Local Search Methods for Design in Aeronautics

Manuel Bompard and Jacques Peter

Department of Computational Fluid Dynamics and Aeroacoustics
ONERA. The french aerospace lab
BP 72 - 29 av. de la Division Leclerc
92322 Châtillon Cedex

e-mail : jacques.peter@onera.fr / manuel.bompard@onera.fr

ABSTRACT

Many shape optimization problems of industrial interest can be converted in mathematical finite-dimensional optimizations through discretization and parametrization. Hence a good knowledge of global and local optimization algorithms is important to design engineers. This part of the lecture series describes the most classical local optimization methods. All of these algorithms - except the simplex - require the gradient of the functions of interest with respect to the design parameters. The different ways to compute those derivatives - often called "sensitivities" - are also described.

1 Introduction

Numerical optimization aims at locating the minima of a regular function (called objective function) on a finite-dimensional design space, while satisfying a certain number of constraints (expressed as inequality verified by the so-called constraint functions). More precisely, local optimization aims at finding a local optimum in the neighborhood of an initial guess, whereas global optimization aims at finding the global optimum on the whole design space. These problems are, of course, the mathematical counterparts of mechanical optimization problems - like drag minimization of an aicraft or total pressure maximization of a supersonic aircraft air intake - as soon as (a) a mesh and a simulation tool are available ; (b) the solid shape has been parametrized/a remeshing tool is available to propagate its deformation to the whole mesh ; (c) the objective and constraints have been expressed as functions of the geometry and state variables.
Numerical optimization for airplane design was used almost as soon as simulation codes appeared. The aerodynamic optimizations carried out by G.N. Van der Plaats at NASA in the mid 70's illustrate this early interest in optimization [1]. At that time, 2D and simple 3D configurations were considered, simplex or descent methods were used and the gradients required by descent methods were estimated by the finite-differences. Since then, the framework of aerospace optimization has known at least three drastic extensions:
(1) several global optimization methods have been defined and intensively used (evolutionary algorithm, particle swarm, ant colony, simulated annealing,...) ;
(2) surrogate functions (neural network, Kriging, polynomial regression, support vector machine,...) have been used for a part of the evaluation of the global optimization methods leading to significant cost reductions ;
(3) adjoint vector and direct differentiation method have been defined , studied and more and more often

used to computate the gradients necessary for descent algorithms.

This part of the course focuses on local optimization methods. Section 1 and 2 gather general informations, basic definitions and theorems. Section 3 is dedicated to the broadly used simplex method. Section 4 is devoted to descent methods. Section 5 describes the adjoint and direct methods that can efficiently compute the gradient of the functions of interest with respect to the design parameters.

Section 3 and 4 are joint work with M. Marcelet (ONERA). Section 5 is joint work with R.P. Dwight (TU-Delft).

2 Notations, mathematical problem and properties

2.1 Mathematical optimization problem

In this first subsection, the classical notations of a mathematical finite-dimensional optimization problem are defined. Let α be the current vector of the input space (design vector). Let us denote n_f its dimension. The vector α is supposed to vary in D_α (the design space), a parallelepiped of $\mathbb{R}^{n_f}$.
The objective function to minimize on D_α is denoted $\mathcal{J}(\alpha)$. The constraints of the problem are supposed to be formulated through n_c functions $\mathcal{G}_j, \quad j \in [1, n_c]$, that are negative at admissible design points. Only inequality constraint are considered here as for most practical design problems an adequate choice of the design parameters allows to avoid equality constraints.
Obviously the local and global optimization problems read

• Global optimum search

$$Seek \quad for \quad \alpha^* \quad in \quad D_\alpha \quad such \quad that \quad \mathcal{J}(\alpha^*) = Min\, \mathcal{J}(\alpha) \quad on \quad D_\alpha$$
$$\forall j \in [1, n_c] \quad \mathcal{G}_j(\alpha^*) \leq 0.$$

• Local optimum search

$$Seek \quad for \quad \alpha^* \quad in \quad D_\alpha \quad such \quad that \quad \mathcal{J}(\alpha^*) = Min\, \mathcal{J}(\alpha) \quad on \quad V_{\alpha^*}$$
$$(V_{\alpha^*} \quad neighborhood \quad of \quad \alpha^*)$$
$$\forall j \in [1, n_c] \quad \mathcal{G}_j(\alpha^*) \leq 0.$$

In most common situations, the objective and constraint functions are at least continuous. In this course, the functions are supposed to have C^1 regularity, and in some sections, C^2 regularity.

2.2 Optimization problem stemming from a numerical simulation

In the framework of aircraft or turbomachinery design, the functions of interest depend on a distributed state field and geometrical variables that are linked by a system of equations discretizing physical partial differential equations on a computational domain. It is rather difficult to write a general presentation fitting all disciplines. For this reason this subsection is "aerodynamics-oriented".
Let us note $S(\alpha)$, the coordinates of the surface mesh of a solid body. This function is supposed to be C^1

regular. From any surface mesh $S(\alpha)$ a volumic mesh $X(\alpha)$ is built. This function is also supposed to have C^1 regularity. Its Jacobian $dX/d\alpha$ can always be estimated by finite-differences and in some cases by the following product of Jacobians

$$\frac{dX}{d\alpha} = \frac{dX}{dS}\frac{dS}{d\alpha}$$

The state variables (aerodynamic conservative variables at the center of the cells, for example, for a CFD simulation) are noted W (vector of size n_a). State variables and mesh satisfy the discrete equations of fluid mechanics (a discrete form of (RaNS) equations for example),

$$R(W, X) = 0$$

(in general, nonlinear set of n_a equations). These equations of mechanics are also supposed to have C^1 regularity with respect to (w.r.t.) its two vector arguments. For a specific mesh $X_i = X(\alpha_i)$, the equations for mechanics define a set of n_a nonlinear equations and n_a unknowns (the components of W). Let us now consider the state variable W_i associated with mesh X_i [1]. More precisely, we shall suppose that

$$R(W_i, X_i) = 0 \quad det[(\partial R/\partial W)(W_i, X_i)] \neq 0$$

The implicit funtion theorem allows us to define W as a C^1 function of X in a neighborhood of X_i, and then, thanks to the regularity of $X(\alpha)$, as a C^1 function of α in a neighborhood of α_i. For the remainig part of this subsection this property is assumed to be true for the whole design space D_α.
We may now use the notation $W(\alpha)$ (C^1 function from $\mathbb{R}^{n_f}$ *to* $\mathbb{R}^{n_a}$) and rewrite the discrete state equations

$$R(W(\alpha), X(\alpha)) = 0$$

Note that when $R(W, X)$ and $X(\alpha)$ have C^k ($k > 1$) regularity, the implicit function theorem garantees C^k regularity for $W(\alpha)$. This property is needed when computing the derivatives of order k of objective and constraints functions (with most often k=1 or 2).
In case of an optimization problem associated with a framework of numerical simulation, the objective function may be written

$$\mathcal{J}(\alpha) = J(W(\alpha), X(\alpha))$$

The constraint functions $\mathcal{G}_j(\alpha)$ have the same dependencies $\mathcal{G}_j(\alpha) = g_j(W(\alpha), X(\alpha))$. A shape, corresponding to a vector α (size n_f) is said to be admissible if and only if

$$\forall j \in [1, n_c] \quad g_j(W(\alpha), X(\alpha)) \leq 0.$$

(n_c number of contraint functions).

[1] in aerodynamics, for certain geometry and boundary conditions the steaty flow may not be unique. Such a troublesome situation is obviously not compatible with design optimization and is not discussed here.

- The global optimisation problem now reads

$$\begin{aligned} &Seek\ for\ \alpha^*\ in\ D_\alpha\ such\ that \\ &\mathcal{J}(\alpha^*) = J(W(\alpha^*), X(\alpha^*)) = Min\, \mathcal{J}(\alpha)\ \ sur\ \ D_\alpha \\ &\forall j \in [1, n_c] \quad g_j(W(\alpha^*), X(\alpha^*)) \leq 0. \end{aligned}$$

Obviously no optimization algorithm can pretend to systematically find one global optimum without more hypothesis, in particular convexity, than C^1 regularity of J and g_j (which has been adopted to allow derivatives computations).

- The local optimisation problem is also rewritten

$$\begin{aligned} &Seek\ for\ \ \alpha^*\ \ in\ \ D_\alpha\ \ such\ \ that \\ &\mathcal{J}(\alpha^*) = J(W(\alpha^*), X(\alpha^*)) = Min\, \mathcal{J}(\alpha)\ sur\ V_{\alpha^*} \\ &(V_{\alpha^*}\ being\ a\ neighborhood\ of\ \alpha^*) \\ &\forall j \in [1, n_c] \quad g_j(W(\alpha^*), X(\alpha^*)) \leq 0. \end{aligned}$$

2.3 The Karush-Kuhn-Tucker condition

For the unconstrained optimization of a C^2 function of $\mathbb{R}^{n_f}$, classical conditions of existence for minima read :
- local optimum located in α^* - $\nabla\mathcal{J}(\alpha^*) = 0$ is a necessary condition. $\nabla\mathcal{J}(\alpha^*) = 0$ and $H(\alpha^*)$ positive definite (H hessian matrix of $\mathcal{J}$) is a sufficient condition.
- global optimum located in α^* - $\nabla\mathcal{J}(\alpha^*) = 0$ is a necessary condition. $\nabla\mathcal{J}(\alpha^*) = 0$ and $H(\alpha)$ positive definite on $\mathbb{R}^{n_f}$ is a sufficient condition.
For a constrained problem on a finite size domain, the necessary condition for optimality is more complex to express. Actually, we first have to introduce more explicit notations for the parallelepiped design space D_α:

$$D_\alpha = [\alpha_{1,l}, \alpha_{1,u}] \times [\alpha_{2,l}, \alpha_{2,u}] \times [\alpha_{3,l}, \alpha_{3,u}] \times ... \times [\alpha_{n_f,l}, \alpha_{n_f,u}]$$

Then the domain bounds are rewritten as $2n_f$ additionnal constraints:

$$\left\{ \begin{array}{l} \mathcal{G}_{n_c+1}(\alpha) = \alpha_{1,l} - \alpha_1 \\ \mathcal{G}_{n_c+2}(\alpha) = \alpha_1 - \alpha_{1,u} \\ \mathcal{G}_{n_c+3}(\alpha) = \alpha_{2,l} - \alpha_2 \\ ... \\ \mathcal{G}_{n_c+2n_f}(\alpha) = \alpha_{n_f} - \alpha_{n_f,u} \end{array} \right.$$

For an optimization problem with inequality constraints the Karush-Kuhn-Tucker conditions are:

KKT conditions

$$\begin{array}{l} \alpha^* \text{is an admissible state} \\ \nabla \mathcal{J}(\alpha^*) + \Sigma \lambda_j^* \nabla \mathcal{G}_j(\alpha^*) = 0 \\ \lambda_j^* \mathcal{G}_j(\alpha^*) = 0 \qquad \lambda_j^* \geq 0. \end{array}$$

The third line means that only the constraints attaining the limit value zero may have their gradient included in the linear combination of the second line.

By introducing a lagrangian $\mathcal{L}(\alpha, \lambda_1, \cdots, \lambda_{n_c}) = \mathcal{J}(\alpha) + \Sigma \lambda_j \mathcal{G}_j(\alpha)$, we can write KKT conditions:

KKT conditions (alternative formulation)

$$\begin{array}{l} \alpha^* \text{is an admissible state} \\ \nabla_\alpha \mathcal{L}(\alpha^*, \lambda_1^*, \cdots, \lambda_{n_c'}^*) = 0 \\ \lambda_j^* \mathcal{G}_j(\alpha^*) = 0 \qquad \lambda_j^* \geq 0. \end{array}$$

The Karush-Kuhn-Tucker condition is a necessary condition for optimality. This condition is the counterpart for constrained problems of the necessary condition $\nabla \mathcal{J} = 0$ for unconstrained problems. It is a sufficient condition only when objective and constraint functions are convex.

As this condition is not very intuitive, it is suggested that the reader draws some sketches for $n_f = 2$ with a minimum reached inside D_α
- with one inequality constraint attaining its bound $\mathcal{G}_1(\alpha) = 0$. Drawing iso-lines of $\mathcal{J}$ and $\mathcal{G}_1$ helps understanding the theorem ;
- with two inequality constraints attaining their bound $\mathcal{G}_1(\alpha) = 0$, $\mathcal{G}_2(\alpha) = 0$. Drawing iso-lines of the three functions of the problem also helps understanding the theorem.

2.4 Solving KKT conditions for elementary problems in $\mathbb{R}^2$

In this section, three examples of solution of KKT optimality conditions are given. To improve readability, we set $\alpha = \alpha_1$ and $\beta = \alpha_2$.

2.4.1 With one linear inequality constraint

The following problem is considered:

$$\begin{cases} \text{Min } \alpha^2 + 3\beta^2 \\ \text{subject to } g_1(\alpha, \beta) = 7 - 2\alpha - 3\beta \leq 0 \end{cases}$$

The corresponding Lagrangian function is:

$$\mathcal{L}(\alpha, \beta, \lambda) = \alpha^2 + 3\beta^2 + \lambda(7 - 2\alpha - 3\beta)$$

The KKT conditions are:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \alpha}(\alpha^*, \beta^*, \lambda_1^*) = 2\alpha^* - 2\lambda_1^* = 0 \\ \frac{\partial \mathcal{L}}{\partial \beta}(\alpha^*, \beta^*, \lambda_1^*) = 6\beta^* - 3\lambda_1^* = 0 \\ g_1(\alpha^*, \beta^*) \leq 0 \quad \lambda_1^* \geq 0 \quad \lambda_1^* g_1(\alpha^*, \beta^*) = 0 \end{cases}$$

◁ First case: constraint g_1 is active. Following system has to be solved:

$$\begin{aligned} 2\alpha^* - 2\lambda_1^* &= 0 \\ 6\beta^* - 3\lambda_1^* &= 0 \\ 7 - 2\alpha^* - 3\beta^* &= 0 \end{aligned}$$

The first two equations yield $\alpha^* = 2\beta^* = \lambda_1^*$. With the third, we obtain $(\alpha^*, \beta^*, \lambda_1^*) = (2, 1, 2)$

◁ Second case: constraint g_1 is not active. Following system has to be solved:

$$\begin{aligned} 2\alpha^* &= 0 \\ 6\beta^* &= 0 \\ \lambda_1^* &= 0 \end{aligned}$$

But the solution $(0, 0, 0)$ gives $g_1(\alpha^*, \beta^*) = 7 > 0$, in contradiction with the hypothesis.

◁ Conclusion. The KKT conditions give the following unique solution : $(\alpha^*, \beta^*) = (2, 1)$ and $\lambda_1^* = 2$.

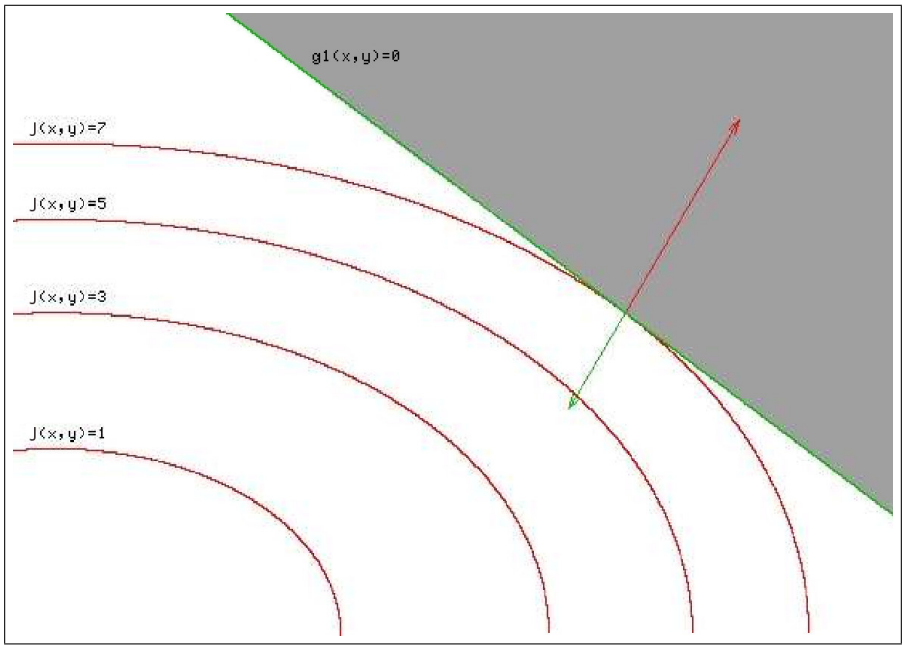

Figure 1: Solution of problem 1

2.4.2 With one nonlinear inequality constraint

The following problem is considered:

$$\begin{cases} \text{Min } \alpha^2 + 3\beta^2 \\ \text{subject to } g_2(\alpha,\beta) = 1 - \alpha + \beta^2 \leq 0 \end{cases}$$

The solution of the KKT conditions yield $(\alpha^*, \beta^*) = (1, 0)$ and $\lambda_2^* = 2$:

$$\begin{aligned} \nabla J(1,0) + \lambda \nabla g_2(1,0) = 0 \quad &\Leftrightarrow \quad \begin{bmatrix} 2 \\ 0 \end{bmatrix} + \lambda \begin{bmatrix} -1 \\ 0 \end{bmatrix} = 0 \\ &\Leftrightarrow \quad \begin{bmatrix} 2 \\ 0 \end{bmatrix} = \lambda \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{aligned}$$

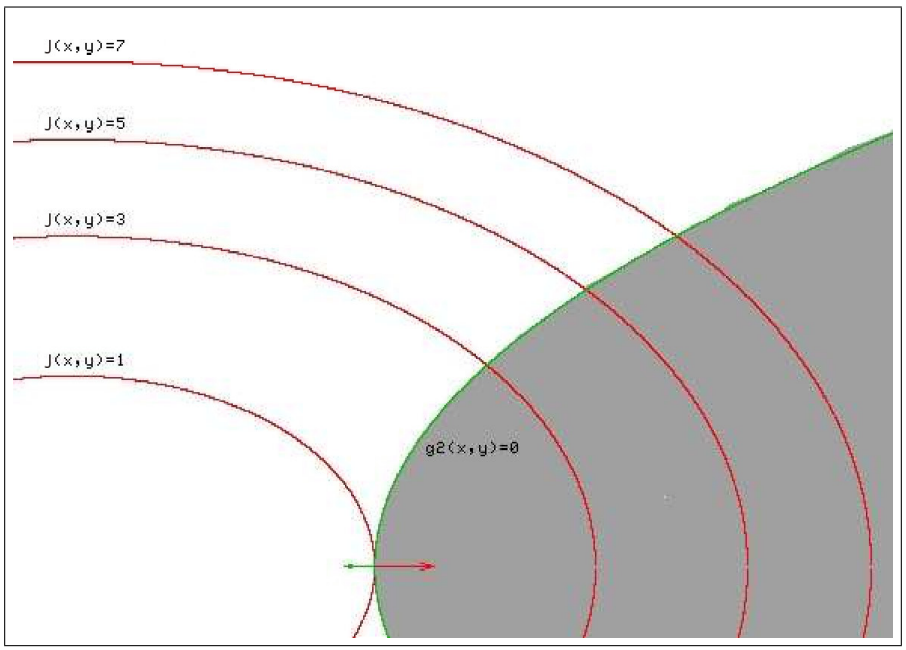

Figure 2: Solution of problem 2

2.4.3 With two inequality constraints

The following problem is considered:

$$\begin{cases} \text{Min } \alpha^2 + 3\beta^2 \\ \text{subject to } g_2(\alpha,\beta) = 1 - \alpha + \beta^2 \leq 0 \\ \text{and } g_3(\alpha,\beta) = 4 - \alpha - 2\beta \leq 0 \end{cases}$$

The corresponding Lagrangian function is:

$$\mathcal{L}(\alpha, \beta, \lambda_2, \lambda_3) = \alpha^2 + 3\beta^2 + \lambda_2(1 - \alpha + \beta^2) + \lambda_3(4 - \alpha - 2\beta)$$

The KKT conditions are:

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial \alpha}(\alpha^*, \beta^*, \lambda_2^*, \lambda_3^*) = 2\alpha^* - \lambda_2^* - \lambda_3^* = 0 \\ \frac{\partial \mathcal{L}}{\partial \beta}(\alpha^*, \beta^*, \lambda_2, \lambda_3) = 6\beta^* + 2\lambda_2^*\beta^* - 2\lambda_3^* = 0 \\ g_2(\alpha^*, \beta^*) \leq 0 \;\; \lambda_2 \geq 0 \;\; \lambda_2 g_2(\alpha^*, \beta^*) = 0 \\ g_3(\alpha^*, \beta^*) \leq 0 \;\; \lambda_3 \geq 0 \;\; \lambda_3 g_3(\alpha^*, \beta^*) = 0 \end{cases}$$

◁ First case: the two constraints are active. The following system of equations has to be solved:

$$\begin{aligned} 2\alpha^* - \lambda_2^* - \lambda_3^* &= 0 \\ 2\beta^*(3 + \lambda_2^*) - 2\lambda_3^* &= 0 \\ 4 - \alpha^* - 2\beta^* &= 0 \\ 1 - \alpha^* + (\beta^*)^2 &= 0 \end{aligned}$$

The last two equations are equivalent to $(\beta^*)^2 + 2\beta^* - 3 = 0$ and $\alpha^* = 1 + (\beta^*)^2$. The solutions for β of the first equation are -3 and +1.

1. $\beta^* = -3$ gives $\alpha^* = 10$. Solving for the 2x2 linear system $(\lambda_2^*, \lambda_3^*)$ leads to a negative λ_2^*;
2. $\beta^* = 1$ gives $\alpha^* = 2$. Solving for the 2x2 linear system $(\lambda_2^*, \lambda_3^*)$ yields $(\lambda_2^*, \lambda_3^*) = (\frac{1}{2}, \frac{7}{2})$

◁ Second case : only constraint g_3 is active. The following system of equations is to be solved:

$$\begin{aligned} 2\alpha^* - \lambda_3^* &= 0 \\ 6\beta^* - 2\lambda_3^* &= 0 \\ 4 - \alpha^* - 2\beta^* &= 0 \\ \lambda_2^* &= 0 \end{aligned}$$

The first two equations give $2\alpha^* = 3\beta^* = \lambda_3^*$. With the third equation, we obtain $(\alpha^*, \beta^*, \lambda_3^*) = (\frac{12}{7}, \frac{8}{7}, \frac{24}{7})$. At this point, the constraint g_2 is violated ($g_2(\frac{12}{7}, \frac{8}{7}) = \frac{29}{49} > 0$).

◁ Third case: only constraint g_2 is active. The following system of equations needs to be solved:

$$\begin{aligned} 2\alpha^* - \lambda_2^* &= 0 \\ 2\beta^*(3 + \lambda_2^*) &= 0 \\ \lambda_3^* &= 0 \\ 1 - \alpha^* + (\beta^*)^2 &= 0 \end{aligned}$$

In the second equation, we retain only the option $\beta^* = 0$. So we obtain $(\alpha^*, \beta^*, \lambda_2^*) = (1, 0, 2)$. At this point, the constraint g_3 is violated ($g_3(1, 0) = 3 > 0$).

◁ Fourth case: Constraints g_2 and g_3 are inactive. The following system of equations needs to be solved:

$$\begin{aligned} 2\alpha^* &= 0 \\ 6\beta^* &= 0 \\ \lambda_2^* &= 0 \\ \lambda_3^* &= 0 \end{aligned}$$

But the solution $(\alpha, \beta, \lambda_2^*, \lambda_3^*) = (0, 0, 0, 0)$ leads to $g_3(\alpha^*, \beta^*) = 4 > 0$ and $g_2(\alpha^*, \beta^*) = 1 > 0$.

◁ Conclusion. The unique minimun of our problem is reached at $(2, 1)$, as it can be checked on the graphics below. Let us also note that $\lambda_2^* = \frac{7}{2}$ and $\lambda_3^* = \frac{1}{2}$ are the coefficients of the linear combination of the gradient of the objective function and the (active) constraints that satisfies the first KKT condition $\nabla_\alpha \mathcal{L} = 0$. This is illustrated by figure 3.

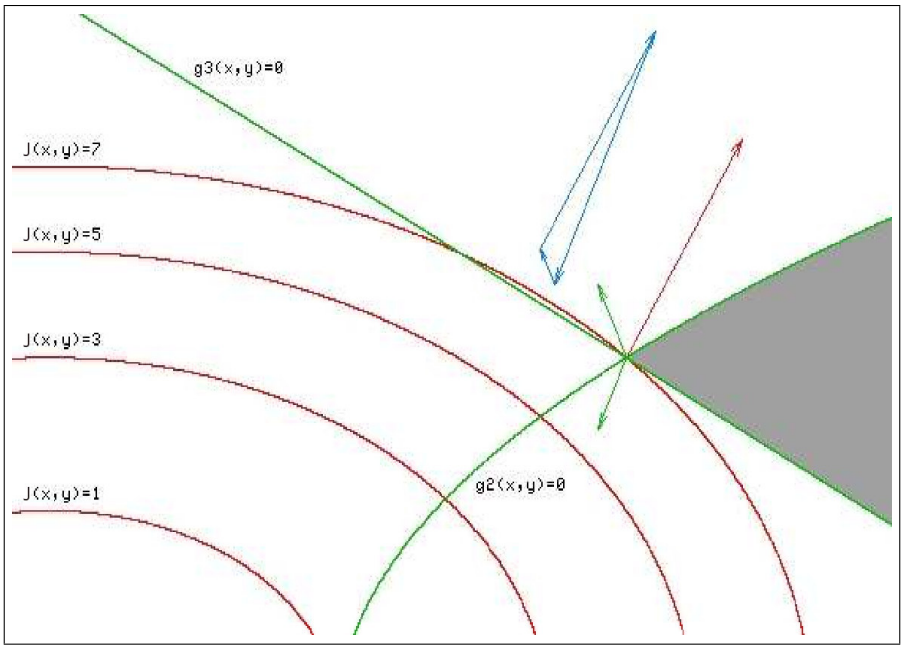

Figure 3: Solution of problem 3

3 Simplex method

This method is both deterministic and simple (as it uses only values of the function of interest). It was introduced in 1965 by Nelder et Mead [2] and appeared to be efficient, robust and easy to code. Actually, its robustness allows in some circumstances to subtitute approximate values of the function of interest to the exact ones. The simplex algorithm was the starting point for the definition of many derived algorithms called "pattern search methods".
It is supposed here that the whole search path is located in an admissible region or that no constraint appears

in the minimization problem. In opposition to classical descent methods, the initialisation procedure requires the evaluation of the function of interest for a set of $n_f + 1$ points (the vertices of a *simplex* of $\mathbb{R}^{n_f}$). Most often this set is built around a specific point α^1, the n_f other points being defined by following relationship [2] :

$$\alpha^{i+1} = \alpha^1 + \lambda_i e^i \quad i = 1, ..., n_f$$

where vectors e^i, $(i = 1, ..., n_f)$ define a basis of $\mathbb{R}^{n_f}$ The scalar factors λ_i, $(i = 1, ..., n_f)$ are constants. Most often they are all equal to a single value. The set of $n_f + 1$ points can be called either $polytop$ or $simplex$ and is denoted $P =< \alpha^1, \alpha^2, ..., \alpha^{n_f+1} >$. The minimization algorithm proceeds as follows :

1. Sort the points so that :
$$\mathcal{J}(\alpha^1) \leq \mathcal{J}(\alpha^2) \leq ... \leq \mathcal{J}(\alpha^{n_f+1})$$
2. The center of gravity of the $(\alpha^1, \alpha^2, ..., \alpha^{n_f})$ (all points of simplex P but the worst) is considered and denoted $\overline{\alpha}$
3. A new point (α^r) is defined by reflection of the worst point of the polytop w.r.t the center of gravity
$$\alpha^r = (1 + a)\overline{\alpha} - a\alpha^{n_f+1}$$
(coefficient $a > 0$ is called the reflection coefficient of the method)
   (a) If $\mathcal{J}(\alpha^r) < \mathcal{J}(\alpha^1)$, α^r if a very interesting point for the purpose of minimization. A trial is made to go further in the direction of α^r. Following point is defined
   $$\alpha^e = b\alpha^r + (1 - b)\overline{\alpha}$$
   where $b > 0$ is called the expansion coefficient of the polytop. If $J(\alpha^e) < J(\alpha^r)$ then α^e is selected to replace α^{n_f+1}. The algorithm goes back to step 1. Otherwise α^r is finaly selected to replace α^{n_f+1} ; The algorithm goes back to step 1.
   (b) If $\mathcal{J}(\alpha^1) \leq J(\alpha^r) < J(\alpha^{n_f})$ then α^r is accepted to replace α^{n_f+1} ; The algorithm goes back to step 1.
   (c) If $\mathcal{J}(\alpha^n) \leq \mathcal{J}(\alpha^r) < \mathcal{J}(\alpha^{n_f+1})$ then the algorithm tries an internal contraction by evaluating the point
   $$\alpha^c = c\overline{\alpha} + (1 - c)\alpha^r$$
   where c is the contraction coefficient. If $\mathcal{J}(\alpha^c) < \mathcal{J}(\alpha^r)$ then α^c is kept to replace α^{n_f+1} ; Execution goes back to step 1. Otherwise α^r is finaly accepted to replace α^{n_f+1} ; Algorithm goes back to step 1.
   (d) If $\mathcal{J}(\alpha^{n_f+1}) \leq \mathcal{J}(\alpha^r)$ then α^r is a very bad point for the purpose of the minimization. The algorithm tries an internal contraction of the polytop
   $$\alpha^c = c\overline{\alpha} + (1 - c)\alpha^{n_f+1}$$
   If $\mathcal{J}(\alpha^c) < \mathcal{J}(\alpha^{n_f+1})$ then α^c is selected to replace α^{n_f+1} ; Algorithm goes back to step 1. Otherwise the whole simplex is shrinked. All points of the polytop but the best $\alpha^i, i =$

[2] caution : do not mix the lower index of vector components in $\alpha = (\alpha_1, \alpha_2, ..., \alpha_{n_f})$ with the upper index of design vector number

$2, ..., n_f + 1$ are replaced by

$$\alpha^i = d\alpha^i + (1-d)\alpha^1$$

where d is the shrinking coefficient of the algorithm.

The algorithm is stopped when the displacement of polytop becomes small enough, that is

$$\frac{1}{n}\sum_{i=1}^{n_f} ||\alpha_i^{k+1} - \alpha_i^k||^2 < \epsilon$$

(i index of points in the sorting of step (1), k index of iterations) Standard values for reflection (a), expansion (b), contraction (c) and shrinking coefficients(d) are :

$$(a, b, c, d) = (1, 2, 1/2, 1/2)$$

The location of the candidate new points is illustrated in figure 4 for $n_f = 3$.

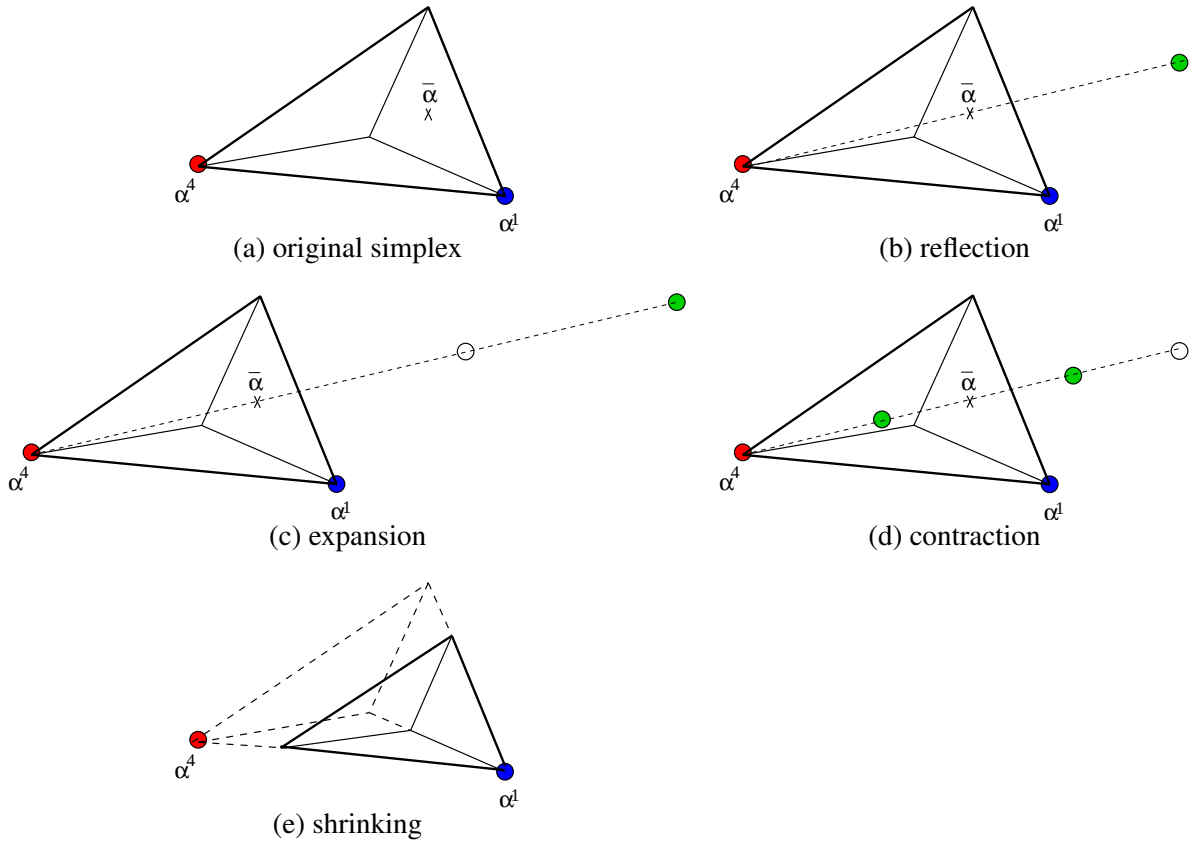

Figure 4: simplex – search of new points

4 Descent methods

The graphics in this section are adapted from those of the classical book of G.N. Vanderplaats "Numerical optimization techniques for engineering design: with applications." [3] or from http://www.mathworks.com.

A descent method is an iterative method for the solution of a local optimization problem. Starting from an initial point, it attempts to converge to an optimum, using local information at the current point to compute the next iterate.

In the following pages, we first present the most popular algorithm for 1D minimization. Then, we focus on some unconstrained optimization methods before presenting constrained optimisation methods.

4.1 Function of one variable

In many algorithms of multi-dimensionnal optimization, after a descent direction d^k has been defined, a one dimensional minimization along d_k is performed. This means that a step t is sought to diminish significantly $q(t) = \mathcal{J}(\alpha^k + td^k)$. A good line-search is obviously desirable, but the number of exact evaluations of $\mathcal{J}$ must remain as low as possible : the goal is not to find the optimal $t > 0$ at an intermediate step but to reduce is the objective function $\mathcal{J}$ efficiently at each iteration. Conversely a bad line-search can slow down the global algorithm and a compromise must be found between the performance of the line-search and the number of evaluations of $\mathcal{J}$.
This minimization along a descent direction during multi-dimensionnal optimization is one of the reasons for the interest in 1D-minimization.

4.1.1 Academic methods

Methods of order 0 Methods of order 0 can be used to find the minimum of a function of one variable in case function values only can be computed.

Algorithm 1 Methods of order 0

Set $t_L = a$, $t_R = b$ and $k = 1$;
while convergence level not reached **do**
  Choose search point t_k^- and t_k^+;
  Compute $q(t_k^-)$ and $q(t_k^+)$;
  if $q(t_k^-) \leq q(t_k^+)$ **then**
    $t_R = t_k^+$
  else
    $t_L = t_k^-$
  end if
  $k = k + 1$
end while

Various strategies to choose the search point have been developed:

- if the original interval is divided in three equal parts, it can be shown that the convergence is linear with a rate of $\sqrt{\frac{2}{3}}$;

- if the golden number ($\lambda = \frac{1+\sqrt{5}}{2}$) is used ($t_k^- = \frac{\lambda}{1+\lambda}t_L + \frac{1}{1+\lambda}t_R$ and $t_k^+ = \frac{1}{1+\lambda}t_L + \frac{\lambda}{1+\lambda}t_R$). One of the two points of step k, is also a search point of step $k+1$. The convergence rate is then $\frac{1}{\lambda}$.

Dichotomy In some cases, the considered function is smooth and the derivative of $q(t)$ is known. It can then be used to improve the descent rate.

Algorithm 2 Dichotomy

Set $t_L = a$, $t_R = b$,$k = 1$ and stop=`false`;
while *not* stop **do**
  Choose search point t_k as $t_k = \frac{t_L+t_R}{2}$;
  Compute $q'(t_k)$;
  if $q'(t_k^-) = 0$ **then**
    stop=`true`;
  else
    if $q'(t_k^-) > 0$ **then**
      $t_R = t_k$
    else
      $t_L = t_k$
    end if
    $k = k + 1$
  end if
end while

It can be shown that this method converges linearly with a rate of 0.5. There are many other methods of academic interest to find the minimum of function of one variable ; all of them use a large number of evaluations of $q(t)$ (possibly $q'(t)$). For industrial applications, a satisfactory method is based on a compromise between the accuracy of the computed solution and the required number of evaluations of $q(t)$ (each of them, of course, requiring an evaluation of $\mathcal{J}$).

4.1.2 Wolfe line-search

From this point of view, in case the derivative of $q(t)$ is available, one of the most efficient methods is the Wolfe line-search, based on the following rules (where $0 < m_1 < m_2 < 1$):

- a step is too large if $q(t) > q(0) + m_1 t q'(0)$;
- a step is too small if $q'(t) < m_2 q'(0)$;
- else the step is satisfactory.

This rules are illustrated on the figure 5.
Based on these rules, the algorithm is given hereafter.
In many applications, obtaining derivatives $q'(t)$ of $q(t)$ is computationally expensive. One adapted method of Wolfe linear search, called the Goldstein and Price method, replaces $q'(t)$ by the average slope $\frac{q(t)-q(0)}{t}$. It is described in the next subsection.

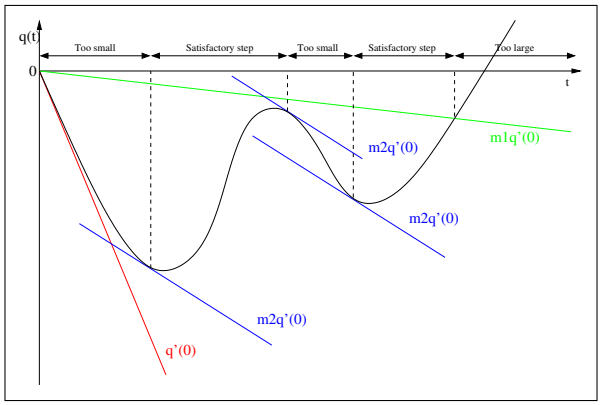

Figure 5: Wolfe's rules

Algorithm 3 Wolfe line-search

Set $t_L = 0$, $t_R = +\infty$ and stop=`false`; Choose an initial stepsize t and value for parameters m_1 and m_2 with $0 < m_1 < m_2 < 1$ and $\beta > 1$;
while *not* stop **do**
  if $q(t) > q(0) + m_1 t q'(0)$ or $q'(t) < m_2 q'(0)$ **then**
    if $q(t) > q(0) + m_1 t q'(0)$ **then**
      $t_R = t$ (*current step is too large*)
    end if
    if $q'(t) < m_2 q'(0)$ **then**
      $t_L = t$ (*current step is too small*)
    end if
    if t_R is real **then**
      $t = \frac{t_L + t_R}{2}$
    else
      $t = \beta t_L$
    end if
  else
    stop=`true` (*current step is satisfactory*)
  end if
end while

4.1.3 Goldstein and Price method

By substituting $q'(t)$ by $\frac{q(t)-q(0)}{t}$, condition which defines too small step becomes :

$$q(t) - q(0) < m_2 t q'(0)$$

The definition of too small and too large steps becomes :

- a step is too large if $q(t) > q(0) + m_1 t q'(0)$;

- a step is too small if $q(t) < q(0) + m_2 t q'(0)$;
- else the step is satisfactory.

It is equivalent to say that a step is satisfactory if the average slope is bounded by $m_1 q'(0)$ and $m_2 q'(0)$.

Even with these modifications, Wolfe line-search requires a large number of objective funtion evaluations. In most industrial cases, it is replaced by a parabolic-approximation approach.

4.1.4 Polynomial approximations

Polynomial approximations is one of the most effective techniques for finding the minimum of a function of one variable. The method has the advantage of requiring only a few function evaluations at each step. Conversely, it can lead to a very poor approximation for highly nonlinear functions. The approximation with a polynomial of degree 2 is presented:

$$\tilde{q}(t) = a_0 + a_1 t + a_2 t^2$$

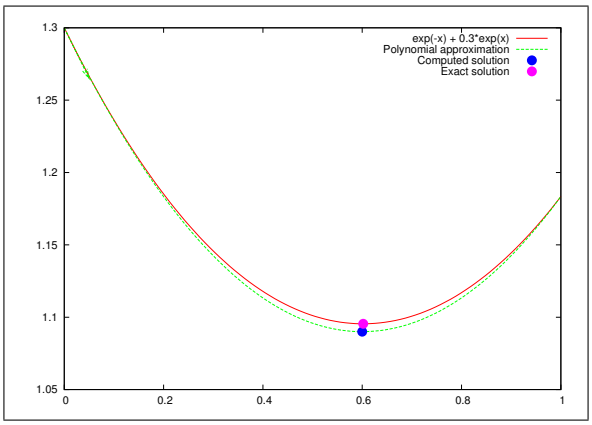

Figure 6: Polynomial approximations

The evaluation of the function q and its first derivative q' at $t_0 = 0$ leads to the following system $(a_0, a_1) = (q(t_0), q'(t_0))$. Evaluation of the function q in an another point t_1 is needed to compute the coefficient a_2:

$$\begin{aligned} a_0 + a_1 t_1 + a_2 t_1^2 = q(t_1) \quad &\Leftrightarrow \quad q(t_0) + q'(t_0) t_1 + a_2 t_1^2 = q(t_1) \\ &\Leftrightarrow \quad a_2 = \frac{q(t_1) - a_0 - a_1 t_1}{t_1^2} \end{aligned}$$

After the polynomial coefficients are found, the location of the minimum of the polynomial is identified, which is simple and inexpensive. For a polynomial approximation of degree 2, the result is obvious ($t^* = \frac{-a_1}{2a_2}$). More information about polynomial approximations can be found in Vanderplaats[3].

4.1.5 Line-search for constrained optimization

Section 4.3 will describe several algorithms adapted for constrained optimization and requiring a line-search. In this case, the methods presented in previous sections 4.1.[1-4] must be generalized and applied to function $r(t) = \Phi(\alpha_k + td_k)$ instead of $q(t) = \mathcal{J}(\alpha_k + td_k)$. The function Φ includes a penalty term to account for non-admissible areas of the design space. The most popular penalty function is the following :

$$\Phi(\alpha) = \mathcal{J}(\alpha) + \sum_{i=1}^{n_c} \sigma_i g_i^+(\alpha) \tag{1}$$

$$g_i^+(\alpha) = \min(0, g_i(\alpha))^2 \tag{2}$$

4.2 Descent methods for unconstrained optimization

In this part, we focus on the search of an optimum of the objective function on the design space D_α. In this section an the next one, the sequence of points of several algorithms is illustated in 2D for Rosenbrock banana function :

$$\mathcal{J}(\alpha_1, \alpha_2) = (\alpha_1 - 1)^2 + 100(\alpha_2 - \alpha_1^2)^2$$

4.2.1 Steepest Descent Methods

The most intuitive method uses, at each iteration, the opposite of the gradient as the descent direction.

Algorithm 4 Steepest Descent algorithm with fixed step

We set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ is given;
while $\|\nabla\mathcal{J}(\alpha^k)\| \geq \epsilon$ **do**
  Compute $d^k = -\nabla\mathcal{J}(\alpha^k)$.
  Update current iterate: $\alpha^{k+1} = \alpha^k + d^k$.
  Set $k = k + 1$
end while

This method can be improved by finding, at each iteration, one optimal step by resolving 1D optimisation problem by methods presented above.

Algorithm 5 Steepest Descent algorithm with optimal step

We set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ is given;
while $\|\nabla\mathcal{J}(\alpha^k)\| \geq \epsilon$ **do**
  Compute $d^k = -\nabla\mathcal{J}(\alpha^k)$.
  Find t^* by line-search on $q(t) = J(\alpha^k + td^k)$.
  Update current iterate: $\alpha^{k+1} = \alpha^k + t^* d^k$.
  Set $k = k + 1$
end while

Actually, theses methods are short-sighted and have a zig-zagging behaviour for simple test cases (for example, with the Rosenbrock banana function, see figure 7). This behaviour affects convergence speed and

robustness of this method. Conjugate gradient methods is one of the mean developped to circumvent these difficulties.

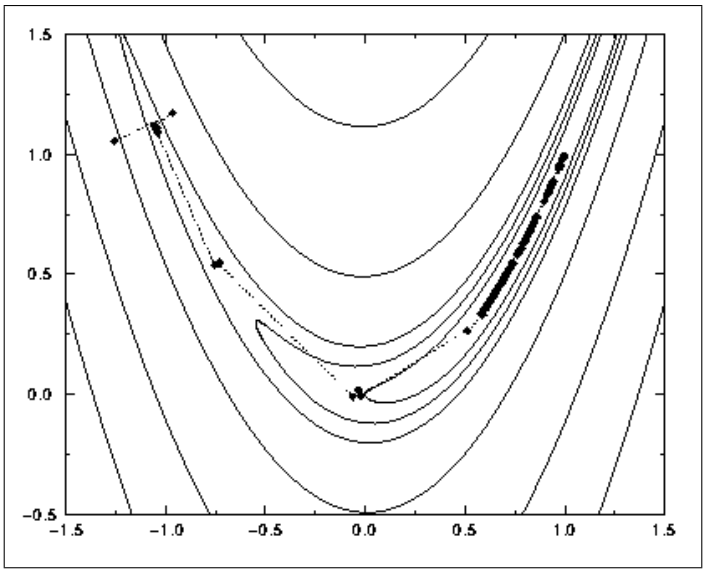

Figure 7: Steepest descent path for Rosenbrock banana function

4.2.2 Conjugate Gradient methods

The idea behind Conjugate gradient methods comes from the analysis of the behaviour of the steepest descent for the specific case of positive definite quadratic forms ($\mathcal{J}(\alpha) = \frac{1}{2}\alpha^T H\alpha + b^T\alpha$). In this case, the conditioning of positive definite matrix H strongly affects the robustness and convergence speed of the steepest descent. To improve robustness, Conjugate Gradient methods uses at step k a descent direction d_k orthogonal to d_{k-1} in the sense of H - $\left(d^{k-1}\right)^T Hd^k = 0$.

$$d^k = -\nabla\mathcal{J}(\alpha^k) + \beta^k d^{k-1} \quad \beta^k = \frac{\left(\nabla\mathcal{J}(\alpha^k)\right)^T Hd^{k-1}}{\left(d^{k-1}\right)^T Hd^{k-1}}$$

In this simple case (positive definite matrix H), the search of the unique minimum of $\mathcal{J}$ is equivalent to the resolution of the linear system $H\alpha + b = 0$ (search for the unique point where gradient of $\mathcal{J}$ is zero) and it can be proven that the algorithm converges to the unique solution in n_f or less steps.

Two formulas have been proposed for computation of β^k for extension to non quadratic cases. The first one is based on an other formula of β^k in the quadratic positive definite case $\beta_k = \frac{\|\nabla\mathcal{J}(\alpha^k)\|^2}{\|\nabla\mathcal{J}(\alpha^{k-1})\|^2}$ which can be directly applied to a non-quadratic function as it does not refer anymore to matrix H. The second extension, proposed by Polak and Gibière in 1969, reduces also to the same algorithm in the quadratic positive definite case (see below, as then $\nabla\mathcal{J}(\alpha^k)^T.\nabla\mathcal{J}(\alpha^{k-1}) = 0$). It is known to lead to a more efficient algorithm for specific applications.

- Fletcher-Reeves

$$\beta^k = \frac{\|\nabla \mathcal{J}(\alpha^k)\|^2}{\|\nabla \mathcal{J}(\alpha^{k-1})\|^2}$$

- Polak-Ribiere

$$\beta^k = \frac{\|\nabla \mathcal{J}(\alpha^k)\|^2}{\|\nabla \mathcal{J}(\alpha^{k-1})\|^2} - \frac{\left(\nabla \mathcal{J}(\alpha^k)\right)^T \nabla \mathcal{J}(\alpha^{k-1})}{\|\nabla \mathcal{J}(\alpha^{k-1})\|^2}$$

Algorithm 6 Conjugate Gradient algorithm

We set $k = 0$ and $d_{-1} = 0$; an initial iterate α^0 and a stopping tolerance ϵ is given;
while $\|\nabla \mathcal{J}(\alpha^k)\| \geq \epsilon$ **do**
  Compute $d^k = -\nabla \mathcal{J}(\alpha^k) + \beta^k d^{k-1}$.
  Find t^* by line-search on $q(t) = J(\alpha^k + t d^k)$.
  Update current iterate: $\alpha^{k+1} = \alpha^k + t^* d^k$.
  Set $k = k + 1$
end while

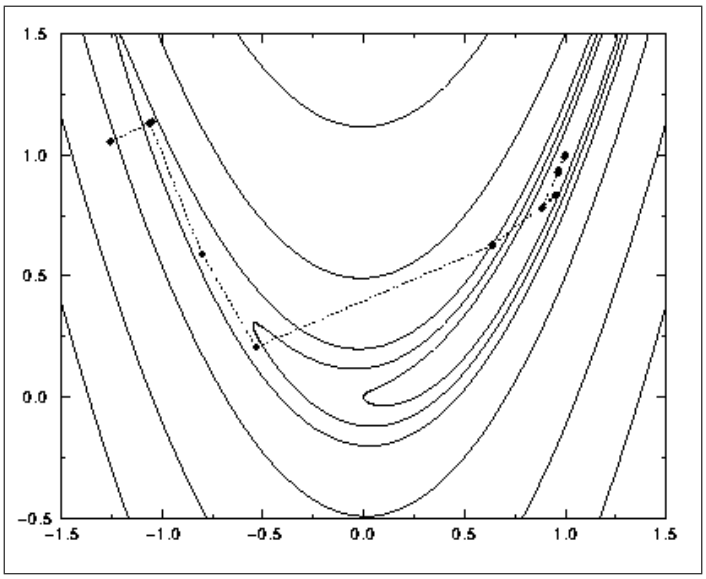

Figure 8: Conjugate Gradient path for Rosenbrock banana function

A more efficient algorithm requires more information, in particular information about the second derivatives of the function to minimize. Newton methods have been devised for this reason.

4.2.3 Newton methods

The classical Newton method is originally a method to find the roots of the equation $z(\alpha) = 0$ by approximating the function z by successive linear expansions. Starting from the current iterate α^k, substituting z by its linear approximation leads to :

$$z(\alpha^k + d^k) = z(\alpha^k) + \nabla z(\alpha^k)d^k + o(\|d^k\|). \tag{3}$$

Neglecting the term $o(\|d^k\|)$ yields $z(\alpha^k)+\nabla z(\alpha^k)d^k = 0$, the solution of which, $d^k = -\left[\nabla z(\alpha^k)\right]^{-1} z(\alpha^k)$, allows to calculate the next iterate $\alpha^{k+1} = \alpha^k + d^k$.

The same method can be used as an optimization algorithm, by solving the optimality equation presented in section 3. In this case, the function $z(\alpha)$ becomes the gradient $\nabla\mathcal{J}$ of the objective function $\mathcal{J}$ and the gradient ∇z, its Hessian $\nabla^2\mathcal{J}$. At each iteration k, the descent direction has to be computed by the formula $d^k = -\left[\nabla^2\mathcal{J}(\alpha^k)\right]^{-1}\nabla\mathcal{J}(\alpha^k)$.

Algorithm 7 Newton algorithm

We set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given;
while $\|\nabla\mathcal{J}(\alpha^k)\| \geq \epsilon$ **do**
  Compute $d^k = -\nabla^2\mathcal{J}(\alpha^k)\nabla\mathcal{J}(\alpha^k)$.
  Update current iterate: $\alpha^{k+1} = \alpha^k + d^k$.
  Set $k = k + 1$
end while

The important advantage of this method resides in its convergence in the neighborhood of the solution, which is superlinear in general and quadratic (at each iteration, the number of exact digits is doubled) if $\mathcal{J}$ has C^3 regularity.

Besides, the drawbacks of Newton's method are also well-known:

- the Hessian is required: in most engineering problems, an explicit form of the objective function is unavailable. The Hessian must be computed numerically which requires a large number of the objective function evaluations ;
- in high dimensional spaces, the solution of the linear system at each iteration is very CPU demanding ;
- Newton methods diverge violently far from the optimal point ;
- this algorithm converges on the closest stationary point, not necessarily the global minimum.

Quasi-Newton methods were developed to circumvent these drawbacks.

4.2.4 Principle of Quasi-Newton methods

Considering these drawbacks, the quasi-Newton realizes an improvement over the above Newton method based on two main ideas:

- First, the stability problems of the method can be avoided by adding a line-search process in the algorithm. Actually, noting that the requirement $\mathcal{J}(\alpha^{k+1}) < \mathcal{J}(\alpha^k)$ enforces stability, the Newton

increment d^k is considered as a direction, along which a line-search is performed to diminish the function $q(t) = \mathcal{J}(\alpha^k + td^k)$. It can be proved that the line-search is possible if and only if the Hessian of the objective function is positive definite. Line-search algorithms are presented in section (4.1).

- Secondly, rather than computing the Hessian matrix, its inverse is approximated by a matrix $\bar{H}$ which evolves during the iterations. This matrix can be chosen symmetric positive definite. An algorithm following this approach will be presented in the next section.

Algorithm (8) describes the steps of a generic quasi-Newton algorithm.

Algorithm 8 Generic quasi-Newton algorithm

Set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given; an initial matrix $\bar{H}^0$, positive definite (generally the identity matrix) is also chosen.
while $\|\nabla\mathcal{J}(\alpha^k)\| \geq \epsilon$ **do**
  Compute $d^k = -\bar{H}^k\nabla\mathcal{J}(\alpha^k)$.
  Make a line-search initialized by $t = 1$.
  Update current iterate: $\alpha^{k+1} = \alpha^k + td^k$.
  Compute the new matrix $\bar{H}^{k+1}$.
  Set $k = k + 1$
end while

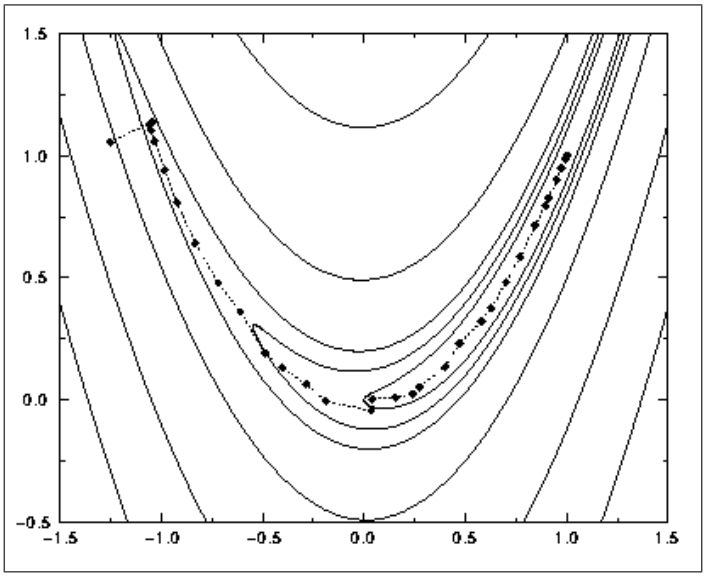

Figure 9: quasi-Newton path on Rosenbrock banana function

4.2.5 BFGS method

Let us now explain how the matrix $\bar{H}$ evolves during the iterations. The method presented here was introduced by C. Broyden[4], R. Fletcher[5], D. Goldfarb[6] and D. Shanno[7]. It certainly is the most popular quasi-Newton algorithm. The matrix $\bar{H}^{k+1}$ is computed from $\bar{H}^k$ and some vectors attached to iteration k and $k+1$ computations according to following formulas

$$\begin{aligned} s^k &= \alpha^{k+1} - \alpha^k \\ y^k &= \nabla\mathcal{J}(\alpha^{k+1}) - \nabla\mathcal{J}(\alpha^k) \\ \bar{H}^{k+1} &= \bar{H}^k - \frac{s^k(y^k)^T\bar{H}^k + \bar{H}^k y^k (s^k)^T}{(y^k)^T s^k} + \left[1 + \frac{(y^k)^T\bar{H}^k y^k}{(y^k)^T s^k}\right]\frac{s^k(s^k)^T}{(y^k)^T s^k} \end{aligned}$$

4.3 Constrained optimization

Multidimensionnal constrained optimization problems are now considered : the optimum is searched in the admissible sub-domain of the design space. Actually, most complex indutrial problems lead to multidimensionnal constrained optimization.

4.3.1 Sequential linear programming (SLP)

This method solves a sequence of linear problems approximating the posed non-linear one. α^0 being an initial guess of the solution the algorithm reads [3]

Algorithm 9 Sequential linear programming

Set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given.
while KKT-conditions not satisfied **do**
  Find $\delta\alpha^k$ which minimizes $\mathcal{J}(\alpha) \simeq \mathcal{J}(\alpha^k) + \nabla\mathcal{J}(\alpha^k).\delta\alpha$;
  subject to $\mathcal{G}_j(\alpha) \simeq \mathcal{G}_j(\alpha^k) + \nabla\mathcal{G}_j.\delta\alpha \leq 0 \quad j = 1, n_c$
  and $\alpha_{i,l} \leq \alpha + \delta\alpha \leq \alpha_{i,u} \quad i = 1, n_f$
  Update $\alpha^{k+1} = \alpha^k + \delta\alpha^k$
  Set $k = k + 1$
end while

A linear problem has to be solved at each iteration of the algorithm. Its solution may not be an admissible state for the non-linear problem (i.e. it may violate one of the constraints). It is nevertheless possible to have non-admissible states as intermediate solution of the algorithm and to converge towards the exact admissible solution. Another issue of the SLP algorithm is that the solution of the linear problem may not be bounded. In this case, the norm of $\delta\alpha$ is bounded by the user. In pratice it appears that the corresponding bound has to be lowered as the number of iterations increases. Sequential Linear Programming is well-suited for many optimization problems but its parameter - the bound of $\delta\alpha$ as a function of the number of iterations - is critical and has to be carefully defined.

[3] caution : do not mix the lower index of vector components in $\alpha = (\alpha_1, \alpha_2, ..., \alpha_{n_f})$ with the upper index of design vector number

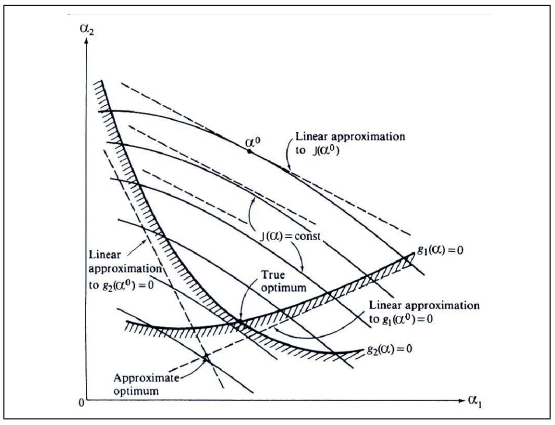

Figure 10: Sequential linear programming

4.3.2 The method of centers

As explained in previous sub-section, (SLP) algorithm generates many intermediate inadmissible states α^j (draw a sketch in the case of two convex contraints and two des ign parameters). The methods of centers, although also based on linear approximations of objective and constraints functions, defines almost only admissible intermediate states.
Let point α^0 be the initial guess for the minimization problem. α^1 is then build as the center of the largest sphere (in $\mathbb{R}^{n_f}$) included in following subspace of $\mathbb{R}^{n_f}$.

$$\begin{cases} \nabla \mathcal{J}(\alpha^0).(\alpha - \alpha^0) \leq 0. \\ \mathcal{G}_j(\alpha^0) + \nabla \mathcal{G}_j.(\alpha - \alpha^0) \leq 0 \qquad j = 1, n_c \end{cases} \tag{4}$$

The reader is invited to draw some figures to check that the sphere is not always tangent to all planes defined by linearization of the constraints in α^0. Hence, just as for (SLP), $\delta\alpha = \alpha^1 - \alpha^0$ has to be bounded. Let d also denote the first increment $d = \delta\alpha = (\alpha^1 - \alpha^0)$. The distances of center α_1 to the hyperplanes defined by differentiation of objective and constraint functions are

$$\mathcal{D} = -\frac{\nabla \mathcal{J}(\alpha^0).d}{|\nabla \mathcal{J}(\alpha^0)|}$$

$$\mathcal{D}_j = -\frac{\mathcal{G}_j(\alpha^0) + \nabla \mathcal{G}_j(\alpha^0).d}{|\nabla \mathcal{G}_j(\alpha^0)|}$$

The algorithm has to find the radius of the sphere r and its center α^1 - or equivalently -the displacement d from α^0 to α_1 solving the linear problem defined by $\mathcal{D} \geq r$, $\mathcal{D}_j \geq r$:

$$\begin{cases} \text{Maximize } r, \text{ find } d \\ \nabla \mathcal{J}(\alpha^0).d + |\nabla \mathcal{J}(\alpha^0)| r \leq 0. \\ \nabla \mathcal{G}_j(\alpha^0).d + |\nabla \mathcal{G}_j.(\alpha^0)| r \leq -\mathcal{G}_j(\alpha^0) \qquad j = 1, n_c \end{cases} \tag{5}$$

This linear optimisation problem is solved in r and d and an other iteration is done with α^{*1} as starting point.

Algorithm 10 The method of centers

Set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given.
while KKT-conditions not satisfied **do**
  Find d^k which maximizes r;
  subject to $\nabla\mathcal{J}(\alpha^k).d^k + |\nabla\mathcal{J}(\alpha^k)|r \leq 0.$
  and $\nabla\mathcal{G}_j(\alpha^k).d^k + |\nabla\mathcal{G}_j.(\alpha^k)|r \leq -\mathcal{G}_j(\alpha^k) \qquad j = 1, n_c$
  Update $\alpha^{k+1} = \alpha^k + d^k$
  Set $k = k + 1$
end while

Several authors proposed to combine this method with a gradient method defining the new position of the estimated minimum as

$$\alpha^1 = \alpha^0 + d + \beta\frac{\nabla\mathcal{J}(\alpha^0)}{|\nabla\mathcal{J}(\alpha^0)|}$$

where the factor β must be small in order to stay in the linearly admissible region (β = -r/2 for example).

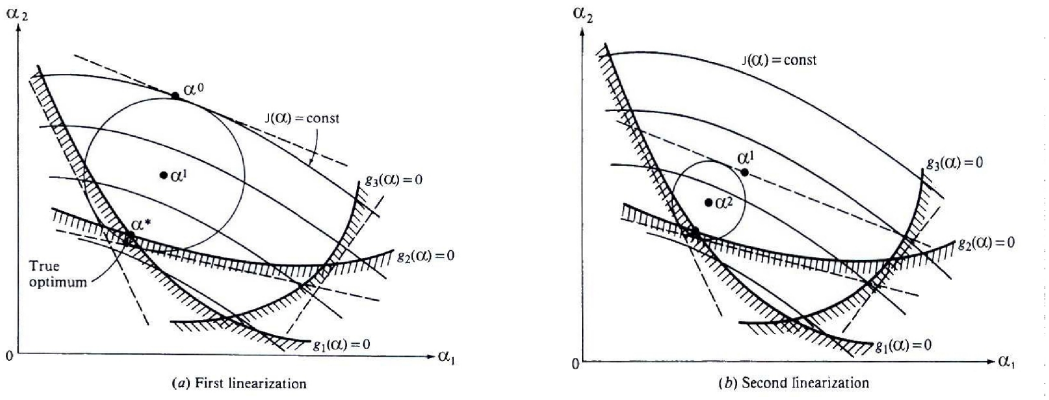

Figure 11: Method of centers

4.3.3 Feasible direction method

In opposition to the two previous methods, this one directly works on the non linear equations of the problem. Its goal is to build a sequence of points $\alpha^{(p)}$ such that

$$\alpha^{(p)} = \alpha^{(p-1)} + ld^{(p)}$$

where the displacement along direction d leads to lower values of both, the objective the active constraints [4]. After d has been defined, the factor l is determined by a monodimensional optimization. Let us now derive the definition of d. As before, the index are the ones of the first iteration of the algorithm. The vector d must

[4] all constraints satisfying $\mathcal{G}_j(\alpha^{(p-1)}) = 0$

satisfy

$$\nabla \mathcal{J}(\alpha^0).d \leq 0$$
$$\nabla \mathcal{G}_j(\alpha^0).d \leq 0 \quad \forall j / \mathcal{G}_j(\alpha^0) = 0.$$

The tricky point is the determination of the vector d ensuring the best descent. For a simple two dimensional problem ($n_f = 2$) with one active convex constraint $\mathcal{G}_1$, it is easy to check on a sketch that the minimisation $\nabla J(\alpha^0).d$ with $\nabla \mathcal{G}_1(\alpha^0).d \leq 0.$ will lead to a vector d that will lead to a nonadmissible state α_1. To tackle this issue, scalar factors θ_j are included in the problem

$$\nabla \mathcal{G}_j(\alpha^0).d + \theta_j \leq 0 \quad \theta_j > 0. \, \forall j / \mathcal{G}_j(\alpha^0) = 0.$$

One wants to link the value of θ_j with the one of $\nabla \mathcal{J}(\alpha^0).d$ Eventually, the research of he best direction for descent is reformulated as follows

$$\begin{cases} \text{Maximize } \beta, \text{find } d \\ \nabla \mathcal{J}(\alpha^0).d + \beta \leq 0. \\ \nabla \mathcal{G}_j(\alpha^0).d + \theta_j \beta \leq 0. \quad \forall j \; \mathcal{G}_j(\alpha^0) = 0. \\ d \text{ bounded} \end{cases} \tag{6}$$

Obviously, if $\mathcal{G}_j$ is a linear constraint then θ_j = 0. For non linear constraints the simplest choice is θ_j = 1. More complex formulas are presented in [3].

Algorithm 11 Feasible direction method

Set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given.
while KKT-conditions not satisfied **do**
  Find d^k which maximizes β;
  subject to $\nabla \mathcal{J}(\alpha^k).d^k + \beta \leq 0.$
  $\nabla \mathcal{G}_j(\alpha^k).d^k + \theta_j \beta \leq 0. \quad \forall j / \mathcal{G}_j(\alpha^k) = 0$ and d^k bounded.
  Compute optimal step t minimizing $q(t) = \mathcal{J}(\alpha^k + t d^k)$
  Update $\alpha^{k+1} = \alpha^k + t d^k$
  Set $k = k + 1$
end while

4.3.4 Sequential quadratic programming (SQP)

In this method, an auxiliary fonction of the descent vector d is indroduced. It is a quadratic approximation of $\mathcal{J}(\alpha + d)$. The algorithm reads

$$\begin{cases} \text{Minimize } Q(d) = \mathcal{J}(\alpha) + \nabla \mathcal{J}(\alpha).d + \frac{1}{2} d^T B d \\ \nabla \mathcal{G}_j(\alpha).d + \delta_j \mathcal{G}_j(\alpha) \leq 0. \quad j = 1, n_c \end{cases} \tag{7}$$

B is a positive definite matrix equal to the identity matrix I at the first step and to an approximation of the hessian for the next iterations. Parameter δ_j is taken equal to 1 if the contraint is strictly respected ($\mathcal{G}_j(\alpha) < 0$) and to a value in [0,1] if the current design point α violates contraint $\mathcal{G}_j(\alpha)$. See [3] for more information about hessian estimation, and monodimensional search after determination of d.

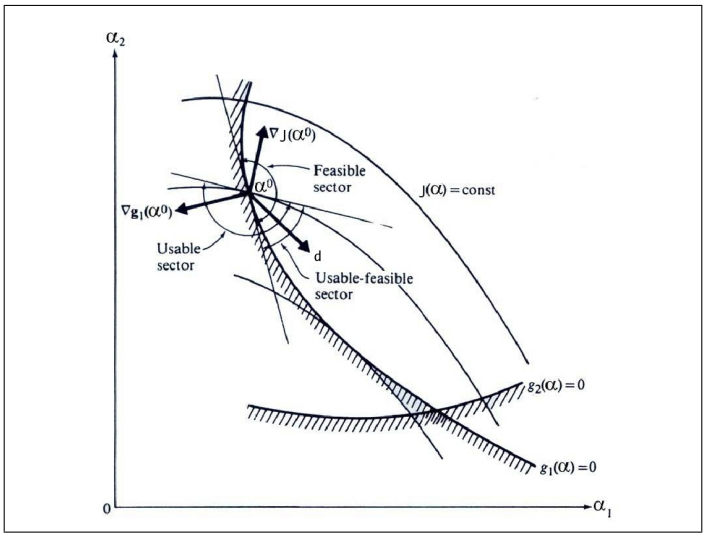

Figure 12: Method of feasible direction

Algorithm 12 Sequential quadratic programming

Set $k = 0$; an initial iterate α^0 and a stopping tolerance ϵ are given; an initial hessian matrix is also chosen;
an initial hessian matrix B^0 is also chosen;
while KKT-conditions not satisfied **do**
  Find d^k which minimizes $Q(d) = \mathcal{J}(\alpha^k) + \nabla\mathcal{J}(\alpha^k).d + \frac{1}{2}d^T B^k d$;
  subject to $\nabla\mathcal{G}_j(\alpha^k).d + \delta_j\mathcal{G}_j(\alpha^k) \leq 0. \quad j = 1, n_c$
  Update $\alpha^{k+1} = \alpha^k + d^k$
  Build B^{k+1} from, for example, BFGS formula
  Set $k = k + 1$
end while

5 Sensitivity evaluation for descent methods

Most of this section is extracted from a review article by R.P. Dwight and J. Peter [8] where more details can be found.

5.1 Introduction

This section will derive in some detail the principle methods of gradient evaluation, firstly finite differences, followed by discrete and continuous versions of the direct and adjoint methods. We use the convention of uppercase symbols for discrete, and lowercase for continuous quantities, so that W is the solution of the discretized governing equations $R(W, X) = 0$ on the mesh X. The discrete adjoint variable will be denoted

Λ. The continuous flow and adjoint solutions are w and λ respectively.

Most 3D gradient computations for fluid dynanics use (RaNS) equations with "frozen μ_t"-assumption. This asumption leads sometimes to large discrepancy between finite-difference references ans computed gradients. A warning about that is absolutely necessary in the foreword of this section

5.2 Finite Differences

The application of finite differences to an entire flow solver is by far the simplest means of obtaining solution gradients, as it requires no modification of the solver itself. As a result it was one of the first sensitivity evaluation methods to be used.

To proceed, the numerical flow solution corresponding not only to α but also to perturbed states $\alpha + \delta\alpha$ and possibly $\alpha - \delta\alpha$ is calculated. For the typical case of $\delta\alpha$ representing a geometry modification, this implies a mesh deformation $X(\alpha + \delta\alpha)$, and a new flow solution on the modified mesh satisfying

$$R\left(W(\alpha + \delta\alpha), X(\alpha + \delta\alpha)\right) = 0.$$

An approximation of objective functions derivatives in the direction $\delta\alpha$ can then be computed using a finite difference formula, such as the second-order accurate formula

$$\begin{aligned} \frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha}\delta\alpha &\simeq \frac{1}{2}\left[\mathcal{J}(\alpha + \delta\alpha) - \mathcal{J}(\alpha - \delta\alpha)\right] \\ &= \frac{1}{2}\left[J(W(\alpha + \delta\alpha), X(\alpha + \delta\alpha)) - J(W(\alpha - \delta\alpha), X(\alpha - \delta\alpha))\right]. \end{aligned} \tag{8}$$

The entire matrix $\mathrm{d}\mathcal{J}(\alpha)/\mathrm{d}\alpha$ may be evaluated at a cost of $2 \times n_f$ flow solutions, or if a first-order difference is used $n_f + 1$ flow solutions, making the method impractical for large design spaces.

Another serious disadvantage is that the choice of the step size $\|\delta\alpha\|$ is critical to the accuracy of the result. If it is too small then rounding errors become significant; particularly if the convergence level of the steady state computation is low. For example if ε is the machine epsilon (the smallest number that may be added to 1 giving a number distinct from 1), then

$$\|\delta\alpha\| \gg \varepsilon\|\alpha\|, \qquad \|\delta\alpha\| \gg \varepsilon\frac{|\mathcal{J}(\alpha)|}{\left|\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha}\right|},$$

are weak constraints on $\|\delta\alpha\|$. In practice however $\mathcal{J}$ is rarely evaluated to machine accuracy, and $\|\delta\alpha\|$ must be increased accordingly. On the other hand too large a value of $\|\delta\alpha\|$ invalidates the neglection of higher order terms in the Taylor expansion in (8). The choice is case and parameter dependent, and in practice the only means of guaranteeing accuracy is to perform a convergence study on $\|\delta\alpha\|$ for each parameter, at considerable cost.

Finite differences have been used since the 70s in the context of shape optimization. Early contributions include works of Hicks et al. [9] and Vanderplaats et al. [1] for aerofoil design, and later wing design [10], using the method of feasible directions in which the lift was held constant by moving in the design space only normal to the lift gradient. Here the flow was modeled with a small perturbation full potential equation solver. Early contribution also include articles by Reneaux and Thibert [11, 12]. As 3d optimization was considered with finite difference gradients, the number of design parameters and the cost of computations became severe problems. In an attempt to reduce the number of parameters needed *aerofunction shapes* were

introduced for example by Aidala et al. [13], defined to be aerodynamically meaningful parameterizations of an aerofoils. These might consist of geometry perturbations that control leading edge pressure, or shock position [14, 15].

However the fundamental limitations of finite differences have lead to the investigation of alternative means of gradient evaluation.

5.3 The Discrete Direct Method

Under the assumption that the discrete residual R is once continuously differentiable with respect to the flow field and mesh in a neighbourhood of $W(\alpha)$ and $X(\alpha)$, the discrete form of the governing equations $R(W, X) = 0$ may be differentiated with respect to α to give

$$\frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} = -\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}. \tag{9}$$

This may be regarded as a linear system in unknowns $\mathrm{d}W/\mathrm{d}\alpha$, where $\mathrm{d}X/\mathrm{d}\alpha$ may be evaluated by finite differences as in the previous section, and the partial derivatives could be evaluated for example by hand. The dimension of the system is the number of degrees of freedom in the non-linear equations n_W, and it can be regarded as a linearization of those equations. Of course R is seldom differentiable everywhere, but in practice this rarely causes difficulties [16].

Given the n_f solutions $\mathrm{d}W/\mathrm{d}\alpha$ the derivatives of $\mathcal{J}$ are

$$\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} \tag{10}$$

where again the partial derivatives are in principle easy to evaluate, as J is a known, explicit function of W and X. Hence the $2 \times n_f$ non-linear solutions required for second-order flow finite differences have been replaced by one non-linear and n_f linear solutions, all of dimension n_W, and the dependence on finite differences has been confined to the relatively cheap mesh update procedure.

This method was considered as early as 1982 by Bristow and Hawk for a subsonic panel method [17, 18], and again in 1989 for the transonic perturbations equations by Elbanna et al. [19]. In the early 90s it was applied to the compressible Euler equation by two teams at Old Dominion University; that of Baysal [20, 21, 22, 23] and that of Taylor and Hou [24]. Many of these articles are concerned not only with gradient evaluation, but also the linearization of the solver as a tool for investigating small perturbation to the base flow. On unstructured grids the idea was pursued by Newmann, Taylor et al. from 1995 onwards [25, 26, 27, 28].

5.4 The Discrete Adjoint Method

There are many ways to derive the discrete adjoint equations, the one given here is chosen for its similarity to the derivation of the continuous adjoint presented in the following sections. Let the direct linearization (9) be premultiplied by an arbitrary line vector Λ^T of dimension n_W, so that

$$\Lambda^T \frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \Lambda^T \left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right) = 0, \quad \forall \Lambda \in \mathbb{R}^{n_W}.$$

Adding this expression to (10)

$$\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \Lambda^T\frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \Lambda^T\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right), \quad \forall\Lambda\in\mathbb{R}^{n_W}, \tag{11}$$

and factorizing

$$\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha} = \left(\frac{\partial J}{\partial W} + \Lambda^T\frac{\partial R}{\partial W}\right)\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \Lambda^T\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right), \quad \forall\Lambda\in\mathbb{R}^{n_W},$$

isolates the term $\mathrm{d}W/\mathrm{d}\alpha$, which may be eliminated by choosing the arbitrary vector Λ to satisfy

$$\left(\frac{\partial J}{\partial W} + \Lambda^T\frac{\partial R}{\partial W}\right) = 0 \quad \text{or equivalently} \quad \left(\frac{\partial R}{\partial W}\right)^T\Lambda = -\left(\frac{\partial J}{\partial W}\right)^T, \tag{12}$$

the *adjoint equation*, a linear system in unknowns Λ the *adjoint solution*, with respect to the objective function J. Given Λ the sensitivities may be written

$$\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \Lambda^T\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right).$$

The critical point is that, because α does not appear in (12), that linear system must only be solved once for each J. Hence the full matrix $\mathrm{d}\mathcal{J}/\mathrm{d}\alpha$ may be evaluated at a cost of $n_{\mathcal{J}}$ linear system solutions, substantially independent of n_f. Note that the issues described in Section 5.3 with regard to evaluation of the partial derivatives above also apply here.

Perhaps the first application of this method was given by Shubin and Frank in 1991 for a quasi one-dimensional nozzle flow using the compressible Euler equations [29, 30, 31], and was denoted there the "implicit gradient approach" as contrast to the direct approach. Baysal et al. also recognized its potential, and offered it as an alternative to the direct approach when $n_{\mathcal{J}} \gg n_f$ [20, 21, 22, 23]. Later examples of the discrete adjoint are given in more specific contexts elsewhere in this article.

5.5 Derivation of the Continuous Adjoint Equations

In this approach the adjoint of the continuous governing equations with respect to a given objective function is derived, before being discretized. As already mentioned, the first appearance was due to Pironneau [32], with Jameson providing the first treatment for compressible flows [33], which is similar to that given here. For a more detailed introductory treatment see Abergel and Teman [34], Giles and Pierce [35], and Jameson [36]. The approach has since been reproduced by a variety of authors [37].

It is no longer easy to present the theory independently of the particular equations considered, therefore we consider the 2d Euler equations in body-fitted coordinates in two dimensions. At the end of this section, references for more general cases are presented.

It is assumed that the problem in physical space with a body-fitted structured grid, can be transformed into computational coordinates (ξ,η), see Figure 13, in such a way that the transformation of D_{xy} to $D_{\xi\eta}$ is direct, that $D_{\xi\eta}$ is a rectangular domain $[\xi_{\min},\xi_{\max}]\times[\eta_{\min},\eta_{\max}]$ and that $\xi_{\min}$ corresponds to the profile surface. Note that whereas the coordinate change operator depends on α, $D_{\xi\eta}$ itself does not.

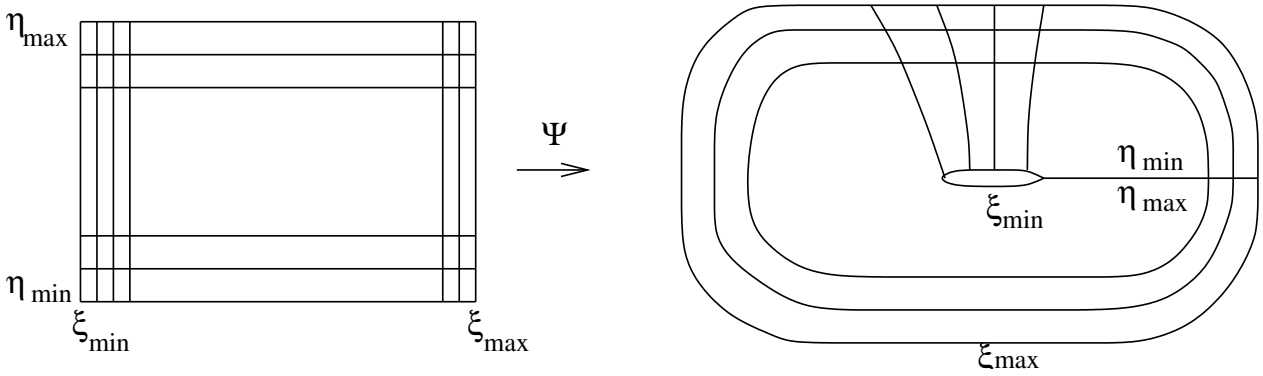

Figure 13: Body-fitted to computational coordinate domain transformation

Let K be the determinant of the Jacobian of the coordinate transformation

$$K(\xi,\eta) = \det \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{pmatrix}$$

representing the change in size of a volume element under the transformation. Then the Euler equations in the computational coordinates are

$$\frac{\partial F(W)}{\partial \xi} + \frac{\partial G(W)}{\partial \eta} = 0, \tag{13}$$

where

$$W = K \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} \quad F(W) = K \begin{pmatrix} \rho U \\ \rho U u + p\frac{\partial \xi}{\partial x} \\ \rho U v + p\frac{\partial \xi}{\partial y} \\ \rho U H \end{pmatrix} \quad G(W) = K \begin{pmatrix} \rho V \\ \rho V u + p\frac{\partial \eta}{\partial x} \\ \rho V v + p\frac{\partial \eta}{\partial y} \\ \rho V H \end{pmatrix}.$$

The slip-wall boundary condition is $U = 0$ on $\xi = \xi_{\min}$, and a farfield condition is applied to the $\xi_{\max}$ boundary. The objective function formulated in the new coordinate system is

$$\mathcal{J}(\alpha) = \int_{\xi_{\min}} J_1(w)\frac{\partial y}{\partial \eta}\,\mathrm{d}\eta + \int_{D_{\xi\eta}} J_2(w)K(\xi,\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta, \tag{14}$$

where the domain of integration is now independent of α.

The continuous adjoint equations can now be derived as follows: first write the first variation of the flow equations with respect to the design parameter α. Referring to (13) there are two distinct types of variation: the flux terms $f(w)$ and $g(w)$ vary with α, because the flow changes in the transformed coordinate space when the shape changes, and independently all metric terms also depend on α:

$$f(w) \to f(w) + \frac{\partial f}{\partial w}\frac{\mathrm{d}w}{\mathrm{d}\alpha}\delta\alpha, \quad \frac{\partial x}{\partial \eta} \to \frac{\partial x}{\partial \eta} + \frac{\partial^2 x}{\partial \eta \partial \alpha}\delta\alpha.$$

Introduce $a(w) = \mathrm{d}f(w)/\mathrm{d}w$ and $b(w) = \mathrm{d}g(w)/\mathrm{d}w$ the flux Jacobians, then the linearized equation

corresponding to (13) is

$$\begin{aligned}&\frac{\partial}{\partial\xi}\left\{\left(a(w)\frac{\partial y}{\partial\eta}-b(w)\frac{\partial x}{\partial\eta}\right)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right\}+\frac{\partial}{\partial\eta}\left\{\left(-a(w)\frac{\partial y}{\partial\xi}+b(w)\frac{\partial x}{\partial\xi}\right)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right\}\\&+\frac{\partial}{\partial\xi}\left\{f(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}-g(w)\frac{\partial^2 x}{\partial\eta\partial\alpha}\right\}+\frac{\partial}{\partial\eta}\left\{-f(w)\frac{\partial^2 y}{\partial\xi\partial\alpha}+g(w)\frac{\partial^2 x}{\partial\xi\partial\alpha}\right\}=0,\end{aligned}\tag{15}$$

and similarly for (14):

$$\begin{aligned}\frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha}=&\int_{\xi_{\min}}\left(\frac{dJ_1(w)}{dw}\frac{\mathrm{d}w}{\mathrm{d}\alpha}\frac{\partial y}{\partial\eta}+J_1(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}\right)\mathrm{d}\eta\\&+\int_{D_{\xi\eta}}\left(\frac{\mathrm{d}J_2(w)}{\mathrm{d}w}\frac{\mathrm{d}w}{\mathrm{d}\alpha}K(\xi,\eta)+J_2(w)\frac{\partial K(\xi,\eta)}{\partial\mathrm{d}\alpha}\right)\mathrm{d}\xi\,\mathrm{d}\eta.\end{aligned}\tag{16}$$

Before continuing it is convenient to introduce notation for the flux Jacobian in the computational mesh directions ξ and η,

$$a_1(w,\xi,\eta)=\left(a(w)\frac{\partial y}{\partial\eta}-b(w)\frac{\partial x}{\partial\eta}\right),\quad a_2(w,\xi,\eta)=\left(-a(w)\frac{\partial y}{\partial\xi}+b(w)\frac{\partial x}{\partial\xi}\right).\tag{17}$$

The idea behind the following procedure is to add to (16) the inner product of the linearized governing equations with an arbitrary four-component Lagrange multiplier λ, analogously to the discrete case in Section 5.4. Then we search for a condition on λ for the gradient to be independent of the $\mathrm{d}w/\mathrm{d}\alpha$ terms. In this case we assume that the flow and adjoint solutions, w and λ, are once continuously differentiable with respect to the computational coordinates, i.e. $w,\lambda\in C^1(D_{\xi\eta})^4$. Note that this is a very different restriction to that necessary in the discrete case. Proceeding from (15) we have that

$$\begin{aligned}&\int_{D_{\xi\eta}}\lambda^T\left\{\frac{\partial}{\partial\xi}\left(a_1(w,\xi,\eta)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right)+\frac{\partial}{\partial\eta}\left(a_2(w,\xi,\eta)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right)\right\}\mathrm{d}\xi\,\mathrm{d}\eta+\\&\int_{D_{\xi\eta}}\lambda^T\left\{\frac{\partial}{\partial\xi}\left(f(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}-g(w)\frac{\partial^2 x}{\partial\eta\partial\alpha}\right)+\frac{\partial}{\partial\eta}\left(-f(w)\frac{\partial^2 y}{\partial\xi\partial\alpha}+g(w)\frac{\partial^2 x}{\partial\xi\partial\alpha}\right)\right\}\mathrm{d}\xi\,\mathrm{d}\eta=0.\end{aligned}$$

Using integration by parts, and the fact that the flow sensitivity and coordinate derivatives such as $\partial^2 y/\partial\xi\partial\alpha$ are taken to be zero at the farfield we have

$$\begin{aligned}&-\int_{D_{\xi\eta}}\frac{\partial\lambda^T}{\partial\xi}a_1(w,\xi,\eta)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\,\mathrm{d}\xi\,\mathrm{d}\eta-\int_{D_{\xi\eta}}\frac{\partial\lambda^T}{\partial\eta}a_2(w,\xi,\eta)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\,\mathrm{d}\xi\,\mathrm{d}\eta-\\&\int_{D_{\xi\eta}}\frac{\partial\lambda^T}{\partial\xi}\left(f(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}-g(w)\frac{\partial^2 x}{\partial\eta\partial\alpha}\right)-\frac{\partial\lambda^T}{\partial\eta}\left(-f(w)\frac{\partial^2 y}{\partial\xi\partial\alpha}+g(w)\frac{\partial^2 x}{\partial\xi\partial\alpha}\right)\mathrm{d}\xi\,\mathrm{d}\eta+\\&\int_{\xi_{\min}}\lambda^T a_1(w,\xi,\eta)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\,\mathrm{d}\eta+\int_{\xi_{\min}}\lambda^T\left(f(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}-g(w)\frac{\partial^2 x}{\partial\eta\partial\alpha}\right)\mathrm{d}\eta=0.\end{aligned}$$

Adding this expression to (16) and extracting terms multiplying $\mathrm{d}w/\mathrm{d}\alpha$ we obtain, from the volume and

surface integrals respectively:

$$\begin{cases} \dfrac{\mathrm{d}J_2(w)}{\mathrm{d}w} K(\xi,\eta) - \dfrac{\partial \lambda^T}{\partial \xi} a_1(w,\xi,\eta) - \dfrac{\partial \lambda^T}{\partial \eta} a_2(w,\xi,\eta) = 0, & \text{on } D_{\xi,\eta}, \\ \lambda^T a_1(w,\xi,\eta) + \dfrac{\mathrm{d}J_1(w)}{\mathrm{d}w} \dfrac{\partial y}{\partial \eta} = 0, & \text{on } \xi_{\min}, \end{cases} \tag{18}$$

the *continuous adjoint* equations and boundary conditions.

One very significant feature of this problem is that not all objective functions J_1 lead to a well posed adjoint boundary condition, because the flux Jacobian a_1 is singular at a slip wall [33]. For the compressible Euler equations described here functions of pressure are admissible, which is fortunate given that integral forces on a profile are thereby allowed. On the other hand for the Navier-Stokes equations there is no clear way of accounting for wall shear-stresses, and hence viscous drag [38]. Given a solution of (18) the gradients of $\mathcal{J}$ may be written

$$\begin{aligned} \frac{\mathrm{d}\mathcal{J}(\alpha)}{\mathrm{d}\alpha} = & \int_{\xi_{\min}} J_1(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} \,\mathrm{d}\eta + \int_{\xi_{\min}} \lambda^T \left(f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) \mathrm{d}\eta \\ & - \int_{D_{\xi\eta}} \frac{\partial \lambda^T}{\partial \xi} \left(f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) \mathrm{d}\xi \,\mathrm{d}\eta \\ & - \int_{D_{\xi\eta}} \frac{\partial \lambda^T}{\partial \eta} \left(-f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) \mathrm{d}\xi \,\mathrm{d}\eta \\ & + \int_{D_{\xi\eta}} J_2(w) \frac{\partial K(\xi,\eta)}{\partial \mathrm{d}\alpha} \,\mathrm{d}\xi \,\mathrm{d}\eta. \end{aligned} \tag{19}$$

For the extension to the Navier-Stokes equations, also derived in curvilinear coordinates, see the seminary articles of Jameson, Martinelli and Pierce of 1997 [39], while a more discursive treatment is given in [40], which also includes considerations related to the use of the thus obtained gradients in shape optimization.

For finite volumes on unstructured meshes such coordinate transforms as described above are not used, and a formulation in physical coordinates is necessary. One approach was first published by Anderson and Venkatakrishnan in 1998 [38], and one year later by Hiernaux and Essers [41, 42]. Also of interest is an early adjoint formulation of the thin shear-layer equations in physical coordinates [43].

All theory presented in this section has been under the assumption of continuously differentiable flow and adjoint solutions, and therefore is only valid for flows without shocks. To alleviate this restriction the integrals involved the derivation above may be split into parts downstream and upstream of a shock, before applying the integration by parts formula and imposing the Rankine-Hugoniot condition on the boundary. The extension of the continuous adjoint to discontinuous flows is a difficult topic, and the reader is referred to works by Iollo et al. [44, 45], Cliff et al. [46], Giles et al. [47] and Gunburger [48]. However despite the theory, it should be noted that difficulties with discontinuous solutions have only been seen to occur numerically for special cases, and in practice they are not an issue.

Finally we note that the continuous direct formulation, embodied by (15), rarely occurs in the gradient evaluation literature. Pelletier et al. [49, 50] applied it to incompressible flows, and the one of few compressible references is of Sharp and Sirovitch for hypersonic profile optimization [51]. At the end, of course, the continuous equations are discretized (using most classical CFD schemes) to define are computational problem of finite dimension.

5.6 Fully Linearized or Frozen Turbulence Modeling

Though frozen eddy-viscosity is increasingly the norm, there are indeed many authors who have linearized turbulence models by hand or using AD. To our knowledge three classes of models have been linearized in the literature:

1. The algebraic model of Baldwin-Lomax was differentiated by for example Le Moigne et al. [52] and Kim et al. [53], both of whom used their method for profile optimization. Pham [54] adjointed the algebraic Michel model for evaluating sensitivities in three dimensions in turbomachinery.

2. The one-equation model of Spalart-Allmaras was linearized by under others Giles et al. [55], and the team of Anderson, Nielsen and Bonhaus at NASA Langley [56, 57, 58] who compared resulting gradients with frozen eddy-viscosity gradients [59]. The complete linearized code was applied to the 3d optimization of an isolated wing. Further examples include Nemec and Zingg [60] and Brezillon et al. [61] who both presented profile optimizations, in the latter case with multiple design points and constraints.

3. The two-equation transport models $k-\epsilon$, $k-\omega$ SST and Wilcox $k-\omega$ have been differentiated and applied to sub- and transonic profile design, as well as high-lift profile and setting optimization by Kim et al. [62, 63]. The former model has also been adjointed in the context of turbomachinery by Renac et al. [64, 65].

There is however a notable lack of linearized transport equation turbulence models applied to configurations significantly more complex than isolated wings with fully attached flow [59] or 2d high-lift profiles [63]. We suspect this is consequence, not of the difficulty of performing the linearization itself or accounting for the coupling with the mean-flow, but of the problems associated with the solution of the resulting linear system, which may be exceptionally poorly conditioned [66].

In any case in a very large number of articles the turbulence is fixed. In both the continuous and discrete cases this is achieved simply by adjointing only the mean-flow equations and treating turbulent quantities appearing therein as constants. For one-equation models these are just the eddy-viscosity, for two-equation models the turbulent kinetic energy k also appears in the internal energy and hence in the pressure. Hence in addition simplifying the solution of the system, the number of equations to be solved is reduced. Notable examples of this approach are due to Jameson et al. [67], Soemarwoto [68] and Valentin [69].

There are additionally several authors that are principally interested in AD, and the linearization of turbulence models as a secondary issue: Hou et al. [70] for Baldwin-Lomax, and Mohammadi [71] for a $k-\epsilon$ model. As a related point of interest, Bischof et al. [72] used an AD tool in forwards mode to determine the sensitivity of flow over a backward-facing step (particularly the re-attachment point) to empirical parameters of several turbulence models, the idea being that a large sensitivity to an experimentally determined parameter is a weakness of the model.

5.7 Use of sensitivities for variance analysis

In the context of industrial applications, an adequate local optimum of a function is not the only result which is sought for. In particular, the sensitivity of the objective function with respect to small variations in the design parameters is also of gret interest.
For a constrained optimization problem, most often, the first derivative $\frac{d\mathcal{J}}{d\alpha}$ is not equal to zero and its value is the requested information.
Conversely, if $\frac{d\mathcal{J}}{d\alpha} = 0$, then either a local sampling of function $\mathcal{J}$ is needed or a computation of second derivatives - not described here, see [73].

6 Applications

6.1 Discrete adjoint method at ONERA

ONERA publications relative to discrete adjoint method are :
- *Discrete adjoint in $elsA$ (part I): method/theory* [74] and *Improving accuracy robustness of discrete direct differentiation method and discrete adjoint method for aerodynamic shape optimization* [65]. These conference articles present the baseline of the discrete direct differentiation and adjoint method for aerodynamics at ONERA ;
- *Sensitivity analysis of a strongly coupled aero-structural system using discrete direct and adjoint methods* [75] presents the extension of the gradient computation methods to aeroelasticity ;
- *Aerodynamic Sensitivity Analysis of the RANS Equations via the Recursive Projection Method* [76]. This paper focuses on the efficient resolution of linear systems of the gradient methods.

6.2 Turbomachinery, helicopter and civil aircraft shape optimizations at ONERA

Many turbomachinery and civil aircraft shape optimizations have been carried out at the applied aerodynamics departement of ONERA. Even if various local and global optimization algorithms have been used, most often, local optimization was performed using conjugate gradient for unconstrained optimization and feasible direction method for constrained optimization. Discrete adjoint method (see previous section) provided the gradients needed by the algorithms for most of the recent studies.

Two turbomachinery blade optimization - out of three presented in an ASME paper by Burguburu et al. [77] - are summarised in the next two sessions. Other recent studies include :
- *Discrete adjoint in $elsA$ (part II): application to aerodynamic design optimization* [78] This paper presents the 2006 status of aerodynamic shape optimization for civil aircrafts at ONERA. It presents three applications among which the enhancement of aerodynamic performance of a supersonic commercial transport wing (multipoint optimization) and the shape optimization of the engine pylon of a large transport aircraft;
- *Single and Multi-point Aerodynamic optimizations of a supersonic transport aircraft wing using optimization strategies involving adjoint method and genetic algorithm* G. Carrier [79];
- *Efficient design optimization by physics based direct manipulation free-form deformation* Yamazaki et al. [80];
- *Aerodynamic shape optimization of Hovering Rotors using a discrete adjoint of RANS equations* A. Dumont et al. [81] (an improved version is also to be soon published in the AHS Journal) This paper gives the status of the discrete adjoint for hovering rotors at ONERA. Basic shape optimization of Rotor 7A and advanced shape optimization of rotor ERATO are also presented.

At last, some conference articles dealing with multi-disciplinary optimization are listed below :
- *Multi-disciplinary Optimization of a Supersonic Transport Aircraft Wing Planform* G. Carrier [82];
- *Aerodynamic and structural optimization of powerplant integration under the wing of a transonic transport aircraft* S. Mouton et al. [83];
- *Towards aerodynamic design by optimisation of a transonic aircraft in a multi-disciplinary context* G. Carrier et al. [84].

6.3 Optimization of a turbine stage. Quasi 3D flow. [77]

6.3.1 Mesh, parametrisation, mesh deformation

The optimization presented here concerns the shape modification of the stator of a turbine stage. The objective is to improve the efficiency by only modifying the stator, but taking into account the rotor bladings. So this optimization is carried out by computing the whole stage. The mesh, represented on Fig. 11, is made of 4 blocks per row, with a O3H topology for each of them. One can notice the large number of nodes around the stator blade. A particular attention has been paid to keep the mesh as homogeneous as possible, especially at the interface of the wheels. As illustrated on Fig. 12, the inlet absolute flow angle is close to the stator

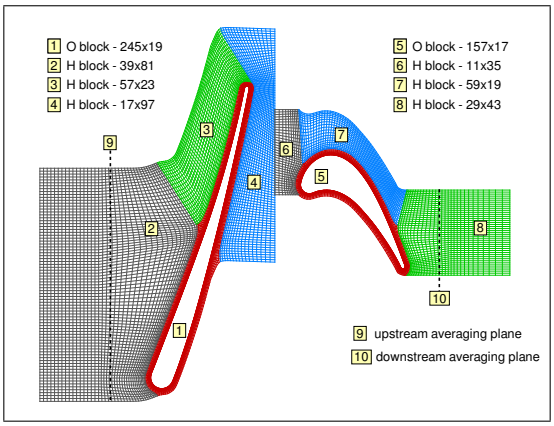

Figure 14: O3H-O3H Mesh. Test case 1.

exit absolute flow angle. The deviation is rather small ($<20^o$) compared to the standard stator turbine blade where the deviation can be greater than 70^o. The relative Mach number flowfield plotted on Fig. 12 shows a subsonic flow in the major part of the stage. Only a supersonic zone at the suction side of the stator can be observed downstream of the leading edge. In the rotor, the relative Mach number remains lower than 0.7. The camber line and the suction side of the blade are modified, one after the other, by the shape optimization

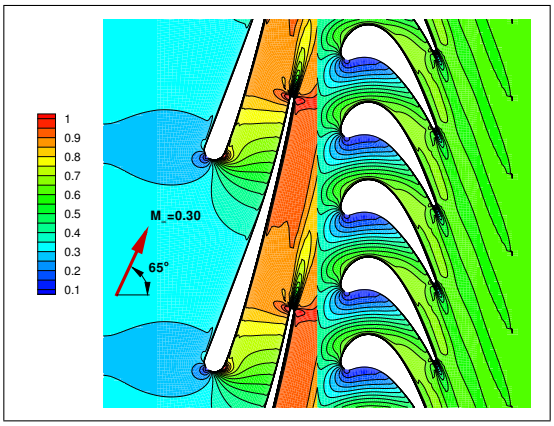

Figure 15: Relative Mach-number iso-lines. Test case 1.

process. The deformation of the suction side profile is based on degree seven Bézier function. This function of the reduced abscissa defines the displacement of the profile points along the local normal vector. For the application, two coefficients of the degree six Bézier function are frozen, so that there are only five design variables for the deformation of the suction side profile.
The modification of the camber (as a function of the abscissa along the central line of the blade) is somehow different. Once again, changes w.r.t. to the initial shape are defined, but the coefficients of the Bézier curve defining the new camber are not anymore the shape parameters. The change in camber is defined by

$$\delta_{camb}(u) = \sum_{k=0}^{k=5} \Psi_k B_k^5(u)$$

The 6 coefficients Ψ_k correspond to 6 reduced abscissa along the central line of the blade, u_k in $[0, 1]$; they are computed iteratively from Ψ_0= 0 and five design parameters α_i (i $\in$ $[1, 5]$) using recursive formula

$$\Psi_k = \Psi_{k-1} + (u_k - u_{k-1})tan(\alpha_k)$$

This means that α_k define Ψ_k steering the difference between two successive Ψ_k.

This displacement of the wall points is exactly transmitted to all points of the corresponding mesh line, orthogonal to the wall. Its is damped in the three H domains.

6.3.2 Objective and constraints

The objective function and constraints are evaluated by averaging the aerodynamic flowfield with a momentum average. The averaging planes are located upstream (2) of the stator and downstream (1) of the rotor. As the goal is to improve the performance of the stage, the objective function is taken as the opposite of the "gas power" of the stage:

$$\mathcal{J}(\alpha) = -\underline{m} C_p (T_{i_2} - T i_1)$$

Of course the mass flow must be maintained to its initial value. The (only) constraint of the problem is thus

$$\mathcal{G}_1(\alpha) = \underline{m}(\alpha) - \underline{m}(\alpha_0)$$

Also notice that the improvement of the gas power is linked to the improvement of the efficiency because the total- to-total pressure ratio remains constant during the optimization. This is a result of the constraint on the mass flow and the same downstream static pressure for all the calculations.
Eventually, different constraints are implicitly applied through the definition of the parametrization, for technological reasons and in order to improve the robustness of the optimizer. The stator axial chord remains constant when applying a deformation and the design variables are restricted to 'reasonable' values (i.e. leading to feasible bladings). For example, the camber line is not allowed to be curved by more than 4^o and the blade thickness can only vary among $\pm 50\%$ of the initial thickness.

6.3.3 Optimization using feasible direction method

A two step optimization is carried out. First, only the camber line of the blade is optimized (with zero deformation of the suction side of the blade). Then, the suction side profile of the blade is also optimised while the camber line is equal to the one defined by the first step of the optimization. Respectively, six and seven

descent iterations are performed for the two steps of the optimization.
As the adjoint method was not available at ONERA in 2003, finite difference gradients were computed. After the descent direction was found by feasible direction method (see 4.3.3) third order polynomial approximation was used (see 4.1.4) to find a step value. This led to $(5+3) \times (6+7) = 104$ analysis computations. Convergence history is presented by figure 16.

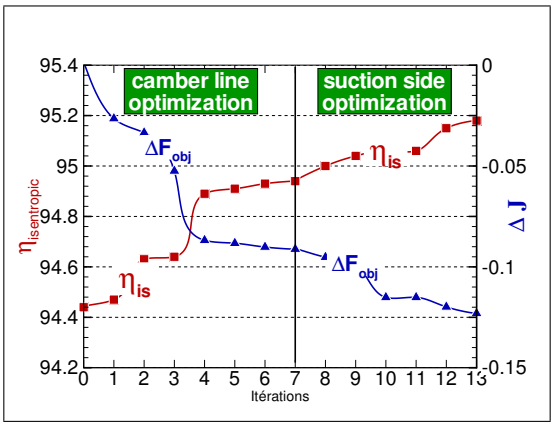

Figure 16: Convergence history. Test case 1.

6.3.4 Results

The variation of the turbine stage outputs is summarized in next array

$\Delta\mathcal{J}$	$\Delta\eta_{isen}$	Δm	ΔR_{p_i}	ΔR_{T_i}
Δ	Δ	%	%	%
-0.12	+0.82	-0.08	0.0	0.87

As indicated in the table, the variations of the overall performance show that there is no need to apply a constraint on the total-to-total pressure ratio.
The normalized density gradient contours plotted on fig. 17 allow a more precise analysis. One can not only observe a large reduction of the gradients downstream the leading edge, but also a redution of the gradient at the trailing edge, on the pressure side. Moreover, the boundary layer on the suction side is thinner on the optimized blade.

6.4 Single rotor blade. 3D flow [77]

A 3D blade of a transonic compressor is optimized. The CFD calculation is carried out on the isolated rotor with account of the tip clearance. (RANS) equations with wall-law boudary conditions are considered.

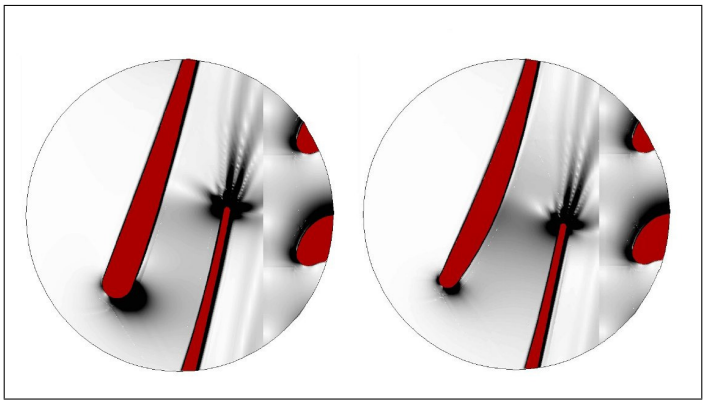

Figure 17: $\frac{||\vec{grad}\rho||}{\rho}$ **Cas test 1.**

6.4.1 Mesh, parametrization, mesh deformation

The grid is made up of 3 blocks (H-O-H topology) in the main duct (Fig. 19) and of 2 blocks (O-H topology) in the tip clearance zone, for a total of 196573 nodes. On the periodic boundaries, the nodes are not coincident, but lie on the same surface for an accurate interpolation. An adiabatic wall boundary condition is applied in the rotating frame at the hub and on the blade. A fixed wall boundary condition is applied at the casing. The grid is suited for computations using wall functions (in order to reduce the CPU time). Figure 18 presents a view of the mesh.
The operating point is in a transonic state: the relative Mach number at the tip of the blade is about 1.2 but the compressor is not choked in these conditions.

For the shape parametrization, the same principle as described with the quasi 3D approach is extended

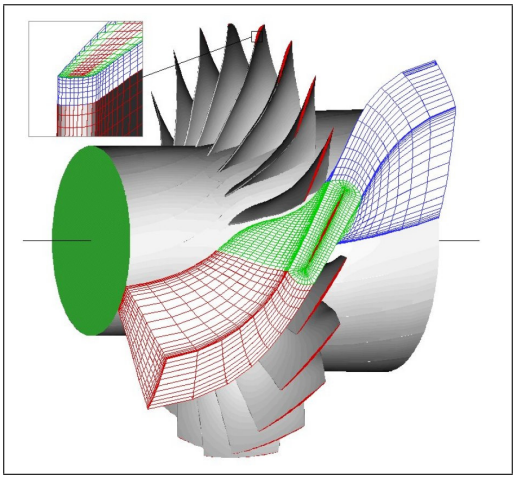

Figure 18: Configuration. Mesh. Test case 2.

to 3D. The blade surface is extracted from the grid, and a deformation zone is defined on this surface by its indices. A current point in this zone is defined by its reduced coordinate $(u, v) \in [0, 1] \times [0, 1]$ (chord

and in the span direction respectively). In the present application, the zone where geometrical modifications are applied covers the suction side, from the leading edge to the trailing edge and from hub to casing. The displacement is defined by a Bézier surface. The Bézier surface, defining the displacement along the local normal, is linked to the parameters P_{kl} by the relation:

$$\delta(u,v) = \sum_{k=0}^{k=n} \sum_{l=0}^{l=m} P_{kl} B_k^n(u) B_l^m(v)$$

(where $B_k^n(u)$, just as before, is the k^{th} Bernstein polynom of degree n). More precisely the degree of $\delta(u,v)$ is $n = 6$ for reduced coordinate u and $m = 3$ for reduced coordinate v. Next plot (figure 19) defines the fixed and free coefficients. The nine free coefficients $(P_{32}, P_{33}, P_{34}, P_{42}, P_{43}, P_{44}, P_{52}, P_{53}, P_{54})$ are the design parameters. The deformation of the solid wall is propagated and damped inside the domains, almost like for the quasi 3D case. Neverthless, specific care is taken near the tip clearange (it is important that it keeps a strictly constant heigh).

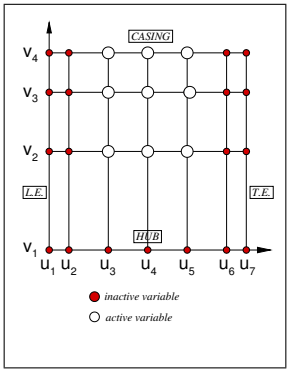

Figure 19: Active/inactive control points of the Bézier surface. Test case 2.

6.4.2 Objective, contraints functions

The objective is the isentropic efficiency of the blade

$$\mathcal{J}(\alpha) = 100 \times (1 - \eta_{isen})$$

No constraint is included.

6.4.3 Optimization

For this 3D case, only 3 iterations of optimization are performed. This optimization requires 41 calls to the CFD solver for a total calculation time of 41 000 seconds (11 hours) on NEC SX5. As the adjoint method was not available at ONERA in 2003, finite difference gradients were provided to the feasible direction method algorithm. Final values of the main parameters are presented in next array.

$\Delta\mathcal{J}$	$\Delta\eta_{isen}$	$\Delta\underline{m}$	ΔR_{p_i}	ΔR_{T_i}
Δ	Δ	%	%	%
-1.07	+1.07	+1.2	-0.1	-0.2

One can notice a low variation of the mass flow (but larger than in the quasi 3D application). The total-to-total pressure and temperature ratio remains almost constant. The deformations of the blade are plotted on Fig. 20 with an amplification factor of 10. As we can see on the optimized blade, the new shape at 80% and at 50% of the reduced span follows the same trend as observed in the quasi 3D test case: the thickness is increased in the rear part of the blade and a curvature change is applied at 30% of the chord. At 80% of the span, the blade thickness is reduced downstream of the leading edge. A mechanical constraint should be applied in this part of the blade to avoid any vibration problems. At 20% of the span, the blade thickness is increased from the leading edge to the trailing edge with a maximum located at 60% of the chord. The

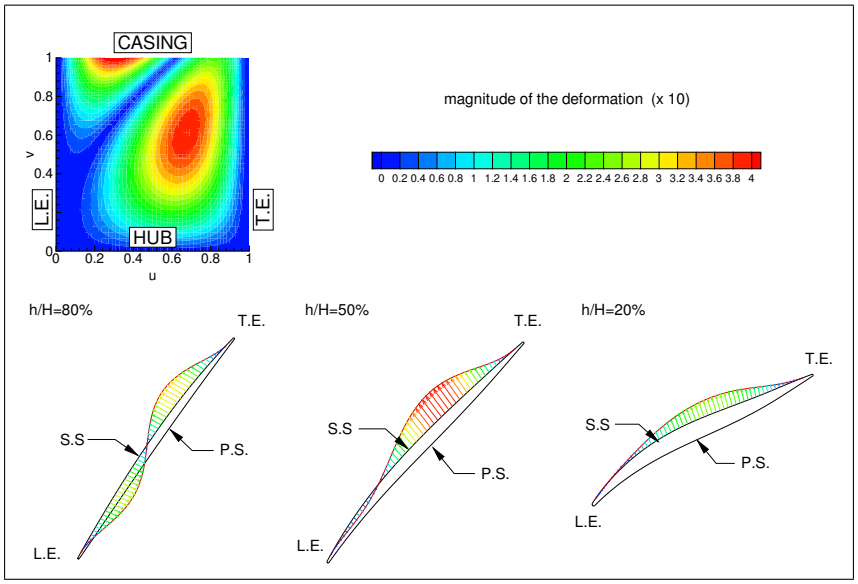

Figure 20: Final deformation (amplified). Test case 2.

radial distribution of aerodynamic characteristics at one axial chord downstream of the blade are plotted on Fig. 21. The most significant result is indicated by the radial evolution of the efficiency difference between the optimized and the reference blade: the efficiency improvement is larger than 5 points at 95% of the span, close to the tip clearance. Most of the efficiency improvement is located between 50% of the blade span and casing although the blade has changed in the lower part of the blade. The evolution of the other quantities does not present large modification as compared to the reference: the increment of mass flow from 70% of the span to casing is balanced by a 2% reduction in the lower part. The total-to-total pressure ratio evolution is not modified after the optimization except in the tip clearance region. Some differences occur on the flow angle (more than 1^o from 80% of the span to casing) but they stay low in the major part of the duct; only the tip clearance presents some variations larger than 2^o.

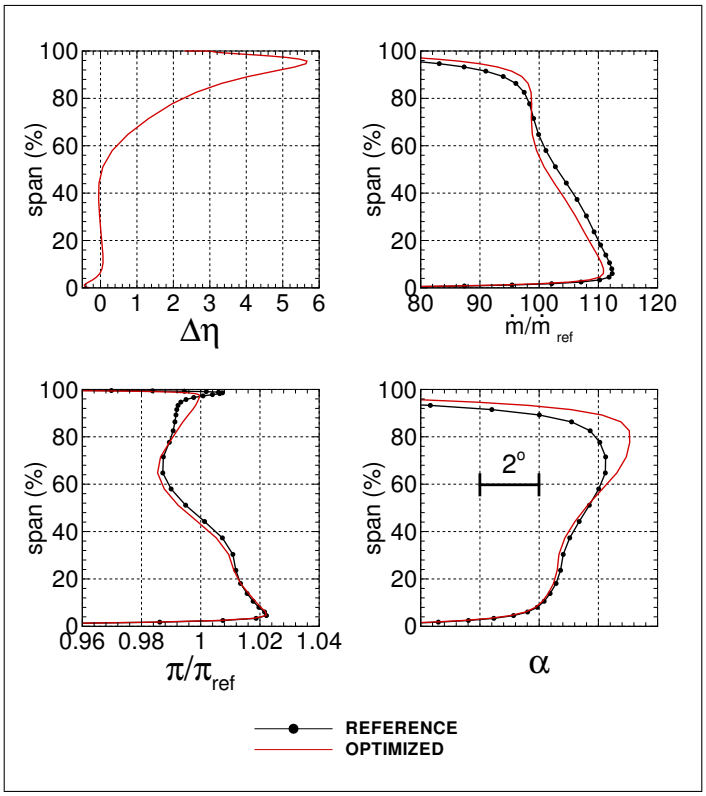

Figure 21: Spanwise change in aerodynamic outputs. Test case 2.

7 Conclusion

Local optimization for aeronautics has been a very active topic since the 70's. As it appears in other lectures of this RTO course, local and global optimization methods are nowdays often combined in an efficient search of global optimum. Nevertheless, a good knowledge of the basics of local optimization is still essential. Besides, some questions related to local optimization remain open today. For example, in aerodynamic shape design, the solution of the adjoint equation for complex flow equations and complex geometries, is not yet a routine process.

8 Acknowledgements

The author are deeply grateful to their colleagues S. Burguburu (DAAP/H2T), R.P. Dwight (TU-Delft) and M. Marcelet (ONERA/DADS) for their contributions, encouragements and sharing of ideas. The author are also very grateful to J.-A. Désidéri (INRIA) for his careful check of these lecure-notes.

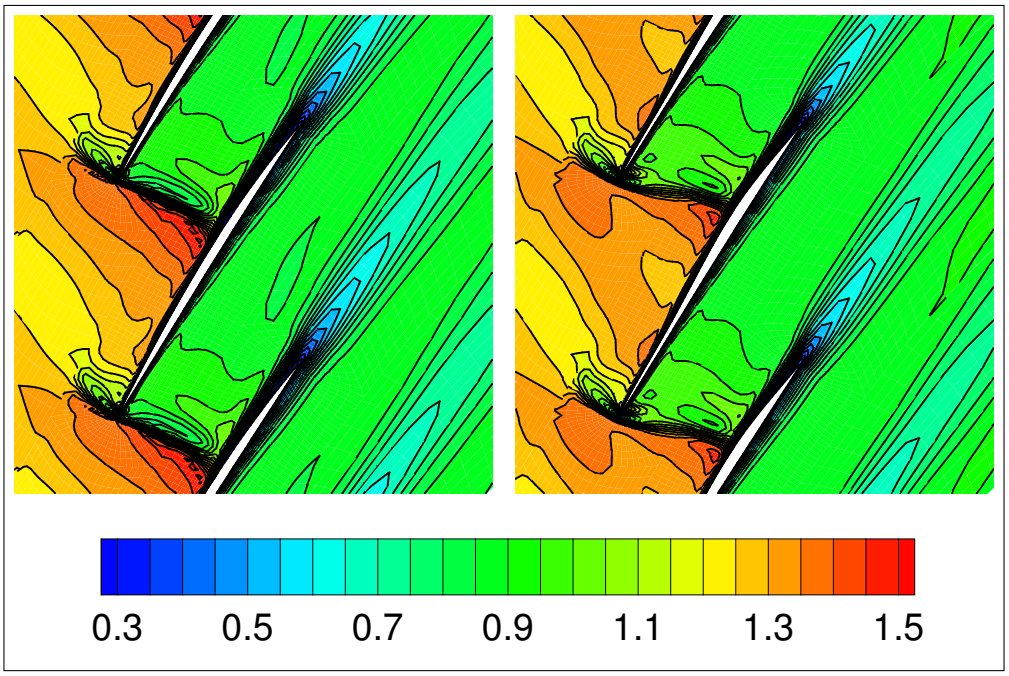

Figure 22: Iso-Mach number lines at 90% span. Test case 2.

References

[1] G. VanderPlaats, R. Hicks, Numerical airfoil optimization using a reduced number of design coordinates, Tech. Rep. TMX 73151, NASA (1976).

[2] J. Nelder, R. Mead, A simplex method for function minimization, Computer Journal 7 (4) (1965) 308–313.

[3] G. Vanderplaats, Numerical optimization techniques for engineering design: with applications, McGraw Hill, 1984.

[4] C. Broyden, The convergence of a class of double-rank minimization algorithms., Journal of the Institute of Mathematics and Its Applications. 6 (1970) 76–90.

[5] R. Fletcher, A new approach to variable metric algorithms., Computer Journal. 13 (1970) 317–322.

[6] D. Goldfarb, A family of variable metric updates derived by variational means., Mathematics of Computation. 24 (1970) 23–26.

[7] D. Shanno, Conditioning of quasi-newton methods for function minimization., Mathematics of Computation. 24 (1970) 647–656.

[8] J. Peter, R. Dwight, Numerical sensitivity analysis for aerodynamic optimization: a survey of approaches, Computers and Fluids 39 (2010) 373–391.

[9] R. Hicks, E. Murman, G. VanderPlaats, An assessment of airfoil design by numerical optimization, Tech. Rep. TMX 3092, NASA (1974).

[10] R. Hicks, P. Henne, Wing design by numerical optimization, in: AIAA Paper Series, Paper 77-1247, 1977.

[11] J. Reneaux, Méthode de définition de profils par optimisation numérique, La Recherche Aérospatiale 5 (1984) 303–321.

[12] J. Reneaux, J. Thibert, The use of numerical optimization for airfoil design, in: AIAA Paper Series, Paper 85-5026, 1985.

[13] P. Aidala, W. Davis, W. Mason, Smart aerodynamic optimization, in: AIAA Paper Series, Paper 83-1863, 1983.

[14] J. Reuther, S. Cliff, R. Hicks, C. van Dam, Practical design optimization of wing/body configurations using the Euler equations, in: AIAA Paper Series, Paper 92-2633, 1992.

[15] J. Hager, S. Eyi, K. Lee, Design efficiency evaluation for transonic airfoil optimization: a case for Navier-Stokes design, in: AIAA Paper Series, Paper 93-3112, 1993.

[16] T. Matsuzawa, M. Hafez, Optimum shape design using adjoint equations for compressible flows with shock waves, Computational Fluid Dynamics Journal 7 (3) (1998) 74–89.

[17] D. Bristow, J. Hawk, Subsonic panel method for the efficient analysis of multiple geometry perturbations, Tech. Rep. CR 3528, NASA (1982).

[18] D. Bristow, J. Hawk, Subsonic panel method for designing wing surface from pressure distribution, Tech. Rep. CR 3713, NASA (1983).

[19] H. Elbanna, L. Carlson, Determination of aerodynamic sensitivity coefficients in the transonic and supersonic regimes, in: AIAA Paper Series, Paper 89-0532, 1989.

[20] O. Baysal, M. Eleshaky, Aerodynamic design sensitivity analysis methods for the compressible Euler equations, Journal of Fluids Engineering 113 (4) (1991) 681–688.

[21] O. Baysal, M. Eleshaky, G. Burgreen, Aerodynamic shape optimization using sensitivity analysis on third-order Euler equations, in: AIAA Paper Series, Paper 91-1577, 1991.

[22] O. Baysal, M. Eleshaky, Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics, AIAA Journal 30 (1992) 718–725.

[23] O. Baysal, Flow analysis and design optimization methods for nozzle afterbody of a hypersonic vehicle, Tech. Rep. CR 4431, NASA (1992).

[24] A. Taylor III, V. Korivi, G. Hou, Sensitivity analysis applied to the Euler equations: A feasibility study with emphasis on variation of geometric shape, in: AIAA Paper Series, Paper 91-0173, 1991.

[25] J. Newman III, A. Taylor III, G. Burgreen, An unstructured grid approach to sensitivity analysis and shape optimization using the Euler equations, in: AIAA Paper Series, Paper 95-1646, 1995.

[26] J. Newman III, A. Taylor III, Three dimensional aerodynamic shape sensitivity analysis shape sensitivity analysis and design optimization using the Euler equations on unstructured grids, in: AIAA Paper Series, Paper 96-2464, 1996.

[27] A. Taylor III, J. Newman III, R. Barnwell, Aerodynamic shape sensitivity analysis and design optimization of complex configurations on unstructured grids, in: AIAA Paper Series, Paper 97-2275, 1997.

[28] J. Newman III, A. Taylor III, A. Barnwell, P. Newman, G. Hou, Overview of sensitivity analysis and shape optimization for complex aerodynamic configurations, Journal of Aircraft 36 (1) (1999) 97–116.

[29] G. Shubin, P. Frank, A comparison of implicit gradient approach and the variational approach to aerodynamic design optimization, Tech. Rep. AMS-TR-163, Boeing Computer Service, Applied Mathematics and Statistics (June 1991).

[30] G. Shubin, Obtaining cheap optimization gradients from computational aerodynamics codes, Tech. Rep. AMS-TR-164, Boeing Computer Service, Applied Mathematics and Statistics (June 1991).

[31] P. Frank, G. Shubin, A comparison of optimisation-based approaches for a model computational aerodynamics design problem, Journal of Computational Physics 98 (1992) 74–89.

[32] O. Pironneau, On optimum profiles in Stokes flow, Journal of Fluid Mechanics 59 (1) (1973) 117–128.

[33] A. Jameson, Aerodynamic design via control theory, Journal of Scientific Computing 3 (3) (1988) 233–260.

[34] F. Abergel, R. Temam, On some control problems in fluid mechanics, Theoretical and Computational Fluid Dynamics 1 (1) (1990) 303–325.

[35] M. Giles, N. Pierce, An introduction to the adjoint approach to design, in: Proceedings of ERCOFTAC Workshop on Adjoint Methods, 1999.

[36] A. Jameson, Optimum aerodynamic design via boundary control, in: AGARD-FDP-VKI Special Course, 1994.

[37] N. Gauger, J. Brezillon, Aerodynamic shape optimization using the adjoint method, Journal of the Aeronautical Society of India 54 (3).

[38] W. Anderson, V. Venkatakrishnan, Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation, Computers and Fluids 29 (1998) 443–480.

[39] A. Jameson, L. Martinelli, N. Pierce, Optimum aerodynamic design using the Navier-Stokes equations, Theoretical and Computational Fluid Dynamics 10 (1) (1998) 213–237.

[40] A. Jameson, Aerodynamic shape optimization using the adjoint method, in: VKI Course in Optimization, 2003.

[41] S. Hiernaux, J.-A. Essers, An optimal control theory based algorithm to solve 2d aerodynamic shape optimisation problems for inviscid and viscous flows, in: Proceedings of the RTO-AVT Symposium on Aerodynamic Design and Optimisation of Flight Vehicles, 1999.

[42] S. Hiernaux, J. Hessers, Aerodynamic optimization using Navier-Stokes equations and optimal control theory, in: AIAA Paper Series, Paper 99-3297, 1999.

[43] M. Giles, N. Pierce, Adjoint equations in CFD: Duality, boundary conditions and solution behaviour, in: AIAA Paper Series, Paper 97-1850, 1997.

[44] A. Iollo, M. Salas, S. Ta'asan, Shape optimization governed by the Euler equations using an adjoint method, Tech. Rep. 93-78, ICASE (1993).

[45] A. Iollo, M. Salas, Contribution to the optimal shape design of two-dimensional internal flows with embedded shocks, Journal of Computational Physics 125 (1996) 124–134.

[46] E. Cliff, M. Heikenschloss, A. Shenoy, On the optimality system for a 1d Euler flow problem, in: AIAA Paper Series, Paper 96-3993, 1996.

[47] M. Giles, N. Pierce, On the properties of solutions of the adjoint Euler equations, in: Proceedings of the 6th ICFD Conference on Numerical Method for Fluid Dynamics, Oxford, 1998.

[48] M. Gunzburger, Sensitivities, adjoint and flow optimization, International Journal for Numerical Methods in Fluids 35 (1999) 53–78.

[49] D. Pelletier, E. Turgeon, D. Lacasse, J. Boorggaard, Adaptivity, sensitivity and uncertainty: Towards standards in CFD, in: AIAA Paper Series, Paper 2001-0192, 2001.

[50] D. Pelletier, E. Turgeon, S. Etienne, J. Boorggaard, Reliable sensitivity analysis via an adaptive sensitivity equation method, in: AIAA Paper Series, Paper 2002-2578, 2002.

[51] H. Sharp, L. Sirovitch, Constructing a continuous parameter range of computational flows, AIAA Journal 27 (10) (1989) 1326–1331.

[52] A. Le Moigne, N. Qin, A discrete adjoint method for aerodynamic sensitivities for Navier-Stokes, in: Proceedings of CEAS Cambridge, 2002.

[53] C. Kim, C. Kim, O. Rho, S. Lee, Aerodynamic sensitivity analysis for the Navier-Stokes equations, in: AIAA Paper Series, Paper 99-0402, 1999.

[54] C.-T. Pham, Linéarisation du flux visqueux des equations de Navier-Stokes et de modèles de turbulence pour l'optimisation aérodynamique en turbomachines, Ph.D. thesis, L'Ecole Nationale Supérieure d'Arts et Métiers (September 2006).

[55] M. Giles, M. Duta, J. Muller, Adjoint code developments using the exact discrete approach, in: AIAA Paper Series, Paper 2001-2596, 2001.

[56] W. Anderson, D. Bonhaus, Aerodynamic design on unstructured grids for turbulent flows, Tech. Rep. TM 112867, NASA (1997).

[57] W. Anderson, D. Bonhaus, Airfoil design optimization on unstructured grids for turbulent flows, AIAA Journal 37 (2) (1999) 185–191.

[58] E. Nielsen, W. Anderson, Recent improvements in aerodynamic design optimization on unstructured meshes, AIAA Journal 40 (6) (2002) 1155–1163.

[59] E. Nielsen, W. Anderson, Aerodynamic design optimization on unstructured meshes using the Navier-Stokes equations, AIAA Journal 37 (11) (1999) 185–191.

[60] N. Nemec, D. Zingg, Towards efficient aerodynamic shape optimization based on the Navier-Stokes equations, in: AIAA Paper Series, Paper 2001-2532, 2001.

[61] J. Brezillon, R. Dwight, Discrete adjoint of the Navier-Stokes equations for aerodynamic shape optimization, in: Evolutionary and Deterministic Methods for Design, EUROGEN, Munich, 2005.

[62] C. Kim, C. Kim, O. Rho, Sensitivity analysis for the Navier-Stokes equations with two equations turbulence models, AIAA Journal 39 (5) (2001) 838–845.

[63] C. Kim, C. Kim, O. Rho, Effects of constant eddy viscosity assumption on gradient-based design optimization, in: AIAA Paper Series, Paper 2002-0262, 2002.

[64] F. Renac, C.-T. Pham, J. Peter, Sensitivity analysis for the RANS equations coupled with a linearized turbulence model, in: AIAA Paper Series, Paper 2007-3948, 2007.

[65] J. Peter, J. Mayeur, Improving accuracy and robustness of a discrete direct differentiation method and discrete adjoint method for aerodynamic shape optimization, in: Proceedings of ECCOMAS, Egmond ann Zee, 2006.

[66] R. Dwight, J. Brezillon, Adjoint algorithms for the optimization of 3d turbulent configurations, in: Proceedings of the 15th STAB Symposium, 2006.

[67] A. Jameson, N. Pierce, L. Martinelli, Optimum aerodynamic design using the Navier-Stokes equations, in: AIAA Paper Series, Paper 97-101, 1997.

[68] B. Soemarwoto, The variational method for aerodynamic optimization using the Navier-Stokes equations, Tech. Rep. 97-71, ICASE (Dec 1997).

[69] V. Valentin, Optimisation aérodynamique 3d des aubages dans les turbomachines axiales multi-etages, Ph.D. thesis, Université Paris 6. (June 2002).

[70] G. Hou, V. Maroju, A. Taylor, V. Korivi, P. Newman, Transonic turbulent airfoil design optimization with automatic differentiation in incremental iterative form, in: AIAA Paper Series, Paper 97-0101, 1995.

[71] B. Mohammadi, A new optimal shape design procedure for inviscid and viscous flows, International Journal for Numerical Methods in Fluids 25 (2) (1997) 183–203.

[72] C. Bischof, H. Buecker, A. Rasch, Sensitivity analysis of turbulence models using automatic differentiation, SIAM Journal on Scientific Computing 26 (2) (2005) 510–522.

[73] L. Sherman, A. Taylor III, L. Green, A. Newman, G. Hou, V. Korivi, First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods, Journal of Computational Physics 129 (1996) 307–331.

[74] J. Peter, Discrete adjoint method in elsa (part i): method/theory, in: Proceedings of the ONERA-DLR Aerospace Symposium (ODAS), Toulouse, 2006.

[75] M. Marcelet, J. Peter, G. Carrier, Sensitivity analysis of a strongly coupled aero-structural system using discrete direct and adjoint methods, European Journal of Computational Mechanics 17 (2008) 1077–1106.

[76] F. Renac, Aerodynamic sensitivity analysis of the rans equations via the recursive projection method., in: AIAA Paper Series, Paper 2010-4364, 2010.

[77] S. Burguburu, C. Toussaint, C. Bonhomme, G. Leroy, Numerical optimization of turbomachinery bladings, in: ASME Paper GT 2003-38310, 2003.

[78] I. Salah El Din, G. Carrier, S. Mouton, Discrete adjoint method in elsA (Part 2): Application to aerodynamic design optimisation, in: Proceedings of the 7th ONERA-DLR Aerospace Symposium (ODAS), Toulouse, 2006.

[79] G. Carrier, Single and multi-point aerodynamic optimizations of a supersonic transport aircraft wing using optimization strategies involvong adjoint method and genetic algorithm, in: Proceedings of ER-COFTAC Workshop "Design optimization: methods and applications", Las Palmas, 2006.

[80] W. Yamazaki, S. Mouton, G. Carrier, Efficient design optimization by physics-based direct manipulation free-form deformation, in: AIAA/SISMO MDO Paper Series, 2008.

[81] A. Dumont, A. Le Pape, J. Peter, S. Huberson, Aerodynamic shape optimization of hovering rotors using a discrete adjoint of the rans equations., in: Proceedings of 5th American Helicopter Society annual forum, Grapevine, Texas USA, 2009.

[82] G. Carrier, Multi-disciplinary optimization of a supersonic transport aircraft wing planform, in: Proceedings of 4th European Congress on Computational Methods, ECCOMAS 2004, Jyvaskyla, Finland, 2004.

[83] S. Mouton, J. Laurenceau, G. Carrier, Aerodynamic and structural optimization of powerplant integration under the wing of a transonic transport aircraft, in: Proceedings of 42th AAAF Symposium on Applied Aerodynamics, Nice, 2007.

[84] G. Carrier, S. Mouton, M. Marcelet, C. Blondeau, Towards aerodynamic design by optimisation of a transonic aircraft in a multi-disciplinary context, in: Proceedings of first CEAS Air and Space Conference Berlin, 2007.