Differentiable and quasi-differentiable methods for Optimal Shape Design

http://www.ann.jussieu.fr/pironneau

Bijan Mohammadi\textsuperscript{2} & Olivier Pironneau\textsuperscript{1}

\textsuperscript{1}University of Paris VI, Lab. J.-L. Lions, \textsuperscript{2}University of Montpellier II, Lab. de Mathématiques

Olivier.Pironneau@upmc.fr  Bijan.Mohammadi@Univ.Montp2.fr

Machine Learning Conference
Outline

1. Conceptual Gradient Algorithm
2. Discretization
   • Summary
3. Topological Gradient-type Algorithms
4. Gradient Free Methods

More details in

More Books

Important Applications

- **Aerodynamics**: Shape optimization to improve airplanes, cars, ventilators, turbines...
- **Hydrodynamics**: wave drag of boats, pipes, by-pass, harbors...
- **Electromagnetics**: Stealth airplane, antenna, missiles...
- **Combustion**: Car and airplane engines, scramjets...
- **Turbulence**: Delay the separation of boundary layers, reduce turbulent drag (active control, deformable airplane...)

Pironneau (LJLL)
Important Applications

- **Aerodynamics**: Shape optimization to improve airplanes, cars, ventilators, turbines...
- **Hydrodynamics**: wave drag of boats, pipes, by-pass, harbors...
- **Electromagnetics**: Stealth airplane, antenna, missiles...
- **Combustion**: Car and airplane engines, scramjets...
- **Turbulence**: delay the separation of boundary layers, reduce turbulent drag (active control, deformable airplane...)
Main Topics for Shape Optimization

\[
\min_{v \in V \subset \mathbb{R}^d} E(v)
\]

- **Black Box Optimization:** use only \( v \to E(v) \)
- **Differentiable Optimization:** use also \( \text{grad}_v E(v) \)

\[
E(v + \delta v) = E(v) + \langle \text{grad} E(v), \delta v \rangle + o(\|\delta v\|)
\]
\[
\delta v = -\rho \text{grad} E(v) \implies E(v + \delta v) - E(v) \approx -\rho \|\text{grad} E(v)\|^2
\]

- **Constrained Optimization:** \( V = \{ v \in H : f(v) = 0, \ g(v) \leq 0 \} \)
- **Multi-criteria and Pareto optimality:**

\[
E(v) = \sum_i \alpha_i E_i(v) \iff \exists \ w : E_i(w) \leq E_i(v) \ \forall i
\]

- **Topological Optimization:** Embed the problem into a larger class

Pironneau (LJLL)
Differentiable and quasi-differentiable methods: MLA09 5 / 51
An Academic Problem

\[ \min_{S \in S_d} \left\{ \int_D \left| \psi - \psi_d \right|^2 : -\Delta \psi = 0, \text{ in } C - \dot{S}, \quad \psi|_S = 0, \quad \psi|_{\partial C} = \psi_d \right\} \]

Wind tunnel Design by adapting $S$ so that flow is uniform in $D$. Flow is irrotational inviscid and 2D.
Existence of Solution

**Theorem**

\[
\min_{\nu \in V \subset \mathbb{R}^d} E(\nu)
\]

has a solution if \( V \) is closed, \( E \) is bounded from below, l.s.c. and either \( V \) is bounded or \( \lim_{||x|| \to \infty} E(x) = +\infty \)

Thus if one can show that the criteria of the OSD problem is l.s.c. a solution will exist. It has been shown by Sverak that this is so if the number of connected component of \( \Omega \) is bounded.

In Allaire, Bucur, Delfour et al, it is shown that a penalization of the perimeter of the unknown surface also induce existence in 2D.

**Theorem** The following problem has at least one solution:

\[
\min_{\mathbf{s} \in S_d} \left\{ \int_D |\psi - \psi_d|^2 + \epsilon |\mathbf{s}|^2 : -\Delta \psi = 0, \text{ in } C - \mathbf{s}, \quad \psi|_{\mathbf{s}} = 0 \quad \psi|_{\partial C} = \psi_d \right\}
\]

Uniqueness is almost impossible to prove;
Existence of Solution

**Theorem**

\[ \min_{v \in V \subset \mathbb{R}^d} E(v) \]

has a solution if \( V \) is closed, \( E \) is bounded from below, l.s.c. and either \( V \) is bounded or \( \lim_{||x|| \to \infty} E(x) = +\infty \)

Thus if one can show that the criteria of the OSD problem is l.s.c. a solution will exist. It has been shown by Sverak that this is so if the number of connected component of \( \Omega \) is bounded.

In Allaire, Bucur, Delfour et al, it is shown that a penalization of the perimeter of the unknown surface also induce existence in 2D.

**Theorem** The following problem has at least one solution:

\[ \min_{S \in S_d} \left\{ \int_D |\psi - \psi_d|^2 + \epsilon |S|^2 : -\Delta \psi = 0, \text{ in } C - \dot{S}, \; \psi|_S = 0 \; \psi|_{\partial C} = \psi_d \right\} \]

Uniqueness is almost impossible to prove;
Sensitivity Analysis

\[-\Delta \psi^\epsilon = f \quad \text{in } \Omega^\epsilon \quad \psi^\epsilon = 0 \text{ on } \Gamma^\epsilon := \{x + \epsilon \alpha n : x \in \Gamma\}\]

**Definition** If \(\psi'_\alpha := \lim \frac{1}{\epsilon} (\psi^\epsilon - \psi)\) exists then \(\psi\) is Gateau differentiable with respect to \(\Gamma\) in the direction \(\alpha\). If \(\psi'_\alpha\) is linear in \(\alpha\) then \(\psi\) is Frechet differentiable. Similarly

\[
\psi^{\epsilon \alpha} = \psi + \epsilon \psi'_\alpha + \frac{\epsilon^2}{2} \psi''_\alpha
\]

To compute \(\psi'\) and \(\psi''\) notice that, by linearity, they satisfy the same PDE but with \(f = 0\). By Taylor expansion, \(x \in \Gamma\):

\[
0 = \psi^{\epsilon \alpha} (x + \epsilon \alpha n) = \psi^{\epsilon \alpha} (x) + \epsilon \alpha \frac{\partial \psi^{\epsilon \alpha}}{\partial n} (x) + \frac{\epsilon^2 \alpha^2}{2} \frac{\partial^2 \psi}{\partial n^2} (x) + \ldots
\]

Therefore

\[
-\Delta \psi'_\alpha = 0 \quad \psi'_\alpha |\Gamma = -\alpha \frac{\partial \psi}{\partial n}, \quad -\Delta \psi''_\alpha = 0 \quad \psi''_\alpha |\Gamma = -\alpha \frac{\partial \psi'_\alpha}{\partial n} - \frac{\alpha^2}{2} \frac{\partial^2 \psi}{\partial n^2}
\]
Consider the Wind Tunnel Problem with $S^\varepsilon = \{x + \varepsilon \alpha n : x \in S\}$. Think of the PDE as the implicit definition of $S \rightarrow \psi(S)$. Then $J()$ is a function of $S$ only:

$$J(S^\varepsilon) = \int_D |\psi^\varepsilon - \psi_d|^2 = \int_D |\psi - \psi_d|^2 + 2\varepsilon \int_D (\psi^\varepsilon - \psi_d) \psi'_\alpha + o(\varepsilon)$$

with $\Delta \psi'_\alpha = 0$, $\psi'_\alpha|_S = -\alpha \frac{\partial \psi}{\partial n}$, $\psi'_\alpha|_{\Gamma-S} = 0$.

If $J$ is Frechet differentiable there exists $\xi$ such that $J'_\alpha = \int_S \xi \alpha$. To find $\xi$ we must use the adjoint trick and introduce

$$-\Delta p = (\psi^\varepsilon - \psi_d) I_D, \quad p|_\Gamma = 0$$

Then

$$2 \int_D (\psi^\varepsilon - \psi_d) \psi'_\alpha = -2 \int_{\Omega} \psi'_\alpha \Delta p = -2 \int_{\Omega} \Delta \psi'_\alpha p + \int_{\Gamma} (\frac{\partial p}{\partial n} \psi'_\alpha + \frac{\partial \psi'_\alpha}{\partial n} p)$$

Corollary

$$J'_\alpha = 2 \int_S \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n} \alpha$$
Conceptual Algorithm

0. Choose a shape $S^0$, a small number $\rho > 0$ and set $m=0$.
1. Compute $\psi^m$ and $p^m$ by solving

$$
-\Delta \psi^m = 0, \quad \psi^m|_{S^m} = 0, \quad \psi^m|_{\Gamma_d} = \psi_d \\
-\Delta p^m = (\psi^m - \psi_d)I_D, \quad p|_{\Gamma_m} = 0
$$

2. Set

$$
\alpha = -\rho \frac{\partial p^m}{\partial n} \frac{\partial \psi^m}{\partial n} \quad S^{m+1} = \{x + \alpha n : x \in S^m\}
$$

3. Set $m \leftarrow m + 1$ and go to 1.

It works because

$$
J(S^{m+1}) = J(S^m) + \int_{S^m} \xi \alpha = J(S^m) - 2\rho \int_{S^m} \left(\frac{\partial p^m}{\partial n} \frac{\partial \psi^m}{\partial n}\right)^2 + o(\alpha)
$$

Notice that there is a loss of regularity from $S^m$ to $S^{m+1}$!
Implementation with freefem++

```plaintext
real xl = 5, L=0.3;
mesh th = square(30,30,[x,y*(0.2+x/xl)]);
func D=(x>0.4+L && x<0.6+L)*(y<0.1);
func psid = 0.8*y;
fespace Vh(th,P1);
Vh psi,p,w;

problem streamf(psi,w)=int2d(th)(dx(psi)*dx(w) + dy(psi)*dy(w))
    +on(1,4,psi=y/0.2)+on(2,psi=y/(0.2+1.0/xl)) + on(3,psi=1);

problem adjoint(p,w)=int2d(th)(dx(p)*dx(w) + dy(p)*dy(w))
    - int2d(th)(D*(psi-psid)*w)+ on(1,2,3,4,p=0);
Vh a=0.2+x/xl, gradE;
for(int i=0;i<100;i++){
    streamf; adjoint;
    real E = int2d(th)(D*(psi-psid)^2)/2;
    gradE = dx(psi)*dx(p) + dy(psi)*dy(p);
    a=a(x,0)-50*gradE(x,a(x,0))*x*(1-x);
    th = square(30,30,[x,y*a(x,0)]);
}
```

Execute

Pironneau (LJLL)
Differentiable and quasi-differentiable methods
MLA09 11 / 51
Oscillations

Pironneau (LJLL) Differentiable and quasi-differentiable methods MLA09 12 / 51
\[ \alpha = -\rho \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n} \]

\[ S^{m+1} = \{ x + \alpha n : x \in S^m \} \]

can be replaced by

\[ \frac{d^2 \tilde{\alpha}}{ds^2} = \rho \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n} \alpha(s_0) = \tilde{\alpha}(s_1) = 0. \quad S^{m+1} = \{ x + \tilde{\alpha} n : x \in S^m \} \]

\[ \Rightarrow \quad J(S^{m+1}) - J(S^m) = \frac{2}{\rho} \int_{S^m} \tilde{\alpha} \frac{d^2 \tilde{\alpha}}{ds^2} = -\frac{2}{\rho} \int_{S^m} (\frac{d\tilde{\alpha}}{ds})^2 + o(\rho) \]

Alternatively one may use a smoothing operator like

\[ \beta \rightarrow \gamma(\beta) = v \text{ where } v \text{ is solution of } -\Delta v = 0 \quad \frac{\partial v}{\partial n}|_\Gamma = \beta. \]

Let \( S^{m+1} = \{ x + \gamma(\beta) n : x \in S^m \} \) with \( \beta = \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n} \)

\[ J(S^{m+1}) - J(S^m) = 2\rho \int_{\Gamma} \gamma(\beta)\beta = 2\rho \int_{\Gamma} v \frac{\partial v}{\partial n} = -2\rho \int_{\Omega} |\nabla v|^2 \]
Geometric Constraints

- Projected Gradient: the case $\int_\Omega = 1$.

$$\Gamma' = \{ x + \alpha n(x) : x \in \Gamma \} \Rightarrow \delta J = \int_\Gamma \chi \alpha ds + o(|\alpha|)$$

$$\Gamma' = \{ x + (\alpha - \frac{1}{|\Gamma|} \int_\Gamma \alpha) n(x) : x \in \Gamma \} \Rightarrow \delta \int_\Omega = \int_\Gamma (\alpha - \frac{1}{|\Gamma|} \int_\Gamma \alpha) + o(|\alpha|) = o(|\alpha|)$$

$$\delta J = \int_\Gamma (\chi - \frac{1}{|\Gamma|} \int_\Gamma \chi)(\alpha - \frac{1}{|\Gamma|} \int_\Gamma \alpha) ds + o(|\alpha|)$$

- Penalization: replace $J$ by

$$J + \frac{1}{\epsilon}|F(\Omega)|^2 + \frac{1}{\omega}|G(\Omega)|^2$$

to maintain $F(\Omega \leq 0)$, $G(\Omega) = 0$
Geometric Constraints

- Projected Gradient: the case $\int_{\Omega} = 1$.

\[
\Gamma' = \{ x + \alpha n(x) : x \in \Gamma \} \Rightarrow \delta J = \int_{\Gamma} \chi \alpha ds + o(|\alpha|)
\]

\[
\Gamma' = \{ x + (\alpha - \frac{1}{|\Gamma|} \int_{\Gamma} \alpha) n(x) : x \in \Gamma \}
\]

\[
\Rightarrow \delta \int_{\Omega} = \int_{\Gamma} (\alpha - \frac{1}{|\Gamma|} \int_{\Gamma} \alpha) + o(|\alpha|) = o(|\alpha|)
\]

\[
\delta J = \int_{\Gamma} (\chi - \frac{1}{|\Gamma|} \int_{\Gamma} \chi)(\alpha - \frac{1}{|\Gamma|} \int_{\Gamma} \alpha) ds + o(|\alpha|)
\]

- Penalization: replace $J$ by

\[
J + \frac{1}{\epsilon} |F(\Omega^+)|^2 + \frac{1}{\omega} |G(\Omega)|^2
\]

to maintain $F(\Omega \leq 0), \ G(\Omega) = 0$
State Constraints

\[
\min_{\nu} \{ J(u, \nu) : A u = g(\nu), \ F(u, \nu) \leq 0 \}
\]

where \( A \) is a linear invertible operator.

\[
\delta J = J'_u \delta u + J'_v \delta v \text{ with } A \delta u = g'_v \delta v, \ F'_u \delta u + F'_v \delta v \leq 0 \text{ if } F(u, \nu) = 0
\]

Introducing \( A^T p = J'_u, \ A^T q = F'_u \) leads to

\[
J'_u \delta u = \delta u \cdot A^T p = p \cdot A \delta u = p \cdot g'_v \delta v \quad F'_u \delta u = \delta u \cdot A^T p = q \cdot A \delta u = q \cdot g'_v \delta v
\]

\[
\delta J = (p \cdot g'_v + J'_v) \delta v \text{ with } (q \cdot g'_v + F'_v) \delta v \leq 0 \text{ if } F(u, \nu) = 0
\]

A direction of descent is built from this.

Notice that two adjoint vectors are needed
Example

Build a stealth airfoil with "good" aerodynamic properties

\[
\min_S J := \int_D |u|^2 : \int_S \frac{\partial \psi}{\partial n} = a
\]

\[
\omega^2 u + \Delta u = 0, \text{ in } \Omega \quad u|_\Gamma = g
\]

\[
-\Delta \psi = 0, \text{ in } \Omega \quad \psi|_\Gamma = \psi_d
\]

Requires the following

Lemma

\[
\Gamma' = \{ x + \alpha n : x \in \Gamma \} \Rightarrow \delta \int_\Gamma f = \int_\Gamma \alpha \left( \frac{\partial f}{\partial n} - \frac{f}{R} \right)
\]

where \( R \) is the mean radius of curvature.
Example

Computed by A. Baron
The Minimum Drag Problem

\[ J(\Omega) \equiv \min_{\Omega \in \mathcal{C}, \text{vol}(\mathcal{C})=1} \int_{\Omega} \frac{1}{2} ||\nabla u||^2 \, dx : \quad u|_{\partial \Omega} = g \]

\[ u \nabla u + \nabla p - \nu \Delta u = 0, \quad \nabla \cdot u = 0, \]

The solution exists in 2D if the nb of connected component is bounded. In 3D ? but probably yes because the criteria is the energy of the system. For safety regularize by adding \( \epsilon \text{call}(\mathcal{C}) \).

Proposition

\[ \partial \Omega' = \{ x + \alpha n(x) : \, x \in \partial \Omega \} \Rightarrow \delta J = \int_{\partial \Omega} \chi \alpha ds + o(|\alpha|) \]

\[ \delta J = \int_{\partial \Omega} \alpha \frac{\partial u}{\partial n} \cdot \left( \frac{1}{2} \frac{\partial u}{\partial n} + \frac{\partial w}{\partial n} \right) + o(|\alpha|) \]

where \(-u \nabla w + w \nabla u^T + \nabla q - \nu \Delta w = \nu \Delta u, \quad \nabla \cdot w = 0, \quad w|_{\partial \Omega} = 0\)

From JFM 73. See also, Modi, Gunzberger, Tasan, Jameson... Q?

What minimal norm on \( \alpha \)?
The Minimum Drag Problem

\[ J(\Omega) \equiv \min_{\Omega \in C, \text{vol}(C)=1} \int_{\Omega} \frac{1}{2} \| \nabla u \|^2 dx : \quad u|_{\partial \Omega} = g \]

\[ u\nabla u + \nabla p - \nu \Delta u = 0, \quad \nabla \cdot u = 0, \]

The solution exists in 2D if the nb of connected component is bounded. In 3D? but probably yes because the criteria is the energy of the system. For safety regularize by adding \( \epsilon \text{call}(C) \).

**Proposition**

\[ \partial \Omega' = \{ x + \alpha n(x) : x \in \partial \Omega \} \Rightarrow? \quad \delta J = \int_{\partial \Omega} \chi \alpha ds + o(|\alpha|) \]

\[ \delta J = \int_{\partial \Omega} \alpha \frac{\partial u}{\partial n} \cdot \left( \frac{1}{2} \frac{\partial u}{\partial n} + \frac{\partial w}{\partial n} \right) + o(|\alpha|) \]

where \(-u \nabla w + w \nabla u^T + \nabla q - \nu \Delta w = \nu \Delta u, \quad \nabla \cdot w = 0, \quad w|_{\partial \Omega} = 0\)

From JFM 73. See also, Modi, Gunzberger, Tasan, Jameson... Q?

What minimal norm on \( \alpha \)?
Proof

Recall that $\delta \int_{\Omega} f = \int_{\Gamma} \alpha f$. Then

$$
\delta J = \int_{\Omega} \nabla u \cdot \nabla \delta u + \frac{1}{2} \int_{\partial \Omega} \alpha |\nabla u|^2 + o(|\alpha|)
$$

and $\delta u \nabla u + u \nabla \delta u + \nabla \delta p - \nu \Delta \delta u = 0$, $\nabla \cdot \delta u = 0$, $\delta u|_{\Gamma} = -\alpha \frac{\partial u}{\partial n}$

So

$$
\int_{\Omega} \nabla u \cdot \nabla \delta u = -\int_{\Omega} \delta u \Delta u
$$

$$
= -\frac{1}{\nu} \int_{\Omega} (-u \nabla w + w \nabla u^T + \nabla q - \nu \Delta w) \delta u
$$

$$
= -\frac{1}{\nu} \int_{\Omega} (\nabla \cdot (u \otimes \delta u + \delta u \otimes u) - \nu \Delta \delta u) w - \int_{\Gamma} \nu \frac{\partial w}{\partial n} \delta u + o(|\alpha|)
$$
Example

Minimum drag object of given area at Reynolds 50 (Courtesy of Kawahara et al.).
Compressible Flows

Euler or Navier-Stokes equations

\[
W = \begin{pmatrix} 
\rho \\
\rho u \\
\rho E 
\end{pmatrix} \quad \partial_t W + \nabla \cdot F(W) - \nabla \cdot G(W, \nabla W) = 0
\]

\[W(0, x) = 0, \quad + \text{B.C.}\]

Involves an adjoint equation

\[
\partial_t P + (F'(W) - G'_{,1}(W, \nabla W)^T \nabla P) - \nabla \cdot (G'_{,2}(W, \nabla W)^T \nabla P) = 0
\]
Some Realizations - A. Jameson (I)

Plain vs Sobolev Gradients

Before & after optimization
Optimization of the Boeing 747: 10% wing drag saving (5% aircraft drag)
Falcon jet: $C_D$ decreases from 234 to 216
Outline

1. Conceptual Gradient Algorithm

2. Discretization
   - Summary

3. Topological Gradient-type Algorithms

4. Gradient Free Methods
Summary

- Optimal Shape Design of $S$ relies on Optimization
  \[
  \min_{S} J(u, S) : A(S)u = f
  \]

- The Continuous problem is well posed after regularization
  \[
  \min_{S} J(u, S) + \epsilon |S|^2 : A(S)u = f
  \]

- The $L^2$ local gradient $\chi$ is computable by calculus of variation:
  \[
  \delta J = \int_{S} \chi \alpha + o(|\alpha|), \quad S(\alpha) = \{x + \alpha(x)n(x) : x \in S\}
  \]

- The Sobolev gradient is the right tool for gradient methods:
  \[
  -\Delta_S \beta = \chi, \quad S^{n+1} = \{x - \rho \beta(x)n(x) : x \in S^n\}
  \]
\[
\min_{S \in S_d} J(S) := \left\{ \int_D |\psi - \psi_d|^2 : -\Delta \psi = 0, \text{ in } C - \hat{S}, \quad \psi|_S = 0 \quad \psi|_{\partial C} = \psi_d \right\}
\]

Discretization of gradients \( J'_\alpha = \nabla \psi \nabla p \) where \(-\Delta p = 2I_D(\psi - \psi_d)\), \( p|_\Gamma = 0 \) or derivation of gradient for the discrete problem?

**Optimization of the Discrete Problem**

- The Finite Element Method,
- Discrete Gradients
- Finite Volume Methods
The Finite Element Method

\( \Omega \) is covered with triangles \( T_k \) and \( q^i \) are the vertices. The PDE of the wind tunnel problem is approximated by

\[
\int_{\Omega} \nabla \psi_h \nabla w_h = 0, \quad \psi_h|_S = 0, \quad \psi_h|_\Gamma = \psi_d
\]

for all \( w_h \) continuous and affine on each \( T_k \) and zero on \( \partial \Omega \).

\[
J = \int_D ||\psi_h - \psi_d||^2 + \epsilon \int_S \left| \frac{d^2 \alpha}{dn^2} \right|^2
\]

Let \( \delta q_h(x) = \sum_i \delta q_i w(x) \), the basis \( \{ w^j \} \), the hat function of \( q_j \).
Summary: Continuous versus Discrete Gradient

\[
\min_{s \in S_d} J(s) := \left\{ \int_D |\psi - \psi_d|^2 : -\Delta \psi = 0, \text{ in } C - \dot{S}, \quad \psi|_{S} = 0 \quad \psi|_{\partial C} = \psi_d \right\}
\]

\[
\delta J = \int_S \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n} \quad \text{with} \quad -\Delta p = 2(\psi - \psi_d)l_D
\]

use normal displacement \(\approx v\) : \(-\Delta v = 0, \quad \frac{\partial v}{\partial n}|_{\Gamma} = \frac{\partial p}{\partial n} \frac{\partial \psi}{\partial n}\)

For the discrete system

\[
\min_{q^i \in Q_d} J(S_h) = \left\{ \int_D |\psi_h - \psi_d|^2 : \int_\Omega \nabla \psi_h \cdot \nabla w^j = 0, \quad \forall j \quad \psi_h|_{S} = 0 \quad \psi_h|_{\partial C} = \psi_d \right\}
\]

\[
\delta J = \int_\Omega (\nabla \psi_h (\nabla \delta q_h + \nabla \delta q_h^T) \nabla p_h - \nabla \psi_h \cdot \nabla p_h \nabla \cdot \delta q_h) = \sum \chi_j \delta q^j
\]

with \(\int_\Omega \nabla p_h \nabla w^j = 2 \int_D (\psi_h - \psi_d) w^j, \quad p_h \in V_{0h}\)

And use a smoothed version of \(\chi_j\) to move the vertices and find the new shape (and triangulation).
Topological Optimization

- Applies when the topology is not known
- Black-box favors Genetic algorithm (yet slow)
- Combine topological and geometrical shape design?

From T. Borrval and J. Petterson

From Schoenauer et al
Topological Derivatives

Following the work of L. Tartar and N. Kikuchi, J. Sokolowski came with the following idea (f has zero mean, $B(0,1)$ the unit ball)

$$-\Delta u = f \text{ in } \Omega, \quad u|_{\Gamma} = 0,$$
$$-\Delta u^\varepsilon = f \text{ in } \Omega \setminus B(x_0, \varepsilon), \quad u^\varepsilon|_{\Gamma} = 0,$$
$$u'_{x_0}(x) = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon^\gamma} (u^\varepsilon - u)$$

exists and is not identically 0 or $+\infty$ for some value of $\gamma$.

**Theorem** For the Neumann (resp Dirichlet) problem $\gamma = 2$ (resp $\log \varepsilon$) in 2D and $u'$ solves

$$\int_{\Omega} \nabla u \cdot \nabla w = c \nabla u \nabla w|_{x_0}$$

This is sufficient for gradient type algorithm, but convergence is usually a problem.
Applications of Topological Optimization

Stokes flow drag optimization (courtesy of M. Masmoudi)
Micro Channel flow (Borrval and Petterson)

Optimization of a micro channel flow averaged vertically gives

\[
\min_{z(x) \in Z} j(u) := \int_D (u - u_d)^2 : \frac{5}{2z^2} u - \Delta u + \nabla p = 0, \quad \nabla \cdot u = 0, \quad u|_{\Gamma} = g
\]

where the pointwise values of functions of \(Z\) are equal to \(\epsilon\) or \(h\). Let \(\rho = 2.5z^{-2}\); notice that

\[
[\rho u] = \bar{\rho}[u] + [\rho]\bar{u} \quad \bar{a} = \frac{a_1 + a_2}{2} \quad [a] = a_1 - a_2
\]

Therefore if \(u'\) exists, the derivative w/r “\(\rho_2\) becoming \(\rho_1\)” at \(x_0\), it must be

\[
\bar{\rho} u' + \rho' \bar{u} - \Delta u' + \nabla p' = 0 \quad \nabla \cdot u' = 0 \text{ with } \rho' = [\rho]\delta(x - x_0), \quad u'|_{\Gamma} = 0
\]

But \(u\) is continuous so \(\bar{u} = u\). Introduce the adjoint state \(v, q\)

\[
\bar{\rho} v - \Delta v + \nabla q = 0 \quad \nabla \cdot q = 2(u - u_d)\chi_D, \quad v|_{\Gamma} = 0
\]

\[\Rightarrow j' = -[\rho]u(x_0)p(x_0)\]

Replace : \(\rho_2\) by \(\rho_1\) at \(x_0\) when \([\rho]u(x_0)p(x_0) > 0\)
Important Applications

**Solid mechanics:** Weight optimization of airplanes, cars, parts...

Topological optimization of the weight of a stool for a given strength (courtesy of F. Jouve et al)
Now consider the same algorithm with parameter refinement

**Algorithm**

*(Steepest descent with refinement)*

```plaintext
while \( h > h_{\text{min}} \) do

{ 
  while \( \| \text{grad}_z J_h(z^m) \| > \epsilon h^n \) do

  { 
    \( z^{m+1} = z^m - \rho \text{grad}_z J_h(z^m) \) where \( \rho \) such that,
    \[
      - \beta \rho \| w \|^2 < J_h(z^m - \rho w) - J_h(z^m) < -\alpha \rho \| w \|^2
    \]
    with \( w = \text{grad}_z J_h(z^m) \) Set \( m := m + 1; \)
  } 

  \( h := h/2; \)

} 
```
Steepest Descent and Inexact Gradients

- Convergence obvious: it is either S.Descent or $\text{grad} J_h \rightarrow 0$ because $h \rightarrow h/2$.
- Gain in speed: we do not need the exact gradient $\text{grad}_z J_h$!
- Let $N$ be an iteration parameter and $J_{h,N} \approx J_h$ and $\text{grad}_z J_{h,N} \approx \text{grad}_z J_h$ in the sense that

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

Add $K$ and $N(h)$ with $N(h) \rightarrow \infty$ when $h \rightarrow 0$:
Algorithm

(E. Polak et al) (Steepest descent with Goldstein's rule mesh refinement and approximate gradients)

\[
\begin{align*}
\text{while } h > h_{\min} \{ \\
\quad \text{while } |\nabla z_N J^m| > \epsilon h^{\gamma} \{ \\
\quad \quad \text{try to find a step size } \rho \text{ with } w = \nabla z_N J(z^m) \\
\quad \quad \quad - \beta \rho \|w\|^2 < J(z^m - \rho w) - J(z^m) < -\alpha \rho \|w\|^2 \\
\quad \quad \text{if success then} \\
\quad \quad \quad z^{m+1} = z^m - \rho \nabla z_N J^m; \quad m := m + 1; \}
\quad \text{else } N := N + K; \\
\} \\
\text{h := h/2; \quad N := N(h);} \\
\end{align*}
\]
The convergence could be established from the observation that Goldstein’s rule gives a bound on the step size:

\[-\beta \rho \nabla_z J \cdot h < J(z + \rho h) - J(z) = \rho \nabla_z J \cdot h + \frac{\rho^2}{2} J''(\xi) hh\]

\[\Rightarrow \rho > 2(\beta - 1) \frac{\nabla_z J \cdot h}{J''(\xi) hh} \quad \text{so} \quad J^{m+1} - J^m < -2 \frac{\alpha(1 - \beta)}{\|J''\|} |\nabla_z J|^2\]

Thus at each grid level the number of gradient iterations is bounded by $O(h^{-2\gamma})$. Therefore the algorithm does not jam hence convergence.
Mesh Refinements
Finite Difference Gradient

\[
\frac{f(x + h) - f(x)}{h} = f'(x) + f^{(2)} \frac{h}{2} - f^{(3)} \frac{h^2}{6} + \ldots
\]

\[
\Re \frac{f(x + ih) - f(x)}{ih} = f'(x) + O(h^2)
\]

\[
\frac{f(x + h) - f(x - h)}{2h} = f'(x) + f^{(3)} \frac{h^2}{6} + f^{(5)} \frac{h^4}{60} + \ldots
\]

\[
\frac{f(x + h) - f(x - h)}{4h} + \Re \frac{f(x + ih) - f(x - ih)}{4ih} = f'(x) + O(h^6)
\]
Principle of Automatic Differentiation

Let $J(u) = |u - u_d|^2$, then its differential is

$$\delta J = 2(u - u_d)(\delta u - \delta u_d) \quad \frac{\partial J}{\partial u} = 2(u - u_d)(1.0 - 0.0)$$

Obviously the derivative of $J$ with respect to $u$ is obtained by putting $\delta u = 1, \delta u_d = 0$. Now suppose that $J$ is programmed in C/C++ by

```c
double J(double u, double u_d) {
    double z = u - u_d;
    z = z * (u - u_d);
    return z;
}

int main() {
    double u = 2.0, u_d = 0.1;
    cout << J(u, u_d) << endl;
}
```

A program which computes $J$ and its differential can be obtained by writing above each differentiable line its differentiated form:
A simple example (cont)

class ddouble {public: double v,d;
double(double a, double b=0){ v = a; d=b;}};

ddouble JandDJ(ddouble u, ddouble u_d)
{
    ddouble z;
    z.d = u.d - u_d.d;
    z.v = u.v-u_d.v;
    z.d= z.d*(u.v-u_d.v) + z.v*(u.d - u_d.d);
    z = z*(u-u_d);
    return z;
}

int main()
{
    ddouble u(2.,1.), u_d= 0.1, J = JandDJ(u,u_d);
    cout << J << " dJ="<<dJ<<endl;
}
The class ddouble

class ddouble
{
public:
    double v[2];

double(double a, double b=0)
    { v[0] = a; v[1]=b; }

double operator=(const ddouble& a)
    {
        val[1] = a.v[1]; val[0]=a.v[0];
        return *this;
    }

friend ddouble operator-(const ddouble& a,const ddouble& b)
    {
        ddouble c;
        c.v[0] = a.v[0] - b.v[0];
        return c;
    }

friend ddouble operator*(const ddouble& a,const ddouble& b)
    {
        ddouble c;
        c.v[1] = a.v[1]*b.v[0] + a.v[0]* b.v[1];
        c.v[0] = a.v[0] * b.v[0];
        return c;
    }
};
#include "ddouble.hpp"

double J(ddouble u, ddouble u_d){
    ddouble z = u - u_d;
    z = z * (u - u_d);
    return z;
}

int main(){
    ddouble u = 2, u_d = 0.1;
    u.v[1] = 1;
    cout << J(u, u_d).v[1] << endl;
}

Simply replace all double by ddouble and link with the class lib.
A few pitfalls: e.g.

double sqrt(ddouble x){
    ddouble y;
    y.v[1] = x.v[1] / sqrt(fabs(x.v[0]) + eps);
    y.v[0] = sqrt(x.v[0]);
    return y;
}
program newtontest
  x=0.0;
  al=0.5
  subroutine newton(x,n,al)
    call newton(x,10,al)
    do i=1,n
      f = x-alpha*cos(x)
      fp= 1+alpha*sin(x)
      x=x-f/fp
    enddo
    return
  end
  end

2n adjoint variables are needed! while the theory is

\[ f(x, \alpha) = 0 \Rightarrow x' f'_x + f'_\alpha = 0 \Rightarrow x' = -\frac{f'_\alpha}{f'_x} \]

So it is better to understand the output of AD-reverse and clean it. see www.autodiff.org
program newtontest
x=0.0
xb=2
al=0.5
call newton_b(x,xb,5,al,alb)
write(*,*) x,xb
end

SUBROUTINE NEWTON_B(x,xb,n,al,alb)
    DO i=1,n
        alb = alb+SIN(x)*fpb-COS(x)*fb
        xb = xb + al*COS(x)*fpb & + (al*SIN(x)+1.0)*fb
    CALL POPREAL4(f)
    x = x - f/fp
ENDDO
END
Optimization of a wing profile

Drag is mostly pressure by the shock. The lift & area are imposed

\[ J(u, p, \theta) = F \cdot u_\infty + \frac{1}{\epsilon} |F \times u_\infty - C_l|^2 + \frac{1}{\beta} (\int_S dx - a)^2 \]

with \( F = \int_S (\rho n + (\mu \nabla u + \nabla u^T)) \) and Navier-Stokes + \( k - \epsilon \) + wall laws
Optimization of a 3D Business Jet

Done by B. Mohammadi in a few hours on a workstation
Gradient Free Methods

The main motivation is the non access to the source and the prototyping speed

- Powells’ NEWUOA
- Evolutionary algorithms
- Hybrid methods

[width=5cm]rastrigin Rastrigin’s test with 20 param
Proposed by L. Dumas

- Random initialization of a population
- Until convergence do:
  - GA evolution (selection, crossover and mutation)
  - If stagnation during three generations then three iterations of BFGS on the current best individual
- Repeat

Optimization of a mockup car with 4 param. (Dumas-Muyr)
Perspectives

- Parallel and Stream Computing (MPI and CUDA)
- Enormous systems: automatic Differentiations?
- Link with CAD
- Progresses of G.A. algorithms

Bis petit obscurum et condit se Luna tenebris (Nostradamus)

"For Optimal Shape Design the future lies in mixing Gradient Free methods with Differentiable Optimization".

The End