Chapter 6
Consistent Approximations and Approximate Gradients

Contents

Abstract

In this chapter we shall apply the theory of consistent approximations to speedup the numerical computation of optimal shapes. It was introduced by Polak (see Polak(1997)) to study how discretization parameters of an optimization problem could be refined within the algorithm loop. The same idea can be used here: can we do some work on a coarse grid, then use the solution as an initializer for a finer grid and so on ?
In the process we shall find the right spaces to fit the theory and understand the connection between the discrete and the continuous optimal shape design problems; we will discover which is the best norm and the best variable for shape design and in what sense the discrete gradients should converge to the continuous one.
Application to academic test cases and to real life wing design will be given.

1 Generalities

We wish to merge grid refinement into the optimization loop so as to do most of the optimization iterations on a coarse mesesh and few on the fine meshes. It cannot be done at random, some thoery is needed. To this end consider

$$\min_{z \in Z} J(z)$$

and its discretization (h could be the mesh size)

$$\min_{z \in Z_h} J_h(z).$$

where Z_h is a finite dimensional subspace of the Hilbert space Z.
Consider the method of steppest descent to achieve "precision ϵ" with Armijo's rule for the step size with parameters $0 < \alpha < \beta < 1$:

Algorithm 1 *(Steepest descent with Armijo's rule)*
. ***while*** $\|\text{grad}_z J_h(z^m)\| > \epsilon$ ***do***
. {
. $z^{m+1} = z^m - \rho \text{grad}_z J_h(z^m)$ *where* ρ *is any number satisfying, with* $w = \text{grad}_z J_h(z^m)$

$$. \quad -\beta\rho\|w\|^2 < J_h(z^m - \rho w) - J_h(z^m) < -\alpha\rho\|w\|^2$$

. $m := m + 1$;
. }

Now consider the same algorithm with grid refinement with parameter $\gamma > 1$

Algorithm 2 *(Steepest descent with Armijo's rule and mesh refinement)*

. ***while*** $h > h_{min}$ ***do***
. {
. ***while*** $\|\text{grad}_z J_h(z^m)\| > \epsilon h^\gamma$ ***do***
. {
. $z^{m+1} = z^m - \rho \text{grad}_z J_h(z^m)$ *where* ρ *is any positive real such that, with* $w = \text{grad}_z J_h(z^m)$

$$. \quad -\beta\rho\|w\|^2 < J_h(z^m - \rho w) - J_h(z^m) < -\alpha\rho\|w\|^2$$

. $m := m + 1;$
. }
. $h := h/2;$
. }

Convergence is fairly obvious because at each grid level the convergence is that of the gradient method and when the mesh size is decreased the norm of the gradient is required to decrease as well; so at least for the diagonal sequence m_j for which the test on the gradient fails, we have : $\text{grad}_z J_h(z^{m_j}) \to 0$ as $j \to \infty$. To prove the convergence of the entire sequence z^m is more difficult, even in strictly convex situations because nothing insures that J_h decreases when h is decreased.

Another possible gain in speed arise from the observation that we may not need to compute the exact gradient $\text{grad}_z J_h$! Indeed in many situations it involves solving PDEs and/or nonlinear equations by iterative schemes; can we stop the iterations earlier on coarse meshes ? To this end assume that N is an iteration parameter and that $J_{h,N}$ and $\text{grad}_z J_{h,N}$ denote approximations of J_h and $\text{grad}_z J_h$ in the sense that

$$\lim_{N\to\infty} J_{h,N}(z) = J_h(z) \quad \lim_{N\to\infty} \text{grad}_{zN} J_{h,N}(z) = \text{grad}_z J_h(z)$$

Non consider the following algorithm with additional parameter K and $N(h)$ with $N(h) \to \infty$ when $h \to 0$:

Algorithm 3 *(Steepest descent with Armijo's rule mesh refinement and approximate gradients)*

. ***while*** $h > h_m in$
. {
. ***while*** $|\text{grad}_{zN} J^m| > \epsilon h^\gamma$
. {
. ***try*** *to find a step size* ρ *with* $w = \text{grad}_{zN} J(z^m)$

. $-\beta\rho\|w\|^2 < J(z^m - \rho w) - J(z^m) < -\alpha\rho\|w\|^2$

. ***if*** *success* ***then***
. $\{z^{m+1} = z^m - \rho \text{grad}_{zN} J^m; \quad m := m + 1;\}$
. ***else*** $N := N + K;$

. }
. $h := h/2; \quad N := N(h);$
. }

The convergence could be established from the observation that Armijo's rule gives a bound on the step size:

$$-\beta\rho \mathrm{grad}_z J \cdot h < J(z+\rho h) - J(z) = \rho \mathrm{grad}_z J \cdot h + \frac{\rho^2}{2} J'' hh$$

$$\Rightarrow \quad \rho > 2(\beta - 1)\frac{\mathrm{grad}_z J \cdot h}{J''(\xi)hh}$$

so that

$$J^{m+1} - J^m < -2\frac{\alpha(1-\beta)}{\|J''\|}|\mathrm{grad}_z J|^2$$

Thus at each grid level the number of gradient iterations is bounded by $O(h^{-2\gamma})$. Therefore the algorithm does not jam and as before the norm of the gradient decreases with h.

a number of hypothesis are needed for the above such as C^1 continuity with respect to parameters, boundedness from below for J etc. To make the above more precise we recall the hypothesis made by E.Polak in [7].

2 Consistent Approximations

More precisely consider an optimization problem with linear constraints

$$(\mathcal{P}) \qquad \min_{z \in \mathcal{O}} J(z)$$

where $\mathcal{O}$, the set of admissible variables, is the subset of a Hilbert space $\mathcal{H}$ whose scalar product is denoted by $\langle , \rangle$.
Then consider a finite dimensional approximated problem

$$(\mathcal{P}_h) \qquad \min_{z_h \in \mathcal{O}_h} J_h(z_h)$$

Assume that $J : O \longrightarrow R$ and $J_h : O_h \longrightarrow R$ are continuous and differentiable.

The *Optimality Functions* of the problems are

$$\theta(z) = -\|\mathbf{P}\text{grad}_z J(z)\| \quad \theta_h(z) = -\|\mathbf{P}_h\text{grad}_z J_h(z)\|$$

where $\mathbf{P}$ and $\mathbf{P}_h$ are the projection operator from $\mathcal{H}$ to $\mathcal{O}$ and $\mathcal{H}$ to $\mathcal{O}_h$ respectively. Recall the definition of projectors, for instance

$$< \mathbf{P}z, w >=< z, w > \quad \forall w \in \mathcal{O}, \quad \mathbf{P}z \in \mathcal{O}$$

Notice that θ (resp θ_h) is always negative and if it is zero at $\tilde{z}$ then this point is a candidate optimizer for $\mathcal{P}$ (reps $\mathcal{P}_h$).

We assume that both θ and θ_h are at least sequentially upper-semi continuous in z.

2.1 Consistent Approximation

For an *optimality function* θ_h for $\mathcal{P}_h$ with similar properties, the pairs $\{\mathcal{P}_h, \theta_h\}$ are *consistent approximations* to the pair $(\mathcal{P}, \theta)$, if

1. For every $z \in \mathcal{O}$, there exist a sequence $\{z_h\}$ with $z_h \in \mathcal{O}_h$, such that $z_h \longrightarrow z$ and $\forall \{z_h\}_h$ such that $z_h \in \mathcal{O}_h$ and $z_h \longrightarrow \tilde{z}$, as $h \to 0$, we have: $\tilde{z} \in \mathcal{O}$ (*consistency*).
2. $J_h(z_h) \longrightarrow J(z)$, as $h \to 0$ (*continuity in h*)
3. $\forall$ $\{z_h\}_h$ of $\mathcal{O}_h$ converging to $\tilde{z}$, $\overline{\lim_{h\to 0}} \theta_h(z_h) \leq \theta(\tilde{z})$.

We summarize these hypotheses by the key-words, "continuity in z and h" for J and J_h, "upper semi continuity in z and h" for θ and θ_h, and "inclusion" and "consistency" for $\mathcal{O}_h$. Notice that the first two items are the definition of "epi-convergence" of $\mathcal{P}_h$ to $\mathcal{P}$.

Algorithm 4 *(Conceptual)*

1. *Choose a converging sequence of discretization spaces* $\{\mathcal{O}_{h_n}\}$ *with* $\mathcal{O}_{h_n} \subset \mathcal{O}_{h_{n+1}}$ *for all* n. *Choose* $z^0, \epsilon^0, \beta \in]0, 1[$.
2. *Set* $n = 0, \epsilon = \epsilon^0, h = h_0$

3. *Compute z_m^n by performing m iterations of a descent algorithm on $\mathcal{P}_h$ from the starting point z^n so as to achieve*

$$\theta_h(z_m^n) > -\epsilon$$

4. *Set $\epsilon = \beta\epsilon, h = h_{n+1}, z^{n+1} = z_m^n, n = n+1$ and go to Step 3.*

Theorem 1 *(Polak[7])*
If $\mathcal{P}_h$ are consistent approximations of $\mathcal{P}$ then any accumulation point z^* of $\{z^n\}$ generated by Algorithm 4 satisfies $\theta(z^*) = 0$.

3 Application to a Control Problem

For a bounded domain Ω with boundary Γ consider the following model problem, which is in the class of problems that arise when a transpiration condition is used to simplify an optimal shape design problem:

$$\min_{v \in L^2(\Gamma)} \{J(v) = \|u - u_d\|_1^2 : \quad u - \Delta u = f \text{ in } \Omega, \quad \frac{\partial u}{\partial n}|_\Gamma = v\}$$

where

$$\|u - u_d\|_1^2 = \int_\Omega [(u - u_d)^2 + |\nabla(u - u_d)|^2]$$

Discretized by triangular Finite Elements with $u \in V_h$ continuous piecewise linear on a triangulation $\mathcal{T}_h$:

$$\min_{v \in L_h} \{J_h(v) = \|u - u_{dh}\|_1^2 : \quad \int_\Omega (uw + \nabla u \nabla w) = \int_\Gamma vw, \ \forall w \in V_h\}$$

where L_h is the space of piecewise constant functions on Γ_h and where u_{dh} is an approximation of u_d in V_h (the interpolation of u_d for instance).

The variations of J with respect to variations in v are computed as follows (all equalities are up to higher order terms in δv):

$$\delta J = 2 \int_\Omega [(u - u_d)\delta u + \nabla(u - u_d) \cdot \nabla \delta u]$$

$$\text{with} \qquad \int_\Omega (\delta u w + \nabla \delta u \nabla w) = \int_\Gamma \delta v w, \ \forall w \in H^1(\Omega)$$

Therefore the "derivative" of J with respect to v is $J' = 2(u - u_d)$ because

$$\delta J = 2 \int_\Gamma \delta v (u - u_d).$$

The same computation is performed for J_h without change and leads to

$$\delta J_h = 2 \int_\Gamma \delta v_h (u_h - u_{dh}).$$

Let us define

$$\theta = -\|u - u_d\|_{0,\Gamma}, \quad \theta_h = - \sup_{v_h \in L_h:\ \|v_h\|_0 = 1} \int_\Gamma v_h (u_h - u_{dh}).$$

Remark 1 Note that if s_j is a segment of length $|s_j|$ of the discretization of Γ, θ_h is the L^2 norm of the piecewise constant function

$$\Theta_h = \frac{1}{|s_j|} \int_{s_j} (u_h - u_{dh}).$$

With the method of steepest descent, Algorithm 4 becomes

Algorithm 5 *(Control with mesh refinement)*
Assume that we have an automatic mesh generator which yields a triangulation of Ω in a deterministic manner once the discretization of the boundary Γ is chosen.

1. *Choose an initial segmentation h_0 of Γ. Set $v^0_{h_0,0} = 0, \epsilon^0 = 1, \beta = 0.5$ (for instance).*

2. *Set $n = 0, \epsilon = \epsilon^0, h = h_0$*

3. *Compute $v^n_{h,m}$ by performing m iterations of the type*

$$v^n_{h,m+1} = v^n_{h,m} - \rho(u^n_{h,m} - u_{dh})$$

where ρ is the best real number which minimizes J_h in the direction $u^n_{h,m} - u_{dh}$ and $u^n_{h,m}$ is the FEM solution of the PDE with $v^n_{h,m}$. until

$$(\sum_j (\frac{1}{|s_j|} \int_{s_j} (u_h - u_d))^2 |s_j|)^{1/2} < \epsilon$$

4. *Divide some segments on Γ. Set $\epsilon = \epsilon/2, v^{n+1}_{h,0} = v^n_{h,m}, n = n + 1$ and go to Step 3.*

3.1 Verification of the hypothesis

.

- *Inclusion:* When a segment of the discrete boundary is divided in two segment we do not have exactly

$$h' < h \Rightarrow L_{h'} \subset L_h$$

because the boundary is curved; but this is a minor technical point and we can consider that L_h refers to the curved boundary divided in segments and use an over parametric triangular finite element for the triangles which use these edges.

- *Continuity:* We have the following implications

$$\begin{aligned} v^n \overset{L^2(\Gamma)}{\longrightarrow} v, \quad \Rightarrow u^n \overset{H^1(\Omega)}{\longrightarrow} u, \Rightarrow \quad J(v^n) \to J(v) \\ \Rightarrow \quad \|u^n - u_d\|^2 \to \|u - u_d\|^2. \end{aligned}$$

So J, θ , J_h and θ_h are continuous with respect to v because

$$\begin{aligned} v_j^n \overset{\mathcal{R}}{\longrightarrow} v_j, \quad \Rightarrow u_h^n \overset{H^1(\Omega)}{\longrightarrow} u_h, \quad \Rightarrow J_h(v_h^n) \to J_h(v_h) \\ \Rightarrow \sup_{v_h \in L_h:\ \|v_h\|=1} \int_\Gamma v_h(u_h^n - u_{dh}) \to \sup_{v_h \in L_h:\ \|v_h\|=1} \int_\Gamma v_h(u_h - u_{dh}) \end{aligned}$$

- *Consistency* is obvious by simple discretization of Γ

$$\forall v \in L^2(\Gamma), \quad \exists \quad \{v_h\}, \quad v_h \to v \ : \ u_h \to u, \quad \|u_h - u_{dh}\|^2 \to \|u - u_d\|^2.$$

and also because if $v_h \in L_h \to v$ then $v \in L^2(\Gamma)$.

- *Continuity as* $h \to 0$ of J_h and of θ_h This is the most constringent property to verify usually but here it is easy by the convergence properties of the Finite Element method:

$$\begin{aligned} v_h \to v \Rightarrow u_h \to u \Rightarrow J_h(u_h) \to J(u) \\ \Rightarrow \sup_{v_h \in L_h:\ \|v_h\|=1} \int_\Gamma v_h(u_h - u_{dh}) \to \sup_{v \in L^2:\ \|v\|=1} \int_\Gamma v(u - u_d) \end{aligned}$$

Mesh\Iteration	$iter1$	$iter2$	$iter3$	$iter4$
$mesh1$	3818.53	1702.66	1218.47	
$mesh2$	469.488	285.803		
$mesh3$	113.584	74.4728		
$mesh4$	23.1994			
$mesh5$	7.86718			

Table 1: History of the convergence and values of the cost function

Remark 2 For simplicity we have purposely chosen an example which does not require an adjoint, but for those which do there is hardly any additional difficulty; for instance

$$J(v) = \|u - u_d\|_{0,\Omega} \Rightarrow J' = p|_\Gamma : \quad p - \Delta p = 2(u - u_d), \quad \frac{\partial p}{\partial n} = 0$$
$$v^n \overset{L^2(\Gamma)}{\longrightarrow} v, \quad \Rightarrow u^n \overset{H^1(\Omega)}{\longrightarrow} u, \quad \Rightarrow p^n \overset{H^1(\Omega)}{\longrightarrow} p \Rightarrow J'^n \to J'$$

3.2 Numerical Example

Consider an ellipse with a circular hole and boundary control on the hole. Table 1 shows the excellent performance of the method
Figures 1-4 show the triangulations generated each time the boundary is refined and Table 1 shows the values of the criteria for each iteration and each mesh. Algorithm 5 has performed 3 iterations on mesh 1, then went to mesh 2 because θ_h has decreased below the threshold ϵ; 2 iterations were sufficient on mesh 2, etc. The computing time saved is enormous when compared to the same done on the finest mesh only.

4 Application to Optimal Shape Design

Consider a simple model problem where the shape is to be found that brings u, solution of a PDE, nearest to u_d in a subregion D of the entire domain Ω. The unknown shape Γ is a portion of the entire boundary $\partial\Omega$: it is parametrized by its distance α to a reference smooth boundary Σ.

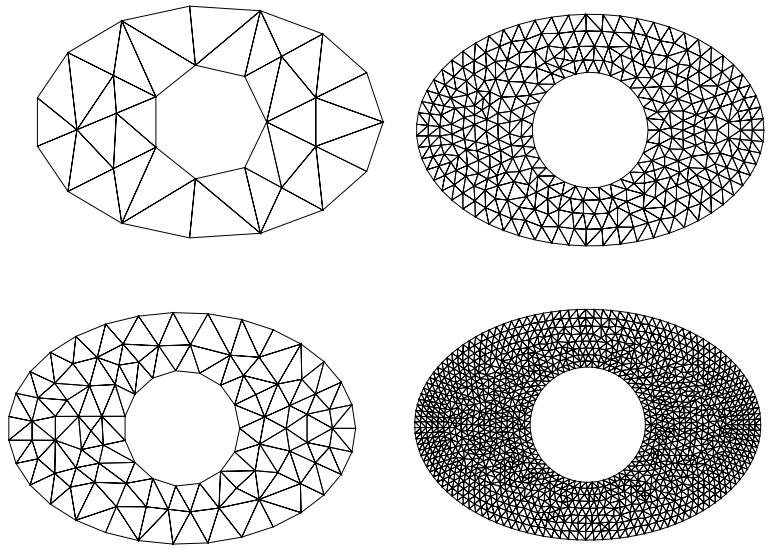

Figure 1: The first 4 meshes generated by algorithm 5

4.1 Problem Statement

More concretely given,

$$D \subset \Omega,\ u_d \in H^1(D),\ g \in H^1(\Omega),\ I \subset K \subset \mathcal{R},\ \ \Sigma = \{x(s)\ :\ s \in K\}$$

we consider

$$\begin{aligned} &\min_{\alpha \in H_0^2(I)} J(\alpha) = \int_D (u - u_d)^2 \qquad \text{subject to} \\ &u - \Delta u = 0 \ \text{ in } \ \Omega(\alpha), \quad \frac{\partial u}{\partial n}|_{\Gamma(\alpha)} = g, \\ &\text{where } \ \Gamma(\alpha) \equiv \partial\Omega(\alpha) = \{x(s) + \tilde{\alpha}(s) n(x(s))\ :\ s \in K\} \end{aligned}$$

where $\tilde{\alpha}$ is the extension by zero in K of α.

Recall that

$$H_0^2(I) = \{\alpha \in L^2(I)\ :\ \alpha', \alpha'' \in L^2(I),\ \ \alpha(a) = \alpha'(a) = 0\ \ \forall a \in \partial I\}$$

and that $\|\alpha''\|_0 = \|d^2\alpha/ds^2\|_0$ is a norm in that space.

Remark 3 *The problem may not have a unique solution. To insure uniqueness one may add a regularizing term and replace the cost function by*

$$J_\epsilon(\alpha) = \int_D (u - u_d)^2 + \epsilon \int_\Sigma |\frac{d^2\alpha}{ds^2}|^2$$

Let us denote the unknown part of the boundary by

$$S(\alpha) = \{x(s) + \alpha(s)n(x(s)) \ : \ s \in I\}$$

For simplicity let us assume that g is always zero on S.

4.2 Discretization

The discrete problem is

$$\min_{\alpha \in L_h \subset H_0^2(I)} J_\epsilon(\alpha) = \int_D (u - u_d)^2 + \epsilon \int_\Sigma |\frac{d^2\alpha}{ds^2}|^2$$
$$\text{subject to} \quad \int_{\Omega(\alpha)} (uv + \nabla u \nabla v) = \int_{\Gamma(\alpha)} gv \ \forall v \in V_h, \quad u \in V_h$$

where V_h is the usual Lagrange Finite Element space of degree 1 on triangles except that the boundary triangles have a curved side because in the following definition $S(\alpha)$ is a cubic spline.
The space L_h is the finite dimensional subspace of $H_0^2(I)$ defined as the set of cubic splines which passes through the vertices which would we would have used otherwise to define a feasible polygonal approximation of the boundary. This means that the discretization of Ω is done as follows

1. Give a set of n_f boundary vertices $q^{i_1}, ..., q^{i_{n_f}}$, construct a polygonal boundary near Σ.

2. Construct a triangulation of the domain inside this boundary with an automatic mesh generator, i.e. mathematically the inner nodes are assumed to be linked to the outer ones by a given map

$$q^j = Q^j(q^{i_1}, ..., q^{i_{n_f}}), \quad n_f < j < n_v$$

3. Construct $\Gamma(\alpha)$, the cubic splines from the $q^{i_1}, ..., q^{i_{n_f}}$, set α to be the normal distance from Σ to $\Gamma(\alpha)$.

4. Construct V_h by using triangular finite elements and overparametric curved triangular elements on the boundary.

This may seem complex but it is a handy construction for several reasons. Among other things, the discrete regularized cost function J_h coincide with theregularized continuous J and because L_h is a finite subspace of the (infinite) set of admissible parameters H_0^2.

4.3 Optimality Conditions: the continuous case

As before, by calculus of variations

$$\delta J = 2\int_D (u-u_d)\delta u + 2\epsilon \int_\Sigma \frac{d^2\alpha}{ds^2}\frac{d^2\delta\alpha}{ds^2}$$

with $\delta u \in H^1(\Omega(\alpha))$ and

$$\int_{\Omega(\alpha)} (\delta u v + \nabla \delta u \nabla v) + \int_\Sigma \delta\alpha(uv + \nabla u \nabla v) = 0 \ \forall v \in H^1(\Omega(\alpha)).$$

Introduce an adjoint $p \in H^1(\Omega(\alpha))$

$$\int_{\Omega(\alpha)} (pq + \nabla p \nabla q) = 2\int_D (u-u_d)q, \quad \forall q \in H^1$$

i.e.

$$p - \Delta p = I_D u, \quad \frac{\partial p}{\partial n} = 0$$

Then

$$\delta J = -\int_\Sigma \delta\alpha(up + \nabla u \nabla p - 2\epsilon \frac{d^4\alpha}{ds^4})$$

4.3.1 Definition of θ

So we should take

$$\theta = -\|up + \nabla u \nabla p - 2\epsilon \frac{d^4\alpha}{ds^4}\|_{-2}$$

i.e. solve

$$\frac{d^4\Theta}{ds^4} = up + \nabla u \nabla p \ \ on \ \ I, \quad \Theta = \frac{d\Theta}{ds} = 0 \ \ on \ \ \partial I$$

and take $\theta = -\|\Theta'' + 2\epsilon\alpha''\|_{0,\Sigma}$.

4.4 Optimality Conditions: the discrete case

Let w^j be the hat function attached to vertex q^j. If some vertices q^j vary by δq_j we define

$$\delta q_h(x) = \sum_j \delta q_j w^j(x)$$

and we know that (Pironneau[1983])

$$\delta w^k = -\nabla w^k \cdot \delta q_h \qquad \int_{\delta\Omega} f = \int_\Omega \nabla \cdot (f \delta q_h) + o(|\delta q_h|)$$

Hence

$$J(\alpha + \delta\alpha) = 2\int_D (u_h - u_{dh})\delta u_h + 2\epsilon \int_\Sigma \frac{d^2\alpha}{ds^2}\frac{d^2\delta\alpha}{ds^2},$$

Furthermore and by definition of δu_h

$$\delta \sum_i u_i w^i = \sum_i (\delta u_i w^i + u_i \delta w^i) = \delta u_h + \delta q_h \cdot \nabla u_h$$

the partial variation δu_h is found by

$$\delta \int_{\Omega(\alpha)} (u_h w^j + \nabla u_h \nabla w^j) = \int_\Omega (\nabla \cdot (u w^j \delta q_h) + \delta u_h w^j + \nabla \delta u_h \nabla w^j)$$
$$+ \int_\Omega (u_h \delta q_h \cdot \nabla w^j + \nabla u_h \nabla \delta q_h \nabla w^j + u_h \delta w^j + \nabla u_h \nabla \delta w^j) = 0$$

Hence

$$\int_\Omega (\delta u_h w^j + \nabla \delta u_h \nabla w^j =$$
$$\int_\Omega (\nabla u_h (\nabla \delta q_h + \nabla \delta q_h^T)\nabla w^j - (u_h w^j + \nabla u_h \cdot \nabla w^j)\nabla \cdot \delta q_h)$$

So introduce an adjoint $p_h \in V_h$

$$\int_\Omega (p_h w^j + \nabla p_h \nabla w^j) = 2\int_D (u_h - u_{dh}) w^j \quad \forall j$$

And finally

$$\delta J_h = \int_\Omega (\nabla u_h (\nabla \delta q_h + \nabla \delta q_h^T)\nabla p_h - (u_h p_h + \nabla u_h \cdot \nabla p_h)\nabla \cdot \delta q_h) + 2\epsilon \int_\Sigma \frac{d^2\alpha}{ds^2}\frac{d^2\delta\alpha}{ds^2}$$

4.5 Definition of θ_h

Let $e^1 = (1,0)^T,\ e^2 = (0,1)^T$ be the coordinate vectors of R^2, let χ^j be the vector of R^2 of components

$$\chi^j_k = \int_\Omega (\nabla u_h (\nabla w^j e^k + (\nabla w^j e^k)^T)\nabla p_h - (u_h p_u + \nabla u_h \cdot \nabla p_h)\nabla \cdot w^j e^k.$$

Because the inner vertices are linked to the boundary ones by the maps Q^j, let us introduce

$$\xi^j_k = \chi^j_k + \sum_{q^i \notin \Gamma} \chi^i \partial_{q^j_k} Q^i.$$

Then obviously

$$\delta J = \sum_1^{n_f} \xi^j \cdot \delta q^j + 2\epsilon \int_\Sigma \frac{d^2\alpha}{ds^2}\frac{d^2\delta\alpha}{ds^2}$$

It is possible to find a β so as to express the first discrete sum as an integral on Σ of $\frac{d^2\beta}{ds^2}\frac{d^2\delta\alpha}{ds^2}$; it is some sort of variational problem in L_h:

$$\text{Find } \beta \in L_h \text{ such that } \int_\Sigma \frac{d^2\beta}{ds^2}\frac{d^2\lambda^j}{ds^2} = \xi^j \cdot n_\Sigma, \quad \forall j = 1, ..., n_f;$$

where λ^j is the cubic spline obtained by a unit normal variation of the boundary vertex q^j only.
Then the "derivative" of J_h is the function $\Theta_h \ : \ s \in I \to \beta''(s) + 2\epsilon\alpha''(s)$ and the function θ_h is

$$\theta_h = -\|\beta'' + 2\epsilon\alpha''\|_{L^2_0(I)}$$

4.6 Implementation Trick

This may be unnecessarily complicated in practice. A pragmatic summary of the above is that β is solution of a fourth order problem and that its second derivative is the gradient, so why not set a discrete fourth order problem on the normal component of the vertex themselves. In the case $\epsilon = 0$ this would be

$$\frac{1}{h^4}[q'^{j+2}_n - 4q'^{j+1}_n + 6q'^j_n - 4q'^{j-1}_n + q'^{j-2}_n] = \xi^j,$$
$$q'^0_n = q'^1_n = q'^{n_f-1}_n = q'^{n_f}_n = 0$$

and then the norm of the second derivative of the result for θ_h

$$\Theta_h \approx -(\sum_j \frac{1}{h^2}[q_n^{j+1} - 2q_n^j + q_n^{j-1}]^2)^{\frac{1}{2}}$$

Another way is to notice that instead of a smoother on q^j we can construct displacements which have the same effects: Given ξ let

$$-\Delta\vec{u} - \nabla(\nabla \cdot \vec{u}) = 0, \quad u|_\Sigma = \xi$$

and then let

$$-\Delta\vec{v} = \vec{u}, \quad \frac{\partial\vec{v}}{\partial n} = 0$$

Obviously

$$\xi \in H^s(\Sigma) \Rightarrow \vec{u} \in H^{s+1/2}(\Omega) \Rightarrow \vec{v}|_\Sigma \in H^{s+2}$$

So $\Theta = v|_\Sigma$ is a smoothed version of ξ to the right degree of smoothing. This discretization of this method is obvious and it has another advantage in that Θ being defined everywhere all mesh points can be moved in the direction Θ.

Algorithm 6 *(OSD with mesh refinement) An adaptation of Algorithm 1 to this case is*

1. *Choose an initial set of boundary vertices.*
2. *Construct a finite element mesh, construct the spline of the boundary.*
3. *Solve the discrete PDE and the discrete adjoint PDE.*
4. *compute Θ_h (or its approximation (cf. subsection above))*
5. *if $\theta_h = -\|\Theta_h\| > -\epsilon$ add points to the boundary mesh, update the parameters and go back to Step 2.*

4.7 Orientation

What the theory of consistent approximation has done for us is to

- Give us an indication about the right spaces for the parameters (smooth radius of curvatures because of H_0^2 and fixed slopes at fixed end points)

- Give us a "smoother" before updating the mesh (a fourth order smoother)
- Give us an error indicator to refine the mesh.

We need to verify all the hypothesis listed in the beginning of the chapter to make sure that Algorithm 6 converges. This is done in Appendix 1 of this chapter. It is based on a result which is interesting in itself because it shows that the motion of the inner vertices can be neglected when compared to the motion of the boundary nodes.

Lemma 1

$$\delta J_h = -\int_\Sigma (\delta q_h \cdot n_\Sigma (u_h p_h + \nabla u_h \nabla p_h) + 2\epsilon \frac{d^4 \alpha_h}{ds^4}) + O(h\delta q_h) + o(h)$$

Proof
We note that a change of variable $x = Q(X)$ in the following integral gives

$$\int_{Q^{-1}(\Omega)} ((u_h p_h + \nabla u_h (\nabla Q \nabla Q^T) \nabla p_h) \det \nabla Q^{-1} = \int_\Omega (u_h p_h + \nabla u_h \nabla p_h)$$

Take $X(x) = x + \delta q_h(x)$ then $\nabla Q = I - \nabla \delta q_h$, $\det \nabla Q^{-1} \approx 1 + \nabla \cdot \delta q_h$ so

$$\begin{aligned}
\delta J_h = \int_\Omega (\nabla u_h (\nabla \delta q_h + \nabla \delta q_h^T) \nabla p_h - (u_h p_h + \nabla u_h \cdot \nabla p_h) \nabla \cdot q_h + ... \\
= \sum_k \int_{Q^{-1}(T_k) \setminus T_k} (u_h p_h + \nabla u_h \nabla p_h) + ... \\
= -\int_\Sigma \delta q_h \cdot n_\Sigma (u_h p_h + \nabla u_h \nabla p_h) \\
- \sum_k \int_{\partial T_k - \Sigma} \delta q_h \cdot \vec{n} (\nabla u_h \cdot \nabla p_h) + o(h) + ...
\end{aligned}$$

The last integral involves jumps of $\nabla u_h \cdot \nabla p_h$ across the edges of the elements; error estimates for the finite element method shows that this is $O(h)$ because in the limit $h \to 0$ these jumps are zero.

4.8 Numerical example

4.8.1 Effect of the smoother

Consider first an inverse problem with incompressible inviscid flow modeled by

$$-\Delta \psi = 0 \ \text{ in } \Omega \qquad \psi_{in} = \psi_{out} = y \quad \psi_{y=0} = 0, \quad \psi_{top} = 1. \tag{1}$$

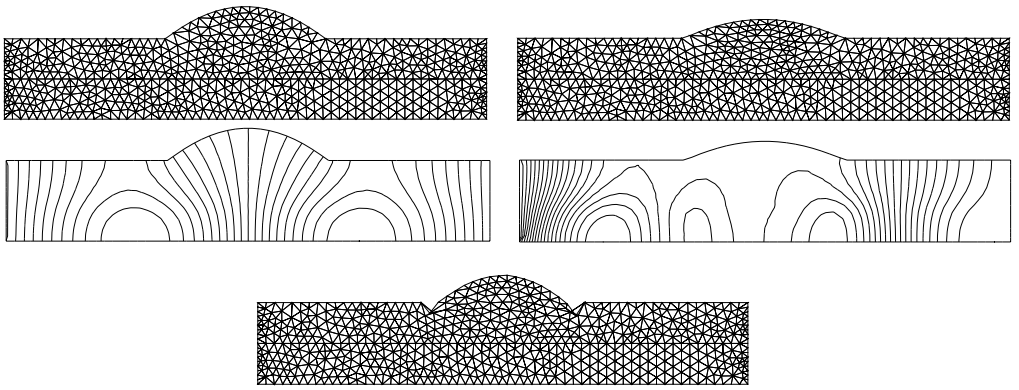

Figure 2: Effect of the smoother (i.e. a good choice for the spaces). Initial guess (left) and after 10 iterations. Without smoother after 4 iterations the mesh is no longer compatible (bottom). The level lines for inital guess and the final results are shown; while the first ones are in (0,0.1) the final ones are in (0,0.001).

When Ω is a rectangular pipe $\Omega^* = (0, L) \times (0, 1)$ then $\psi = y$ and the velocity u_d is uniformly equal to (1,0). Let $v_d = (0, 1)i$, D be $(0, L) \times (0, 1/2)$ and consider

$$\min_{\Omega} \int_D |\nabla \psi - v_d|^2 \text{ subject to (1) in } \Omega$$

Assume that only the part of the top boundary Σ in the interval $x \in (1/3, 2/3)$ varies. By starting the optimal shape algorithm with a non rectangular pipe we should recover the rectangular one. We apply the method without mesh refinement but with different scalar products to define the gradients. In one case we use the above theory, i.e. the scalar product of $H_0^2(\Sigma)$. In the second case we use $L^2(\Sigma)$, which amounts to use directly the variations of criteria without smoother. The results shown on figure 1 proves the necessity of the smoother; after 4 iterations the top boundary oscillates too much and the mesh is no longer compatible. In this case the mesh nodes are moved proportionally to the motion of the top boundary.

4.9 A Nozzle Optimization

4.9.1 Statement of the problem

Consider a symmetric nozzle of length $L > 0$. Incompressible potential flow is modeled by

$$\varepsilon\varphi - \Delta\varphi = 0 \text{ in } \Omega, \quad \left.\frac{\partial\varphi}{\partial n}\right|_{\Gamma} = g_{|\Gamma}, \tag{2}$$

where $0 < \varepsilon << 1$ is a regularizing constant used to avoid the singularity of Neumann conditions. The inflow and outflow velocities can be rescaled so that

$$g_{|\Gamma^1} = -1, \quad g_{|\Gamma^2} = \frac{|\Gamma^1|}{|\Gamma^2|}, \quad g_{|\Gamma^3 \cup \Gamma^4} = 0. \tag{3}$$

where $|\Gamma|$ denotes the length of Γ.
We consider the inverse problem of designing a nozzle that gives a flow as close as possible to a prescribed flow u_d in a subset of Ω, say in a given region $D = [0, L] \times [0, d]$. One way to fulfill this requirement is through the optimization problem

$$\min_{\Gamma^3 \in \mathcal{O}'} J_0(\Gamma^3) = \int_D \|\nabla\varphi - u_d\|^2, \tag{4}$$

where $\mathcal{O}'$ the set of admissible boundaries :

$$\mathcal{O}' = \{\Gamma^3 : D \cap \Gamma^3 = \emptyset,\, \Omega \subset \Omega_0, \partial\Gamma^3 \subset \partial\Gamma^4\}, \tag{5}$$

for some given security rectangle $\Omega_0 = (0, L) \times (0, H)$. The upper parts of Γ^4 are horizontal.
As stated, the set $\mathcal{O}'$ is too large and the problem may not have a solution. Let $I = (a, b)$ be an interval and let $\mathcal{O}$ be the set of admissible boundaries which have a representation $\{x, y(x)\}_{x \in \bar{I}}$ in the (x, y) plane:

$$\begin{aligned}\mathcal{O} = \{\Gamma^3_\alpha = \{x, \alpha(x)\}_{x \in \bar{I}},\, \alpha \in H^2(I) : d \le \alpha(x) \le H, \forall x \in I,\\ \alpha(a) = |\Gamma^1|, \alpha(b) = |\Gamma^2|, \alpha'(a) = \alpha'(b) = 0\}\end{aligned}$$

Note that $\mathcal{O}$ is a subspace of $H_0^2(I)$ and hence it inherits the Hilbert structure of $H_0^2(I)$.
We will minimize the function (6).
So our problem is

$$(\mathcal{P}) \qquad \min_{\Gamma^3_\alpha \in \mathcal{O}} J(\Gamma^3_\alpha) = \int_D |\nabla\varphi - u_d|^2 + \varepsilon \int_I \left|\frac{d^2\alpha}{dx^2}\right|^2 dx, \tag{6}$$

where φ is the solution of (2) and the second term is here to ensure existence of a solution to the optimization problem.

4.9.2 Computation of the Gâteaux derivative

In one dimension, $H^s \hookrightarrow C^k,\ s > k + 1/2$, hence all admissible α are C^1 at least. We will use a norm on α associated with the scalar product of $H_0^2(I)$:

$$\langle \alpha, \beta \rangle_{H_0^2(I)} = \int_I \frac{d^2\alpha}{dx^2}\frac{d^2\beta}{dx^2}dx, \quad \|\alpha\|_{H_0^2(I)} = \langle\ \alpha, \alpha\ \rangle^{\frac{1}{2}}_{H_0^2(I)} \tag{7}$$

Now, $\mathrm{grad}_\alpha J$ the $H_0^2(I)$ gradient of J is related to the Gâteaux derivative $J'_\alpha(\beta)$ by

$$J'_\alpha(\beta) = \frac{d}{d\lambda}J(\Gamma^3_{\alpha+\lambda\beta})|_{\lambda=0} = \langle\ \mathrm{grad}_\alpha J, \beta\ \rangle_{H_0^2(I)} = \int_I \frac{d^4}{dx^4}\mathrm{grad}_\alpha J.\beta = \int_I J'_\alpha.\beta,$$

hence $\mathrm{grad}_\alpha J$ is the solution of

$$\frac{d^4}{dx^4}\mathrm{grad}_\alpha J = J'_\alpha \text{ in } I \quad \mathrm{grad}_\alpha J = \frac{d}{dx}\mathrm{grad}_\alpha J = 0 \text{ at } x = a \text{ and } x = b, \tag{8}$$

where J'_α is defined by the following theorem.
This scalar product will force the regularity of the boundaries when an iterative process like the gradient method will be applied.

Theorem 2 *The function J'_α is given by*

$$J'_\alpha = -\frac{1}{n_2(\alpha)}(\varepsilon\varphi p + \nabla\varphi.\nabla p) + 2\varepsilon\frac{d^4\alpha}{dx^4}. \tag{9}$$

where $n_2(\alpha)$ is the y component of the exterior normal to Γ^3_α and $p \in H^1(\Omega)$ is the adjoint state, which is the solution of the Neumann problem

$$\int_\Omega (\varepsilon p\omega + \nabla p.\nabla\omega) = 2\int_D \nabla\omega.(\nabla\varphi - u_d)\,dxdy \qquad \forall\omega \in H^1(\Omega). \tag{10}$$

Proof. Let

$$\varphi'_\alpha(\beta) = \frac{d}{d\lambda}\varphi(\Gamma^3_{\alpha+\lambda\beta})|_{\lambda=0} \quad \in H^1(\Omega), \tag{11}$$

the Gâteaux derivative of φ. It is the solution of

$$\forall\omega \in H^1(\Omega), \int_\Omega (\varepsilon\varphi'_\alpha(\beta)\omega + \nabla\varphi'_\alpha(\beta).\nabla\omega)\,dxdy \tag{12}$$

$$= -\int_{\Gamma_\alpha} \beta(\varepsilon\varphi\omega + \nabla\varphi.\nabla\omega - \frac{\partial}{\partial n}(g\omega) + \frac{g\omega}{R})\,d\gamma \tag{13}$$

where R is the radius of curvature of Γ_α and $d\gamma$ the unit length element on the boundary of Γ_α. But in our case $g \equiv 0$ near and on Γ_α^3 and $\beta = 0$ on $\Gamma_\alpha^i, i \neq 3$.
Then the variation of J is written as follow

$$\frac{d}{d\lambda} J(\Gamma^3_{\alpha+\lambda\beta})|_{\lambda=0} = 2\int_D \nabla\varphi'_\alpha(\beta).(\nabla\varphi - u_d)\, dxdy + 2\int_I \varepsilon \frac{d^2\alpha}{dx^2}\frac{d^2\beta}{dx^2}\, dx, \quad (14)$$

Next, introduce p as above and replace ω by p in (10). As $\beta \in C^1$, we can use the mean value theorem for integrals and approximate the element of volume $dxdy$ by $\beta d\gamma$ thus using the definition of p with ω replaced by $\varphi'_\alpha(\beta)$ and (12):

$$\frac{d}{d\lambda} J(\Gamma^3_{\alpha+\lambda\beta})|_{\lambda=0} = -\int_{\Gamma^3_\alpha} \beta(\varepsilon\varphi p + \nabla\varphi.\nabla p)\, d\gamma + 2\int_I \varepsilon \frac{d^2\alpha}{dx^2}\frac{d^2\beta}{dx^2} \quad (15)$$

$$= \int_I \beta[2\varepsilon\frac{d^4\alpha}{dx^4} - \frac{1}{n_2(\alpha)}(\varepsilon\varphi p + \nabla\varphi.\nabla p)] \quad (16)$$

$\diamond$

4.10 Numerical results

We use a conjugate gradient in H_0^2 with mesh refinement. The initial mesh has 72 nodes, 110 triangles, 5 control points. At the end, after 4 refinements, the mesh is made of 9702 nodes, 18980 triangles, 80 control points. We also compare with the results of the direct computation : we optimize directly on the big mesh (9702 nodes). It appears that computation with refinement is better (Figure 4) but also faster than the direct computation (direct computation : 34h11m57.60s, iterative computation : 3h37m31.49s).

4.11 Drag reduction for an Airfoil

This is a drag reduction problem for Euler flow with constraints on the lift and the volume treated by penalty. The cost function is given by:

$$J(x) = C_d + \alpha|C_l - C_l^0| + \beta|Vol - Vol_0|,$$

where α and β are penalty parameters, C_d is the drag coefficient, C_l and C_l^0 are the actual and initial lift coefficients and Vol and Vol_0 the actual and initial volumes. The initial airfoil is a RAE2822. The design takes place at

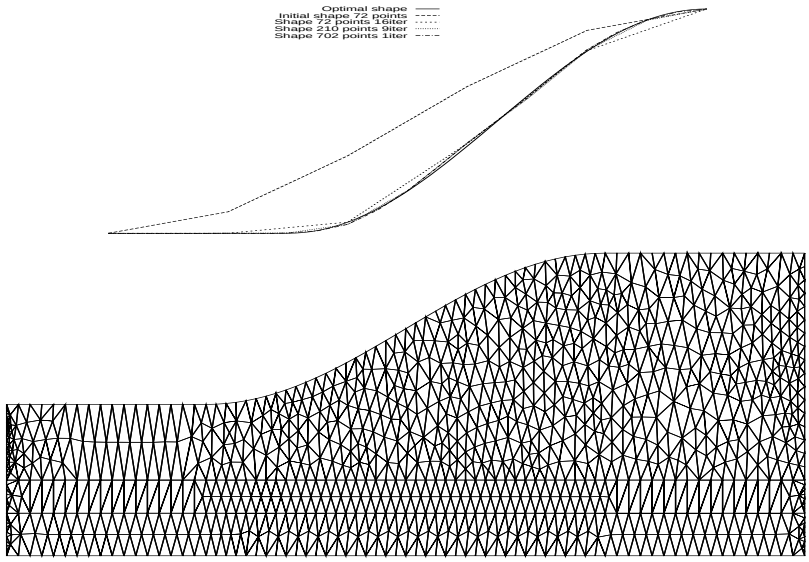

Figure 3: Some shapes generated by the algorithm on its way to convergence and the triangulation for one of them

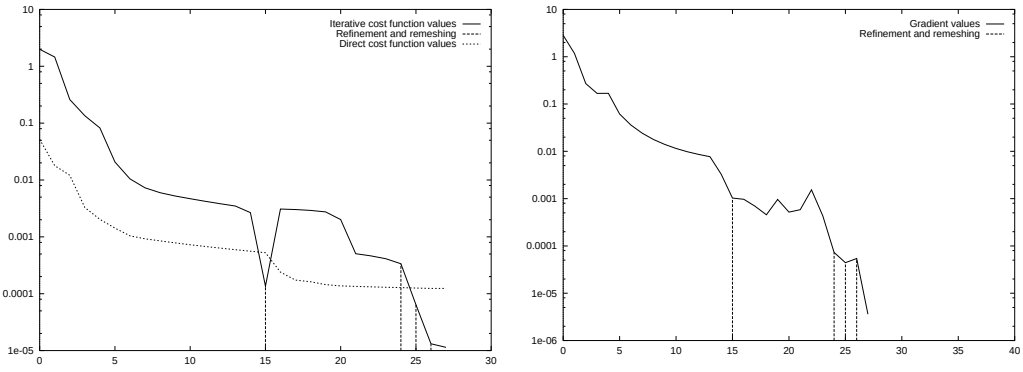

Figure 4: Versus iteration number, left: values of J, direct computation and computation with refinement. Right: Norm of the gradient

Mach number of 0.734 and Reynolds number of 6.510^6. The different ingredients are those described in chapter 5. A Delaunay mesh generator with adaption by the auxiliary metric techniques is used. The metric is based on the intersection of the metrics for the conservation variables as described in chapter 5.

With P1 discretization, fourth order elliptic problems need mixed formulations. Here, we used second order elliptic smoothers while the theory asks for a fourth order one. Furthermore, to avoid a global smoothing of the deformations, the smoothing has only been applied locally. More precisely, we solve, by Jacobi iterations, the following system:

$$(I - \varepsilon(x_w)\Delta)\delta\tilde{x}_w = \delta x_w, \tag{17}$$

$$\delta\tilde{x}_w = \delta x_w = 0 \quad \text{on wedges},$$

where $\delta\tilde{x}_w$ is the smoothed shape variation for the nodes which define the shape and δx_w is the variation given by the optimality conditions. In this computation, all the nodes on the discretized unknown shape are control points. We noticed that the scheme is improved if it is made 'local' (as described in chapter 5), meaning that if the predicted shape is locally smooth, it remains unchanged during this step, as it would be the case with the fourth order operator. In other words, $\varepsilon(x)$ is made to tends to zero in the Jacobi loop if

$$\frac{\delta_{ij}(\delta x_w)}{(\delta x_w)_T} < 10^{-3},$$

where $\delta_{ij}(\delta x_w)$ is the difference between the variations of the two nodes of each of the segments of a surface triangle (nodes of a boundary edge in 2D) and $(\delta x_w)_T$ the mean variation on this triangle (edge in 2D).

5 Approximate Gradients

In control theory there is a practical rule that it is always better to use the exact gradient of the discrete problem rather than a discretized version of the gradient of the continuous problem. But in some cases it is difficult or unnecessary expensive to compute the exact gradient of the discrete problem. For example we have seen that the in the cost function the variations due to the motions of the inner nodes are an order of magnitude smaller than those

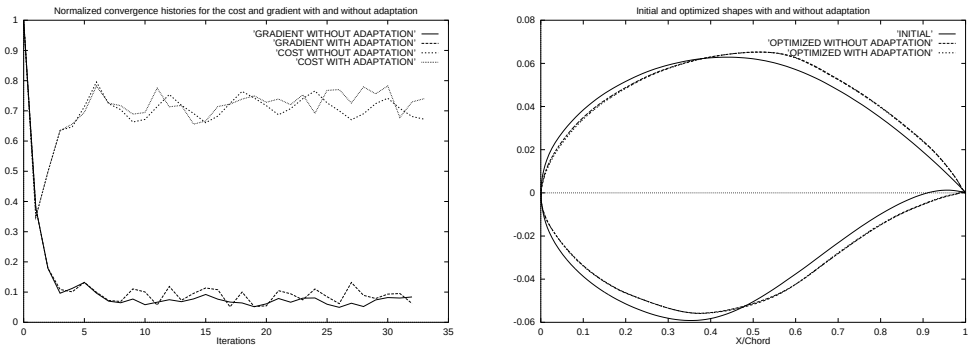

Figure 5: Shape optimization at transonic viscous turbulent regime: cost function and norm of gradients for the airfoil optimization for the optimization alone and when combined with adaptation (left) and the initial and final shapes for each approaches (right). With adaptation, we can be confident that the mesh is always adequate for intermediate states.

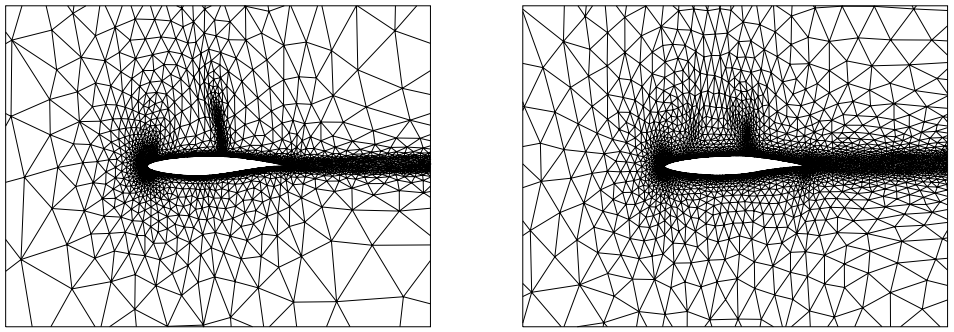

Figure 6: Shape optimization at transonic viscous turbulent regime: Initial and final meshes when using adaptation during optimization.

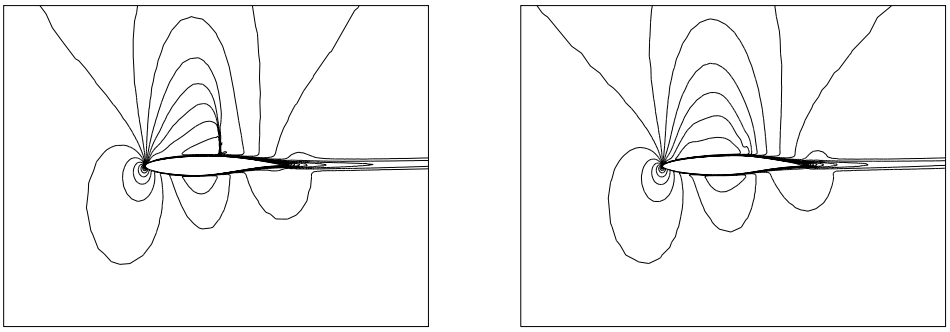

Figure 7: Shape optimization at transonic viscous turbulent regime: Initial and final iso-Mach contours for the previous meshes.

due to the boundary nodes. It would seem reasonable then to forget then. However if this is done too soon the algorithm may jam.
For nonlinear problems there would be a great speed up if we could link the number of iterations of the nonlinear solver with the optimization loop index. Another example comes from parallel computing. When the Schwarz algorithm is used to solves the PDEs it is not easy to compute the derivatives of the discretized problems. We analyze below one such problem in the simpler context of boundary control.

5.1 A control problem with Domain Decomposition

Consider the boundary control problem (S is a given subset of the boundary Γ of an open bounded subset of R^d, Ω)

$$\min_{v\in L^2(S)}\{J(v)=\int_\Omega[(u-u_d)^2+|\nabla(u-u_d)|^2]: \text{subject to}$$
$$u-\Delta u=0 \text{ in } \Omega,\quad \frac{\partial u}{\partial n}|_S=\xi v \quad u_{\Gamma-S}=u_d\} \tag{18}$$

The v-derivative of J is easy to obtain from

$$\delta J=2\int_\Omega((u-u_d)\delta u+\nabla(u-u_d)\cdot\nabla\delta u+o(|v|)=\int_S\xi(u-u_d)\delta v$$

This is because the PDE in variational form is

$$\int_\Omega (uw + \nabla u \cdot \nabla w) = \int_S \xi v w \quad \forall w \in H^1_{0_{\Gamma - S}}(\Omega)$$

To approximate the problem let us use a finite Element Method with $u \in V_h$ continuous piecewise linear on the triangles of a triangulation of Ω:

$$\min_{v \in V_h} J_h(v) = \int_\Omega [(u-u_d)^2 + |\nabla(u-u_d)|^2] : \quad \int_\Omega (uw + \nabla u \cdot \nabla w) = \int_S \xi v w \ \forall w \in V_h \tag{19}$$

where V_h is an approximation of $H^1_{0_{\Gamma - S}}(\Omega)$. Then the discrete optimality conditions are obtained in the same way

$$\delta J = \int_S \xi (u - u_d) \delta v$$

5.1.1 The Schwarz Algorithm

Let $\Omega = \Omega_1 \cup \Omega_2$, let $\Gamma = \partial\Omega$ and $\Gamma_{ij} = \partial\Omega_i \cap \Omega_j$. The multiplicative Schwarz algorithm for the Laplace equation starts from a guess u_1^0, u_2^0 and computes the solution of

$$u - \Delta u = f \text{ in } \Omega, \quad u|_\Gamma = u_\Gamma$$

as the limit in n of $u_i^n, i = 1, 2$ defined by

$$\begin{aligned}
& u_1^{n+1} - \Delta u_1^{n+1} = f \text{ in } \Omega_1, \\
& u^{n+1}|_{\Gamma \cap \overline{\Omega}_1 - S} = u_\Gamma \quad u_1^{n+1}|_{\Gamma_{12}} = u_2^n \quad \frac{\partial u_1^{n+1}}{\partial n}|_S = \xi v \\
& u_2^{n+1} - \Delta u_2^{n+1} = f \text{ in } \Omega_2, \\
& u^{n+1}|_{\Gamma \cap \overline{\Omega}_2 - S} = u_\Gamma \quad u^{n+1}|_{\Gamma_{21}} = u_1^n \quad \frac{\partial u_2^{n+1}}{\partial n}|_S = \xi v
\end{aligned}$$

5.1.2 Derivative of the Discrete Problem

The discretized problem is

$$\begin{aligned}
& \min_{v \in V_h} J_h^N(v) = \|u^N - u_d\|_\Omega^2 : \quad u_j^0 = 0, \quad n = 1..N \quad \forall w \in V_h \\
& u_j^n|_{\partial\Omega_{ij}} = u_j^{n-1} \quad \int_{\Omega_j} [u_j^n w + \nabla u_j^n \nabla w] = \int_S \xi v w
\end{aligned}$$

where N is the number of Schwarz iterations. The exact discrete optimality conditions of the discrete problems are:

$$p^N - \Delta p^N = 2(u^N - u_d) \quad p^{N-1} - \Delta p^{N-1} = 0 \quad p^{N-1}_{\Gamma_{ij}} = p^N ...$$

These are difficult to implement because we must store all intermediate functions generated by the Schwarz algorithm and integrate the system for p^n in the reverse order.
So here we will try to compute approximate optimality functions; the criteria for mesh refinement will be based on the v- derivative of J

$$\theta_h = -\|u - u_d\|_S$$

where u_h is computed by N iterations of the Schwarz algorithm.

5.1.3 Example

The algorithm model presented in the introduction is

Algorithm 7 *(Steepest descent with Armijo's rule mesh refinement and approximate gradients)*

. ***while*** $h > h_min$
. {
. ***while*** $|\text{grad}_{zN} J^m| > \epsilon h^\gamma$
. {
. ***try*** *to find a step size* ρ *with*

. $-\beta\rho|\text{grad}_{zN} J^m|^2 < J(z^m - \rho\text{grad}_{zN} J^m) - J(z^m) < -\alpha\rho|\text{grad}_{zN} J^m|^2$

. ***if*** *success* ***then***
. $\{z^{m+1} = z^m - \rho\text{grad}_{zN} J^m; \quad m := m + 1;\}$
. ***else*** $N := N + K;$
. }
. $h := h/2; \quad N := N(h);$
. }

Here z is the boundary control v, J for the cost function defined by (19) and the gradient $\text{grad}_{zN} J$ is $u - u_d$ computed by N iterations of the Schwarz algorithm.

Let us denote by a, b the boundary the domain Ω_1, c, d the boundary of Ω_2; a is inside Ω_2 and c is inside Ω_1, and let S be the boundary of a small circle. Functions named with capital letters are defined on Ω_1 others on Ω_2. The following gives a complete description of the algorithm used. The test for increasing the number of Schwarz algorithm [5] is based on the decrease of cost function being less than $0.1(0.8)^N$ and the mesh refinement, controlled by and integer giving $h = O(1/n)$, is based

i) either on the norm of the approximate gradient of J_h, the function θ_h being less than $\epsilon(n) = 10^{-n}$

ii) or on the decrease of the cost function being greater than $0.001(0.8)^n$.

Naturally other choices are possible.
We give below the program in freefem language[2] for the problem.

```
    N=1,eps=1,n=1, critest = 1e30.
    buildmesh th(n);
    buildmesh TH(n);// Build meshes for the two
                    // subdomains with C*n vertices.
    eps:=1; grad:= eps; n:=1; ii:=0; ud = exp(-x*sqrt(2))*sin(y);
    xi=sin(30*(x-1.15))+sin(30*(y-0.5)); v=0; u=0; U=0; p=0; P=0;
    N:=0;

for i=0 to itermax do {
  if(grad>=eps)then
  {
    U=10; u=1; P=10; p=0;
    for k:=0 to N do
    {
       solve(TH,U){ pde(U) U-laplace(U)=0; on(e)U=u; on(e1)U=ud};
       solve(th,u){ pde(u) u-laplace(u)=0; on(a) u=U;
                              on(a1,b,c)u=ud; on(S) dnu(u)=xi*v};
       solve(TH,P){ pde(P) P-laplace(P)=0; on(e) P=p; on(e1)P=0};
       solve(th,p){ pde(p) p-laplace(p)=0; on(a) p=P;
                         on(a1,b,c)p=0; on(S) dnu(p)= xi*(u-ud)};
    };
// uu and PP are  u and p on th and U and P on TH
```

```
    crit := int(Omega)((uu-ud)^2 +(dx(uu-ud))^2 +(dy(uu-ud))^2);
    w:= int(Omega((uu-ud)*pp+dx(uu-ud)*dx(pp)+dy(uu-ud)*dy(pp));
    z:= int(Omega)(pp^2+(dx(pp))^2+(dy(pp))^2);
    rho:=w/z;
  if(crit > critold-0.1*pow(0.8,N)) then {N:=N+1;  critold:=1e30}
   else {
            v = v - rho*xi*(uu-ud);
            grad:=int(S)(xi*xi*(uu-ud)^2);
            critold:=crit;
          };
  } else{
            grad:=eps;  n:=n+1;
            buildmesh th(n);  buildmesh TH(n);
            eps := eps/10; critold:=1e20;
        };
};
```

In this algorithm the optimal step size is computed exactly because the criteria is quadratic. Note that the integrals are computed on the whole domain so uu, pp denotes either u, p or U, P or $(u+U)/2, (p+P)/2$ depending if we are in Ω_1, Ω_2 or $\Omega_1 \cap \Omega_2$.

5.2 Numerical results

The domain is made of the unit circle plus quadrangle made by the rectangle $(0,3) \times (0,1)$ minus the unit triangle and minus a disk S. The function which is to be recovered by the optimization process is $u_d = e^{-x\sqrt{2}} sin(y)$. The weight on the control has been purposely chosen with oscillations: $\xi = \sin(30*(x-1.15)) + \sin(30*(y-0.5))$. We have an automatic mesh generator controlled by a parameter n and the number of points on the boundary are proportional to n. The number of Schwarz iterations are initialized at 1.
The results are shown on figure 8. After 30 iterations the gradient is 10^{-6} times its initial value, while without mesh refinement it has been divided by 100 only (multigrid effect).
Figure 9 shows the computed solution u (left) and the error $u - u_d$ (right).
Figure 10 shows the second mesh on the left and the 7th finite element mesh (the last is the 9th) on the right. Both are generated automatically by a Delaunay-Voronoi mesh generator from a uniform distribution of points on

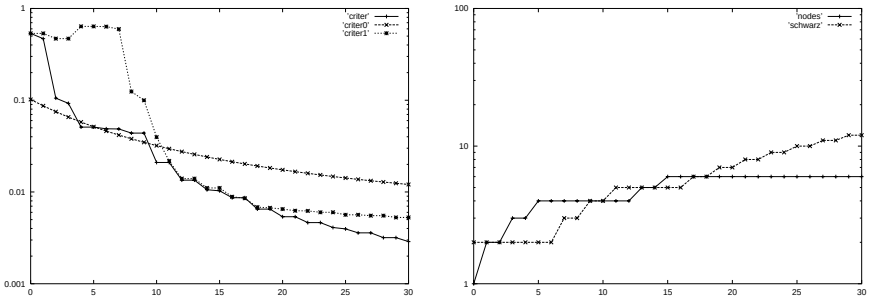

Figure 8: evolution of the cost function J (left) without mesh refinement (curve 'criter0') and with mesh refinement either on the norm of the gradient (case i above, curve 'criter1') or on the decrease of the cost function (case ii, curve 'criter') and of the number of Schwarz iterations N and mesh parameter n (right) are shown versus the iteration number for case i.

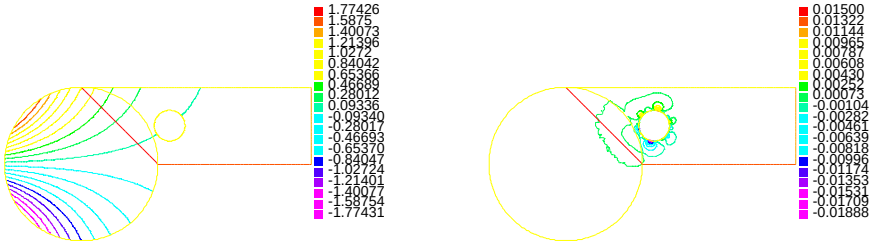

Figure 9: These plots show the computed solution u (left) and the error $u - u_d$ (right).

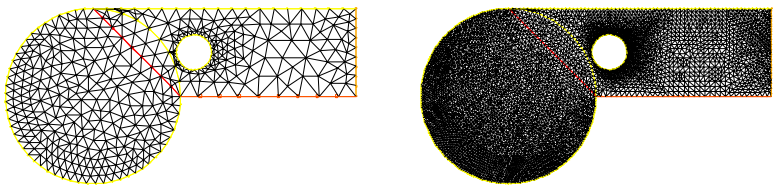

Figure 10: Second (left) and 7th mesh (right)

the boundaries.

6 Conclusion

It is a well known rule in optimization to use the exact gradient of the approximate problems rather than the approximate gradient of the exact problem. We have shown here that mesh refinement within the optimization loop enables us to slacken the above rule. Our motivation is two folds; first, there are problems where the exact gradient of the approximate problem cannot be computed simply; secondly there is a multi-grid effect in combining mesh refinement with descent algorithm which results in an order of magnitude in speed-up.

References

[1] Ph. Ciarlet, (1978) *The Finite Element Method. for Elliptic problems,* North-Holland.

[2] D. Bernardi, F.Hecht, K. Otsuka, O. Pironneau (1999)*freefem+, a finite element software to handle several meshes.* Downloadable from ftp://ftp.ann.jussieu.fr/pub/soft/pironneau/

[3] P. Grisvard (1985) *Elliptic problems in non-smooth domains* . Pitman.

[4] O. Ladyzhenskaya, and Ural'tseva, (1968) *Linear and quasilinear elliptic equations*, Academic Press, New York.

[5] P.L. Lions : On the Schwarz alternating method. I,II,III. Int Symposium on Domain decomposition Methods for

[6] O. Pironneau, (1984) *Optimal Shape Design of Elliptic Systems* Springer-Verlag, New York.

[7] E. Polak, (1997) *Optimization: Algorithms and Consistent Approximations*, Springer, New York.

7 Appendix

7.1 Verification of the hypothesis of Theorem 1

7.2 Inclusion

If $L_{h'}$ is obtained by $q^{i_1}, ..., q^{i_{n'_f}}$ and this set contains the vertices that yielded L_h then obviously

$$L_h \subset L_{h'} \subset H_0^2(I)$$

7.3 Continuity

From [4], we known about continuous dependence of the solution of a PDE with respect of data that:

$$\alpha^n \overset{H_0^2(I)}{\longrightarrow} \alpha, \quad \Rightarrow u^n \overset{H^1(\Omega)}{\longrightarrow} u, \quad \Rightarrow J^n \to J.$$

Similarly in the discrete case, the spline is continuous with respect to the vertex position so

$$q^{i\ n} \to q^i \Rightarrow \alpha_h^n \overset{H_0^2(I)}{\longrightarrow} \alpha_h, \quad \Rightarrow u_h^n \overset{H^1(\Omega)}{\longrightarrow} u_h, \quad \Rightarrow J_h^n \to J_h.$$

7.4 Consistency

It is obvious that

$$\forall \alpha,\ \exists \alpha_h \to \alpha \quad \text{with } J_h \to J.$$

because one just puts vertices on the continuous boundary, apply the construction and as the number of vertices increases the discrete curve converges to the continuous one if the following is observed:

- Corners of the continuous curve are vertices of the discrete curves
- the distance between boundary vertices converges uniformly to zero.

7.5 Continuity of θ

Conjecture : There exists ε such that $\alpha \in H_0^2 \quad \Rightarrow u \in H^{3/2+\epsilon}(\Omega)$.
Arguments: We know that $\alpha \in C^{0,1} \quad \Rightarrow u \in H^{3/2}$ and $\alpha \in C^{1,1} \quad \Rightarrow u \in H^2$ [3].
This technical point of functional analysis is needed for the continuity of θ. If so the following convergence properties hold

$$\begin{aligned} \alpha^n \xrightarrow{H^2} \alpha, \quad \Rightarrow u^n \xrightarrow{H^{3/2+\epsilon}(\Omega)} u, \quad \Rightarrow u^n|_\Sigma \xrightarrow{H^{1+\epsilon}} u|_\Sigma, \\ p^n \xrightarrow{H^{3/2+\epsilon}(\Omega)} p, \quad \Rightarrow p^n|_\Sigma \xrightarrow{H^{1+\epsilon}} p|_\Sigma, \\ \Rightarrow \nabla u^n \nabla p^n|_\Sigma \xrightarrow{L^2(\Sigma)} \nabla u \nabla p|_\Sigma \end{aligned}$$

7.6 Continuity of $\theta_h(\alpha_h)$

Recall that a variation $\delta\alpha_h$ (i.e. a boundary vertex variation $\delta q^j, j \in \Sigma$) implies variations of all inner vertices $\delta\alpha, \delta q^k, \forall k$
The problem is that θ is a boundary integral on Σ and θ_h is a volume integral! We must explain why

$$\begin{aligned} \delta J_h = \int_\Omega (\nabla u_h (\nabla \delta q_h + \nabla \delta q_h^T) \nabla p_h - \nabla u_h \cdot \nabla p_h \nabla \cdot q_h \\ +2\epsilon \int_\Sigma \frac{d^2\alpha}{ds^2} \frac{d^2\delta\alpha}{ds^2} \xrightarrow{?} \delta J = -\int_\Sigma \delta\alpha (up + \nabla u \nabla p + 2\epsilon \frac{d^4\alpha}{ds^4}) \end{aligned}$$

This is the object of Lemma 1.

7.7 Convergence

It comes from the theory of Finite Element Error Analysis ([1]):
Lemma

$$| \int_{\Sigma} \nabla u_h \nabla p_h - \nabla u \nabla p \; | \leq C h^{1/2} (\|p\|_2 + \|u\|_2)$$

and the following triangular inequalities

- $|a_h b_h - ab| = (a_h - a)(b_h - b) + b(a_h - a) + (b_h - b)a$
  $\leq |b||a_h - a| + |a||b_h - b| + |a_h - a|^2 + |b_h - b|^2$
- $|\nabla u_h - \nabla u|_{0,\Sigma} \leq |\nabla(u_h - \Pi_h u)|_{0,\Sigma} + |\nabla(\Pi_h u - u)|_{0,\Sigma}$

plus an inverse inequality for the first term and an interpolation for the second.