INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2009; **60**:365–389
Published online 19 August 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1894

Efficient and robust algorithms for solution of the adjoint compressible Navier–Stokes equations with applications

Richard P. Dwight*,† and Joël Brezillon

German Aerospace Center (DLR), Lilienthalplatz 7, Braunschweig D-38108, Germany

SUMMARY

The complete discrete adjoint equations for an unstructured finite volume compressible Navier–Stokes solver are discussed with respect to the memory and time efficient evaluation of their residuals, and their solution. It is seen that application of existing iteration methods for the non-linear equation—suitably adjointed—has a property of guaranteed convergence provided that the non-linear iteration is well behaved. For situations where this is not the case, in particular for strongly separated flows, a stabilization technique based on the Recursive Projection Method is developed. This method additionally provides the dominant eigenmodes of the problem, allowing identification of flow regions that are unstable under the basic iteration. These are found to be regions of separated flow. Finally, an adjoint-based optimization with 96 design variables is performed on a wing–body configuration. The initial flow has large regions of separation, which are significantly diminished in the optimized configuration. Copyright © 2008 John Wiley & Sons, Ltd.

Received 8 January 2007; Revised 28 April 2008; Accepted 13 July 2008

KEY WORDS: discrete adjoint; recursive projection method; eigensystem analysis; aerodynamic optimization

1. INTRODUCTION

Adjoint equations are increasingly growing in importance in the field of computational aerodynamics, as emphasis shifts from the modelling of physical phenomena to their control and optimization. These applications introduce a class of minimization problems to the field, where the extremal value of some functional of the flow solution in some parameter space is desired. The parameterization of three-dimensional aircraft geometries can involve many hundreds of design parameters however [1], and all efficient solution approaches therefore rely on gradients of the functional in the design space. The gradients themselves can be most efficiently evaluated using

*Correspondence to: Richard P. Dwight, German Aerospace Center (DLR), Lilienthalplatz 7, Braunschweig D-38108, Germany.
†E-mail: richard.dwight@dlr.de

adjoint methods [2], that provide the gradient of a single functional with respect to a large number of parameters with a cost only very weakly proportional to the number of the parameters.

The first use of this kind of sensitivity analysis in fluid dynamics was in 1973 and it was due to Pironneau for the Stokes equations [3, 4]. However, it was not until 1988 that Jameson extended the procedure to inviscid compressible flows, enabling the optimization of transonic aerofoil profiles [5]. The discrete adjoint approach, where the discretized rather than the continuous equations are linearized, was first performed for the compressible Euler equations by Elbanna and Carlson [6], followed by Taylor *et al.* [7] and Baysal and Eleshaky [8]. A review of aerodynamic shape optimization techniques may be found in [9] and of sensitivity analysis and optimization for complex configurations in [10]. We also note a recent collection of literature on aerodynamic design [11].

However, the adjoint equations result directly from a linearization of the non-linear equations, and their solution is hence a problem of comparable difficulty to the solution of the original problem and—as will be seen—it is sometimes significantly more difficult. The first issue is the evaluation of the discrete adjoint residual, which requires the transpose of the Jacobian of the spatial discretization. Typically, it is not practical to store this matrix explicitly, due to its size, but its inline evaluation for each residual would be too costly. In Section 2 a compromise is found based on a particular factorization of second-order terms in the spatial discretization.

Although it is possible to apply the same iterative methods to the adjoint problem as used for the non-linear problem as done in Section 4, often these do not converge due to the lack of asymptotic convergence of the method as applied to the original problem. It then becomes necessary to apply some stabilization technique to the iteration. The Recursive Projection Method (RPM) is one such technique and is considered in Section 5. The RPM method provides the dominant eigenmodes of the iteration as a side-effect, suggesting an eigensystem analysis of the original iteration with the aim of isolating the cause of the instability. In Section 6 this is performed for a transonic aerofoil with and without flow separation, and the separation is identified as the region in which the iteration amplifies the error.

Adjoint solutions for such flows are essential in practice, as demonstrated in Section 7, where drag minimization of wing–body aircraft geometry with a large region of corner and trailing-edge separation are considered. The separation are the dominant source of removable drag in this case, and after optimization with a gradient-based method, the area of detached flow is reduced significantly.

We forego a detailed description of the compressible Navier–Stokes equations and their unstructured finite volume discretization in the DLR *TAU*-Code, referring the interested reader to [12–15], and note only that the method uses the standard Jameson–Schmitt–Turkel (JST) flux with scalar artificial dissipation [16] and the well-known one-equation Spalart–Allmaras turbulence model with Edwards modification [17]. The resulting semi-discrete system is expressed as

$$\frac{\mathrm{d}W}{\mathrm{d}t} + R(W, \alpha) = 0 \tag{1}$$

where W is the vector of unknown flow variables, α is some arbitrary scalar parameter of the flow problem or geometry, t is the time and R is the *residual*. Given this we are interested in formulating and solving linear systems of the form

$$\frac{\partial R}{\partial W}\Theta = -\frac{\partial R}{\partial \alpha} \tag{2}$$

DOI: 10.1002/fld

for unknowns Θ, the *primal* equation, and more especially

$$\left[\frac{\partial R}{\partial W}\right]^{\mathrm{T}}\Lambda=-\left[\frac{\partial I}{\partial W}\right]^{\mathrm{T}} \tag{3}$$

for unknowns Λ and scalar *cost function* $I(W,\alpha)$, the *adjoint* equation. We denote by $\partial R/\partial W$ the *Jacobian* and the term *linear problem* refers to either (2) or (3). Given the solution to one of these, the derivative of I with respect to α may be expressed as

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha}=\frac{\partial I}{\partial\alpha}+\left[\frac{\partial I}{\partial W}\right]^{\mathrm{T}}\Theta=\frac{\partial I}{\partial\alpha}+\Lambda^{\mathrm{T}}\frac{\partial R}{\partial\alpha} \tag{4}$$

whereby evaluating this expression for m different α and n different I requires either m solutions of (2) or n solutions of (3). For typical aerodynamic optimization problems $m\gg n$, as we are concerned with a few output forces, and the parameterization of 3d geometries, and the adjoint becomes significantly more efficient.

2. APPROXIMATION OF THE JACOBIAN

The convective terms within (1) are discretized using the JST scheme, the numerical flux across a face $\{ij\}$ with normal vector n_{ij} being expressed as

$$\begin{aligned}\hat{f}^{\mathrm{c}}_{ij}&=\tfrac{1}{2}(f^{\mathrm{c}}(W_i)+f^{\mathrm{c}}(W_j))\cdot n_{ij}\\&\quad-\tfrac{1}{2}|\lambda^{\mathrm{c}}_{ij}|\{\varepsilon^{(2)}_{ij}(W_j-W_i)-\varepsilon^{(4)}_{ij}(L_j(W)-L_i(W))\}\end{aligned} \tag{5}$$

where f^{c} is the exact convective flux, $|\lambda^{\mathrm{c}}|$ is the maximum local convective eigenvalue, $\varepsilon^{(2,4)}$ control the switching between first- and third-order dissipation and

$$L_i(W)=\sum_{k\in\mathcal{N}(i)}(W_k-W_i) \tag{6}$$

where $\mathcal{N}(i)$ is the set of all immediate neighbours of grid point i. A complete linearization of this flux may be found in [13], and covers three full pages. More problematic than this is that its stencil includes not only immediate, but also next-neighbouring points, and the Jacobian therefore has next-neighbour fill-in.

In previous work in 2d [18] this Jacobian was stored explicitly. Restarted GMRES was applied with an ILU(4) preconditioner as a solution method. This scheme provided an adjoint solution in approximately 5% of the time required for the non-linear field, but required a factor of 11 times more memory than the standard solver. This approach is no longer tenable in 3d as the memory requirements given in Table I show. This table was generated empirically by running the sequential codes for one 2d and one 3d turbulent test case with an LU-SGS smoothed 3W multigrid cycle. The discrepancy between the two sets of results is due to the much increased number of faces in 3d. Notable is the memory usage of the original non-linear code; we believe 1.2 million points in 1 GB of memory is unusually efficient for an unstructured solver. On the other hand, mere storage of the full Jacobian in 3d (without the additional memory needed by ILU and GMRES) multiplies the memory costs of the code by a factor of 13.5. Even though the approach reduces evaluation

DOI: 10.1002/fld

Table I. The memory requirements and run-time of the standard code and two linearized versions of the code for representative 2d (hybrid 28k points) and 3d (hybrid 120k points) grids.

	Standard *TAU*	Full Jac.*	Reduced Jac.*
Memory (Bytes) 2d	12M	100M	44M
Factor increase 2d	×1.0	×8.3	×3.7
Points in 1GB 2d	2.3×10^6	280×10^3	640×10^3
Time res. eval 2d	×1.0	×1.3	×0.7
Time Jac. eval 2d	—	×26.2	×2.6
Memory (Bytes) 3d	100M	1350M	400M
Factor increase 3d	×1.0	×13.5	×4.0
Points in 1GB 3d	1.2×10^6	89×10^3	300×10^3
Time res. eval 3d	×1.0	×1.7	×0.6
Time Jac. eval 3d	—	×28.0	×3.0

*Complete code, using same iterative procedure as standard *TAU*.

of the linear residual to a matrix–vector multiplication, this is a factor 1.7 times more expensive to evaluate than the non-linear residual due to the memory bandwidth bottleneck.

To alleviate these deficiencies the following device is introduced: by making the assumption that the ε and $|\lambda|$ are constant, a simplification of the adjoint residual is possible [18, 19]. Treat L as an independent variable, so that the derivative and adjoint of (5) may be expressed, respectively, as

$$\frac{\mathrm{d}\hat{f}^{\mathrm{c}}}{\mathrm{d}W}=\frac{\partial\hat{f}^{\mathrm{c}}}{\partial W}+\frac{\partial\hat{f}^{\mathrm{c}}}{\partial L}\frac{\partial L}{\partial W} \tag{7}$$

$$\left[\frac{\mathrm{d}\hat{f}^{\mathrm{c}}}{\mathrm{d}W}\right]^{\mathrm{T}}=\left[\frac{\partial\hat{f}^{\mathrm{c}}}{\partial W}\right]^{\mathrm{T}}+\left[\frac{\partial L}{\partial W}\right]^{\mathrm{T}}\left[\frac{\partial\hat{f}^{\mathrm{c}}}{\partial L}\right]^{\mathrm{T}} \tag{8}$$

whereby all matrices on the RHS of these equations have immediate neighbour fill-in only. Further $\partial\hat{f}^{\mathrm{c}}/\partial L$ is a multiple of the identity at each grid face and $\partial L/\partial W$ is trivial. If $\partial L/\partial W$ is calculated on-the-fly, memory requirements are only slightly greater than those required for a single Jacobian with immediate neighbour fill-in.

In addition, the matrices may be stored very naturally on the edges and nodes of the grid by which the adjoint residual may be evaluated in two loops over all edges by introducing an intermediate variable Λ^* as follows:

$$\Lambda^*=\left[\frac{\partial\hat{f}^{\mathrm{c}}}{\partial L}\right]^{\mathrm{T}}\Lambda \tag{9}$$

$$\left[\frac{\mathrm{d}\hat{f}^{\mathrm{c}}}{\mathrm{d}W}\right]^{\mathrm{T}}\Lambda=\left[\frac{\partial\hat{f}^{\mathrm{c}}}{\partial W}\right]^{\mathrm{T}}\Lambda+\left[\frac{\partial L}{\partial W}\right]^{\mathrm{T}}\Lambda^* \tag{10}$$

Since viscous and turbulence diffusion fluxes are used, which involve only gradients normal to grid faces (the so-called thin shear-layer discretization), the remaining terms of the Jacobian contain only immediate neighbour information and may be formed exactly. The performance of the resulting

DOI: 10.1002/fld

method is given in the third column of Table I, the factor 4 increase in memory being of particular importance, which is considered acceptable. If it happens to be unacceptable there remains the option of computing the Jacobian on-the-fly, giving a code of similar memory requirements to the original, but with a linear residual three times more expensive than the non-linear residual.

The assumption of constant dissipation coefficients necessarily affects the accuracy of the adjoint solution; however, this error was systematically investigated in [18] together with several other Jacobian approximations, and found to be insignificant for the purposes of gradient evaluation and optimization. A theoretical rationale for this result was provided in [13]. Note also that—depending on the form of the ε—by choosing suitable intermediate variables it may be possible to treat the complete dissipation operator in a similar manner, so that no constant coefficient approximation must be made. This step was considered unnecessary given the results of [18].

3. PARALLELIZATION OF THE ADJOINT CODE

Parallelization of the adjoint residual code does not fit easily into the pattern established by parallelization of the original code. This latter follows a domain decomposition approach; the global grid is divided into n non-overlapping point sets $\mathscr{K}=\{\mathscr{K}_1,\mathscr{K}_2,\ldots,\mathscr{K}_n\}$, each of which is supplemented by all points that immediately neighbour a point in the set and are not in the set themselves:

$$\hat{\mathscr{K}}_m=\{i\in\mathscr{K}\setminus\mathscr{K}_m:\exists j\in\mathscr{K}_m \text{ s.t. } i\in\mathscr{N}(j)\} \tag{11}$$

to give domains $\mathscr{D}_m=(\mathscr{K}_m,\hat{\mathscr{K}}_m)$. Variables defined at points in $\hat{\mathscr{K}}_m$ are copied from their native domain after they have been updated there, and before they are needed in $\mathscr{D}_m$. This parallel copying operation is the only parallel operation needed on variables defined at grid points. The variables defined at $\hat{\mathscr{K}}_m$ are never modified within $\mathscr{D}_m$.

To illustrate the difficulty with parallelization of adjoint consider the code sample

$$y_i:=x_j-x_i \tag{12}$$

evaluated on domain $\mathscr{D}_m$ and (borrowing the notation of algorithmic differentiation, see e.g. [2]) its adjoint

$$\bar{x}_i:=-\bar{y}_i,\quad \bar{x}_j:=\bar{y}_i \tag{13}$$

If $i,j\in\mathscr{K}_m$ there is no difficulty, but if $i\in\mathscr{K}_m$ and $j\in\hat{\mathscr{K}}_m$ ($j\in\mathscr{K}_l$ say), then the code requires that a point in $\hat{\mathscr{K}}_m$ be updated. There are two possible approaches: either transfer the second operation of (13) to $\mathscr{D}_l$, or perform it within $\mathscr{D}_m$ and communicate the result 'backwards' to $\mathscr{D}_l$. In most circumstances the former is more natural, but in situations where the stencil is not symmetric, e.g. where there is no corresponding $y_j:=x_i-x_j$ to (12) above, the latter can be simpler. For the second approach, it should also be carefully noted that a single point may receive contributions from multiple neighbouring domains during one 'backward' communication. In any case, each such occurrence must be explicitly handled if the parallelized code is to be correct.

As an example, in the present effort $\partial\hat{f}^{\mathrm{c}}/\partial W$ is not transposed in memory as would be required for evaluation of $R_j^{\mathrm{adj}}:=[\partial\hat{f}_i^{\mathrm{c}}/\partial W_j]^{\mathrm{T}}\Lambda_i$ local to $\mathscr{D}_l$. Rather R_j^{adj} is evaluated on $\mathscr{D}_m$ and communicated. The complete resulting parallel adjoint code has negligibly more communication than the original code and therefore scales similarly with number of processors.

DOI: 10.1002/fld

4. APPLICATION OF EXISTING FIXED-POINT ITERATIONS

Efficient solution methods for the non-linear problem (1) have been an important topic of research for some time, and significant experience has been gained in this area. Most methods currently in use employ some form of *fixed-point iteration* (FPI), a class of methods including Runge–Kutta, multigrid and implicit schemes, which may all be written

$$M(W^{n+1}-W^n)=-R(W^n) \tag{14}$$

for some *preconditioning* matrix M. Given the level of advancement of these techniques, it is worth considering whether they can be applied to the solution of the corresponding linear equations. In fact by rearranging (14), and linearizing about $\tilde{W}$, where $R(\tilde{W})=0$, we have

$$W^{n+1}=(I-M^{-1}A)W^n+g(\tilde{W})+\mathcal{O}\|W^n-\tilde{W}\|^2 \tag{15}$$

where A is the Jacobian evaluated at the exact solution and $g(\cdot)$ is some function independent of n. The relation to the linearized problem is immediately apparent, and informed by this we consider the FPI

$$M(\Theta^{n+1}-\Theta^n)=\frac{\partial R}{\partial \alpha}-A\Theta^n \tag{16}$$

or rearranged:

$$\Theta^{n+1}=(I-M^{-1}A)\Theta^n+M^{-1}\frac{\partial R}{\partial \alpha} \tag{17}$$

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 behaviour is identical. However, this is only true if A is the exactly formulated Jacobian based on a highly converged non-linear solution. 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

$$M^{\mathrm{T}}(\Lambda^{n+1}-\Lambda^n)=\left[\frac{\partial I}{\partial W}\right]^{\mathrm{T}}-A^{\mathrm{T}}\Lambda^n \tag{18}$$

so that

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

where $M^{-\mathrm{T}}$ denotes the inverse transpose of M. Since any real matrix and its transpose have identical eigenspectra, the asymptotic convergence rate of (19) will be the same as that of (17).

Further, by requiring that $\Theta^0=\Lambda^0=0$, and expanding (17) and (19) to the Nth iteration we have, respectively,

$$\Theta^{N+1}=-\sum_{n=0}^{N}(I-M^{-1}A)^nM^{-1}\frac{\partial R}{\partial \alpha} \tag{20}$$

$$\Lambda^{N+1}=-\sum_{n=0}^{N}(I-M^{-\mathrm{T}}A^{\mathrm{T}})^nM^{-\mathrm{T}}\left[\frac{\partial I}{\partial W}\right]^{\mathrm{T}} \tag{21}$$

DOI: 10.1002/fld

given which it quickly follows that

$$[\Lambda^{N+1}]^{\mathrm{T}} \frac{\partial R}{\partial \alpha} = \left[\frac{\partial I}{\partial W}\right]^{\mathrm{T}} \Theta^{N+1} \tag{22}$$

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

These two properties identical asymptotic convergence, and a relation on the transient convergence are very desirable. The first effectively provides a guarantee that, given convergence of the non-linear problem, the linear problem will also converge. Ideally this would mean that the level of intervention required from a user of the code to run the linear code subsequent to the non-linear code is minimal; CFL numbers and such do not need to be adjusted. The second property, which holds to working accuracy, aids development of the adjoint code considerably by providing a means of verifying that the Jacobian and FPI have been correctly adjointed. For example this test revealed that the Laplacian residual smoothing present in the Runge–Kutta algorithm was not quite symmetric, as might have been expected.

In the present work the approximately factored implicit LU-SGS scheme [13, 20] is adjointed. Written

$$(\tilde{D}+\tilde{L})\tilde{D}^{-1}(\tilde{D}+\tilde{U})x = b \tag{23}$$

where $\tilde{D}$, $\tilde{L}$ and $\tilde{U}$ are the block diagonal, lower triangular and upper triangular parts, respectively, of an approximate first-order Jacobian $\tilde{A}$. The LHS matrix clearly identifies with M. The corresponding adjoint iteration is therefore

$$(\tilde{D}+\tilde{U})^{\mathrm{T}}\tilde{D}^{-\mathrm{T}}(\tilde{D}+\tilde{L})^{\mathrm{T}}x = b \tag{24}$$

where the order of the forward and backward sweeps has been reversed by the application of the transpose. Similarly the adjoint of linear multigrid may be found by writing the iteration in matrix notation and transposing. As a result, the transpose of the original interpolation operator becomes the adjoint prolongation operator, and the transpose of the prolongation the adjoint interpolation. Also if the relaxation steps are performed before the sweep in the original method, they are performed after the sweep in the adjoint method. The adjointing of Runge–Kutta is somewhat more involved [21], but follows the same basic pattern of writing the iteration in matrix notation and transposing.

In all cases it is necessary to work in the same variables as the non-linear iteration (which is in conservative flow variables), if the iterations are to be consistent. If the Jacobian is formulated in terms of derivatives with respect to primitive variables it must first be transformed. The memory and CPU requirements of the adjoint FPIs are identical to their primal counterparts. Also of interest is that all the standard non-adjointed iterative methods perform very poorly on the adjoint problem. For the untransposed Runge–Kutta method on a simple adjoint test case, it was necessary to reduce the CFL number to $\frac{1}{4}$ of its original value to achieve stability. This effect was not as significant for multigrid, likely due to the similarity of the original and adjoint formulations.

A numerical demonstration of the two properties derived here is shown in Figure 1 using LU-SGS and a 3W multigrid cycle for a transonic NACA0012 aerofoil. Plotted are the convergence of the non-linear, linear and adjoint codes, together with the gradient $\mathrm{d}C_{\mathrm{D}}/\mathrm{d}\alpha$ from the primal and adjoint (where here α is the angle-of-attack), as well as the difference between these two gradients. Note that the asymptotic convergence of all codes is identical and the error in the two versions

DOI: 10.1002/fld

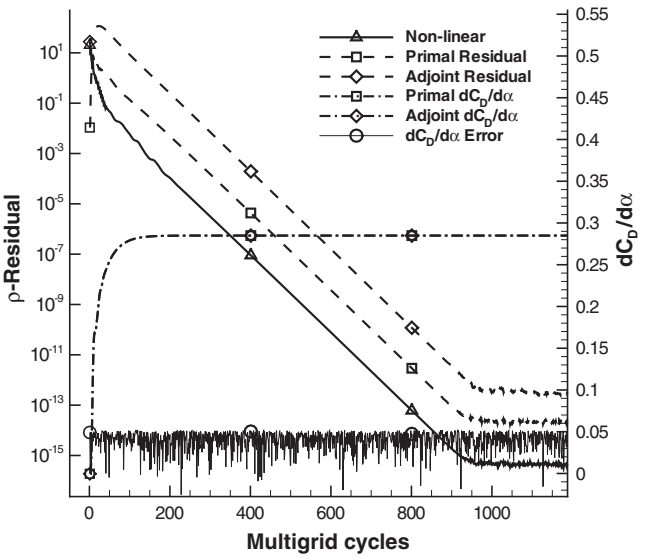

Figure 1. Convergence of non-linear, primal and adjoint codes for an NACA0012 test case. Also shown are transients of $\mathrm{d}C_\mathrm{D}/\mathrm{d}\alpha$.

of $\mathrm{d}C_\mathrm{D}/\mathrm{d}\alpha$ rests at the truncation error of the floating point arithmetic used. The monitoring of such gradients is also a useful way for a user of the code to estimate the progress of convergence, corresponding to monitoring of forces in the non-linear case.

5. THE RECURSIVE PROJECTION METHOD

Despite the guarantees regarding convergence provided by the theory of the previous section, 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 three reasons, all of which violate the parity of (14) and (16). Either (a) the non-linear solution is not sufficiently converged or (b) there is a discrepancy between the linear and non-linear problem due to some approximation of the Jacobian or (c) the FPI applied to the non-linear problem does not converge asymptotically itself.

All three cases appear regularly in practice. An engineer may reasonably consider a computation converged when the integrated forces that interest her no longer vary significantly, though this may occur prior to the asymptotic regime. With regard to (b), as already described, the Jacobian is simplified to obtain a scheme with reasonable memory costs. Although the simplification is very slight, it is not difficult to find a situation in which the non-linear problem converges and the linear diverges, simply by increasing the CFL number until the iteration is right on its stability boundary for the non-linear problem.

However, experience suggests the most significant is the third case—particularly for problems involving complex geometries—where minor unsteady physical phenomena are almost always present and reflected in the FPI due to the pseudo-time marching approach, leading to eventual

DOI: 10.1002/fld

stalling or limit cycles in the convergence. Treating these cases strictly correctly would require an unsteady computation, which may be an order of magnitude more expensive than a stationary calculation, and in practice the latter tends to converge to an acceptable level anyway. However, the lack of true asymptotic convergence often leads to instability in the linear problem.

In an effort to understand and mitigate these phenomena, we consider the RPM, originally developed by Schroff and Keller in 1993 for the purposes of bifurcation analysis and stabilization of unstable procedures [22, 23]. Since then it has been applied in aerodynamics for convergence acceleration [24, 25], and stabilization of linear frequency domain solvers [26].

The idea is to regard the transient solution of the linear problem as a sum of eigenvectors of the relaxation operator $\Phi=(I-M^{-1}A)$. The application of Φ to an approximate solution then corresponds to a product of each eigenvector with its corresponding eigenvalue. Divergence of the iteration implies that there is at least one eigenvalue of Φ with modulus greater than unity. Assuming that the number of such eigenvalues is small, and that the space spanned by their eigenvectors is known, call it $\mathscr{P}$, then it must be possible to solve the projection of the problem onto this low-dimensional subspace using some expensive but stable method, while solving the projection onto the complimentary subspace $\mathscr{Q}$ using the original FPI iteration, which is known to be stable there.

Newton–Raphson is typically used on $\mathscr{P}$. The space of dominant eigenvectors is determined as the calculation progresses by applying the principle that the difference between successive applications of the FPI on $\mathscr{Q}$ form a power iteration on the dominant eigenvalues of Φ restricted to $\mathscr{Q}$.

In detail, consider the relaxation operator of (17) written

$$N(x)=(I-M^{-1}A)x+M^{-1}b \tag{25}$$

where x and b are the unknown and RHS, respectively. Let V be an orthonormal basis of $\mathscr{P}$, then orthogonal projection operators onto $\mathscr{P}$ and $\mathscr{Q}$ may be written, respectively, $P=VV^{\mathrm{T}}$ and $Q=I-VV^{\mathrm{T}}$. Further define $x_P=Px$, $x_Q=Qx$. Then the RPM iteration may be written

$$x_Q^{n+1}=QN(x^n) \tag{26}$$

$$x_P^{n+1}=x_P^n+(I-P\Phi P)^{-1}[PN(x^n)-x_P^n] \tag{27}$$

where

$$x^{n+1}=x_P^{n+1}+x_Q^{n+1} \tag{28}$$

The 'derivative' term in the Newton iteration (27) may be rearranged as follows:

$$(I-P\Phi P)^{-1}=V(I-V^{\mathrm{T}}\Phi V)^{-1}V^{\mathrm{T}} \tag{29}$$

$$=V(I-H)^{-1}V^{\mathrm{T}} \tag{30}$$

where H is a square matrix of size dimension of $\mathscr{P}$, whose inversion is cheap if $\mathscr{P}$ is of low dimension. The inversion is performed only once for each basis, using LU factorization [27]. Also the term $PN(x^n)$ in the Newton iteration does not require an additional residual evaluation, as $N(x^n)$ is already available from (26). The projection of this vector onto $\mathscr{P}$ is the only mesh-size-dependant part of the Newton iteration (excluding the one-time evaluation of H), which is therefore extremely cheap.

DOI: 10.1002/fld

To determine the basis itself consider the sequence

$$K=[\Delta x_Q^n, \Delta x_Q^{n-1}, \ldots, \Delta x_Q^{n-k+1}] \tag{31}$$

where $\Delta x_Q^n = x_Q^{n+1} - x_Q^n$ are readily available from the iteration on $\mathscr{Q}$. For a general non-linear N, a Taylor expansion of $QN(x^n)$ gives

$$x_Q^{n+1} = QN(x^n) \tag{32}$$

$$= Q\left(N(x^{n-1}) + \frac{\partial N}{\partial x}\Delta x^n + \mathcal{O}(\Delta x^n)^2\right) \tag{33}$$

$$= Q(x^n + \Phi\Delta x^n + \mathcal{O}(\Delta x^n)^2) \tag{34}$$

$$= x_Q^n + Q\Phi Q\Delta x^n + Q\Phi P\Delta x^n + \mathcal{O}(\Delta x^n)^2 \tag{35}$$

where $Q\Phi P \approx 0$ because $\mathscr{P}$ is an approximately invariant subspace of Φ, so

$$\Delta x_Q^n = x_Q^{n+1} - x_Q^n \tag{36}$$

$$= Q\Phi Q\Delta x^{n-1} + \mathcal{O}(\Delta x_Q^n)^2 + \mathcal{O}(\Delta x_P^n)^2 \tag{37}$$

In our case N is linear, so all higher-order derivatives in the Taylor expansion vanish and $Q\Phi P = 0$, so

$$\Delta x_Q^n = Q\Phi Q\Delta x^{n-1} \tag{38}$$

Therefore, K is the k dimensional Krylov space for $Q\Phi Q$ seeded with Δx_Q^0. Asymptotically this subspace will tend to contain the dominant k eigenvectors of $Q\Phi Q$, as they are the components of Δx_Q^0 most amplified by repeated application of the operator.

Via QR factorization an orthogonal basis $\bar{Q}$ for K is obtained

$$K = \bar{Q}\bar{R} \tag{39}$$

whereby column pivoting is used such that the sequence of diagonal elements $|\bar{R}_{ii}|$ of the upper triangular matrix $\bar{R}$ is decreasing. Additional vectors for V are then chosen on the basis of the *Krylov Acceptance Ratio*, κ. For the largest i satisfying

$$\frac{|\bar{R}_{ii}|}{|\bar{R}_{i+1i+1}|} > \kappa \tag{40}$$

the first i columns of $\bar{Q}$ are added to V. Since these vectors lie in $\mathscr{Q}$ they are already orthogonal to the existing V, and further orthogonalization is not necessary.

The user-defined constant κ controls the accuracy, and indirectly the size, of the basis. If κ is very large the largest eigenmodes must dominate the others by a large margin. This corresponds to many applications of the operator N, and hence many iterations, but also implies that the power iteration will be more highly converged, and the resulting basis therefore more accurate. Unfortunately κ is highly problem dependant, as the convergence of the power iteration depends on the distribution of the dominant eigenvectors. Based on our experience we choose κ in the

DOI: 10.1002/fld

range 100–10 000. In practice it is seen that the stability and convergence of the RPM iteration is quite sensitive to the choice of κ. For this reason in future we intend to investigate a modification of RPM that has already been used for non-linear problems, where the accuracy of each vector in the basis is continually assessed during the course of the iteration, and vectors are removed when their accuracy is deemed too low [24].

Note that eigenvalues of real matrices have either zero real part or occur in complex conjugate pairs. If (40) is to be able to identify these pairs, which grow at identical rates, then the dimension k of the Krylov subspace must be at least 3. Similarly if there are n eigenvalues of Φ of *identical* modulus then k must be at least $n+1$. This latter situation has not been observed in practice for $n>2$.

The storage of V and K dominates the memory requirements of the method, so it is advantageous to keep k small. In practice the number of dominant eigenvalues treatable by the Newton method is limited by the storage of V, rather than the CPU cost. In the following we set $k=3$ and place an upper limit of 20 on the size of V.

Since $\mathscr{P}$ contains the dominant eigenvalues of Φ it is possible to solve an eigenvalue problem on this low-dimensional subspace to obtain estimates for its eigenvalues and eigenvectors, which may allow more detailed analysis of the FPI with respect to a specific problem on a specific grid. The eigenvalues of Φ restricted to $\mathscr{P}$ are given by

$$P\Phi P\bar{\Omega}=\bar{\Omega}\bar{\Xi} \tag{41}$$

Rearranging $P\Phi P$ as in (29) we have

$$H\Omega=\Omega\Xi \tag{42}$$

where Ω has k columns consisting of eigenvectors, and Ξ is the diagonal $k\times k$ matrix of eigenvalues. The sought dominant eigenvectors of Φ are then

$$\bar{\Omega}=V\Omega \tag{43}$$

A major advantage of the RPM is that its implementation requires no modification of the flow solver, requiring as it does only application of solver FPI iterations to given input vectors. This makes it possible to implement the entire algorithm in Matlab [25], calling the external flow solver each time $N(x)$ is required. In contrast to the use of GMRES as a stabilizer, RPM does not require *a priori* knowledge of the size of the unstable subspace, which is expanded as necessary.

As for all Krylov methods, RPM does not offer grid-independent convergence, and is therefore likely to scale well with problem size only in combination with a FPI containing multigrid. Further, as problem complexity increases, the number of unstable modes is also likely to increase, resulting in a larger basis with its associated costs.

6. INFLUENCE OF DETACHED FLOW ON LINEAR CONVERGENCE

To demonstrate the effect unsteady flow phenomena can have on the convergence of the adjoint problem, and to evaluate RPM, we consider two cases: the RAE 2822 aerofoil Case 9 and Case 10 [28]. The only differences between these two cases are the Mach and Reynolds numbers, but Case 9 is fully attached while Case 10 has a large region of shock-induced separation, and a small separated region near the trailing edge. These physics can be seen in Figures 4 and 5, whereby a negative skin friction coefficient indicates a region of reversed flow above the surface.

DOI: 10.1002/fld

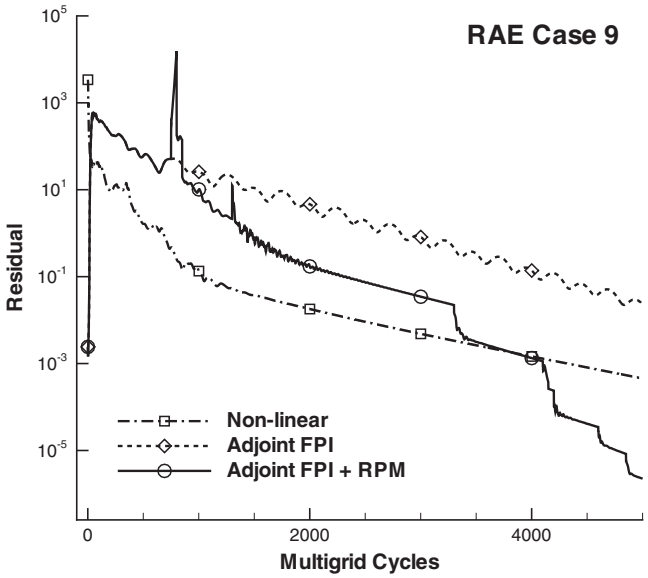

Figure 2. Convergence of non-linear and adjoint problems for RAE Case 9. The adjoint problem is solved once with a standard FPI alone, and once with RPM stabilization.

Convergence histories of the flow solutions are shown in Figures 2 and 3, respectively, where LU-SGS with multigrid with exactly the same settings is used in both cases. The attached case eventually reaches a region of asymptotic convergence, but Case 10 enters a limit cycle after about 1000 iterations and converges no further. However, by this point the lift and drag are well converged, and an engineer might reasonably be satisfied with these values. In fact the parameters of LU-SGS were specially chosen such that this situation would occur, in an attempt to model a circumstance that is common for complex geometries.

The cause of the limit cycle in such cases is not immediately clear and is difficult to establish with any certainty. Most likely it is a result of physical unsteadiness of the detached flow being partially modelled by the FPI, which is ultimately a pseudo time-stepping iteration. However, it could equally be due to discretization problems, such as the limiter involved in the gradient evaluation oscillating between two or more values at some point in the field, or any other non-smooth part of the discretization oscillating.

At any rate, the lack of non-linear asymptotic convergence means that one sufficient condition for convergence of the linear problem has not been met. Figures 2 and 3 also show the convergence of the linear problem (without RPM), and as might be expected, the adjoint of Case 9 using the same FPI converges and that of Case 10 diverges. The engineer who then wishes to optimize Case 10 (perhaps to remove the separation) with an adjoint-based method cannot, even though she can reliably obtain force coefficients. This is clearly an unacceptable situation.

Consider now the application of RPM to the two cases, also shown in the two figures, the discontinuities in the residual history being the iterations at which the basis is extended. For Case 9 the original iteration is stable, so the dominant eigenmodes identified by RPM are stable modes with eigenvalues close to 1, which dominate the asymptotic convergence. Hence here RPM

DOI: 10.1002/fld

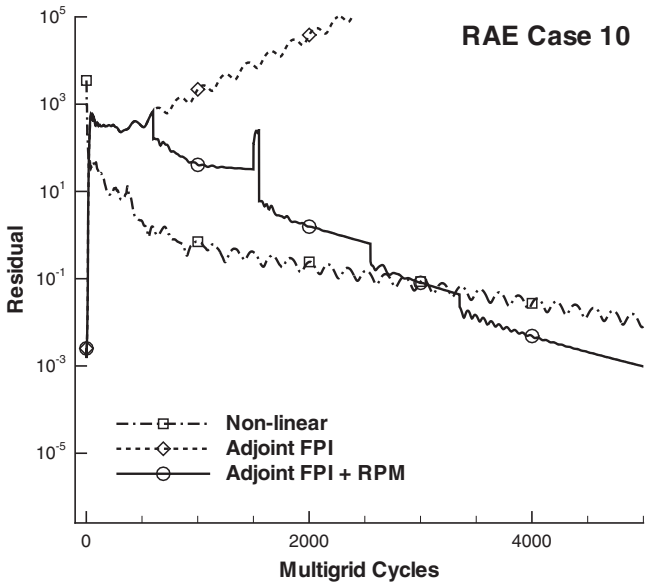

Figure 3. Convergence of non-linear and adjoint problems for RAE Case 10.

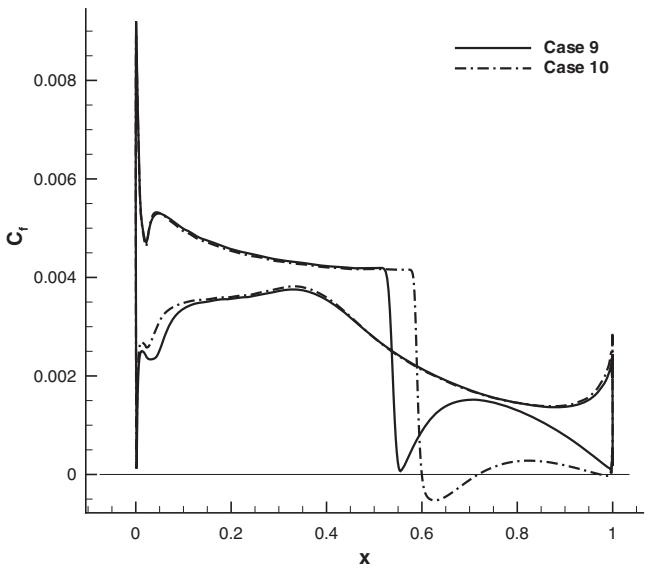

Figure 4. Skin friction coefficient on surface for RAE Cases 9 and 10.

operates as a convergence acceleration algorithm. To emphasize this effect, κ was set to a low value (200), as a result of which a basis is found very quickly, but is fairly inaccurate. This can be seen in the spike near the beginning of the convergence: initially the two curves are identical, then RPM finds a first basis vector whose low accuracy causes divergence until a second improved

DOI: 10.1002/fld

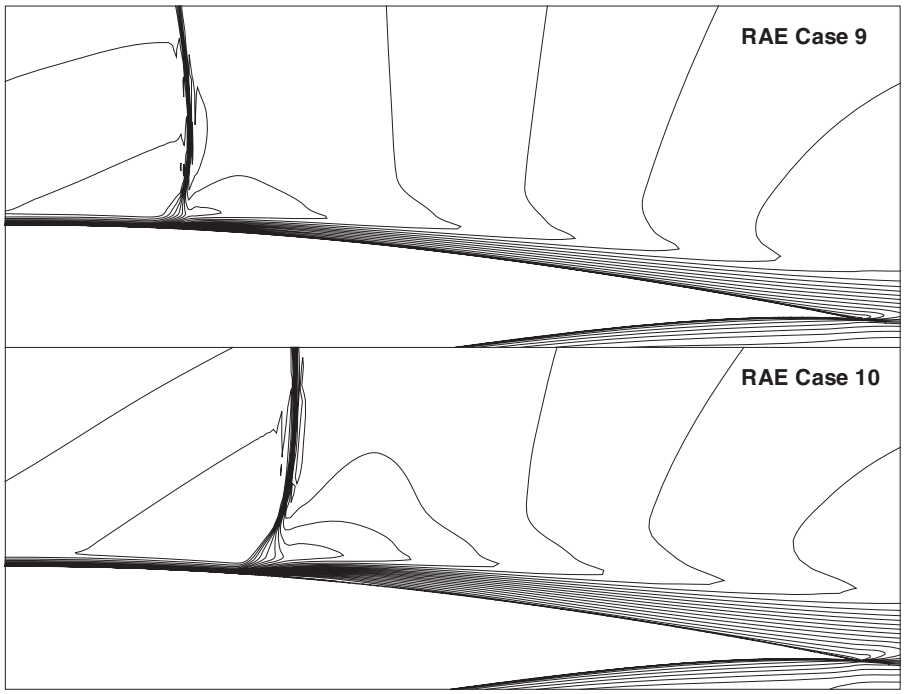

Figure 5. Contours of Mach number for RAE Cases 9 and 10.

vector is found. RPM then recovers rapidly giving superlinear convergence. Depending on case and the level of residual required RPM is typically 1.5–4 times faster than the original method in terms of CPU time as well as multigrid cycles.

However, we are more concerned with robustness and problems that do not converge initially. RPM with $\kappa = 200$ applied to the adjoint problem on Case 10 gives the convergence shown in Figure 3. The problem is solved in a similar CPU time to the stable case.

Approximations to the eigenvalues of the FPI operator, as obtained via the power iteration of RPM are plotted in Figures 6 and 7 for Cases 9 and 10, respectively. As must be the case—given the linear convergence behaviour—all eigenvalues of Case 9 lie within the unit circle, and therefore all modes converge. Four eigenvalues of Case 10 lie outside, and the eigenmodes corresponding to these eigenvalues are amplified at each iteration of the scheme. Effectively, the diverging components of the problem have been isolated.

Plotting the eigenvectors themselves allows identification of those *regions* of the flow responsible for slow convergence and divergence, respectively. Each eigenvector has a similar structure to a solution vector, with five complex components at each grid point; in Figures 8 and 9 the L^2 norm at each point is taken, representing the overall size of the vector in all components at that point. The two eigenvectors shown correspond to the eigenvalues marked `a` and `b` in the previous figures, and it should be noted that they therefore show different things: for Case 9 parts of the field that are most slowly damped, for Case 10 parts of the field that are diverging.

From these eigenmodes much information may be obtained that is ordinarily unavailable. For example, it becomes clear that in Case 9 the convergence rate of the iteration is limited by

DOI: 10.1002/fld

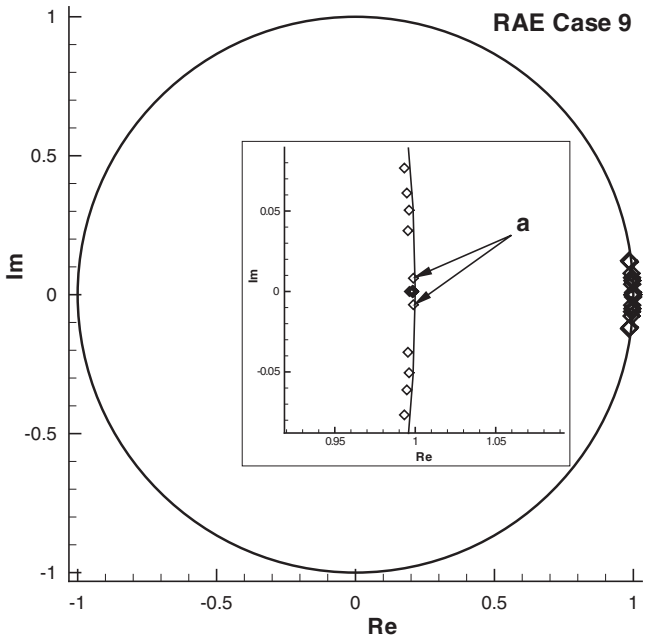

Figure 6. Approximate dominant eigenvalues of LU-SGS with multigrid applied to RAE Case 9, determined with a power iteration.

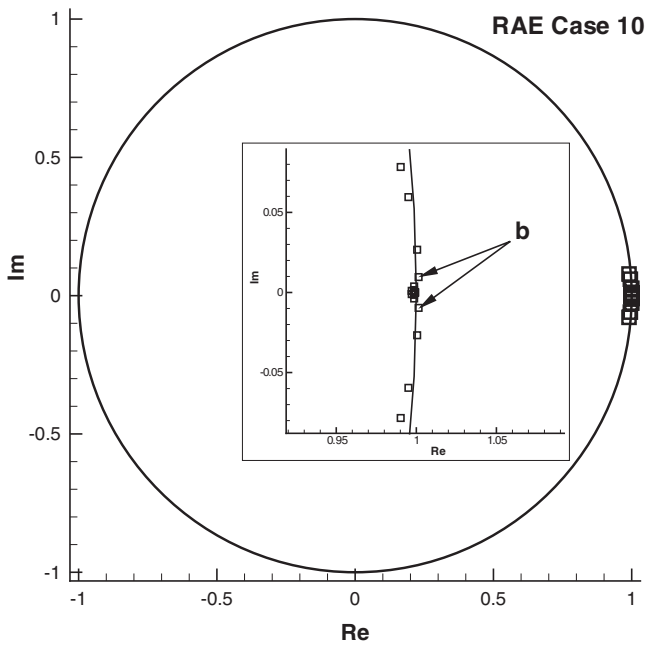

Figure 7. Approximate dominant eigenvalues of LU-SGS with multigrid applied to RAE Case 10.

DOI: 10.1002/fld

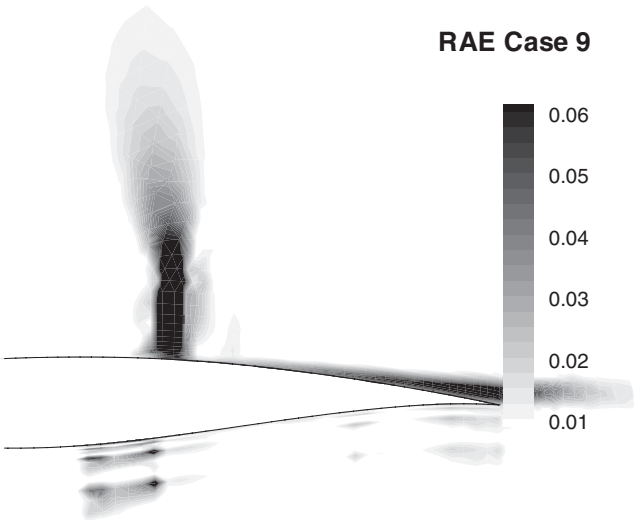

Figure 8. The dominant eigenvector of Case 9, corresponding to the eigenvalue marked a in Figure 6.

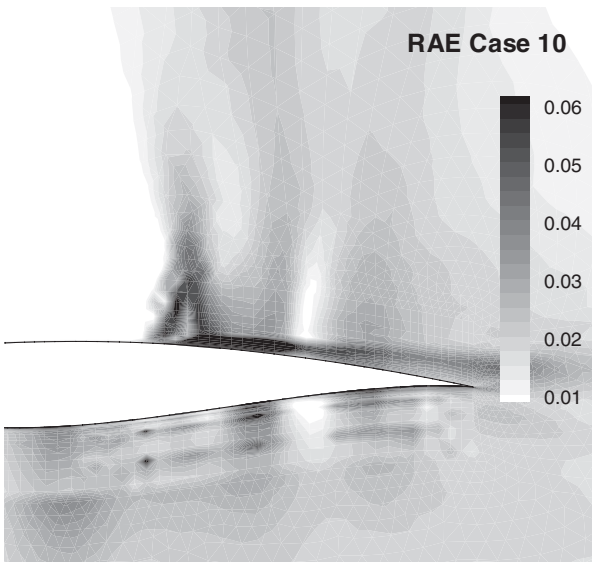

Figure 9. The dominant eigenvector of Case 10, corresponding to the eigenvalue marked b in Figure 7.

the convergence of the flow in the regions of the shock and upper surface negative pressure gradient. Devising an FPI that improves the convergence somewhere else, for example, near the stagnation point, would therefore result in no overall improvement in the asymptotic convergence rate. Similarly in Case 10 the cause of the non-linear limit cycle instability has been positively

DOI: 10.1002/fld

identified as the separation, in particular the largest recirculating region immediately behind the shock. Any treatment of the instability, in either the non-linear or linear problem, must necessarily consider the detachment. Additionally in both modes dark spots appear in the field under the aerofoil where no special physical features are present, indicating mesh or discretization problems.

Of course the eigenmodes depend on the FPI, and so the analysis relates to physical phenomena only over the discretization. On the other hand this is a feature that makes the analysis useful in the study of FPIs. Of particular interest is that, in all cases examined so far with no exceptions, those with some area of detached flow required stabilization for solution of the linear problem, and those without separation were stable under the FPI alone. Further investigations will concentrate on the cause of this phenomenon. Also eigenmode analysis will be applied to attempt to systematically categorize the behaviour of some common FPIs with respect to certain flow features, the goal being to quantify the *local* influence of, for example different directional multigrid coarsening algorithms. The wider use of Krylov methods also introduces the need for an FPI that is a good Krylov preconditioner, though not necessarily a good multigrid smoother. Where previously Fourier analysis was an essential tool for studying multigrid smoothers, we expect eigenmode analysis to be useful for studying Krylov preconditioners.

7. GRADIENT-BASED OPTIMIZATION

The first application of the adjoint method in aerodynamics was to flow control [29, 30], and the discrete adjoint method developed in the previous sections is applied here in a similar context.

7.1. Optimization of a transonic aerofoil

Consider drag reduction of a transonic RAE 2822 single element aerofoil at a Reynolds number of 6.5×10^6, and a Mach number of 0.730, whereby the lift must be held constant at a lift coefficient of 0.8. The geometry is parameterized using 20 design variables that modify the camberline of the aerofoil with Hicks–Henne bump functions [31]. The thickness of the aerofoil is not permitted to change, and as a result no additional geometrical constraints are necessary. The baseline geometry has a strong shock on the upper surface, which is the main source of pressure drag; the optimization problem therefore substantially consists of the removal of the shock.

The lift constraint is enforced explicitly by varying the angle of attack during the evaluation of the drag in the non-linear RANS solver, the so-called *target-lift* mode. Because we wish to minimize the drag at the target lift C_{L}^*, rather than at the pre-existing lift C_{L}, the objective function must be modified to consider the lift constraint consistently [32]

$$I = C_{\mathrm{D}} - \frac{(\partial C_{\mathrm{D}}/\partial\alpha)}{(\partial C_{\mathrm{L}}/\partial\alpha)}(C_{\mathrm{L}} - C_{\mathrm{L}}^*) \tag{44}$$

a consequence of which is that the accuracy of the gradients of lift is also important for the optimization. The computation of the adjoint gradients also includes a finite difference component, due to the dependence of R and I on the computational grid X, which introduces terms $\partial R/\partial X$ and $\partial I/\partial X$ into (4). Since a explicit linearization of these terms is not presently available in the code, they are also evaluated using central differencing, at a cost of two mesh deformations per design variable. Hence increasing the number of design variables still incurs relatively minor additional

DOI: 10.1002/fld

costs, see for example [1]. The conjugate gradient (CG) method with line-searches, employing gradients from the adjoint procedure is used to solve the minimization problem.

As a preliminary step the accuracy of the adjoint gradients is evaluated by comparison with finite difference gradients, see Figure 10. Agreement with the pressure part of the drag is excellent, the visible differences in the friction drag gradients are principally due to the error in the finite difference approximation, which is large due to the small absolute value of the friction drag.

Having found the gradients to be satisfactory an optimization was performed, and the convergence is shown in Figure 11, where the squares represent adjoint gradient evaluations. Requiring

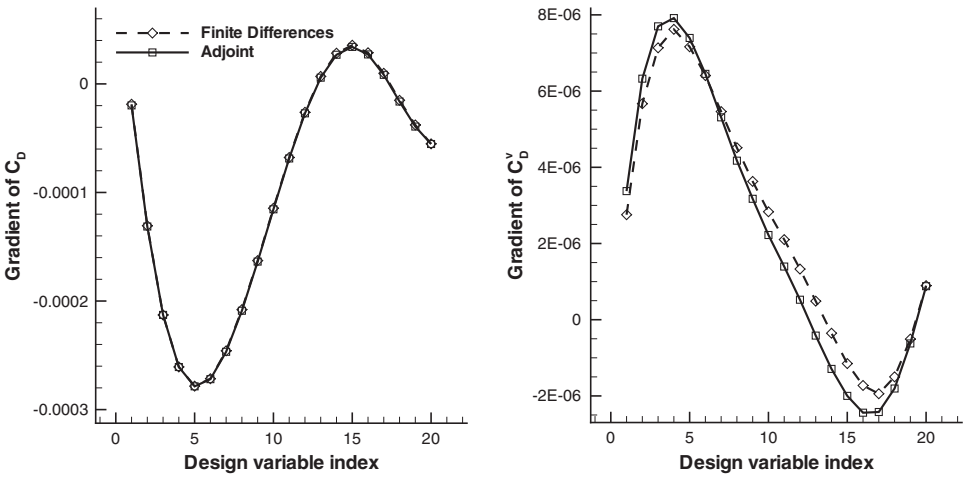

Figure 10. Gradients of total and viscous drag obtained using finite-differences and the discrete adjoint. The 20 design variables parameterize the aerofoil camberline from the leading- to trailing-edge.

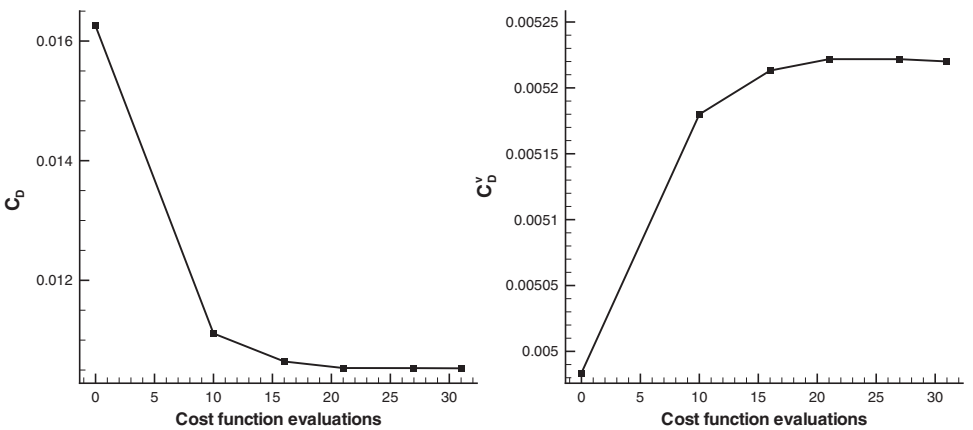

Figure 11. Convergence of the conjugate gradient algorithm for drag minimization of the RAE2822 at constant lift. The convergence is plotted against the number of cost-function evaluations (i.e. non-linear RANS computations).

DOI: 10.1002/fld

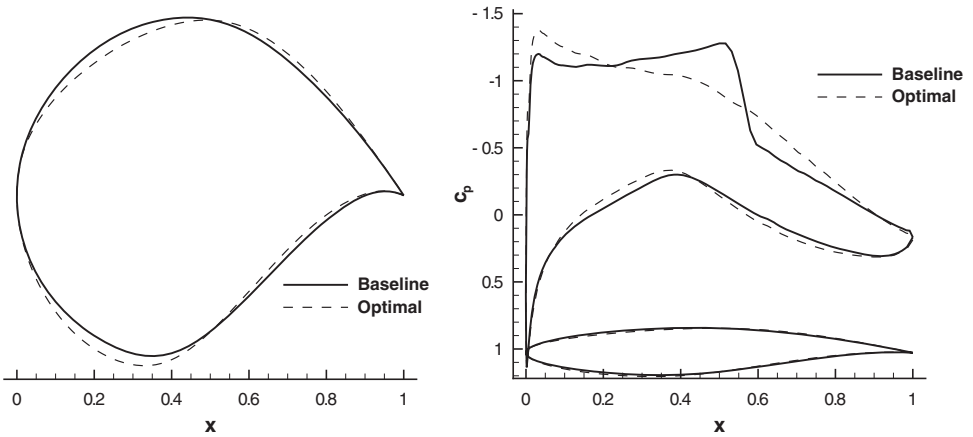

Figure 12. Baseline and optimized geometries and pressure distributions for the RAE.

only about 30 flow solutions, the drag is reduced by 35%, and by examining the surface pressure distribution, shown in Figure 12, the shock can be seen to have been removed completely. The baseline and optimal geometries are also shown in Figure 12.

7.2. Optimization of the DLR-F6 wing–body configuration

The next problem considered is again drag minimization at constant lift of the DLR-F6 wing–body configuration. We use a similar strategy to that already described in [18, 33]. The on-flow is at Mach 0.75, at a relatively low Reynolds number of 3×10^6 and a lift coefficient of $C_L=0.5$. The sub-design Reynolds number causes the flow to become detached over a large portion of the trailing edge, and there is additionally a region of separation on the upper surface of the wing where it meets the fuselage.

The surface of the computational grid is shown in Figure 13 and is entirely structured, a structured layer of constant thickness extends from this in order to resolve the boundary layer, the stacks are then topped with pyramids and the remainder of the domain is filled with tetrahedra [34]. The final mesh is coarse, with a total of 120 thousand points; however, it captures well the large regions of separated flow already mentioned. The memory requirements and relative performance of the adjoint solver for this specific case are given in Table I.

For the adjoint problem on this geometry, adjointed LU-SGS with multigrid alone was unconditionally unstable (as was Runge–Kutta) as expected, so that the only recourse was to a stabilized iteration. Figure 14 shows convergence of the RPM iteration with $k=3$ and $\kappa=1\times10^3$. The QR factorization of the Krylov space was evaluated every 50 multigrid cycles to determine if a new basis vector had been found. Convergence of the problem on the $\mathscr{P}$ and $\mathscr{Q}$ subspaces individually is also shown, along with the dimension of $\mathscr{P}$ (without axis) revealing when the basis updates were made. Owing to robustness problems, only the adjoint mean-flow equations were solved while the eddy-viscosity was frozen, resulting in a discrepancy between the non-linear and linear problems. The effect on the calculated gradients of this approximation was investigated in [18], and found to be acceptable for the purposes of optimization.

The corresponding eigenvalues, plotted in Figure 15, show that in total there were eight unstable modes. Examining the associated eigenvectors shows that they are all large in the regions of

DOI: 10.1002/fld

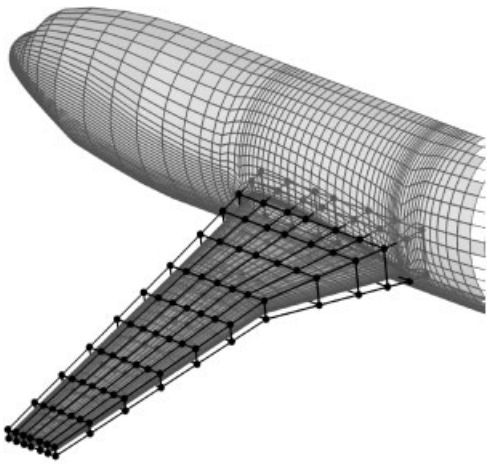

Figure 13. Surface mesh and parameterization of the DLR F6 geometry.

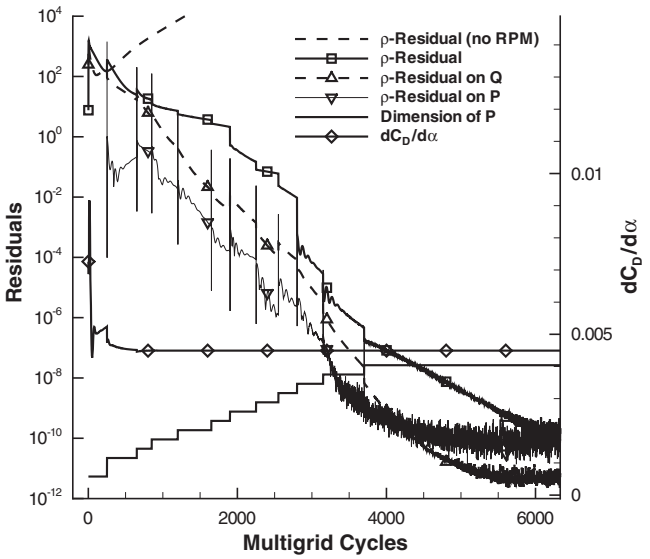

Figure 14. Convergence of the adjoint problem with and without RPM. The residuals on the subspaces $\mathscr{P}$ and $\mathscr{Q}$ are also shown.

separation, and particularly in the recirculating corner flow. The eigensystem analysis has therefore allowed the conclusive identification of the adjoint convergence difficulties with the separation in the flow.

The parameterization of the wing is shown in Figure 13, and is a combination of free-form deformation for the shape, and a superimposed twist parameter set. The dark lines show the position of the free-form deformation control box [35], the vertical positions of the 84 paired

DOI: 10.1002/fld

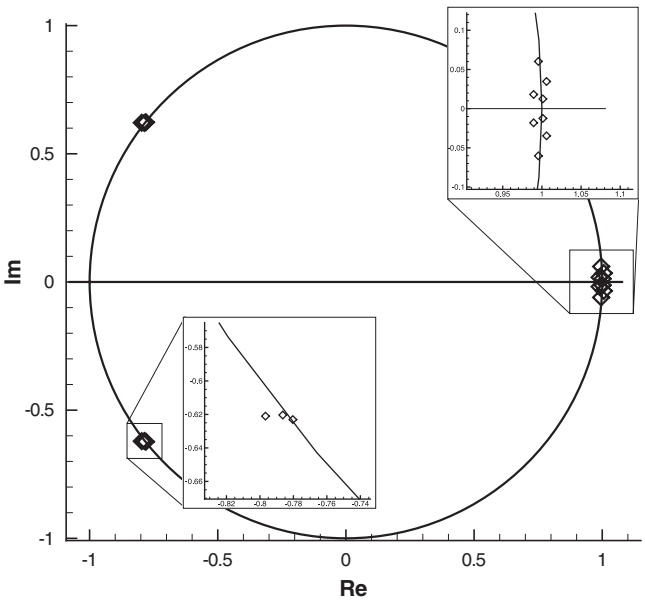

Figure 15. Dominant eigenvalues of LU-SGS smoothed multigrid applied to the F6 configuration, determined with RPM.

vertices marked with a black circle are the design variables. By locking the vertical movement of each pair of vertices together, the thickness of the wing is held constant, and the tendency of the optimizer to choose the thinnest possible wing (in the absence of structural information) is countered. Note that since the bounding box passes inside the fuselage, the wing–body junction also varies, and this is accounted for by the geometry and grid generation process. Twist of the wing is parameterized separately using a spline with 12 degrees of freedom. These design variables were chosen to be a good generic parameterization of the wing, without paying special attention to the separation region. One effective means of eliminating the corner separation entirely would be to allow deformation of the fuselage to allow fairing-like shapes, but this is not admitted in this conceptual study. The lift constraint is again met by modifying the angle-of-attack in the *target-lift* mode of the solver.

To examine the errors in the gradient produced due to the approximations of the Jacobian, the gradients of C_D with respect to a reduced set of eight design variables were computed using central finite differences with various step sizes ε, for the initial geometry. The adjoint gradients were calculated similarly, using various step sizes for the evaluation of the mesh sensitivities $\partial R/\partial X$. The results are shown in Figure 16 where it is apparent that the two sets of gradients agree well, and that the adjoint gradients are less sensitive to the step size—as might be expected.

Given the successful gradient validation we proceed to the optimization, the convergence of which is shown in Figure 17. The horizontal axis shows the number of calls to the flow solver (both linear and non-linear), thereby approximately representing the computational effort required. Symbols indicate gradient evaluations. As the scale of the design space is initially unknown the first gradient is used to estimate the step in the design space necessary to produce a 1% reduction in C_D. After 32 solver calls the conjugate gradient algorithm was unable to reduce the drag further, giving

DOI: 10.1002/fld

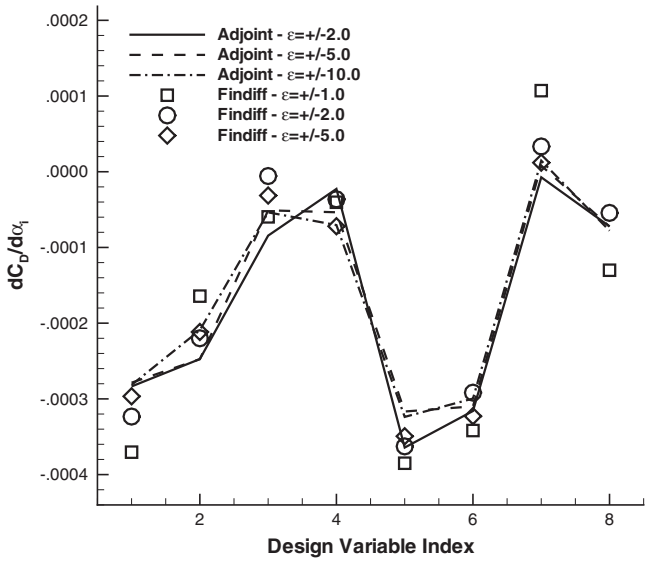

Figure 16. Comparison of gradients evaluated with finite differences and adjoint for various step sizes. For the adjoint gradients the step sizes are those applied in the determination of the mesh sensitivities.

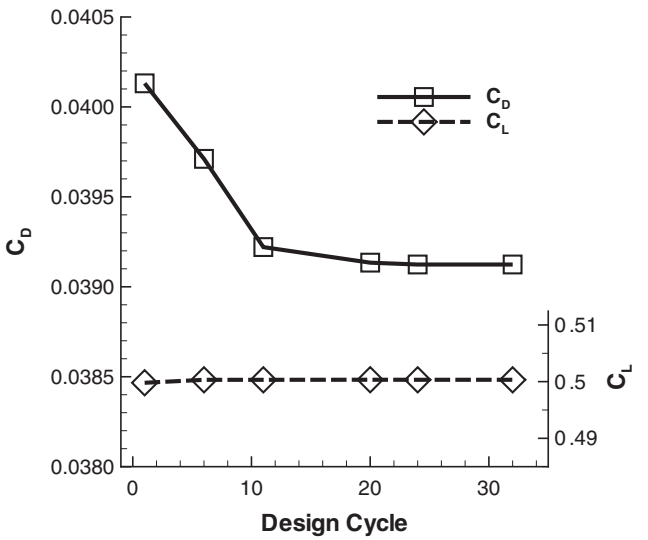

Figure 17. Convergence of the drag minimization F6 optimization, lift and drag.

a final reduction of about 10 drag counts. In contrast a similar optimization with 42 parameters produced a reduction of only 8 counts on the same mesh (in a similar CPU time) [1], emphasizing the need for a comprehensive parameterization.

DOI: 10.1002/fld

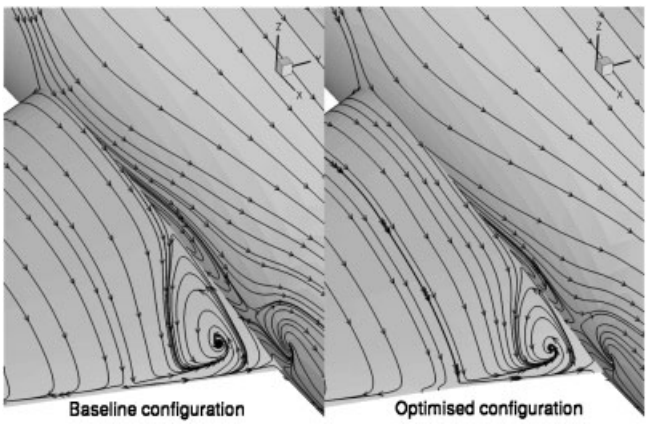

Figure 18. Effect of the optimization on the corner separation of the geometry.

The effect of the optimization was to reduce the region of corner separation considerably, which can be seen in Figure 18. This was achieved principally through a reduction in the angle of attack, with the associated loss of lift being compensated by an increase in wing twist. Though not completely eliminating the separation, it is doubtful that this is possible within the design space considered. A further optimization, using identical techniques but involving only a parameterization of the fuselage was performed, and it was found that by this means the recirculation could be completely removed.

8. CONCLUSIONS

Our immediate goal in the investigation of the discrete adjoint method was to develop a solver that is at least as robust and efficient as the non-linear solver, as only then can it reliably be applied to practical engineering problems in optimization and adaptation. The algorithms described in this paper go a long way towards achieving this goal, in particular the memory requirements and time per residual evaluation are acceptable. The adjoints of the non-linear solution algorithms allow efficient convergence, at least in cases that behave well non-linearly. For the cases that behave poorly RPM offers a means of increasing robustness and obtaining a solution, and is also efficient in memory provided there are not more than 10–20 unstable modes in the problem. However, there remain cases that are presently insoluble with reasonable computer resources. Some contain too many unstable modes, others diverge before RPM has the chance to find a basis (for example, with an apparent extremely large eigenvalue appearing in the turbulence equations). The tackling of these problems represents the next challenge. It is envisaged that use of eigensystem analysis will aid the design of FPIs especially suited as preconditioners to RPM and Krylov methods. The parallelization of these algorithms is also necessary for practical applications.

To demonstrate the power of the resulting adjoint solver it was applied to the optimization of a transport aircraft configuration with a large number of design variables, whereby a significant reduction in drag was achieved via a reduction in the extent of separated flow. This calculation would not have been possible without the developments in efficiency and robustness described in this paper.

DOI: 10.1002/fld

ACKNOWLEDGEMENTS

We are very grateful to Stefan Görtz for sharing his extensive insight into and experience of the recursive projection method.

REFERENCES

1. Brezillon J, Brodersen O, Dwight R, Ronzheimer A, Wild J. Development and application of a flexible and efficient environment for aerodynamic shape optimisation. *Proceedings of the ONERA-DLR Aerospace Symposium* (*ODAS*), Toulouse, 2006.
2. Griewank A. *Evaluating Derivatives*, *Principles and Techniques of Algorithmic Differentiation*. Frontiers in Applied Mathematics, vol. 19. SIAM: Philadelphia, 2000. ISBN 08-987-1451-6.
3. Pironneau O. On optimum profiles in Stokes flow. *Journal of Fluid Mechanics* 1973; **59**(1):117–128.
4. Pironneau O. On optimum design in fluid mechanics. *Journal of Fluid Mechanics* 1974; **64**(2):97–110.
5. Jameson A. Aerodynamic design via control theory. *Journal of Scientific Computing* 1988; **3**(3):233–260.
6. Elbanna H, Carlson L. Determination of aerodynamic sensitivity coefficients in the transonic and supersonic regimes. AIAA Paper Series, *Paper 89-0532*, 1989.
7. Taylor III A, Korivi V, Hou G. Sensitivity analysis applied to the Euler equations: a feasibility study with emphasis on variation of geometric shape. AIAA Paper Series, *Paper 91-0173*, 1991.
8. Baysal O, Eleshaky M. Aerodynamic design sensitivity analysis methods for the compressible Euler equations. *Journal of Fluids Engineering* 1991; **113**(4):681–688.
9. Dulikravitch G. Aerodynamic shape design optimization: status and trends. *Journal of Aircraft* 1992; **29**(6): 1020–1026.
10. Newman III J, Taylor III A, Barnwell A, Newman P, Hou G. Overview of sensitivity analysis and shape optimization for complex aerodynamic configurations. *Journal of Aircraft* 1999; **36**(1):97–116.
11. Uthup B, Koruthu S-P, Sharma R-K, Priyadarshi P (eds). *Recent Trends in Aerospace Design and Optimization*. Tata-McGraw Hill: New Delhi, 2005.
12. Gerhold T, Galle M, Friedrich O, Evans J. Calculation of complex 3D configurations employing the DLR TAU-Code. AIAA Paper Series, *AIAA-1997-0167*, 1997.
13. 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.
14. Kroll N, Rossow C-C, Schwamborn D. *The MEGAFLOW-Project—Numerical Flow Simulation for Aircraft*, Progress in Industrial Mathematics, vol. 8. Springer: New York, 2006. DOI: 10.1007/3-540-28073-1, ISBN 978-3-540-28072-9.
15. Gerhold T, Hannemann V, Schwamborn D. On the validation of the DLR-TAU Code. In *New Results in Numerical and Experimental Fluid Mechanics*, Notes on Numerical Fluid Mechanics, Nitsche W, Hilbig R (eds). Springer: Heidelberg, 1999; 426–433. ISBN 3-528-03122-0.
16. Jameson A, Schmidt W, Turkel E. Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes. AIAA Paper Series, *AIAA-1981-1259*, 1981.
17. Edwards J, Chandra S. Comparison of eddy-viscosity transport turbulence models for three-dimensional shock-separated flowfields. *AIAA Journal* 1996; **34**(4):1–16.
18. Dwight R, Brezillon J. Effect of various approximations of the discrete adjoint on gradient-based optimization. *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, AIAA-2006-0690, 2006.
19. Mavriplis D. Personal Communication, 2004.
20. Yoon S, Jameson A. An LU-SSOR scheme for the Euler and Navier–Stokes equations. *AIAA Journal* 1988; **26**:1025–1026.
21. Giles MB. On the iterative solution of adjoint equations. In *Automatic Differentiation*: *From Simulation to Optimization*, Corliss G, Faure C, Griewank A, Hascoet L, Naumann U (eds). Springer: Berlin, 2001; 145–152.
22. Schroff G, Keller H. Stabilization of unstable procedures: the recursive projection method. *SIAM Journal on Numerical Analysis* 1993; **30**(4):1099–1120.
23. Keller H. RPM: a remedy for instability. In *Collected Lectures on the Preservation of Stability under Discretization*, *SIAM Proceedings in Applied Mathematics*, vol. 109, Estep D, Tavener S (eds). 2002; 185–196.
24. Möller J. Aspects of the recursive projection method applied to flow calculations. *Ph.D. Thesis*, NADA, KTH, Stockholm, Sweden, 2005. ISBN 91-7283-940-6, TRITA-NA-0444.

25. Görtz S, Möller J. Evaluation of the recursive projection method for efficient unsteady turbulent CFD simulation. *ICAS*, Yokohama, Japan, 2004.
26. Campobasso M, Giles M. Stabilization of a linear flow solver for turbomachinery aeroelasticity by means of the recursive projection method. *AIAA Journal* 2004; **42**(9):1765–1774.
27. Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J, Du Croz J, Greenbaum A, Hammarling S, McKenney A, Sorensen D. *LAPACK Users' Guide* (3rd edn). Society for Industrial and Applied Mathematics: Philadelphia, PA, 1999.
28. Cook P, McDonald M, Firmin M. *Aerofoil RAE 2822—Pressure Distributions and Boundary Layer and Wake Measurements*. AGARD-AR, vol. 138, Chapter 6. AGARD—Advisory Group for Aerospace Research & Development: Neuilly-sur-Seine, France, 1979.
29. Pironneau O. On optimum design in fluid mechanics. *Journal of Fluid Mechanics* 1974; **64**:97–110.
30. Jameson A. Aerodynamic design via control theory. *Journal of Scientific Computing* 1988; **3**:233–260.
31. Hicks R, Henne P. Wing design by numerical optimization. *Journal of Aircraft* 1978; **15**:407–412.
32. Reuther J, Alonso J, Rimlinger M, Sanders D, Jameson A. Constrained multipoint aerodynamic shape optimization using an adjoint formulation and parallel computers. *Journal of Aircraft* 1999; **36**:51–60.
33. Brezillon J, Dwight R. Discrete adjoint of the Navier–Stokes equations for aerodynamic shape optimization. *Evolutionary and Deterministic Methods for Design*, *EUROGEN*, Munich, Germany, 2005.
34. Wild J. Acceleration of aerodynamic optimization based on RANS-equations by using semi-structured grids. *Design Optimization International Conference*, Athens, 2004.
35. Ronzheimer A. Shape based on freeform deformation in aerodynamic design optimization. *ERCOFTAC Design Optimization International Conference*, Athens, 2004.

DOI: 10.1002/fld