V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal,14-17 June 2010

ONE-SHOT SHAPE OPTIMIZATION USING THE EXACT HESSIAN

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

National Technical University of Athens,
School of Mechanical Engineering,
Lab. of Thermal Turbomachines.
Parallel CFD & Optimization Unit.
P.O. Box 64069, 15710 Athens, GREECE.
e-mail: dpapadim@mail.ntua.gr*, kgianna@central.ntua.gr†

Key words: Aerodynamic Shape Optimization, Adjoint, Direct Differentiation, Hessian Matrix, Second-Order Sensitivities, Newton Method

Abstract. *One-shot Newton methods based, fully or partially, on the computation of the exact Hessian matrix in aerodynamic shape optimization problems are proposed and compared. They start by computing the exact first- and second-order sensitivities of the objective function with respect to the design variables, i.e. by solving the flow, adjoint and direct differentiation equations, the one after the other. The continuous adjoint method, for the problems governed by the Euler equations, is used. The combination of the direct differentiation of the flow equations and the adjoint method is the most economical way to compute the exact Hessian matrix. Based on both gradient and Hessian, the shape is updated using the Newton equation. From this point on, the flow and adjoint equations can be solved in a coupled manner along with either the direct differentiation equations (exact Newton, by computing the exact Hessian anew) or the BFGS updating formula (quasi-Newton, by approximating the Hessian matrix based on the gradient) and the shape updating expression. These approaches will be referred to as "one-shot (exact) Newton method" and "one-shot, exactly initialized, quasi-Newton method", respectively. Both are compared in terms of the required CPU cost with other optimization algorithms, including their segregated variants. For the comparison, an application concerned with the inverse design of an airfoil cascade is used.*

1 INTRODUCTION

In the field of aerodynamics, the adjoint approach, [1, 2, 3], is often used to support shape optimization problems related to several configurations. By means of the adjoint approach, the gradient of any functional with respect to the design variables can be computed at CPU cost which does not depend on the number of optimization variables, as opposed to other approaches (such as finite differences). A conventional optimization algorithm based on the adjoint approach performs cycles, each of which comprises the numerical solution of the flow and adjoint equations, the one after the other, followed by the application of a shape updating formula, such as steepest descent. In each cycle, the flow equations must reach a very low residual error, before initiating the solution of the adjoint equations; in contrast, it is usually enough for the adjoint equations to run until ~ 4 orders of magnitude drop in the residual.

As a more efficient alternative to their segregated solution, the flow, adjoint and shape updating equations can be solved simultaneously. Such an algorithm has been proposed in [4, 5], by making use of the multigrid strategy to solve the flow and adjoint equations, with the design process embedded within the multigrid cycles. Similar approaches, based on the simultaneous pseudo–time stepping technique, with the same pseudo–time step for the ensemble of equations, can be found in [6, 7, 8]. Also, in [9], divided differences (instead of the adjoint approach) have been used whereas a progressive algorithm updating the design variables after partially converging the flow and, then, the adjoint equation, can be found in [10, 11]. All the aforementioned papers rely on standard first-order descent algorithms or quasi-second-order (i.e. quasi-Newton) ones, [6, 7, 8], using Hessian matrix approximations.

None of the methods overviewed before requires the knowledge of the exact Hessian matrix. A literature survey on the computation of the latter led to a seriously limited number of relevant papers. In [12], the computation of second-order sensitivities of an aerodynamic optimization function, using automatic differentiation, has been presented. The Hessian matrix computation in structural optimization, [13], continuous approaches for the Hessian computation in heat conduction problems, [14] and a different handling of the same problem for variational data assimilation problems in meteorology, [15, 16, 17], must be reported. The importance of exactly computing the Hessian matrix, focusing on the influence of the condition number on the convergence of an optimization algorithm, is discussed in [18].

In previous works by the present authors, [19], [20], [21], [22], four methods to compute the exact Hessian, based on all possible ways of coupling direct differentiation and adjoint approaches, have been presented. This classification can also be found in a few previous papers, such as [12] and [14]. A noticeable contribution of [19], [20], [21] and [22] was to develop both the continuous and discrete (with hand differentiation) approaches for the Hessian matrix computation and apply them to the inverse design of 2D cascades and ducts using the exact and/or quasi-Newton approach. The (inviscid or viscous) flow,

adjoint and direct differentiation (if necessary) equations were all solved in a segregated manner. Also, in [22], the combination of exact and quasi-Newton in a scheme in which the exact Hessian matrix was computed once in the beginning, before switching permanently to simpler Hessian updating formulas (such as BFGS, SR1, etc, [23]) proved to reduce the CPU cost, even in problems with many design variables for which the computation of the exact Hessian matrix within each and every cycle is not affordable.

In this paper, efficient optimization methods which: (a) require the computation of the Hessian matrix, at least once and (b) employ the one-shot solution scheme are proposed. They will be compared with steepest descent, exact and quasi-Newton methods, in which all equations are solved in a segregated manner. For the computation of the Hessian matrix, the direct differentiation of the flow equations with respect to the design variables and the continuous adjoint method are used. Without loss in generality, the applications shown correspond to inviscid flows.

2 HESSIAN-BASED ONE-SHOT OPTIMIZATION METHODS

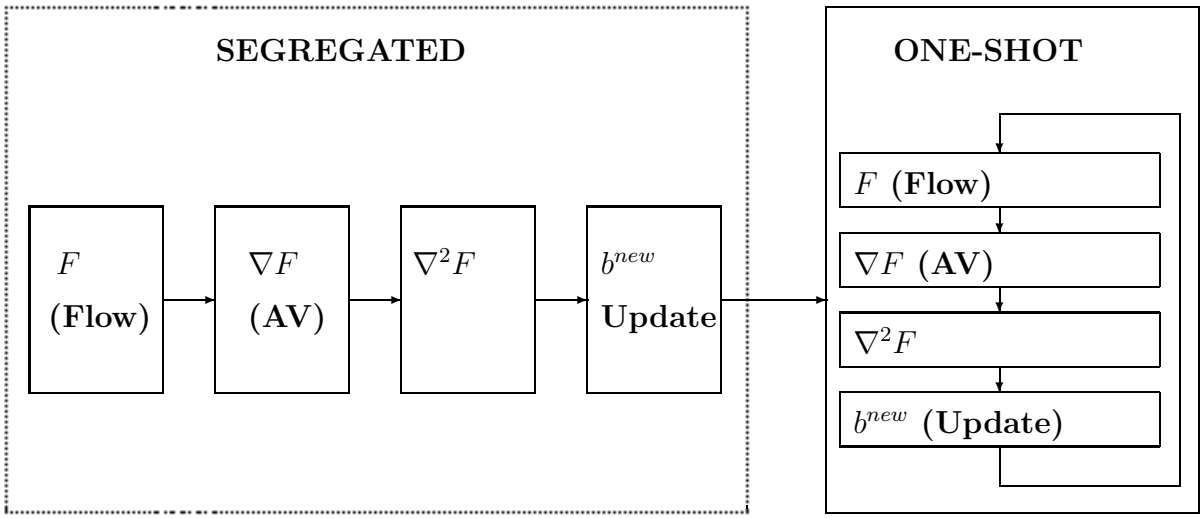
Based on the literature, [7, 8], a conventional one-shot approach for the minimization of the objective function F makes use of the simultaneous solution of the flow and adjoint (AV) equations, including the design variables b update at the end of each iteration. Steepest descent or any quasi-Newton algorithm with Hessian matrix approximations can be employed. For instance, according to the BFGS quasi-Newton method (Broyden, Fletcher, Goldfarb, Shanno, [23]),

$$b_i^{\kappa+1} = b_i^{\kappa} + \Delta b_i^{\kappa} \ , \qquad \Delta b_i^{\kappa} = -B_{ij}^{\kappa} \ (\nabla F)_j^{\kappa} \tag{1}$$

where B^{κ} is the current (i.e. at the κ-th cycle) approximation to the inverse of $\nabla^2 F$, given by the equation

$$B_{ij}^{\kappa} = \left[\delta_{il} - \frac{s_i^{\kappa-1} r_l^{\kappa-1}}{r_n^{\kappa-1} s_n^{\kappa-1}}\right] B_{lm}^{\kappa-1} \left[\delta_{mj} - \frac{r_m^{\kappa-1} s_j^{\kappa-1}}{r_n^{\kappa-1} s_n^{\kappa-1}}\right] + \frac{s_i^{\kappa-1} s_j^{\kappa-1}}{r_n^{\kappa-1} s_n^{\kappa-1}} \tag{2}$$

where $s_i^{\kappa-1} = b_i^{\kappa} - b_i^{\kappa-1}$, $r_i^{\kappa-1} = (\nabla F)_i^{\kappa} - (\nabla F)_i^{\kappa-1}$ and δ_{ij} is the Kronecker symbol. In its starting phase, a one-shot algorithm requires the convergence of the flow and AV equations once. This is carried out in a segregated manner, before updating the design variables for the first time. From this point on, the simultaneous solution (one iteration at a time) of the flow, AV and update equations is carried out. Such an algorithm is schematically shown below:



where $\nabla^2 F$ is computed via $BFGS$ in both the segregated (left part) and one-shot (right part) phases; according to the abbreviations introduced in the Results section, this will be referred to as the OQN method.

Based on the above, one of the herein proposed algorithms (abbreviated to OEN; see the Results section) can be formed by the following steps:

1. At the starting phase, compute ∇F and $\nabla^2 F$, by solving the flow, AV and direct differentiation (DD) equations, one after the other (segregated DD-AV approach; see section 3).

2. Update the design variables b using the Newton equation.

3. Iteratively perform one iteration of the flow, AV and DD equations and compute the gradient and Hessian matrix (one-shot DD-AV approach). Return to step 2.

A second algorithm, also proposed and tested (abbreviation: OEQN) in this paper, includes the following steps:

1. At the starting phase, compute ∇F and $\nabla^2 F$, as in the previous algorithm.

2. Update b using the Newton equation, as in the previous algorithm.

3. Perform one iteration of the solution of the flow and AV equations, by approximating the Hessian matrix using BFGS. Return to step 2.

The previous flowchart can readily be adapted to both new algorithms. It suffices to complete the methods used for the computation of $\nabla^2 F$ (two boxes). The $\nabla^2 F$ computation "box" in the segregated part of the flowchart must be marked with $DD - AV$ for

both algorithms and the similar "box" in the one-shot part with $DD-AV$ (for OEN) or $BFGS$ (for OEQN).

3 COMPUTATION OF THE EXACT HESSIAN: THE DD-AV METHOD

In this paper, the development of the continuous adjoint approach makes use of the Euler equations for compressible flows,

$$\frac{\partial U_n}{\partial t}+\frac{\partial f_{nk}}{\partial x_k}=0 \tag{3}$$

as state equations, where $1\leq n\leq\Lambda$ for either 2D (where $\Lambda=4$) or 3D ($\Lambda=5$). The summation convention applies where repeated indices appear. The conservative flow variables U_n and the inviscid fluxes f_{nk} are given by

$$\begin{bmatrix} U_1 \\ U_q \\ U_\Lambda \end{bmatrix}=\begin{bmatrix} \rho \\ \rho u_{q-1} \\ E \end{bmatrix},\quad \begin{bmatrix} f_{1k} \\ f_{qk} \\ f_{\Lambda k} \end{bmatrix}=\begin{bmatrix} \rho u_k \\ \rho u_k u_{q-1}+p\delta_{k,q-1} \\ u_k(E+p) \end{bmatrix} \tag{4}$$

($2\leq q\leq\Lambda-1$, $1\leq k\leq\Lambda-2$). In eq. 4, ρ, p, u_k and $E=\rho e+\frac{1}{2}\rho u_k^2$, stand for the density, pressure, velocity components and total energy per unit volume, respectively.

In inverse design problems, the objective function to be minimized is the integral of the deviation of the pressure distribution $p(S)$ from a given target distribution $p_{tar}(S)$ along the solid walls S_w. Thus,

$$F=\frac{1}{2}\int_{S_w}(p-p_{tar})^2 dS \tag{5}$$

The first- and second-order sensitivities of F with respect to the design variables b_i (geometrical quantities which, based on Bezier polynomials, Splines, etc, control the aerodynamic shape) are

$$\frac{\delta F}{\delta b_i}=\int_{S_w}(p-p_{tar})\frac{\delta p}{\delta b_i}dS+\frac{1}{2}\int_{S_w}(p-p_{tar})^2\frac{\delta(dS)}{\delta b_i} \tag{6}$$

$$\begin{aligned}\frac{\delta^2 F}{\delta b_i\delta b_j} &= \int_{S_w}\frac{\delta p}{\delta b_i}\frac{\delta p}{\delta b_j}dS+\int_{S_w}(p-p_{tar})\frac{\delta^2 p}{\delta b_i\delta b_j}dS+\int_{S_w}(p-p_{tar})\frac{\delta p}{\delta b_i}\frac{\delta(dS)}{\delta b_j}\\ &+\int_{S_w}(p-p_{tar})\frac{\delta p}{\delta b_j}\frac{\delta(dS)}{\delta b_i}+\frac{1}{2}\int_{S_w}(p-p_{tar})^2\frac{\delta^2(dS)}{\delta b_i\delta b_j}\end{aligned} \tag{7}$$

which both depend on the first- and second-order sensitivities of the flow variables.

In the direct differentiation-adjoint variable (DD-AV, or merely "direct-adjoint", [22]) approach, the first-order sensitivities are derived by solving the differentiated flow equations

$$\frac{\delta}{\delta b_i}\left(\frac{\partial f_{nk}}{\partial x_k}\right)=0\Rightarrow\frac{\partial}{\partial b_i}\left(\frac{\partial f_{nk}}{\partial x_k}\right)=0\Rightarrow\frac{\partial}{\partial b_i}\left(A_{nmk}\frac{\partial U_m}{\partial x_k}\right)=0 \tag{8}$$

where $A_{nmk} = \frac{\partial f_{nk}}{\partial U_m}$. For eq. 8 to be exact, the interior grid point coordinates must not be affected by variations in b_i. Similarly, the boundary conditions for eq. 8 are derived from the differentiation of the boundary conditions imposed to the flow variables. For instance, along the wall boundaries, where $u_k n_k = 0$ (no-penetration condition),

$$\frac{\delta(u_k n_k)}{\delta b_i} = 0 \Rightarrow \frac{\partial u_k}{\partial b_i} n_k = -\frac{\partial u_k}{\partial x_l}\frac{\delta x_l}{\delta b_i} n_k - u_k \frac{\delta n_k}{\delta b_i} \tag{9}$$

where $\frac{\delta x_l}{\delta b_i}$ and $\frac{\delta n_k}{\delta b_i}$ depend on the shape parameterization and may even lead to closed form expressions. The computation of the second-order sensitivities of the flow variables is avoided by means of the adjoint approach. By introducing the adjoint variables Ψ_n, the second-order sensitivities of F are equal to those of F_{aug}, where

$$\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} = \frac{\delta^2 F}{\delta b_i \delta b_j} + \int_\Omega \Psi_n \frac{\partial^2}{\partial b_i \partial b_j}\left(\frac{\partial f_{nk}}{\partial x_k}\right) d\Omega + \mathcal{G}_{ij} \tag{10}$$

where

$$\begin{aligned}
\mathcal{G}_{ij} &= \int_S \Psi_n \frac{\partial}{\partial b_j}\left(\frac{\partial f_{nk}}{\partial x_k}\right)\frac{\delta x_l}{\delta b_i} n_l dS \\
&+ \int_S \Psi_n \frac{\partial}{\partial b_i}\left(\frac{\partial f_{nk}}{\partial x_k}\right)\frac{\delta x_l}{\delta b_j} n_l dS + \int_S \Psi_n \frac{\partial f_{nk}}{\partial x_k}\frac{\delta^2 x_l}{\delta b_i \delta b_j} n_l dS \\
&+ \int_S \Psi_n \frac{\partial^2 f_{nk}}{\partial x_k \partial x_m}\frac{\delta x_m}{\delta b_j}\frac{\delta x_l}{\delta b_i} n_l dS + \int_S \Psi_n \frac{\partial f_{nk}}{\partial x_k}\frac{\delta x_l}{\delta b_i}\frac{\delta(n_l dS)}{\delta b_j}
\end{aligned} \tag{11}$$

Proving eqs. 10 and 11 is beyond the scope of this paper. $\frac{\partial}{\partial x_k}$ and $\frac{\partial^2}{\partial b_i \partial b_j}$ can be interchanged, yielding

$$\int_\Omega \Psi_n \frac{\partial^2}{\partial b_i \partial b_j}\left(\frac{\partial f_{nk}}{\partial x_k}\right) d\Omega = \int_\Omega \Psi_n \frac{\partial}{\partial x_k}\left(\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j}\right) d\Omega \tag{12}$$

Through integration by parts,

$$\int_\Omega \Psi_n \frac{\partial}{\partial x_k}\left(\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j}\right) d\Omega = -\int_\Omega \frac{\partial \Psi_n}{\partial x_k}\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j} d\Omega + \int_S \Psi_n \frac{\partial^2 f_{nk}}{\partial b_i \partial b_j} n_k dS$$

or

$$\begin{aligned}
\int_\Omega \Psi_n \frac{\partial}{\partial x_k}\left(\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j}\right) d\Omega = &- \int_\Omega \left(A_{nmk}\frac{\partial \Psi_n}{\partial x_k}\right)\frac{\partial^2 U_m}{\partial b_i \partial b_j} d\Omega + \int_\Omega \frac{\partial A_{nmk}}{\partial b_j}\frac{\partial U_m}{\partial b_i}\frac{\partial \Psi_n}{\partial x_k} d\Omega \\
&+ \int_S \Psi_n \frac{\partial^2 f_{nk}}{\partial b_i \partial b_j} n_k dS
\end{aligned} \tag{13}$$

since

$$\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j} = A_{nmk}\frac{\partial^2 U_m}{\partial b_i \partial b_j} + \frac{\partial A_{nmk}}{\partial U_l}\frac{\partial U_m}{\partial b_i}\frac{\partial U_l}{\partial b_j} \tag{14}$$

where $A_{nmk} = \frac{\partial f_{nk}}{\partial U_l}$ and $\frac{\partial A_{nmk}}{\partial U_l}\frac{\partial U_m}{\partial b_i}\frac{\partial U_l}{\partial b_j} = \frac{\partial A_{nmk}}{\partial U_l}\frac{\partial U_l}{\partial b_i}\frac{\partial U_m}{\partial b_j}$. Note that $S = S_w \cup S_{I,O}$, where $S_{I,O}$ denotes the inlet-outlet boundary of the flow domain.

The relation between the partial ($\frac{\partial^2}{\partial b_i \partial b_j}$) and total ($\frac{\delta^2}{\delta b_i \delta b_j}$) second-order sensitivities of F is

$$\begin{aligned}\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j} &= \frac{\delta^2 f_{nk}}{\delta b_i \delta b_j} - \frac{\partial^2 f_{nk}}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} \\ &- \frac{\partial^2 f_{nk}}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} - \frac{\partial^2 f_{nk}}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} - \frac{\partial f_{nk}}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j}\end{aligned} \tag{15}$$

Along the boundary S (either S_w or $S_{I,O}$), the following condition

$$\frac{\delta^2 f_{nk}}{\delta b_i \delta b_j} n_k = \frac{\delta^2 (f_{nk} n_k)}{\delta b_i \delta b_j} - \frac{\delta f_{nk}}{\delta b_i}\frac{\delta n_k}{\delta b_j} - \frac{\delta f_{nk}}{\delta b_j}\frac{\delta n_k}{\delta b_i} - f_{nk}\frac{\delta^2 n_k}{\delta b_i \delta b_j} \tag{16}$$

is valid. Note also that along S_w in specific, the no–penetration condition gives

$$\frac{\delta^2 (f_{nk} n_k)}{\delta b_i \delta b_j} = N_n \frac{\delta^2 p}{\delta b_i \delta b_j} + \frac{\delta^2 N_n}{\delta b_i \delta b_j} p + \frac{\delta N_n}{\delta b_i}\frac{\delta p}{\delta b_j} + \frac{\delta N_n}{\delta b_j}\frac{\delta p}{\delta b_i} \tag{17}$$

where $(N_1, N_2, N_3, N_4, N_5) = (0, pn_1, pn_2, pn_3, 0)dS$. By appropriately rearranging its terms, eq. 13 becomes

$$\begin{aligned}\int_\Omega \Psi_n \frac{\partial}{\partial x_k}\left(\frac{\partial^2 f_{nk}}{\partial b_i \partial b_j}\right) d\Omega &= -\int_\Omega \left(A_{nmk}\frac{\partial \Psi_n}{\partial x_k}\right)\frac{\partial^2 U_m}{\partial b_i \partial b_j} d\Omega + \int_\Omega \frac{\partial A_{nmk}}{\partial b_j}\frac{\partial U_m}{\partial b_i}\frac{\partial \Psi_n}{\partial x_k} d\Omega \\ &+ \int_{S_{I,O}} \Psi_n \frac{\delta^2 f_{nk}}{\delta b_i \delta b_j} n_k dS + \int_{S_w} \Psi_{n+1} n_n \frac{\delta^2 p}{\delta b_i \delta b_j} dS + \int_{S_w} (\Psi_{k+1} p - \Psi_n f_{nk})\frac{\delta^2 n_k}{\delta b_i \delta b_j} dS \\ &+ \int_{S_w}\left(\Psi_{k+1}\frac{\delta p}{\delta b_i} - \Psi_n \frac{\delta f_{nk}}{\delta b_i}\right)\frac{\delta n_k}{\delta b_j} dS + \int_{S_w}\left(\Psi_{k+1}\frac{\delta p}{\delta b_j} - \Psi_n \frac{\delta f_{nk}}{\delta b_j}\right)\frac{\delta n_k}{\delta b_i} dS \\ &- \int_{S_w} \Psi_n \left(\frac{\partial^2 f_{nk}}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 f_{nk}}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} + \frac{\partial^2 f_{nk}}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} + \frac{\partial f_{nk}}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j}\right) n_k dS\end{aligned} \tag{18}$$

Note that, along $S_{I,O}$, the nodal coordinates remain invarian so, there, $\frac{\partial \Phi}{\partial b_i} = \frac{\delta \Phi}{\delta b_i}$ for any

Φ. Combining eqs. 7 and 18, we get

$$\begin{aligned}
\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} &= \int_{S_w} \frac{\delta p}{\delta b_i}\frac{\delta p}{\delta b_j} dS + \underbrace{\int_{S_w} (p-p_{tar}) \frac{\delta^2 p}{\delta b_i \delta b_j} dS}_{SWCR} + \int_{S_w} (p-p_{tar}) \frac{\delta p}{\delta b_i}\frac{\delta (dS)}{\delta b_j} \\
&+ \int_{S_w} (p-p_{tar}) \frac{\delta p}{\delta b_j}\frac{\delta (dS)}{\delta b_i} + \frac{1}{2}\int_{S_w} (p-p_{tar})^2 \frac{\delta^2 (dS)}{\delta b_i \delta b_j} \\
&- \underbrace{\int_{\Omega} \left(A_{nmk}\frac{\partial \Psi_n}{\partial x_k} \right) \frac{\partial^2 U_m}{\partial b_i \partial b_j} d\Omega}_{FAE} + \int_{\Omega} \frac{\partial A_{nmk}}{\partial b_j}\frac{\partial U_m}{\partial b_i}\frac{\partial \Psi_n}{\partial x_k} d\Omega + \underbrace{\int_{S_{I,O}} \Psi_n \frac{\delta^2 f_{nk}}{\delta b_i \delta b_j} n_k dS}_{IOBC} \\
&+ \underbrace{\int_{S_w} \Psi_{n+1} n_n \frac{\delta^2 p}{\delta b_i \delta b_j} dS}_{SWCR} + \int_{S_w} (\Psi_{k+1} p - \Psi_n f_{nk}) \frac{\delta^2 n_k}{\delta b_i \delta b_j} dS \\
&+ \int_{S_w} \left(\Psi_{k+1}\frac{\delta p}{\delta b_i} - \Psi_n \frac{\delta f_{nk}}{\delta b_i} \right) \frac{\delta n_k}{\delta b_j} dS + \int_{S_w} \left(\Psi_{k+1}\frac{\delta p}{\delta b_j} - \Psi_n \frac{\delta f_{nk}}{\delta b_j} \right) \frac{\delta n_k}{\delta b_i} dS \\
&- \int_{S_w} \Psi_n A_{nqk} n_k \left(\frac{\partial^2 U_q}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 U_q}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} \right) dS \\
&- \int_{S_w} \Psi_n \left(\frac{\partial^2 f_{nk}}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 f_{nk}}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} + \frac{\partial^2 f_{nk}}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} + \frac{\partial f_{nk}}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j} \right) n_k dS + \mathcal{G}_{ij} \quad (19)
\end{aligned}$$

Integrals marked with FAE, $SWCR$ and $IOBC$ are eliminated by satisfying the field adjoint equation and its boundary conditions along S_w and $S_{I,O}$. The field adjoint equation (in which a the pseudo-time term has been added) is given by

$$\frac{\partial \Psi_n}{\partial t} - A_{mnk}\frac{\partial \Psi_m}{\partial x_k} = 0 \quad (20)$$

the boundary conditions along the wall read

$$p - p_{tar} + \Psi_{r+1} n_r = 0 \quad (21)$$

and the inlet and outlet boundaries are derived from

$$\frac{\delta^2 U_n}{\delta b_i \delta b_j}(A_{nm}\Psi_m) = 0 \quad (22)$$

Note that, [22], the same equations (field equations and boundary conditions) would be solved to compute first-order sensitivities using the adjoint approach.

The remaining terms in eq. 19 yield an expression for $\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j}$, which is as follows

$$\begin{aligned}\frac{\delta^2 F_{aug}}{\delta b_i \delta b_j} &= \int_{S_w} \frac{\delta p}{\delta b_i}\frac{\delta p}{\delta b_j} dS + \int_{S_w} (p-p_{tar}) \frac{\delta p}{\delta b_i}\frac{\delta (dS)}{\delta b_j} + \int_{S_w} (p-p_{tar}) \frac{\delta p}{\delta b_j}\frac{\delta (dS)}{\delta b_i} \\ &+ \frac{1}{2}\int_{S_w} (p-p_{tar})^2 \frac{\delta^2 (dS)}{\delta b_i \delta b_j} + \int_{S_w} (\Psi_{k+1} p - \Psi_n f_{nk}) \frac{\delta^2 n_k}{\delta b_i \delta b_j} dS \\ &+ \int_{S_w} \left(\Psi_{k+1} \frac{\delta p}{\delta b_i} - \Psi_n \frac{\delta f_{nk}}{\delta b_i} \right) \frac{\delta n_k}{\delta b_j} dS + \int_{S_w} \left(\Psi_{k+1} \frac{\delta p}{\delta b_j} - \Psi_n \frac{\delta f_{nk}}{\delta b_j} \right) \frac{\delta n_k}{\delta b_i} dS \\ &+ \int_{\Omega} \frac{\partial A_{nmk}}{\partial b_j}\frac{\partial U_m}{\partial b_i}\frac{\partial \Psi_n}{\partial x_k} d\Omega \\ &- \int_{S_w} \Psi_n \left(\frac{\partial^2 f_{nk}}{\partial b_i \partial x_l}\frac{\delta x_l}{\delta b_j} + \frac{\partial^2 f_{nk}}{\partial b_j \partial x_l}\frac{\delta x_l}{\delta b_i} + \frac{\partial^2 f_{nk}}{\partial x_l \partial x_m}\frac{\delta x_l}{\delta b_i}\frac{\delta x_m}{\delta b_j} + \frac{\partial f_{nk}}{\partial x_l}\frac{\delta^2 x_l}{\delta b_i \delta b_j} \right) n_k dS + \mathcal{G}_{ij} \end{aligned} \quad (23)$$

Regarding the CPU cost of the methods examined in this paper, particularly those based on the computation of the exact Hessian matrix, we have to distinguish segregated from one-shot variants. It is evident that the starting (segregated) phase of the OEN or OEQN variants is identical to the first cycle of the SEN one. One SEN cycle costs as many as $2+N$ EFS; this is the sum of one EFS for solving the state equations, another EFS for the adjoint equations and N EFS for solving the N DD equations. A detailed quantitative comparison of the cost of the examined methods can be made based on the figures of this paper. Note that one "CPU time unit" (as marked on the abscissa axis of most of them) corresponds to the CPU cost of performing one iteration of the iterative scheme used to solve the flow (or adjoint or DD) equations. Therefore, one EFS corresponds to a number of "CPU time units", determined, among other, by the grid size or the case under examination.

4 APPLICATIONS-DISCUSSION

To assess the developed methods and the corresponding programmed software, they have all been used to solve the same problem, namely the inverse design of a cascade airfoil at inviscid flow conditions. The same problem is examined twice, with 6 and 12 design variables. Let us first list the eight methods-variants used and their abbreviations:

SSD: The Segregated Steepest Descent method, based on the AV approach to compute the gradient of F. Within each optimization cycle, the flow and adjoint equations are solved (adequately converged) in a segregated manner.

OSD: The One-shot Steepest Descent method, in which the flow, adjoint and geometry updating equations are solved simultaneously.

SQN: The Segregated Quasi-Newton method based on the BFGS updating formula for approximating $\nabla^2 F$ and the adjoint method for computing ∇F. The flow and adjoint equations are solved in a segregated manner.

OQN: The One-shot Quasi-Newton (i.e. BFGS) method, in which the flow, adjoint and geometry updating equations are solved simultaneously.

SEN: The Segregated Exact-Newton method, based on the computation of the exact Hessian matrix within each Newton step afresh. The flow, adjoint and DD equations are solved in a segregated manner.

OEN: The One-shot Exact-Newton method, in which the flow, adjoint and DD equations are solved simultaneously.

SEQN: The Segregated Exact-Quasi Newton method, which relies on the exact Newton method for the first cycle and, then, switches to the inexpensive BFGS for any subsequent cycle.

OEQN: The One-shot Exact-Quasi Newton method, in which, in the first phase, the flow, adjoint and DD equations are solved in a segregated manner and, then, the flow, adjoints equations and b updating (based on BFGS) formula are solved simultaneously.

The eight variants are compared in terms of the number of optimization cycles and, in specific, the CPU time units required to reach the optimal solution. One time unit corresponds to the CPU time of a single iteration of the flow, adjoint or direct-differentiation equations. A reasonable assumption is made that all of them have the same cost.

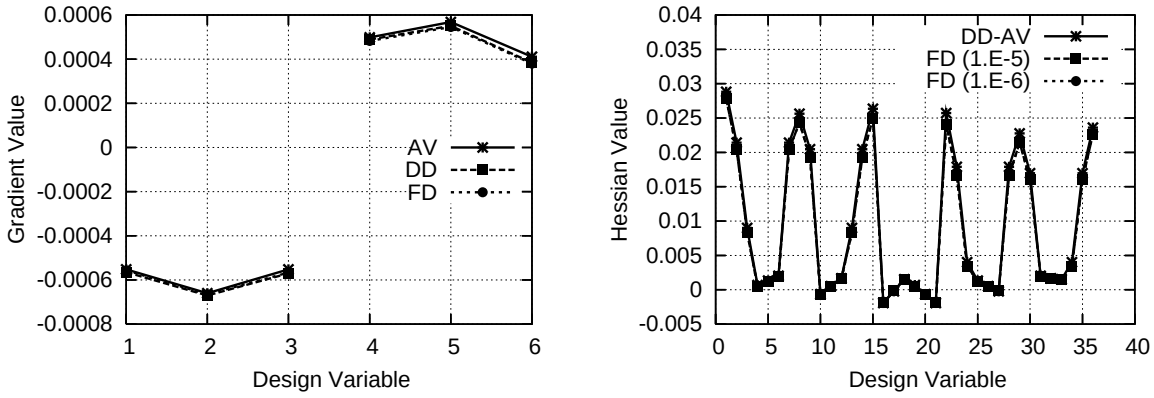


Figure 1: Inverse design of a cascade airfoil with 6 design variables. Left: first-order derivatives of F computed using AV, DD and FD. Right: Hessian matrix ($\nabla^2 F$) element values computed using the continuous DD-AV approach and FD; the first 6 values correspond to the first row of the Hessian and so forth.

The case examined is concerned with the inverse design of a symmetric cascade airfoil, based on a given pressure distribution along S_w. The flow conditions are: inlet flow angle $\alpha_1 = 35^o$ and isentropic exit Mach number $M_{2,is} = 0.4$. The airfoil contour is parameterized

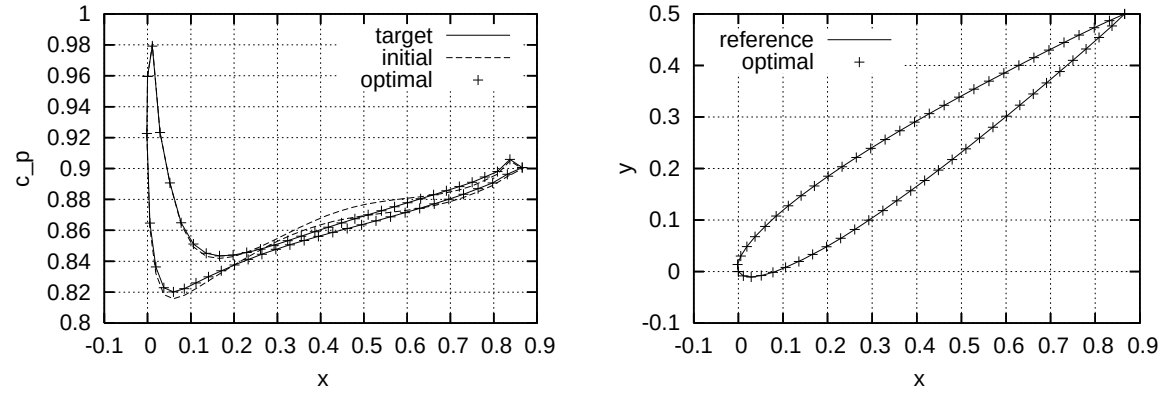


Figure 2: Inverse design of a cascade airfoil with 6 design variables: Comparison of the optimal and target wall pressure coefficient distributions and the corresponding airfoil contours (not in scale). The reference shape is the one used to create the target pressure distribution.

using two Bézier polynomials, with 3 degrees of freedom per airfoil side, leading to 6 design variables in total.

First-order sensitivities computed using the AV approach, DD and finite differences (FD) are compared in fig. 1, left. The corresponding values are almost identical, proving (among other) the ability to accurately compute the flow variable sensitivities $\frac{\delta U_j}{\delta b_i}$ (should this be required by the method in use). The comparison of the Hessian matrix values computed using the DD–AV approach and FD (with two ϵ values; ϵ stands for the FD step) is shown in fig. 1, right. The comparison is excellent. The use of FD with two ϵ values ($\epsilon = 10^{-5}$ and 10^{-6}) guarantees that ϵ–independent results are obtained.

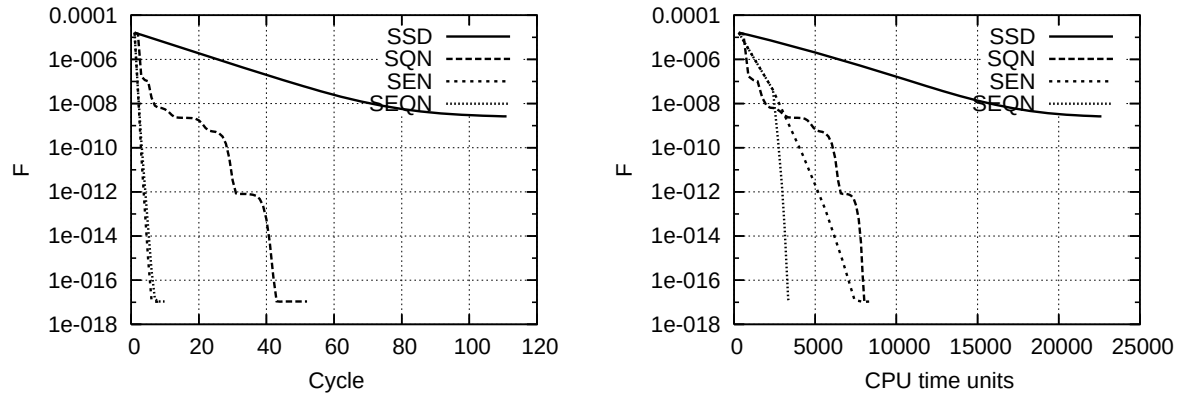


Figure 3: Inverse design of a cascade airfoil with 6 design variables: Reduction rates of the objective function value using the four segregated variants (SSD, SQN, SEN and SEQN) in terms of the number of optimization cycles (left) and the CPU cost (right). Each cycle comprises the convergence of the flow, AV and DD (if necessary) equations and the update of the shape. In each new cycle, the iterative solution of each set of equations starts from the converged solution of the previous cycle.

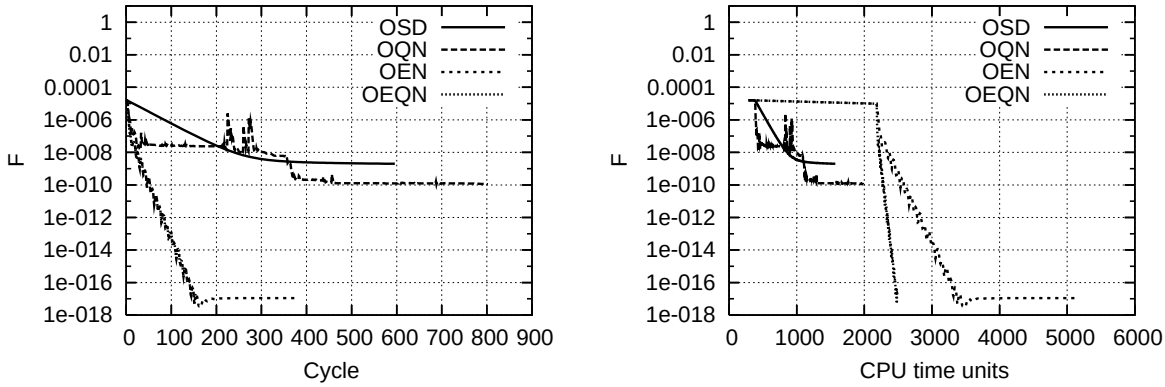


Figure 4: Inverse design of a cascade airfoil with 6 design variables: Reduction rates of F using the four one-shot variants (OSD, OQN, OEN and OEQN) in terms of the number of optimization cycles (left) and CPU cost (right).

The initial and optimal pressure distributions along the airfoil sides are shown in fig. 2 together with the target pressure one and the corresponding airfoil contours. The optimal solution shown is the one obtained by the OEN method; nevertheless, all but the SSD variants converged to almost the same distributions.

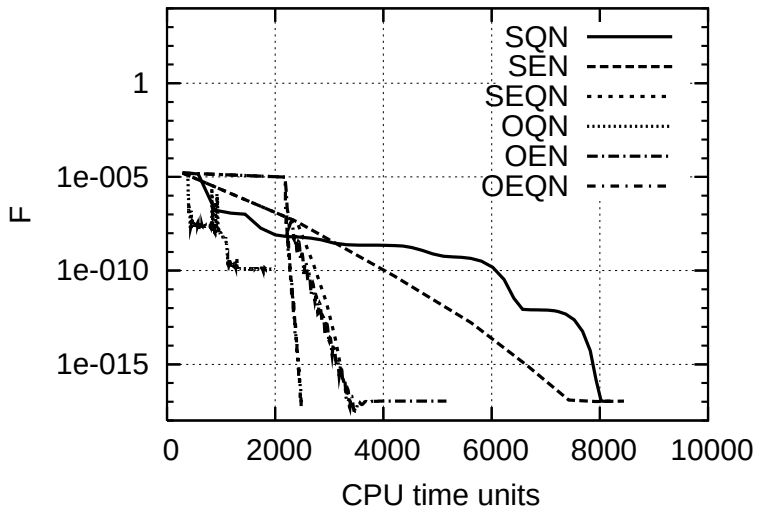


Figure 5: Inverse design of a cascade airfoil with 6 design variables: Comparison of the convergence histories of segregated (as in fig. 3) and one-shot variants (as in fig. 4).

Fig. 3 illustrates the convergence histories of the four segregated variants. In each optimization cycle, the flow, AV and DD equations (the latter is needed by the SEN method or the starting phase of SEQN) are fully converged. From fig. 3 (left), it can be deduced that both SEN and SEQN reduced the F value by almost 15 orders of magnitude

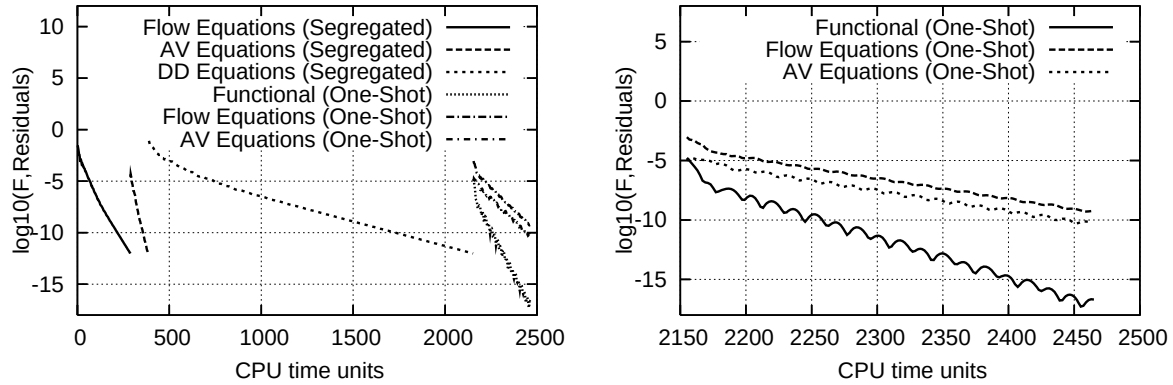


Figure 6: Inverse design of a cascade airfoil with 6 design variables: Reduction rates of the residuals of the flow, adjoint and DD equations solved separately during the first phase of the OEQN variant and the convergence histories of the flow and adjoint equation residuals as well as F during the second (one-shot) phase. A close-up view of the second phase curves is also shown (right).

within the first 8 cycles. However, in terms of CPU cost, SEQN noticeably outperforms SEN, requiring about half of its CPU cost to converge. Despite the low cost per cycle of SQN (it comprises the solution of the flow and adjoint equations only) which is by far the most efficient gradient based method, its overall convergence rate is even worse than that of SEN. Also, its convergence curve is oscillatory.

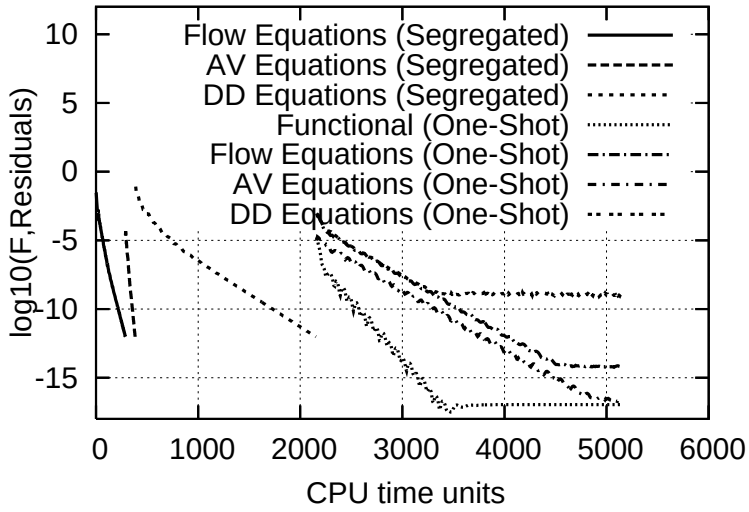


Figure 7: Inverse design of a cascade airfoil with 6 design variables: Reduction rates of the residuals of the flow, AV and DD equations solved separately during the first optimization cycle and the simultaneous reduction rates of flow, adjoint and DD residuals and the functional value during the one-shot part of the OEQN algorithm.

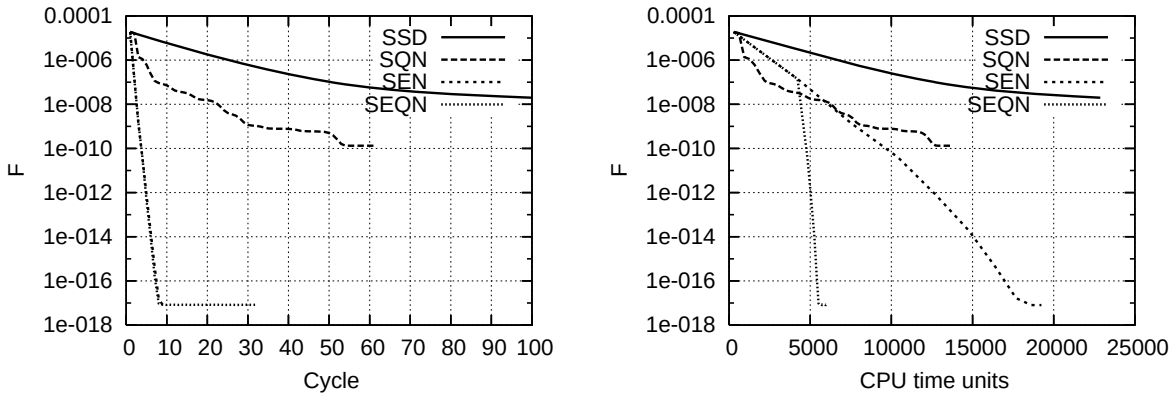


Figure 8: Inverse design of a cascade airfoil with 12 design variables: Reduction rates of the objective function value using the four segregated variants (SSD, SQN, SEN and SEQN), in terms of the number of optimization cycles (left) and CPU cost (right).

The convergence rates of the four one-shot variants, shown in fig. 4, lead to the same conclusions. The superiority of OEN and, especially, that of OEQN is obvious. From the number of cycles required to converge, it is concluded that there are good reasons to compute the exact Hessian. So, for instance, OQN fails to converge to the global optimum and performs slightly better than OSD. However, from the same figure (right) it becomes clear that the best trade-off is to compute the exact Hessian in the first cycle and, then, update it using the non-costly BFGS formula. Similar to their segregated variants, the high CPU cost for computing the Hessian is compensated by the efficient second-order search direction information obtained by solving the Newton equation. The proposed OEQN variant profits from the use of an "almost accurate" Hessian matrix (since BFGS is initiated with the exact Hessian, instead of an arbitrary matrix) and the economy achieved if this is to be computed only once.

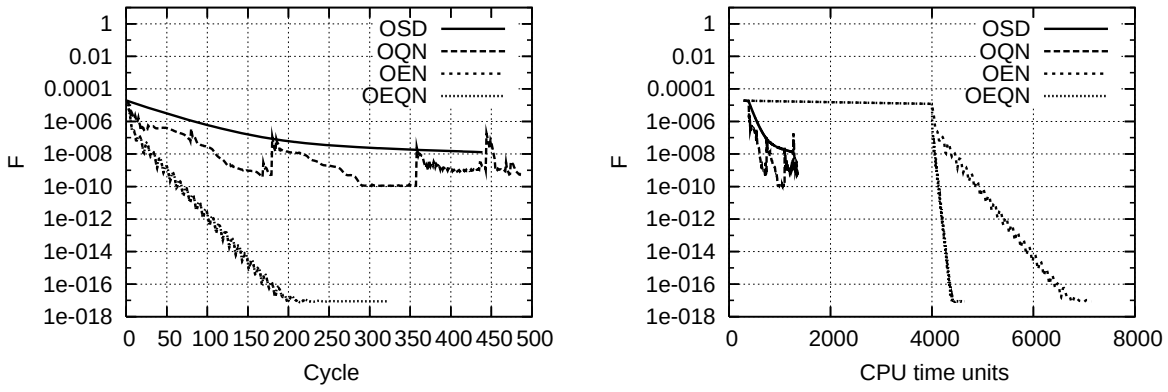


Figure 9: Inverse design of a cascade airfoil with 12 design variables: Reduction rates of the objective function value using the four one-shot variants (OSD, OQN, OEN and OEQN), in terms of the number of optimization cycles (left) and CPU cost (right).

The methods presented separately in figs. 3 and 4 are compared with each other in fig. 5. Note that each one-shot variant is quite faster than the corresponding segregated scheme. This conclusion can also be drawn from the comparison of the SSD and OSD convergence plots which, however, are not included in fig. 5, to keep it as readable as possible. It can easily be concluded that the most efficient method is OEQN. Within no more than eight times the cost of solving the flow equations, OEQN reduces F by 12 orders of magnitude!

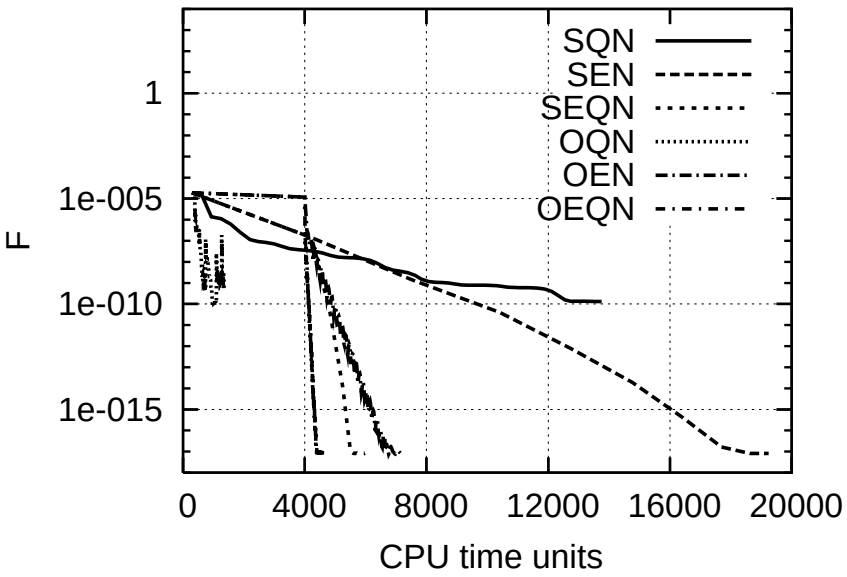


Figure 10: Inverse design of a cascade airfoil with 12 design variables: Comparison of the convergence histories of segregated (fig. 8) and one-shot variants (fig. 9).

The analysis of the optimization cycles required for the convergence of the most efficient variant (OEQN) is shown in fig. 6. In this figure (left), the first three curves, from left to right, show the convergence rates of the residuals of the flow, AV and DD equations, solved separately (the one after the other) during the first phase. These ∼2150 CPU time units correspond to the first, almost horizontal part of the curve marked with SEQN, in fig. 5. The next three curves correspond to the simultaneous solution of the flow and AV equations and the Newton update formula. It is obvious that the greater part of the required CPU cost is associated with the first cycle, due to the solution of the DD equations.

A similar analysis of the convergence history of the OEN variant is shown in fig. 7. The only difference is that the solution of the DD equations during the one-shot (second) phase increases the total CPU cost.

The same case was also examined using 12 (instead of 6) design variables. Figs. 8, 9 and 10 show the same plots as before and can be used to rank sort the proposed variants

in terms of computational cost. It is encouraging that, even if the number of design variables is twice as many as in the previous case, the same conclusions can be drawn. The superiority of the OEQN approach is evident. The gain from approximately updating (BFGS) the Hessian matrix in all but the first cycle, instead of re-computing it exactly, is obviously higher, since twice as many DD equations must now be solved. It can also be seen that SQN has difficulties in sufficiently lowering the value of F whereas SEN has stable but costly convergence rates. A compromise between these two approaches gave practically birth to the SEQN approach, which outperforms both of them. Compared to SEQN, its one-shot variant (OEQN) is capable of further reducing the CPU cost.

5 CONCLUSIONS

This paper presented and tested two one-shot optimization algorithms, namely (a) the Newton method with the exactly computed Hessian matrix in each cycle and (b) the exactly initialized, quasi-Newton approach. Six other variants have also been programmed and tested. Numerical experiments on the solution of a cascade airfoil inverse design problem showed that:

- In aerodynamic optimization, the use of second-order sensitivities is desirable, since it dramatically reduces the number of required optimization cycles. However, with many design variables, the CPU cost per cycle of the exact Newton method, being proportional to their number, becomes almost prohibitive and, certainly, much higher than that of pure gradient-based methods.

- This problem can be overcome by means of a hybrid approach in which the exact Hessian is computed only once in the beginning and, from this point on, this is merely updated using approximate formulas, such as BFGS. Numerical tests have shown that this approach outperforms both exact and quasi-Newton methods, irrespective of the number of design variables.

- A further noticeable improvement is achieved by applying a one-shot algorithm in which the state and optimization equations are solved simultaneously. In all cases, the one-shot method outperforms its conventional counterpart.

- The use of the one-shot, exactly initialized, quasi-Newton algorithm reduces dramatically the total CPU cost. The cost for reducing the inverse design functional value by almost ten orders of magnitude (being much more than what engineers desire in real-world applications) was found to be less than ten times the cost of solving the flow equations (for the numbers of design variables used in our studies).

Acknowledgement

The first author was supported by a grant from the Basic Research Program (PEBE, 2008) of the National Technical University of Athens.

REFERENCES

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

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

[3] A. Jameson and J. Reuther, Control theory based airfoil design using the Euler equations, *AIAA* Paper 94-4272 (1994).

[4] S. Taasan, G. Kuruvila and M.D. Salas, Aerodynamic design and optimization in one-shot, *AIAA* Paper 91-005 (1992).

[5] G. Kuruvila, S. Taasan and M.D. Salas, Airfoil design and optimization by the one-shot method, *AIAA* Paper 95-0478 (1995).

[6] S.B. Hazra, An efficient method for aerodynamic shape optimization, *AIAA* Paper 2004-4628 (2004).

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

[8] S.B. Hazra and V. Schulz, Simultaneous pseudo-timestepping for aerodynamic shape optimization problems with state constraints, *SIAM J. Sci. Comput*, to appear (2006).

[9] C. Held and A. Dervieux, One-shot airfoil optimisation without adjoint, *Computers and Fluids*, **31**, 1015–1049 (2002).

[10] A. Dadone and B. Grossman, Progressive optimization of inverse fluid dynamic design problems, *Computers and Fluids*, **29**, 1–32 (2000).

[11] A. Dadone and B. Grossman, Fast convergence of inviscid fluid dynamic design problems, *Computers and Fluids*, **32**, 607–627 (2003).

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

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

[14] G.W. Hou and J. Sheen, Numerical methods for second-order shape sensitivity analysis with applications to heat conduction problems, *International Journal for Numerical Methods in Engineering*, **36**, 417–435 (1993).

[15] F.X. Le Dimet, I.M. Navon and D.N. Daescu, Second-order information in data assimilation, *Monthly Weather Review*, **130(3)**, 629–648 (2002).

[16] F. Veerse, D. Auroux and M. Fisher, Limited-memory BFGS Diagonal Preconditioners for a data assimilation problem in meteorology, *Optimization and Engineering*, **1**, 323–339 (2000).

[17] D.N. Daescu and I.M. Navon, An analysis of a hybrid optimization method for variational data assimilation, *International Journal of Computational Fluid Dynamics*, **17(4)**, 299–306 (2003).

[18] E. Arian and S. Taasan, Analysis of the Hessian for aerodynamic optimization: Inviscid flow, *Computers and Fluids*, **28(7)**, 853–877 (1999).

[19] D.I. Papadimitriou and K.C. Giannakoglou, Computation of the Hessian matrix in aerodynamic inverse design using continuous adjoint formulations, *Computers and Fluids*, **37(8)**, 1029–1039 (2008).

[20] D.I. Papadimitriou and K.C. Giannakoglou, Direct, adjoint and mixed approaches for the computation of Hessian in airfoil design problems, *International Journal for Numerical Methods in Fluids*, **56**, 1929–1943 (2008).

[21] D.I. Papadimitriou and K.C. Giannakoglou, The continuous direct-adjoint approach for second-order sensitivities in viscous aerodynamic inverse design problems, *Computers and Fluids*, **38**, 1539–1548 (2008).

[22] D.I. Papadimitriou and K.C. Giannakoglou, Aerodynamic shape optimization using adjoint and direct approaches. *Archives of Computational Methods in Engineering*, **15(4)**, 447–488 (2008).

[23] D.P. Bertsekas, Nonlinear Programming, *Athena Scientific*, 2nd edition (1999).