AIAA JOURNAL
Vol. 42, No. 7, July 2004

Variable-Fidelity Aerodynamic Optimization for Turbulent Flows Using a Discrete Adjoint Formulation

Alan Le Moigne* and Ning Qin†
University of Sheffield, Sheffield, England S1 3JD, United Kingdom

An aerodynamic shape optimization methodology based on a discrete adjoint solver for Navier–Stokes flows is described. The flow solver at the heart of this optimization process is a Reynolds-averaged Navier–Stokes code for multiblock structured grids. It uses Osher's approximate Riemann solver and the algebraic turbulence model of Baldwin–Lomax. A corresponding discrete Navier–Stokes adjoint solver is derived analytically. It has to calculate accurately the Jacobian, including the effect of the turbulence modeling. The shape deformations are parameterized by the use of a Bézier–Bernstein formulation. The optimization is gradient-based and employs the variable-fidelity optimization method of Alexandrov et al. that combines low- (Euler equations on a coarse grid) and high-fidelity (Navier–Stokes equations on a fine grid) models for better efficiency. The accuracy of the adjoint solver is verified through comparison with finite difference. An airfoil drag minimization problem and the three-dimensional Navier–Stokes optimization of the ONERA M6 wing are presented.

Nomenclature

$B_{k,N}$ = Bernstein polynomial
C_{wake} = wake constant in the Baldwin–Lomax model
c_1 = constant in the variable-fidelity method
c_2 = constant in the variable-fidelity method
F = function in the Baldwin–Lomax model
$\boldsymbol{F}$ = flux vector
$F_{\max}$ = maximum value of the function F across the boundary layer in the Baldwin–Lomax model
F_{wake} = function in the Baldwin–Lomax model
g_i = inequality constraint
h_j = equality constraint
NDV = number of design variables
$\boldsymbol{P}$ = vector of primitive variables, $(\rho, u, v, w, p)^t$
P_{yk} = y coordinate of a Bézier control point
$\boldsymbol{Q}$ = vector of conservative variables, $(\rho, \rho u, \rho v, \rho w, \rho E)^t$
$\boldsymbol{R}$ = residual vector
r = parameter measuring the performance of the low-fidelity model in the variable-fidelity method
r_1 = constant in the variable-fidelity method
r_2 = constant in the variable-fidelity method
$\boldsymbol{s}$ = vector of variables in the low-fidelity optimization of the variable-fidelity method
t = time
U_{diff} = maximum difference in velocity amplitude across the boundary layer in the Baldwin–Lomax model
u = normalized computational arclength along a curve
$\boldsymbol{X}$ = vector of grid points
x, y, z = Cartesian coordinates
$y_{\max}$ = value of y_n for which $F(y_n)$ is equal to $F_{\max}$ in the Baldwin–Lomax model
$\mathcal{B}$ = coefficient in the beta-correction technique of the variable-fidelity method
$\mathcal{B}_q$ = linear approximation of $\mathcal{B}$
$\boldsymbol{\beta}$ = vector of design variables
Δ = increment
$\Delta_{\max}$ = upper bound on the radius of the trust region in the variable-fidelity method
Δ_q = bound on the radius of the trust region at iteration q in the variable-fidelity method
δy = shape increment in the y direction
$\boldsymbol{\lambda}$ = adjoint vector, $(\lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5)^t$ in three dimensions
μ_t = turbulent viscosity
Φ = merit function composed of the objective function augmented with a penalty term

Subscripts

hi = high-fidelity model
L = relative to the left-hand side of the interface in the Riemann solver
lo = low-fidelity model
R = relative to the right-hand side of the interface in the Riemann solver

Superscripts

l = lower bound
t = tranpose operator
u = upper bound
$*$ = converged flow solution
$\sim$ = approximate Jacobian
$\sim$ = indicates a corrected model in the variable-fidelity method

Introduction

AERODYNAMIC optimization[1] has been gaining importance in the aeronautical industry and has become a useful tool for aerodynamic design of wings and aircraft. A popular choice to perform such tasks is to use an efficient gradient-based optimizer, along with an adjoint solver, coupled to an existing computational fluid dynamics (CFD) flow solver, to calculate efficiently and accurately the required sensitivity derivatives. Jameson[2] popularized this technique in aerodynamic optimization after some theoretical work by Pironneau[3,4] and some early computational implementations.[5,6] The method Reuther et al.[7,8] and Jameson et al.[9] developed[7] is known as the continuous adjoint method and is now widely employed.[10–12] An alternative method was proposed later, where the flow equations are first discretized and then differentiated, known as the discrete adjoint formulation.[13–18]

In the present work, a discrete adjoint method for three-dimensional turbulent flows on structured multiblock grids is

---

Received 25 April 2003; revision received 16 December 2003; accepted for publication 20 February 2004. Copyright © 2004 by Alan Le Moigne and Ning Qin. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/04 $10.00 in correspondence with the CCC.

*Research Associate, Department of Mechanical Engineering, Mappin Street; a.lemoigne@sheffield.ac.uk. Member AIAA.

†Professor, Department of Mechanical Engineering, Mappin Street; n.qin@sheffield.ac.uk. Senior Member AIAA.

developed. The discrete approach is chosen rather than the continuous one for two main reasons: in the continuous method, there is some uncertainty surrounding the theoretical need for a boundary condition at the shock that does not appear in the discrete version[19]; the discrete approach can reuse the flow solver Jacobian and iterative method, making it conceptually straightforward. Other discrete adjoint solvers that linearize accurately the turbulence viscosity exist, but they use either the Spalart–Allmaras[20] turbulence model[13,15,21,22] or two-equation models.[14,23] Here, the algebraic model of Baldwin–Lomax[24] is employed. The simpler nature of this turbulence model should make it more efficient for optimization, especially in three dimensions, but its linearization is also more difficult because it is a full stencil model.[25]

However, having an efficient adjoint solver to calculate the sensitivity derivatives is not sufficient to get an efficient optimization method, especially for three-dimensional industrial problems involving turbulent flows. Direct application of a gradient-based optimization algorithm such as a quasi-Newton method to solve this kind of problem proves to be computationally demanding. The algorithm indeed requires successive evaluations of the objective function and of its gradient, which implies several solutions of the flow and adjoint equations that are relatively costly on a three-dimensional fine grid. To improve the efficiency of the optimization, we follow the work of Alexandrov et al.,[26–28] who propose to employ a variable-fidelity optimization [also known as AMMO (approximation and model management optimization)] method that combines low- and high-fidelity models. The idea is to carry out most of the optimization on a low-fidelity cheap-to-run model that globally represents the behavior of the high-fidelity model and to correct from time to time the low-fidelity model so that it better represents the high-fidelity one. In Refs. 26 and 28, Alexandrov tested their method by using the Euler equations on a coarse grid for the low-fidelity model, as well as on a fine grid for the high-fidelity model. We follow the methodology of Ref. 27 where they changed both the physics and the grid refinement with the Euler equations on a reasonably coarse Euler mesh for the low-fidelity model and the turbulent Navier–Stokes equations on a Navier–Stokes mesh for the high-fidelity model. They obtain up to a fivefold improvement in efficiency compared to direct high-fidelity optimization methods.

This paper first presents the CFD flow solver used in the optimization process and then the adjoint solver coupled to it. Some details are given about the implementation in this code of the matrix–vector product between the exact Jacobian and the adjoint vector. The shape parameterization is then described before the variable-fidelity method of Alexandrov et al. is detailed. Comparison with a finite difference method will show that the adjoint solver provides accurate sensitivity derivatives. The optimization process is applied to the drag minimization of a two-dimensional airfoil and of the ONERA M6 wing.[29] The two-dimensional optimization is also performed with a direct high-fidelity approach to assess the performance of the variable-fidelity method.

Flow Equations

The governing equations are the three-dimensional Reynolds-averaged Navier–Stokes equations discretized in space with a cell-centered finite volume formulation on structured multiblock grids. The convective flux calculation follows Osher's approximate Riemann solver (see Refs. 30 and 31) with a MUSCL scheme[32–34] for higher-order accuracy, the viscous fluxes are calculated using a central discretization. The algebraic turbulence model of Baldwin–Lomax[24] is employed for turbulent flows. The boundary conditions are applied through the use of two layers of halo cells.

The temporal discretization follows an implicit formulation.[35] After linearization of the residual, the equations to solve iteratively are

$$\left[\frac{1}{\Delta t}\frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{P}} + \frac{\partial \widetilde{\boldsymbol{R}}(\boldsymbol{Q}^n)}{\partial \boldsymbol{P}}\right]^n \Delta \boldsymbol{P} = -\boldsymbol{R}(\boldsymbol{Q}^n) \tag{1}$$

with

$$\boldsymbol{P}^{n+1} = \boldsymbol{P}^n + {}^n\Delta \boldsymbol{P} \tag{2}$$

Note that these equations are solved for primitive variables $\boldsymbol{P}$ rather than conservative ones. Local time stepping is employed and the left-hand side (LHS) Jacobian of Eq. (1) is approximate: It is only first-order accurate in space for the convective fluxes, resulting in a seven-block-diagonal matrix. (See Refs. 36 and 37 for details about the linearization of the convective terms and, in particular, of Osher's scheme.) For simplicity, this structure is maintained for the diffusive terms at the cost of further approximations. The linearization of the turbulent viscosity is also neglected. Despite these approximations, the boundary conditions are accurately linearized. Equation (1) is solved approximately by the use of the block incomplete lower-upper decomposition with no fill-in or BILU(0).

Discrete Adjoint Equations

The discrete adjoint equations are introduced to calculate the sensitivity derivative of the objective function that can be written as

$$F = F[\boldsymbol{P}^*(\boldsymbol{\beta}), \boldsymbol{X}(\boldsymbol{\beta}), \boldsymbol{\beta}] \tag{3}$$

The sensitivity derivative of the objective function (3) is

$$\frac{\mathrm{d}F}{\mathrm{d}\beta_k} = \left(\frac{\partial F}{\partial \boldsymbol{P}^*}\right)^t \frac{\mathrm{d}\boldsymbol{P}^*}{\mathrm{d}\beta_k} + \left(\frac{\partial F}{\partial \boldsymbol{X}}\right)^t \frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\beta_k} + \frac{\partial F}{\partial \beta_k} \tag{4}$$

In Eq. (4), the only term difficult to compute is $\mathrm{d}\boldsymbol{P}^*/\mathrm{d}\beta_k$. Its calculation is avoided by the introduction of an adjoint vector, as follows.

Let us write the discrete residual of the steady-state equations as

$$\boldsymbol{R}[\boldsymbol{P}^*(\boldsymbol{\beta}), \boldsymbol{X}(\boldsymbol{\beta}), \boldsymbol{\beta}] = \boldsymbol{0} \tag{5}$$

The derivative of $\boldsymbol{R}$ to the design variable β_k gives

$$\frac{\mathrm{d}\boldsymbol{R}}{\mathrm{d}\beta_k} = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{P}^*}\frac{\mathrm{d}\boldsymbol{P}^*}{\mathrm{d}\beta_k} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\beta_k} + \frac{\partial \boldsymbol{R}}{\partial \beta_k} = \boldsymbol{0} \tag{6}$$

Equation (6), multiplied by an adjoint vector $\boldsymbol{\lambda}$ is added to Eq. (4), which becomes, after rearrangement,

$$\frac{\mathrm{d}F}{\mathrm{d}\beta_k} = \left[\left(\frac{\partial F}{\partial \boldsymbol{P}^*}\right)^t + \boldsymbol{\lambda}^t \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{P}^*}\right]\frac{\mathrm{d}\boldsymbol{P}^*}{\mathrm{d}\beta_k} + \left[\left(\frac{\partial F}{\partial \boldsymbol{X}}\right)^t + \boldsymbol{\lambda}^t \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}}\right]\frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\beta_k} + \frac{\partial F}{\partial \beta_k} + \boldsymbol{\lambda}^t \frac{\partial \boldsymbol{R}}{\partial \beta_k} \tag{7}$$

To avoid having to calculate $\mathrm{d}\boldsymbol{P}^*/\mathrm{d}\beta_k$ for each design variable, the term multiplying this quantity is set to zero to give the adjoint equation

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{P}}\right)^t \boldsymbol{\lambda} = -\frac{\partial F}{\partial \boldsymbol{P}} \tag{8}$$

This latter equation is written in the incremental iterative form[38–40] with the addition of a fictitious time term

$$\left[\frac{1}{\Delta t}\frac{\partial \boldsymbol{Q}}{\partial \boldsymbol{P}} + \frac{\partial \tilde{\boldsymbol{R}}(\boldsymbol{Q}^*)}{\partial \boldsymbol{P}}\right]^{t\,n} \Delta \boldsymbol{\lambda} = -\left\{\frac{\partial F}{\partial \boldsymbol{P}} + \left[\frac{\partial \boldsymbol{R}(\boldsymbol{Q}^*)}{\partial \boldsymbol{P}}\right]^t \boldsymbol{\lambda}^n\right\} \tag{9}$$

with

$$\boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^n + {}^n\Delta \boldsymbol{\lambda} \tag{10}$$

In this way, the LHS of Eq. (9) is the same as in Eq. (1), before transposition. To solve Eq. (9), the same BILU(0) as for the flow equations is employed for simplicity. Equation (9) is solved iteratively until convergence.

Although approximations are allowed for the LHS Jacobian of Eq. (9), the right-hand side (RHS) Jacobian has to be accurate. Note that this implies a full, rather than sparse, matrix on the RHS due to the full stencil of the algebraic turbulence modeling. To store such a large and dense matrix would require a very large computer memory. To avoid this, the full Jacobian is not stored, but each of its elements is calculated at every iteration and is multiplied by

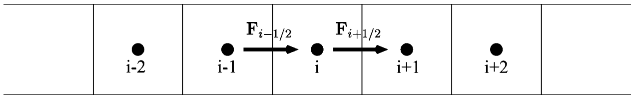


Fig. 1 Higher-order inviscid fluxes.

the corresponding component of the adjoint vector. This prevents a memory problem, but it is at the cost of the recalculation of the terms constituting the Jacobian at every iteration and, thus, impacts on the overall computing time. A summary of the calculation of the RHS multiplication with an accurate Jacobian is given next. This only considers what happens for interior points; an accurate treatment of the boundary conditions is needed close to a block boundary.[25]

Let us denote by $\mathbf{rhs}_{i,j,k}$ the term $[\partial \boldsymbol{R}(\boldsymbol{Q}^*)/\partial \boldsymbol{P}]^t \boldsymbol{\lambda}^n$ corresponding to the computational cell i, j, k. With consideration given to the convective fluxes in three dimensions,

$$\mathbf{rhs}_{i,j,k} = \sum_{l=i-2}^{i+2} \left(\frac{\partial \boldsymbol{R}_{l,j,k}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{l,j,k} + \sum_{l=j-2}^{j+2} \left(\frac{\partial \boldsymbol{R}_{i,l,k}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,l,k}$$

$$+ \sum_{l=k-2}^{k+2} \left(\frac{\partial \boldsymbol{R}_{i,j,l}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j,l} - 2 \left(\frac{\partial \boldsymbol{R}_{i,j,k}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j,k} \tag{11}$$

To further detail the derivation, we consider only the one-dimensional problem of Fig. 1. The use of the MUSCL scheme[32–34] makes each flux depend on four cells; hence,

$$\frac{\partial \boldsymbol{R}_i}{\partial \boldsymbol{P}_i} = \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_i} - \frac{\partial \boldsymbol{F}_{i-\frac{1}{2}}}{\partial \boldsymbol{P}_i}, \qquad \frac{\partial \boldsymbol{R}_i}{\partial \boldsymbol{P}_{i-1}} = \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{i-1}} - \frac{\partial \boldsymbol{F}_{i-\frac{1}{2}}}{\partial \boldsymbol{P}_{i-1}}$$

$$\frac{\partial \boldsymbol{R}_i}{\partial \boldsymbol{P}_{i-2}} = -\frac{\partial \boldsymbol{F}_{i-\frac{1}{2}}}{\partial \boldsymbol{P}_{i-2}}, \qquad \frac{\partial \boldsymbol{R}_i}{\partial \boldsymbol{P}_{i+1}} = \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{i+1}} - \frac{\partial \boldsymbol{F}_{i-\frac{1}{2}}}{\partial \boldsymbol{P}_{i+1}}$$

$$\frac{\partial \boldsymbol{R}_i}{\partial \boldsymbol{P}_{i+2}} = \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{i+2}} \tag{12}$$

The differentiation $\partial \boldsymbol{F}_{i+1/2}/\partial \boldsymbol{P}_i$, for example, involves the linearization of Osher's Riemann solver (see Refs. 36 and 37) and of the MUSCL scheme in addition to the transformation between conservative and primitive variables because we apply Osher's scheme to conservative variables, whereas the MUSCL scheme is applied to primitive variables. (The MUSCL scheme can be easily differentiated because our slope limiter is differentiable.) Hence, for example,

$$\frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_i} = \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{Q}_R} \frac{\partial \boldsymbol{Q}_R}{\partial \boldsymbol{P}_R} \frac{\partial \boldsymbol{P}_R}{\partial \boldsymbol{P}_i} + \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{Q}_L} \frac{\partial \boldsymbol{Q}_L}{\partial \boldsymbol{P}_L} \frac{\partial \boldsymbol{P}_L}{\partial \boldsymbol{P}_i} \tag{13}$$

Note that except for the terms related to the differentiation of the MUSCL scheme, most of the other terms involved in the calculation of the convective Jacobian are readily available from the first-order Jacobian employed in the RHS of Eqs. (1) and (9).

For the viscous fluxes in three dimensions, under the assumption that the turbulent viscosity is frozen, we have

$$\mathbf{rhs}_{i,j,k} = \sum_{m=j-1}^{j+1} \sum_{l=i-1}^{i+1} \left(\frac{\partial \boldsymbol{R}_{l,m,k}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{l,m,k}$$

$$+ \sum_{m=k-1}^{k+1} \sum_{l=i-1}^{i+1} \left(\frac{\partial \boldsymbol{R}_{l,j,m}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{l,j,m} - \sum_{l=i-1}^{i+1} \left(\frac{\partial \boldsymbol{R}_{l,j,k}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{l,j,k}$$

$$+ \left(\frac{\partial \boldsymbol{R}_{i,j-1,k+1}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j-1,k+1} + \left(\frac{\partial \boldsymbol{R}_{i,j+1,k+1}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j+1,k+1}$$

$$+ \left(\frac{\partial \boldsymbol{R}_{i,j-1,k-1}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j-1,k-1} + \left(\frac{\partial \boldsymbol{R}_{i,j+1,k-1}}{\partial \boldsymbol{P}_{i,j,k}} \right)^t \cdot \boldsymbol{\lambda}_{i,j+1,k-1} \tag{14}$$

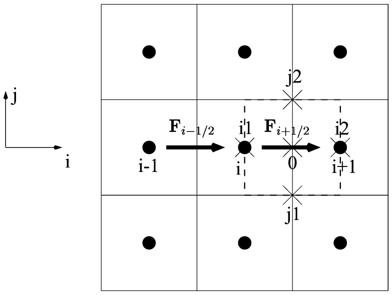


Fig. 2 Viscous laminar fluxes.

To explain the viscous derivation in one dimension, we introduce Fig. 2. Because of the use of Gauss's theorem for the central discretization of the viscous fluxes, the flux $\boldsymbol{F}_{i+1/2}$ depends on $\boldsymbol{P}_0, \boldsymbol{P}_{i1}$, $\boldsymbol{P}_{i2}, \boldsymbol{P}_{j1}, \boldsymbol{P}_{j2}, \boldsymbol{P}_{k1}$, and $\boldsymbol{P}_{k2}$. Note that $k1$ and $k2$ are in the third dimension not represented in Fig. 2. These latter terms are calculated as

$$\boldsymbol{P}_0 = \tfrac{1}{2}(\boldsymbol{P}_{i,j,k} + \boldsymbol{P}_{i+1,j,k}), \qquad \boldsymbol{P}_{i1} = \boldsymbol{P}_{i,j,k}$$

$$\boldsymbol{P}_{j2} = \tfrac{1}{4}(\boldsymbol{P}_{i,j,k} + \boldsymbol{P}_{i+1,j,k} + \boldsymbol{P}_{i,j+1,k} + \boldsymbol{P}_{i+1,j+1,k}), \ldots \tag{15}$$

Hence,

$$\frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{i,j,k}} = \frac{1}{2} \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_0} + \frac{1}{4} \left(\frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{j1}} + \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{j2}} + \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{k1}} + \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{k2}} \right)$$

$$+ \frac{\partial \boldsymbol{F}_{i+\frac{1}{2}}}{\partial \boldsymbol{P}_{i1}} \tag{16}$$

The linearization of the turbulent viscosity adds another term to the viscous Jacobian that can be decomposed as

$$\frac{\partial \boldsymbol{F}^v}{\partial \boldsymbol{P}} = \left. \frac{\partial \boldsymbol{F}^v}{\partial \boldsymbol{P}} \right|_{\mu_t = \text{frozen}} + \frac{\partial \boldsymbol{F}^v}{\partial \mu_t} \frac{\partial \mu_t}{\partial \boldsymbol{P}} \tag{17}$$

the first term on the RHS corresponding to the preceding derivation. In the Baldwin–Lomax model,[24] the turbulent viscosity is calculated along rays, and its value depends on all of the cells along these rays because the boundary between the inner and outer eddy viscosity regions is not known in advance and neither is the location of $F_{\max}$ for the outer turbulent viscosity. As an approximation, when these positions are known for a particular ray, μ_t depends numerically on 17 cells,[25] the position of which varies from ray to ray. This makes it possible to linearize μ_t. This is a straight differentiation of the way the eddy viscosity is calculated and is not presented in detail here.

Problems were encountered concerning the maximum and minimum functions used for the outer eddy viscosity. An approximation is made here for practical convenience

$$\frac{\partial F_{\max}}{\partial \boldsymbol{P}_l} \approx \left. \frac{\partial F}{\partial \boldsymbol{P}_l} \right|_{\text{where } F = F_{\max}} \tag{18}$$

In the calculation of the wake function

$$F_{\text{wake}} = \min\left(y_{\max} F_{\max}, C_{\text{wake}} y_{\max} U_{\text{diff}}^2 \big/ F_{\max} \right) \tag{19}$$

the term $y_{\max} F_{\max}$ comes from Cebeci–Smith's model for wall-bounded boundary layers (see Ref. 41), whereas the term $C_{\text{wake}} y_{\max} U_{\text{diff}}^2 / F_{\max}$ comes from Prandtl's model for free shear flows and should be used in the airfoil wake. Because, in our case, μ_t is always calculated above the surface of an airfoil or wing, the definition of F_{wake} can be modified as

$$F_{\text{wake}} = y_{\max} F_{\max} \tag{20}$$

which should not change the accuracy of the model for attached boundary-layer flows. This simplification avoids the problem with the nondifferentiability of the min function in calculating F_{wake}. To be consistent, this modification was also included in the flow solver. These approximations and simplifications have proven to work well through numerical tests. The overall sensitivity derivatives calculated by the adjoint solver agreed quite well with derivatives calculated by finite difference.

Once the adjoint equation (9) has been solved for the adjoint vector, the sensitivity equation (7) can be simplified to become

$$\frac{\mathrm{d}F}{\mathrm{d}\beta_k} = \left(\frac{\partial F}{\partial \boldsymbol{X}}\right)^t \frac{d\boldsymbol{X}}{d\beta_k} + \boldsymbol{\lambda}^t \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{X}} \frac{\mathrm{d}\boldsymbol{X}}{\mathrm{d}\beta_k} \tag{21}$$

Note that the last two terms of Eq. (7) have been eliminated because, for a pure aerodynamic shape optimization, the design variables β_k only influence the residual vector $\boldsymbol{R}$ and the objective function through the computational grid variations. (The design variable relative to the increment of angle of incidence produces, in this work, a rotation of the whole computational grid, and, thus, this simplification is valid.) Except for the adjoint vector $\boldsymbol{\lambda}$, the other terms in Eq. (21) are relatively easy to calculate by application of the chain rule of differentiation to the flow solver routines that compute the residual and the objective function. In the present case, these routines have been hand differentiated.

Surface Parameterization and Grid Update

A Bézier–Bernstein parameterization is employed to represent the shape to be modified by the optimization process. The parameterization is applied to a perturbation that is added to the original shape. For a two-dimensional airfoil, deformations are limited to the vertical coordinates y

$$y_{\text{current}} = y_{\text{initial}} + \delta y \tag{22}$$

where

$$\delta y = \sum_{k=0}^{N} B_{k,N}(u) P_{yk} \tag{23}$$

with the Bernstein polynomials

$$B_{k,N}(u) = \{N!/[k!(N-k)!]\}u^k(1-u)^{N-k} \tag{24}$$

The design variables are the coefficients P_{yk}. The normalized arc length u is rather arbitrary for deformations in only one direction. It is taken as[16]

$$u = \sqrt{x} \tag{25}$$

where x is the nondimensionalized chordwise position of the point of ordinate y_{initial}. This choice for u concentrates design changes to the leading-edge region of the airfoil where x is small.

This parameterization is applied independently to the upper and lower surface curves of the airfoil whereas the leading- and trailing-edge points are kept fixed. An additional design variable controls the change in angle of incidence of the whole section. This parameterization is extended to three dimensions to represent a wing shape: The wing is divided into a series of master sections that are connected linearly, each of these sections being free to deform according to the two-dimensional parameterization presented earlier.

Each time the geometry is modified, the CFD grid around it has to be changed. This is performed through an analytical grid deformation technique[18] that propagates smoothly the geometry change through the volume grid. This grid deformation technique is hand-differentiated to provide the grid sensitivities $\mathrm{d}\boldsymbol{X}/\mathrm{d}\beta_k$ needed in Eq. (21).

Variable-Fidelity Method

The variable-fidelity method of Alexandrov et al.[26–28] is based on two concepts: the trust-region and the corrected low-fidelity model. The trust-region concept is similar to move limits in conventional optimization. This means that the real high-fidelity model is successively approximated by a surrogate model and that this approximation is assumed to be valid inside successive delimited regions of the design space. If, during the preceding iteration, the surrogate model approximated very well the high-fidelity model, this trust region is extended for the new iteration, whereas if it performed badly, it is restricted. If the agreement between the surrogate and the real model was good, but not exceptional, the radius of the trust region is left unchanged.

The other component of the method is the corrected low-fidelity model. Let us assume that we have a high-fidelity model F_{hi} that is to be minimized and a low-fidelity model F_{lo}. At each iteration q, the corrected low-fidelity model $\tilde{F}_{\text{lo}}$ that is used in the optimization is required to satisfy first-order consistency with the high-fidelity model, that is,

$$\tilde{F}_{\text{lo}}(\boldsymbol{\beta}^q) = F_{\text{hi}}(\boldsymbol{\beta}^q), \qquad \nabla\tilde{F}_{\text{lo}}(\boldsymbol{\beta}^q) = \nabla F_{\text{hi}}(\boldsymbol{\beta}^q) \tag{26}$$

This ensures that the corrected low-fidelity model $\tilde{F}_{\text{lo}}$ behaves like F_{hi} in the neighborhood of $\boldsymbol{\beta}^q$. Alexandrov et al.[26–28] used a beta-correction technique to construct $\tilde{F}_{\text{lo}}$ with

$$\tilde{F}_{\text{lo}}(\boldsymbol{\beta}) = \mathcal{B}_q(\boldsymbol{\beta})F_{\text{lo}}(\boldsymbol{\beta}) \tag{27}$$

where $\mathcal{B}_q$ is the linear approximation

$$\mathcal{B}_q(\boldsymbol{\beta}) = \mathcal{B}(\boldsymbol{\beta}^q) + \nabla\mathcal{B}(\boldsymbol{\beta}^q)^t(\boldsymbol{\beta} - \boldsymbol{\beta}^q) \tag{28}$$

of $\mathcal{B}$ defined as

$$\mathcal{B}(\boldsymbol{\beta}) = F_{\text{hi}}(\boldsymbol{\beta})/F_{\text{lo}}(\boldsymbol{\beta}) \tag{29}$$

around the point $\boldsymbol{\beta}^q$. It is easy to check that such a corrected model satisfies the requirements (26).

The optimization problem to be solved can be presented mathematically as[42,43]

$$\begin{array}{llll} \underset{\beta}{\text{minimize}} & F(\boldsymbol{\beta}), & & \text{objective function} \\ \text{subject to} & g_i(\boldsymbol{\beta}) \le 0, & i = 1, l, & \text{inequality constraints} \\ & h_j(\boldsymbol{\beta}) = 0, & j = 1, m, & \text{equality constraints} \\ & \beta_k^l \le \beta_k \le \beta_k^u, & k = 1, \text{NDV}, & \text{side constraints} \end{array} \tag{30}$$

Let us now describe how the variable-fidelity method is applied to this problem. The algorithm employed in this work is the one corresponding to a sequential quadratic programming[42,43] (SQP) method in the work of Alexandrov et al.[28] A corrected low-fidelity model has to be constructed for the high-fidelity objective function F_{hi}, but also for the constraints $g_{i\,\text{hi}}$ and $h_{j\,\text{hi}}$ of the optimization problem (30). These are denoted $\tilde{F}_{\text{lo}}$, $\tilde{g}_{i\,\text{lo}}$, and $\tilde{h}_{j\,\text{lo}}$, respectively. A global merit function for the high-fidelity constrained optimization problem is needed to assess the overall performance of a design point after each low-fidelity optimization. Typically, this is a function composed of the high-fidelity objective function associated with a penalty term for the constraints. Let us denote this merit function by

$$\Phi_{\text{hi}} = f(F_{\text{hi}}, g_{i\,\text{hi}}, h_{j\,\text{hi}}) \tag{31}$$

A merit function for the corrected low-fidelity model is constructed in the same way:

$$\tilde{\Phi}_{\text{lo}} = f(\tilde{F}_{\text{lo}}, \tilde{g}_{i\,\text{lo}}, \tilde{h}_{j\,\text{lo}}) \tag{32}$$

Note that $\tilde{\Phi}_{\text{lo}}$ is not a low-fidelity corrected version of Φ_{hi}, despite the notation.

From an initial design point $\boldsymbol{\beta}^0$, the iteration q of the variable-fidelity method is as follows:

1) Calculate $F_{\text{hi}}(\boldsymbol{\beta}^q)$; $g_{i\,\text{hi}}(\boldsymbol{\beta}^q)$, $i = 1, l$; and $h_{j\,\text{hi}}(\boldsymbol{\beta}^q)$, $j = 1, m$, and $F_{\text{lo}}(\boldsymbol{\beta}^q)$; $g_{i\,\text{lo}}(\boldsymbol{\beta}^q)$, $i = 1, l$; and $h_{j\,\text{lo}}(\boldsymbol{\beta}^q)$, $j = 1, m$.

2) Build the corrected low-fidelity model $\tilde{F}_{\text{lo}}$ for the objective function around point $\boldsymbol{\beta}^q$. If they have not already been calculated, this requires the computation of $\nabla F_{\text{hi}}(\boldsymbol{\beta}^q)$ and $\nabla F_{\text{lo}}(\boldsymbol{\beta}^q)$.

3) Calculate $\nabla g_{i\,\text{hi}}(\boldsymbol{\beta}^q)$, $i=1,l$; and $\nabla h_{j\,\text{hi}}(\boldsymbol{\beta}^q)$, $j=1,m$ and $\nabla g_{i\,\text{lo}}(\boldsymbol{\beta}^q)$, $i=1,l$; and $\nabla h_{j\,\text{lo}}(\boldsymbol{\beta}^q)$, $j=1,m$, if they do not already exist.

4) Solve the following simplified low-fidelity optimization problem for $\boldsymbol{s}$:

$$\begin{aligned}
\underset{s}{\text{minimize}}\quad & \tilde{F}_{\text{lo}}(\boldsymbol{\beta}^q+\boldsymbol{s}) & \\
\text{subject to}\quad & \tilde{g}_{i\,\text{lo}}(\boldsymbol{\beta}^q)+\nabla\tilde{g}_{i\,\text{lo}}(\boldsymbol{\beta}^q)^t\boldsymbol{s}\le 0, & i=1,l \\
& \tilde{h}_{j\,\text{lo}}(\boldsymbol{\beta}^q)+\nabla\tilde{h}_{j\,\text{lo}}(\boldsymbol{\beta}^q)^t\boldsymbol{s}=0, & j=1,m \\
& \beta_k^l\le\beta_k^q+s_k\le\beta_k^u, & k=1,\ \text{NDV} \\
& \|\boldsymbol{s}\|\le\Delta_q &
\end{aligned}\tag{33}$$

It is a simplified problem because the constraints have been linearized. Note that due to the first-order consistency requirements, the linearized constraints are equivalent to

$$g_{i\,\text{hi}}(\boldsymbol{\beta}^q)+\nabla g_{i\,\text{hi}}(\boldsymbol{\beta}^q)^t\boldsymbol{s}\le 0,\qquad h_{j\,\text{hi}}(\boldsymbol{\beta}^q)+\nabla h_{j\,\text{hi}}(\boldsymbol{\beta}^q)^t\boldsymbol{s}=0\tag{34}$$

A constraint is added on the norm of $\boldsymbol{s}$. This is the result of the trust-region method that limits the application of the corrected low-fidelity models to the neighborhood of $\boldsymbol{\beta}^q$. We use the Euclidean L_2 norm for the norm of $\boldsymbol{s}$ because it is differentiable and is, hence, easy to incorporate as a constraint in the SQP algorithm that we chose to employ to solve the low-fidelity optimization problem.

5) When the optimum for $\boldsymbol{s}$ is found, assess the new design point at $\boldsymbol{\beta}^q+\boldsymbol{s}$ and the performance of the corrected low-fidelity model by the computation of

$$r=\frac{\Phi_{\text{hi}}(\boldsymbol{\beta}^q)-\Phi_{\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})}{\Phi_{\text{hi}}(\boldsymbol{\beta}^q)-\tilde{\Phi}_{\text{lo}}(\boldsymbol{\beta}^q+\boldsymbol{s})}\tag{35}$$

This requires the calculation of $F_{\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})$; $g_{i\,\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})$, $i=1,l$; and $h_{j\,\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})$ $j=1,m$, and $F_{\text{lo}}(\boldsymbol{\beta}^q+\boldsymbol{s})$; $g_{i\,\text{lo}}(\boldsymbol{\beta}^q+\boldsymbol{s})$, $i=1,l$; and $h_{j\,\text{lo}}(\boldsymbol{\beta}^q+\boldsymbol{s})$, $j=1,m$.

6) Update $\boldsymbol{\beta}^q$ and Δ_q. The update of the new design point is as follows:

$$\begin{aligned}
&\text{if } \Phi_{\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})>\Phi_{\text{hi}}(\boldsymbol{\beta}^q)\text{ then } \boldsymbol{\beta}^{q+1}=\boldsymbol{\beta}^q\\
&\text{else } \boldsymbol{\beta}^{q+1}=\boldsymbol{\beta}^q+\boldsymbol{s}
\end{aligned}$$

That is, if there is no improvement in the global merit function, the new design is discarded, and the process has to start again from the same initial point; otherwise, the new design point is kept. Our update for the trust-region radius, which is the critical part of the method, differs from the algorithm proposed in Ref. 28. It is as follows:

$$\begin{aligned}
&\text{if } \Phi_{\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})>\Phi_{\text{hi}}(\boldsymbol{\beta}^q)\text{ then } \Delta_{q+1}=c_1\Delta_q\\
&\text{else}\\
&\quad\text{if } r<r_1 \text{ then } \Delta_{q+1}=c_1\Delta_q\\
&\quad\text{else if } r>r_2 \text{ then } \Delta_{q+1}=\min(c_2\|\boldsymbol{s}\|,\Delta_{\max})\\
&\quad\text{else } \Delta_{q+1}=\Delta_q
\end{aligned}$$

with the following values for the constants: $r_1=0.1$, $r_2=0.75$, $c_1=0.5$, and $c_2=2.0$. The main difference from Ref. 28 is that the update of Δ_q depends both on the value of $\Phi_{\text{hi}}(\boldsymbol{\beta}^q+\boldsymbol{s})$ compared to $\Phi_{\text{hi}}(\boldsymbol{\beta}^q)$ and on the value of r and not just on r. Indeed, whatever the value of r, we believe that the radius of the trust region has to be reduced if the design has not been improved at the preceding iteration. Because the first-order consistency requirements ensure that the corrected low-fidelity model behaves like the high-fidelity model close to the initial design point, the low-fidelity optimization is bound to find a better design point after reducing, possibly several times, the radius of the trust region. Otherwise, this means that an optimum has been reached. The value of r only plays a role when the design has been improved. The update when $r>r_2$ is taken from Refs. 44 and 45. Numerical experiments showed that it was better than the update suggested in Refs. 28 and 46.

7) If the process has not converged, then start a new iteration $(q+1)$ at step 2. The information calculated in step 1 should already exist for the new point from the calculation of r.

FFSQP[47] is used to perform the low-fidelity optimization (33). It is a feasible SQP algorithm, which means that it always satisfies the constraints. We chose a feasible algorithm because of the need to satisfy at least the constraint on the norm of $\boldsymbol{s}$ at the end of each low-fidelity optimization to implement the trust-region method.

The variable-fidelity method is, in fact, very similar to a direct SQP method, but instead of approximating the high-fidelity model by a quadratic function, the variable-fidelity method approximates it by a corrected low-fidelity model. The quadratic approximation is only a mathematical model, whereas the corrected low-fidelity model is supposed to contain some physical knowledge that makes it behave like the high-fidelity model. Hence, the variable-fidelity method should be better than a direct SQP method if the low-fidelity model is fast enough, as we will see later in the example of the airfoil drag minimization.

Results

Accuracy of Adjoint Solver

A straightforward way to assess the accuracy of the adjoint solver is to compare derivatives obtained by a finite difference method with those computed by the adjoint solver. This is what is presented in this section for laminar and turbulent flows in two dimensions. In both cases, the upper surface of the airfoil is parameterized with 10 Bézier design variables and sensitivities to the third and sixth are investigated. The objective functions are lift and drag coefficients. Central finite differencing is employed to compute the finite difference derivatives.

For the viscous laminar case, the NACA0012 airfoil at a freestream Mach number of 0.80, an incidence of 1.0 deg, a Reynolds number based on the airfoil chord of 500, and a freestream temperature of 16 K, is used. The grid around the airfoil is relatively coarse, with 129 × 33 points divided in four blocks. The flow solver and the adjoint solver are converged to a total residual of 10^{-12} for both methodologies. An investigation of the sensitivity of the finite difference method to the step size employed is carried out. The results are presented in Tables 1 and 2 for the lift and drag coefficients, respectively. They both show that the finite difference method is not very sensitive to the step size in this case, with a very good agreement between the three results. In Table 2, for a step size of 10^{-8}, there were not enough available significative digits to be able to distinguish the three results for $\mathrm{d}C_D/\mathrm{d}\beta_k$. The agreement between the finite difference and the adjoint solver is excellent, with three to four significative digits in common and, obviously, the correct sign each time (thus, the right direction in optimization).

The RAE2822 airfoil at a freestream Mach number of 0.725, an incidence of 2.54 deg, a Reynolds number based on the airfoil chord of 6.5×10^6, and a freestream temperature of 283 K is employed for the turbulent case. The grid has the same characteristics

Table 1 Accuracy of the derivatives of lift for the NACA0012 airfoil in laminar flow

	Central finite difference				
Design variable	$\varepsilon=10^{-4}$	$\varepsilon=10^{-6}$	$\varepsilon=10^{-8}$	Adjoint	Difference, %
Third Bézier parameter	0.5273168	0.5273169	0.5273169	0.5273075	0.002
Sixth Bézier parameter	−0.1796504	−0.1796505	−0.1796505	−0.1795517	0.055

Table 2 Accuracy of the derivatives of drag for the NACA0012 airfoil in laminar flow

	Central finite difference				
Design variable	$\varepsilon = 10^{-4}$	$\varepsilon = 10^{-6}$	$\varepsilon = 10^{-8}$	Adjoint	Difference, %
Third Bézier parameter	0.09127480	0.09127479	0.09127475	0.09128026	0.006
Sixth Bézier parameter	0.13399757	0.13399757	0.1339976	0.13401192	0.011

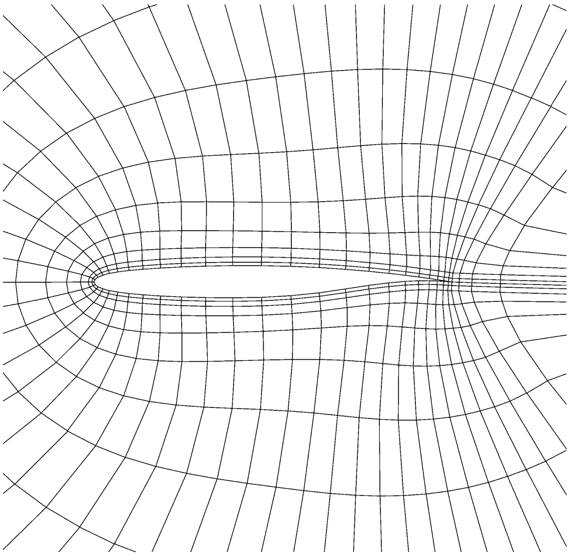
Table 3 Accuracy of the derivatives of lift for the RAE2822 airfoil in turbulent flow

Design variable	Central finite difference $\varepsilon = 10^{-6}$	Adjoint	Difference, %
Third Bézier parameter	1.29579	1.29312	0.21
Sixth Bézier parameter	−0.440811	−0.439373	0.33
Incidence	0.141057	0.140811	0.17

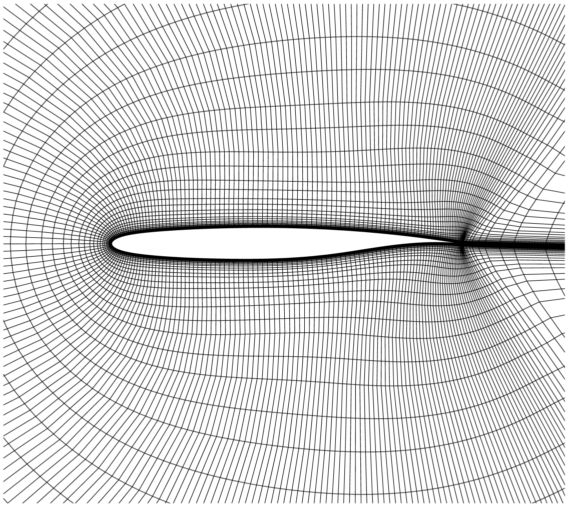
Table 4 Accuracy of the derivatives of drag for the RAE2822 airfoil in turbulent flow

Design variable	Central finite difference $\varepsilon = 10^{-6}$	Adjoint	Difference, %
Third Bézier parameter	−0.00825440	−0.00824008	0.17
Sixth Bézier parameter	0.206073	0.205986	0.04
Incidence	0.00817769	0.00817888	0.01

as the preceding case: four blocks for a total of 129 × 33 points. For this test case, the flow solver is converged to 2.7×10^{-12} because the convergence stalled around this value and did not go beyond. This is, however, already a very good convergence level. There was no problem for the adjoint solver, and so each time it is converged to 10^{-12}. Because the earlier study did not show a significant sensitivity of the finite difference method to the step size, a value of 10^{-6} that seemed reasonable is chosen here. In addition to the two Bézier parameters already used, sensitivities with respect to the design variable controlling the increment in angle of incidence are also calculated. The resulting sensitivities for lift and drag are shown in Tables 3 and 4. The agreement between the finite difference and the adjoint method is not as good as for the preceding case, but it is, nevertheless, sufficient for engineering applications. The difference is supposed to come from the approximations made when the maximum function in the linearization of the Baldwin–Lomax turbulence model [Eq. (18)] is differentiated. Notice, however, that for this case the order of magnitude of the sensitivity derivatives is quite different from one objective function to another, or from one design variable to another; yet the adjoint method is accurate each time and finds the correct sign even for very low values.



a) Low-fidelity grid



b) High-fidelity grid

Fig. 3 Variable-fidelity grids employed in the two-dimensional constrained drag minimization.

Airfoil Optimization

The constrained drag minimization problem of a two-dimensional airfoil is presented in this section. This work was part of the European Multidisciplinary Design and Optimisation for Blended Wing Body Configuration (MOB) project.[48] One of the aims of this project was to improve the design of a blended wing–body aircraft.[49] The initial geometry presented a strong shock wave on the outer wing, and the purpose of this work was to minimize this shock wave by performing a two-dimensional optimization on the outer wing section. The flow conditions are a freestream Mach number of 0.725, a unit Reynolds number of 5.691×10^6/m, and a freestream temperature of 218.8 K. The local lift coefficient at which the airfoil operates serves as a target C_L at which the optimization has to be carried out. A geometrical constraint on the interior volume (which is, in fact, an area because the problem is only two dimensional) of the airfoil is added to prevent the optimizer from thinning it too much to reduce drag. The optimization problem is as follows:

$$\begin{aligned} &\text{minimize} && C_D \\ &\text{subject to} && C_L \geq 0.652 \\ &&& V_0 \leq V \leq 2V_0 \end{aligned} \tag{36}$$

The upper and lower surfaces of the airfoil are free to deform, and the perturbation added to the baseline curves is parameterized with 10 Bézier parameters for each surface. Because the leading- and trailing-edge points are fixed, this makes 16 design variables. In addition, the angle of incidence is considered a design variable.

Both the variable-fidelity method and a direct high-fidelity optimization are employed to solve this problem to compare the performance of both approaches. For the variable-fidelity approach, the low-fidelity model solves an inviscid optimization problem on a coarse Euler grid of 65 × 17 nodes, whereas the high-fidelity model solves the Navier–Stokes equations on a mesh of 273 × 69 grid points. The two grids are shown in Fig. 3. The optimization problem (36) becomes a corrected low-fidelity model similar to Eq. (33):

$$
\begin{aligned}
\underset{\boldsymbol{s}}{\text{minimize}} \quad & \frac{\widetilde{C_{D\,\mathrm{lo}}}(\boldsymbol{\beta}^q + \boldsymbol{s})}{\left\| \mathrm{d}\widetilde{C_{D\,\mathrm{lo}}}(\boldsymbol{\beta}^q) \big/ \mathrm{d}\boldsymbol{\beta}^q \right\|} \\
\text{subject to} \quad & \frac{0.652 - C_{L\,\mathrm{hi}}(\boldsymbol{\beta}^q) - \nabla C_{L\,\mathrm{hi}}(\boldsymbol{\beta}^q)^t \boldsymbol{s}}{\left\| \mathrm{d}C_{L\,\mathrm{hi}}(\boldsymbol{\beta}^q) \big/ \mathrm{d}\boldsymbol{\beta}^q \right\|} \le 0 \\
& \frac{V_0 - V_{\mathrm{hi}}(\boldsymbol{\beta}^q) - \nabla V_{\mathrm{hi}}(\boldsymbol{\beta}^q)^t \boldsymbol{s}}{\left\| \mathrm{d}V_{\mathrm{hi}}(\boldsymbol{\beta}^q) \big/ \mathrm{d}\boldsymbol{\beta}^q \right\|} \le 0 \\
& \frac{-2V_0 + V_{\mathrm{hi}}(\boldsymbol{\beta}^q) + \nabla V_{\mathrm{hi}}(\boldsymbol{\beta}^q)^t \boldsymbol{s}}{\left\| \mathrm{d}V_{\mathrm{hi}}(\boldsymbol{\beta}^q) \big/ \mathrm{d}\boldsymbol{\beta}^q \right\|} \le 0 \\
& \beta_k^l - \beta_k^q \le s_k \le \beta_k^u - \beta_k^q, \qquad k = 1, 17 \\
& \|\boldsymbol{s}\| \le \Delta_q
\end{aligned}
\tag{37}
$$

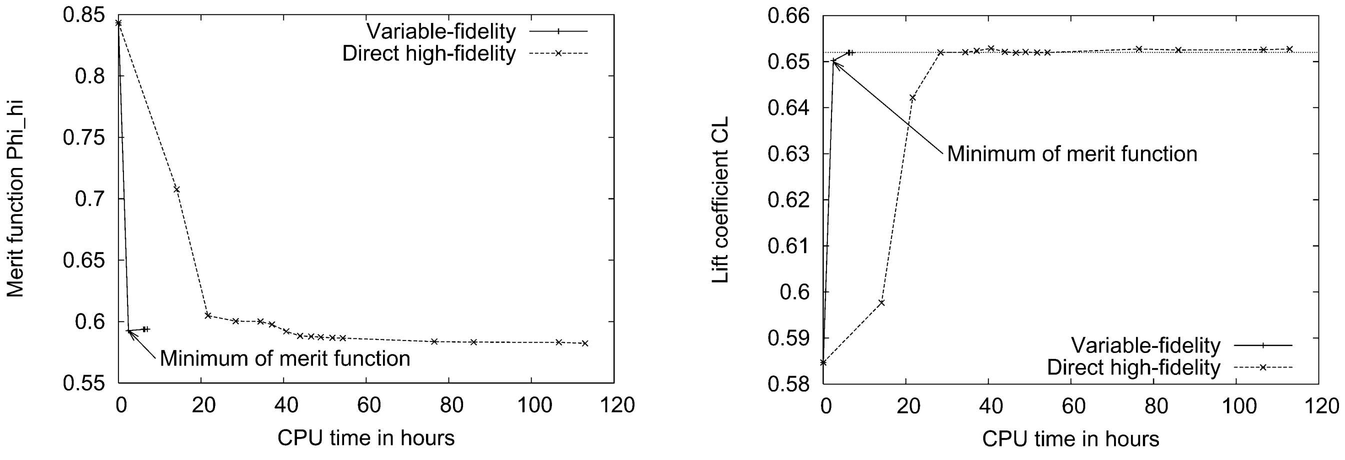
Note that all of the functions are scaled by the norm of their initial gradient.[42] The merit function Φ_{hi} that is used by the variable-fidelity method to assess the high-fidelity performance of new designs is

$$
\Phi_{\mathrm{hi}} = \frac{C_D}{\left\| \mathrm{d}C_D \big/ \mathrm{d}\boldsymbol{\beta}^0 \right\|} + 10 \frac{(C_L - 0.652)^2}{\left\| \mathrm{d}C_L \big/ \mathrm{d}\boldsymbol{\beta}^0 \right\|^2} + 10 \frac{(V - V_0)^2}{\left\| \mathrm{d}V_0 \big/ \mathrm{d}\boldsymbol{\beta}^0 \right\|^2} \tag{38}
$$

The direct high-fidelity optimization employs a standard SQP algorithm provided in the form of the NAG subroutine E04UCF.[50] It solves the same Navier–Stokes equations on the same grid as the high-fidelity model in the variable-fidelity optimization.

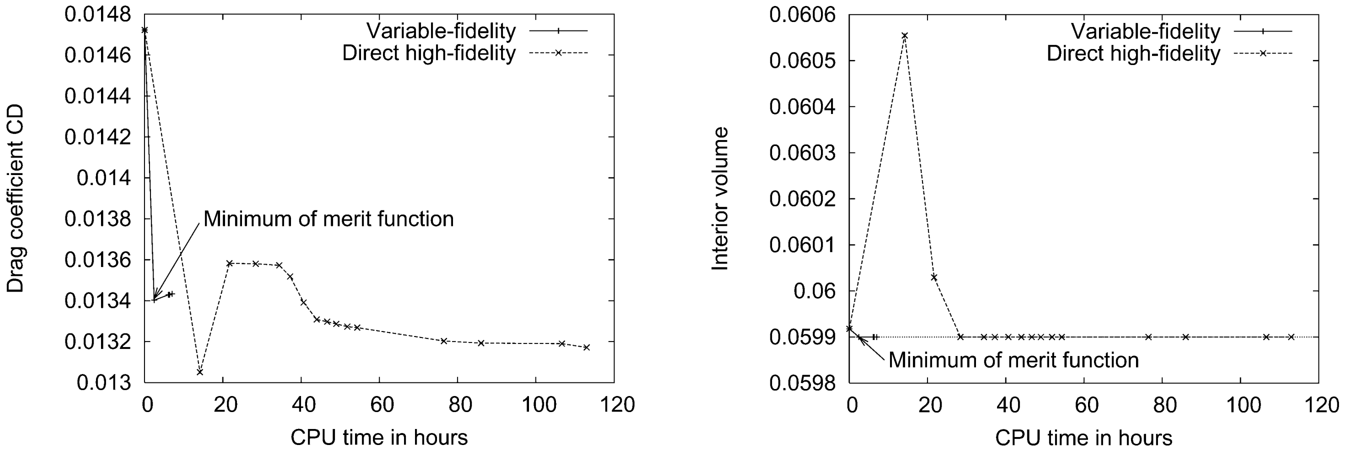
The optimizations are both started with the angle of incidence set to 2 deg, which makes the initial design point infeasible for the constraint on lift. The flow solver is run each time until the total residual has reached 10^{-6}, or until 5000 implicit iterations have been completed, whichever comes first. Similarly, the adjoint solver is stopped when the residual has also reached 10^{-6}, or when 3000 implicit iterations have been performed. The maximum number of iterations chosen should be sufficient to reach the six-order convergence requirement when starting from freestream conditions. In addition, both the flow and adjoint solvers use their restart capabilities to start each calculation from the preceding solution to reduce the computing time as much as possible. The optimizations are started from converged flow and adjoint solutions on the baseline geometry. Mild convergence requirements and a maximum number of 20 optimization cycles are imposed for the low-fidelity optimization in the variable-fidelity method because it does not need to be very accurate, but has to be as fast as possible. The calculations are run on a single Sun UltraSPARC III Cu 900-MHz processor.

Figure 4 shows the optimization history for the two methods. Although no merit function Φ_{hi} is used in the direct high-fidelity approach, one is calculated a posteriori from the data obtained and is plotted in Fig. 4a to provide a means of comparison of the two methods that accounts for the progress both in the objective function and in the satisfaction of the constraints. Figure 4a shows that the variable-fidelity method finds its best point after only one cycle, that is, one low-fidelity optimization is enough to find a minimum. As a result, it is very fast (2.4 h) compared to the direct high-fidelity method that reaches the same level of improvement only after 40 h. However, even if it takes a long time, the direct high-fidelity approach is able to reach a better design point than the variable-fidelity method. This is confirmed by Figs. 4a and 4c, where the direct high-fidelity optimization finds points with a lower drag and that better satisfy the constraint on lift.



a) Merit function Φ_{hi}

c) Lift coefficient C_L



b) Drag coefficient C_D

d) Interior volume

Fig. 4 Evolution of different parameters during the airfoil optimizations.

For the same level of improvement, the shapes and pressure distributions obtained by the two methods are plotted in Fig. 5. The optimized airfoils are more cambered than the initial one, and the shock wave on the upper surface has almost been eliminated. The aerodynamic coefficients of these optimized airfoils are summarized in Table 5, showing almost 23% of drag reduction. Table 6 compares the performance of the two optimization methods by providing the number of calls to the aerodynamic functions and gradients to reach the same level of improvement. For the high-fidelity count, one call to the flow solver that provides both the value of C_L and C_D counts as one, whereas, for the gradient, one call to the adjoint solver only provides the gradient of C_L or C_D, so that if both are needed, this counts as two. For the low-fidelity model because the constraints have been linearized [Eq. (37)], only the value of C_D and of its gradient are needed, thus reducing the number of gradient evaluations. Table 6 clearly demonstrates the advantage of the use of the variable-fidelity method as opposed to a direct optimization method because the number of calls to high-fidelity functions and gradients is significantly reduced. However, we are probably fortunate here to reach the best design point in only one low-fidelity optimization, and, in general, the comparison should be slightly more balanced. In addition, the direct high-fidelity optimization has its merit because it can achieve better improvement, at least in this example.

Table 5 Aerodynamic coefficients of the optimized airfoils at the target C_L; for direct high-fidelity method, the airfoil is the one obtained after 40 h of CPU time

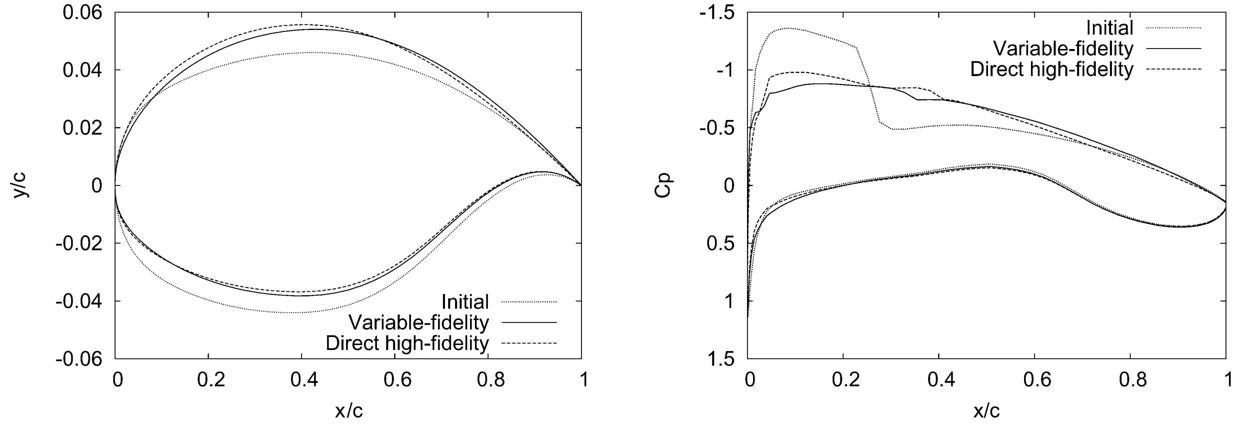
Geometry	C_L	C_D	Improvement in C_D, %
Initial	0.6520	0.01737	——
Variable fidelity	0.6520	0.01343	−22.7
Direct high fidelity	0.6521	0.01340	−22.9

Table 6 Comparison of the number of function and gradient evaluations to reach the same improvement

Optimization method	High-fidelity		Low-fidelity		CPU time, h
	Funtions	Gradients	Funtions	Gradients	
Variable fidelity	2	2	10	10	2.4
Direct high fidelity	10	20	——	——	40.6

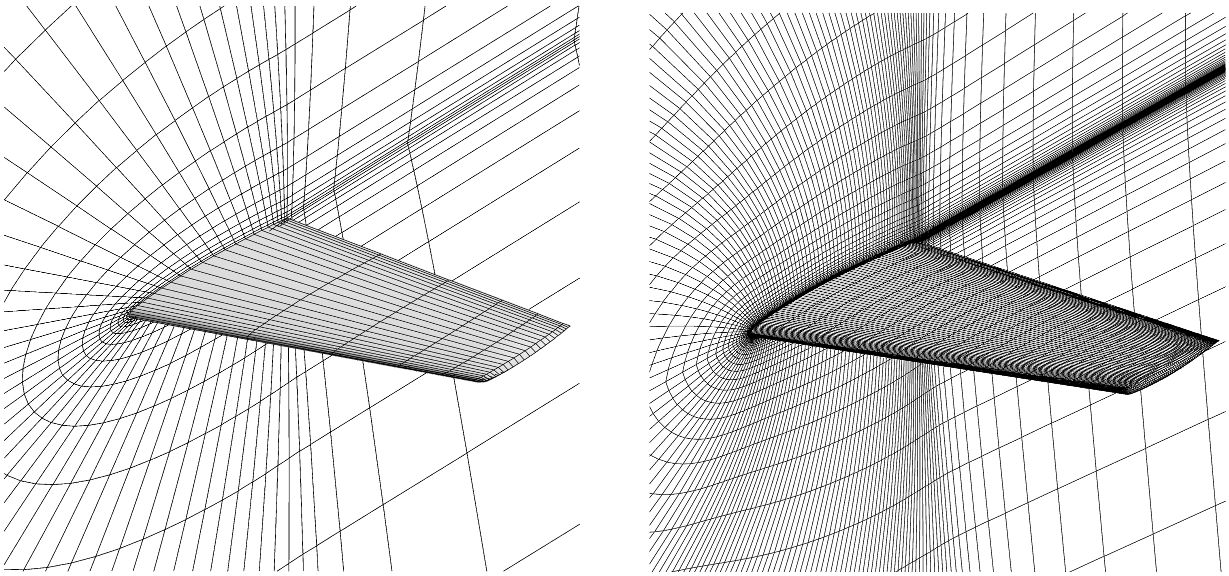
Optimization of the ONERA M6 Wing

This section describes the Navier–Stokes optimization of the ONERA M6 wing[29] using the variable-fidelity method only. This is a good test case for three-dimensional optimization, and similar optimizations of the M6 wing can be found in the literature for viscous turbulent flows.[15,23,51] The test case selected is the classical 3.06 deg of incidence with a Mach number of 0.84 and a Reynolds number of 11.7×10^6 based on the mean aerodynamic chord. The aim of the optimization is to reduce drag while the lift coefficient obtained at these conditions is maintained. The wing is divided into six master sections connected linearly in the spanwise direction. These sections are parameterized in the same way as the two-dimensional



a) Airfoil shapes **b) Chordwise pressure distributions at the target C_L**

Fig. 5 Results of the airfoil optimizations; for the direct high-fidelity method, the airfoil is the one obtained after 40 h of CPU time.



a) Low-fidelity grid **b) High-fidelity grid**

Fig. 6 Variable-fidelity grids employed in the constrained drag minimization of the ONERA M6 wing.

airfoil of the preceding section, that is, with eight active Bézier parameters controlling the shape modifications of each surface. In addition, a design variable controls the increment in twist of each master section. The most outboard master section does not undergo shape modification to avoid complexity in deforming the wing tip. It is only allowed to twist. This twist change is transmitted to all of the grid sections that form the tip and that are situated outboard of this last master section. In total, the optimization, hence, uses 86 design variables ($5 \times 17 + 1$). The optimization problem is as follows:

$$\begin{aligned} &\text{minimize} && C_D \\ &\text{subject to} && C_L \geq 0.2696 \\ &&& V_{0i} \leq V_i \leq 2V_{0i}, \qquad i = 1, \ldots, 5 \end{aligned} \tag{39}$$

As for the preceding case, the low-fidelity model solves the Euler equations on a $65 \times 17 \times 7$ grid divided into four blocks, whereas the high-fidelity model solves the Navier–Stokes equations on a $193 \times 49 \times 33$ grid (Fig. 6). The corrected low-fidelity model is very similar to Eq. (37), but with a constraint on internal volume for the first five master sections. The merit function is

$$\Phi_{\text{hi}} = \frac{C_D}{\left\| \mathrm{d}C_D / \mathrm{d}\boldsymbol{\beta}^0 \right\|} + 10 \frac{(C_L - 0.2696)^2}{\left\| \mathrm{d}C_L / \mathrm{d}\boldsymbol{\beta}^0 \right\|^2} + 10 \sum_{i=1}^{5} \frac{(V_i - V_{0i})^2}{\left\| \mathrm{d}V_{0i} / \mathrm{d}\boldsymbol{\beta}^0 \right\|^2} \tag{40}$$

During the optimization, the flow and adjoint solvers are required to converge to a total residual of 10^{-5}, or are stopped after 5000 and 3000 implicit iterations, respectively. As for the airfoil optimization, the process is started from converged flow and adjoint solutions on the baseline M6 wing geometry, and each code uses its restart capability to save computing time. Because of the high computing cost involved in solving such a three-dimensional turbulent problem, both the flow and adjoint solvers are parallelized with OpenMP and domain decomposition to reduce perceived computing time. The computation is run on a DEC Alpha Unix server with 4 EV6.7 processors running at 667 MHz and 4 GB of RAM.

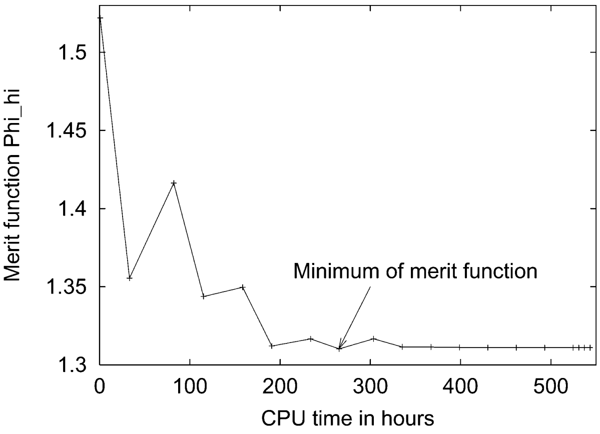
The optimization history is shown in Fig. 7, the computing time displayed being the sum of the effort of the four processors. The history is oscillatory, but the process manages to reduce drag while maintaining lift as is required. For information, the number of function and gradient calls are provided in Table 7. Table 8 shows that the drag decrease is due to an improvement in the pressure drag, whereas the skin-friction drag increases slightly. Because of the small change in surface area, the skin-friction drag is not expected to change a lot. The decrease of the pressure drag is largely coming from a reduction of the wave drag as the contour lines of pressure coefficient in Fig. 8 show: The characteristic lambda shock wave of the original ONERA M6 wing has been weakened substantially, except in the tip region, where limited shape changes were allowed. This is confirmed by the chordwise pressure distributions in Fig. 9 at the stations where experimental data[29] are available for the baseline geometry. At all of the stations, the strength of the shock wave has

Table 7 Number of function and gradient evaluations during the M6 wing optimization; CPU = 265 h

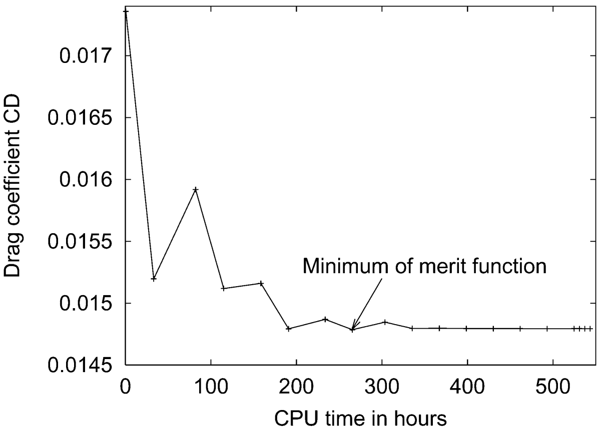
Evaluation	High-fidelity	Low-fidelity
Functions	8	44
Gradients	8	35

Table 8 Aerodynamic coefficients of the optimized M6 wing

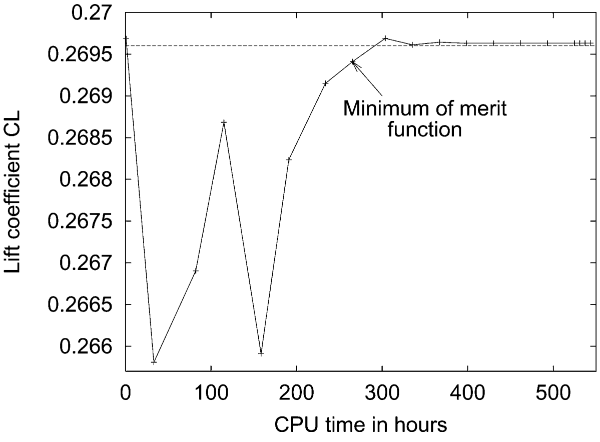
Geometry	C_L	$C_{D\,\text{total}}$	$C_{D\,\text{press}}$	$C_{D\,\text{fric}}$
Initial	0.2697	0.01736	0.01241	0.00495
Optimized	0.2694	0.01478	0.00973	0.00505



a) Merit function Φ_{hi}



b) Drag coefficient C_D



c) Lift coefficient C_L

Fig. 7 Evolution of different parameters during the ONERA M6 wing optimization.

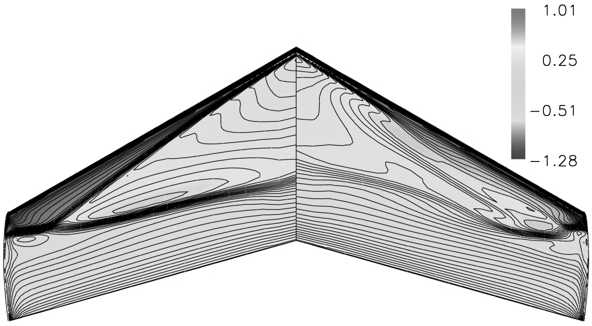
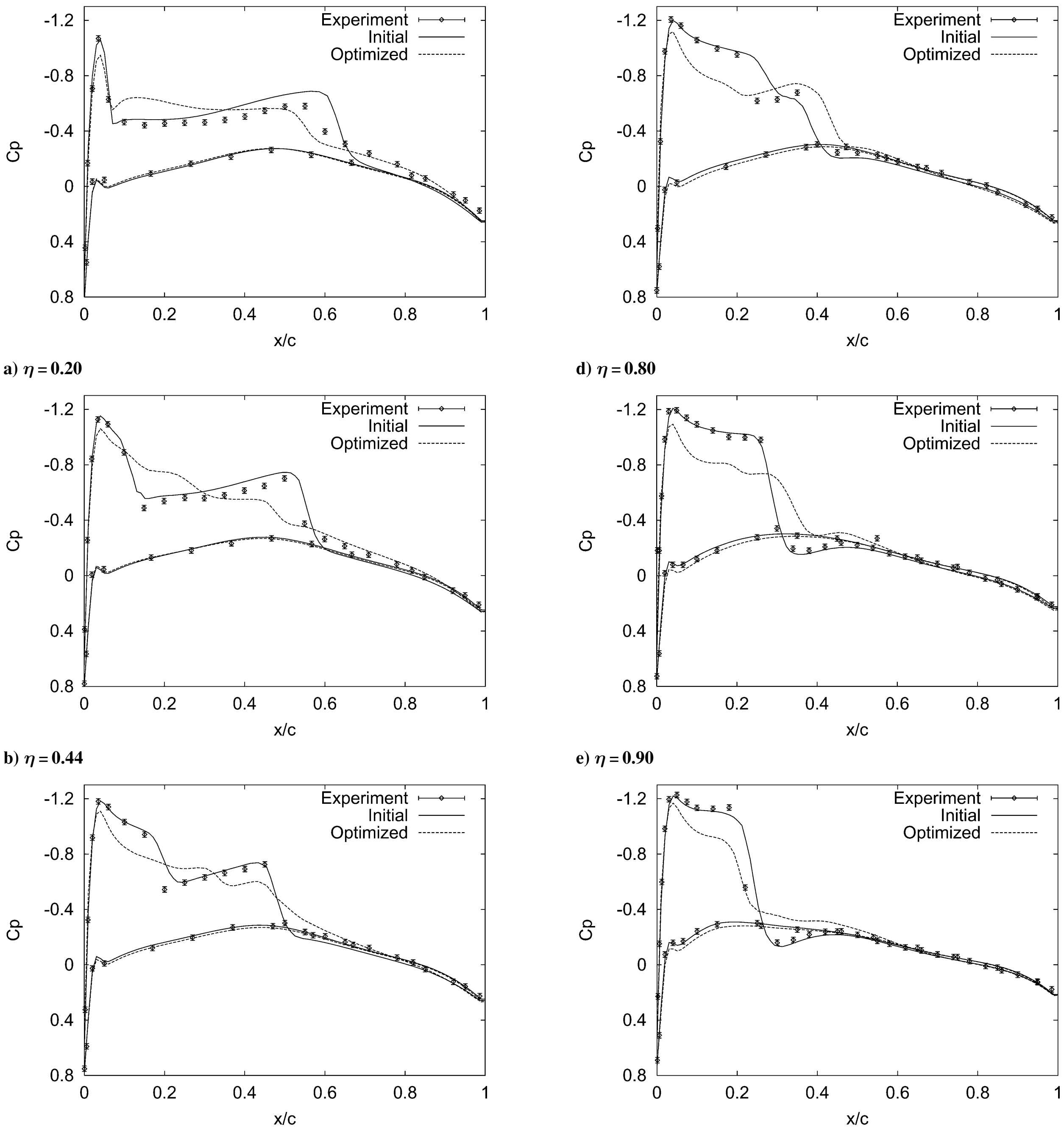


Fig. 8 Comparison of the contours of pressure coefficient on the upper surface of the ONERA M6 wing before and after optimization.

Table 9 Comparison of aerodynamic coefficients obtained for optimized ONERA M6 wings

	Initial			Optimized			
Origin of data	C_L	C_D	L/D	C_L	C_D	Improvement in C_D, %	L/D
Present work	0.2697	0.01736	15.54	0.2694	0.01478	−14.9	18.23
Nielsen and Anderson[15]	0.253	0.0168	15.06	0.253	0.0142	−15.5	17.82
Lee et al.[23]	0.2622	0.01751	14.97	0.2580	0.01586	−9.4	16.27
Sung and Kwon[51]	N/A	N/A	N/A	−2.1%	N/A	−12.8	N/A



a) $\eta = 0.20$

b) $\eta = 0.44$

c) $\eta = 0.65$

d) $\eta = 0.80$

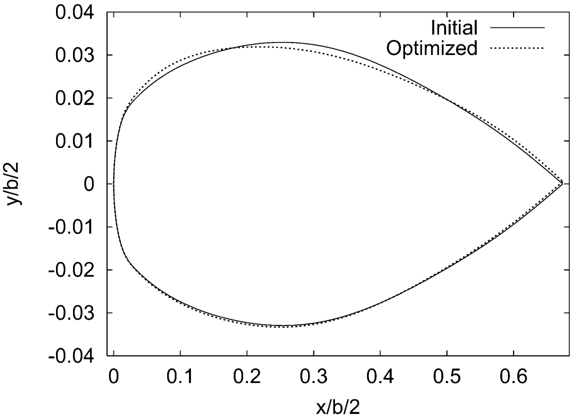
e) $\eta = 0.90$

f) $\eta = 0.95$

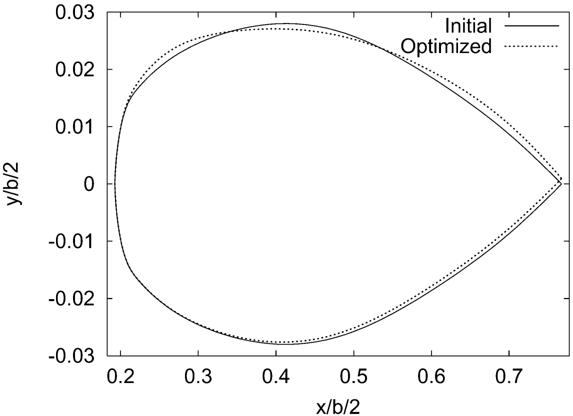
Fig. 9 Chordwise C_p distributions for the optimized ONERA M6 wing.

been reduced, and it has almost disappeared at the stations $\eta = 0.44$ and 0.65. Notice also the fair agreement between the computation on the baseline geometry and the experimental data, proving the validity of the results.

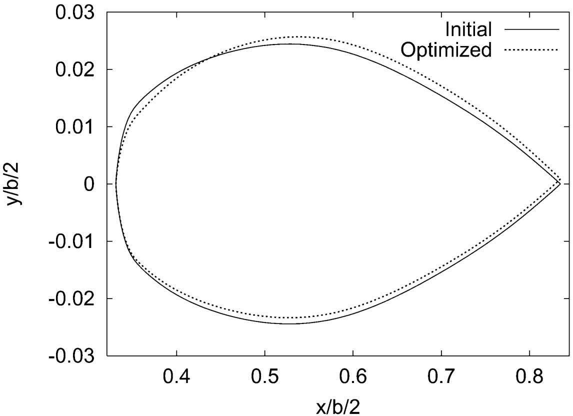
The resulting shape changes are displayed in Fig. 10 for the six master sections used to represent and deform the wing. These changes are relatively small but prove to be sufficient to improve the aerodynamics significantly. These results show that this optimization is successful and compares well with other Navier–Stokes optimizations found in the literature, as can be seen in Table 9. The drag improvement is similar to that of Nielsen and Anderson,[15] although their geometry changes are much more pronounced, and they obtain very different pressure distributions. The present pressure distributions are closer to those of Lee et al.[23] and Sung and Kwon,[51] although their overall drag improvements are not as good.



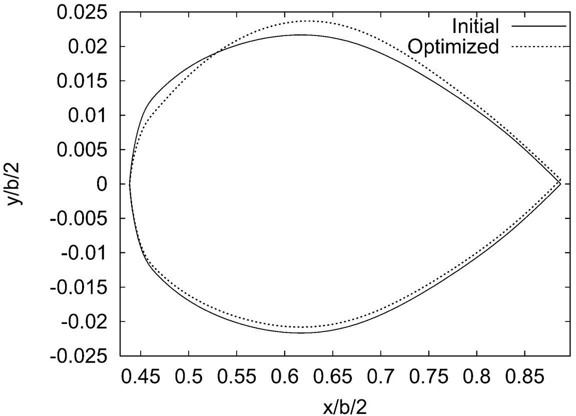
a) Root section



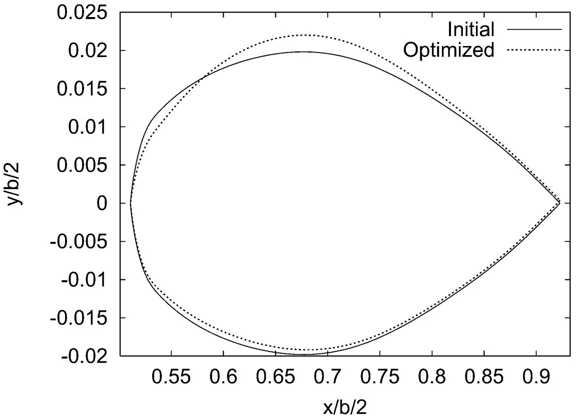
b) $\eta = 0.34$



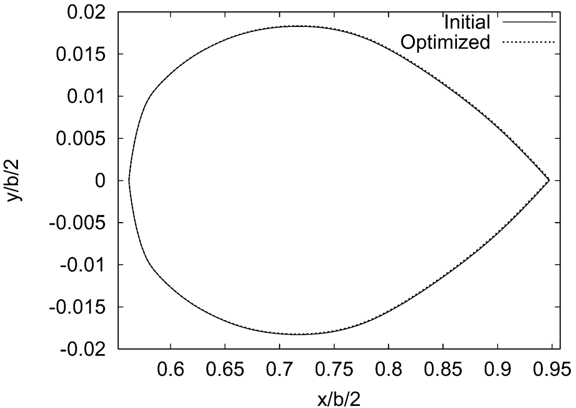
c) $\eta = 0.58$



d) $\eta = 0.77$



e) $\eta = 0.90$



f) $\eta = 0.98$

Fig. 10 Shape modifications of the master sections of the ONERA M6 wing.

Summary

An aerodynamic optimization process for three-dimensional turbulent flows has been presented. It is based on an implicit Reynolds-averaged Navier–Stokes flow solver for structured multiblock grids that uses Osher's approximate Riemann solver. A discrete adjoint solver associated with the flow solver has been derived analytically. This involves the calculation of a matrix–vector product of the exact Jacobian and the adjoint vector. Details have been given about how this is achieved for the high-order convective fluxes and the viscous fluxes. The effect of the full stencil turbulence modeling has been taken into account in the Jacobian. The description of this matrix–vector product is only valid for interior points. In practice, at the boundary points, the linearization of the boundary conditions has to be added.

The efficiency of the optimization process is improved by the use of a variable-fidelity method that performs most of the optimization on a low-fidelity cheap-to-run model (solving the Euler equations on a coarse grid in our case), thus reducing the overall computing time. This was demonstrated by the airfoil drag minimization problem presented in this paper, where the variable-fidelity method reduces the number of high-fidelity function evaluations by a factor of 10 and the total computing time by a factor of 17 compared to a direct high-fidelity optimization.

Comparison with finite differences have shown that the adjoint solver provides accurate sensitivity derivatives. The optimization method has been applied successfully to the drag minimization problem of an airfoil and of the ONERA M6 wing, where drag reductions of 23 and 15%, respectively have been achieved while lift, interior volume, and, for the M6 wing, the planform have been maintained.

Acknowledgments

This work was funded by a Scholarship from Cranfield College of Aeronautics, England, U.K., and the European Union under Growth Programme Contract G4RD-CT1999-0172 for the MOB project. The authors acknowledge AEMDesign for kindly providing the optimizer FFSQP used in the present work.

References

[1] Newman, J. C., Taylor, A. C., Barnwell, R. W., Newman, P. A., and Hou, G. J.-W., "Overview of Sensitivity Analysis and Shape Optimization for Complex Aerodynamic Configurations," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 87–96.

[2] Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, No. 3, 1988, pp. 233–260.

[3] Pironneau, O., "On Optimum Profiles in Stokes Flow," *Journal of Fluid Mechanics*, Vol. 59, No. 1, 1973, pp. 117–128.

[4] Pironneau, O., "On Optimum Design in Fluid Mechanics," *Journal of Fluid Mechanics*, Vol. 64, No. 1, 1974, pp. 97–110.

[5] Glowinski, R., and Pironneau, O., "On the Numerical Computation of the Minimum-Drag Profile in Laminar Flow," *Journal of Fluid Mechanics*, Vol. 72, 1975, pp. 385–389.

[6] Angrand, F., "Optimum Design for Potential Flows," *International Journal for Numerical Methods in Fluids*, Vol. 3, 1983, pp. 265–282.

[7] Reuther, J., Jameson, A., Farmer, J., Martinelli, L., and Saunders, D., "Aerodynamic Shape Optimization of Complex Aircraft Configurations via an Adjoint Formulation," AIAA Paper 96-0094, Jan. 1996.

[8] Reuther, J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers," *AIAA Journal*, Vol. 36, No. 1, 1999, pp. 51–74.

[9] Jameson, A., Pierce, N. A., and Martinelli, L., "Optimum Aerodynamic Design Using the Navier–Stokes Equations," AIAA Paper 97-0101, Jan. 1997.

[10] Cross, M. R., "Aerodynamic Design Using the Euler Adjoint Approach," International Congress of Aeronautical Sciences, Paper ICAS-2000-2.7.1., Aug. 2000.

[11] Sung, C., and Kwon, J. H., "Accurate Aerodynamic Sensitivity Analysis Using Adjoint Equations," *AIAA Journal*, Vol. 38, No. 2, 2000, pp. 243–250.

[12] Soemarwoto, B. I., "The Variational Method for Aerodynamic Optimization Using the Navier–Stokes Equations," NASA CR-97-206277, ICASE Rept. 97-71, 1997.

[13] Giles, M. B., Duta, M. C., and Muller, J.-D., "Adjoint Code Developments Using the Exact Discrete Approach," AIAA Paper 2001-2596, June 2001.

[14] Kim, C. S., Kim, C., and Rho, O.-H., "Aerodynamic Sensitivity Analysis for Turbulent Flows on Chimera Overlaid Grids," *AIAA Journal*, Vol. 39, No. 5, 2001, pp. 838–845.

[15] Nielsen, E. J., and Anderson, W. K., "Recent Improvements in Aerodynamic Design Optimization on Unstructured Meshes," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1155–1163.

[16] Cormery, M., "Adjoint Operator Approach to Aerodynamic Shape Optimisation for 3D Transonic Flows," *Proceedings of the 4th European Computational Fluid Dynamics Conference ECCOMAS 98*, Vol. 1, Wiley, New York, 1998, pp. 610–616.

[17] Burgreen, G. W., "Three-Dimensional Aerodynamic Shape Optimization Using Discrete Sensitivity Analysis," Ph.D. Dissertation, Dept. of Mechanical Engineering, Old Dominion Univ., Norfolk, VA, May 1994.

[18] Burgreen, G. W., and Baysal, O., "Three-Dimensional Aerodynamic Shape Optimization Using Discrete Sensitivity Analysis," *AIAA Journal*, Vol. 34, No. 9, 1996, pp. 1761–1770.

[19] Giles, M. B., and Pierce, N. A., "On the Properties of Solutions of the Adjoint Euler Equations," 6th ICFD Conference on Numerical Methods for Fluid Dynamics, March 1998.

[20] Spalart, P. R., and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aerospatiale*, Vol. 1, 1994, pp. 5–21; also AIAA Paper 92-0439, Jan. 1992.

[21] Anderson, W. K., and Bonhaus, D. L., "Aerodynamic Design on Unstructured Grids for Turbulent Flows," NASA TM 112867, 1997.

[22] Nemec, M., and Zingg, D. W., "Towards Efficient Aerodynamic Shape Optimization Based on the Navier–Stokes Equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154, also AIAA Paper 2001-2532, 2001.

[23] Lee, B. J., Kim, C. S., Kim, C., Rho, O.-H., and Lee, K. D., "Parallelized Design Optimization for Transonic Wings Using Aerodynamic Sensitivity Analysis," AIAA Paper 2002-0264, Jan. 2002.

[24] Baldwin, B. S., and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, Jan. 1978.

[25] Le Moigne, A., "A Discrete Navier–Stokes Adjoint Method for Aerodynamic Optimisation of Blended Wing–Body Configurations," Ph.D. Dissertation, College of Aeronautics, Cranfield Univ., Cranfield, England, U.K., Dec. 2002.

[26] Alexandrov, N. M., Lewis, R. M., Gumbert, C. R., Green, L. L., and Newman, P. A., "Optimization with Variable-Fidelity Models Applied to Wing Design," AIAA Paper 2000-0841, Jan. 2000.

[27] Alexandrov, N. M., Nielsen, E. J., Lewis, R. M., and Anderson, W. K., "First-Order Model Management with Variable-Fidelity Physics Applied to Multi-Element Airfoil Optimization," AIAA Paper 2000-4886, Sept. 2000.

[28] Alexandrov, N. M., Lewis, R. M., Gumbert, C. R., Green, L. L., and Newman, P. A., "Approximation and Model Management in Aerodynamic Optimization with Variable-Fidelity Models," *Journal of Aircraft*, Vol. 38, No. 6, 2001, pp. 1093–1101.

[29] Schmitt, V., and Charpin, F., "Pressure Distributions on the ONERA-M6-Wing at Transonic Mach Numbers," *Experimental Data Base for Computer Program Assessment*, AR-138, AGARD, 1979.

[30] Osher, S., and Solomon, F., "Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws," *Mathematics of Computation*, Vol. 38, No. 158, 1982, pp. 339–374,

[31] Chakravarthy, S. R., and Osher, S., "Numerical Experiments with the Osher Upwind Scheme for the Euler Equations," *AIAA Journal*, Vol. 21, No. 9, 1983, pp. 1241–1248.

[32] Van Leer, B., "Towards the Ultimate Conservative Difference Scheme. III—Upstream-Centered Finite-Difference Schemes for Ideal Compressible Flow," *Journal of Computational Physics*, Vol. 23, March 1977, pp. 263–275.

[33] Van Leer, B., "Towards the Ultimate Conservative Difference Scheme. IV—A New Approach to Numerical Convection," *Journal of Computational Physics*, Vol. 23, March 1977, pp. 276–299.

[34] Van Leer, B., "Towards the Ultimate Conservative Difference Scheme. V—A Second-Order Sequel to Godunov's Method (for Ideal Compressible Flow)," *Journal of Computational Physics*, Vol. 32, July 1979, pp. 101–136.

[35] Qin, N., and Ludlow, D. K., and Shaw, S. T., "A Matrix-Free Preconditioned Newton/GMRES Method for Navier–Stokes Solutions," *International Journal for Numerical Methods in Fluids*, 2000, Vol. 33, No. 3, pp. 223–248.

[36] Spekreijse, S. P., "Multigrid Solution of the Steady Euler Equations," Ph.D. Dissertation, Centrum Voor Wiskunde en Informatica, Amsterdam, 1987.

[37] Shaw, S. T., "Numerical Study of the Unsteady Aerodynamics of Helicopter Rotor Aerofoils," Ph.D. Dissertation, College of Aeronautics, Cranfield Univ., Cranfield, England, U.K., 1999.

[38] Korivi, V. M., Taylor, A. C., Newman, P. A., Hou, G. J.-W., and Jones, H. E., "An Approximately Factored Incremental Strategy for Calculating Consistent Discrete Aerodynamic Sensitivity Derivatives," *Journal of Computational Physics*, Vol. 113, 1994, pp. 336–346.

[39] Sherman, L. L., Taylor, A. C., Green, L. L., Newman, P. A., Hou, G. J.-W., and Korivi, V. M., "First- and Second-Order Aerodynamic Sensitivity Derivatives via Automatic Differentiation with Incremental Iterative Methods," *Journal of Computational Physics*, Vol. 129, No. 2, 1996, pp. 307–331.

[40] Newman, P. A., Hou, G. J.-W., Jones, H. E., Taylor, A. C., and Korivi, V. M., "Observations on Computational Methodologies for Uses in Large-Scale, Gradient-Based, Multidisciplinary Design," *4th AIAA/USAF/NASA/OAI Symposium on Multidisciplinary Analysis and Optimization*, Vol. 1, AIAA, Washington, DC, 1992, pp. 531–542.

[41] Wilcox, D., *Turbulence Modeling for CFD*, DCW Industries, Inc., La Canada, CA, 1993, Chap. 3.

[42] Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design: With Applications*, McGraw–Hill, New York, 1984.

[43] Gill, P. E., Murray, W., and Wright, M. H., *Practical Optimization*, Academic Press, London, 1981.

[44] Lewis, R. M., "A Trust Region Framework for Managing Approximation Models in Engineering Optimization," *6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Vol. 2, AIAA, Reston, VA, 1996, pp. 1053–1055.

[45] Alexandrov, N. M., "Robustness Properties of a Trust Region Framework for Managing Approximations in Engineering Optimization," *6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Vol. 2, AIAA, Reston, VA, 1996, pp. 1056–1059.

[46] Alexandrov, N. M., Dennis, J. E., Lewis, R. M., and Torczon, V., "A Trust Region Framework for Managing the Use of Approximation Models in Optimization," NASA CR-201745, ICASE Rept. 97-50, 1997.

[47] Zhou, J. L., Tits, A. L., and Lawrence, C. T., "User's Guide for FFSQP Version 3.7: a FORTRAN Code for Solving Constrained Nonlinear (Minimax) Optimization Problems, Generating Iterates Satisfying all Inequality and Linear Constraints," Systems Research Center, TR-92-107r2, Univ. of Maryland, College Park, MD, 1997.

[48] Morris, A. J., "MOB a European Distributed Multidisciplinary Design and Optimization Project," AIAA Paper 2002-5444, Sept. 2002.

[49] Qin, N., Vavalle, A., Le Moigne, A., Laban, M., Hackett, K., and Weinerfelt, P., "Aerodynamic Studies for Blended Wing Body Aircraft," AIAA Paper 2002-5448, Sept. 2002.

[50] Numerical Algorithm Group, "E04UCF-NAG Fortran Library Routine Documentation," available at http://www.nag.co.uk/numeric/fl/manual/pdf/E04/e04ucf.pdf [cited 1 June 2004].

[51] Sung, C., and Kwon, J. H., "Aerodynamic Design Optimization Using the Navier–Stokes and Adjoint Equations," AIAA Paper 2001-0266, Jan. 2001.

A. Messac
Associate Editor