EVOLUTIONARY METHODS FOR DESIGN, OPTIMIZATION AND CONTROL
P. Neittaanmäki, J. Périaux and T. Tuovinen (Eds.)

ADJOINT METHODS FOR GRADIENT– AND HESSIAN–BASED AERODYNAMIC SHAPE OPTIMIZATION

Kyriakos C. Giannakoglou* **Dimitrios I. Papadimitriou**

Parallel CFD & Optimization Unit, Lab. of Thermal Turbomachines,
National Technical University of Athens (NTUA)
P.O. Box 64069, Athens 157 10, Greece
**email: kgianna@central.ntua.gr*

Abstract. A framework for computing first and second order sensitivity derivatives of objective functions, which are in use in aerodynamic shape optimization methods, is presented. The key component of this framework is the continuous adjoint approach. According to the present formulation, the expression for the gradient of any objective function contains only boundary integrals, irrespective of whether the objective function is a field or boundary integral. This leads to increased accuracy and less computational burden, as far as gradient and Hessian computations are of concern. Since the twice application of the adjoint approach to compute the Hessian matrix is computationally intensive, the so–called direct–adjoint approach (where the gradient is computed using the direct sensitivity approach and the Hessian by subsequently employing the adjoint formulation) has been developed and used. The so–computed first and second derivatives are compared to those computed by finite differences and, as it will be demonstrated, are in excellent agreement with them. When the exact Hessian is computable, the Newton method is employed for the purpose of optimization. Newton method outperforms any other optimization method (steepest descent, conjugate gradients, quasi–Newton) at least for the number of design variables used in our demonstration examples. The parallelization of the Hessian computation is straightforward and this helps overcoming the cost of computing the Hessian matrix in problems with too many design variables.

Key words: Shape Optimization, Continuous Adjoint, Hessian Matrix, Turbomachinery Design

1 Introduction

In aerodynamic shape optimization problems, the most well known advantage of deterministic algorithms is their capability to reach the global optimum with low computational cost, as long as the starting solution is such that the descent

algorithm cannot be trapped in local optimal solutions. Deterministic methods require the gradient of the objective function with respect to the design variables and the availability of an efficient means to compute it. The adjoint method is a tool which, in contrast to finite difference schemes, computes the gradient of the objective function with the same CPU cost as that required for the solution of the flow equations.

The adjoint approach has been introduced by Pironneau[1,2] for elliptic fluid flow equations and, then, Jameson[3] extended it to hyperbolic systems. Nowadays, adjoint methods are in widespread use in several aerodynamic problems[4–8].

In the literature, adjoint formulations for the computation of the Hessian matrix of an objective function, in aerodynamic optimization problems, are extremely rare. A method to compute the Hessian matrix in structural optimization problems[9] and another one based exclusively on the adjoint approach for the shallow water equations in variational data assimilation problems in meteorology[10] can be reported. Note that we are interested in methods which compute the exact Hessian; In contrast, the literature concerning methods to compute the approximate Hessian is rich[11].

In the first part of this paper, both the direct and the corresponding adjoint approach for the gradient computation, in their so–called continuous form, are presented. The metrics–free adjoint formulation, which has been first presented for inviscid flows[12] and, then, extended to viscous flows[13], is compared to the conventional adjoint formulation in which the gradient expression depends on field integrals with variations in metrics.

In the second part, direct, adjoint and mixed continuous formulations for the computation of the exact Hessian matrix design problems in aerodynamics are presented. Starting from either the direct or the adjoint approach for the computation of the functional gradient, either the direct or the adjoint approach can be used to compute the Hessian matrix. Thus, four different approaches to compute the exact Hessian matrix, with different computational cost, have been devised. The more efficient approach proves to be the so–called *direct–adjoint* approach. In the results section, the exact Hessian matrix is used to support the Newton method for the inverse design of a turbomachinery cascade with less CPU cost than the BFGS[11] algorithm, which is based on the exact gradient and an approximate (thus, computationally inexpensive) Hessian.

2 Continuous Adjoint Approach for the Gradient Computation

2.1 Flow Equations and Discretisation

The flow equations of a compressible viscous fluid are used as state equations. These are expressed in the usual vector form, as

$$\vec{R}(\vec{U}) = \frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{f}_i^{inv}}{\partial x_i} - \frac{\partial \vec{f}_i^{vis}}{\partial x_i} = \vec{0} \tag{1}$$

where $\vec{U}$ is the vector of conservative variables $\vec{U} = \left[\rho \ , \ \rho\vec{v}^T \ , \ E\right]^T$ and $\vec{f}_i^{inv} = \left[\rho u_i \ , \ \rho u_i \vec{v}^T + p\vec{\delta}_i^T \ , \ u_i(E+p)\right]^T$, $\vec{f}_i^{vis} = \left[0 \ , \ \vec{\tau}_i^T \ , \ u_j\tau_{ij} + q_i\right]^T$ are the inviscid and viscous fluxes, respectively. According to the standard notation, u_i, $E{=}\rho e + \frac{1}{2}\rho u_i^2$,

$\vec{\tau}_i$, and $q_i = k\frac{\partial T}{\partial x_i}$ stand for the velocity components, total energy, vector of viscous stresses and heat fluxes, respectively. Also, $\vec{\delta}_i$ is the Kronecker symbol.

The discretization of eq. 1 on a structured or unstructured grid is based on the finite volume method, using vertex–centered control volumes. The inviscid fluxes which cross the boundaries of the control volume are computed by means of the Roe's upwind scheme[14], so the numerical inviscid flux associated with the edge formed by nodes P and Q is given by

$$\vec{h}_{PQ} = \frac{1}{2}\left(A_L\vec{U}_L + A_R\vec{U}_R\right) - \frac{1}{2}|A_{PQ}|\left(\vec{U}_L - \vec{U}_R\right) \tag{2}$$

where $A_i = \frac{\partial \vec{f}_i^{inv}}{\partial \vec{U}}$ and $A = A_i n_i$, n_i being the unit vector which is normal to each segment forming the finite volume boundary. Second order accuracy is obtained using appropriate Taylor expansions for $\vec{U}_L$ and $\vec{U}_R$, based on $\vec{U}_P$, $\vec{U}_Q$ and the local gradients of $\vec{U}$, fig. (1).

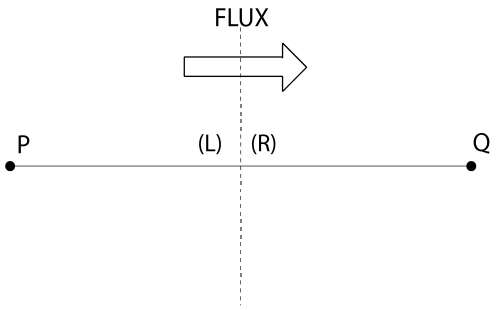

Figure 1: Grid edge PQ and the associated finite volume boundary (dotted line). In any 1D Riemann problem the left (L) and right (R) states, used to define the inviscid flux vector, are shown.

The computation of viscous fluxes is straightforward, by considering a linear distribution of $\vec{U}$ over any grid cell.

2.2 Inverse Design in Inviscid Flows – The Adjoint Approach

Aiming at the design of an aerodynamic shape (airfoil, duct, etc, in order to restrict ourselves to 2D shapes only; note that the extension to 3D is straightforward and has already been presented by the authors[13,15]) the objective function quantifies the deviation of pressure distribution $p(S)$ from a known target distribution $p_{tar}(S)$ along the solid walls S_w. Thus

$$F = \frac{1}{2}\int_{S_w}(p - p_{tar})^2 dS \tag{3}$$

The variation in F, due to any variation in the design variables $\vec{b}$, is written as follows

$$\delta F = \frac{1}{2}\int_{S_w}(p - p_{tar})^2\delta(dS) + \int_{S_w}(p - p_{tar})\delta p dS \tag{4}$$

To set up the adjoint formulation, the variation in the so–called augmented function $F_{aug} = F + \int_\Omega \vec{\Psi}^T\vec{R}(\vec{U})d\Omega$ is defined by integrating the product of the adjoint variables $\vec{\Psi}$ and the variation in the flow equations (the inviscid part of eq. 1) over the flow domain Ω and adding it to δF. Thus

$$\delta F_{aug} = \delta F + \int_\Omega \vec{\Psi}^T\delta\left(\frac{\partial \vec{f}_i^{inv}}{\partial x_i}\right)d\Omega \tag{5}$$

The following equation

$$\delta\left(\frac{\partial \vec{f}_i^{inv}}{\partial x_i}\right) = \frac{\partial(\delta \vec{f}_i^{inv})}{\partial x_i} - \frac{\partial \vec{f}_i^{inv}}{\partial x_k}\frac{\partial(\delta x_k)}{\partial x_i} \tag{6}$$

can be proved[13]. It expresses the variation in the gradient of any flow quantity in terms of the gradient of the variation of the same quantity and an additional term which includes the gradient of variations in nodal coordinates. Through integration by parts and the Gauss divergence theorem, the variation in F is expressed as follows

$$\delta F = \frac{1}{2}\int_{S_w}(p-p_{tar})^2\delta(dS) - \int_{S_w}\frac{\partial \vec{U}}{\partial x_k}^T {A_n}^T \vec{\Psi}\delta x_k dS + \int_{S_w}(\Psi_{i+1}p - \vec{\Psi}^T\vec{f_i})\delta(n_i dS) \quad (7)$$

Eq. 7 is expressed in terms of the adjoint variables which are computed by discretizing and solving the field adjoint equations[13]

$$\frac{\partial\vec{\Psi}}{\partial t} - {A_i}^T\frac{\partial\vec{\Psi}}{\partial x_i} = \vec{0} \quad (8)$$

satisfying

$$(p - p_{tar}) + \Psi_{i+1}n_i = 0 \quad (9)$$

over the solid walls S_w and

$$\delta\vec{U}^T(A_n^T\vec{\Psi}) = 0 \quad (10)$$

at the inlet and outlet. After satisfying eqs. 8 to 10, the gradient expression, eq. 7, includes only boundary integrals in terms of geometrical variations. Variations such as $\delta(dS)$, δx_k or $\delta(n_i dS)$ can be computed analytically with negligible computational cost, provided that the shape parameterization has been defined.

As in the flow equations, a Roe–like scheme is used for the discretisation of the adjoint equations. According to this scheme, the adjoint fluxes are given by

$$\begin{aligned}\vec{t}_{PQ}(P) &= \frac{1}{2}A_P^T(\Psi_L + \Psi_R) + \frac{1}{2}|A_{PQ}|(\Psi_L - \Psi_R)\\ \vec{t}_{PQ}(Q) &= \frac{1}{2}A_Q^T(\Psi_L + \Psi_R) + \frac{1}{2}|A_{PQ}|(\Psi_L - \Psi_R)\end{aligned} \quad (11)$$

2.3 Comments on the Adjoint Approach

(a) On a structured grid, an equation of the form[12]

$$\delta\left(\frac{\partial\vec{f}_i^{inv}}{\partial x_i}\right) = \frac{\partial(\delta\vec{f}_i^{inv})}{\partial x_i} + \frac{\partial\vec{f}_i^{inv}}{\partial\xi^j}\delta\left(\frac{\partial\xi^j}{\partial x_i}\right) \quad (12)$$

can be used instead of eq. 6. Note that $\frac{\partial\xi^j}{\partial x_i}$ are the metrics associated with the standard transformation of the grid from the physical to the computational or transformed domain. Eqs. 6 and 12 are equivalent but the former is much more general since it is not tailored to structured grids. Using eq. 12, it can be proved that the variation in F can alternatively be expressed as follows

$$\delta F = \frac{1}{2}\int_{S_w}(p-p_{tar})^2\delta(dS) - \int_{\Omega}\vec{\Psi}^T\frac{\partial f_i}{\partial\xi^j}\delta\left(\frac{\partial\xi^j}{\partial x_i}\right)d\Omega + \int_{S_w}(\Psi_{i+1}p - \vec{\Psi}^T\vec{f_i})\delta(n_i dS) \quad (13)$$

which, in contrast to eq. 7, contains a field integral in terms of the variation in metrics. The standard way to compute this integral and the $\delta\left(\frac{\partial\xi^j}{\partial x_i}\right)$ terms, in particular,

is through finite differences which require successive remeshing tasks. This procedure affects badly both the computational accuracy and CPU cost per optimization cycle.

(b) Eq. 7 can also be derived by directly expressing the variation in F_{aug} using the (zero) partial (instead of total) variation in the residual of the flow equations. In this case, we define

$$\delta F_{aug} = \delta F + \int_{\Omega} \vec{\Psi}^T \frac{\partial}{\partial b_j} \left(\frac{\partial \vec{f}_i^{inv}}{\partial x_i} \right) d\Omega \tag{14}$$

The interchange of partial derivatives ($\frac{\partial}{\partial b_j}$ and $\frac{\partial}{\partial x_i}$) is now possible and by integrating by parts and using

$$\frac{\delta \Phi}{\delta b_i} = \frac{\partial \Phi}{\partial b_i} + \frac{\partial \Phi}{\partial x_l} \frac{\delta x_l}{\delta b_i} \tag{15}$$

(written for any flow quantity Φ) it leads to eq. 7.

3 Continuous Direct Approach for the Gradient Computation

The continuous adjoint approach computes the objective function gradient at the extra cost of numerically solving the linear adjoint pde's, eqs. 8. Thus, the CPU cost per optimization cycle (based on steepest descent or BFGS, etc) is approximately equal to that of two "equivalent" flow solutions, irrespective of the number N of design variables. Thus, at first glance, the presentation of the so–called direct sensitivity approach to the computation of gradients might be useless. It is evident that the CPU cost of the direct approach is high since the computation of the gradient of an objective function with respect to N design variables requires N equivalent flow solutions ($N+1$ solutions per cycle).

However, the direct sensitivity approach is presented herein for two reasons. The first reason is that this is an indispensable part of the so–called direct–adjoint approach for the computation of the exact Hessian matrix. The second (minor) reason is that, with the mathematical background presented in section 2, the direct sensitivity approach becomes more accurate and cheaper since we may get rid of any field integral of variations in grid metrics.

The continuous direct approach is based on the computation of variations in flow quantities (such as δp which appears in eq. 4 or $\delta \vec{U}$ in general) from the linearized form of eq. 1. We first express the partial derivative of the (steady) flow equations as follows

$$\frac{\partial}{\partial b_i} \left(\frac{\partial \vec{f}_k}{\partial x_k} \right) = \vec{0} \tag{16}$$

which transforms to

$$\frac{\partial}{\partial x_k} \left(A_k \frac{\partial \vec{U}}{\partial b_i} \right) = \vec{0} \tag{17}$$

Note that it is much more convenient to start from eq. 16 than from the total variation of $\frac{\partial \vec{f}_k}{\partial x_k}$ (i.e. using eq. 15 for $\Phi = \frac{\partial \vec{f}_k}{\partial x_k}$) which requires repetitive calls to the grid generation software. The discretisation of eq. 17 is based on a Roe–like upwind

scheme, for the conservation of fluxes expressed as (fig. 1)

$$\vec{g}_{i,PQ} = \frac{1}{2}\left(A_P \frac{\partial \vec{U}}{\partial b_i}|_L + A_Q \frac{\partial \vec{U}}{\partial b_i}|_R\right) - \frac{1}{2}|A_{PQ}|\left(\frac{\partial \vec{U}}{\partial b_i}|_L - \frac{\partial \vec{U}}{\partial b_i}|_R\right) \tag{18}$$

Along the solid walls, the variation in the no–penetration condition $u_i n_i$=0 yields the boundary flux

$$\vec{g}_i|_w = \vec{N}\frac{\partial p}{\partial b_i}|_w - \vec{N}_U \rho\left(u_k \frac{\delta n_k}{\delta b_i} + \frac{\partial u_k}{\partial x_l}\frac{\delta x_l}{\delta b_i} n_k\right)|_w \tag{19}$$

where $\vec{N} = [0, n_1, n_2, 0]^T$ and $\vec{N}_U = [1, u_1, u_2, \frac{E+p}{\rho}]^T$. Homogeneous conditions are employed at the inlet/outlet of the flow domain, based on $\frac{\partial p_t}{\partial b_i}|_I$=0 , $\frac{\partial T_t}{\partial b_i}|_I$=0 , $\frac{\partial \alpha}{\partial b_i}|_I$=0 (inlet, I) and $\frac{\partial p}{\partial b_i}|_O$=0 (outlet, O).

4 Continuous Approaches for the Hessian Computation

The Hessian matrix of F with respect to the design variables can be computed starting from either the adjoint–based variational expression, eq. 7, or the direct one, eq. 4, which are rewritten here as follows

$$\begin{aligned}\frac{\delta F}{\delta b_i} &= \frac{1}{2}\int_{S_w}(p-p_{tar})^2 \frac{\delta(dS)}{\delta b_i} - \int_{S_w}\left({A_n}^T \vec{\Psi}\right)^T \frac{\partial \vec{U}}{\partial x_k}\frac{\delta x_k}{\delta b_i} dS \\ &+ \int_{S_w}(\Psi_{k+1} p - \vec{\Psi}^T \vec{f}_k)\frac{\delta(n_k dS)}{\delta b_i}\end{aligned} \tag{20}$$

and

$$\frac{\delta F}{\delta b_i} = \int_{S_w}(p-p_{tar})\frac{\delta p}{\delta b_i} dS + \frac{1}{2}\int_{S_w}(p-p_{tar})^2 \frac{\delta(dS)}{\delta b_i} \tag{21}$$

Starting from any of the above expressions for the first derivatives, we may then employ either the direct or the adjoint approach to compute second derivatives. This leads to four different combinations: *adjoint–direct*, *adjoint–adjoint*, *direct–direct* and *direct–adjoint*, three of which are presented below. The analysis of the *direct–direct* approach is omitted due to its high computational cost, which is equal to $\frac{1}{2}N(N+1)+N+1$ equivalent flow solutions per optimization cycle.

4.1 The *Adjoint–Direct* Approach

The variation in eq. 20 with respect to b_j leads to the following expression for the second derivatives of F

$$\begin{aligned}\frac{\delta^2 F}{\delta b_i \delta b_j} &= \frac{1}{2}\int_{S_w}(p-p_{tar})^2 \frac{\delta^2(dS)}{\delta b_i \delta b_j} + \int_{S_w}(p-p_{tar})\frac{\delta p}{\delta b_j}\frac{\delta(dS)}{\delta b_i} - \int_{S_w}\left({A_n}^T\vec{\Psi}\right)^T \frac{\partial \vec{U}}{\partial x_k}\frac{\delta^2 x_k}{\delta b_i \delta b_j} dS \\ &- \int_{S_w}\left({A_n}^T \frac{\delta \vec{\Psi}}{\delta b_j}\right)^T \frac{\partial \vec{U}}{\partial x_k}\frac{\delta x_k}{\delta b_i} dS - \int_{S_w}\left({A_n}^T\vec{\Psi}\right)^T \frac{\partial \vec{U}}{\partial x_k}\frac{\delta x_k}{\delta b_i}\frac{\delta(dS)}{\delta b_j} \\ &- \int_{S_w}\left({A_n}^T\vec{\Psi}\right)^T \frac{\delta}{\delta b_j}\left(\frac{\partial \vec{U}}{\partial x_k}\right)\frac{\delta x_k}{\delta b_i} dS + \int_{S_w}\left(\Psi_{k+1} p - \vec{\Psi}^T \vec{f}_k\right)\frac{\delta^2(n_k dS)}{\delta b_i \delta b_j} \\ &+ \int_{S_w}\left(\frac{\delta \Psi_{k+1}}{\delta b_j} p - \frac{\delta \vec{\Psi}^T}{\delta b_j}\vec{f}_k\right)\frac{\delta(n_k dS)}{\delta b_i} + \int_{S_w}\left(\Psi_{k+1}\frac{\delta p}{\delta b_j} - \vec{\Psi}^T \frac{\delta \vec{f}_k}{\delta b_j}\right)\frac{\delta(n_k dS)}{\delta b_i}\end{aligned} \tag{22}$$

According to eq. 22, to compute the values of the (symmetric) Hessian matrix of F at any iterate during the optimization loop, apart from $\vec{U}$ and $\vec{\Psi}$, their sensitivities $\frac{\delta\vec{U}}{\delta b_i}$ and $\frac{\delta\vec{\Psi}}{\delta b_i}$ are also necessary. $\frac{\delta\vec{U}}{\delta b_i}$ can be computed by numerically solving eqs. 17 whereas the computation of $\frac{\delta\vec{\Psi}}{\delta b_i}$ requires a similar formulation based on the adjoint equations and boundary conditions (not presented here). Each of them requires the numerical solution of N pde's. Thus, $2N$+2 equivalent flow solutions per optimization cycle are due.

4.2 The *Adjoint–Adjoint* Approach

It is straightforward to use a twice augmented objective function $\widehat{F}_{aug}$, based on different sets of Lagrange multipliers for the flow ($\vec{\Lambda}_j, j$=$1, N$) and adjoint ($\vec{M}_j, j$=$1, N$) equations. The second order variation in $\widehat{F}_{aug}$ is expressed as

$$\frac{\delta^2\widehat{F}_{aug}}{\delta b_i \delta b_j} = \frac{\delta^2 F}{\delta b_i \delta b_j} + \int_\Omega \vec{\Lambda}_j^T \frac{\partial}{\partial b_j}\left(\frac{\partial \vec{f}_k}{\partial x_k}\right) d\Omega + \int_\Omega \vec{M}_j^T \frac{\partial}{\partial b_j}\left(A_k^T \frac{\partial \vec{\Psi}}{\partial x_k}\right) d\Omega \tag{23}$$

Eq. 23 is integrated by parts to give

$$\begin{aligned}\frac{\delta^2\widehat{F}_{aug}}{\delta b_i \delta b_j} &= \frac{\delta^2 F}{\delta b_i \delta b_j} - \int_\Omega \left(A_k^T \frac{\partial \vec{\Lambda}_j}{\partial x_k}\right)^T \frac{\partial \vec{U}}{\partial b_j} d\Omega + \int_S \left({A_n}^T \vec{\Lambda}_j\right)^T \frac{\partial \vec{U}}{\partial b_j} dS \\ &- \int_\Omega \frac{\partial \left(A_k \vec{M}_j\right)}{\partial x_k}^T \frac{\partial \vec{\Psi}}{\partial b_j} d\Omega + \int_S \left(A_n \vec{M}_j\right)^T \frac{\partial \vec{\Psi}}{\partial b_j} dS \end{aligned} \tag{24}$$

where $\frac{\delta^2 F}{\delta b_i \delta b_j}$ is given by eq. 22. In eq. 24, the elimination of the field and boundary terms which depend on $\frac{\partial \vec{U}}{\delta b_j}$ and $\frac{\partial \vec{\Psi}}{\delta b_j}$ gives rise to the two sets of adjoint equations for Λ_j and M_j (2N equations). Thus, the total CPU cost per optimization cycle is still equal to $2N$+2 equivalent flow solutions.

4.3 The *Direct–Adjoint* Approach

The derivative of eq. 21 with respect to b_j gives

$$\begin{aligned}\frac{\delta^2 F}{\delta b_i \delta b_j} &= \int_{S_w} \frac{\delta p}{\delta b_i}\frac{\delta p}{\delta b_j} dS + \int_{S_w} (p - p_{tar}) \frac{\delta^2 p}{\delta b_i \delta b_j} dS + \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_i}\frac{\delta (dS)}{\delta b_j} \\ &+ \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_j}\frac{\delta (dS)}{\delta b_i} + \frac{1}{2}\int_{S_w} (p - p_{tar})^2 \frac{\delta^2 (dS)}{\delta b_i \delta b_j} \end{aligned} \tag{25}$$

Starting from eq. 25, the second order variation in F_{aug} is defined as

$$\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} = \frac{\delta^2 F}{\delta b_i \delta b_j} + \int_\Omega \vec{\Psi}^T \frac{\partial}{\partial x_k}\left(\frac{\partial^2 \vec{f}_k}{\partial b_i \partial b_j}\right) d\Omega \tag{26}$$

where the partial derivatives in the right–hand side integrand have been interchanged. The integration of eq. 26 by parts gives

$$\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} = \frac{\delta^2 F}{\delta b_i \delta b_j} - \int_\Omega \left(A_k^T \frac{\partial \vec{\Psi}}{\partial x_k}\right)^T \frac{\partial^2 \vec{U}}{\partial b_i \partial b_j} d\Omega + \int_S \vec{\Psi}^T \frac{\partial^2 \vec{f}_k}{\partial b_i \partial b_j} n_k dS \tag{27}$$

By twice employing eq. 15, for Φ=$\vec{f}_k$, we obtain

$$\frac{\delta^2 \vec{f}_k}{\delta b_i \delta b_j} = \frac{\partial^2 \vec{f}_k}{\partial b_i \partial b_j} + \frac{\partial^2 \vec{f}_k}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 \vec{f}_k}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} + \frac{\partial^2 \vec{f}_k}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} + \frac{\partial \vec{f}_k}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j} \quad (28)$$

Eq. 28 may be solved for the partial derivatives as follows

$$\begin{aligned}\frac{\partial^2 \vec{f}_k}{\partial b_i \partial b_j} n_k &= \frac{\delta^2 \vec{f}_k}{\delta b_i \delta b_j} n_k - \frac{\partial^2 \vec{f}_k}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} n_k \\ &- \frac{\partial^2 \vec{f}_k}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} n_k - \frac{\partial^2 \vec{f}_k}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} n_k - \frac{\partial \vec{f}_k}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j} n_k \end{aligned} \quad (29)$$

The first term on the right–hand side of eq. 29 is finally written as

$$\frac{\delta^2 \vec{f}_k}{\delta b_i \delta b_j} n_k = \frac{\delta^2 \left(\vec{f}_k n_k\right)}{\delta b_i \delta b_j} - \frac{\delta \vec{f}_k}{\delta b_i}\frac{\delta n_k}{\delta b_j} - \frac{\delta \vec{f}_k}{\delta b_j}\frac{\delta n_k}{\delta b_i} - \vec{f}_k \frac{\delta^2 n_k}{\delta b_i \delta b_j} \quad (30)$$

Due to the no–penetration condition, the flux vector across a solid wall segment is given by $\vec{f}_k n_k$=$p[0, n_1, n_2, 0]^T$, so

$$\frac{\delta^2 \left(\vec{f}_k n_k\right)}{\delta b_i \delta b_j} = \vec{N}\frac{\delta^2 p}{\delta b_i \delta b_j} + \frac{\delta^2 \vec{N}}{\delta b_i \delta b_j} p + \frac{\delta \vec{N}}{\delta b_i}\frac{\delta p}{\delta b_j} + \frac{\delta \vec{N}}{\delta b_j}\frac{\delta p}{\delta b_i} \quad (31)$$

Substituting eq. 25 and eqs. 28 to 31 into eq. 27 and after a series of mathematical rearrangements we obtain

$$\begin{aligned}\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} &= \int_{S_w} \frac{\delta p}{\delta b_i}\frac{\delta p}{\delta b_j} dS + \underbrace{\int_{S_w} (p - p_{tar}) \frac{\delta^2 p}{\delta b_i \delta b_j} dS}_{SWCR} + \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_i}\frac{\delta (dS)}{\delta b_j} \\ &+ \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_j}\frac{\delta (dS)}{\delta b_i} + \frac{1}{2}\int_{S_w} (p - p_{tar})^2 \frac{\delta^2 (dS)}{\delta b_i \delta b_j} - \underbrace{\int_\Omega \left(A_k^T \frac{\partial \vec{\Psi}}{\partial x_k}\right)^T \frac{\partial^2 \vec{U}}{\delta b_i \delta b_j} d\Omega}_{FAE} \\ &+ \underbrace{\int_{S_{I,O}} \vec{\Psi}^T \frac{\delta^2 (A_n \vec{U})}{\delta b_i \delta b_j} dS}_{IOBC} + \underbrace{\int_{S_w} \vec{\Psi}^T \vec{N} \frac{\delta^2 p}{\delta b_i \delta b_j} dS}_{SWCR} + \int_{S_w} (\Psi_{k+1} p - \vec{\Psi}^T \vec{f}_k) \frac{\delta^2 n_k}{\delta b_i \delta b_j} dS \\ &+ \int_{S_w} \left(\Psi_{k+1}\frac{\delta p}{\delta b_i} - \vec{\Psi}^T \frac{\delta \vec{f}_k}{\delta b_i}\right) \frac{\delta n_k}{\delta b_j} dS + \int_{S_w} \left(\Psi_{k+1}\frac{\delta p}{\delta b_j} - \vec{\Psi}^T \frac{\delta \vec{f}_k}{\delta b_j}\right) \frac{\delta n_k}{\delta b_i} dS \\ &- \int_{S_w} \vec{\Psi}^T A_k n_k \left(\frac{\partial^2 \vec{U}}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 \vec{U}}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} + \frac{\partial^2 \vec{U}}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} + \frac{\partial \vec{U}}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j}\right) dS \end{aligned} \quad (32)$$

The field integral marked with FAE is eliminated by satisfying the field adjoint equations, eq. 8 (not surprisingly, the adjoint equations are identical to those used to compute the gradient of F). The term marked with $SWCR$ is also eliminated provided that the adjoint compatibility relation, eq. 9, is satisfied along the solid

walls. Finally, the boundary conditions along the inlet and outlet can be derived from

$$\vec{\Psi}^T \frac{\delta^2(A_n \vec{U})}{\delta b_i \delta b_j} = \vec{0} \quad \Rightarrow \quad \delta^2 \vec{U}^T (A_n^T \vec{\Psi}) = \vec{0} \tag{33}$$

eliminating, thus, the term marked with $IOBC$ in eq. 32. The solution of eq. 8, with eqs. 9 and 33 as boundary conditions, provides the $\vec{\Psi}$ field. The Hessian matrix is, then, computed by

$$\begin{aligned}
\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} &= \int_{S_w} \frac{\delta p}{\delta b_i} \frac{\delta p}{\delta b_j} dS + \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_i} \frac{\delta (dS)}{\delta b_j} + \int_{S_w} (p - p_{tar}) \frac{\delta p}{\delta b_j} \frac{\delta (dS)}{\delta b_i} \\
&+ \frac{1}{2} \int_{S_w} (p - p_{tar})^2 \frac{\delta^2 (dS)}{\delta b_i \delta b_j} + \int_{S_w} (\Psi_{k+1} p - \vec{\Psi}^T \vec{f}_k) \frac{\delta^2 n_k}{\delta b_i \delta b_j} dS \\
&+ \int_{S_w} \left(\Psi_{k+1} \frac{\delta p}{\delta b_i} - \vec{\Psi}^T \frac{\delta \vec{f}_k}{\delta b_i} \right) \frac{\delta n_k}{\delta b_j} dS + \int_{S_w} \left(\Psi_{k+1} \frac{\delta p}{\delta b_j} - \vec{\Psi}^T \frac{\delta \vec{f}_k}{\delta b_j} \right) \frac{\delta n_k}{\delta b_i} dS \\
&- \int_{S_w} \vec{\Psi}^T A_k n_k \left(\frac{\partial^2 \vec{U}}{\partial b_i \partial x_l} \frac{\delta x_l}{\delta b_j} + \frac{\partial^2 \vec{U}}{\partial b_j \partial x_l} \frac{\delta x_l}{\delta b_i} + \frac{\partial^2 \vec{U}}{\partial x_l \partial x_m} \frac{\delta x_l}{\delta b_i} \frac{\delta x_m}{\delta b_j} + \frac{\partial \vec{U}}{\partial x_l} \frac{\delta^2 x_l}{\delta b_i \delta b_j} \right) dS
\end{aligned} \tag{34}$$

Provided that the linearised flow equations are solved for the computation of the flow variable sensitivities $\frac{\delta \vec{U}}{\delta b_j}$, at the cost of N equivalent flow solutions, only one additional system of adjoint equations needs to be solved for the computation of the Hessian matrix. Thus, the total computational cost of the *direct–adjoint* approach is equal to $N+2$ equivalent flow solutions per optimization cycle. It is evident that the *direct–adjoint* approach is the less time–consuming among the four possible approaches.

5 Application to the Design of a 2D Turbine Cascade

This section presents the inverse design of a 2D turbine cascade (with a predefined target pressure distribution) in inviscid flow conditions, using gradient–based and Hessian–based optimization algorithms. The continuous adjoint approach is used to compute the gradient of the objective function with respect to the design variables and the direct–adjoint approach is implemented for the computation of the Hessian matrix. After the validation of the gradient and Hessian values, several optimization algorithms are compared with respect to their convergence characteristics. The convergence histories of the functional, gradient and Hessian values are illustrated.

The Euler equations are solved for the flow through the cascade. The flow conditions are: α_1=19 and M_2=0.6. The airfoil is parameterized using Bezier polynomials and 16 control points among which the ordinates of the 14 internal points are free to move (N=14). The initial set of the design variables defines an airfoil that produces a pressure distribution which is far apart from the reference (target) one. The objective function gradient values for the initial configuration, computed using the metrics–free adjoint approach, the conventional adjoint approach, the direct sensitivity analysis and a central finite difference scheme are illustrated in fig. (2), left. The comparison of gradient values is excellent. We recall that the direct approach and finite differences require the solution of the linearised flow equations or the flow

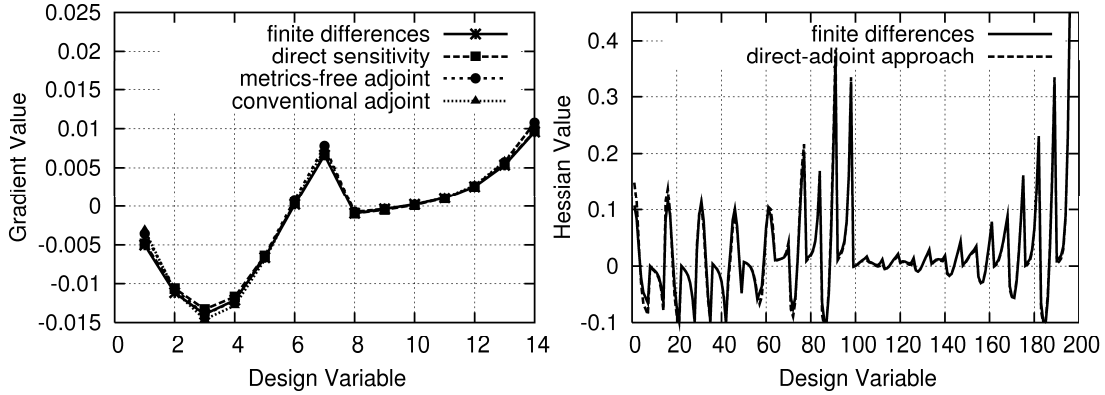

Figure 2: Inverse design of a 2D turbine cascade. Left: objective function gradient values using the metrics–free and the traditional adjoint approaches, the direct sensitivity analysis and finite differences. Right: Hessian matrix values using the direct–adjoint approach and finite differences. The first 14 values correspond to the first row of $\frac{d^2F}{db_i db_j}$ and so forth (14 columns $\times$ 14 rows = 196 values; they are all shown here, although the Hessian matrix is symmetric).

equations, respectively, as many times as the number of the design variables ($N+1$ equivalent flow solutions). In contrast, in the adjoint approaches, the cost per optimization cycle is equal to that required for the solution of the flow and adjoint equations (two equivalent flow solutions). Also, the metrics–free adjoint approach is less time–consuming than the conventional one, since it avoids the 2N calls to the grid generation software per cycle (for central finite difference schemes). The Hessian matrix values computed using the direct–adjoint approach are compared to finite differences in fig. (2), right. Again, the comparison is excellent.

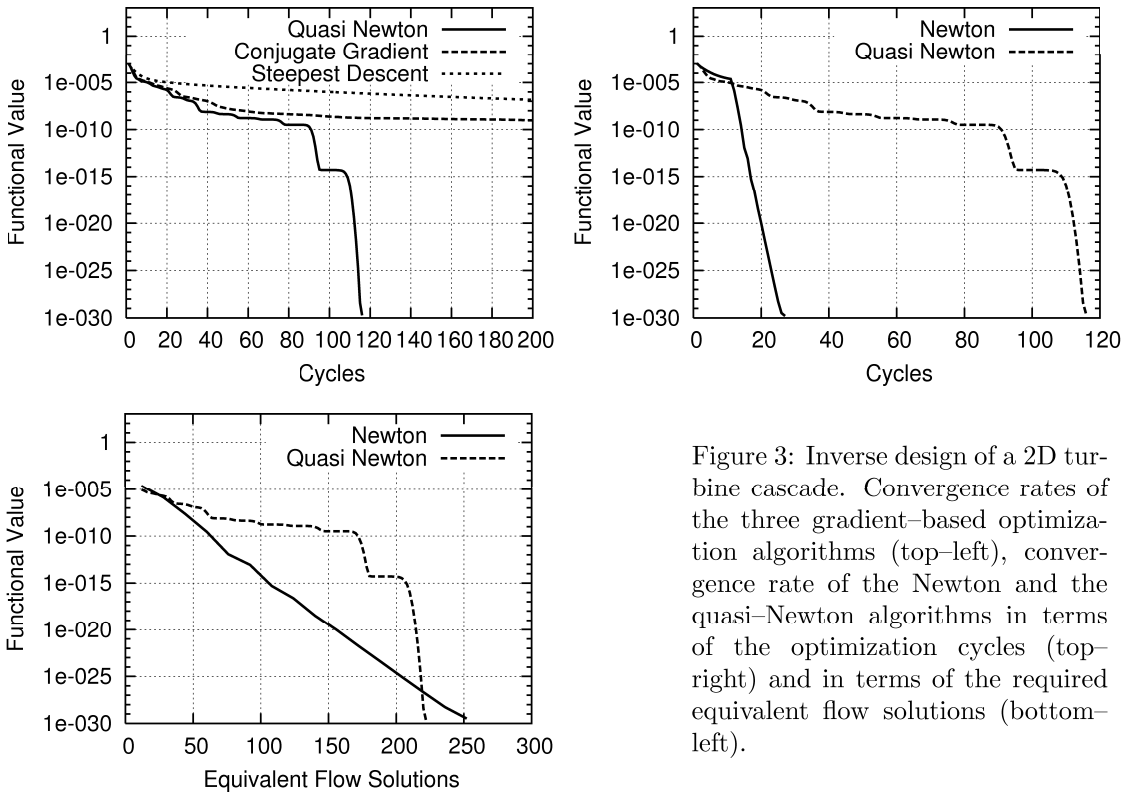

Figure 3: Inverse design of a 2D turbine cascade. Convergence rates of the three gradient–based optimization algorithms (top–left), convergence rate of the Newton and the quasi–Newton algorithms in terms of the optimization cycles (top–right) and in terms of the required equivalent flow solutions (bottom–left).

The comparison of the convergence histories for three different gradient–based optimization algorithms is shown in fig. (5), top–left. The conjugate gradient method is faster than steepest descent algorithm while the superiority of the quasi–Newton (BFGS) method over the previous ones is obvious. At the top–right of fig. (5), the BFGS algorithm is compared to the exact Newton method, based on the (exact) Hessian computed using the direct–adjoint approach. In order to accelerate the overall convergence rate, the steepest descent algorithm is used for the first ten cycles. It is obvious that the Newton method outperforms the quasi–Newton algorithm, in terms of the required optimization cycles. However, to be fair in comparison, fig. (5), bottom, shows the convergence rate of the two algorithms in terms of the required equivalent flow solutions. So, the x–axis scale is multiplied by 2 in the quasi–Newton approach and 16, $(N+2=14+2)$, in the Newton approach. Even in this case, the Newton approach performs better, especially at the first and most important optimization cycles.

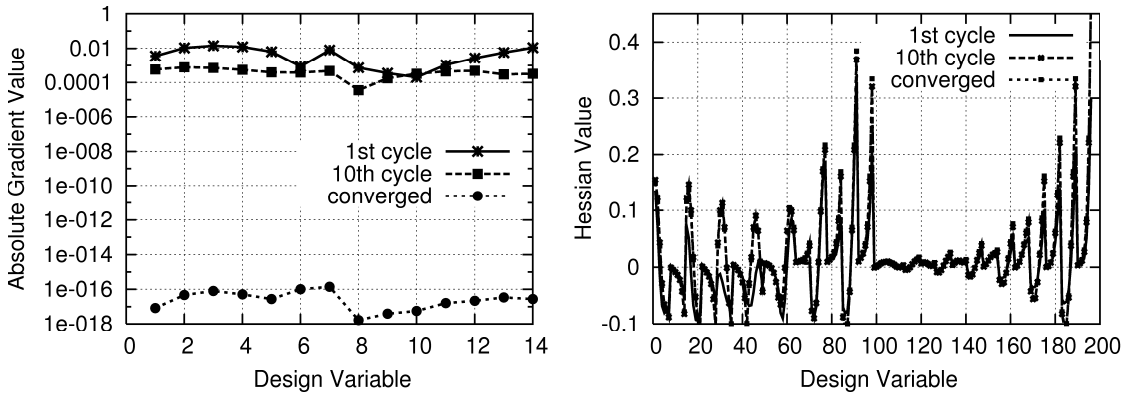

Figure 4: Inverse design of a 2D turbine cascade. Convergence history of the inverse design functional gradient (left) and Hessian (right) values during the optimization.

Gradient and Hessian values, computed using the metrics–free adjoint approach is illustrated in fig. 4. It can be seen that the gradient absolute values tend to zero upon convergence while the corresponding Hessian matrix values undergo minor changes; major changes can be observed only during the first cycles.

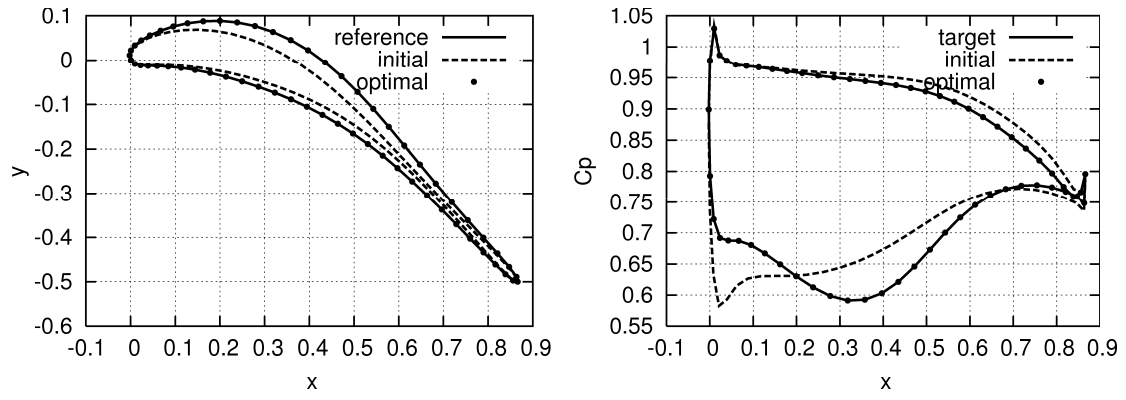

Figure 5: Inverse design of a 2D turbine cascade. Comparison of the initial, optimal and reference airfoil contours and the corresponding pressure distributions.

In fig. 5, the initial, reference and optimal (computed using the exact Newton method) contours as well as the corresponding pressure distributions are compared. In the case of steepest descent or conjugate gradient, similar figures can be obtained. It is clear that for the deviation between the target and optimal pressure distributions to vanish, these two algorithms should keep turning for a prohibitively long period of time; with the exact Newton method, this is achieved within less than 30 cycles.

6 Conclusions

Continuous adjoint and direct approaches for the computation of the gradient and the Hessian matrix of functionals used for the optimal design of aerodynamic shapes were presented. Both first and second order sensitivity expressions are free of field integrals, resulting to lower CPU cost and increased accuracy, since the computation of field geometrical variations is avoided. Concerning the Hessian matrix, four different approaches were presented and compared for the total required computational cost. The more effective one, i.e. the so–called *direct–adjoint* approach (in which the direct approach was used to compute the flow sensitivities and the adjoint one to compute the Hessian matrix values) was adopted. Although the total CPU cost depends on the number of design variables, the Newton method based on the exact Hessian matrix proved to outperform the quasi–Newton approach, which was the best–performing Hessian–free algorithm.

REFERENCES

[1] Pironneau O. On optimum design in fluid mechanics. *Journal of Fluid Mechanics*, 64:97–110, 1974.

[2] Pironneau O. *Optimal shape design for elliptic systems*. Springer-Verlag, New York, 1984.

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

[4] Jameson A. Optimum aerodynamic design using CFD and control theory. AIAA Paper 95-1729, AIAA 12th Computational Fluid Dynamics Conference, San Diego, June 1995.

[5] Pierce NA, Giles MB. An introduction to the adjoint approach to design. *Flow, Turbulence and Combustion*, 65(3-4):393–415, 2000.

[6] Elliot J and Peraire J. Aerodynamic design using unstructured meshes. AIAA Paper 96-1941, 1996.

[7] Anderson WK, Bonhaus DL. Airfoil design on unstructured grids for turbulent flows. *AIAA Journal*, 37(2):185–191, 1999.

[8] Hazra S, Schulz V, Brezillon J, Gauger N. Aerodynamic shape optimization using simultaneous pseudo-timestepping. *Journal of Computational Physics*, 204(1):46–64, 2005.

[9] Tortorelli D, Michaleris P. Design sensitivity analysis: Overview and review. *Inverse Problems in Engineering*, 1:71–105, 1994.

[10] Le Dimet FX, Navon IM, Daescu DN. Second-order information in data assimilation. *Monthly Weather Review*, 130(3):629–648, 2002.

[11] Bertsekas DP. *Nonlinear Programming.* Athena Scientific, 2nd edition, 1999.

[12] Jameson A, Kim S. Reduction of the adjoint gradient formula in the continous limit. AIAA-2003-0040, AIAA 41th Aerospace Sciences Meeting and Exhibit, Reno NV, January 2003.

[13] Papadimitriou DI, Giannakoglou KC. A continuous adjoint method with objective function derivatives based on boundary integrals for inviscid and viscous flows. *Computers and Fluids*, 36:325–341, 2007.

[14] Roe P. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–371, 1981.

[15] Papadimitriou DI, Giannakoglou KC. Compressor blade optimization using a continuous adjoint formulation. ASME TURBO EXPO, GT2006/90466, Barcelona, 2006.