Computers & Fluids 39 (2010) 373–391

Contents lists available at ScienceDirect

Computers & Fluids

journal homepage: www.elsevier.com/locate/compfluid

Review

Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches

Jacques E.V. Peter [a], Richard P. Dwight [b,*]

[a] ONERA BP 72 – 29 av. de la Division Leclerc, 92322 Chatillon Cedex, France
[b] Aerodynamics Group, TU Delft, P.O. Box 5058, 2600GB Delft, The Netherlands

ARTICLE INFO

Article history:
Received 19 November 2007
Received in revised form 21 July 2009
Accepted 24 September 2009
Available online 6 October 2009

Keywords:
Sensitivity analysis
Linearized Navier–Stokes
Flow solver linearization
Adjoint method
Duality preserving iteration
Krylov stabilization
Aerodynamic optimization

ABSTRACT

The calculation of the derivatives of output quantities of aerodynamic flow codes, commonly known as numerical sensitivity analysis, has recently become of increased importance for a variety of applications in flow analysis, but the original motivation came from the field of aerodynamic shape optimization. There the large numbers of design variables needed to parameterize surfaces in 3D necessitates the use of gradient-based optimization algorithms, and hence efficient and accurate evaluation of gradients. In this context over the last 20 years a variety of approaches have been developed to supply these gradients, raising particular challenges that have required novel algorithms. In this paper, we examine the historical development of these approaches, describe in some detail the theoretical background of each major method and the associated numerical techniques required to make them practical in an engineering setting. We give examples from our own experience and describe what we consider to be the state-of-the-art in these methods, including their application to optimization of complex 3D aircraft configurations.

Contents

* Corresponding author.
E-mail addresses: jacques.peter@onera.fr (J.E.V. Peter), r.p.dwight@tudelft.nl (R.P. Dwight).

0045-7930/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compfluid.2009.09.013

1. Introduction

In 1987, Sobieszczanski-Sobieski, a specialist in multi-disciplinary optimization working at NASA, made a plea to the CFD community, to extend their codes from simple aerodynamic analysis to sensitivity analysis [1], by which he meant evaluation of the derivatives of aerodynamic quantities (typically depending on both geometry and flow) with respect to some parameterization of the geometry. Such an evaluation is demanding because the flow itself depends on the geometry over a system of partial differential equations. He pointed out the lead of computational solid mechanics over CFD in this respect, and the significant benefits sensitivity analysis can provide to shape design through the use of gradient-based optimization. His arguments were widely heard, and in the following years much effort was invested in efficient and accurate evaluation of sensitivities.

However, despite considerable progress over 20 years, sensitivity analysis has only recently enjoyed widespread use in engineering practice. There are perhaps two principal causes of this: (i) the questionable suitability of gradient-based optimization methods for many engineering problems, and (ii) the lack of availability of cheap, accurate gradients. A close third might be the extra effort involved in setting up gradient-based optimizations, due to the need for smooth mesh deformation, preconditioning of design variables, and the lack of robustness to the failure of any step of the process. This last is not true of, e.g., genetic algorithms, for which a flow code failure is simply a member of the population that is not evaluated.

Issue (i) is an unavoidable consequence of the locality of gradient-based methods: in a design space with many local optima they will tend to find the nearest (with respect to the starting point), not the best, local optimum. Further there is evidence to suggest that configurations such as multi-element high-lift devices have just such highly oscillatory design spaces [2]. This problem may be tackled with hybrid optimization algorithms that combine non-deterministic techniques for broad searches, with gradient-based methods for detailed optimization, a significant topic of current research [3,4].

However issue (ii) is perhaps more critical in general, and that with which this article is concerned. Inevitably the availability of sensitivity analysis tools for codes lags behind the codes themselves, a situation exacerbated by the considerable effort required to linearize complex solvers. A good example is the lack of reliable adjoint solvers for Navier–Stokes problems until recently. Initially the difficult and limited adjoint formulation in the continuous context slowed development [5], see Section 2.5. Later the stability of iterations for the viscous adjoint in many situations proved problematic [6], examined in Section 4. Even today there are few adjoint solvers that apply linearized turbulence models to large problems – predominantly due to poor stability – instead they rely on constant eddy-viscosity approximations, resulting in an associated gradient error, discussed in Section 3.

The situation deteriorates when multi-disciplinary problems are considered, and the sensitivities of coupled multi-code systems are required. Nonetheless these problems are worth overcoming, as when they are available, gradient-based algorithms combined with adjoint gradients are the only methods which can offer rapid optimizations for extremely large numbers of design variables [7,8], as needed for the aerodynamic shape design of air vehicles.

In this work, therefore we consider each of the principle methods of sensitivity evaluation in detail, and discuss their range of applicability and strengths and weaknesses with reference to published results. We present the derivation of each technique in a uniform notation, and collect references concerning both the history, and recent developments in each method. We also present generalizations to higher-order derivatives and coupled systems, and recent applications to optimization from the authors.

The gradient evaluation problem may be succinctly stated as follows: let $\mathscr{J}(\alpha)$ be a vector of *objective functions* (in fact functionals of the flow solution such as lift and drag) and possibly constraints of dimension $n_{\mathscr{J}}$, where α is a vector of *design variables* of dimension n_α which parameterize the problem, and in particular its geometry. Then the aim is to evaluate $\mathrm{d}\mathscr{J}/\mathrm{d}\alpha$, the gradients of the objective functions and constraints with respect to the design variables, most often for the purposes of optimization using gradient-based methods such as steepest descent and conjugate gradients.

The obvious first approach is to apply finite differences to values of $\mathscr{J}$ evaluated at perturbed values of α, but it will be seen in Section 2.1 that this method has many drawbacks. Other than finite differences, techniques may be divided into two broad categories whose relative efficiency depends on the values of $n_{\mathscr{J}}$ and n_α. In the *direct* approach a linearization of the governing equations is constructed, allowing the sensitivity of the entire flow with respect to each α to be evaluated, given which sensitivities of $\mathscr{J}$ are easy and cheap to construct. The dominant cost in this case is the solution of n_α linear equations. In contrast in the *adjoint* approach a dual of the linearized problem is constructed for each $\mathscr{J}$. The solutions of these dual problems allow easy and cheap evaluation of sensitivities of $\mathscr{J}$ with respect to any design variable. The cost of evaluating the full derivative matrix $\mathrm{d}\mathscr{J}/\mathrm{d}\alpha$ is therefore dominated by the cost of $n_{\mathscr{J}}$ dual linear problems. In aerodynamics the adjoint approach is almost always preferable, as problems are typically characterized by a low number of cost functions (the integrated forces on a body), and a large number of design variables, required to parameterize curves and surfaces in three dimensions; hence $n_\alpha \gg n_{\mathscr{J}}$.

Another distinct classification of sensitivity evaluation techniques is into *continuous* methods, where the governing equations are linearized and adjointed before being discretized, and *discrete* methods where the linearization and adjointing is performed on the discretization of the non-linear system. The issue of which approach is most suitable under which circumstances has been a subject of much research and contention over the past decade, and we devote Section 2.7 to the subject.

Perhaps the first use of this kind of sensitivity analysis in fluid dynamics was in 1973 and due to Pironneau, who derived the continuous adjoint of Stokes equations [9] and later of the incompressible Euler equations [10], applying the result to profile optimization. However it was not until 1988 that Jameson extended the procedure to inviscid compressible flows, allowing the optimization of transonic aerofoil profiles [11], after which point the method became truly widespread. The continuous direct

method in contrast has been seldom considered in an aerodynamic context [12].

The discrete direct method was considered as early as 1982 by Bristow and Hawk [13,14] for panel methods, then in 1989 by Elbanna and Carlson [15] for the compressible Euler equations, followed by Taylor et al. [16] and Baysal and Eleshaky [17] in the early 1990s. The later in the same article also considered the adjoint of the discretized equations, which in the same year were considered by Shubin and Frank [18,19] for one-dimensional duct flow.

Herein, we are concerned purely with the evaluation of gradients; a review of aerodynamic shape optimization techniques may be found in Ref. [20], of multi-disciplinary design in Ref. [21], and of sensitivity analysis and optimization for complex configurations in Ref. [22]. We also note a recent collection of literature in aerodynamic design [23].

The structure of this article is as follows: the various gradient computation methods will be derived in Section 2, and their accuracy under various circumstances discussed in Section 3. Iterative solution methods for the resulting equations, which are as ill-conditioned as the original flow equations are described in Section 4, with particular reference to *duality preserving* methods for the adjoint equations. Section 5 is devoted to recent generalizations of gradient evaluation methods, namely to higher-order derivatives, adjoint mesh deformation, and multi-disciplinary problems. Finally, Section 6 presents two state-of-the-art gradient-based optimization results using adjoint techniques developed by the present authors. Throughout references are presented to facilitate further reading and highlight points of practical relevance and areas of ongoing research.

2. Sensitivity evaluation methods

This section will derive in some detail the principle methods of gradient evaluation, firstly finite differences, followed by discrete and continuous versions of the direct and adjoint methods. We use the convention of uppercase symbols for discrete, and lowercase for continuous quantities, so that W is the solution of the discretized governing equations $R(W,X)=0$ on the mesh X. The discrete adjoint variable will be denoted Λ. The continuous flow and adjoint solutions are w and λ, respectively. Throughout, the design variables α and objective functions $\mathscr{J}(\alpha)=J(W(\alpha),X(\alpha))$ are taken to be vector quantities, so that $\mathrm{d}\mathscr{J}/\mathrm{d}\alpha$ generally represents a matrix of derivatives.

2.1. Finite differences

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

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

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

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

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

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

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

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

are weak constraints on $\|\delta\alpha\|$. In practice however $\mathscr{J}$ is rarely evaluated to machine accuracy, and $\|\delta\alpha\|$ must be increased accordingly. On the other hand too large a value of $\|\delta\alpha\|$ invalidates the neglection of higher-order terms in the Taylor expansion in (1). The choice is case and parameter dependent, and in practice the only means of guaranteeing accuracy is to perform a convergence study on $\|\delta\alpha\|$ for each parameter, at considerable cost.

Another difficulty is that the evaluation of the $\mathscr{J}$s at the two different αs in (1) should be as similar as possible, in order to aid error cancellation. For example, the mesh topology should not be changed, implying some means of mesh deformation is required. It may also be beneficial to use the same initial solution and number of iterations in both cases. For more complex configurations the appearance of multiple solutions can also be an issue [24].

All these issues may be alleviated by using a complex finite difference formula, such as

$$\frac{\mathrm{d}\mathscr{J}}{\mathrm{d}\alpha}\delta\alpha\simeq\Im[\mathscr{J}(\alpha+i\delta\alpha)],$$

where $\Im$ represents the imaginary part. As there is no longer any difference of $\mathscr{J}$s in this expression, it does not suffer from cancellation error, and $\|\delta\alpha\|$ may be chosen as machine zero with no loss of accuracy [25]. However, the solver must be modified to accept complex variables throughout, negating the main advantage of finite differences.

Finite differences have been used since the 1970s in the context of shape optimization. Early contributions include works of Hicks et al. [26] and Vanderplaats and Hicks [27] for aerofoil design, and later wing design [28], using the method of feasible directions in which the lift was held constant by moving in the design space only normal to the lift gradient. Here, the flow was modeled with a small perturbation full potential equation solver. Early contribution also include articles by Reneaux and Thibert [29,30]. As 3D optimization was considered with finite difference gradients, the number of design parameters and the cost of computations became severe problems. In an attempt to reduce the number of parameters needed *aerofunction shapes* were introduced for example by Aidala et al. [31], defined to be aerodynamically meaningful parameterizations of an aerofoils. These might consist of geometry perturbations that control leading edge pressure, or shock position [32,33].

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

2.2. The discrete direct method

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

$$\frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha}=-\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}. \tag{2}$$

This may be regarded as a linear system in unknowns $\mathrm{d}W/\mathrm{d}\alpha$, where $\mathrm{d}X/\mathrm{d}\alpha$ may be evaluated by finite differences as in the previous section, and the partial derivatives could be evaluated, for example, by hand. The dimension of the system is the number of degrees of freedom in the non-linear equations n_W, and it can be regarded as a linearization of those equations. Of course R is seldom differentiable everywhere, but in practice this rarely causes difficulties [34].

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

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}, \tag{3}$$

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

This method was considered as early as 1982 by Bristow and Hawk for a subsonic panel method [13,14], and again in 1989 for the transonic perturbations equations by Elbanna et al. [15]. In the early 1990s it was applied to the compressible Euler equation by two teams at Old Dominion University; that of Baysal [17,35–37] and that of Taylor and Hou [16]. Many of these articles are concerned not only with gradient evaluation, but also the linearization of the solver as a tool for investigating small perturbation to the base flow. On unstructured grids the idea was pursued by Newmann, Taylor et al. from 1995 onwards [38–40,22].

2.3. The discrete adjoint method

There are many ways to derive the discrete adjoint equations, the one given here is chosen for its similarity to the derivation of the continuous adjoint presented in the following sections. Let the direct linearization (2) be premultiplied by an arbitrary line vector Λ^T of dimension n_W, so that

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

Adding this expression to (3)

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \Lambda^T\frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} + \Lambda^T\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right), \quad \forall \Lambda \in \mathbb{R}^{n_W}, \tag{4}$$

and factorizing

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

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

$$\left(\frac{\partial J}{\partial W} + \Lambda^T\frac{\partial R}{\partial W}\right) = 0 \quad \text{or equivalently} \quad \left(\frac{\partial R}{\partial W}\right)^T\Lambda = -\left(\frac{\partial J}{\partial W}\right)^T, \tag{5}$$

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

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

The critical point is that, because α does not appear in (5), that linear system must only be solved once for each J. Hence the full matrix $\mathrm{d}\mathscr{J}/\mathrm{d}\alpha$ may be evaluated at a cost of $n_\mathscr{J}$ linear system solutions, substantially independent of n_α. Note that the issues described in Section 2.2 with regard to evaluation of the partial derivatives above also apply here.

Perhaps the first application of this method was given by Shubin and Frank in 1991 for a quasi one-dimensional nozzle flow using the compressible Euler equations [18,41,19], and was denoted there the "implicit gradient approach" as contrast to the direct approach. Baysal et al. also recognized its potential, and offered it as an alternative to the direct approach when $n_\mathscr{J} \gg n_\alpha$ [17,35–37]. Later examples of the discrete adjoint are given in more specific contexts elsewhere in this article.

2.4. Evaluation of discrete Jacobians

It was simply stated in the previous sections that the partial derivatives in the above expressions, in particular the Jacobian $\partial R/\partial W$, are straightforward to construct. This, while strictly true, misrepresents the substantial effort involved. Although R is typically a known, explicit function of W, it is also typically extremely complex in any non-trivial code, containing flux-functions, reconstruction operators, characteristic boundary conditions, turbulence models, etc., which must all be differentiated and reimplemented. Furthermore, the accuracy of the resulting gradients depends on the completeness and exactness of the linearization; unlike the Jacobian required for an implicit iterative solution scheme for example, which may be simplified arbitrarily without affecting the final result of the computation, only its efficiency. By hand, performing the full linearization is a very considerable, but not completely impractical undertaking [42]. Nonetheless it is sometimes sensible to make simplifying assumptions by neglecting the differentiation of certain terms [43], thereby introducing an error into the gradients which must be examined numerically, see Section 3.

A further complication is that the Jacobian, despite being a sparse matrix, has typically (depending on the discretization stencil size and the number of flow equations) two orders of magnitude more non-zero entries than a solution vector, and if stored explicitly would represent a memory bottleneck in most codes [7]. This storage may be avoided by evaluation of Jacobian-vector products either entirely or partially on-the-fly, and applying solution methods that require only these matrix-vector products. Of course the time required for the evaluation is also important, and partially on-the-fly evaluations offer an effective compromise.

An example of the memory costs involved, from the authors' own experience is given in Table 1 taken from [6], for two linearizations of the unstructured finite volume solver *TAU*, with explicit time-integration. The first implementation stores the full Jacobian explicitly, and the second uses a partially on-the-fly evaluation in which terms of the spatial discretization with a stencil of immediate neighbors only are pre-calculated and stored, whereas those involving next-neighbors are evaluated as needed. The initial non-linear code is provided for reference, and is extremely memory efficient, able to perform a computation on a grid of 1.2 million points within 1 GB of memory. With the full Jacobian the code requires 13.5 times as much memory, which can be reduced to 4 times with the on-the-fly implementation. The CPU cost of a single

Table 1
The memory requirements and run-time of the standard code and two linearized versions of the code for a representative 3D (hybrid 120k points) grid.

	Standard *TAU*	Full Jac.	On-the-fly Jac.
Memory (bytes)	100 M	1350 M	400 M
Factor increase	×1.0	×13.5	×4.0
Points fitting in 1 GB	1.2×10^6	89×10^3	300×10^3
Time for a residual eval.	×1.0	×1.7	×0.6
Time for the Jacobian eval.	–	×28.0	×3.0

linear residual evaluation (Jacobian-vector product), and the cost of building the Jacobian (or partial Jacobian) is given in the final two rows of Table 1. In all cases the costs are of similar *order* to those of the non-linear code.

A technique which avoids the implementation and memory problems associated with the per-hand approach is using finite differences to evaluate Jacobian-vector products directly. As is the case for finite differences applied to the cost function the choice of step-size is vital to the accuracy. Evaluation of the *transpose* Jacobian-vector products needed by the adjoint is more difficult and was investigated by Anderson et al. [25], who used complex differences to maintain accuracy. They found however the cost of a single transpose Jacobian-vector product to be equivalent to about 12 non-linear residual evaluations, an order of magnitude more expensive than the per-hand approach.

Outside the context of this article is the rapidly developing field of *automatic differentiation* (AD) [44,45], which aims to provide tools (compilers or code generators) for automatically linearizing and adjointing computer codes, thereby reliving the development burden involved. We comment only that at the present time we do not know of any applications of such methods to large problems in aerodynamics, possibly due to the fact that the resulting codes tend to require very significantly greater computational resources than hand-differentiated counterparts, despite the existence of good theoretical upper bounds on the costs [44]. A more realistic immediate goal for AD could be in assisting hand-differentiation, for example by automatically differentiating and transposing individual routines [46].

2.5. Derivation of the continuous adjoint equations

In this approach, the adjoint of the continuous governing equations with respect to a given objective function is derived, before being discretized. As already mentioned, the first appearance was due to Pironneau [9], with Jameson providing the first treatment for compressible flows [11], which is similar to that given here. For a more detailed introductory treatment see Abergel and Temam [47], Giles and Pierce [48], and Jameson [49]. The approach has since been reproduced by a variety of authors [50].

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

It is assumed that the problem in physical space with a body-fitted structured grid, can be transformed into computational coordinates (ξ,η), see Fig. 1, in such a way that the transformation of D_{xy} to $D_{\xi\eta}$ is direct, that $D_{\xi\eta}$ is a rectangular domain $[\xi_{\min},\xi_{\max}]\times[\eta_{\min},\eta_{\max}]$ and that $\xi_{\min}$ corresponds to the profile surface. Note that whereas the coordinate change operator depends on α, $D_{\xi\eta}$ itself does not.

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

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

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

$$\frac{\partial F(W)}{\partial\xi}+\frac{\partial G(W)}{\partial\eta}=0, \tag{6}$$

where

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

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

$$\mathscr{J}(\alpha)=\int_{\xi_{\min}}J_1(w)\frac{\partial y}{\partial\eta}\quad \mathrm{d}\eta+\int_{D_{\xi\eta}}J_2(w)K(\xi,\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta, \tag{7}$$

where the domain of integration is now independent of α.

The continuous adjoint equations can now be derived as follows: first write the first variation of the flow equations with respect to the design parameter α. Referring to (6) there are two distinct types of variation: the flux terms $f(w)$ and $g(w)$ vary with α, because the flow changes in the transformed coordinate space when the shape changes, and independently all metric terms also depend on α:

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

Introduce $a(w)=\mathrm{d}f(w)/\mathrm{d}w$ and $b(w)=\mathrm{d}g(w)/\mathrm{d}w$ the flux Jacobians, then the linearized equation corresponding to (6) is

$$\begin{aligned}&\frac{\partial}{\partial\xi}\left\{\left(a(w)\frac{\partial y}{\partial\eta}-b(w)\frac{\partial x}{\partial\eta}\right)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right\}\\ &+\frac{\partial}{\partial\eta}\left\{\left(-a(w)\frac{\partial y}{\partial\xi}+b(w)\frac{\partial x}{\partial\xi}\right)\frac{\mathrm{d}w}{\mathrm{d}\alpha}\right\}\\ &+\frac{\partial}{\partial\xi}\left\{f(w)\frac{\partial^2 y}{\partial\eta\partial\alpha}-g(w)\frac{\partial^2 x}{\partial\eta\partial\alpha}\right\}\\ &+\frac{\partial}{\partial\eta}\left\{-f(w)\frac{\partial^2 y}{\partial\xi\partial\alpha}+g(w)\frac{\partial^2 x}{\partial\xi\partial\alpha}\right\}\\ &=0,\end{aligned} \tag{8}$$

and similarly for (7):

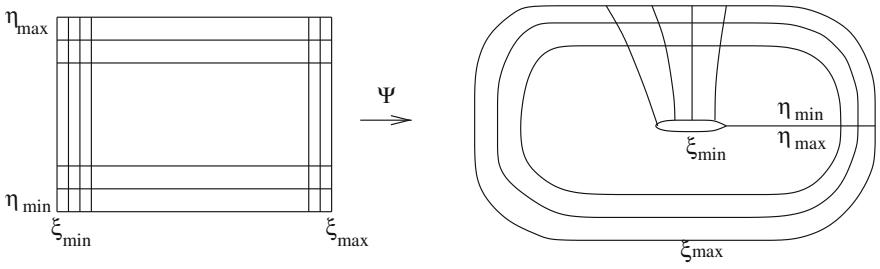

Fig. 1. Body-fitted to computational coordinate domain transformation.

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \int_{\xi_{\min}} \left(\frac{dJ_1(w)}{dw} \frac{\mathrm{d}w}{\mathrm{d}\alpha} \frac{\partial y}{\partial \eta} + J_1(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} \right) \mathrm{d}\eta + \int_{D_{\xi\eta}} \left(\frac{\mathrm{d}J_2(w)}{\mathrm{d}w} \frac{\mathrm{d}w}{\mathrm{d}\alpha} K(\xi,\eta) + J_2(w) \frac{\partial K(\xi,\eta)}{\partial \mathrm{d}\alpha} \right) \mathrm{d}\xi\,\mathrm{d}\eta. \tag{9}$$

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

$$a_1(w,\xi,\eta) = \left(a(w) \frac{\partial y}{\partial \eta} - b(w) \frac{\partial x}{\partial \eta} \right), \quad a_2(w,\xi,\eta) = \left(-a(w) \frac{\partial y}{\partial \xi} + b(w) \frac{\partial x}{\partial \xi} \right). \tag{10}$$

The idea behind the following procedure is to add to (9) the inner product of the linearized governing equations with an arbitrary four-component Lagrange multiplier λ, analogously to the discrete case in Section 2.3. Then we search for a condition on λ for the gradient to be independent of the $\mathrm{d}w/\mathrm{d}\alpha$ terms. In this case we assume that the flow and adjoint solutions, w and λ, are once continuously differentiable with respect to the computational coordinates, i.e., $w, \lambda \in C^1(D_{\xi\eta})^4$. Note that this is a very different restriction to that necessary in the discrete case. Proceeding from (8) we have that

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

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

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

Adding this expression to (9) and extracting terms multiplying $\mathrm{d}w/\mathrm{d}\alpha$ we obtain, from the volume and surface integrals, respectively:

$$\begin{cases} \frac{\mathrm{d}J_2(w)}{\mathrm{d}w} K(\xi,\eta) - \frac{\partial \lambda^T}{\partial \xi} a_1(w,\xi,\eta) - \frac{\partial \lambda^T}{\partial \eta} a_2(w,\xi,\eta) = 0, & \text{on } D_{\xi,\eta}, \\ \lambda^T a_1(w,\xi,\eta) + \frac{\mathrm{d}J_1(w)}{\mathrm{d}w} \frac{\partial y}{\partial \eta} = 0, & \text{on } \xi_{\min}, \end{cases} \tag{11}$$

the *continuous adjoint* equations and boundary conditions.

One very significant feature of this problem is that not all objective functions J_1 lead to a well posed adjoint boundary condition, because the flux Jacobian a_1 is singular at a slip wall [11]. For the compressible Euler equations described here functions of pressure are admissible, which is fortunate given that integral forces on a profile are thereby allowed. On the other hand for the Navier–Stokes equations there is no clear way of accounting for wall shear-stresses, and hence viscous drag [51]. Given a solution of (11) the gradients of $\mathscr{J}$ may be written

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \int_{\xi_{\min}} J_1(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} \mathrm{d}\eta + \int_{\xi_{\min}} \lambda^T \left(f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) \mathrm{d}\eta - \int_{D_{\xi\eta}} \frac{\partial \lambda^T}{\partial \xi} \left(f(w) \frac{\partial^2 y}{\partial \eta \partial \alpha} - g(w) \frac{\partial^2 x}{\partial \eta \partial \alpha} \right) \mathrm{d}\xi\,\mathrm{d}\eta - \int_{D_{\xi\eta}} \frac{\partial \lambda^T}{\partial \eta} \left(-f(w) \frac{\partial^2 y}{\partial \xi \partial \alpha} + g(w) \frac{\partial^2 x}{\partial \xi \partial \alpha} \right) \mathrm{d}\xi\,\mathrm{d}\eta + \int_{D_{\xi\eta}} J_2(w) \frac{\partial K(\xi,\eta)}{\partial \mathrm{d}\alpha} \mathrm{d}\xi\,\mathrm{d}\eta. \tag{12}$$

For the extension to the Navier–Stokes equations, also derived in curvilinear coordinates, see the seminary articles of Jameson et al. of 1997 [5], while a more discursive treatment is given in Ref. [52], which also includes considerations related to the use of the thus obtained gradients in shape optimization.

For finite volumes on unstructured meshes such coordinate transforms as described above are not used, and a formulation in physical coordinates is necessary. One approach was first published by Anderson and Venkatakrishnan in 1998 [51], and 1 year later by Hiernaux and Essers [53,54]. Also of interest is an early adjoint formulation of the thin shear-layer equations in physical coordinates [55].

All theory presented in this section has been under the assumption of continuously differentiable flow and adjoint solutions, and therefore is only valid for flows without shocks. Given a discontinuity which moves due to some parameter perturbation, the change in the solution between the old shock location, and the new one is not small at all. To alleviate this restriction the integrals involved the derivation above may be split into parts downstream and upstream of a shock, before applying the integration by parts formula and imposing the Rankine–Hugoniot condition on the boundary. The extension of the adjoint to discontinuous flows is a difficult topic, and the reader is referred to works by Iollo et al. [56,57], Cliff et al. [58], Giles and Pierce [59] and Gunburger [60], and most especially Pironneau et al. [61–63].

However despite the theory, it should be noted that difficulties with discontinuous solutions have only been seen to occur numerically for special cases, and in transonic design problems they are not an issue.

Finally, we note that the continuous direct formulation, embodied by (8), rarely occurs in the gradient evaluation literature. Pelletier et al. [64,65] applied it to incompressible flows, and the one of few compressible references is of Sharp and Sirovitch for hypersonic profile optimization [12].

2.6. Spatial discretization of the continuous adjoint equations

In the discrete case after derivation of the adjoint one is finished in the sense that the method is completely prescribed, the spatial discretization of the linear problem having been directly derived from that of the non-linear problem. In the continuous case it remains to discretize the adjoint equation (11), and the expression for the gradient (12).

Given the close relationship of the non-linear and adjoint problems, it is reasonable for discretization techniques of the original problem to be applied to the adjoint. In the case of compressible flows for example, upwind or artificial viscosity flux formulations are appropriate, and most authors use similar schemes for flow analysis and gradient computation, and even some of the same subroutines [49]. As a result the discretization expression and implementation tends to be simpler than in the discrete case, where *derivatives* of the discretization are required.

For completeness we present a single example discretization of (11) (and with zero volume component to the objective function, $J_2 = 0$), taken from Ref. [11], and used subsequently without modification by Brezillon and Gauger [66]. A central flux with artificial viscosity is used, on cells in a structured grid denoted by their indices i,j:

$$K_{ij}\frac{\partial \lambda_{ij}}{\partial t} = \frac{1}{2}\left(\hat{a}_1(\lambda_{i+1j} - \lambda_{i-1j}) + \hat{a}_2(\lambda_{ij+1} - \lambda_{ij-1})\right) + d_{i+\frac{1}{2}j} - d_{i-\frac{1}{2}j} + d_{ij+\frac{1}{2}} - d_{ij-\frac{1}{2}},$$

where d are diffusion fluxes on the control volume faces $i+\frac{1}{2},j$, etc., and $\hat{a}_1$ and $\hat{a}_2$ are discretized versions of the flux Jacobians (10), with $a(w)$ and $b(w)$ evaluated at cell centers and

$$\frac{\partial x}{\partial \xi} \simeq \frac{1}{2}\left(\left.\frac{\partial x}{\partial \xi}\right|_{i-\frac{1}{2}j} + \left.\frac{\partial x}{\partial \xi}\right|_{i+\frac{1}{2}j}\right), \quad \frac{\partial y}{\partial \eta} \simeq \frac{1}{2}\left(\left.\frac{\partial y}{\partial \eta}\right|_{ij-\frac{1}{2}} + \left.\frac{\partial y}{\partial \eta}\right|_{ij+\frac{1}{2}}\right),$$

etc., an average of metrics on each side of the cell. Note that because of this last approximation the flux through a face in one direction is in general different to that in the other, and as a consequence the discretization is non-conservative, corresponding to the fact that the adjoint equations are also not conservative. This latter point is easy to overlook given their origin in a conservation equation, and initially it is tempting to try to impose a flux balance using the modified flux $(\partial f/\partial w)^T\lambda$ averaged on cell faces, which naturally fails.

The dominant contributors to the continuous adjoint in aerodynamics must be considered to be Jameson and coworkers, and two detailed descriptions of their approach to discretization may be found in Refs. [5,67]. For the convective terms they use a Jameson–Schmidt-Turkel, Roe, or H-CUSP flux. Viscous fluxes use a cell- vertex approximation of velocity and temperature gradients on a dual mesh. The flow and adjoint equations are separately advanced to a steady state using a Runge–Kutta scheme specifically tuned for each system, respectively, and a multigrid procedure. Other significant contributions have been made by Giles and Pierce [55,48], Weinerfelt and Enoksson [68] and Brezillon et al. [66,50]. The continuous adjoint on unstructured grids has been pursued by Anderson and Venkatakrishnan [51], Soto et al. [69], Jameson et al. [70], Castro et al. [71], and Widhalm et al. [72] among others.

2.7. Discrete versus continuous adjoint

The discrete and continuous formulations of the adjoint problem clearly represent two very different ways of doing essentially the same thing, and the question naturally arises: which method should be preferred in a given situation? This issue proved quite contentious in the early days of adjoint development, where several groups promoted their own approaches, and the accuracy and range of validity of each approach was still unclear. Since that time it has been shown by a variety of authors that both the discrete and continuous formulations give more than sufficient accuracy in practice, and in fact the dominant error source in a typical adjoint gradient evaluation is likely to be the finite difference approximation of the metric sensitivities $\mathrm{d}X/\mathrm{d}\alpha$ [6].[1] As a consequence the question of which method is most appropriate has become largely an implementational one.

Several authors have dealt with the issue directly by comparing codes based on continuous and discrete formulations, notably Giles and Pierce [48] and Nadarajah and Jameson [73]. Among the points raised are the following:

- The discrete adjoint gives the exact gradient of the discrete objective function (agreeing with the results of finite differences), whereas the continuous adjoint gives an approximation to the continuous gradient based on some alternative discretization. Discrete gradients are therefore beneficial for an optimizer, which receives gradient information consistent with objective function evaluations. Also the agreement with finite differences eases debugging and verification of the discrete code.
- A discrete adjoint code may be derived in a purely algorithmic fashion from an existing code, as demonstrated by the existence of *automatic differentiation* tools [44], whereas a continuous formulation often requires domain-specific insight to perform the discretization.
- Because of the close relationship of the discrete adjoint and the flow discretization, it is often possible to modify the iterative solver of the flow problem to give an iterative solver for the adjoint problem that converges at the same asymptotic rate, see Section 4.
- The continuous code is often considerably simpler than the discrete in terms of operation count and memory requirements, as well as implementationally. However it is quite possible to build an efficient discrete code without explicit storage of $\partial R/\partial W$ (at a cost of some small inaccuracy in the gradients), mitigating the first two points [6].
- Mathematical and physical understanding of the adjoint problem is aided by consideration the continuous approach, as exemplified by analytic adjoint solutions for 1D Euler problems [74].
- As already noted in Section 2.5, the continuous equations are only well posed for certain objective functions, most seriously impeding the evaluation of shear–stress drag gradients in viscous flow. In order words boundary conditions do not exist for every objective function in the continuous case, whereas they are automatic in the discrete case.
- Additional source terms in turbulence models complicate the derivation of the continuous adjoint. To the best of our knowledge no such treatment has yet been published.
- Generalization to higher-order derivative evaluation is much easier in the discrete case, as addressed in Section 5.2. The corresponding continuous formulation is too involved to be addressed here.

There have also been efforts towards hybrid schemes [73], where the discrete informs the discretization of the continuous adjoint, or the discrete is used to extend the set of allowed objective functions of the continuous, or the discrete is used for the turbulence equations and the continuous for the mean-flow equations. However in much such situations it is easier to implement a fully discrete code.

One pertinent question is: assuming smooth solutions, do the two approaches converge to the same result as mesh spacing tends to zero? It turns out that a consistent discretization of the flow equations is not sufficient for its adjoint be a consistent discretization of the continuous adjoint equations. A discretization that does have this property is known as *adjoint consistent*, and for finite element methods corresponding conditions on the discretization have been derived [75].

3. Accuracy of sensitivity computations

In many circumstances it is much easier and cheaper to obtain some approximation of the solution of the adjoint problem, and hence the gradient, than it is to obtain the exact (discrete) gradient. The question then arises: how accurate must gradients be for the application under consideration? In optimization, steepest descent and (restarted) conjugate-gradient algorithms have proven very robust to poor quality gradient data [43]. This is not surprising: provided the gradient direction is not more than 90° incorrect

[1] While these may be evaluated to high accuracy if great care is taken and a parameter study performed on the step-size for each design variable, in practice such attention to detail is computationally expensive and difficult to automate.

these methods will produce an improvement in the objective function at each iteration. BFGS on the other hand attempts to construct a local quadratic model of the design space using the gradient, and this process compounds gradient error. Also if gradients are to be used in a response surface method, for example, gradient-enhanced Kriging, then poor gradients will lead to a poor response surface [76]. However that still leaves many applications where only good approximations to gradients are necessary.

This section principally concerns the discrete adjoint, using simplifications of the Jacobian, and their effect on the accuracy of resulting gradients. Given an exact and complete linearization of the discrete flow solver excellent agreement with finite differences is regularly seen in the literature [77,43,48]. However, because of the effort required to construct this linearization for all but the most modest flow solvers, it is perhaps not surprising that in a large proportion of references some simplifying approximation is made, thereby affecting the accuracy of the gradient. In the continuous adjoint, approximations are also made due to the difficulty of the treatment of turbulence models, which are not always formulated as continuous equations. In fact the neglection of the linearization of turbulence models, commonly denoted the *frozen eddy-viscosity* approximation, is so widely used that it may be considered standard, and is treated in detail in Section 3.1. Several other common approximations are discussed briefly in Section 3.2.

Apart from these explicitly added sources of inaccuracy there are more fundamental and unresolved issues regarding the mesh convergence of adjoint gradients. Giles et al. [78] considered a case with a strong shock which wandered between mesh points as a design parameter was varied. The lift of the aerofoil was seen to be somewhat dependent on the position of the shock within the local mesh cell, as a result of which the design parameter-lift curve took on a slightly scalloped shape. As the mesh was refined the amplitude of the scalloping was reduced as expected – but as it always took the same shape the amplitude of its *gradient* was not reduced. Fortunately the effect was not seen for cases with weaker shocks, and has not appeared elsewhere, although we are not aware of any other attempts to demonstrate grid convergence of gradients, adjoint or otherwise.

3.1. Fully linearized or frozen turbulence modeling

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

(i) The algebraic model of Baldwin–Lomax was differentiated by, for example, Le Moigne and Qin [79] and Kim et al. [80], both of whom used their method for profile optimization. Pham [81] adjointed the algebraic Michel model for evaluating sensitivities in three dimensions in turbomachinery.
(ii) The one-equation model of Spalart–Allmaras was linearized by under others Giles et al. [82], and the team of Anderson, Nielsen, and Bonhaus at NASA Langley [83–85] who compared resulting gradients with frozen eddy-viscosity gradients [77]. The complete linearized code was applied to the 3D optimization of an isolated wing. Further examples include Nemec and Zingg [86] and Brezillon and Dwight [87] who both presented profile optimizations, in the latter case with multiple design points and constraints.
(iii) The two-equation transport models k–ϵ, k–ω SST. and Wilcox k–ω have been differentiated and applied to sub- and transonic profile design, as well as high-lift profile and setting optimization by Kim et al. [88,89]. The former model has also been adjointed in the context of turbomachinery by Renac et al. [90,91].

There is however a notable lack of linearized transport equation turbulence models applied to configurations significantly more complex than isolated wings with fully attached flow [77] or 2D high-lift profiles [89]. We suspect this is consequence, not of the difficulty of performing the linearization itself or accounting for the coupling with the mean-flow, but of the problems associated with the solution of the resulting linear system, which may be exceptionally poorly conditioned [42].

In any case in a very large number of articles the turbulence is fixed. In both the continuous and discrete cases this is achieved simply by adjointing only the mean-flow equations and treating turbulent quantities appearing therein as constants. For one-equation models these are just the eddy-viscosity, for two-equation models the turbulent kinetic energy k also appears in the internal energy and hence in the pressure. Hence in addition simplifying the solution of the system, the number of equations to be solved is reduced. Notable examples of this approach are due to Jameson et al. [92], Soemarwoto [93], and Valentin [94].

There are additionally several authors that are principally interested in AD, and the linearization of turbulence models as a secondary issue: Hou et al. [95] for Baldwin–Lomax, and Mohammadi [96] for a k–ϵ model. As a related point of interest, Bischof et al. [97] used an AD tool in forwards mode to determine the sensitivity of flow over a backward-facing step (particularly the re-attachment point) to empirical parameters of several turbulence models, the idea being that a large sensitivity to an experimentally determined parameter is a weakness of the model.

3.2. Other approximations

After frozen eddy-viscosity, perhaps the second most common approximation is that of neglecting the derivatives of artificial diffusion coefficients in convective terms when using a centered convective flux approach [98,43,85], which has almost no effect on the gradient while allowing a considerable simplification of the corresponding linearized terms.

A much stronger approximation is the use of first-order convective fluxes in the linear calculation [99–103], which has the advantage of reducing the fill-in of the system matrix to immediate mesh neighbors only, and reducing the stiffness of the problem dramatically.

In some articles authors have linearized a simplified viscous flux [104,105], and in a very limited number of articles neglected viscous terms completely [67,106], although this latter has several serious complications (no-slip boundary conditions must be replaced, and a separate Euler mesh is necessary for the adjoint).

Perhaps the most extreme "approximation" – which can barely be called adjoint at all – was suggested by Mohammadi [107–109], who proposed ignoring *all* aerodynamic contributions to the gradient on the basis that in some very particular cases metric terms dominate, i.e.

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha} \simeq \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}, \tag{13}$$

thereby completely obviating calculation of the aerodynamic adjoint. Mohammadi suggested this might be useful in the case where the objection function is an integral of a boundary quantity, and the design parameters deform the shape of that boundary. However at best a gradient vector qualitatively resembling the correct vector will be obtained. Subsequently Kim et al. [110] compared the idea with a full gradient for the design of supersonic transport high-lift devices, and discovered that for their case, while giving reasonable results at the leading edge it performed poorly at the trailing edge, where aerodynamic variations were stronger.

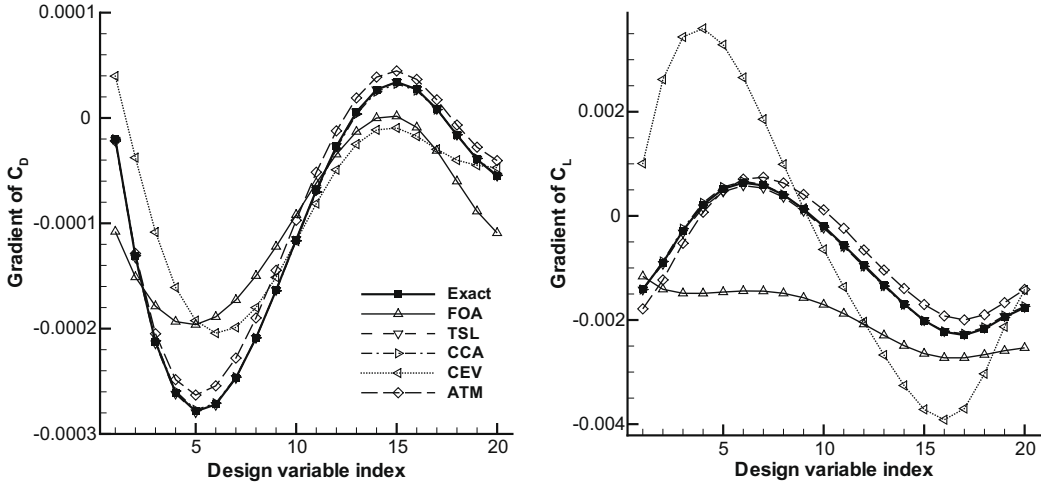

Fig. 2. Gradients of shape design variables for a transonic RAE2822, obtained using a variety of approximate adjoints. Key, the adjoint uses: FOA, first-order fluxes; TSL, simplified viscous fluxes; CCA, frozen dissipation coefficients; CEV, frozen eddy-viscosity; ATM, alternative turbulence model.

3.3. Impact on, and influence of gradient accuracy

Given the widespread use of adjoint approximations there have been relatively few published studies examining the effect on the gradient of various approximations. Nielsen and Anderson [103,77] examined the effect of first-order flux and frozen eddy-viscosity approximations for a high-lift configuration, and observed errors in gradients of several design variables of typically 90% with the former and 30% with the later method. On this basis they rejected these approximations.

However a pertinent question in this context is: how important is the accuracy of the gradient to the optimization speed and accuracy? Kim et al. [89,111] tackled this question for the frozen eddy-viscosity assumption by performing the same optimization using gradients from an adjoint with and without the approximation. They observed significant discrepancies in the gradient that for some cases resulted in significantly different optimal geometries when using a BFGS optimizer, which builds an approximation to the Hessian from the gradients.

A similar study [43] that considered five separate adjoint approximations, showed on the other hand that, provided a robust optimizer is used (in this case conjugate gradients (CG)), then even very poor gradients result in almost identical optima to highly accurate gradients. This effect is shown in Figs. 2 and 3 reproduced from Ref. [43]. The first figure shows the variation in gradient for a transonic RAE2822 with 20 shape design variables, for the adjoint approximations considered, where it is evident that, e.g., frozen eddy-viscosity introduces a large error. The second shows the convergence of drag minimization optimizations using these gradients: in every case the same optimum is reached to within an accuracy of 4%. Similar results hold for high-lift optimization, and – consistent with the results of Kim – the use of the more gradient sensitive BFGS damages this independence. This result perhaps explains in part the use of and reliance on these approximations, though inaccurate gradients do complicate and decrease the robustness of the design iteration, and there is some problem dependence [8].

4. Solution strategies for the primal and adjoint equations

The discrete adjoint and direct equations are both linear problems, so it is somewhat counter-intuitive that – for large test cases – they are as hard to solve as the original flow equations and often significantly harder. This is due to the demands of stability: iterations applied to linear problems never stall as is common for flow problem, if they do not converge asymptotically they diverge, combined with the difficulty of CPU and memory efficient evaluation of the linearized and adjoint residual.

From the simplest perspective the (discretized) continuous or discrete adjoint equations are nothing other than sparse linear

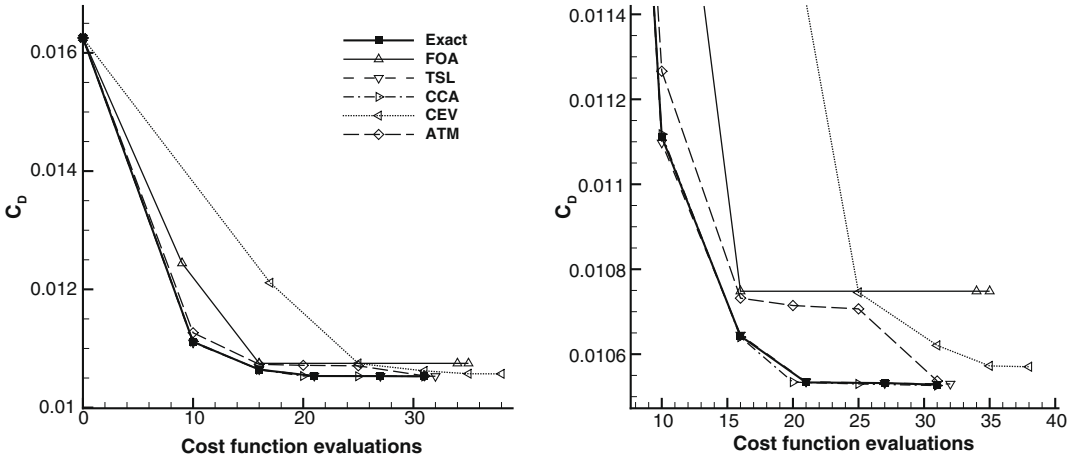

Fig. 3. Convergence of conjugate gradient transonic RAE2822 optimization using gradients from a variety of approximate adjoints. Symbols indicate gradient evaluations.

systems, and so in principle amenable to the standard solution techniques for such problems, for example, Jacobi, Gauss–Seidel or incomplete lower–upper (ILU) factorization preconditioned Krylov methods. Indeed among authors who first considered discrete sensitivity analysis, *direct* inversion was possible – because of the small size and banded structure of the Jacobians considered – and popular [17,35,36]. The straightforward application of ILU preconditioned GMRES to the adjoint problem has been advocated by Nemec and Zingg [86], who apply it in a similar manner within a Newton method for the flow problem. This technique can be extremely efficient, giving a method that for 2D cases provides a solution in about 5% of the CPU time required for the flow calculation if ILU(4) is used [43]. However the memory requirements can be 10–30 times those of the original solver due to the storage of the factorized matrix and the Krylov basis, and the effectiveness of ILU as a preconditioner has been seen to decrease rapidly with problem size and stiffness – most particularly in three dimensions – requiring an increase of the fill-in level to impractical values [43,86].

These difficulties have been alleviated somewhat by several authors including Newman et al. [112], Nielsen et al. [103] and Nemec and Zingg [113], who all applied ILU (usually ILU(0)) to a linearization of a first-order accurate flow discretization. The idea is that if

$$\frac{\partial R^{1\text{st}}}{\partial W}\left(W^{n+1}-W^{n}\right)=-R(W^{n}),$$

is an effective iterative method for the non-linear problem, where $\partial R/\partial W^{1\text{st}}$ is some approximate simplified Jacobian, then

$$\left[\frac{\partial R^{1\text{st}}}{\partial W}\right]^{T}\left(\Lambda^{n+1}-\Lambda^{n}\right)=\frac{\partial J}{\partial W}^{T}-\frac{\partial R}{\partial W}^{T}\Lambda^{n}, \tag{14}$$

should be an effective method for the adjoint problem. The use of the first-order Jacobian on the LHS reduces the stiffness of the linear system to be solved at each iteration so that ILU(0) rather than ILU(4) or higher is sufficient. (As an aside an interesting paper on the conditioning of $\partial R/\partial W$ for upwind and central convective flux discretizations with and without low Mach number preconditioning is due to Baysal and Eleshaky [17].)

The general principle is that because the flow and adjoint problems are closely related in a very particular way, if an iterative method is effective for the non-linear problem, then some suitable "adjointing" of that method should be effective for the adjoint problem. In the example above this adjointing is performed simply by the transpose operation on the LHS of (14). The principle was formalized by Giles [114], and is discussed in detail in the following section. It is of particular importance because the efficient solution of flow problems has been a subject of intense research since the beginnings of CFD, and a wide variety of effective algorithms have been developed such as multigrid, Runge–Kutta with residual smoothing, and approximately factored implicit methods, all of which may be adjointed and applied to the adjoint problem with the expectation of identical convergence behavior. Increasingly therefore variants of iterations used in the non-linear case are applied to the adjoint case in practice.

This is not quite the end of the story however, as there are regularly situations in practice where the original iterative method for the flow equations stalls, implying divergence for the corresponding adjoint iteration. This may be remedied by reintroducing a Krylov method as a form of stabilization, see Section 4.2. With these techniques the solution of the adjoint problem can be made as robust and efficient as that of the original flow problem, and two examples are given.

4.1. Duality preserving fixed-point iterations

An insight due to Giles [114,82,78] based on work by Christianson [115] instructs the construction of dual iterations that give rates of convergence of the adjoint problem identical to those of the non-linear problem. The method has since been widely adopted [116,98,6].

In essence, if a particular fixed-point iteration has a given *asymptotic* rate of convergence for a given non-linear problem, the same iteration applied to the corresponding linearized problem will have the same rate of convergence, assuming that the linearization is exact, complete, and based on the fully converged non-linear solution. In practical applications this is a very desirable property: given a successful non-linear computation a linear computation can be immediately performed with the same iterative method (and CFL number), and convergence within a fixed number of iterations is guaranteed. It will now be seen how this property may be extended to the adjoint equations.

A general fixed-point iteration, including, e.g., Runge–Kutta, multigrid and implicit schemes, may be written

$$M(W^{n+1}-W^{n})=-R(W^{n}), \tag{15}$$

for some *preconditioning* matrix M. By rearranging (15), and linearizing about the exact solution $\widetilde{W}$, we have

$$W^{n+1}=(I-M^{-1}A)W^{n}+g(\widetilde{W})+\mathcal{C}\|W^{n}-\widetilde{W}\|^{2}, \tag{16}$$

where $A=\partial R/\partial W$ is the Jacobian evaluated at the exact solution and $g(\cdot)$ is some function independent of n. A close relation to the linearized problem is apparent, and informed by this we consider the fixed-point iteration (FPI)

$$M(\Theta^{n+1}-\Theta^{n})=\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right)-A\Theta^{n}, \tag{17}$$

or rearranged:

$$\Theta^{n+1}=(I-M^{-1}A)\Theta^{n}+M^{-1}\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right). \tag{18}$$

The coefficients of W^n and Θ^n in these two iterations are identical, so we can conclude that – for a sufficiently converged non-linear iteration – the errors reduce at the same rate; i.e., the asymptotic convergence behavior is identical. The rate of convergence itself is given by the dominant eigenvalue of $(I-M^{-1}A)$.

To achieve the same for the adjoint equation consider the FPI

$$M^{T}(\Lambda^{n+1}-\Lambda^{n})=\left[\frac{\partial J}{\partial W}\right]^{T}-A^{T}\Lambda^{n}, \tag{19}$$

so that

$$\Lambda^{n+1}=(I-M^{-T}A^{T})\Lambda^{n}+M^{-T}\left[\frac{\partial J}{\partial W}\right]^{T}. \tag{20}$$

where M^{-T} denotes the inverse transpose of M. Since any real matrix and its transpose have identical eigenspectra, the asymptotic convergence rate of (20) will be the same as that of (18). For implicit methods constructing the adjoint FPI is easy when M is explicitly available [117]; Runge–Kutta may be treated by first constructing M as a matrix, transposing, and then deconstructing [118].

A further convergence statement may be made, by requiring that $\Theta^0=\Lambda^0=0$ and expanding (18) and (20) to the Nth iteration. Then we have, respectively

$$\Theta^{N+1}=-\sum_{n=0}^{N}(I-M^{-1}A)^{n}M^{-1}\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right), \tag{21}$$

$$\Lambda^{N+1}=-\sum_{n=0}^{N}(I-M^{-T}A^{T})^{n}M^{-T}\left[\frac{\partial J}{\partial W}\right]^{T}, \tag{22}$$

given which it quickly follows that

$$\left[\Lambda^{N+1}\right]^{T}\left(\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}\right)=\left[\frac{\partial J}{\partial W}\right]^{T}\Theta^{N+1}, \tag{23}$$

i.e., that the primal and adjoint results for $\mathrm{d}J/\mathrm{d}\alpha$ will be identical not only after the equations have fully converged, but also at every

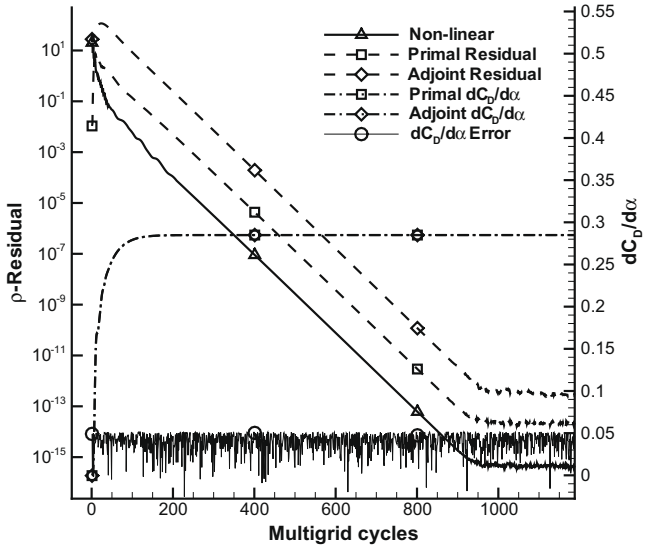

Fig. 4. Convergence of non-linear, primal, and adjoint codes for an NACA0012 test case. Also shown are transients of $dC_D/d\alpha$.

intermediate step. This relation is useful in verifying the correctness of the adjoint FPI and discretization [118,98]. As a numerical example of these results see Fig. 4 reproduced from [6]. The gradients produced by the primal and adjoint codes are identical to machine accuracy at every iteration, as well as the convergence rates of both methods being identical.

4.2. Krylov stabilization

Despite the guarantees regarding convergence provided by the theory of the previous section for both the primal and adjoint equations, for problems involving turbulence modeling there are regularly situations in which it is possible to obtain a reasonably converged solution of the non-linear equations, but not of the corresponding linear equations. This can occur for two reasons, both of which violate the parity of (15) and (17). Firstly there could a discrepancy between the linear and non-linear problem, for example, due to a Jacobian approximation. This effect leads to some prominent authors to advocate the use of only the exact linearization when constructing the adjoint [116,119]. However, there is a second difficulty: the FPI applied to the non-linear problem might not converge asymptotically itself, but rather stall in some limit cycle, which in fact is almost the standard situation in complex 3D applications. This behavior does not necessarily indicate a deficient iterative method, but may often be traced to physical instationary behavior that is not correctly captured by the stationary flow approximation.

In such a position one simple solution is to add extra artificial dissipation to the adjoint problem which is not present in the flow problem. This may be done explicitly [104,91], or indirectly by linearizing only first-order convective fluxes as already discussed [99,43]. The remarks of Section 3.3 regarding the accuracy of resulting gradient apply, and the technique tends only to delay the point at which the iteration starts to diverge rather than removing the instability altogether.

A preferable alternative is to apply a Krylov method. Such methods have been in use for some time in the context of linearized flow solvers [120,113], but only recently have they come to be regarded as a stabilization technique for an existing iteration, rather than simply as a generic linear solver. This perspective was first advanced in the context of sensitivity analysis in CFD by Campobasso and Giles [121,122]. Campobasso showed on the basis of eigensystem analysis that the number of diverging modes in the linearized problem is generally quite small, of the order of 10 s, even for problems with millions of degrees of freedom. The standard iterative method (which may be a duality preserving iteration in the adjoint case) damps all other modes more-or-less effectively, but the presence of even one diverging mode is fatal. Krylov methods identify the most strongly diverging modes of a system and treat them specially, in the case of GMRES [123] by directly minimizing the residual on the subspace spanned by those modes, in the case of the recursive projection method (RPM) [124,125] by using a Newton iteration. Provided all diverging modes are so treated, the resulting iteration will be stable.

Campobasso chose RPM over the more commonly used GMRES, as it has the property of only adding a vector to the Krylov basis (to be specially treated) when an instability is detected, thus minimizing the size of the basis required for stability and thereby the memory overhead. Note that for many Krylov methods (including GMRES and RPM) convergence behavior is a function only of the distribution of the eigenvalues of the system matrix, and therefore transposing the matrix has no effect on convergence. Hence such Krylov methods are in some sense automatically duality preserving. This approach has been applied since by several groups [7] including the current second author [6].

The effect on convergence of applying a Krylov solver can be quite dramatic, as shown in Fig. 5 for adjoint high Reynolds number viscous flow about a DLR-F6 transport aircraft wing-body configuration. The short dotted line shows the rapidity with which the duality preserving iteration diverges without stabilization. With RPM activated, initially the calculation proceeds without modification until a diverging mode is identified. When found this mode is added to the solution subspace P, which will be treated with a Newton method, while the complementary subspace Q will be treated with the standard iteration as before. Shown in Fig. 5 are the flow residuals on each of the subspaces P and Q, as well as the dimension of P and the convergence of an integral quantity derived from the adjoint solution, $dC_D/d\alpha$. The jumps in the residuals correspond to the identification of a diverging mode, and subsequent modification of P and Q. As can be seen only 12 modes are needed in P to bring the calculation to machine accuracy.

Furthermore, if after a while RPM does not identify any more diverging modes, it will add modes which converge most slowly

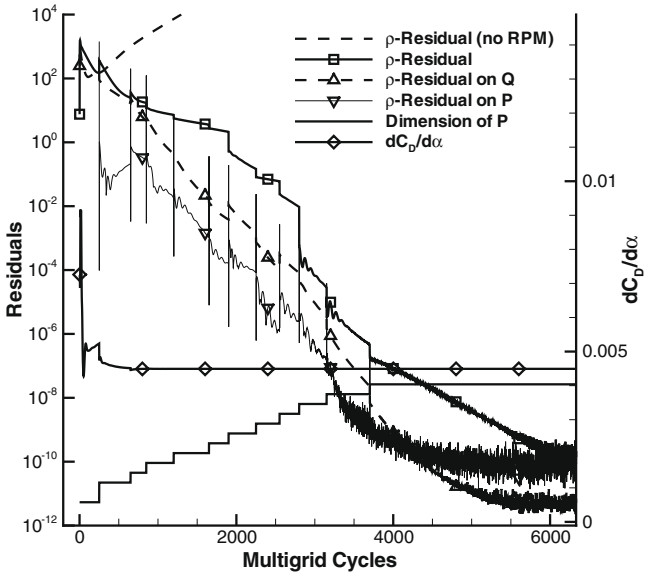

Fig. 5. Stabilization of a high-Reynolds number DLR-F6 adjoint computation with RPM. The diverging dashed line shows a duality preserving iteration without RPM, the remaining lines show the convergence with RPM. Also plotted is the increasing dimension of the RPM P subspace (without scale).

to P, thereby improving the asymptotic convergence rate, an effect which can be observed in the difference in convergence rate before and after the 2000th iteration in Fig. 5.

5. Generalizations of gradient evaluation

Up until now we have considered the evaluation of first derivatives of quantities only dependent on the flow. We discuss now three generalizations: extension of the adjoint procedure to include the mesh deformation, to higher-order derivatives, and to coupled structure–fluid systems. The extension to unsteady flows, including optimization of time-dependent problems, has recently been performed by Mani and Mavriplis [126], and will not be discussed here.

5.1. Adjoint mesh deformation

Through the use of adjoint flow equations the cost of a gradient evaluation for (typically) 50 design variables or more becomes dominated by the cost of deforming the mesh for each design variable, necessary for the evaluation of the metric sensitivities by finite differences. In the continuous case a partial solution to this is given by approximate surface formulations of the volume integral for the gradient, first proposed by Weinerfelt and Enoksson [68], which require only deformation of the surface, but which come with a loss of accuracy. In the discrete case there is a more elegant and exact solution due to Nielsen and Park [119] who suggested constructing additionally the adjoint of the mesh deformation equations.

Let the mesh X satisfy $D(X,\alpha)=0$. Differentiating this equation with respect to α, multiplying by a mesh deformation adjoint variable Γ, and adding to (4) from Section 2.3, which already contains the linearization of the flow equation gives:

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha}=\frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}+\frac{\partial J}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha}+\Lambda^T\frac{\partial R}{\partial W}\frac{\mathrm{d}W}{\mathrm{d}\alpha}+\Lambda^T\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}+\Gamma^T\frac{\partial D}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}+\Gamma^T\frac{\partial D}{\partial \alpha}\qquad\forall\Lambda,\Gamma\in\mathbb{R}^{n_W}$$

As before, $\mathrm{d}W/\mathrm{d}\alpha$ can be eliminated by factorizing it out and setting its coefficient to zero using Λ, giving the familiar flow adjoint equation. However now the same may be done for $\mathrm{d}X/\mathrm{d}\alpha$ using Γ, giving

$$\frac{\partial D}{\partial X}^T\Gamma=-\frac{\partial J}{\partial X}^T-\frac{\partial R}{\partial X}^T\Lambda,$$

where Λ is already known, an adjoint mesh deformation equation. Note that the additional partial derivatives $\partial R/\partial X$ and $\partial J/\partial X$ must be available. Similar difficulties apply as for the evaluation of $\partial R/\partial W$, with the significant difference that the metric sensitivities must only be evaluated once, and hence efficiency is of reduced importance.

Given Γ the objective function derivative becomes

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha}=\Gamma\frac{\partial D}{\partial\alpha}^T.$$

Hence $n_{\mathscr{J}}$ adjoint mesh deformations must be performed rather than n_α direct mesh deformations. The gradient evaluation is almost completely independent of the number of design variables, allowing optimization of 3D configurations using all surface mesh points as design variables [119]. Mavriplis [7] did the same for a 3D transport aircraft wing-body design problem, using this technique among others to perform a complete Navier–Stokes drag minimization optimization in about 6 h on a small cluster of eight processors. The idea has also recently been used by Jakobsson and Amoignon [127], who applied a single volume spline for both mesh deformation and geometry parameterization. Differentiating the spline allowed construction of the gradients with respect to the design variables directly.

Alternatively the gradient computation can be split in two parts, making the aerodynamic gradient post-processing and adjoint equation resolution fully independent of the geometric design process. This technique is a consequence of the relation

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

whereby the two factors on the right-hand side may be calculated independently. The drawback of the method compared to more standard discrete gradient computation method is that the matrix $\frac{\partial R}{\partial X}$ is needed explicitly, rather than just the product $\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha}$, which can be easily estimated by finite differences.

5.2. Higher-order derivatives using discrete methods

Higher-order derivatives have applications in optimization algorithms based on Newton methods, where a root of the gradient of $\mathscr{J}$ is sought, and hence second-derivatives of $\mathscr{J}$ are required. There are of course many applications of higher-order Taylor expansions of flow solvers, for example, response surface construction. Another use of great recent interest is in uncertainty analysis, where it is often desirable to compute the probability distribution of an output variable given those of several input variables. Using a first-order expansion the mean and variance of the output distribution may be modeled, but second-order is necessary to account for skewness, third-order for kurtosis, and so on.

Here a generalization of the adjoint treatment to second-order derivatives only is considered, the associated development, accuracy, and efficiency problems already being very significant, and becoming much worse for higher derivatives.

As in Section 2, we consider the effort (in terms of linear system solutions required) to evaluate the full tensor of second-derivatives for several approaches. Whereas there the choice was between direct or adjoint, here there are four possibilities, corresponding to the two methods of solving for the first and second derivatives. These were first investigated by Sherman et al. [128], and we follow his convention of denoting them DD.DD, DD.AV, AV.DD, and AV.AV, respectively, where DD stands for "direct differentiation" and AV "adjoint vector". The objective functions, flow solutions and discrete flow equations are now assumed to be locally twice differentiable with respect to their arguments.

Because of the complexity of the vector algebra we introduce i,j, to index degrees of freedom of the flow solution, m,n those of the mesh, and a,b,c to index entries in the objective function and design variable vectors, and use summation convention. Then (3) becomes

$$\frac{\mathrm{d}\mathscr{J}_c}{\mathrm{d}\alpha_a}=\frac{\partial J_c}{\partial X_m}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}+\frac{\partial J_c}{\partial W_i}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a},$$

and second derivatives are

$$\begin{aligned}\frac{\mathrm{d}^2\mathscr{J}_c}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}=&\frac{\partial^2 J_c}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}+\frac{\partial J_c}{\partial X_m}\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\\&+\frac{\partial^2 J_c}{\partial W_i\partial X_m}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)\\&+\frac{\partial^2 J_c}{\partial W_i\partial W_j}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}+\frac{\partial J_c}{\partial W_i}\frac{\mathrm{d}^2W_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b},\end{aligned}\tag{24}$$

so that there are then $n_\alpha(n_\alpha+1)/2$ second-order flow derivatives $\mathrm{d}W/\mathrm{d}\alpha\mathrm{d}\alpha$ required to evaluate the second-derivative tensor.

5.2.1. DD.DD

The most direct approach is to solve (2) for each $\mathrm{d}W/\mathrm{d}\alpha$ term, and derive the corresponding second-order equation by differentiating (2) with respect to α:

$$\begin{aligned}\left(\frac{\partial R}{\partial W_i}\right)\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a} &= -\frac{\partial R}{\partial X_m}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a},\\ \left(\frac{\partial R}{\partial W_i}\right)\frac{\mathrm{d}^2W_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b} &= -\frac{\partial^2 R}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}\\ &\quad -\frac{\partial R}{\partial X_m}\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}-\frac{\partial^2 R}{\partial X_m\partial W_i}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)\\ &\quad -\frac{\partial^2 R}{\partial W_i\partial W_j}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b},\end{aligned} \tag{25}$$

giving a total of $\frac{n_\alpha(n_\alpha+1)}{2}+n_\alpha$ linear systems to be solved. Once found the flow derivatives may be substituted into (24) to obtain the desired gradient.

At this point it is worth examining the effort involved in applying this method. First Eq. (25) must be constructed requiring the evaluations of terms of the form $\partial R/\partial W\partial W$, etc. and $\mathrm{d}X/\mathrm{d}\alpha\mathrm{d}\alpha$. As before, the former are – in principle – easy to evaluate, R and J being known, explicit functions of their arguments. However, it is now hard to see how the derivation could be performed by hand, and automatic differentiation tools seem to offer the only practical approach. Note also that explicit storage of the tensor $\partial R/\partial W\partial W$ in memory is likely to be impossible for all but the smallest problems. As for the metric derivatives, evaluation using finite differences is extremely step-size sensitive, and will likely lead to large errors. In addition the mesh must be deformed $\mathcal{O}(n_\alpha^2)$ times, though this may be avoided by adjointing the mesh deformation equation as in Section 5.1.

Given these difficulties the numerical solution of the resulting linear systems is a very secondary problem. Though in DD.DD the number of systems is quadratic in n_α we can exploit the fact that they all have the same system matrix, and possibly factorize $\partial R/\partial W$ if the problem is small. In the following it will be shown how the AV formulations can serve to reduce the number of systems to be solved, but they do nothing to help the significant problems in formulation; nonetheless these methods have been implemented and applied to problems in aerodynamics [128].

5.2.2. DD.AV

The second method consists in adding to (24) the product of Eq. (25) by an arbitrary adjoint vector Γ in a similar procedure to that in Section 2.3. Γ is then chosen to cancel the second-order flow sensitivity $\mathrm{d}W/\mathrm{d}\alpha_a\mathrm{d}\alpha_b$:

$$\begin{aligned}\frac{\mathrm{d}^2\mathscr{J}}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b} &= \frac{\partial^2 J}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}+\frac{\partial J}{\partial X_m}\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\\ &\quad +\frac{\partial^2 J}{\partial W_i\partial X_m}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)\\ &\quad +\frac{\partial^2 J}{\partial W_i\partial W_j}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}+\frac{\partial J}{\partial W_i}\frac{\mathrm{d}^2W_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\\ &\quad +\Lambda^T\frac{\partial R}{\partial W_i}\frac{\mathrm{d}^2W_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}+\Lambda^T\frac{\partial^2 R}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}\\ &\quad +\Lambda^T\frac{\partial R}{\partial X_m}\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}+\Lambda^T\frac{\partial^2 R}{\partial X_m\partial W_i}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)\\ &\quad +\Lambda^T\frac{\partial^2 R}{\partial W_i\partial W_j}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}.\end{aligned} \tag{26}$$

The adjoint equation to solve turns out to be the same as for the first-order gradient computation,

$$\frac{\partial R}{\partial W}^T\Lambda=-\frac{\partial J}{\partial W}^T,$$

where Λ is independent of α again. Having solved the n_α direct equations for the $\mathrm{d}W/\mathrm{d}\alpha$ terms, and $n_\mathscr{J}$ adjoint equations for the Λ, i.e., $n_\alpha+n_\mathscr{J}$ linear systems, the derivative tensor is

$$\begin{aligned}\frac{\mathrm{d}^2\mathscr{J}}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b} &= \frac{\partial^2 J}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}+\frac{\partial J}{\partial X_m}\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\\ &\quad +\frac{\partial^2 J}{\partial W_i\partial X_m}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)+\frac{\partial^2 J}{\partial W_i\partial W_j}\\ &\quad \times\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}+\Lambda^T\frac{\partial^2 R}{\partial X_m\partial X_n}\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_n}{\mathrm{d}\alpha_b}+\Lambda^T\frac{\partial R}{\partial X_m}\\ &\quad \times\frac{\mathrm{d}^2X_m}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}+\Lambda^T\frac{\partial^2 R}{\partial X_m\partial W_i}\left(\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_b}+\frac{\mathrm{d}X_m}{\mathrm{d}\alpha_b}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\right)\\ &\quad +\Lambda^T\frac{\partial^2 R}{\partial W_i\partial W_j}\frac{\mathrm{d}W_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b},\end{aligned} \tag{27}$$

where no $\mathrm{d}W/\mathrm{d}\alpha\mathrm{d}\alpha$ terms appear.

5.2.3. AV.DD

We consider AV.DD for completeness, despite the fact that it is always more costly than DD.AV. The starting point for AV.DD and AV.AV is the familiar adjoint expression

$$\frac{\mathrm{d}J}{\mathrm{d}\alpha_a}=\frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha_a}+\Lambda^T\frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha_a},\quad \frac{\partial R}{\partial W}^T\Lambda=-\frac{\partial J}{\partial W}^T,$$

which defines Λ as a function of $W(\alpha)$ and $X(\alpha)$, and hence α. Differentiating both expressions with respect to a design parameter α_b therefore gives

$$\begin{aligned}\frac{\mathrm{d}^2J}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b} &= \frac{\partial^2 J}{\partial X_i\partial X_j}\frac{\mathrm{d}X_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_j}{\mathrm{d}\alpha_b}+\frac{\partial^2 J}{\partial X_i\partial W_j}\frac{\mathrm{d}X_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}+\frac{\partial J}{\partial X_i}\frac{\mathrm{d}^2X_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\\ &\quad +\left(\frac{\mathrm{d}\Lambda^T}{\mathrm{d}\alpha_b}\right)\frac{\partial R}{\partial X_i}\frac{\mathrm{d}X_i}{\mathrm{d}\alpha_a}\\ &\quad +\Lambda^T\left(\frac{\partial^2 R}{\partial X_i\partial X_j}\frac{\mathrm{d}X_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}X_j}{\mathrm{d}\alpha_b}+\frac{\partial^2 R}{\partial X_i\partial W_j}\frac{\mathrm{d}X_i}{\mathrm{d}\alpha_a}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}+\frac{\partial R}{\partial X_i}\frac{\mathrm{d}^2X_i}{\mathrm{d}\alpha_a\mathrm{d}\alpha_b}\right),\end{aligned} \tag{28}$$

and

$$\begin{aligned}\frac{\partial R_i}{\partial W_l}\frac{\mathrm{d}\Lambda_i}{\mathrm{d}\alpha_b} &= -\frac{\partial^2 R_i}{\partial W_l\partial W_j}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}\Lambda_i-\frac{\partial^2 R_i}{\partial W_l\partial X_j}\frac{\mathrm{d}X_j}{\mathrm{d}\alpha_b}\Lambda_i\\ &\quad -\frac{\partial^2 J}{\partial W_l\partial W_j}\frac{\mathrm{d}W_j}{\mathrm{d}\alpha_b}-\frac{\partial^2 J}{\partial W_l\partial X_j}\frac{\mathrm{d}X_j}{\mathrm{d}\alpha_b}.\end{aligned} \tag{29}$$

Required are therefore n_α flow derivatives, $n_\mathscr{J}$ adjoint solutions, and $n_\mathscr{J}\times n_\alpha$ adjoint derivatives, giving $n_\alpha+n_\mathscr{J}+n_\alpha n_\mathscr{J}$ linear problems.

5.2.4. AV.AV

The final method of [128] extends AV.DD by introducing a second adjoint variable Γ. The product of Γ with (29) is added to (28), allowing the elimination of the derivatives of the first adjoint variable Λ, if Γ satisfies

$$\frac{\partial R}{\partial W}\Gamma=-\frac{\partial R}{\partial X}\frac{\partial X}{\partial\alpha}.$$

But this is exactly the direct equation, so $\Gamma=\mathrm{d}W/\mathrm{d}\alpha$, and the expression for the derivative reduces to that of DD.AV (27). Again only the original $n_\alpha+n_\mathscr{J}$ systems must be solved, as for DD.AV. The four formulations differ predominantly in the number of linear systems to be solved, and may be chosen on the basis of n_α and $n_\mathscr{J}$. For typical problems we therefore expect DD.AV/AV.AV to be most appropriate.

5.2.5. Applications

Given the considerable obstacles to the evaluation of second derivatives, it is perhaps surprising the techniques described here

have been implemented for aerodynamic flow codes at all. In fact several groups have independently performed the necessary work, though it is telling that all used automatic differentiation in some form. Sherman, Taylor et al. using ADIFOR [129], successfully tested DD.DD for two 2D viscous aerofoils, one with turbulence [128]. Much more recently they applied both DD.AV and AV.AV to inviscid flow about a NACA0012 [130]. The small test cases and the relatively slow pace of progress are perhaps a testament to the current limitations of AD codes (adjointing or *backward mode* is significantly harder than simple differentiation or *forward mode* for AD tools [44]).

Based on a purely theoretical derivation corresponding to $(DD)^n$ by Guillaume and Masmoudi [131], Aubert et al. applied AD to evaluate the partial derivatives of R and J, and applied the resulting code to the design of 2D and 3D low speed gas mixing devices [132]. By restricting design variables to aerodynamic values at the injection boundary, derivatives with respect to the mesh could be avoided. In further work the code was coupled to a genetic algorithm, the idea being to provide cheap objective function evaluation based on a second-order Taylor expansion near to known values [4]. The problem considered was optimization of a turbomachinery blade.

Most recently Ghate and Giles [46] constructed the Hessian of a simple code via DD.AV, including mesh sensitivities, by applying automatic differentiation extremely selectively. Rather than applying TAPENADE [133] to the entire solver in one shot, it was used to differentiate individual flux evaluation routines and boundary conditions. The remaining summation loops required to build the full derivatives of the residual were then constructed by hand, being substantially identical to the loops of the flow solver itself. This mode of use of AD has the potential to scale well to the much larger codes in use in industry, as well as to third- and higher-order derivatives.

5.3. Extension to aeroelasticity

The need for multi-disciplinary modeling of aircraft is well known, and is follows that multi-disciplinary optimization (MDO) is also necessary. Even if a purely aerodynamic objective function is considered, a wing optimized to be shock free in an unloaded (or *jig*) shape is unlikely to remain so when bent and twisted under a load. Mono-disciplinary optimization of the wing under a fixed structural deformation has also been shown to be insufficient [134,135]. The issues become even more complex when objective functions are also multi-disciplinary. Consider the trade-off between low aerodynamic drag – which requires a thin wing – and low structural weight which requires a thick wing (at least near the root).

Extension of the direct discrete gradient formulas for multi-disciplinary problems were presented as early as 1990 by Sobieszczanski-Sobieski [136]. Aeroelastic optimization was first carried out using finite differences gradients, Haftka [137], Bowman et al. [138], Friedmann [139], Barthelemy et al. [140] and Giunta and Sobieszczanski-Sobieski [141]. Later on, in order to reduce the cost of gradient evaluation, direct and adjoint methods for the multi-disciplinary equations were considered, first for 2D configurations by Ghattas and Li [142] and Moller and Lund [143], then for 3D configurations by Maute et al. [144], Hou and Satyanarayana [145], Maute et al. [146], Martins et al. [147], and Fazzolari [135].

These multi-disciplinary discrete sensitivity equations are briefly summarized in the remaining part of this subsection for an aeroelastic problem. In addition to existing notation introduce the surface displacement field U, and the structure mesh X_S. Meshes X and X_S are taken to match at solid walls and represent the *jig*-shape. This framework is the most widely used; see Farhat et al. for an alternative approach [148,144].

The objective function is taken to be $\mathscr{J}(\alpha) = J(\alpha, X(\alpha), X_S(\alpha), W, U)$, and the state equation reads

$$R(\alpha, X(\alpha), X_S(\alpha), W, U) = (R_A(X(\alpha), W, U), R_S(X_S(\alpha), W, U))^T = 0,$$

where R_A and R_S are aerodynamic and structural parts, respectively. In practical aeronautic applications the dependency of R_S on W corresponds to the aerodynamic stresses applied at the solid wall, whereas the dependency of R_A on U corresponds to the displacement of the aeroelastic structure. Define $Y = (W, U)$, so that differentiating the state equation gives

$$\frac{\mathrm{d}R}{\mathrm{d}\alpha} = \frac{\partial R}{\partial Y}\frac{\mathrm{d}Y}{\mathrm{d}\alpha} + \frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial R}{\partial X_S}\frac{\mathrm{d}X_S}{\mathrm{d}\alpha} = 0,$$

which may be separated into structural and aerodynamic parts

$$\begin{pmatrix} \frac{\partial R_A}{\partial W} & \frac{\partial R_A}{\partial U} \\ \frac{\partial R_S}{\partial W} & \frac{\partial R_S}{\partial U} \end{pmatrix} \begin{pmatrix} \frac{\mathrm{d}W}{\mathrm{d}\alpha} \\ \frac{\mathrm{d}U}{\mathrm{d}\alpha} \end{pmatrix} + \begin{pmatrix} \frac{\partial R_A}{\partial X} \\ \frac{\partial R_S}{\partial X} \end{pmatrix} \frac{\mathrm{d}X}{\mathrm{d}\alpha} + \begin{pmatrix} \frac{\partial R_A}{\partial X_S} \\ \frac{\partial R_S}{\partial X_S} \end{pmatrix} \frac{\mathrm{d}X_S}{\mathrm{d}\alpha} = 0,$$

and resolution for $\mathrm{d}Y/\mathrm{d}\alpha$ allows the computation of objective function gradients as

$$\frac{\mathrm{d}\mathscr{J}(\alpha)}{\mathrm{d}\alpha} = \frac{\partial J}{\partial Y}\frac{\mathrm{d}Y}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X_S}\frac{\mathrm{d}X_S}{\mathrm{d}\alpha}.$$

Yet again, for each state equation an adjoint vector is introduced which multiplies the linearization of that equation, and contributes to the expression for the objective function derivative, in order to factorize out the difficult terms:

$$\begin{aligned} \frac{\mathrm{d}\mathscr{J}}{\mathrm{d}\alpha} &= \frac{\partial J}{\partial Y}\frac{\mathrm{d}Y}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial J}{\partial X_S}\frac{\mathrm{d}X_S}{\mathrm{d}\alpha} + \Lambda^T\left[\frac{\partial R}{\partial Y}\frac{\mathrm{d}Y}{\mathrm{d}\alpha} + \frac{\partial R}{\partial X}\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \frac{\partial R}{\partial X_S}\frac{\mathrm{d}X_S}{\mathrm{d}\alpha}\right] \\ &= \left[\frac{\partial J}{\partial X} + \Lambda^T\frac{\partial R}{\partial X}\right]\frac{\mathrm{d}X}{\mathrm{d}\alpha} + \left[\frac{\partial J}{\partial X_S} + \Lambda^T\frac{\partial R}{\partial X_S}\right]\frac{\mathrm{d}X_S}{\mathrm{d}\alpha} + \left[\frac{\partial J}{\partial Y} + \Lambda^T\frac{\partial R}{\partial Y}\right]\frac{\mathrm{d}Y}{\mathrm{d}\alpha}, \end{aligned}$$

where $\Lambda = [\Lambda_A, \Lambda_S]^T$ corresponding to the decomposition of R into R_A and R_S, satisfies

$$\begin{pmatrix} \frac{\partial R_A}{\partial W}^T & \frac{\partial R_S}{\partial W}^T \\ \frac{\partial R_A}{\partial U}^T & \frac{\partial R_S}{\partial U}^T \end{pmatrix} \begin{pmatrix} \Lambda_A \\ \Lambda_S \end{pmatrix} = \begin{pmatrix} -\frac{\partial J}{\partial W}^T \\ -\frac{\partial J}{\partial U}^T \end{pmatrix}.$$

In practice the resolution of this multi-disciplinary system uses an iteration on *lagged* mono-disciplinary solutions

$$\begin{cases} \frac{\partial R_A}{\partial W}^T \Lambda_A^{(p)} = -\frac{\partial J}{\partial W}^T - \frac{\partial R_S}{\partial W}^T \Lambda_S^{(p-1)} \\ \frac{\partial R_S}{\partial U}^T \Lambda_S^{(p)} = -\frac{\partial J}{\partial U}^T - \frac{\partial R_A}{\partial U}^T \Lambda_A^{(p-1)} \end{cases},$$

converging to the solution of the coupled problem. As for the mono-disciplinary case $n_{\mathscr{J}}$ such systems must be resolved to obtain the complete derivative matrix.

6. Examples of sensitivity applications

In closing we briefly present some applications of sensitivity analysis to optimization. For the period up to 1999 the review of Newmann et al. [22] already mentioned, provides a comprehensive overview of applications in gradient-based design for aerodynamics, so we do not repeat that content here. For the time since we give a limited number of milestones results, to provide an impression of the progress made.

A good test of the progress in gradient-based methods over the last ten years is the high-fidelity coupled structure–fluid optimization of a transport aircraft wing, where the objective is to optimize a multi-disciplinary function – such as range – subject to a large number of both aerodynamic and structural design variables and constraints (including perhaps wing planform variation), necessitating an adjoint formulation. This is clearly a problem of great engineering relevance, which has to date, and to the authors' knowledge, remained unsolved.

By 1999 several groups had performed purely aerodynamic shape optimization of transport aircraft wings [149–151], whereby typically only profile design parameters were used, as without structural penalties wing planform optimization tends to produce wings with unrealistically large span. Starting in 2000 aeroelastic designs began to be performed, but always with a predetermined structure and planform [144,147,135]. In 2004, Leoviriyakit and Jameson [152,153] performed wing planform optimization using a semi-empirical analytical model for wing weight, taking into account aerodynamic loading and planform shape, but without elastic deformation. Finally in 2007 Jameson et al. [154] brought previous work together with an aeroelastic, multi-objective, planform optimization, but without modifications to the wing structure.

In other areas, in 2003 Campobasso et al. [155,156] performed an *unsteady* aerodynamic optimization of turbine blades using an adjointed frequency-domain code to avoid adjointing full time-accurate computations. Lately Mani and Mavriplis [126] have adjointed a full unsteady flow solver, and used it to minimize the time-dependent drag profile of a 2D aerofoil without any loss of lift, for example. For this the adjoint of the volume mesh deformation algorithm was also required. We have also already mentioned the first use of mesh sensitivities in adjoint based optimization in 2005 by Nielsen et al. [119,7].

Finally, we present three recent applications of gradient-based optimization from the authors' personal experience. The first uses a continuous adjoint on a structured grid in inviscid flow, the remaining two consider the Navier–Stokes equations and adjoints on unstructured and structured grids, respectively.

6.1. Navier–Stokes optimization of a wing–fuselage configuration

The only reason to employ an adjoint code is to evaluate sensitivities with respect to a large number of design variables rapidly, which raises the question: are a large number of variables really necessary in design? This issue was partially addressed by Brezillon et al. [8] for the optimization of the DLR-F6 transonic wing–fuselage configuration. The wing was parameterized using a free-form deformation (FFD) control box whose node coordinates formed the design variables. Fig. 6 shows such a box for a very low number of parameters; more detailed control is possible simply by adding nodes. The wing thickness is held constant by coupling nodes on the upper and lower sides of the box. A structured mesh is used in order to reduce the number of mesh points required for the required accuracy.

Table 2
Drag improvement for coarse and fine parameterized optimizations.

Parameters	9	42
$\Delta\alpha$ (°)	−0.30	−0.29
C_L	0.500	0.500
$\Delta C_D \times 10^{-4}$ Total	−1.4	−17.7
$\Delta C_D \times 10^{-4}$ Wing	+1.2	−16.1
$\Delta C_D \times 10^{-4}$ Fuselage	−2.6	−1.6

Gradients were computed using a discrete adjoint solver implemented in the unstructured DLR Tau-Code [157] using a duality preserving LU-SGS smoothed multigrid method. Frozen eddy-viscosity was used, and for stabilization the recursive projection method was applied [6].

The optimization is drag minimization at constant lift. Satisfying the lift constraint requires sensitivities of the lift as well as the drag, hence two adjoint calculations are performed per gradient evaluation, and the angle-of-attack α is chosen to achieve the desired lift. The optimization is performed using conjugate gradients for two different parameterizations, one coarse with nine variables including twist, and one fine with 42 variables. In Table 2, reproduced from Ref. [8], the results are given including drag reduction for the wing and fuselage independently.

What is apparent is that the coarse parameterization achieved a drag reduction purely on the fuselage by means of reducing the angle-of-attack. The drag on the wing was actually *increased* as the optimizer attempted to preserve the lift. In contrast the fine parameterization allowed a significant improvement on the wing,

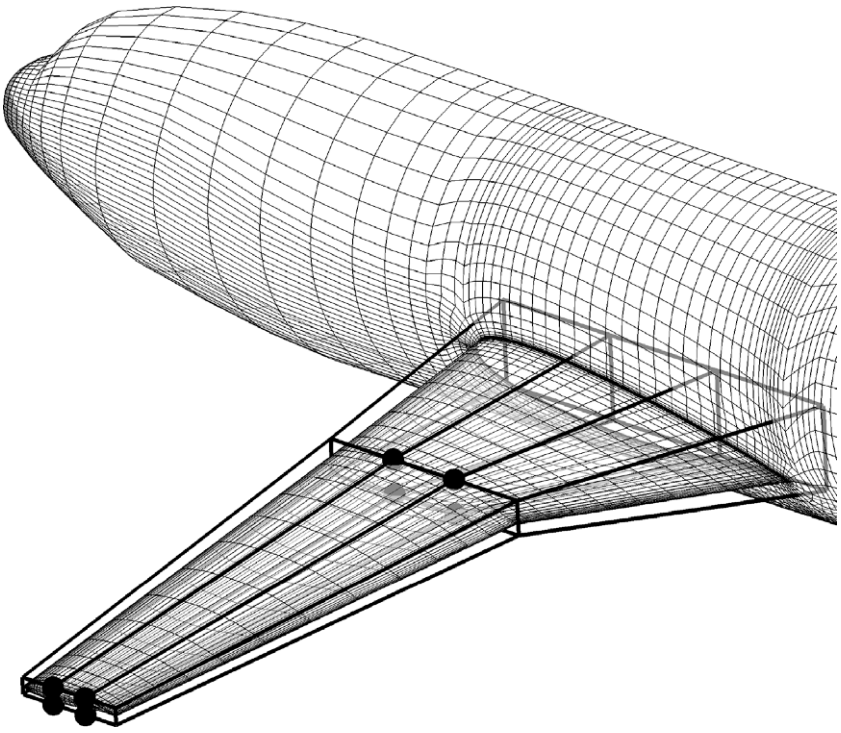

Fig. 6. DLR F6 wing–fuselage showing parameterization of the wing with a free-form deformation box. Black circles denote nodes used as design variables.

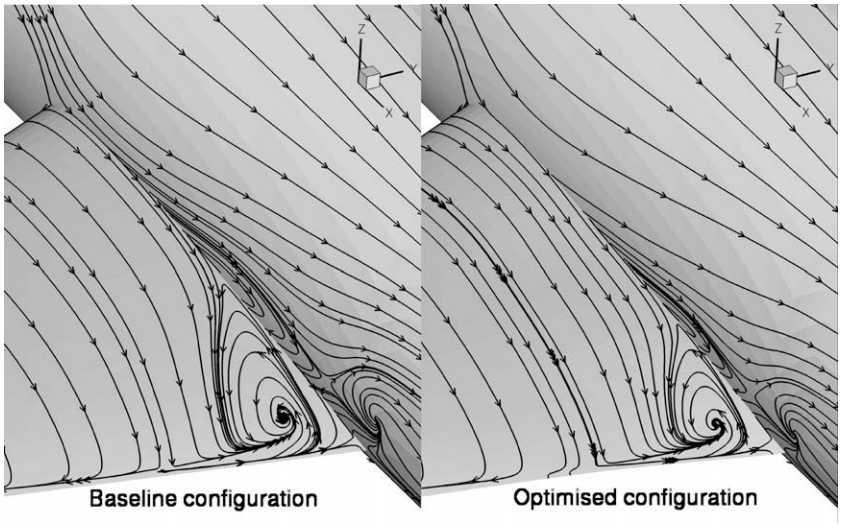

Fig. 7. Comparison of region of corner separation on the F6 before and after optimization.

and a total drag reduction of 18 counts. Because there is no wing-body fairing this configuration always has a region of recirculation in the corner where the wing and fuselage meet, and this separation is the dominant source of removable drag. In Fig. 7, the region of separation is plotted for the two optimized configurations with the original separation boundary marked.

Although the result of any such study must be very dependent on the method of parameterization, and an informed choice may reduce the number of variables required significantly, it seems that – at least in general – there are substantial advantages to large numbers of variables, and hence adjoint is essential.

6.2. Multi-disciplinary optimization of an engine pylon

An optimization much closer to a concrete engineering application was performed by Mouton et al. [158,159] who coupled aerodynamic and structural optimization of the outboard pylon of a generic civil transport aircraft within the European project VIVACE, see Fig. 8 from Ref. [158] for the computational grid. In order to incorporate advanced CFD in the form of the adjoint RANS equations into this multi-disciplinary optimization without building a structure–fluid coupled adjoint as in Section 5.3, they followed a two-level parameterization strategy where three design variables were identified as having a strong coupling role in the problem – namely the vertical and horizontal position of the pylon and its width – the remaining variables of pylon fairing and structural frame shape being almost discipline specific (the latter due to the free space available under the fairing). A coarse sampling of the strongly discipline coupling variables was made, and a discipline specific optimization performed at each sample point. These individual results were fed back into the global optimization problem using a Kriging method.

The individual aerodynamic optimizations were performed using a discrete adjoint solver implemented in the block-structured elsA code using an adjointed implicit method with added dissipation for stabilization, and frozen eddy-viscosity [91]. The pylon was parameterized using 19 design variables and the optimizations were performed with a BFGS method. The relatively large number of design variables and repeated optimizations necessitated a rapid optimization strategy, hence the use of adjoint gradients in this case.

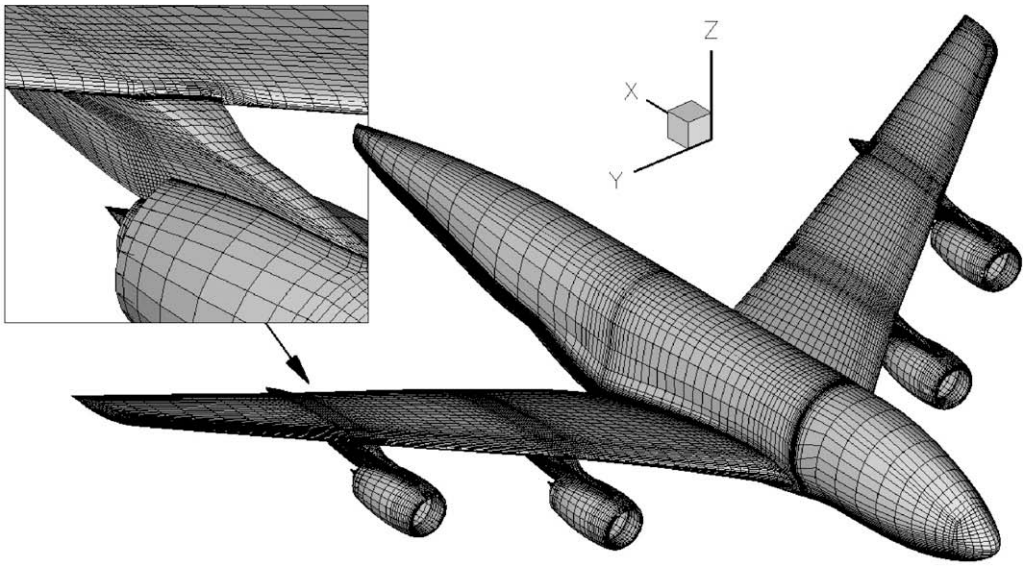

Fig. 8. Surface mesh for the generic transport aircraft with close up of the pylon.

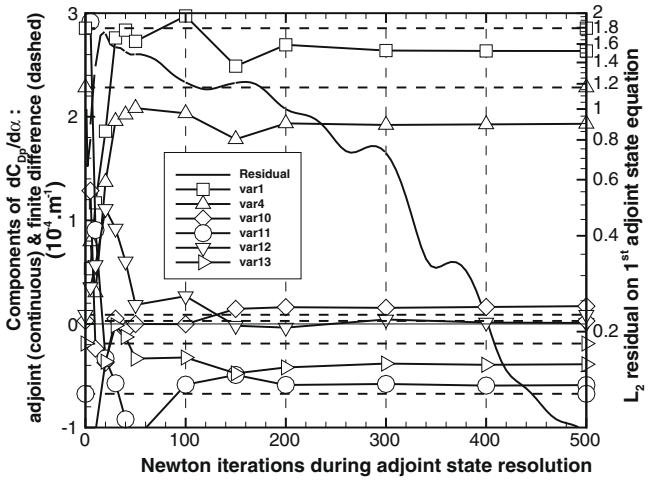

Fig. 9. Convergence of a single adjoint computation for the pylon optimization case. The problem residual is shown, as well as transient gradients of six design variables which are compared to gradients obtained with finite differences (horizontal dashed lines).

The convergence of a typical adjoint computation involved in this optimization is shown in Fig. 9. At regular intervals a selection of gradients obtained from the non-fully converged adjoint solution are plotted in order to evaluate the accuracy of the transient adjoint solution. In this case after 500 multigrid cycles no significant further variation in gradients is seen and the computation is halted. For comparison gradients obtained from finite differences are also plotted and relatively good agreement is seen.

Finally the optimization over all structure and shape variables parameterizing the pylon resulted in a total drag reduction of 1.3 counts.

7. Conclusion

A wide variety of gradient computation approaches have been investigated in the past 20 years. Initially continuous adjoint approaches dominated, being readily constructed for the Euler equations, not requiring linearization of large complex discretizations, and solved with more-or-less the same techniques as the flow equations. More recently, with wider use of RANS codes, the most practical algorithms for optimization, have been discrete – most often adjoint – approaches, where the Jacobian is obtained by hand, and where the resulting system is solved using duality preserving iterations stabilized with Krylov methods. For smaller problems today the Jacobian may be obtained with AD, and solved with generic linear system solution methods, such as ILU preconditioned Krylov methods.

The situation is unlikely to remain so for long however. The trend in simulation today is towards increased coupling and complexity of physical modeling, and there is a demand for not just flow sensitivities, but mesh and parameter sensitivities. Linearizing a code by hand in all these variations is likely to become increasingly impractical, whereas constructing an accurate continuous formulation is likely to be impossible. Hence, if corresponding sensitivity codes are needed they must be obtained with some greater-or-lesser reliance on AD.

However a common mode of application of AD, of applying indiscriminately to an entire flow code, results today in large run-times and memory requirements, which nonetheless might be solved by future developments in AD compilers. However, more fundamentally, such usage does not take into account the need for extra stabilization of the linear iteration, done with Krylov methods and Jacobian reduction today.

It is our belief that, at least in the medium term, industrially relevant linearized codes will be developed by using AD to differentiate individual non-linear routines, which are then assembled by an expert, in the model of Ghate and Giles [46], using the many tricks learnt in the past 20 years.

Acknowledgments

The authors express their gratitude to their colleagues: Joël Brezillon for permitting the use of results and figures from Ref. [8], and for his many useful comments, as well as Meryem Marcelet, Chi-Tuan Pham, and Florent Renac for their useful comments and contributions to Sections 3.1 and 6. Daniel Destarac also provided invaluable corrections in the final stages of preparation.

References

[1] Sobieszczanski-Sobieski J. The case for aerodynamic sensitivity analysis. Technical Report CP 2457, NASA; 1987.
[2] Brezillon J, Wild J. Evaluation of different optimization strategies for the design of a high-lift flap device. In: Evolutionary and deterministic methods for design. Munich: EUROGEN; 2005.
[3] Lombardi G, Mengali G, Beux F. A hybrid genetic based optimization procedure for aircraft conceptual analysis. Optim Eng 2006;7(2):151–71.
[4] Kelner V, Grondin G, Léonard O, Moreau S. Multi-objective optimization of a fan blade by coupling a genetic algorithm and a parametric solver. In: Proceedings of EUROGEN, Munich; 2005.
[5] Jameson A, Martinelli L, Pierce N. Optimum aerodynamic design using the Navier–Stokes equations. Theor Comput Fluid Dyn 1998;10(1):213–37.
[6] Dwight R, Brezillon J, Vollmer D. Efficient algorithms for solution of the adjoint compressible Navier–Stokes equations with applications. In: Proceedings of the ONERA-DLR aerospace symposium (ODAS), Toulouse; 2006.
[7] Mavriplis D. Discrete adjoint-based approach for optimization problems on three-dimensional unstructured meshes. AIAA J 2007;45(4):740–52.
[8] Brezillon J, Brodersen O, Dwight R, Ronzheimer A, Wild J. Development and application of a flexible and efficient environment for aerodynamic shape optimisation. In: Proceedings of the ONERA-DLR aerospace symposium (ODAS), Toulouse; 2006.
[9] Pironneau O. On optimum profiles in Stokes flow. J Fluid Mech 1973;59(1):117–28.
[10] Pironneau O. On optimum design in fluid mechanics. J Fluid Mech 1974;64(2):97–110.
[11] Jameson A. Aerodynamic design via control theory. J Sci Comput 1988;3(3):233–60.
[12] Sharp H, Sirovitch L. Constructing a continuous parameter range of computational flows. AIAA J 1989;27(10):1326–31.
[13] Bristow D, Hawk J. Subsonic panel method for the efficient analysis of multiple geometry perturbations. Technical Report CR 3528, NASA; 1982.
[14] Bristow D, Hawk J, Subsonic panel method for designing wing surface from pressure distribution. Technical Report CR 3713, NASA; 1983.
[15] Elbanna H, Carlson L. Determination of aerodynamic sensitivity coefficients in the transonic and supersonic regimes. In: AIAA paper series, Paper 89-0532; 1989.
[16] Taylor III A, Korivi V, Hou G. Sensitivity analysis applied to the Euler equations: a feasibility study with emphasis on variation of geometric shape. In: AIAA paper series, Paper 91-0173; 1991.
[17] Baysal O, Eleshaky M. Aerodynamic design sensitivity analysis methods for the compressible Euler equations. J Fluids Eng 1991;113(4):681–8.
[18] Shubin G, Frank P. A comparison of implicit gradient approach and the variational approach to aerodynamic design optimization. Technical Report AMS-TR-163, Boeing Computer Service, Applied Mathematics and Statistics, June; 1991.
[19] Frank P, Shubin G. A comparison of optimisation-based approaches for a model computational aerodynamics design problem. J Comput Phys 1992;98:74–89.
[20] Dulikravitch G. Aerodynamic shape design optimization: status and trends. J Aircraft 1992;29(6):1020–6.
[21] Sobieszczanski-Sobieski J, Haftka R. Multidisciplinary aerospace design optimization: survey of recent developments. Struct Optim 1997;14(1):153–60.
[22] Newman III J, Taylor III A, Barnwell A, Newman P, Hou G. Overview of sensitivity analysis and shape optimization for complex aerodynamic configurations. J Aircraft 1999;36(1):97–116.
[23] Uthup B, Koruthu S-P, Sharma R-K, Priyadarshi P, editors. Recent trends in aerospace design and optimization. New Delhi: Tata-McGraw Hill; 2005.
[24] Haftka R. Sensitivity calculations for iteratively solved problems. Int J Numer Methods Eng 1985;21:1535–46.
[25] Anderson W, Newman J, Whitfield D, Nielsen E. Sensitivity analysis for the Navier–Stokes equations on unstructured meshes using complex variables. AIAA J 2001;39(1):56–63.

[26] Hicks R, Murman E, VanderPlaats G. An assessment of airfoil design by numerical optimization. Technical Report TMX 3092, NASA; 1974.
[27] VanderPlaats G, Hicks R. Numerical airfoil optimization using a reduced number of design coordinates. Technical Report TMX 73151, NASA; 1976.
[28] Hicks R, Henne P. Wing design by numerical optimization. In: AIAA paper series, Paper 77-1247; 1977.
[29] Reneaux J. Méthode de définition de profils par optimisation numérique. La Rech Aerospatiale 1984;5:303–21.
[30] Reneaux J, Thibert J. The use of numerical optimization for airfoil design. In: AIAA paper series, Paper 85-5026; 1985.
[31] Aidala P, Davis W, Mason W. Smart aerodynamic optimization. In: AIAA paper series, Paper 83-1863; 1983.
[32] Reuther J, Cliff S, Hicks R, van Dam C. Practical design optimization of wing/body configurations using the Euler equations. In: AIAA paper series, Paper 92-2633; 1992.
[33] Hager J, Eyi S, Lee K. Design efficiency evaluation for transonic airfoil optimization: a case for Navier–Stokes design. In: AIAA paper series, Paper 93-3112; 1993.
[34] Matsuzawa T, Hafez M. Optimum shape design using adjoint equations for compressible flows with shock waves. Comput Fluid Dyn J 1998;7(3):74–89.
[35] Baysal O, Eleshaky M, Burgreen G. Aerodynamic shape optimization using sensitivity analysis on third-order Euler equations. In: AIAA paper series, Paper 91-1577; 1991.
[36] Baysal O, Eleshaky M. Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics. AIAA J 1992;30:718–25.
[37] Baysal O. Flow analysis and design optimization methods for nozzle afterbody of a hypersonic vehicle. Technical Report CR 4431, NASA; 1992.
[38] Newman III J, Taylor III A, Burgreen G. An unstructured grid approach to sensitivity analysis and shape optimization using the Euler equations. In: AIAA paper series, Paper 95-1646; 1995.
[39] Newman III J, Taylor III A. Three dimensional aerodynamic shape sensitivity analysis and design optimization using the Euler equations on unstructured grids. In: AIAA paper series, Paper 96-2464; 1996.
[40] Taylor III A, Newman III J, Barnwell R. Aerodynamic shape sensitivity analysis and design optimization of complex configurations on unstructured grids. In: AIAA paper series, Paper 97-2275; 1997.
[41] Shubin G. Obtaining cheap optimization gradients from computational aerodynamics codes. Technical Report AMS-TR-164, Boeing Computer Service, Applied Mathematics and Statistics, June; 1991.
[42] Dwight R, Brezillon J. Adjoint algorithms for the optimization of 3d turbulent configurations. In: Proceedings of the 15th STAB Symposium; 2006.
[43] Dwight R, Brezillon J. Effect of approximations of the discrete adjoint on gradient-based optimization. AIAA J 2006;44(12):3022–71.
[44] Griewank A. Evaluating derivatives, principles and techniques of algorithmic differentiation, No. 19 in Frontiers in Applied Mathematics, SIAM, Philadelphia; 2000, ISBN 08-987-1451-6.
[45] Corliss G, Le Faure C, Griewank A, Hascoét L, Naumann U, editors. Automatic differentiation: from simulation to optimization, computer and information science. Springer; 2001.
[46] Ghate D, Giles M. Hessian calculation using automatic differentiation. In: AIAA paper series, Paper 2007-4059; 2007.
[47] Abergel F, Temam R. On some control problems in fluid mechanics. Theor Comput Fluid Dyn 1990;1(1):303–25.
[48] Giles M, Pierce N. An introduction to the adjoint approach to design. In: Proceedings of ERCOFTAC workshop on adjoint methods; 1999.
[49] Jameson A. Optimum aerodynamic design via boundary control. In: AGARD-FDP-VKI special course; 1994.
[50] Gauger N, Brezillon J. Aerodynamic shape optimization using the adjoint method. J Aero Soc India 2002;54(3):247–54.
[51] Anderson W, Venkatakrishnan V. Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation. Comput Fluids 1998;29:443–80.
[52] Jameson A. Aerodynamic shape optimization using the adjoint method. In: VKI course in optimization; 2003.
[53] Hiernaux S, Essers J-A. An optimal control theory based algorithm to solve 2d aerodynamic shape optimisation problems for inviscid and viscous flows. In: Proceedings of the RTO-AVT symposium on aerodynamic design and optimisation of flight vehicles; 1999.
[54] Hiernaux S, Hessers J. Aerodynamic optimization using Navier–Stokes equations and optimal control theory. In: AIAA paper series, Paper 99-3297; 1999.
[55] Giles M, Pierce N. Adjoint equations in CFD: duality, boundary conditions and solution behaviour. In: AIAA paper series, Paper 97-1850; 1997.
[56] Iollo A, Salas M, Ta'asan S. Shape optimization governed by the Euler equations using an adjoint method. Technical Report 93-78, ICASE; 1993.
[57] Iollo A, Salas M. Contribution to the optimal shape design of two-dimensional internal flows with embedded shocks. J Comput Phys 1996;125:124–34.
[58] Cliff E, Heikenschloss M, Shenoy A. On the optimality system for a 1d Euler flow problem. In: AIAA paper series, Paper 96-3993; 1996.
[59] Giles M, Pierce N. On the properties of solutions of the adjoint Euler equations. In: Proceedings of the 6th ICFD conference on numerical method for fluid dynamics, Oxford; 1998.
[60] Gunzburger M. Sensitivities, adjoint and flow optimization. Int J Numer Methods Fluids 1999;35:53–78.
[61] Bardos C, Pironneau O. Derivatives and control in the presence of shocks. Comput Fluid Dyn J 2003;11(4):383–92.
[62] Mohammadi B, Pironneau O. Optimal shape design for fluids. Ann Rev Fluids Mech 2004;36:255–79.
[63] Pironneau O. Shape sensitivities and design for fluids with shocks. Int J Comput Fluid Dyn 2003;17(14):235–42.
[64] Pelletier D, Turgeon E, Lacasse D, Boorggaard J. Adaptivity, sensitivity and uncertainty: towards standards in CFD. In: AIAA paper series, Paper 2001-0192; 2001.
[65] Pelletier D, Turgeon E, Etienne S, Boorggaard J. Reliable sensitivity analysis via an adaptive sensitivity equation method. In: AIAA Paper Series, Paper 2002-2578; 2002.
[66] Brezillon J, Gauger N. 2D and 3D aerodynamic shape optimization using the adjoint approach. Aerospace Sci Technol J 2004;8(8):715–27.
[67] Jameson A, Alonso J, Reuther J, Martinelli L, Vassberg J. Aerodynamic shape optimization techniques based on control theory. In: AIAA paper series, Paper 98-2538; 1998.
[68] Weinerfelt P, Enoksson O. Numerical methods for aerodynamic optimization. In: Proceedings of the 8th international symposium on CFD, Swansea, Wales; 1999.
[69] Soto O, Löhner R, Yang C. An adjoint-based design methodology for CFD problems. Int J Numer Methods Heat Fluid Flow 2004;14(6):734–59.
[70] Jameson A, Sriram, Martinelli L, Haimes B. Aerodynamic shape optimization of complete aircraft configurations using unstructured grids. In: AIAA paper series, Paper 2007-4060; 2007.
[71] Castro C, Lozano C, Palacios F, Zuazua E. A systematic continuous adjoint approach to viscous aerodynamic design on unstructured grids. In: AIAA paper series, Paper 2006-0051; 2006.
[72] Widhalm M, Ronzheimer A, Hepperle M. Comparison between gradient-free and adjoint based aerodynamic optimization of a flying wing transport aircraft in the preliminary design. In: AIAA paper series, Paper 2007-4060; 2007.
[73] Nadarajah S, Jameson A. Studies of continuous and discrete adjoint approaches to viscous automatic aerodynamic shape optimization. In: AIAA paper series, Paper 2001-2530; 2001.
[74] Giles M, Pierce N. Analytic adjoint solutions for the quasi-one-dimensional Euler equations. J Fluid Mech 2001;426:327–45.
[75] Hartmann R, Derivation of an adjoint consistent discontinuous Galerkin discretization of the compressible Euler equations. In: Lube G, Rapin G, editors. Proceedings of the BAIL conference, Goettingen; 2006.
[76] Dwight R, Han Z-H. Efficient uncertainty quantification using gradient-enhanced Kriging. In: AIAA paper series, Paper 2009-2276; 2009.
[77] Nielsen E, Anderson W. Aerodynamic design optimization on unstructured meshes using the Navier–Stokes equations. AIAA J 1999;37(11):185–91.
[78] Giles M, Duta M, Muller J-D, Pierce N. Algorithm developments for discrete adjoint methods. AIAA J 2003;41(2):198–205.
[79] Le Moigne A, Qin N. A discrete adjoint method for aerodynamic sensitivities for Navier–Stokes. In: Proceedings of CEAS Cambridge; 2002.
[80] Kim C, Kim C, Rho O, Lee S. Aerodynamic sensitivity analysis for the Navier–Stokes equations. In: AIAA paper series, Paper 99-0402; 1999.
[81] Pham C. Linéarisation du flux visqueux des equations de Navier–Stokes et de modéles de turbulence pour l'optimisation aérodynamique en turbomachines. Ph.D. thesis, L'Ecole Nationale Supérieure d'Arts et Métiers, September; 2006.
[82] Giles M, Duta M, Muller J. Adjoint code developments using the exact discrete approach. In: AIAA paper series, Paper 2001-2596; 2001.
[83] Anderson W, Bonhaus D. Aerodynamic design on unstructured grids for turbulent flows. Technical Report TM 112867, NASA; 1997.
[84] Anderson W, Bonhaus D. Airfoil design optimization on unstructured grids for turbulent flows. AIAA J 1999;37(2):185–91.
[85] Nielsen E, Anderson W. Recent improvements in aerodynamic design optimization on unstructured meshes. AIAA J 2002;40(6):1155–63.
[86] Nemec N, Zingg D. Towards efficient aerodynamic shape optimization based on the Navier–Stokes equations. In: AIAA paper series, Paper 2001-2532; 2001.
[87] Brezillon J, Dwight R. Discrete adjoint of the Navier–Stokes equations for aerodynamic shape optimization. In: Evolutionary and deterministic methods for design. Munich: EUROGEN; 2005.
[88] Kim C, Kim C, Rho O. Sensitivity analysis for the Navier–Stokes equations with two equations turbulence models. AIAA J 2001;39(5):838–45.
[89] Kim C, Kim C, Rho O. Effects of constant eddy viscosity assumption on gradient-based design optimization. In: AIAA paper series, Paper 2002-0262; 2002.
[90] Renac F, Pham C-T, Peter J. Sensitivity analysis for the RANS equations coupled with a linearized turbulence model. In: AIAA paper series, Paper 2007-3948; 2007.
[91] Peter J, Mayeur J. Improving accuracy and robustness of a discrete direct differentiation method and discrete adjoint method for aerodynamic shape optimization. In: Proceedings of ECCOMAS, Egmond ann Zee; 2006.
[92] Jameson A, Pierce N, Martinelli L. Optimum aerodynamic design using the Navier–Stokes equations. In: AIAA paper series, Paper 97-101; 1997.
[93] Soemarwoto B. The variational method for aerodynamic optimization using the Navier–Stokes equations. Technical Report 97-71, ICASE, December; 1997.
[94] Valentin V. Optimisation aérodynamique 3d des aubages dans les turbomachines axiales multi-etages. Ph.D. thesis, Université Paris 6, June; 2002.
[95] Hou G, Maroju V, Taylor A, Korivi V, Newman P. Transonic turbulent airfoil design optimization with automatic differentiation in incremental iterative form. In: AIAA paper series, Paper 97-0101; 1995.

[96] Mohammadi B. A new optimal shape design procedure for inviscid and viscous flows. Int J Numer Methods Fluids 1997;25(2):183–203.
[97] Bischof C, Buecker H, Rasch A. Sensitivity analysis of turbulence models using automatic differentiation. SIAM J Sci Comput 2005;26(2):510–22.
[98] Mavriplis D. Multigrid solution of the discrete adjoint for optimization problems on unstructured meshes. AIAA J 2006;44(1):42–50, Paper 0001-1452.
[99] Beux F, Dervieux A. Exact-gradient shape optimization of a 2d Euler flow. Finite Elem Anal Design 1992;12:281–302.
[100] Beux F, Dervieux A. A hierarchical approach for shape optimization. Technical Report 1868, INRIA; 1993.
[101] Marco N, Beux F. Multilevel optimization: Application to one-shot shape optimum design. Technical Report 2068, INRIA; 1993.
[102] Marco N, Dervieux A. Multilevel parameterization for aerodynamical optimization of 3d shapes. Technical Report 2949, INRIA; 1996.
[103] Nielsen E, Anderson W. Aerodynamic design optimization on structured meshes using the Navier–Stokes equations. In: AIAA paper series, Paper 98-4809; 1998.
[104] Peter J, Drullion F, Pham C-T. Contribution to discrete implicit gradient and discrete adjoint method for aerodynamic shape optimization. In: Proceedings of ECCOMAS, Jyvaskyla; 2004.
[105] Meaux M, Cormery M, Voizard G. Viscous aerodynamic shape optimization based on the discrete adjoint state for 3d industrial configurations. In: Proceedings of ECCOMAS, Jyvaskyla; 2004.
[106] Kuruvila G, Narducci R, Agrawal S. Development and application of TLNS3d-Adjoint: a practical tool for aerodynamic shape optimization. In: AIAA paper series, Paper 2001-2400; 2001.
[107] Mohammadi B. Dynamical approaches and incomplete gradients for shape optimization and flow control. In: AIAA paper series, Paper 99-3374; 1999.
[108] Medic G, Mohammadi B, Petruzzelli N, Stanciu M, Hecht F. 3d optimal shape design for complex flows: applications to turbomachinery. In: AIAA paper series, Paper 99-0182; 1999.
[109] Mohammadi B. Flow control and shape optimization in aeroelastic configurations. In: AIAA paper series, Paper 99-0182; 1999.
[110] Kim H-J, Obayashi S, Nakahashi K. Flap-deflection optimization for transonic cruise performance improvement of supersonic transport wing. J Aircraft 2001;38(4):709–17.
[111] Kim C, Kim C, Rho O. Feasibility study of the constant eddy-viscosity assumption in gradient-based design optimization. J Aircraft 2003;40:1168–76.
[112] Newman III J, Taylor III A, Hou G-W, Jones H. An approximately factored incremental strategy for calculating consistent discrete aerodynamic sensitivity derivatives. J Comput Phys 1994;113:336–46.
[113] Nemec M, Zingg D. A Newton–Krylov algorithm for aerodynamic design using the Navier–Stokes equations. AIAA J 2002;40(6):1146–54.
[114] Giles M. On the iterative solution of the adjoint equations. In: Automatic differentiation: from simulation to optimization; 2001. p. 145–52.
[115] Christianson B. Reverse accumulation and attractive fixed points. Optim Methods Software 1994;3(4):311–26.
[116] Nielsen E, Lu J, Park M, Darmofal D. An exact dual adjoint solution method for turbulent flows on unstructured grids. In: AIAA paper series, Paper 2003-0272; 2003.
[117] Dwight R. Efficiency improvements of RANS-based analysis and optimization using implicit and adjoint methods on unstructured grids. Ph.D. thesis, School of Mathematics, University of Manchester; 2006.
[118] Giles M. On the use of Runge–Kutta time-marching and multigrid for the solution of the steady adjoint equations. In: AD2000 conference in Nice; 2000.
[119] Nielsen E, Park M. Using an adjoint approach to eliminate mesh sensitivities in aerodynamic design. AIAA J 2005;44(5):948–53.
[120] Nielsen E. Aerodynamic design sensitivities on an unstructured mesh using the Navier–Stokes equations and a discrete adjoint formulation. Ph.D. thesis, Virginia State University; 1998.
[121] Campobasso M, Giles M. Stabilization of a linear flow solver for turbomachinery aeroelasticity by means of the recursive projection method. AIAA J 2004;42(9):1765–74.
[122] Campobasso M, Giles M. Stabilizing linear harmonic flow solvers for turbomachinery aeroelasticity with complex iterative algorithms. AIAA J 2006;44(5):1048–59.
[123] Saad Y, Schultz MH. GMRES: a generalized minimum residual algorithm for solving non-symmetric linear systems. SIAM J Sci Stat Comput 1988;7(3):856–9.
[124] Schroff G, Keller H. Stabilization of unstable procedures: the recursive projection method. SIAM J Numer Anal 1993;30(4):1099–120.
[125] Keller H. RPM: a remedy for instability. In: Estep D, Tavener S, editors. Collected lectures on the preservation of stability under discretization, SIAM Proceedings in applied mathematics, vol. 109; 2002. p. 185–96.
[126] Mani K, Mavriplis D. An unsteady discrete adjoint formulation for two-dimensional flow problems with deforming meshes. In: AIAA paper series, Paper 2007-0060; 2007.
[127] Jakobsson S, Amoignon O. Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization. Comput Fluids 2007;36:1119–36.
[128] Sherman L, Taylor III A, Green L, Newman A, Hou G, Korivi V. First- and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods. J Comput Phys 1996;129:307–31.
[129] Bischof C, Carle A, Corliss G, Griewank A, Hovland P. The ADIFOR 2.0 system for the automatic differentiation of Fortran 77 programs. Sci Program 1992;187:1–29.
[130] Taylor III A, Green L, Newman P, Putko M. Some advanced concepts in discrete sensitivity analysis. AIAA J 2003;48(7):1224–9.
[131] Guillaume P, Masmoudi M. Computation of high order derivatives in optimal shape design. J Numer Math 1994;67:231–50.
[132] Aubert S, Tournier J, Rochette M, Blanche J, N'Diaye M, Melem M, et al. Optimisation of a gas mixer using a new parametric solver. In: Proceedings of ECCOMAS CFD, Swansea, Wales; 2001.
[133] Hascoët L. TAPENADE: a tool for automatic differentiation of programs. In: Proceedings of 4th European congress on computational methods, ECCOMAS 2004, Jyvaskyla, Finland; 2004.
[134] Wakayama S. Lifting surfaces design using multidisciplinary optimization. Ph.D. thesis, Stanford University, June; 1994.
[135] Fazzolari A. An aero-structure adjoint formulation for efficient multidisciplinary wing optimization. Ph.D. thesis, Technical University of Braunschweig, September; 2005.
[136] Sobieszczanski-Sobieski J. Sensitivity of complex internally coupled systems. AIAA J 1990;28(1):153–60.
[137] Haftka R. Structural optimization with aeroelastic constraints: a survey of U.S. applications. Int J Vehicle Des 1986;7:381–92.
[138] Bowman K, Grandhi R, Eastep F. Aerodynamic design optimization using sensitivity analysis and computational fluid dynamics. J Struct Optim 1989;1:153–61.
[139] Friedmann P. Helicopter vibration reduction using structural optimization with aeroelastic/multidisciplinary constraints: a survey. AIAA J 1991;28:8–21.
[140] Barthelemy J-F, Wrenn G, Dovi A, Hall L. Supersonic transport wing minimum design integrating aerodynamics and structures. J Aircraft 1994;31:330–8.
[141] Giunta A, Sobieszczanski-Sobieski J. Progress toward using sensitivity derivatives in a high-fidelity aeroelastic analysis of a supersonic transport. In: AIAA paper series, Paper 98-4763; 1998.
[142] Ghattas O, Li X. Domain decomposition methods for the sensitivity analysis of a nonlinear aeroelastic problem. Int J Comput Fluid Dyn 1998;11:113–30.
[143] Moller H, Lund E. Shape sensitivity analysis of strongly coupled fluid–structure interaction problems. In: AIAA paper series, Paper 2000-4823; 2000.
[144] Maute K, Lesoinne N, Farhat C. Optimization of aeroelastic systems using coupled analytical sensitivities. In: AIAA paper series, Paper 2000-0560; 2000.
[145] Hou G, Satyanarayana A. Analytical sensitivity analysis of a static aeroelastic wing. In: AIAA paper series, Paper 2000-4824; 2000.
[146] Maute K, Nikbay N, Farhat C. Coupled analytical sensitivity analysis and optimization of three-dimensional nonlinear aeroelastic systems. AIAA J 2001;39(11):2051–61.
[147] Martins J, Alonso J, Reuther J. High-fidelity aero-structural design optimization of a supersonic business jet. In: AIAA paper series, Paper 2002-1483; 2002.
[148] Farhat C, Lesoinne M, Maman N. Mixed explicit/implicit time integration of coupled aeroelastic problems: three-field formulation, geometric conservation and distributed solution. Int J Numer Methods Fluids 1995;21:807–35.
[149] Jameson A. Re-engineering the design process through computation. J Aircraft 1999;36(1):36–50.
[150] Reuther J, Jameson A, Remlinger M, Saunders D. Constrained multi-point aerodynamic shape optimization using an adjoint formulation and parallel computers, Part I. J Aircraft 1999;36(1):51–60.
[151] Reuther J, Jameson A, Remlinger M, Saunders D. Constrained multi-point aerodynamic shape optimization using an adjoint formulation and parallel computers, Part II. J Aircraft 1999;36(1):61–74.
[152] Leoviriyakit K, Jameson A. Aero-structural wing planform optimization. In: AIAA paper series, Paper 2004-0029; 2004.
[153] Leoviriyakit K, Jameson A. Multi-point wing planform optimization via control theory. In: AIAA paper series, Paper 2005-0450; 2005.
[154] Jameson A, Leoviriyakit K, Shankaran S. Multi-point aero-structural optimization of wings including planform variations. In: AIAA paper series, Paper 2007-764; 2007.
[155] Campobasso M, Duta M, G MB. Adjoint calculation of sensitivities of turbomachinery objective functions. AIAA J Propulsion Power 2003;19(4):693–703.
[156] Duta M, Giles M, Campobasso M. The harmonic adjoint approach to unsteady turbomachinery design. Int J Numer Methods Fluids 2002;40(3–4):323–32.
[157] Gerhold T, Galle M, Friedrich O, Evans J. Calculation of complex 3d configurations employing the DLR TAU-Code. In: AIAA paper series, Paper 97-0167; 1997.
[158] Mouton S, Laurenceau J, Carrier G. Aerodynamic and structural optimization of powerplant integration under the wing of a transonic transport aircraft. In: Proceedings of 42th AAAF symposium on applied aerodynamics, Nice; 2007.
[159] Salah El Din I, Carrier G, Mouton S. Discrete adjoint method in elsA (Part 2): application to aerodynamic design optimisation. In: Proceedings of the ONERA-DLR aerospace symposium (ODAS), Toulouse; 2006.