma_N \cup \Gamma_D \subset \partial\Omega$, with prescribed volume V_0. The functional c admits a shape derivative (see e.g. (Allaire et al. 2004))

$$< c'(\Omega), \theta >= -\int_{\Gamma_{opt}} (2\mu|e(u(\Omega))|^2 + \lambda(\mathrm{div}\,u(\Omega))^2)(\theta \cdot n)ds. \tag{5}$$

2.5.2 A Numerical example: The cantilever

To illustrate the performance of our `FreeFem++` script, we consider the compliance minimization of the following cantilever.

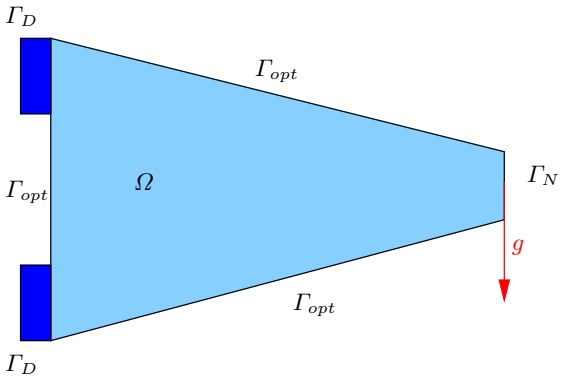

Fig. 1 Initial shape

The algorithm is initialized with the shape shown on figure 1 to which several holes have been added. The resulting optimal designs are displayed on figure (2- 7). As expected, the compliance decreases as the number of holes grows. Nevertheless, the benefit becomes very weak starting from three holes.

	Initial shape	No hole	1 hole
Compliance	40.5	23.3	19.6

	2 holes	3 holes	5 holes	6 holes
Compliance	18.3	17.57	17.51	17.47

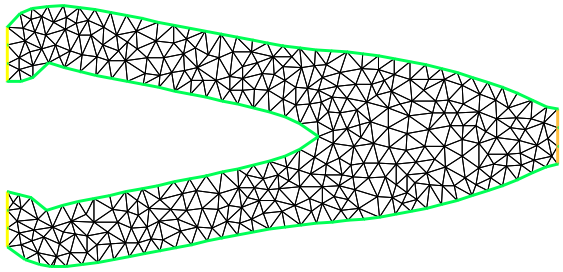

Fig. 2 Optimal design with no hole

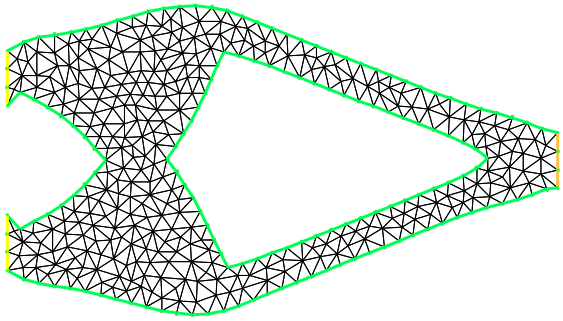

Fig. 3 Optimal design with one hole

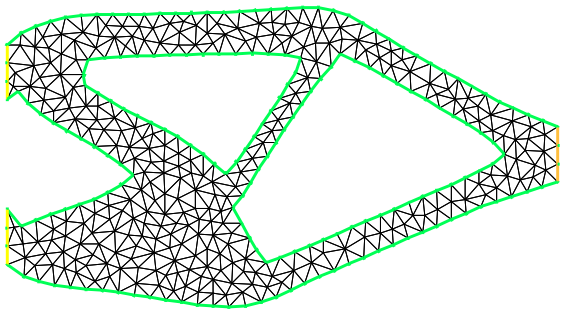

Fig. 4 Optimal design with two holes

2.5.3 Grip optimization

We consider the optimization of a gripping mechanism made of a linear elastic material. We use the same no-

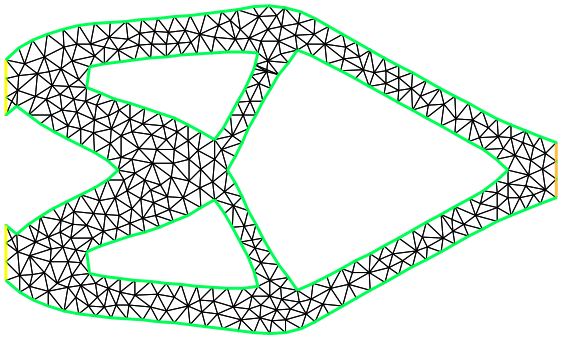

Fig. 5 Optimal design with three holes

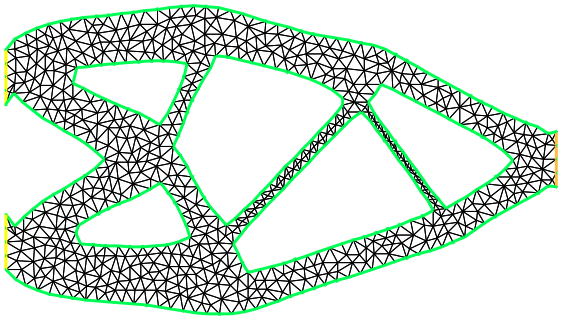

Fig. 6 Optimal design with five holes

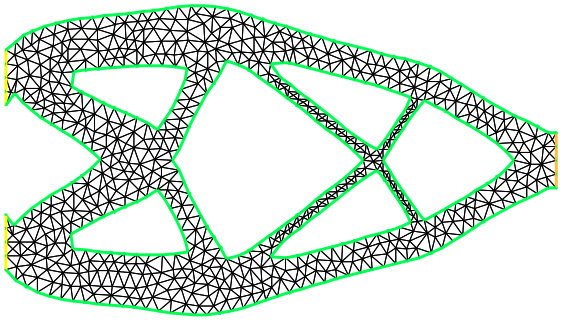

Fig. 7 Optimal design with six holes

tations than those of section 2.5.1. Nevertheless, we assume that the boundary $\partial\Omega$ of the structure splits in four parts. The grip is fixed on Γ_D, submitted to surface loads on Γ_N, free on Γ_{opt}, and in bilateral contact on Γ_c where the piece to grip lies (see figure 8). The displacement $u(\Omega)$ of the grip is the solution of the following variational formulation

$$\text{Find } u(\Omega) \in W(\Omega) := \{q \in H^1(\Omega)^2 \ : \ q = 0 \text{ on } \Gamma_D, \text{ and } q \cdot n = 0 \text{ on } \Gamma_c\},$$

such that for every test function $v \in W(\Omega)$,

$$\int_\Omega 2\mu e(u(\Omega)) \cdot e(v) + \lambda \operatorname{div} u(\Omega) \operatorname{div} v dx = \int_{\Gamma_N} g \cdot v ds.$$

Remark 3 We assume that the contact between the grip and the body is bilateral. It would be more realistic to consider an unilateral contact instead, but it would lead to a non linear, and thus much more difficult, problem.

We want to maximize the pressure $P(\Omega)$

$$P(\Omega) = -\int_{\Gamma_c} (\sigma(\Omega)n) \cdot n \, ds,$$

of the grip on the piece, where $\sigma(\Omega)$ is the stress tensor,

$$\sigma(\Omega) = 2\mu e(u(\Omega)) + \lambda \operatorname{div}(u(\Omega)) \operatorname{Id}.$$

The pressure $P(\Omega)$ can be rewritten as

$$P(\Omega) = -\int_\Omega 2\mu e(u(\Omega)) \cdot e(u_c) + \lambda \operatorname{div} u(\Omega) \operatorname{div} u_c \, dx,$$

where u_c is a vector field in $H^1(\mathbb{R}^2)$ such that $u_c = 0$ on Γ_D and $u_c = n$ on Γ_c. The maximization of $P(\Omega)$ usually leads to disconnected structures. In order to avoid this problem, we instead minimize the functional

$$G(\Omega) = -P(\Omega)/c(\Omega) + \ell|\Omega|.$$

For a given positive pressure $P(\Omega)$ of the grip on the piece, it is advantageous to minimize the compliance $c(\Omega)$ in order to minimize the functional $G(\Omega)$. Thus, we hope that disconnected structures (for which the compliance is typically very high) are not reasonable minimizers of G. Moreover, we add a penalization of the volume in the objective function G (ℓ is a positive real). Hence the minimizers of G are not excessively fat. The shape derivative of P (and thus of G) is easy to compute (see e.g. (Allaire et al. 2004))

$$< P'(\Omega), \theta >= \int_{\Gamma_{opt}} \Big(2\mu e(p(\Omega) - u_c) \cdot e(u(\Omega)) + \lambda \operatorname{div}(p(\Omega) - u_c) \operatorname{div} u(\Omega) \Big) (\theta \cdot n) dx,$$

where $\theta = 0$ on $\partial\Omega \setminus \Gamma_{opt}$. The expression of $P'(\Omega)$ relies on an adjoint state $p(\Omega) \in W(\Omega)$ defined as the solution of

$$\int_\Omega (2\mu e(p(\Omega)) \cdot e(q) + \lambda \operatorname{div} p(\Omega) \operatorname{div} q) \, dx = \int_\Omega (2\mu e(u_c) \cdot e(q) + \lambda \operatorname{div} u_c \operatorname{div} q) \, dx$$

for every $q \in W(\Omega)$.

2.5.4
Numerical example of the optimization of a grip

To illustrate the performance of our algorithm, we consider the optimization of the grip shown on figure 8,

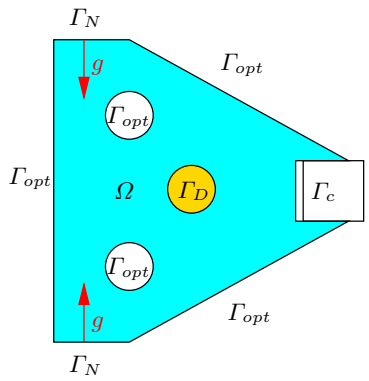

Fig. 8 Initial grip's shape

where the white square is the piece to be gripped by the jaws of the elastic structure Ω. The next figures display the optimal shape (figure 9) and the deformation of the grip when the piece between the jaws is removed (figure 10).

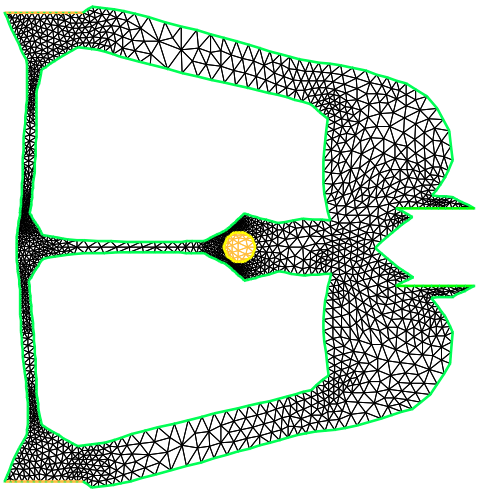

Fig. 9 Optimal design of a grip

2.6 Practical implementation

The boundary variation algorithm is easy to implement in $N = 2$ space dimensions with `FreeFem++`. In this section, we present the main parts of a `FreeFem++` script for the compliance minimization example (see section 2.5.1). For the sake of simplicity, the details presented in section 2.4 have not been included. In any case, our complete `FreeFem++` scripts are available on the web page `http://www. cmap.polytechnique.fr/~optopo`. More informations on the script syntax of `FreeFem++` can be found in (Hecht et al. 2005).

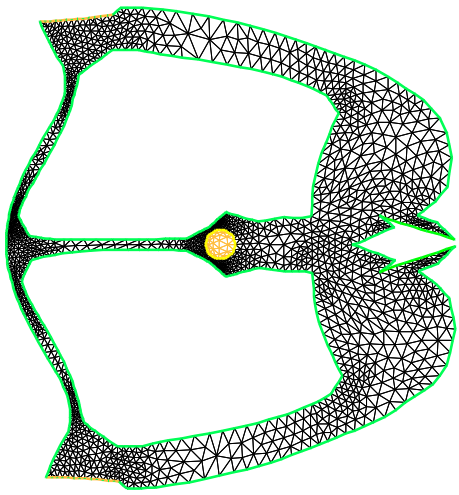

Fig. 10 Deformation of the optimal grip

2.6.1 Initialization

The initial shape Ω_0 is explicitly built, after having defined its boundaries. Each boundary is tagged by a label, according to the type of supported boundary conditions and to its mobility (fixed or not) during the optimization process. Each boundary is parametrized and oriented by a real t. As usual with mesh generators, the user must take care of a proper junction and orientation of the different parts of the boundary. The shape is meshed by triangles: the resulting mesh is denoted by `Sh` in the following code lines.

```
mesh Sh;
//Definition of the boundary labels
int neumann=1; int dirichlet=2; int free=3;
//right boundary (neumann condition)
border a(t=-1,1) {x=20; y=t;label=neumann;};
//left boundary (free or dirichlet condition)
border c1(t=4,2)  {x=0; y=t;label=dirichlet;};
border c2(t=2,-2) {x=0; y=t;label=free;};
border c3(t=-2,-4){x=0; y=t;label=dirichlet;};
//upper and lower boundary (free condition)
border b(t=1,0)
            {x=20.*t;y=4.-3.*t;label=free;};
border d(t=0,1)
            {x=20.*t; y=-4.+3.*t;label=free;};
//a circular hole
border Hole1(t=0,2*pi)
    {x=0.2*cos(t)+8;y=0.2*sin(t);label=free;};
Sh = buildmesh (b(30)+c3(10)+c2(10)
              +c1(10)+d(30)+a(20)+Hole1(-11));
```

2.6.2 Definition of the state equation

In the case of compliance optimization, the state or displacement u is the solution of the linear elasticity equations (3). We use $P2 \times P2$ Lagrange finite elements to compute u. The elasticity problem is described by its variational formulation

```
//Material parameters
real E=15;                      //Young Modulus
real nu=0.35;                   //Poisson ratio
//Lame Moduli
real lambda=E*nu/((1.+nu)*(1.-2.*nu));
real mu=E/(2.*(1.+nu));
//Applied Loads
func g1=0.; func g2=-1.;
//Finite elements space on the finer mesh Sh
fespace WSh(Sh,[P2,P2]);
//Displacement and test functions
WSh [u1,u2],[v1,v2];
////////////////////////////////
// Elasticity problem         //
// (variational formulation) //
////////////////////////////////
problem elasticity([u1,u2],[v1,v2]) =
 int2d(Sh)
  (
    2.*mu*(dx(u1)*dx(v1)+dy(u2)*dy(v2)
    +((dx(u2)+dy(u1))*(dx(v2)+dy(v1)))/2.)
    +lambda*(dx(u1)+dy(u2))*(dx(v1)+dy(v2))
  )
 -int1d(Sh,neumann)(g1*v1+g2*v2)
 +on(dirichlet,u1=0,u2=0);
```

The adjoint problem (if any) and the extension problem (2) (which gives the descent direction $d = (d_1, d_2)$) are defined in a similar way. The following code lines define the extension problem.

```
////////////////////////////////////
// Expression of the shape gradient //
////////////////////////////////////
macro gradientexp()
      -2.*mu*(dx(u1)^2+dy(u2)^2
      +((dx(u2)+dy(u1))^2)/2.)
      -lambda*(dx(u1)+dy(u2))^2
//extension field
WSh [d1,d2];
//test functions for the extension field
WSh [theta1,theta2];
//Lagrange multiplier
real lagrange;
////////////////////////
// Extension problem //
////////////////////////
// H1 scalar product between
// vector-valued functions
macro prodscal(t1,t2,p1,p2)
     dx(t1)*dx(p1)+dy(t1)*dy(p1)
  +dx(t2)*dx(p2)+dy(t2)*dy(p2)+t1*p1+t2*p2
//
problem extension([d1,d2],[theta1,theta2]) =
   int2d(Sh)(prodscal(d1,d2,theta1,theta2))
  +int1d(Sh,free)( (theta1*N.x+theta2*N.y)
                  *(gradientexp + lagrange))
   +on(dirichlet,neumann,d1=0,d2=0);
```

2.6.3 The optimization loop

The optimization loop consists in three main steps: resolution of the state equation, computation of the descent direction in terms of the shape derivative, shape updating.

```
int niter=100;  //Number of iterations
real step=0.1;  //Initial step
//Target volume
real volume0=95.;
//Refreshing step of the Lagrange multiplier
real lagrangestep=3;
//Initialization of the Lagrange multiplier
real volume,perimeter;
elasticity;
volume=int1d(Sh)(x*N.x+y*N.y)/2;
perimeter=int1d(Sh,free)(1.);
lagrange=
       int1d(Sh,free)(gradientexp)/perimeter;
int iter;
////////////////////////////////
//     Optimization Loop      //
////////////////////////////////
for (iter=0;iter< niter;iter=iter+1){
//Solving the state equation
elasticity;
//Update of the Lagrange multiplier
volume=int1d(Sh)(x*N.x+y*N.y)/2;
perimeter=int1d(Sh,free)(1.);
lagrange=0.5*lagrange
   -0.5*int1d(Sh,free)(gradientexp)/perimeter
   +lagrangestep*(volume-volume0)/volume0;
//Computation of the descent direction
extension;
//Update of the shape
Sh = movemesh (Sh,[x+step*d1,y+step*d2]);
plot(Sh);
};
///// END OF THE OPTIMIZATION LOOP /////
plot(Sh,wait=1); //Display the final shape
```

An additional subroutine of FreeFem++, `checkmovemesh`, allows to detect any occurrence of reversed triangles and thus make sure that the mesh is always conforming. Another subroutine, `adaptmesh`, is dedicated to mesh adaptation (i.e. the mesh is automatically refined where large

errors are detected) which greatly improves the efficiency of the computation. Actually mesh adaptation is necessary to prevent mesh degeneracy. Thus, for an effective algorithm it is recommended to insert the following code line at the end of the optimization loop, just before the update of the shape (`movemesh` command)

```
Sh = adaptmesh(Sh,[u1,u2]);
```

Merging together the above script parts yield a 106 lines optimization routine (including comments) that already works nicely. Of course, for a truly efficient routine one needs to add all the refinements given in section 2.4 or to download the scripts on our web page.

3 Topology optimization using homogenization

One limitation of the previous method is its incapacity to change the topology of the structure. Moreover, even for a given topology, the final result depends heavily on the initial shape. In particular, if holes are too close in the initial shape, they tend to merge, and the procedure is more or less stucked, as the thin layer between the holes can not be removed. Overall the previous method usually yields local minima which may be far from global ones. The homogenization method is aimed at optimizing the topology and getting global minima. It relies on the introduction of composite materials which can be shown to yield optimal shapes in a relaxed or generalized sense. Numerically, a composite optimal shape is first computed and then projected on the set of "real" shapes by a penalization procedure which removes the "grey" areas of intermediate densities. In the following, we focus on the minimization of a weighted sum of the compliance and the weight of the structure. Furthermore, we assume that the structure is made of a linear elastic isotropic material and included in a fixed working domain Ω.

3.1 Setting of the initial problem

Let Ω be a fixed bounded working domain in $\mathbb{R}^N$. Let Γ_0, Γ_N and Γ_D be a partition of the boundary of Ω. As previously, $u(\omega)$ denotes the displacement of the structure $\omega \subset \Omega$ which is assumed to be clamped on Γ_D and submitted to surface loads g on Γ_N

$$\begin{cases} \operatorname{div}\sigma = 0 \text{ in } \omega, \text{ with } \sigma = Ae(u), \\ \sigma n = g \quad \text{on } \Gamma_N, \text{ and } \sigma n = 0 \text{ on } \partial\omega \setminus (\Gamma_N \cup \Gamma_D) \\ u = 0 \quad \text{on } \Gamma_D, \end{cases}$$

where $e(u) = (\nabla u + \nabla u^T)/2$ and A is defined by (4). We consider the following compliance minimization problem

$$\min_{\omega \subset \Omega} \int_{\Gamma_N} g \cdot u(\omega)\, dx + \ell|\omega|, \tag{6}$$

where $\ell > 0$ is a Lagrange multiplier for the volume constraint. It can be proved that this problem usually admits no minimizer. The number of holes of a minimizing sequences of (6) increases to infinity, and thus, it does not converge to a classical shape, but to a composite one. The principle of the homogenization method is to extend the minimization problem to such generalized or composite shapes.

3.2 The homogenized problem

The composite shape is described by two variables, the material density $\theta(x) : \Omega \to (0,1)$ and the homogenized Hooke's tensor $A^*(x)$, which represents the underlying micro-structure (the shape of the holes). The displacement of the composite structure is the solution of

$$\begin{cases} \operatorname{div}\sigma = 0 \text{ in } \Omega, \text{ with } \sigma = A^* e(u), \\ \sigma n = g \quad \text{on } \Gamma_N, \text{ and } \sigma n = 0 \text{ on } \Gamma_0 \\ u = 0 \quad \text{on } \Gamma_D. \end{cases} \tag{7}$$

The homogenized problem is defined as

$$\min J(\theta, A^*) = c(\theta, A^*) + \ell \int_\Omega \theta dx, \tag{8}$$

where the minimization takes place over all couples (θ, A^*) such that A^* is a homogenized Hooke's law corresponding to a material density θ. The compliance of the composite structure is defined by

$$c(\theta, A^*) = \int_{\Gamma_N} g \cdot u\, ds = \int_\Omega A^{*-1}\sigma \cdot \sigma\, dx.$$

The minimum is reached by special composites called sequential laminates, obtained as successive layerings of void and material in N orthogonal directions and with adequate proportions. The directions of lamination are given by the eigenvectors of the stress tensor σ. For example, in dimension $N = 2$, the optimal lamination proportions are

$$m_1 = \frac{|\sigma_2|}{|\sigma_1| + |\sigma_2|},\ m_2 = \frac{|\sigma_1|}{|\sigma_1| + |\sigma_2|}, \tag{9}$$

where σ_1 and σ_2 are the eigenvalues of σ. Then, the homogenized Hooke's law of the optimal composite is

$$A^{*-1} = A^{-1} + \frac{1-\theta}{\theta}\left(m_1 f_A^c(e_1) + m_2 f_A^c(e_2)\right)^{-1}, \tag{10}$$

where e_1 and e_2 are the unitary eigenvectors of σ and $f_A^c(e_i)$ are fourth-order tensors defined for any symmetric matrix ξ by

$$f_A^c(e_i)\xi \cdot \xi = A\xi \cdot \xi - \mu^{-1}|A\xi|^2 + \frac{\mu+\lambda}{\mu(2\mu+\lambda)}((A\xi)e_i \cdot e_i)^2.$$

Finally, the optimal density θ is

$$\theta_{opt} = \min\left(1, \sqrt{\frac{\lambda+2\mu}{4\mu(\lambda+\mu)\ell}}(|\sigma_1| + |\sigma_2|)\right). \tag{11}$$

3.3
Numerical algorithm for the homogenization method

By using the principle of minimal complementary energy for the compliance

$$c(\theta, A^*) = \min_{\substack{\text{div}\,\tau=0 \text{ in } \Omega \\ \tau n = g \text{ on } \Gamma_N \\ \tau n = 0 \text{ on } \Gamma_0}} \int_\Omega A^{*-1}\tau \cdot \tau \, dx,$$

the minimization of the homogenized formulation (8) can be seen as an alternate minimization with respect to admissible stresses τ and design parameters (θ, A^*). Thus, we use the following "optimality criteria" algorithm

1. Initialization of the shape (θ_0, A_0^*).
2. Iteration until convergence for $n \geq 0$,
   (a) Compute the stress tensor σ_n by solving (7).
   (b) Compute the new design parameters $(\theta_{n+1}, A_{n+1}^*)$, using the optimality conditions (10-11).

Computing σ_n amounts to solve the linear elasticity system (7) in a displacement formulation and evaluate the resulting stress. The stress tensor is just required to update the design parameters. However, since the above algorithm is an alternate minimization, the objective function is always decreasing.

The resulting optimal design being composite, it is then "projected" on the set of classical shapes by applying again the previous scheme with the following slight modification: the density is updated setting $\theta_{n+1} = \theta_{pen}$, where

$$\theta_{pen} = \frac{1 - \cos(\pi\theta_{opt})}{2},$$

instead of $\theta_{n+1} = \theta_{opt}$. This has the tendency to progressively force the density to take only the values 0 or 1. The success of this penalization process can be explained by recalling that any minimizer of the homogenized problem (8) is attained as the limit of a minimizing sequence of the initial problem (6).

3.4
Practical implementation

The displacement of the structure is computed using $P2$ Lagrange finite elements. The stress tensor σ and the density θ are discretized either by $P1$ Lagrange finite elements or discontinuous $P1$ finite elements. Moreover, the mesh is adapted at each iteration. The composite optimal design is independent of the mesh, on the contrary of the penalized classical optimal design. Therefore, mesh adaptation is crucial here to be able to capture fine details even if the initial mesh was coarse. Finally, as A^* is not coercive, the Hooke's law A^* is replaced by $A^* + \varepsilon A$, with $\varepsilon = 10^{-3}$, in order to invert the stiffness matrix. We use the same basic operators of `FreeFem++` as in the case of geometric shape optimization, so we do not include `FreeFem++` scripts here. The interested reader can find the detailed routine on our web page.

3.5
Numerical results for the homogenization method

We apply the homogenization method to the same cantilever problem of section 2.5.2. The composite optimal design is displayed on figure 11: its compliance is 16.17.

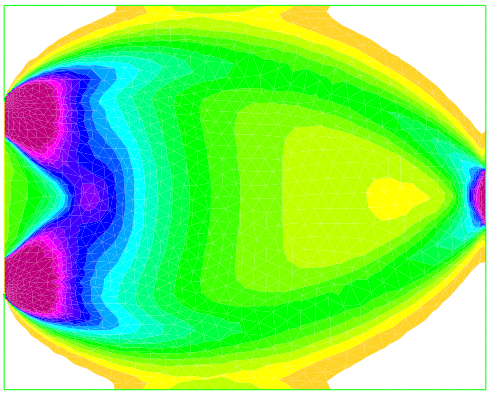

Fig. 11 Density of the optimal composite shape

The penalization step leads to a classical structure with compliance equal to 17.04 (see figure 12).

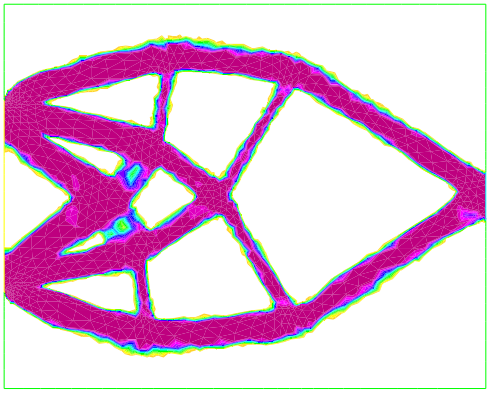

Fig. 12 Optimal shape obtained after penalization

To determine the accuracy of the homogenization method, we apply the boundary variation method to an initial

shape chosen so the final structure has the same topology and almost the same geometry, than the one obtained by homogenization (see figure 13). As illustrated by figure

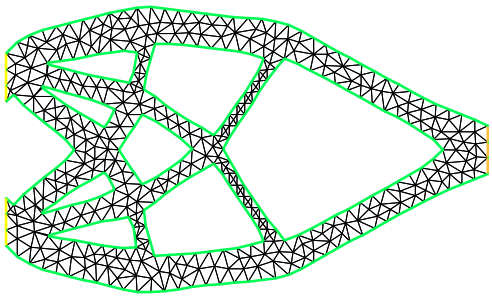

Fig. 13 Optimal shape obtained by geometric optimization

14, both cantilevers match almost exactly. Moreover, the

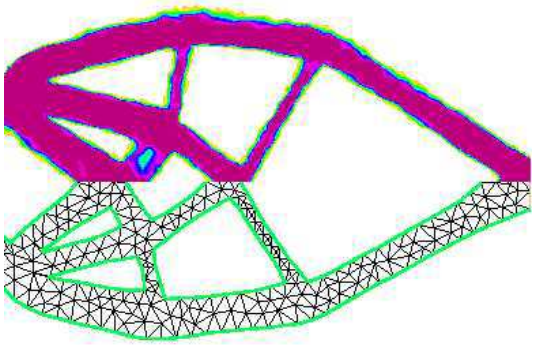

Fig. 14 comparison between the optimal design obtained with each method

compliances are very close: 17.07 for the solution of the boundary variation method compared to 17.04 for the homogenization.

However it is not completely fair to compare compliances for different shapes obtained with different methods because of numerical errors caused by the possibly different meshes and by the approximation of void by a soft material of the order of εA.

4 Conclusion

We have shown the suitability of `FreeFem++` for shape optimization in two dimensions. This free and user-friendly software does not require an heavy investment in programming and allows users to focus on the practical aspects of structural optimization. Our experience with graduate students is extremely positive: numerical homeworks or personal projects based on `FreeFem++` are both an excellent motivation and illustration of shape optimization. Eventually many other methods can be implemented with `FreeFem++`, including SIMP or layout optimization methods like the variable thickness sheet method.

References

Allaire G., *Shape optimization by the homogenization method*, Springer Verlag, New York (2001).

Allaire G., *Conception optimale de structures,* Editions de l'Ecole Polytechnique (2005).

Allaire G., Jouve F., Toader A.-M., Structural optimization using sensitivity analysis and a level-set method, J. Comp. Phys. Vol 194/1, pp.363-393 (2004).

Bendsoe M., Sigmund O., *Topology Optimization. Theory, Methods, and Applications*, Springer Verlag, New York (2003).

Hecht F., Pironneau O., Le Hyaric A., Ohtsuka K., *FreeFem++ Manual*, downloadable at `http://www.freefem.org`

Pironneau O., *Optimal shape design for elliptic systems*, Springer-Verlag, New York, (1984).

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

Sokołowski J., Zolesio J.P., *Introduction to shape optimization: shape sensitivity analysis*, Springer Series in Computational Mathematics, Vol. 10, Springer, Berlin (1992).