AERODYNAMIC OPTIMIZATION BASED ON THE EULER AND NAVIER-STOKES EQUATIONS USING UNSTRUCTURED GRIDS

by

JONATHAN KINDRED ELLIOTT

B.A.I, B.A. Trinity College, Dublin (1987)
S.M., Massachusetts Institute of Technology (1990)

SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1998

Author____
Department of Aeronautics and Astronautics
May 22, 1998

Certified by____________________________________
Jaime Peraire
Thesis Supervisor, Associate Professor of Aeronautics and Astronautics

Certified by____________________________________
Mark Drela
Associate Professor of Aeronautics and Astronautics

Certified by____________________________________
Anthony Patera
Professor of Mechanical Engineering

Accepted by____________________________________
Professor Jaime Peraire
Chairman, Department Graduate Committee

Aerodynamic Optimization Based on the Euler and Navier-Stokes Equations Using Unstructured Grids

by

Jonathan Kindred Elliott

Submitted to the Department of Aeronautics and Astronautics
on May 22, 1998, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

The overall problem area addressed is that of efficient aerodynamic shape design through the use of computational fluid dynamics (CFD). A method is presented for performing optimization — including modal inverse design and lift-constrained drag minimization — based on the 2D and 3D Euler equations and the 2D and 3D laminar Navier-Stokes equations. The discrete adjoint approach was taken and the adjoint solvers developed were based on flow solvers developed for use with unstructured grids. Optimization exercises are presented which demonstrate the effectiveness and practicality of the optimization system in finding credible optimal geometries.

Thesis Supervisor: Jaime Peraire
Title: Thesis Supervisor, Associate Professor of Aeronautics and Astronautics

Acknowledgments

Above all, I would like to express my deepest thanks to my advisor, Jaume Peraire. Time and again, when things seemed to be the most desparate, and the mathematics seemed to be betraying us, he inspired me to stick to fundamental mathematical principles and beat the code into submission.

To Mark Drela, I owe you my appreciation and thanks for innumerable assistances, hugely helpful discussions and the initial inspiration to work in the area of optimization.

To Brian Nishida, thanks for showing me the ropes at MIT again, and above all how to ping residuals.

To Bill Herling and Fritz Roetman I owe you thanks for keeping the project relevant to industrial concerns and providing geometry. To Steve Allmaras, I owe thanks for viscous grids, help with convergence acceleration and numerous other assistances. To the various groups in Boeing for providing, along with Mark Drela, my initial inspiration to work on optimization: Paul Frank, Jean-Francois Barthelemy, Jim Benton, Greg Shubin, Tim Purcell, Mitch Murray, Paul Bogataj, the TRANAIR team and Wen-Huei Jou, just to mention a few.

To my tireless other committee member, Anthony Patera, who was another huge source of inspiration. My committee was surely the best I could have had for this project. I am indebted to all of you.

Thanks to other researchers in the area and CFD in general for many fruitful discussions and/or interactions: Ken Morgan, Dave Darmofal, Tony Jameson, Jimmy Newman, Greg Burgreen, James Reuther, Juan Alonso, Mike Giles and Kyle Anderson.

Thanks to Tom Sorensen, Alex Budge, Ali Merchant, Bob Haimes, Bilal Mughal, Karen Willcox, Karen Bibb and Sean George for various technical discussions and assistance.

Thanks to my officemates, David Oh, Robie Samantha Roy and the folks in the cluster. Oh, and Karen Willcox. But I already mentioned you. You can't be in here twice. Oh well, too late.

On the personal side, thanks to my family who always provided a peaceful haven even in the roughest of weather. Especially Marc and Dad.

And finally, thanks to Annelene for everything. You helped keep a perspective on things. Sorry for not being more of a poet. Maybe I'll improve now...

Contents

List of Figures

List of Tables

Nomenclature

Symbols associated with flow and sensitivity analysis algorithms

$\mathbf{A}^j$	jth flux jacobian
$\mathbf{A}_{ik}$	standard Roe matrix for edge ik
b	wing span
c	chord
C_p	pressure coefficient
C_f	skin friction coefficient resolved in streamwise direction
C_{D_j}	drag coefficient for condition j
C_{L_j}	lift coefficient for condition j
$C_{L_j}^{targ}$	target lift coefficient for condition j
$e_i(y)$	component of mode h_{ij} dependent on y
E	total energy per unit volume
$f_j(x/c)$	component of mode h_{ij} dependent on x
$\mathbf{F}^j$	jth inviscid flux vector
$\mathbf{G}^j$	jth viscous flux vector
$\mathbf{G}_j$	short hand for gradient of $\mathbf{U}$ in direction j
$h_i(x,y)$	design mode associated with β_i
k	coefficient of thermal conductivity
L	limiter function
$\mathbf{L}$	preconditioning matrix for point implicit scheme
M	Mach number
p	pressure
p^*	target pressure
Pr	Prandtl number
$\mathbf{P}_{ik}$	matrix of eigenvectors of $\mathbf{A}_{ik}$
Re	Reynolds number
$\vec{S}$	area vector forming part of boundary of Ω_i
t	time
T	temperature
$\mathbf{U}$	state vector
u_i	component of velocity in direction x_i
$\mathbf{W}$	vector of characteristic variables
x_1,x_2,x_3	Cartesian coordinates
x,y,z	Cartesian coordinates
α_j	angle of attack at condition j
Γ	distance around perimeter bounding Ω

δ_{ij}	Kronecker delta
δ^*	boundary layer streamwise displacement thickness
η	nondimensional spanwise coordinate, $\frac{2y}{b}$
θ	local angle between meanline and x-axis
θ	boundary layer streamwise momentum thickness
$\mathbf{\Lambda}_{ik}$	matrix of eigenvalues of $\mathbf{A}_{ik}$
Λ_{LE}	leading edge sweep angle
Λ_{TE}	trailing edge sweep angle
μ	Mach wave angle
μ	coefficient of viscosity
ρ	density
τ_{ij}	shear stress acting on face i in direction j
ϕ	twist
$\boldsymbol{\psi}$	Lagrange multiplier associated with $\mathbf{R}$
Ω	control volume

Symbols associated with optimization algorithm

$\mathbf{a}_j$	jth constraint gradient, $(\partial c_j/\partial\boldsymbol{\beta})^T$
$\mathbf{A}_k$	constraint jacobian with rows, $\mathbf{a}_j^T$
$\mathbf{B}_k$	approximation to Hessian $\mathbf{G}_k$ at iteration k
$\mathbf{B}_Z$	approximation to projected Hessian $\mathbf{G}_Z$
c_j	constraint associated with condition j
$\mathcal{F}$	objective function
$\tilde{\mathcal{F}}$	quadratic model for objective function
$\mathbf{g}_k$	cost function gradient, $(\partial\mathcal{F}/\partial\boldsymbol{\beta})^T$
$\mathbf{g}_Z$	gradient in subspace Z, $\mathbf{Z}_k\mathbf{g}_k$
$\mathcal{L}$	Lagrangian
M	number of constraints
N	number of design variables
$\mathbf{p}_k$	search direction for iteration k
$\mathbf{p}_Y$	step in range space of $\mathbf{A}_k^T$ to ensure feasibility in line search k
$\mathbf{p}_Z$	search direction in subspace $\mathbf{Z}_k$, $\mathbf{Z}_k\mathbf{p}_k$
P_Q	cost function with quadratic penalty term
$\mathbf{q}_k$	step to return design to feasible space
$\mathbf{s}_k$	step size for iteration k
$\mathbf{Y}_k$	transformation matrix from full design space to subspace spanned by $\mathbf{a}_j$
$\mathbf{Z}_k$	transformation matrix from full design space to subspace orthogonal to all $\mathbf{a}_j$
$\boldsymbol{\beta}$	design variable vector
$\overline{\boldsymbol{\beta}}$	scaled design variable vector
δ	line search parameter
μ_k	Lagrange multiplier associated with c_k
ρ	penalty parameter associated with penalty term $\mathbf{c}^T\mathbf{c}$
ω_j	cost function weight for condition j

Chapter 1

Introduction

The use of Computational Fluid Dynamics (CFD) simulations in aircraft design is already well established. Much less common however, is the use of formal CFD-based optimization procedures to guide the design process. Traditionally, inverse design tools, whereby a prescribed pressure distribution is used to determine the geometry, have been used. The main limitation of inverse methods is that they require *a–priori* knowledge of the pressure distribution that will produce a well posed problem, and lead to a satisfactory solution. An alternative to inverse design is the use of optimization methods. In this case, the geometry is determined so as to minimize a cost functional, such as drag, subject to some constraints. These methods do not require *a–priori* knowledge of the pressure fields and therefore are much more general, and generate well defined problems. An additional advantage of optimization methods is that they can be easily incorporated into multi-disciplinary optimization formulations [44] which offer the potential for aircraft designs with significantly improved fuel burn, through a combination of lower aircraft weight and reduced drag. (For a discussion of successful approaches to incorporating CFD into the aircraft/airfoil design process, the reader is referred to Reference [40].)

1.1 Literature Review

The main obstacle to formal gradient-based optimization procedures is the high cost associated with the calculation of aerodynamic sensitivities. In recent years however, there has been a rapid growth in the use of adjoint methods for the calculation of cost function sensitivities. This is due to the huge potential savings that the method offers, over finite difference and direct method approaches, for optimization problems in which the number of design variables is large. The initial spark was provided by Jameson [25], but there have been other significant contributions [6, 24, 23, 43]. Adjoint methods for the calculation of sensitivities can be formulated in two different forms. The first approach is the variational sensitivity analysis approach, where an analytic equation for the adjoint function is derived and then discretized leading to an approximation to the sensitivities of the exact problem. This means that the validity of the computed sensitivities relies on the discretization errors being sufficiently small. The second is the discrete sensitivity analysis approach, where a discrete formulation of the adjoint problem is derived directly starting from the discretized flow equations. The sensitivities thus obtained are exact for the discrete problem regardless of the discretization errors.

A recent study by Jameson et al [27] has demonstrated drag minimization at fixed lift for the 3D Euler equations using single-block structured grids complemented by an inverse design capability, based on the 3D Navier-Stokes equations discretized on a single-block structured mesh. This combination allows for the solution of a drag-minimizing pressure distribution using the Euler optimizer, and then, inclusion of viscous effects using the inverse design Navier-Stokes tool. Reuther et al [39] demonstrated lift-constrained, multipoint drag minimization based on multi-block structured analysis for the 3D Euler equations. Both of these studies are based on the so called variational sensitivity analysis approach and an explicit, multigrid time-marching scheme.

For the discretization of flowfields over complex domains, unstructured meshes offer significant advantages over alternative methods. They allow for the grids to be generated automatically, dramatically reducing turnaround times, and allow for adaptive grid procedures to be incorporated in a natural manner. For the solution of inviscid flows, unstructured mesh methods are now well developed, and work is currently in an advanced state of progress to incorporate viscous effects.

Newman et al [31] developed a capability to perform Euler optimization based on 2D unstructured grids and the discrete sensitivity analysis approach. An implicit approach was used which incorporated backward Euler timestepping and ILU preconditioned GMRES. Their lift-constrained drag minimization examples demonstrated the power of the method by finding a supercritical high-aft-loaded geometry at the optimum. They have since extended their capability to 3D unstructured grids [30]. The large storage and computational cost requirements associated with the implicit solution limited the usefulness of the approach in 3D. In a previous paper [14], the author also reported the implementation of an optimization system based on the discrete adjoint approach implemented on 3D unstructured meshes using a fully explicit scheme. The low memory requirements and the ensuing parallelization of the flow and sensitivity analysis solvers allowed practical optimization of complete aircraft configurations on rather fine meshes [16].

In terms of optimization based on viscous solvers, this has been demonstrated for the 2D turbulent incompressible [3] and the 2D compressible laminar [15] Navier-Stokes Equations. This thesis reports the extension of the capability reported in [15] to three dimensions. Viscous optimization based on the 2D Euler equations coupled with an integral boundary layer solver has been available for many years [12].

1.2 Inviscid Analysis

This thesis reports more fully on the development of the approach presented in [14] and [16] and the further development to incorporate a 3D lift-constrained, multipoint drag minimization capability based on the Euler equations discretized on unstructured grids. Solution of the problem using the particular features of the latter capability is considered important for the following reasons: (1) unstructured grids allow rapid automatic grid generation around complex geometries; (2) drag minimization rather than inverse pressure design relieves the aerodynamic designer from relying on experience and intuition in specifying the target pressure — although now experience and intuition are required to set the objective function and constraints; (3) incorporation of the lift constraint is critical since, in traditional design procedures, the weight of the aircraft is generally known long before the angle of attack; and (4) inclusion of off-design points in the optimization process is important for military aircraft where performance specifications are often made at more than one condition, but also for transonic commercial aircraft where single-point optimization often results in "point-designed" geometries with poor off-design characteristics.

Therefore, development of an unstructured drag-minimization capability based on the 3D Euler equations will allow efficient design of transonic and supersonic aircraft with wing-influencing engines or with other non–standard topologies, aircraft in high lift, multi-element mode and lateral-directional control configurations such as wings with deployed ailerons or spoilers. For this part, we neglect viscous effects — in the hope that the optimum design point is not significantly affected by their presence — and focus on the development of the modal inverse design and drag minimization capabilities based on the 3D Euler equations for unstructured grids. Specific focus is given to the means by which the constraints are incorporated for the drag minimization capability. The main conclusion of this part of the thesis is that a subspace BFGS optimization strategy provides a practical and robust optimization strategy. The 2D and 3D tests carried out resulted in credible optimal geometries based on inviscid flows, although as is pointed out in the next section, some of these tests tend to emphasize the need to include viscous effects.

1.3 Viscous Analysis

Although optimization based on the Euler equations is a useful capability for performing inverse pressure design of aerodynamic bodies especially for high Reynolds number flows where boundary layers are thin and the decambering associated with their displacement surfaces causes only small shock movements and lift changes, almost any drag minimization exercise based on the Euler equations and applied to modern supercritical wings in cruise condition is *certain to fail.* This is because the baseline wings have been designed to maximize L/D by pushing the upper surface shock as far aft as possible - at least over the highly-loaded outboard portion of the wing where the transonic effects are the most severe and where the wave drag is usually the highest. The beneficial lift improvement associated with this aft movement of the shock is counterbalanced by an increasingly severe adverse pressure gradient in the recovery region which typically, at shock positions of around $60\% - 70\%$, causes trailing edge separation and the associated detrimental effects of drag increase and lift loss. Many modern supercritical wings are therefore designed such that the boundary layer is on the verge of separation as it reaches the trailing edge. Hence it is clear that any drag minimization exercise applied to supercritical transport wings must include viscous effects.

Other flow regimes in which viscous effects play a large role are those associated with high lift takeoff and landing configurations. In these cases, without inclusion of viscous effects, accurate modelling of the outer flow is often not possible *even for fully attached flow.* Typical regimes feature boundary layers whose displacement thicknesses form a significant proportion of the gaps (between wing elements) through which they pass. Hence these boundary layers have a large impact on the beneficial lift improvement often found due to the interaction between elements. (This interelement interaction is heavily exploited by aerodynamic design engineers – but typically using unsatisfactory tools such as 2D viscous or 3D inviscid CFD tools, supported by copious expensive wind tunnel testing.) Furthermore local areas of separation are often present in these regimes, and indeed the maximum lift is often set by the onset of trailing edge separation on one or more elements. Further complications include off the surface slat wake flow reversal which can have a large impact on the lift achieved by landing configurations.

One effective way of including viscous effects is to solve a coupled viscous/inviscid system incorporating, for example, a viscous solver which solves the integral boundary layer equations [13]. This can result in a system which generates solutions more efficiently than Navier-Stokes solvers which typically require many points to resolve the boundary layer, resulting in both higher cost per iteration due to more mesh degrees of freedom and more iterations to convergence due to the requirement of smaller timesteps for stability, for both explicit and implicit time relaxation schemes. There are three factors which stack up against this option. Firstly, for separated flow, use of the direct coupling method (which most naturally fits into our current algorithm) results in an unstable scheme. Fully simultaneous [13] and semi-inverse schemes are possible alternatives which are stable but each would require significant amounts of work to incorporate into our current (explicit or point implicit) relaxation scheme. Secondly, while impressive work has been done on fully 3D integral boundary layer solvers, [32, 29], confidence in the aerospace industry in their results has not yet reached the levels found for their 2D counterparts and some theoretical issues remain to be resolved such as the imposition of hyperbolic boundary conditions. Application of the 2D results via strip theory may introduce a further source of error. Thirdly, use of displacement surfaces to implement the effect of the boundary layer on the inviscid flow leaves open the difficulty of applying this surface at geometric intersections such as wing/body intersections or wing/strut intersections.

On the other hand solving the Navier-Stokes equations throughout the flowfield is a more natural extension of the Euler optimization capability developed as the first part of the thesis research [14, 16] and discussed in the previous section. Also, this capability can serve as a benchmark for future work towards coupled viscous/inviscid solvers.

Although the algorithm presented herein has only been implemented for the 2D and 3D laminar Navier-Stokes equations so far, it represents a significant preliminary step towards the ultimate goal of high Reynold's number, fully turbulent 3D Navier-Stokes optimization. The features of the laminar algorithm allow direct extension to a practical turbulent algorithm. In particular, the residual Jacobian storage scheme implemented here leads to an efficient implementation of a 3D turbulent Navier-Stokes optimization algorithm.

1.4 Overview

The flow analysis algorithm for both inviscid and viscous simulation is described in Section 2.1. Descriptions are given of the optimization problem (Section 2.2), the sensitivity analysis calculation (Section 2.3), the time integration algorithm (Section 2.4), the parallelization of the flow and sensitivity analysis schemes (Section 2.5) and the constrained and unconstrained optimization algorithms (Section 2.6). These flow (Sections 3.1 and 3.2) and sensitivity (Section 3.3) analysis algorithms are validated in Chapter 3. Inviscid (Chapter 4) and viscous (Chapter 5) optimization examples are presented thereafter. Finally, some conclusions and recommendations for future work are also presented (Chapter 6).

Chapter 2

Algorithms

2.1 Flow Analysis

Two different flow analysis schemes are presented which were developed to allow efficient sensitivity calculation for inviscid and viscous flows, respectively. This was because in the course of extending the Euler optimization capability [14, 16, 17] to Navier-Stokes, it was found that switching to a spatial discretization scheme which can be described as mixed finite volume/finite element [34, 35] allowed a far more memory-efficient sensitivity calculation scheme than one based on the original scheme with viscous contributions. Since we also present 2D and 3D Euler results here and since we discuss the relative merits of either scheme vis-à-vis the sensitivity calculation, we find it appropriate to devote some space to discussion of the main components of both spatial discretization schemes.

The first scheme can be considered to be a finite volume scheme directly extended from the inviscid solver discussed in [14, 16]. In this case, the gradients required by both viscous and artificial dissipation fluxes are found exactly in each triangle/tetrahedron, based on the piecewise linear functions used to quantify the spatial variation of $\mathbf{U}$. This scheme will henceforth be referred to as "Scheme I". The second scheme can be considered to be a mixed finite element/finite volume scheme directly extended from the fundamental algorithm underlying FELISA [35], in which the average gradients over each nodal control volume are used for both the artificial dissipation and viscous fluxes. We will henceforth refer to this scheme as "Scheme II". The only difference between the two schemes is the way in which the gradients are calculated.

For illustration purposes, the algorithms are described based on the 2D algorithm. The 3D algorithm is a direct extension thereof. Both schemes begin with the integral form of the Navier-Stokes equations:-

$$\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t} d\Omega = \int_{\Omega} \frac{\partial}{\partial x_j}(\mathbf{G}^j - \mathbf{F}^j) d\Omega = \oint_{\Gamma} (\mathbf{G}^j - \mathbf{F}^j) n_j d\Gamma \tag{2.1}$$

where $\mathbf{U} = (\rho, \rho u_1, \rho u_2, \rho E)^T$,

$$\mathbf{F}^j = \begin{bmatrix} \rho u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ u_j(p + \rho E) \end{bmatrix}, \mathbf{G}^j = \begin{bmatrix} 0 \\ \tau_{1j} \\ \tau_{2j} \\ u_i \tau_{ij} + k\frac{\partial T}{\partial x_j} \end{bmatrix} \tag{2.2}$$

and Ω is a closed control volume with boundary, Γ. Here τ_{ij} is the tensor of viscous stresses and is given by

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3}\mu\delta_{ij}\frac{\partial u_k}{\partial x_k} \tag{2.3}$$

We first assume that $\mathbf{G}^j = 0$ and present the underlying inviscid discretization which is almost identical for both schemes. The discretization of $\mathbf{U}$ and $\mathbf{F}^j$ on an unstructured triangulation of the domain is accomplished using piecewise linear polynomials. The spatial discretization is completed by using the finite volume formulation with control volumes associated with each node i consisting of all triangles having vertex i. The line integration given by Equation (2.1) is performed exactly around the outer boundary of this control volume. (It is noted in passing that an identical discretization can be achieved by forming a Galerkin weighted residual statement using the same piecewise linear functions as the weight functions, and lumping the mass matrix entries onto the diagonal.) Using this spatial discretization, the following set of semi-discrete equations results:-

$$\Omega_i \frac{d\mathbf{U}_i}{dt} + \mathbf{R}_i(\mathbf{U}_i) = \mathbf{0} \tag{2.4}$$

where, through the use of an edge-based formulation [35, 26], the residual at node i can be written as

$$\mathbf{R}_i = \sum_{ik=1}^{K} [\mathbf{R}]_{ik}. \tag{2.5}$$

Here ik represents the edge connecting nodes i and k and the residual increment for an interior edge is given by,

$$[\mathbf{R}]_{ik} = -\frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_k}{2} \cdot \vec{S}_{ik} \tag{2.6}$$

where $\vec{S}_{ik}$ is the area vector associated with edge ik and is calculated as follows (see Figure 2-1):-

$$\vec{S}_{ik} = (y_{k+1} - y_{k-1}, x_{k-1} - x_{k+1}) \tag{2.7}$$

As shown in Equation (A.6) in Appendix A, the residual contribution for a wall boundary edge is

$$[\mathbf{R}_i^b]_{il} = -\{\frac{p_i + p_l}{2}(\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}) + \vec{\mathbf{F}}_i\} \cdot \vec{S}_b - \frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_l}{2} \cdot \vec{S}_{l,l+1} \tag{2.8}$$

where the second term ensures closure of the control volumes (see Figure 2-1) and,

$$\vec{S}_b = (y_l - y_i, x_i - x_l) \tag{2.9}$$

As may also be derived from Equation (A.6), for a farfield boundary edge the contribution to the residual becomes

$$[\mathbf{R}_i^b]_{il} = -\frac{\vec{\mathbf{F}}_i^b + \vec{\mathbf{F}}_l^b + 2\vec{\mathbf{F}}_i}{2} \cdot \vec{S}_b - \frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_l}{2} \cdot \vec{S}_{l,l+1} \tag{2.10}$$

where the superscript b indicates that the flux includes a 1st order diffusive contribution that effects upwinding using Roe-averaging, as described below, between the given node and the freestream state vector $\mathbf{U}_\infty$.

$$\vec{\mathbf{F}}_i^b \cdot \vec{S}_b = \frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_\infty}{2} \cdot \vec{S}_b - \frac{S_b}{2}|\mathbf{A}|(\mathbf{U}_i - \mathbf{U}_\infty)$$

where $\mathbf{A} = \partial(\vec{\mathbf{F}} \cdot \hat{n}_b)/\partial\mathbf{U}$ is the standard Roe matrix [41] resolved in the direction of the boundary normal $\hat{n}_b = \vec{S}_b/S_b$ and can be formed from the Roe-averaging of $\mathbf{U}_i$ and $\mathbf{U}_\infty$. These expressions for the fluxes at the boundaries ensure that correct boundary conditions are imposed and that we

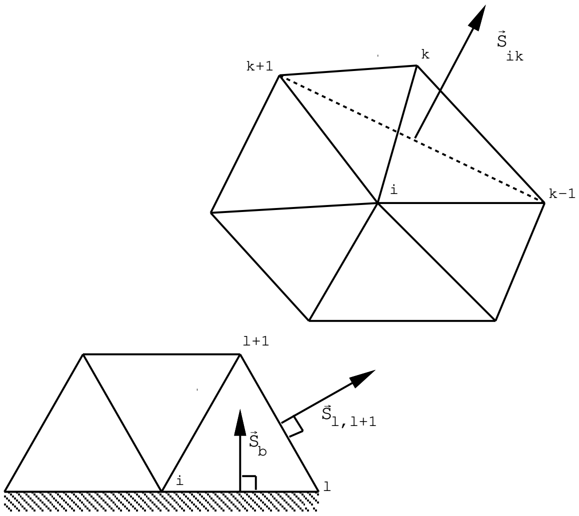

Figure 2-1: Area vectors for boundary and interior edges

have $\mathbf{R}(\mathbf{U}) = 0$ at steady state everywhere in the domain including the nodes on the boundary. This boundary treatment allows simple treatment of both adjoint and flow sensitivity boundary conditions for the sensitivity analysis as pointed out in the next section.

To stabilize the scheme, a dissipative term, premultiplied by a matrix coefficient, is added to the above equations for interior and farfield boundary fluxes.

$$\mathbf{D}_{ik} = \frac{S}{2}|\mathbf{A}(\mathbf{U}_i, \mathbf{U}_k)|(\mathbf{U}_i - \mathbf{U}_k) = \frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|\mathbf{P}^{-1}(\mathbf{U}_i - \mathbf{U}_k) \tag{2.11}$$

where $S = |\vec{S}_{ik}|$ and $\mathbf{A}$ is now resolved in direction of the face normal $\hat{n}_{ik} = \vec{S}_{ik}/S$ and is based on a Roe-averaging of $\mathbf{U}_i$ and $\mathbf{U}_k$. This ensures positivity of the coefficients multiplying increments of characteristic variables, $\Delta\mathbf{W}_{ik} = \mathbf{P}^{-1}\Delta\mathbf{U}_{ik}$ resulting in a scheme with local extremum diminishing (LED) [26] properties (see Appendix B). However, the added dissipation makes the scheme first order accurate and causes it to be too diffusive. To correct this, we add antidiffusion that is limited in a nonlinear manner to maintain the LED properties — similar to the antidiffusive contribution added to the one-dimensional scalar residual in Appendix B. Hence the diffusive flux takes the form:-

$$\mathbf{D}_{ik} = \frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|(\mathbf{P}^{-1}\Delta\mathbf{U}_{ik} - L(\mathbf{P}^{-1}\Delta\mathbf{U}^{-}, \mathbf{P}^{-1}\Delta\mathbf{U}^{+})) \tag{2.12}$$

Several options are available for the limiter function $L(u, v)$. Here we simply take

$$L(u,v) = minmod(\,u,\ v) \tag{2.13}$$

for inviscid calculations and

$$L(u,v) = minmod(\,2u,\;2v,\;\frac{u+v}{2}). \tag{2.14}$$

for viscous calculations. Note that the latter limiter is more likely to return the third argument, and by examination of Equation (2.11), results in a scheme with a desirable third order dissipative flux. The reason for using Equation (2.14) for the viscous calculations is that other researchers [21] have found that this limiter is less dissipative than Equation (2.13). For viscous calculations, drag estimates — especially close to separation — tend to be very sensitive to the amount of artificial dissipation present.

For Scheme I, $\Delta\mathbf{U}^-$, $\Delta\mathbf{U}^+$ are formed exactly from the gradients in the cells that either end of the edge points into [4, 26]. For example,

$$\Delta\mathbf{U}^+ = (\nabla\mathbf{U})^+ \cdot (\vec{x}_i - \vec{x}_k) = (\nabla\mathbf{U})^+ \cdot \vec{x}_{ik} = \epsilon_{ir}\Delta\mathbf{U}_{ir} + \epsilon_{is}\Delta\mathbf{U}_{is} \tag{2.15}$$

where the nodes in the element that the edge points into have been labelled *i,r,s* and $\epsilon_{ir}, \epsilon_{is}$ are, by construction, always positive. To see why this is so, we write the equation for the plane that passes through the three points, k, q and p with coordinates in (x,y,v) space given by (x_k, y_k, v_k), (x_q, y_q, v_q) and (x_p, y_p, v_p). Note that v is intended to represent the dependent variable which might be any one of the components of $\mathbf{U}$ and k, q and p represent the vertices of the element into which the edge points (see Figure 2-2). The equation for the plane is given by (see for example [48])

$$A(x - x_k) + B(y - y_k) + C(v - v_k) = 0 \tag{2.16}$$

where $\hat{n} = (A, B, C)$. Therefore the gradient is simply

$$\nabla v = -(\frac{A}{C}, \frac{B}{C}) \tag{2.17}$$

The normal — in (x,y,v) space — is given by

$$\hat{n} = \frac{\vec{x}_{kq} \times \vec{x}_{kp}}{|\vec{x}_{kq} \times \vec{x}_{kp}|} \tag{2.18}$$

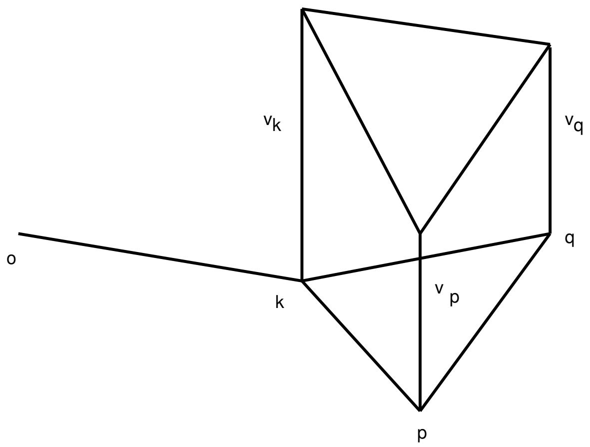

Figure 2-2: Gradient calculation method

and

$$
\begin{aligned}
\frac{\vec{x}_{kq} \times \vec{x}_{kp}}{|\vec{x}_{kq} \times \vec{x}_{kp}|} &= \frac{1}{|\vec{x}_{kq} \times \vec{x}_{kp}|} \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ x_k - x_q & y_k - y_q & v_k - v_q \\ x_k - x_p & y_k - y_p & v_k - v_p \end{vmatrix} \\
&= \frac{1}{|\vec{x}_{kq} \times \vec{x}_{kp}|} \left\{ \begin{array}{c} (y_k - y_q)\Delta v_{kp} - (y_k - y_p)\Delta v_{kq} \\ (x_k - x_p)\Delta v_{kq} - (x_k - x_p)\Delta v_{kp} \\ -2A_{kqp} \end{array} \right\} \\
=> \nabla v &= \frac{-1}{2A_{kqp}} \left\{ \begin{array}{c} (y_k - y_q)\Delta v_{kp} - (y_k - y_p)\Delta v_{kq} \\ (x_k - x_p)\Delta v_{kq} - (x_k - x_q)\Delta v_{kp} \end{array} \right\} \\
=> \nabla v \cdot \vec{x}_{ok} &= \frac{-1}{2A_{kqp}} \{\Delta v_{kp}((y_k - y_q)(x_k - x_o) - (x_k - x_q)(y_k - y_o)) + \\
&\quad \Delta v_{kq}(-(y_k - y_p)(x_k - x_o) + (x_k - x_p)(y_k - y_o))\} \\
&= \frac{1}{A_{kqp}}(-A_{kqo}\Delta v_{kp} + A_{kpo}\Delta v_{kq}). \qquad (2.19)
\end{aligned}
$$

Examination of the signs of these areas reveals that the coefficients must always be positive.

For Scheme II, $\Delta\mathbf{U}^-$, $\Delta\mathbf{U}^+$ are also calculated from the gradients, but here the gradients are calculated as averages over the control volumes. Applying the Divergence Theorem to $(\mathbf{U}, 0, 0)$, for example gives for the average,

$$\overline{\frac{d\mathbf{U}}{dx_1}} = \frac{1}{\Omega}\int_\Omega \nabla\cdot(\mathbf{U},0,0)d\Omega = \frac{1}{\Omega}\oint_\Gamma \mathbf{U}dS^1 \tag{2.20}$$

Evaluating this line integral in the same fashion as the flux integrals in Equation (2.1) results in the following formula which combines the interior and boundary edge contributions as given by Equations (A.4) and (A.6) :-

$$\frac{\partial\mathbf{U}_i}{\partial x_j} = \frac{1}{\Omega_i}\left(\sum_{ik=1}^{\mu_i}\frac{\mathbf{U}_i+\mathbf{U}_k}{2}S^j_{ik} + \sum_{il=1}^{n_{bou}}\left\{\frac{3\mathbf{U}_i+\mathbf{U}_l}{2}S^l_{il} + \frac{\mathbf{U}_i+\mathbf{U}_l}{2}S^j_{l,l+1}\right\}\right) \tag{2.21}$$

where μ_i is the number of nodes neighbouring node i, n_{bou} is the number of boundary edges contributing to the residual at node i and subscript ik represents the edge connecting nodes i and k. The rigorous LED character of the scheme is lost but Scheme II tends to produce smoother and less dissipative solutions than Scheme I for transonic and low supersonic flows.

Note that in both cases, the addition of antidiffusion causes the scheme to now resemble a high order MUSCL scheme. For further details of similar inviscid algorithms, the reader is referred to [35, 34].

For both schemes (although the only viscous results presented herein are for Scheme II), the means for inclusion of viscous terms recycles the nodal gradients used for the dissipative fluxes. For Scheme II, the edge-based data structure can be retained, and $\mathbf{F}^j_i$, for example, in Equation (2.6) is simply replaced by $\mathbf{F}^j_i - \mathbf{G}^j_i$. For Scheme I, on the other hand, inclusion of viscous effects demands that we return to an element-based data structure, whether the viscous terms are included via the Finite Volume or the Finite Element method [36]. This has a detrimental impact on the memory requirements for both flow analysis and sensitivity analysis (as pointed out in the Section 2.3.3).

2.2 Optimization Problem Statement

We consider the general optimization problem of finding an optimal design variable vector $\boldsymbol{\beta}^{opt}$, of dimension N, that minimizes a given cost functional $\mathcal{F}$, that is,

$$\boldsymbol{\beta}^{opt} = \arg\min_{\boldsymbol{\beta}} \mathcal{F}(\boldsymbol{\beta}) \tag{2.22}$$

subject to M constraints of the form

$$c_j(\boldsymbol{\beta}) = 0, \qquad j = 1 \ldots M \tag{2.23}$$

In our case, the cost function is either a weighted sum of the drag coefficients at M flow conditions,

$$\mathcal{F} = \sum_{j=1}^{M} \omega_j C_{D_j}(\boldsymbol{\beta}) \tag{2.24}$$

or the area-integrated sum of squared differences of the actual pressure from a target pressure distribution over the target surface, as given by Equation (D.1). The N components of the design vector $\boldsymbol{\beta}$ are the parameters used to represent the aircraft geometry and, for some cases, the M angles of attack corresponding to each flow condition. ($M > 1$ for multipoint cases.) The constraints considered (in some of the problems) are the lift at each operating point. Thus,

$$c_j(\boldsymbol{\beta}) \equiv C_{L_j}(\boldsymbol{\beta}) - C_{L_j}^{targ} = 0, \qquad j = 1 \ldots M. \tag{2.25}$$

The weights ω_j, associated with each operating point, are given and kept constant throughout the optimization process. The above problem can be easily generalized to incorporate additional constraints, including inequality constraints, without substantially changing the approach to be described below.

The solution of problem (2.22) is attempted using iterative gradient based techniques. We therefore require a feasible initial guess for the parameter vector which corresponds to a baseline geometry and the angles of attack that satisfy the constraints (2.25). The problem is then decomposed into the evaluation of the gradients, or sensitivities, and the updating of the design variables using a suitable descent strategy.

In practice, the drag coefficient at each operating point j, is calculated from a computed numerical solution of the Euler or Navier-Stokes equations. One obvious possibility is to approximate the sensitivities using finite differences which require the computation of flowfields for different values of the design parameters. Whilst this approach may be feasible when the number of design variables is small, it becomes impractical when the dimension of $\boldsymbol{\beta}$, N, is large.

A more convenient approach which allows for exact sensitivities to be calculated is to consider the numerical flow solution as additional design variables which are *constrained* to satisfy the Euler equations at each operating point. If we denote by $\mathbf{U}_j$ the solution that satisfies the Euler equations at the flow condition j, for a given $\boldsymbol{\beta}$, we can write

$$\mathbf{R}_j(\boldsymbol{\beta}, \mathbf{U}_j) = 0, \qquad j = 1 \ldots M. \tag{2.26}$$

This equation *implicitly* defines $\mathbf{U}_j(\boldsymbol{\beta})$ which in turn can be used, in a straightforward manner, to compute C_{D_j} and thus $\mathcal{F}$. Therefore, we can write the following modified but equivalent problem: find $\boldsymbol{\beta}^{opt}$ such that

$$\boldsymbol{\beta}^{opt} = \arg\min_{\boldsymbol{\beta}} \mathcal{F}(\boldsymbol{\beta}, \mathbf{U}_1(\boldsymbol{\beta}), \mathbf{U}_2(\boldsymbol{\beta}), \ldots \mathbf{U}_M(\boldsymbol{\beta})) \tag{2.27}$$

subject to the M original constraints of equation (2.23) plus the M additional constraints of equation (2.26).

2.3 Sensitivity Analysis

Consider, for the purposes of simplicity of notation, that the design variable $\boldsymbol{\beta}$ contains only one component and so can be written as a scalar β. Furthermore, assume that for each condition j, the drag coefficient is given by

$$C_{D_j} = C_{D_j}(\mathbf{U}_j(\beta), \beta). \tag{2.28}$$

Differentiating the steady state version of Equation (2.4) and Equation (2.28) produces, respectively,

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \beta} = -\frac{\partial \mathbf{R}}{\partial \beta} \tag{2.29}$$

and

$$\frac{dC_{D_j}}{d\beta} = \frac{\partial C_{D_j}}{\partial \beta} + \frac{\partial C_{D_j}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \beta} \tag{2.30}$$

The first term on the right-hand side of (2.30) can be calculated relatively easily. The second term is far more difficult. Two approaches are the direct and the adjoint method.

The direct method involves *direct* solution of Equation (2.29) and substitution of the resulting vector $\partial \mathbf{U}/\partial \beta$ into Equation (2.30). The means by which Equation (2.29) is solved is discussed below. Note that Equation (2.29) must be solved once for each component of $\boldsymbol{\beta}$.

The adjoint method is based on the recognition that Equations (2.29) and (2.30) can be combined to give:

$$\left.\frac{dC_{D_j}}{d\beta}\right|_{\mathbf{R}=0} = \frac{\partial C_{D_j}}{\partial \beta} - \left[\frac{\partial C_{D_j}}{\partial \mathbf{U}}\right]^T \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1} \left[\frac{\partial \mathbf{R}}{\partial \beta}\right] \tag{2.31}$$

Due to associativity of matrix multiplication, the right-hand-side double product calculation may be performed either by first multiplying the Jacobian inverse by the term to its right – which gives the *direct* method – or by the term to its left – which gives the *adjoint* method. It is convenient to write the result of the intermediate calculation for the adjoint method in terms of the adjoint variable $\boldsymbol{\psi}$ which satisfies

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \boldsymbol{\psi} = -\left\{\frac{\partial C_{D_j}}{\partial \mathbf{U}}\right\}^T \tag{2.32}$$

and

$$\left.\frac{dC_{D_j}}{d\beta}\right|_{\mathbf{R}=0} = \frac{\partial C_{D_j}}{\partial \beta} + \boldsymbol{\psi}^T \left\{\frac{d\mathbf{R}}{d\beta}\right\}. \tag{2.33}$$

It is noted that equations (2.26) and (2.32) correspond to the stationarity conditions of a Lagrangian formed by augmenting the cost function with the constraint terms

$$\mathcal{L}(\boldsymbol{\beta}, \mathbf{U}_1, \ldots, \mathbf{U}_M, \boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_2) = \mathcal{F}(\boldsymbol{\beta}, \mathbf{U}_1, \ldots, \mathbf{U}_M) + \sum_{j=0}^{M} \boldsymbol{\psi}_j^T \mathbf{R}_j(\mathbf{U}, \boldsymbol{\beta}). \tag{2.34}$$

With this interpretation, the intermediate variables $\boldsymbol{\psi}_j$ play the role of Lagrange multipliers [23]. To see this, we write the optimality conditions, which are found by setting to zero the first variations of $\mathcal{L}$ with respect to $\boldsymbol{\psi}$,

$$\delta\mathcal{L}_{\boldsymbol{\psi}} = \mathcal{L}(\boldsymbol{\beta}, \mathbf{U}, \boldsymbol{\psi} + \delta\boldsymbol{\psi}) - \mathcal{L}(\boldsymbol{\beta}, \mathbf{U}, \boldsymbol{\psi}) = \delta\boldsymbol{\psi}^T \mathbf{R}(\mathbf{U}, \boldsymbol{\beta}) = 0 \tag{2.35}$$

resulting in the steady state version of Equation (2.4); and with respect to $\mathbf{U}$,

$$\delta\mathcal{L}_{\mathbf{U}} = \mathcal{L}(\boldsymbol{\beta}, \mathbf{U} + \delta\mathbf{U}, \boldsymbol{\psi}) - \mathcal{L}(\boldsymbol{\beta}, \mathbf{U}, \boldsymbol{\psi}) = (\boldsymbol{\psi}^T \frac{\partial \mathbf{R}_j}{\partial \mathbf{U}} + \frac{\partial C_D}{\partial \mathbf{U}})\delta\mathbf{U} = 0. \tag{2.36}$$

resulting in Equation (2.32). The third optimality condition, requiring the variations of $\mathcal{L}$ with respect to $\boldsymbol{\beta}$ to be zero, leads to

$$\frac{d\mathcal{L}}{d\boldsymbol{\beta}} = \sum_{j=0}^{M} \omega_j \frac{\partial C_{D_j}}{\partial \boldsymbol{\beta}} + \sum_{j=0}^{M} \omega_j \boldsymbol{\psi}_j^T \left\{\frac{d\mathbf{R}_j}{d\boldsymbol{\beta}}\right\} = 0. \tag{2.37}$$

The matrices $\partial\mathbf{R}/\partial\mathbf{U}$ and $\partial\mathbf{R}/\partial\mathbf{X}$ (the residual sensitivities to grid nodal coordinates, which are needed in the calculation of $\partial\mathbf{R}/\partial\boldsymbol{\beta}$), are constructed in exactly the same fashion as $\mathbf{R}(\mathbf{U})$, namely by looping over the edges of the triangulation. This allows for debugging and comparison with finite difference quantities at the edge level. The corresponding equation to Equation (2.5) for sensitivities is

$$\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}\mathbf{U}_{\beta_j} = \sum_{ik=1}^{K}[\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}]_{ik}\mathbf{U}_{\beta_j} \tag{2.38}$$

where, for example, for an interior edge,

$$\begin{aligned}[\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}]_{ik}\mathbf{U}_{\beta_j} \quad = \quad & [-\frac{\delta_{kj}}{2}\frac{\partial\vec{\mathbf{F}}_k}{\partial\mathbf{U}}\cdot\vec{S}_{ik} + \frac{\partial}{\partial\mathbf{U}_j}(S\mathbf{P}|\mathbf{\Lambda}|)(\Delta\mathbf{W}_{ik} - L(\Delta\mathbf{W}^-, \Delta\mathbf{W}^+)) + \\ & (S\mathbf{P}|\mathbf{\Lambda}|)\frac{\partial}{\partial\mathbf{U}_j}(\Delta\mathbf{W}_{ik} - L(\Delta\mathbf{W}^-, \Delta\mathbf{W})^+))]\mathbf{U}_{\beta_j}\end{aligned}$$

For the adjoint problem the transposed Jacobian is required.

$$(\frac{\partial\mathbf{R}_j}{\partial\mathbf{U}_i})^T\boldsymbol{\psi}_i = \sum_{ik=1}^{K}[\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}]_{ik}\boldsymbol{\psi}_i \tag{2.39}$$

The chain rule is applied to the forcing term derivatives. Thus, we have:-

$$\frac{\partial\mathbf{R}}{\partial\boldsymbol{\beta}} = \frac{\partial\mathbf{R}}{\partial\mathbf{X}}\frac{d\mathbf{X}}{d\boldsymbol{\beta}} \tag{2.40}$$

and $d\mathbf{X}/d\boldsymbol{\beta}$, the variation of the nodal coordinates, $\mathbf{X}$, with respect to the design variables, $\boldsymbol{\beta}$, is also calculated by looping over the edges. The grid movement algorithm and the calculation of the grid movement sensitivities are discussed in the next subsection.

2.3.1 Sensitivity and Adjoint Boundary Conditions

Implicit inclusion of zero-normal velocity boundary conditions in the residual definitions for the Euler equations, as described in the Section 2.1, guarantees that the implicit function theorem — used in deriving Equation (2.29) holds automatically and that correct sensitivities will be calculated. Taking this approach rather than carrying out an à posteriori sweep to zero the nodal normal velocities allows a straightforward boundary treatment and avoids complications associated with differentiating the velocity projection process. For Navier-Stokes simulations, the à posteriori no-slip condition enforcement process cannot be avoided but a similar treatment of boundary adjoint and flow sensitivity variables is far more straightforward as described in the next paragraph.

The effect of the no-slip boundary conditions is to eliminate the momentum equation residuals at the wall from the system to be solved. Therefore the adjoint variables (which are in essence Lagrange multipliers for each nodal residual equation[17, 23]) corresponding to the momentum equations at the wall should have no impact on the final adjoint solution elsewhere [3]. In the current implementation, this has been by performing an à posteriori sweep on the wall boundary nodes in which the momentum adjoint variables are zeroed out.

2.3.2 Mesh Movement

Grid motion is accomplished using Jacobi-like iteration based on the force equilibrium of springs one of which lies along each edge of the triangulation. The relaxation scheme is

$$\vec{\boldsymbol{\delta}}_i^{\,n+1} = \frac{\sum k_{ij}\vec{\boldsymbol{\delta}}_j^{\,n}}{\sum k_{ij}} \tag{2.41}$$

where $k_{ij} = |\Delta\vec{x}_{ij}|^{-1}$ and

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \boldsymbol{\delta}^{n+1} \tag{2.42}$$

and $\boldsymbol{\delta}$ is the vector of nodal displacements. Therefore the grid sensitivity is simply

$$\frac{d\mathbf{X}}{d\boldsymbol{\beta}} = \frac{\partial\boldsymbol{\delta}^N}{\partial\boldsymbol{\beta}} = \frac{\partial\boldsymbol{\delta}^N}{\partial\boldsymbol{\delta}^{N-1}}\frac{\partial\boldsymbol{\delta}^{N-1}}{\partial\boldsymbol{\delta}^{N-2}}\cdots\frac{\partial\boldsymbol{\delta}^1}{\partial\boldsymbol{\delta}^0}\frac{\partial\boldsymbol{\delta}^0}{\partial\boldsymbol{\beta}} \tag{2.43}$$

where we have performed N Jacobi iterations of the grid relaxation scheme. The coefficients of the matrices, $\partial\boldsymbol{\delta}^i/\partial\boldsymbol{\delta}^{i-1}$ in the above product are found from Equation 2.41 while $\partial\boldsymbol{\delta}^0/\partial\boldsymbol{\beta}$ is determined from the expression for the wing surface design variable perturbation.

For the Navier-Stokes calculations, we reverted to a scheme in which the grid movement at a location x_j is simply given by the sum of the airfoil surface modal perturbations corresponding to that value of x_j. This eliminates the potential problem of negative volume elements in the perturbed grid in the boundary layer due to the small mesh spacing in the normal direction. This will obviously have to be dealt with in a more satisfactory manner when extending to more complex configurations. Note that, in this case, there is a mechanism for relaxation of the grid perturbation to zero at the farfield boundary.

2.3.3 Implementation Issues: Jacobian storage

One of the main obstacles to sensitivity calculation based on the Euler and Navier-Stokes equations using the direct and adjoint approaches is that associated with the large memory cost of storing the residual Jacobian matrix, $\partial\mathbf{R}/\partial\mathbf{U}$. The matrix is usually sparse due to compact support of the shape functions used commonly for both the finite volume or finite element residual statements. This results in only the immediate neighbours having an influence on $\mathbf{R}_i$, the residual at node i (an exception would be a spectral scheme). Dissipative fluxes required for stability and shock capturing can extend the stencil to include the neighbours of the neighbours depending on the formulation used. Nonetheless, it is obviously very expensive to store the whole sparse matrix, $\partial\mathbf{R}/\partial\mathbf{U}$.

One obvious improvement is to store the Jacobian entries on an edge basis. This results in the storage costs scaling as $O(N)$ rather than $O(N^2)$ for the full matrix. However the costs are still large and prohibitive for 3D cases unless a parallel architecture with distributed memory is used.

Several approaches have been used in the past to circumvent these memory cost problems. In some cases [6], schemes have been limited to first order accuracy in order that $\partial\mathbf{R}/\partial\mathbf{U}$ may be more easily stored due to the smaller stencil. An alternative option is to resort to schemes in which the entries in $\partial\mathbf{R}/\partial\mathbf{U}$ are recalculated "on-the-fly" [30, 14]. The authors found that although this obviously provides enormous memory savings, it results in large CPU cost increases [17] — about a factor of four increase for our 2D and 3D Euler schemes.

Use of the continuous sensitivity analysis approach [25, 38] avoids these problems since the relaxation scheme used for solving the flow analysis system can be recycled to solve the adjoint problem. Stability can be provided either by reverse biasing of the difference operators due to the reversed direction of the zone of dependence for the adjoint problem or simply by addition of a dissipative flux of the same form as that for the flow analysis scheme. Of course, one of the drawbacks of the continuous sensitivity analysis approach is that calculated sensitivities cannot be exact except at the limit of an infinitely fine grid.

The extension to Navier-Stokes introduces further complications. Use of Scheme I [14, 16] results in a very small stencil for the Euler equations causing low memory costs for $\partial\mathbf{R}/\partial\mathbf{U}$. However, when viscous stresses and heat fluxes are included when extending to the Navier-Stokes equations, Scheme I loses its small stencil advantage. (This is because exact integration of, for example, the viscous fluxes results in dependencies on the 3rd (and 4th in 3D) nodes in the 2 elements containing a given edge.) Switching to an element-based data structure because of this, causes an increase in memory for both flow analysis and a large increase in the size of $\partial\mathbf{R}/\partial\mathbf{U}$ for sensitivity analysis.

Reverting to the control-volume-average gradients in Scheme II and implementing the Jacobian storage scheme described for Scheme I without adaptation results in a far higher memory cost especially in 3D. This is because the residual contribution for a given edge is dependent on the nodal gradients at either end of the edge, and the gradient at node i is in turn dependent on its immediately adjacent neighbours as shown in Equation (2.21). Therefore the residual contribution for edge ik can be expressed as:

$$[\mathbf{R}]_{ik} = [\mathbf{R}]_{ik}(\mathbf{U}_i, \mathbf{U}_k, \mathbf{U}_{i1}, \cdots, \mathbf{U}_{i\mu_i}, \mathbf{U}_{k1}, \cdots, \mathbf{U}_{k\mu_k}) \tag{2.44}$$

$[\mathbf{R}_{ik}]$ can be dependent on as many as $(2 \times 10 + 2)$ nodes in 2D and $(2 \times 40 + 2)$ nodes in 3D. Direct calculation of each dependency obviously results in a prohibitive memory cost for $\partial\mathbf{R}/\partial\mathbf{U}$.

This problem can be avoided by calculating $\partial\mathbf{R}/\partial\mathbf{U}$ in two stages. It can be seen from Equations (2.2), (2.6) and (2.11) that the residual contribution for a given edge can be written as

$$\mathbf{R}(\mathbf{U}) = \mathbf{R}(\mathbf{U}, \mathbf{U}_{x_i}; \mathbf{U}) \tag{2.45}$$

Following previous researchers, e.g.[5], we apply the chain rule and obtain,

$$\begin{aligned} \frac{d\mathbf{R}}{d\mathbf{U}} &= \frac{\partial\mathbf{R}}{\partial\mathbf{U}}_I + \frac{\partial\mathbf{R}}{\partial\mathbf{U}}_{II} \\ &= \frac{\partial\mathbf{R}}{\partial\mathbf{U}}_I + \frac{\partial\mathbf{R}}{\partial\mathbf{U}_{x_j}}\frac{\partial\mathbf{U}_{x_j}}{\partial\mathbf{U}} \end{aligned} \tag{2.46}$$

where we have used the shorthand $\mathbf{U}_{x_j} = \partial\mathbf{U}/\partial x_j$. In addition further savings can be realized by eliminating redundancies in the otherwise, quite large $\partial\mathbf{U}_{x_j}/\partial\mathbf{U}$ matrix. It is found after examining the expression used to calculate $\mathbf{U}_{x_j}$, Equation (2.21), that each conservative variable component is only dependent on neighbouring nodal values of that same conservative variable component. Therefore $\partial\mathbf{U}_{x_j}/\partial\mathbf{U}$ is only nonzero if the differential with respect to the same conservative variable components is sought. Also $\partial\mathbf{U}_{x_j}/\partial\mathbf{U}$ is the same for all conservative variable components. Therefore $\partial\mathbf{U}_{x_j}/\partial\mathbf{U}$ is quite small, and $\partial\mathbf{R}/\partial\mathbf{U}$ can be stored using about the same amount of memory as was required to store the Euler residual jacobian based on the Scheme I discretization.

In practice, a small connectivity array additional to the normal "side" array is needed. This represents the indices surrounding a given node. The entries of the matrix, $\partial\mathbf{U}_{x_j}/\partial\mathbf{U}$, are then accessed according to the ordering of the "surrounding nodes" array. Care is required in performing the chain rule multiplication in order to avoid a prohibitively slow calculation. For example, an inefficient implementation of the multiplication representing the contribution via the gradient — subscript II in Equation (2.46) — to the update of the flow variable sensitivity, would be as follows,

$$RHS_{i,iv} = \frac{\partial R_{i,iv}}{\partial G_{j,jv,jg}} (\frac{\partial G_{j,jv,jg}}{\partial U_{k,jv}} \frac{\partial U_{k,jv}}{\partial \beta}) \tag{2.47}$$

in which the whole calculation takes place within a loop over edges and the parenthesized multiplication is performed using a nested loop over the surrounding points within this loop over edges. (In Equation (2.47), subscripts i, j and k represent nodes, subscripts iv and jv represent components of the state vector $\mathbf{U}$ at points i and j, respectively and jg represents a component of the vector in one of the directions x_1, x_2 or x_3.) However, the calculation can be performed much more efficiently if the parenthesized multiplication is performed as a loop over points with a nested loop over surrounding points

$$\frac{\partial G_{j,jv,jg}}{\partial \beta} = (\frac{\partial G_{j,jv,jg}}{\partial U_{k,jv}} \frac{\partial U_{k,jv}}{\partial \beta}) \tag{2.48}$$

before the final result is obtained using a loop over edges

$$RHS_{i,iv} = \frac{\partial R_{i,iv}}{\partial G_{j,jv,jg}} \frac{\partial G_{j,jv,jg}}{\partial \beta} \tag{2.49}$$

The calculation can be performed in a similar manner for the adjoint method. In this case the following product is required to the find the update to the adjoint variable:-

$$RHS_{k,jv} = \left((\frac{\partial R_{i,iv}}{\partial U_{k,jv}} \right)_{II} \psi_{i,iv} \tag{2.50}$$

$$= \left(\frac{\partial R_{i,iv}}{\partial G_{j,jv,jg}} \frac{\partial G_{j,jv,jg}}{\partial U_{k,jv}} \right) \psi_{i,iv} \tag{2.51}$$

$$= \left(\psi_{i,iv} \frac{\partial R_{i,iv}}{\partial G_{j,jv,jg}} \right) \frac{\partial G_{j,jv,jg}}{\partial U_{k,jv}} \tag{2.52}$$

Here, the order of looping over edges and looping over points must be reversed. First, an intermediate variable was calculated by looping over edges:-

$$q_{j,jv,jg} = \frac{\partial R_{i,iv}}{\partial G_{j,jv,jg}} \psi_{i,iv} \tag{2.53}$$

Then the calculation is completed by looping over points with a nested loop over the surrounding points:-

$$RHS_{k,jv} = \frac{\partial G_{j,jv,jg}}{\partial U_{k,jv}} q_{j,jv,jg} \tag{2.54}$$

2.4 Time Integration

2.4.1 Flow Analysis

In the current research we have used a four-stage explicit time-stepping scheme for the Euler calculations. Specifically, the form of the explicit method is as follows.

$$\begin{aligned}
\mathbf{U}_i^{(1)} &= \mathbf{U}_i^n \\
\mathbf{U}_i^{(2)} &= \mathbf{U}_i^n + \alpha_1 \tfrac{\Delta t_i}{\Omega_i} \mathbf{R}_i^{(1)} \\
\mathbf{U}_i^{(3)} &= \mathbf{U}_i^n + \alpha_2 \tfrac{\Delta t_i}{\Omega_i} \mathbf{R}_i^{(2)} \\
\mathbf{U}_i^{(4)} &= \mathbf{U}_i^n + \alpha_3 \tfrac{\Delta t_i}{\Omega_i} \mathbf{R}_i^{(3)} \\
\mathbf{U}_i^{n+1} &= \mathbf{U}_i^n + \alpha_4 \tfrac{\Delta t_i}{\Omega_i} \mathbf{R}_i^{(4)}
\end{aligned} \tag{2.55}$$

where $\mathbf{R}^{(k)} = \mathbf{R}(\mathbf{U}^{(k)})$ is implied and $(\alpha_1, \alpha_2, \alpha_3, \alpha_4) = (0.25, 0.333, 0.5, 1.0)$.

For the Navier-Stokes calculations it was found that an implicit scheme was needed due to the small sizes of the elements in the boundary layer in the normal direction and the resulting slow convergence rates. For preliminary studies a point implicit time-stepping algorithm scheme was used. It was found to allow at least a doubling of the asymptotic convergence rate for one fourth of the per-iteration CPU cost of the multistage scheme for the adjoint calculation.

The point implicit algorithm follows the ideas elucidated in [20, 21], but differs in that the exact block diagonal entries of the $\partial\mathbf{R}/\partial\mathbf{U}$ matrix are used on the left hand side as opposed to only the first order terms. In brief, the relaxation scheme is

$$\mathbf{L}_i \Delta\mathbf{U}_i = (\Omega_i + \Delta t_i \left.\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right|_{ii}) \Delta\mathbf{U}_i = \Delta t_i \mathbf{R}_i \tag{2.56}$$

where

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n + \Delta\mathbf{U}_i. \tag{2.57}$$

Finally, a viscous correction was made to the Δt_i in an effort to ensure that the viscous time step limit is not exceeded, thereby compromising stability. This correction is given by

$$\Delta t_i = \frac{\Delta t_i^{inv}}{1 + 4/Re_{\Delta s_i}} \tag{2.58}$$

where Δt_i^{inv} is the time step limit found using linearized Fourier analysis for the inviscid scheme [18] and $Re_{\Delta s_i}$ is the Reynolds number based on the length of the smallest edge in the elements surrounding node i. It has been found by several researchers that this stability limit can have a large influence for well-resolved, low Reynolds number flows [46, 1, 9].

2.4.2 Sensitivity Analysis

It is possible to adopt exactly the same relaxation algorithm to solve equations (2.4), (2.29) and (2.32). This is done by introducing an artificial unsteady term into, for example, the adjoint equation resulting in the following semi-discrete equation.

$$\Omega_i \frac{d\boldsymbol{\psi}_i}{dt} + \left[\frac{\partial \mathbf{R}_i}{\partial \mathbf{U}_j}\right]^T \boldsymbol{\psi}_j - \left[\frac{\partial C_{D_k}}{\partial \mathbf{U}_i}\right]^T = 0 \tag{2.59}$$

These schemes possess the same stability properties because the spectral radii corresponding to the direct and adjoint sensitivity analysis schemes are identical to that of the linearized flow analysis scheme in its semidiscrete form

$$\frac{d\Delta \mathbf{U}_i}{dt} \approx -(\frac{1}{\Omega_i})\frac{\partial \mathbf{R}_i}{\partial \mathbf{U}_j}\Delta \mathbf{U}_j^n \tag{2.60}$$

which governs its asympotic convergence behaviour. Here, $\Delta\mathbf{U}_i$ represents the error from the steady state solution. The spectral radius of the errormode amplification matrix implied by Equation (2.59) is the same as that for the linearized flow analysis since transposing a matrix does not change the modulus of its eigenvalues. Indeed for all calculations performed using Scheme I, the asymptotic convergence rate of analysis, direct and adjoint schemes was found to be very close. A sample set of convergence plots based on Scheme I is shown in Figure 2-3.

To extend the point implicit timestepping scheme to the adjoint system, it is necessary to replace $\mathbf{L}$ with $\mathbf{L}^T$, since this ensures that the eigenvalues of the error amplification matrix remain the same as for linearized flow analysis. As can be seen in Figure 2-4 which shows the convergence histories for flow, sensitivity and two adjoint analyses for flow over a NACA0012, identical asymptotic convergence behaviour was found using this approach.

2.4.3 LED CFL-like Condition

It was found that use of a linearized Fourier stability analysis to determine the allowable timestep [18], allows convergence to machine accuracy only for very low CFL numbers for 3D inviscid problems using Scheme I. For practical CFL levels, limit-cycling was observed. It is found that for flow analysis, the solutions produced are acceptable when this limit-cycling occurs at residual levels that are low. However, the sensitivity analyses performed for baseline state vectors $\mathbf{U}$ whose solution convergence histories exhibited this limit-cycling behaviour can be unstable at the CFL numbers that the analyses were run at. For example, at $\sigma = 0.5$, limit cycling was typically observed for flow analysis and the sensitivity analysis was unstable, while for $\sigma = 0.05$, convergence to machine accuracy was typically found for both flow and sensitivity analyses although this was very slow.

The problem was solved by supplementing the local timestep calculation with a check on violations of the LED CFL-like condition. For a scalar equation, TVD conditions for the semidiscrete form of a scheme that can be expressed as

$$\frac{du_o}{dt} = \sum_{i=1}^{K} c_{oi}(u_i - u_o) \tag{2.61}$$

are

$$c_{oi} > 0 \quad i = 1, \cdots, K \tag{2.62}$$

where K represents the total number of nodes that contribute to the residual at node i. When the Forward Euler time discretization is performed, the fully discrete scheme is

$$u_o^{n+1} = u_o^n + \Delta t \sum_{i=1}^{K} c_{oi}(u_i - u_o) = \sum_{i=1}^{K} d_{oi} u_i + \mu_o u_o \tag{2.63}$$

In this case, the TVD conditions become

$$d_{oj} = \Delta t c_{oj} > 0 \quad j = 1, \cdots, K \tag{2.64}$$

$$\mu_o = 1 - \Delta t \sum_{i=1}^{K} c_{oi} > 0 \tag{2.65}$$

The latter is a multidimensional LED CFL-like condition.

For the discretization of the Euler equations using Scheme I (described in Section 2.1), with time integration algorithm given by Equation (2.55), a similar approach is used. Since $\sum_k \vec{S}_{ik} = (0, 0)$, the residual, which includes the "physical" contribution — given by Equation (2.6) — and the dissipation term — given by Equation (2.12) — can be rewritten as

$$
\begin{aligned}
\mathbf{R}_i &= -\sum_{ik=1}^{K}\{\frac{\vec{\mathbf{F}}_k-\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_{ik}-\frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|(\mathbf{P}^{-1}\Delta\mathbf{U}_{ik}-L(\mathbf{P}^{-1}\Delta\mathbf{U}^-,\mathbf{P}^{-1}\Delta\mathbf{U}^+))\} \\
&= -\sum_{ik=1}^{K}\{\frac{S\mathbf{A}(\mathbf{U}_k-\mathbf{U}_o)}{2}-\frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|(\mathbf{P}^{-1}\Delta\mathbf{U}_{ik}-L(\mathbf{P}^{-1}\Delta\mathbf{U}^-,\mathbf{P}^{-1}\Delta\mathbf{U}^+))\} \\
&= -\sum_{ik=1}^{K}\{\frac{S\mathbf{P}(\mathbf{\Lambda}-|\mathbf{\Lambda}|)\mathbf{P}^{-1}(\mathbf{U}_k-\mathbf{U}_o)}{2}+\frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|(L(\mathbf{P}^{-1}\Delta\mathbf{U}^-,\mathbf{P}^{-1}\Delta\mathbf{U}^+))\} \qquad (2.66)
\end{aligned}
$$

If this represents the residual at a typical stage in the Runge-Kutta time integration scheme —- Equation (2.55) — then premultiplying both sides of the expression for the change in state vector, $\Delta\mathbf{U}$, for that stage by $\mathbf{P}^{-1}$ to get $\mathbf{W}^{(n+1)}-\mathbf{W}^{(n)}\approx\mathbf{P}^{-1}(\mathbf{U}^{(n+1)}-\mathbf{U}^{(n)})$, the stage change in the characteristic variable vector:-

$$
\begin{aligned}
\mathbf{P}^{-1}(\mathbf{U}^{(n+1)}-\mathbf{U}^{(n)}) &\approx \mathbf{W}^{(n+1)}-\mathbf{W}^{(n)} \\
&\approx -\alpha_{i_{stage}}\sigma\frac{\Delta t_i}{\Omega_i}\sum_{ik=1}^{K}\{\frac{S(\mathbf{\Lambda}-|\mathbf{\Lambda}|)\Delta\mathbf{W}_{ko}}{2}+\frac{S}{2}|\mathbf{\Lambda}|(L(\Delta\mathbf{W}^-,\Delta\mathbf{W}^+))\} \\
&\approx \Delta t_i(\sum_{ok=1}^{K}(C_{ko}^{int}\Delta\mathbf{W}_{ko}^{int}+C_{ko}^{upw}\Delta\mathbf{W}_{ko}^{upw}+C_{ko}^{ad}\Delta\mathbf{W}_{ko}^{ad})+ \qquad (2.67) \\
&\quad \sum_{kop}^{bou}C_{kip}^{bou}\Delta\mathbf{W}_{ko}^{bou})
\end{aligned}
$$

where the four coefficients in the inner parenthesis represent maximum multipliers of characteristic differences, $\Delta\mathbf{W}=\mathbf{P}^{-1}\Delta\mathbf{U}$. (Note that an approximation sign must be used here since $\mathbf{P}$ is in fact different for every edge.) They correspond to interior "physical" fluxes, 1st order diffusive fluxes, antidiffusive fluxes and boundary fluxes, respectively, and are found by equating coefficients between the last two equations. For example,

$$
C_{ki}^{int}=C_{ki}^{upw}=\frac{\sigma}{\Omega_i}S\lambda_{max} \qquad (2.68)
$$

These two coefficients are the ones that already appear in the calculation based on linearized Fourier stability analysis (usually only for the first order scheme) and are the ones used commonly to calculate the allowable local timestep. It is apparent that an extra term ought to appear if the total variation is desired to decrease. This "antidiffusive" coefficient can be found by performing the following manipulation on the averaging function

$$\begin{aligned} L(\Delta \mathbf{W}_{ko}^{-}, \Delta \mathbf{W}_{ko}^{+}) &= \phi(r_{ko}^{+}) \Delta \mathbf{W}_{ko}^{-} \\ &= \phi(r_{ko}^{+}) \mathbf{P}^{-1} (\epsilon_{or} \Delta \mathbf{U}_{or} + \epsilon_{os} \Delta \mathbf{U}_{os}) \end{aligned} \tag{2.69}$$

Hence we find

$$C_{ki}^{ad} = \frac{\sigma}{A_i} (\epsilon_{ir} + \epsilon_{is}) S \lambda_{max} \phi(r)|_{max} \tag{2.70}$$

where $\phi(r)$ is the widely used notation for the limiter function and is related to $L(u,v)$ in Equation (2.69) by $L(u,v) = u\phi(v/u)$. For the present case, with the minmod limiter, $\phi_{max} = 1$. Finally the LED CFL-like condition analagous to Equation (2.65) for the Euler equations using Scheme I is

$$\mu_i = 1 - \Delta t_i \left(\sum_{ki}^{edges} (C_{ki}^{int} + C_{ki}^{upw} + C_{ki}^{ad}) + \sum_{kip}^{bou} C_{kip}^{bou} \right) \tag{2.71}$$

The supplemental LED check consisted of modifying Δt_i such that $\mu_i > \mu_t = 0$ was obtained. In fact it was found that this led to rather small values for Δt and consequently, slow convergence and subsequent investigation revealed that using a threshhold value of $\mu_t < 0$ allowed quick convergence without limit-cycling. Figure 2-5 shows the convergence histories of a typical flow analysis scheme with and without a check on the LED CFL-like condition (using $\mu_t = -1$). It can be seen that the check leads to dramatically improved convergence properties. Furthermore, the sensitivity calculation, with an identical Δt, was found to be stable for the CFL number the analysis scheme had been run at.

2.5 Parallelization

The resulting solution algorithm is readily parallelised, using domain decomposition and explicit message passing. The implementation is such that the message passing libraries MPI or PVM can be employed and the code can run on a variety of platforms, ranging from clusters of workstations to mainframe parallel machines, such as the IBM SP2 and the CRAY T3D. At present, the mesh generation is carried out in a serial manner and the computational mesh is decomposed after the mesh generation is completed. It has been decided to follow a strategy in which the mesh is partitioned in such a way that edges belong to a single subdomain, whereas nodal points may be shared between several subdomains. The domain is decomposed using a coordinate bisection algorithm. All the communication arrays, necessary for the transfer of information between the subdomains, are evaluated during the domain partitioning stage. Within each subdomain, the edges are subdivided into two groups. One group contains the inter–domain boundary nodes, i.e. the nodes requiring communication. The second group contains interior nodes only. Typically, a loop over all the edges containing boundary nodes is performed first. The values accumulated at the boundary nodes are then broadcast to the neighbouring domains. The interior edges are considered next and the information from the neighbouring domains is received. In this way, communication and computation take place concurrently. The parallel efficiency achieved using this strategy with 12 processors on the IBM SP2 at MIT is higher than 95% for the simulation of practical problems. With this implementation, the inviscid flow analysis code (based on Scheme I) requires 0.32 microseconds per time step per mesh point while the adjoint code (with Jacobian calculations performed on the fly, as discussed above) requires 2.42 microseconds.

2.6 Optimization Algorithm

Two unconstrained optimization algorithms were used: (1) the BFGS algorithm, which is an efficient gradient-based algorithm suitable for use when gradients are calculated via either the adjoint or the direct method and (2) the Inverse Design Newton method, suitable for use especially with the direct method, which uses the flow variable sensitivities to make an estimate of the Hessian matrix. These are discussed in the following two sections. Constrained methods are discussed in the last subsection.

2.6.1 Unconstrained BFGS

In the BFGS method [19, 37], an increasingly accurate estimate, $\mathbf{B}_k$, of the Hessian matrix, $\mathbf{H}$, (whose entries are given by $H_{ij} = \frac{\partial^2 \mathcal{F}}{\partial\beta_i \partial\beta_j}$) is made based on changes in the gradient vector along the optimization path. The expression for the updated approximation to the Hessian at iteration k is

$$\mathbf{B}_{k+1} = \mathbf{B}_k + \frac{1}{\mathbf{g}_k^T \mathbf{p}_k} \mathbf{g}_k \mathbf{g}_k^T + \frac{1}{\delta_k \mathbf{y}_k^T \mathbf{p}_k} \mathbf{y}_k \mathbf{y}_k^T \tag{2.72}$$

where $\mathbf{g}_k$ is the gradient while $\mathbf{p}_k$ is the search direction both at iteration k, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$ and δ_k is the step size from iteration k along direction $\mathbf{p}_k$. A Newton step is then taken, whereby

$$\mathbf{B}_k \mathbf{p}_k = -\mathbf{g}_k \tag{2.73}$$

and the update to the design variable is

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \delta_k \mathbf{p}_k. \tag{2.74}$$

The initial estimate for the Hessian is $\mathbf{B}_0 = \mathbf{I}$.

To find δ_k, an adapted version of the Golden Section line search was used for some inverse design optimization cases discussed in Chapter 4. For those design iterations where the design variable update led to an increase in cost function, due to the nonparabolic nature of the cost function variation for example, the design change was continually reduced by an amount given by the golden section factor $\Delta\beta^{i+1} = \frac{3-\sqrt{5}}{2}\Delta\beta^i$ until a decrease in the cost function was found. Otherwise Equation (2.73) was used to perform just one update on the design vector for each iteration, k, and $\delta_k = 1$ is taken.

2.6.2 Unconstrained Inverse Design Newton

This is a Newton algorithm [12, 19] in which an approximation, $\mathbf{B}_k$, of the Hessian matrix ($\mathbf{H}$) is made to estimate the location of the minimum.

$$B_{ij} \approx \sum_{k=1}^{N_{wing}} \frac{\partial p}{\partial \beta_i} \frac{\partial p}{\partial \beta_j} \Delta s_k \tag{2.75}$$

where N_{wing} is the number of point on the wing or target surface. This estimate becomes increasingly accurate as the target is approached. Once again Equation (2.73) is used to find the update to the design vector. The "informal" form of the Golden section line search described in the previous section was also used with this algorithm.

2.6.3 Constrained algorithms

For constrained optimization problems in Chapters 4 and 5, the approach taken is to incorporate the constraints during the optimization stage. This approach assumes the availability of constraint gradients $\mathbf{a}_j = \partial c_j / \partial \boldsymbol{\beta}$ and *partially* solves a sequence of linearly constrained subproblems. Each linearly constrained subproblem is one in which the constraint is modelled as possessing linear variation,

$$\tilde{\mathbf{c}}_k = \mathbf{c}_k + \mathbf{A}_k \boldsymbol{\beta} \tag{2.76}$$

where $\mathbf{A}_k$ is a matrix whose rows are the constraint gradients, $\mathbf{a}_j^T$, for subproblem k.

We shall first discuss the incorporation of constraints in the context of the Steepest Descent method. The extension to the more efficient BFGS algorithm is discussed afterwards.

The Constrained Steepest Descent method finds a search direction, $\mathbf{p}_k$, by simply subtracting from the steepest descent direction, $-\mathbf{g}_k$, all components of the scaled cost function gradients which are parallel to the set of constraint gradients $\mathbf{a}_j$. The Constrained Steepest Descent method can be shown to be equivalent to the Steepest Descent method in the feasible subspace and in turn to augmenting the cost function with a term of the form $-\boldsymbol{\mu}_k^T\mathbf{c}(\beta)$ to form a Lagrangian [12, 19]. The Lagrange multipliers, $\boldsymbol{\mu}_k$, are determined by solving a system including equations defining the steepest descent direction of the Lagrangian and equations enforcing the constraints. The search direction thus obtained can be expressed as

$$\mathbf{p}_k = -\mathbf{g}_k + \mathbf{A}_k^T[\mathbf{A}_k\mathbf{A}_k^T]^{-1}\mathbf{A}_k\mathbf{g}_k \tag{2.77}$$

which can be alternatively written as

$$\mathbf{p}_k = -\mathbf{Z}_k\mathbf{Z}_k^T\mathbf{g}_k \tag{2.78}$$

where $\mathbf{Z}_k$ is the matrix whose columns form an orthonormal basis for the space spanned by the set of vectors orthogonal to the rows of $\mathbf{A}_k$. In this work, $\mathbf{Z}_k$ is formed by performing an LQ decomposition of $\mathbf{A}_k$.

For every Constrained Steepest Descent direction, a line search is performed. If δ denotes the line search parameter, a one–dimensional quadratic model, $\tilde{\mathcal{F}}(\delta)$, of the objective function along the search direction is constructed. The algorithm used makes use of the slope information available at the start of the line search but is tailored to avoid further slope calculations due to the relatively high cost of adjoint analysis relative to flow analysis in our explicit algorithms. In short, it involves first taking a step, along the search direction, that is small enough so that a reduction in the cost function is guaranteed. Thereafter, the minima of successive one-dimensional quadratic representations of the search space are tested until no further reduction in $\tilde{\mathcal{F}}(\delta)$ is found. The successive quadratic models are generated using slope and function information which follows the quadratic/cubic backtracking algorithm described in Dennis et al [10].

The Constrained Steepest Descent algorithm used in this work varies from the "sequence-of-linear-subproblems" algorithm suggested in [19] in that each linear subproblem is only partially solved. Once an adequate estimate of the minimum in direction $\mathbf{p}_k$ has been found, a new linear subproblem is formed by constructing a new model for the constraint variation as given by Equation (2.76).

The constraint gradients $\mathbf{a}_j$ are computed by solving the constraint adjoint equation — analogous to Equation (2.32) — at $\boldsymbol{\beta}_k$, the design variable vector found at the end of line search k. This modification to the usual approach is intended to avoid wasting effort in finding the optimum for a subproblem, to a high degree of accuracy, while still being far from the minimum of the nonlinearly constrained problem.

The nonlinearity of the constraints requires an algorithm for returning the design back to the feasible subspace at the end of each subproblem line search, k. This is done by performing a projection, in the subspace $\mathbf{A}_k^T$ *spanned* by the constraint gradients, $\mathbf{a}_j$. Therefore, the step $\mathbf{q}_k$ required to return the design to feasibility is a combination of the constraint gradients,

$$\mathbf{q}_k = \mathbf{A}_k^T \mathbf{p}_Y \tag{2.79}$$

where $\mathbf{p}_Y$ is a vector in the subspace Y formed by using the constraint gradients $\mathbf{a}_j$ as a basis. The feasibility condition is simply

$$\tilde{\mathbf{c}}_{\text{feasible}} = \mathbf{c}_k + \mathbf{A}_k \mathbf{q}_k = 0. \tag{2.80}$$

Elimination produces

$$\mathbf{p}_Y = -[\mathbf{A}_k \mathbf{A}_k^T]^{-1} \mathbf{c}_k \tag{2.81}$$

which is an easily solved $M \times M$ linear system. Finally, we arrive at the starting point $\boldsymbol{\beta}_{k+1}$ for the next subproblem

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \delta_k \mathbf{p}_k + \mathbf{q}_k. \tag{2.82}$$

The algorithm for constrained optimization method described, can be extended to the more efficient constrained BFGS (quasi-Newton) method that was actually used in the three–dimensional calculations. In the *unconstrained* BFGS algorithm, an increasingly accurate estimate of the Hessian matrix is built up as discussed in Section 2.7.1. The details of the algorithm are described in [19].

The modification to the BFGS algorithm for incorporation of constraints is simply to form the approximation to the Hessian *in the subspaces* $\mathbf{Z}_k$, found for each subproblem, k. The resulting expression for the updated approximate projected Hessian is

$$\mathbf{B}_{Z_{k+1}} = \mathbf{B}_{Z_k} + \frac{1}{\mathbf{g}_{Z_k}^T \mathbf{p}_{Z_k}} \mathbf{g}_{Z_k} \mathbf{g}_{Z_k}^T + \frac{1}{\delta_k \mathbf{y}_{Z_k}^T \mathbf{p}_{Z_k}} \mathbf{y}_{Z_k} \mathbf{y}_{Z_k}^T \tag{2.83}$$

where $\mathbf{p}_{Z_k} = \mathbf{Z}_k^T \mathbf{p}_k$, $\mathbf{y}_{Z_k} = \mathbf{Z}_k^T(\mathbf{g}_{k+1} - \mathbf{g}_k)$ and δ_k is the step size from the previous iteration. As before, a Newton step is then taken, whereby

$$\mathbf{B}_{Z_k} \mathbf{p}_{Z_k} = -\mathbf{g}_{Z_k} \tag{2.84}$$

and the search direction is found using $\mathbf{p}_{k+1} = \mathbf{Z}\mathbf{p}_{Z_k}$.

The advantage of the BFGS algorithm over the Steepest Descent algorithm is that once a good estimate, $\mathbf{B}_k$, of the local Hessian, $\mathbf{H}$, is formed and once the current design $\boldsymbol{\beta}$ is close enough to $\boldsymbol{\beta}^{opt}$, quadratic Newton convergence is found. Also, for the problems we have encountered, we have found that after a sufficiently accurate first line search, the constrained BFGS algorithm requires only one solution to be performed per line search.

Furthermore, it is noted that implementation of the BFGS algorithm in the subspace $\mathbf{Z}_k$ orthogonal to $\mathbf{A}_k^T$ ensures that the Hessian approximation, $\mathbf{B}_k$, remains positive definite (provided sufficient progress is made towards the minima of each line search) even though the subspace $\mathbf{Z}_k$ in general changes from one iteration to the next. (Note that an algorithm which forms the BFGS approximate Hessian $\mathbf{B}_k$ in untruncated design space, and follows this by projection of the resulting direction $\mathbf{p}_k$ into the feasible subspace, may form a non-positive-definite $\mathbf{B}_k$. This may impede or prevent convergence of the optimization process [19].) For the current algorithm, the fact that $\mathbf{Z}_k$ changes causes $\mathbf{B}_k$ to more slowly approach the actual Hessian $\mathbf{H}$, and thus slows convergence. However, in our tests so far, we still observe adequate asymptotic convergence.

Finally, it is noted that for the BFGS method, each design iteration requires M flow solutions as well as $2M$ adjoint solutions for the sensitivities of the objective and the constraints (barring the first iteration for which we find it useful to perform an accurate line search and therefore nM flow solutions where $n > 1$ is some integer).

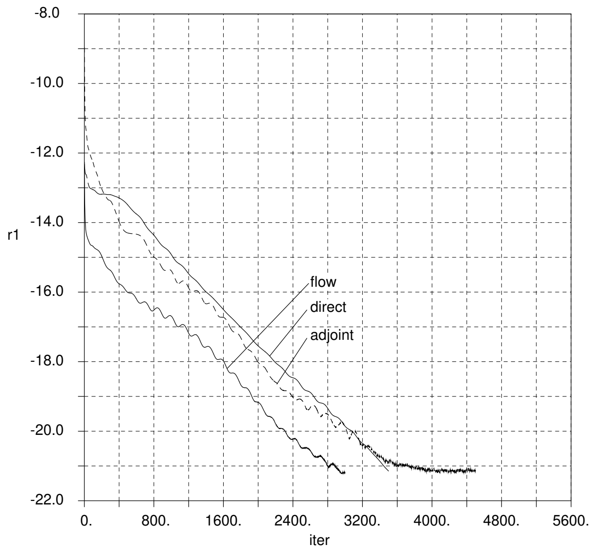

Figure 2-3: Convergence histories for Ni's bump, $M_\infty = 0.675$

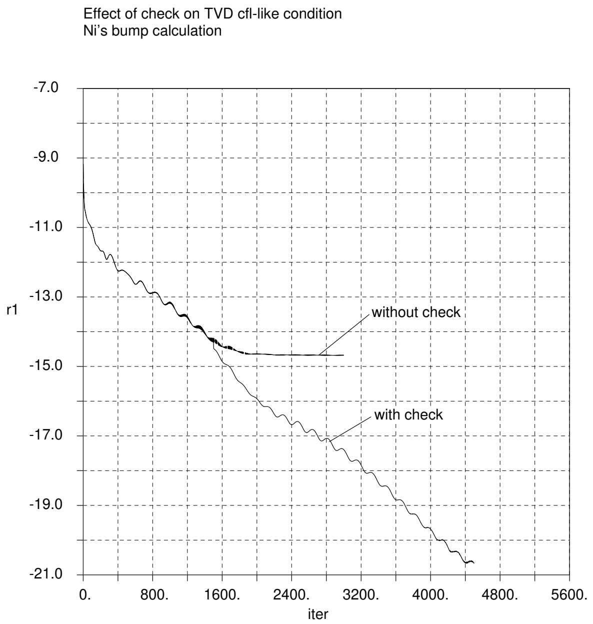

Figure 2-5: Convergence histories with and without LED CFL-like check, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

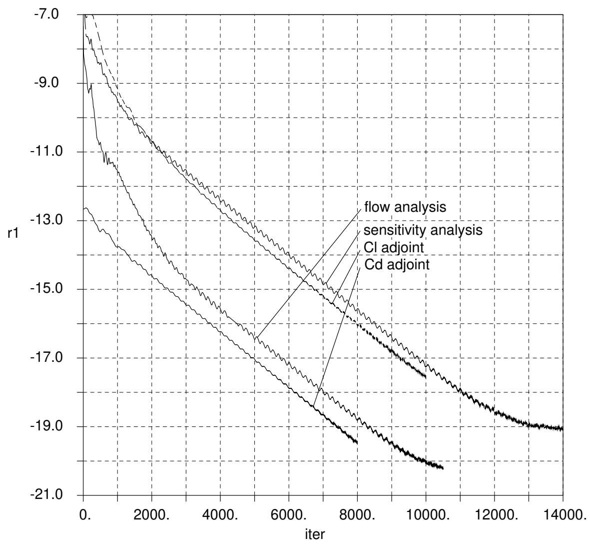

Figure 2-4: Convergence histories for NACA0012 airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

Chapter 3

Validation

This chapter presents a brief review of some of the validation exercises performed for the flow and sensitivity analysis algorithms using Scheme I and Scheme II. Scheme I was used for the inviscid examples presented here and only sensitivity validation examples are presented. Scheme II was used for the viscous examples presented here and both flow analysis and sensitivity analysis validation examples are presented.

3.1 Inviscid Sensitivity Validation

Validation of design variable sensitivities from selected applications are presented in this section for the algorithm referred to above as Scheme I. Table 3.1 shows cost function sensitivities to these design variables. The first example uses C_d and C_l— Equation (D.3) and Equation (D.2) — for the cost function while the last three examples use the area-integrated sum of squared differences of the actual pressure from a target pressure distribution over the target surface, as given by Equation (D.1).

3.1.1 2D Single-element Flowfield — Angle of Attack Design Variable

The grid and C_p distribution from the flow analysis solution are shown in Figures 3-1 and 3-2. Figures 3-3 and 3-4 compare distributions of the local values of $\partial \rho E / \partial \alpha$ as found using the direct method and the finite difference method, respectively. Figure 3-5 shows a comparison of analytic versus finite difference sensitivities $\partial \rho E / \partial \alpha$ for the angle of attack design variable on the airfoil

Application	Cost Function	Design Variable	AD (Adjoint)	AD (Direct)	FD
2D Single	C_l	α	0.231	0.231	0.233
2D Single	C_d	α	1.69e-2	1.69e-2	1.72e-2
2D Single	C_l	camber	24.63	24.63	24.35
2D Single	C_d	camber	1.451	1.451	1.481
2D Multi	Inv Des	flap rot.	1.93e-4	1.91e-4	1.88e-4
3D Single	Inv Des	camber	-0.634	-0.635	-0.656
3D W/B	Inv Des	camber 1	-3033.	-3020.	-2807.

Table 3.1: Comparison of Various Sensitivities

surface. Close agreement is evident in all cases. Use of these direct method solutions as part of the calculation of $dC_l/d\alpha$ and $dC_d/d\alpha$ results in the values listed in the first and second rows of Table 3.1. (Also tabulated are corresponding divided difference and adjoint values.) The third and fourth rows contain sensitivities to the NACA0012 meanline design variable whose associated functional variation in x is given by Equation (C.9).

3.1.2 2D Multielement Airfoil Flowfield — Flap Rotation Design Variable

The grid and flow analysis solutions are shown in Figures 4-1 and 4-3. The direct-method point flow sensitivities were validated by comparing them with corresponding finite difference quantities. An example of this comparison is shown in Figure 3-6 where surface density sensitivity to the flap rotation design variable — given by Equation (C.12) — is plotted versus axial location for finite difference and direct sensitivity analysis. Other variables tested — such as a main element camber variable — Equation (C.9) and a flap vertical deflection variable — Equation (C.11) — exhibited similar agreement. Correlation away from the airfoil surface was also good. Cost function sensitivities as calculated by both direct and adjoint methods - using Equations (2.29) and (2.33), respectively - were validated also by comparing with finite difference. These comparisons are presented in row 5 of Table 3.1 for the flap rotation design variable. Differences on the order of less than 1% were usually found between adjoint and direct methods (depending on the level of convergence), while comparison of the analytic sensitivities versus finite difference produced differences on the order of 2% or so but, in this case, also depending on the size of the design variable perturbation.

3.1.3 3D Infinite Straight Wing Flowfield — Camber Design Variable

The grid and the C_p distribution from the flow analysis solution are shown in Figures 4-7 and 4-9. Figure 3-7 shows a comparison of surface density sensitivity to the camber design variable — Equation (C.9) — as calculated by direct and finite difference methods (using $\Delta\beta = .0001c$). It can be seen that the agreement is quite good, with some of the noisiness in the finite difference plot being due to traversal of discontinuous portions of the limiting function distribution for some edges. Further validation of the sensitivity calculations was provided by performing the full cost function sensitivity calculation for adjoint, direct and finite difference methods. For the camber design variable, the values are printed in row 6 of Table 4.1. (The cost function is calculated based on a target, $p^*(x)$, generated by perturbing the design variables by such an amount as to reproduce the NACA1410 airfoil – or a close replica of it since thickness was applied vertically from the meanline rather than normal to it).

3.1.4 3D Wing/Body Flowfield — Camber Design Variable

The grid and flow analysis solutions are shown in Figures 4-11 and 4-13. Figure 3-8 shows a comparison of analytic versus finite difference sensitivity at $\eta = 66\%$ for a design variable whose associated chordwise $f_j(x)$ and spanwise $g_k(y)$ functions are given by Equation (C.9) and Equation (C.13), respectively. The chordwise and spanwise functions are combined using Equation (C.3). For this design variable, Equation (C.13) is used with $y_{k-1} = y_{root}$, $y_k = y_{midspan}$ and $y_{k+1} = y_{tip}$. Again agreement between the analytic sensitivities and the finite difference sensitivities is quite good with discrepancies being due to the size of the finite difference step size. Similar agreement is found at other span stations and for other state vector variable sensitivities. Full cost function sensitivity calculations were performed and are shown in row 7 of Table 3.1. The cost function in this case is calculated based on a target pressure distribution generated by perturbing the two camber design variables with a chordwise function given by Equation (C.9) as above and two spanwise functions given by Equation (C.13) and two thickness design variables with a chordwise function given by Equation (C.10) and the same two spanwise functions given by Equation (C.13). The first spanwise function is described above. The second uses $y_{k-1} = y_{midspan}$, $y_k = y_{tip}$ and $y_{k+1} = y_{tip}$. The perturbation sizes for the target correspond to vertical movement of the mean line by $\Delta\frac{y_c}{c}(\frac{x}{c} = .4) = .02$ (at the spanwise design variable maxima) for the camber variables and to a change in maximum thickness of $-.04c$ for the thickness variables.

3.2 Viscous Flow Analysis Validation

As mentioned earlier, the flow analysis scheme — Scheme I — used for the inviscid optimization examples is quite dissipative. Although the flow analysis scheme — Scheme II — used for the viscous examples uses a slightly different gradient calculation method for the dissipation, the constants multiplying the dissipative fluxes were chosen to result in approximately the same net amount of dissipation for Schemes I and II. Hence, although the objective of this research is to produce an efficient design optimization scheme rather than a high fidelity flow analysis scheme, it was felt that it would be prudent to verify that the flow analysis scheme does possess some degree of accuracy, and that the artificial dissipation does not overwhelm the physical viscosity present for the Navier-Stokes calculations. Towards this end, 2D and 3D validation exercises are presented which demonstrate that the corresponding codes do reproduce, to an adequate degree of accuracy for the purposes of this research, the expected results for flow over a flat plate and flow over a NACA0012 airfoil.

3.2.1 2D Flat Plate

This test case or variations of it have been extensively tested by other researchers [34, 2, 47]. At sufficiently large distances from the leading edge, the boundary layer profile is expected to closely approximate the Blasius profile [50] — provided sufficient spatial resolution is used. The grid used is shown in Figure 3-10. No slip boundary conditions are applied along the lower boundary from $x = 0$ to $x = 1.5$; free slip boundary conditions are applied on the lower boundary from $x = -1.5$ to $x = 0$; non-reflecting Riemann boundary conditions are applied at the left boundary; pressure is specified while the appropriate characteristic variables are extrapolated from the interior along the top and right boundaries. Flow is from left to right. Freestream conditions are M_∞, $\alpha = 0°$ and $Re = 1000$.

The profiles found at $x = 1.215$ using Scheme II for successively coarser grids (coarsened by removing every other vertical and horizontal in multigrid fashion [2]) are shown in Figure 3-9. It can be seen that the solution spatially converges to a close approximation of the Blasius solution.

3.2.2 2D NACA0012

This subsection presents the results of a more relevant validation example, that of a NACA0012 airfoil immersed in a flow with the following freestream conditions: $M = 0.8, \alpha = 3.5°, Re = 2000$. Without a theoretical solution with which to approximately validate the results, we resort to MSES, a coupled Euler/integral boundary layer code [11]. The grid for the Scheme II solution is shown in Figure 5-9. The C_p and M distributions are shown in Figures 5-10 and 5-11. There are 7081 points in the grid and 17 points in the boundary layer at the trailing edge, although only the first 9 elements are "structured" [33]. This small number of "structured" elements in the boundary layer is the possible cause of the wiggles observable in the C_p distribution. Another possible cause is insufficient spatial resolution in the outer part of the boundary layer. In spite of these low level errors, surface C_p and C_f distributions agree quite well with the corresponding distributions as found along the displacement surface and airfoil surface, respectively, by MSES. This can be seen in Figures 3-11 and 3-12. On the lower surface, agreement is quite good, while on the upper surface, the differences are partly due to the non-zero normal pressure gradient through the boundary layer and partly due to the larger presence of artificial viscosity in the Scheme II solution which has the effect of thickening the boundary layer.

3.2.3 3D Flat Plate

The grid used is shown in Figure 3-13. The boundary conditions applied in this calculation are identical to the 2D flat plate calculation with the exception that an additional symmetry boundary condition is applied at the spanwise (or z) boundaries. Flow is again from left to right. Variation of C_p and ρu for the resulting solution are shown in Figures 3-14 and 3-15.

As in Section 3.2.1, at sufficiently large distances from the leading edge, the boundary layer profile is expected to closely approximate the Blasius profile — provided sufficient spatial resolution is used. The profiles found at $x = 0.96$ — for various locations in the spanwise direction — are shown in Figure 3-16. It can be seen that the solution represents quite a close approximation of the Blasius solution. Slight variation of the boundary layer profiles in the spanwise direction are due to the effect of different control volumes at either spanwise boundary resulting in different discretization errors. It was found that this spanwise variation decreased as the grid was refined, as expected. Convergence to a somewhat close approximation of the Blasius displacement thickness is observed sufficiently far away from the leading edge.

3.2.4 3D NACA0012

A calculation was performed for a two-dimensional wing, formed by extruding the NACA0012 airfoil in the spanwise direction, with freestream conditions of $M = 0.8, \alpha = 3.5°, Re = 2000$. The grid is shown in Figure 3-17. The C_p and M distributions are shown in Figures 3-18 and 3-19. The surface

C_p distributions are compared with the corresponding distributions as found along the displacement surface by MSES in Figure 3-20. Both surfaces exhibit good agreement and boundary layer quantities such as δ^* and C_f (not shown) also agree well.

3.3 Viscous Sensitivity Validation

Validation of design variable sensitivities for selected applications are presented in this section for the viscous algorithm referred to above as Scheme II.

3.3.1 2D NACA0012

Sensitivity calculations were performed based on the solution described in Section 3.2.2. Comparison of the resulting $\partial \rho u / \partial \alpha$ distribution (Figure 3-21) with that found using the finite difference method based on $\Delta\alpha = 0.01\,\mathrm{deg}$ (Figure 3-23), reveals good agreement. Comparison of $\partial C_l/\partial\beta$ and $\partial C_d/\partial\beta$ as found by finite difference, adjoint and direct methods are shown in Table 3.2 below for design variables of α and a NACA 4-series meanline camber mode — Equation (C.9). Quite good agreement is found for α. The discrepancies for the camber mode are believed to be due to the finite difference step size used in making that estimate. $\Delta\beta_c = 0.001$ was used corresponding to a movement in the airfoil surface of 0.1% of chord.

To put a proper perspective on the source of this discrepancy, much can be learned by examining the sensitivities of the only flow quantities that contribute to the lift and drag. Surface pressure sensitivities are compared in Figure 3-22 revealing good agreement. Surface values of $\partial\tau_{11}/\partial\beta$ are shown in Figure 3-24 and also reveal good agreement. Similarly good agreement is observed for $\partial\tau_{12}/\partial\beta$ and $\partial\tau_{22}/\partial\beta$. Since these 4 scalar values are the only ones that contribute to C_l and C_d, $\partial C_l/\partial\beta$ and $\partial C_d/\partial\beta$ are expected to show similarly good agreement. The fact that the discrepancies shown in Table 3.2 are larger than expected is presumably due to the small errors — observable in Figures 3-22 and 3-24 — for an integration which essentially involves taking the difference of two large numbers to obtain a small number. This is a scenario which is conducive to error magnification. Nonetheless it would probably be concluded at this juncture that the analytic derivatives are the ones with higher accuracy after examination of Figures 3-22 and 3-24. Further tests in the course of the optimization exercises (discussed in Section 5.1.2) by comparison of $\Delta\beta_j \partial C_l/\partial\beta_j$ with ΔC_l and $\Delta\beta_j \partial C_d/\partial\beta_j$ with ΔC_d provided further definitive evidence in this regard.

β	$\partial C_l/\partial\beta\|_{FD}$	$\partial C_l/\partial\beta\|_{adj}$	$\partial C_l/\partial\beta\|_{dir}$
camber	-0.9086	-1.0075	-1.0075
α	0.04707	0.04811	0.04811
	$\partial C_d/\partial\beta\|_{FD}$	$\partial C_d/\partial\beta\|_{adj}$	$\partial C_d/\partial\beta\|_{dir}$
camber	-0.0385	-0.04902	-0.04917
α	0.00627	0.00634	0.00634

Table 3.2: Comparison of 2D sensitivities to α and NACA 4-series meanline mode

3.3.2 3D NACA0012

Sensitivity calculations were performed based on the solution described in Section 3.2.4. Comparisons (finite difference versus analytic derivative) of the resulting surface $\partial\rho/\partial\beta$ and $\partial\rho E/\partial\beta$ distributions are shown for the angle of attack design variable (Figures 3-25, 3-27) and the NACA 4-series meanline mode design variable given by Equation (C.9) (Figures 3-26, 3-28) revealing good agreement. The adjoint sensitivity module has been fully validated versus both finite difference and the direct sensitivity module. Table 3.3 shows the results for the parallel version of the code using only C_l as the cost function.

β	$\partial C_l/\partial\beta\|_{FD}$	$\partial C_l/\partial\beta\|_{adj}$
camber	-0.3296	-0.3700
α	0.03943	0.03917

Table 3.3: Comparison of 3D sensitivities to α and NACA 4-series meanline mode

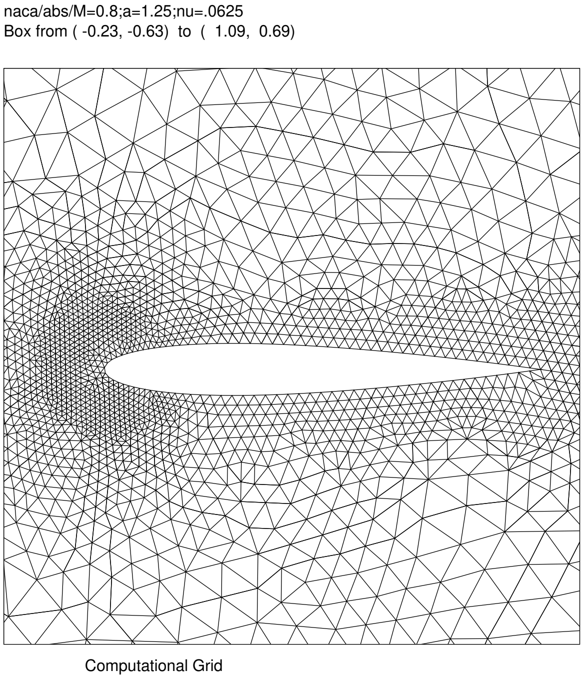

Figure 3-1: NACA0012 Grid

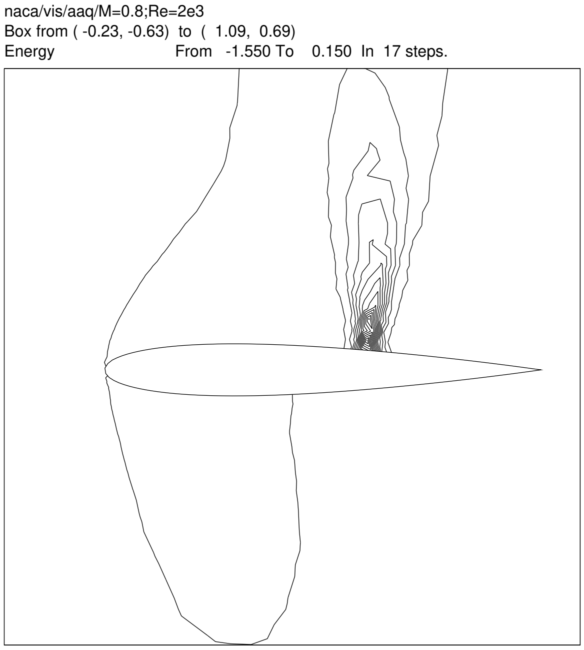

Figure 3-3: NACA0012. Analytically found distribution of $\partial \rho E / \partial \alpha$

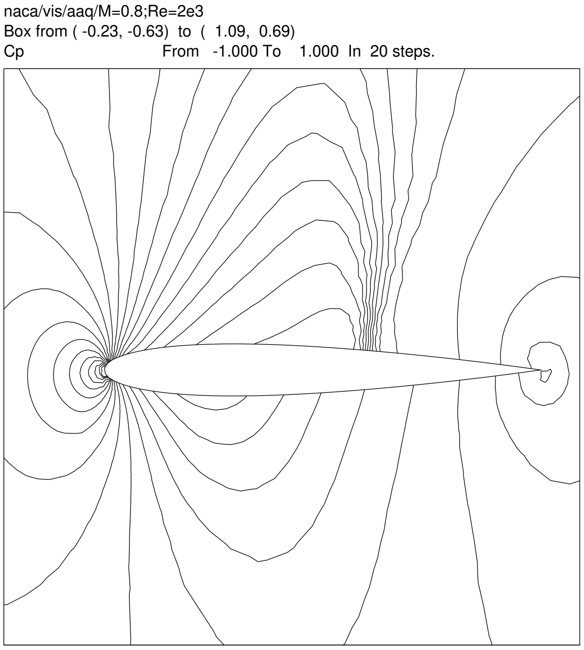

Figure 3-2: NACA0012 Cp Distribution. $M = 0.8$, $\alpha = 1.25°$

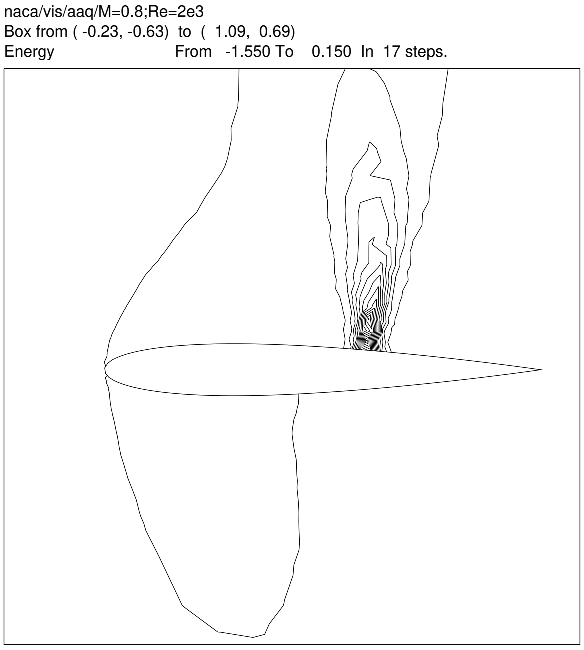

Figure 3-4: NACA0012. Distribution of $\partial \rho E / \partial \alpha$ found using finite difference method.

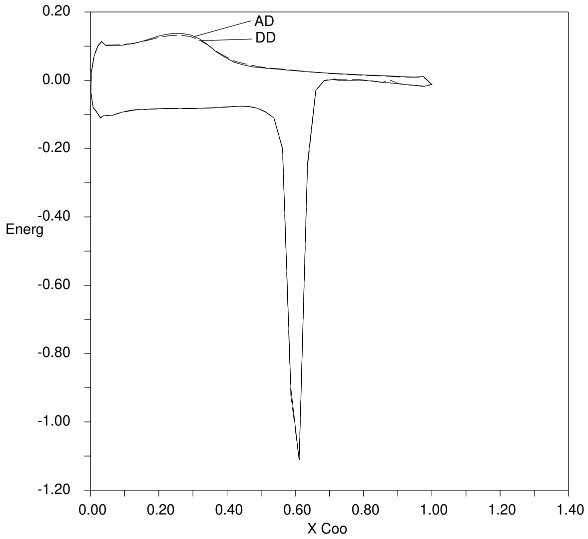

Figure 3-5: NACA0012. Finite Difference and Analytic Sensitivities: $\partial \rho E/\partial \alpha$ v. x

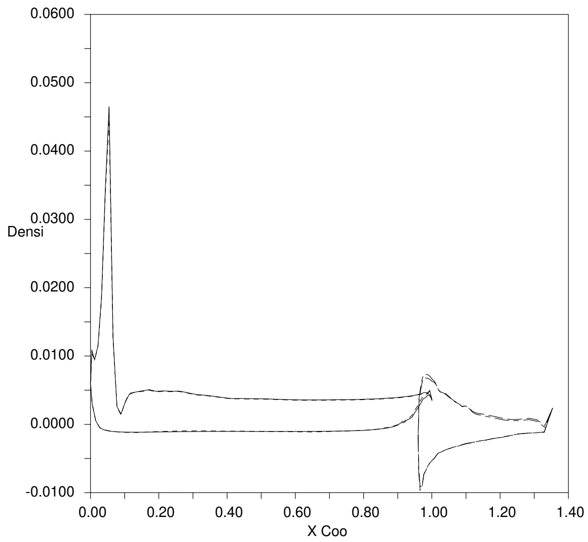

Figure 3-6: Analytic and FD density sensitivity v. x

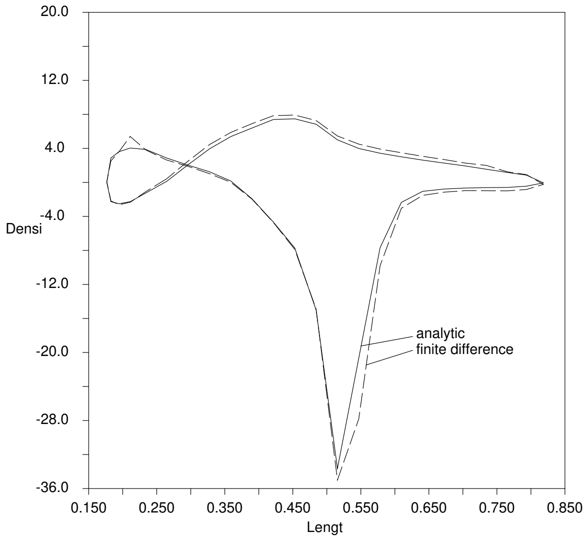

Figure 3-7: 3D Analytic and FD density sensitivity v. x at y=-.35 symmetry plane

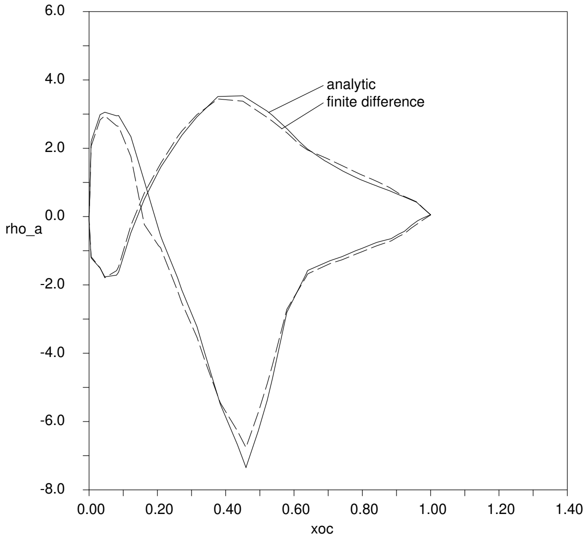

Figure 3-8: 3D Wing/body Analytic and FD density sensitivity v. x/c at 67.5% semispan

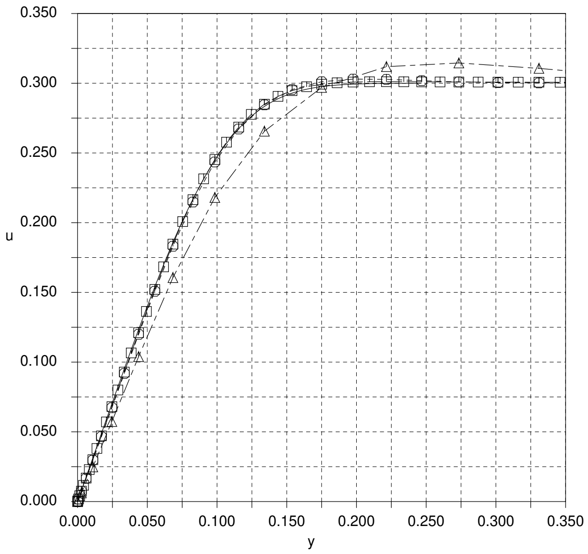

Figure 3-9: Spatial convergence of boundary layer profile at x=1.2 with local Blasius profile

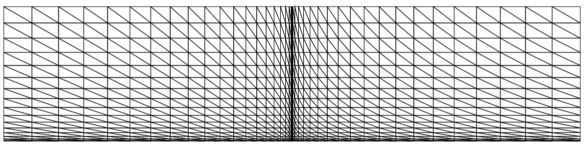

Figure 3-10: 41x17 grid used for flat plate calculations

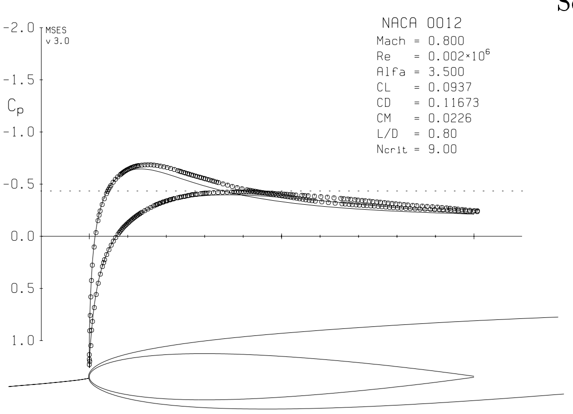

Figure 3-11: Surface C_p distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$: Scheme II and MSES solutions

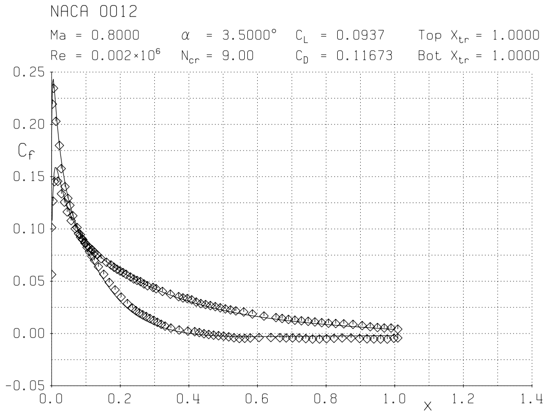

Figure 3-12: Surface C_f distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$: Scheme II and MSES solutions

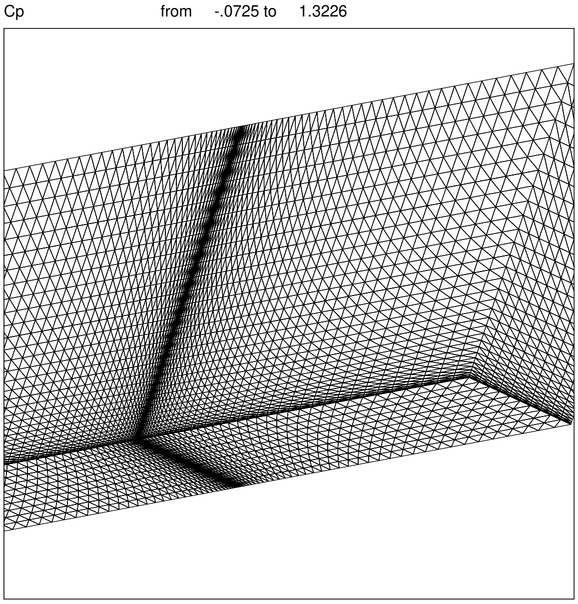

Figure 3-13: Grid used for flat plate calculation

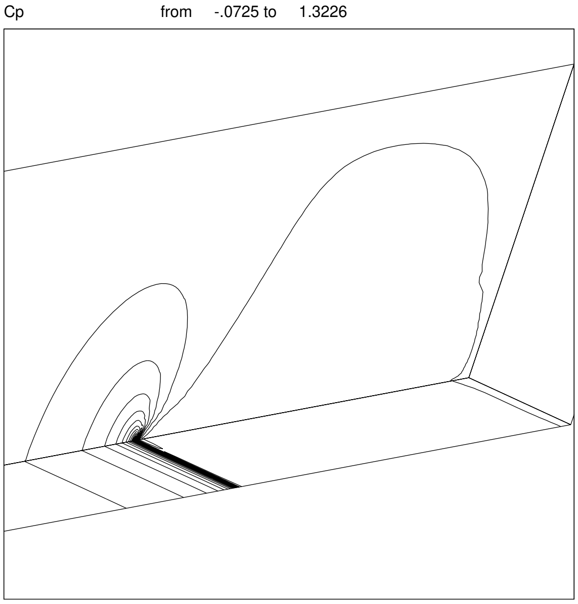

Figure 3-15: C_p distribution for flat plate calculation

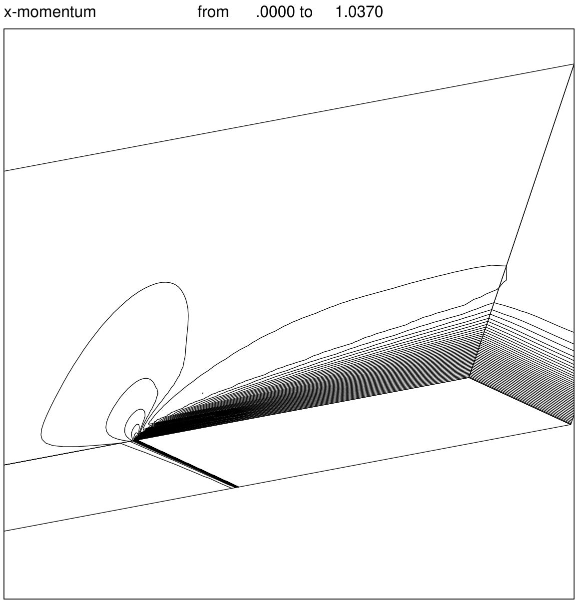

Figure 3-14: ρu distribution for flat plate calculation

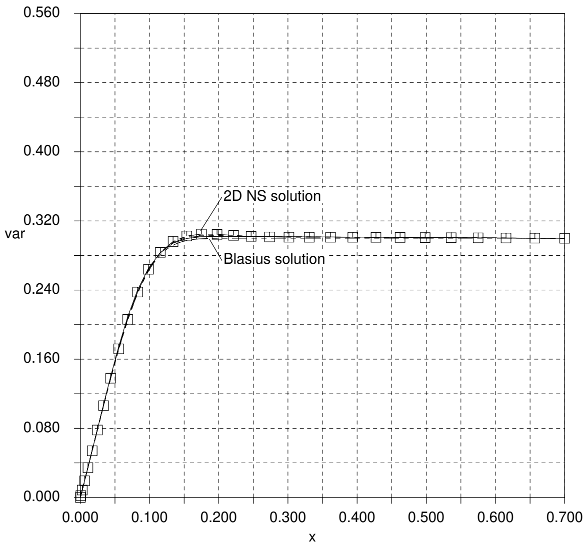

Figure 3-16: Comparison of 3D boundary layers at (x,y)=(0.96,0), (0.96,0.125), (0.96,0.25) with that found using 2D NS solver and Blasius solutions

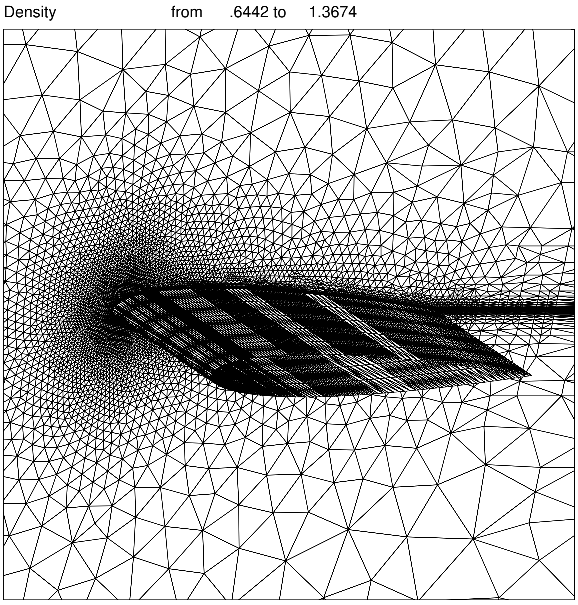

Figure 3-17: Grid used for 3D NACA0012 calculations

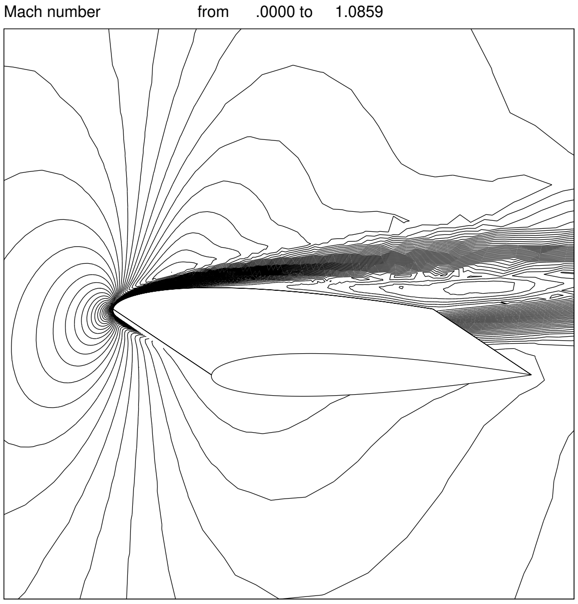

Figure 3-19: M distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

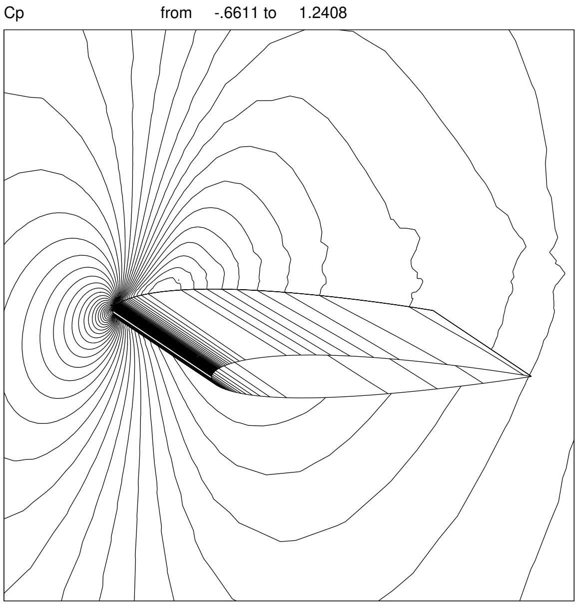

Figure 3-18: C_p distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

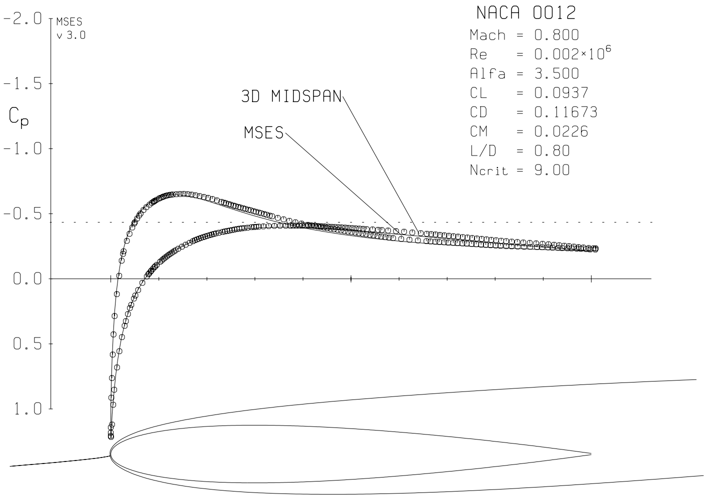

Figure 3-20: Surface C_p distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$: 3D NS solver midspan v. MSES

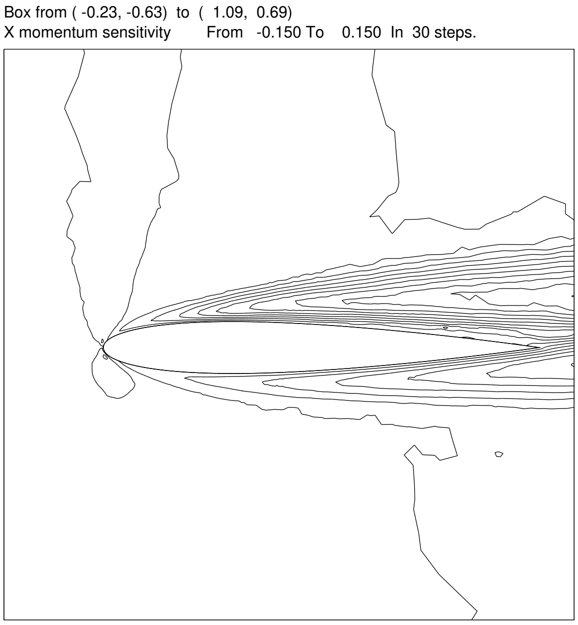

Figure 3-21: $\partial \rho u / \partial \alpha$ distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$ (Scheme II) found by direct method

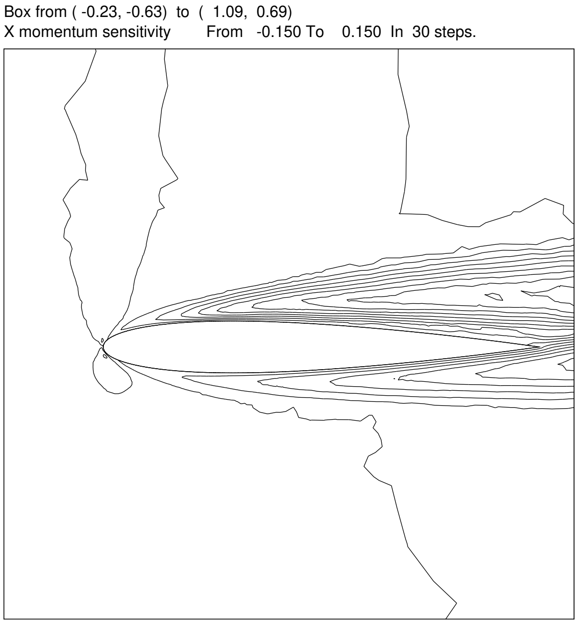

Figure 3-23: $\partial \rho u / \partial \alpha$ distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$ (Scheme II)from finite difference with $\Delta\alpha = 0.01$

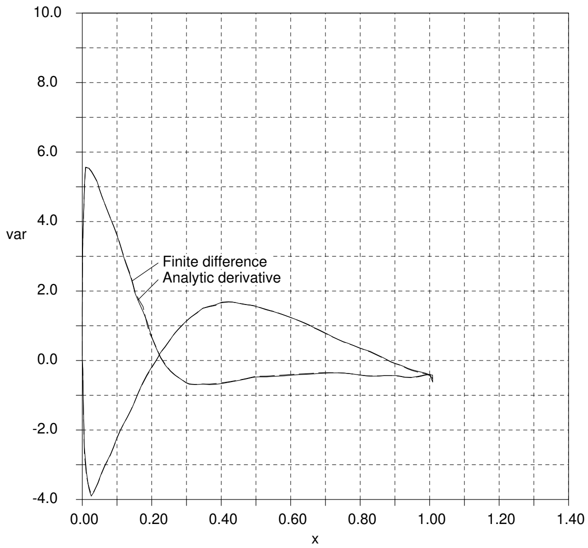

Figure 3-22: Surface $\partial p / \partial \beta_c$ distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$ (Scheme II v. Finite Difference)

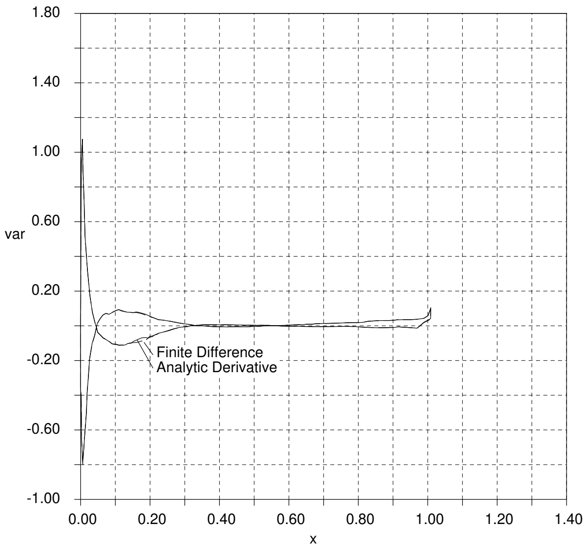

Figure 3-24: Surface $\partial \tau_{11} / \partial \beta_c$ distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$ (Scheme II v. Finite Difference)

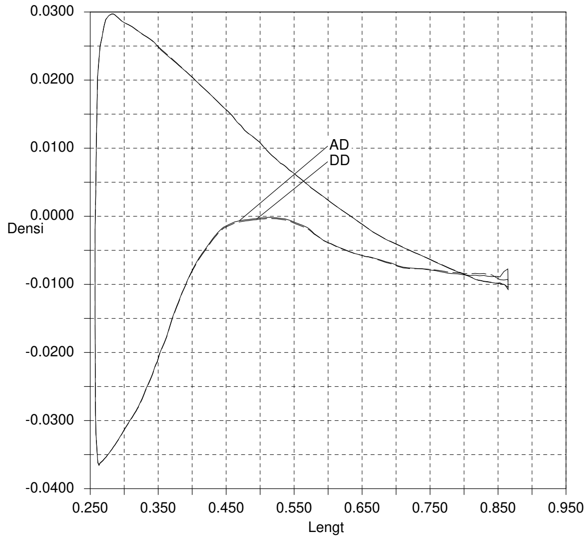

Figure 3-25: $\partial\rho/\partial\alpha$ distribution at midspan for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

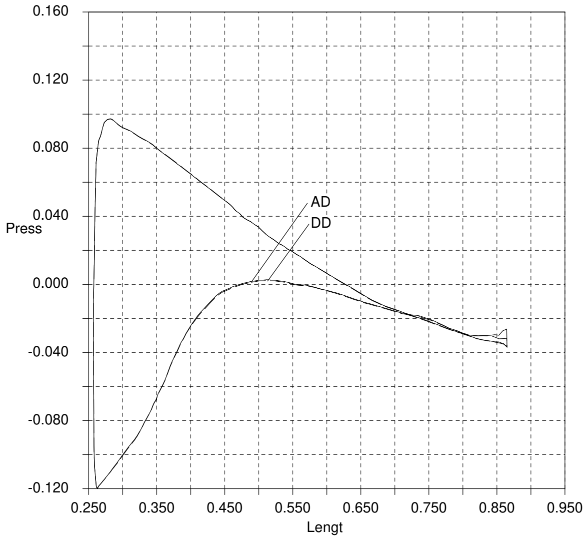

Figure 3-27: $\partial\rho E/\partial\alpha$ distribution at midspan for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

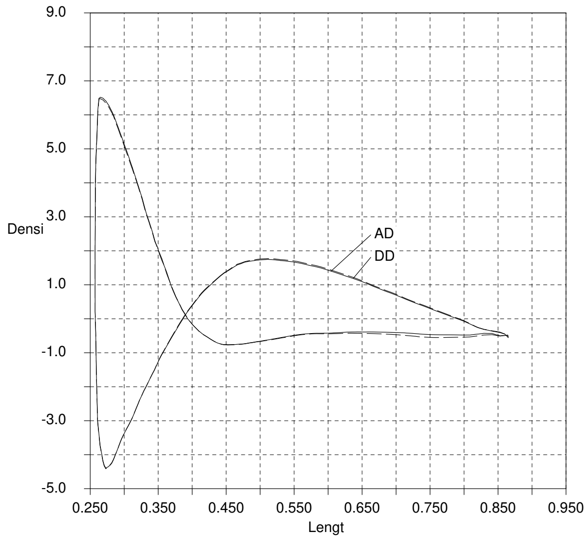

Figure 3-26: $\partial\rho/\partial\beta_c$ distribution at midspan for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

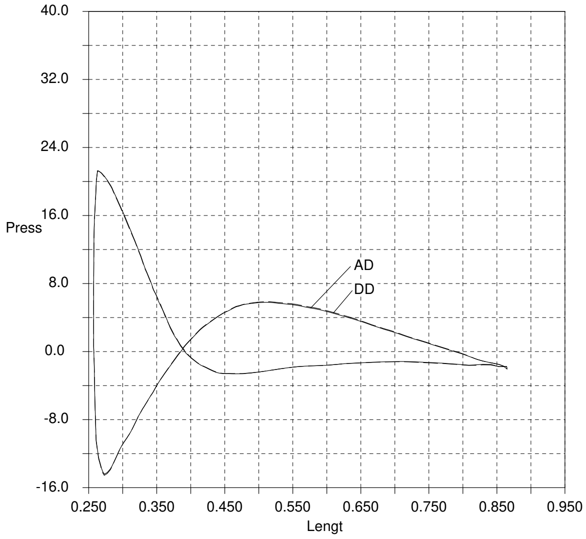

Figure 3-28: $\partial\rho E/\partial\beta_c$ distribution at midspan for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

Chapter 4

Inviscid Results

4.1 Inverse Design Examples

Although it is felt that inverse design is less preferable than drag minimization as a means of designing airfoils, wings and aircraft (due to the requirement for knowledge of a desirable pressure distribution), inverse design exercises, in which the target is generated by perturbing design variables by a known amount, also serves the purpose of providing a more complete validation of the cost function sensitivities across a range of design space. Also, inverse design exercises tend to be better behaved due to the more parabolic nature of the design surface than, for example, a design surface in which the objective function is drag with a constraint on the lift. Hence, the initial tests for the inviscid optimization system were based on inverse design.

4.1.1 2D Multielement Airfoil

To demonstrate the accuracy of the sensitivities, a modal inverse design optimization exercise on a multielement airfoil was performed. The baseline geometry was a NACA4412 main element with a scaled-down NACA4415 flap placed just below and aft of the main's trailing edge with 3% overlap. The initial grid is shown in Figure 4-1. The baseline solution at freestream conditions of $M_\infty = 0.425$, $\alpha = 9.0°$ is shown in Figure 4-3. Three design variables were used: flap rotation angle about a point near the flap leading edge — Equation (C.12); flap vertical deflection — Equation (C.11); and a camber design variable on the main element — Equation (C.9). This camber variable is given by a vertical surface perturbation with the same x variation as the NACA 4-series meanline [49] with maximum camber at 40% of chord. Equation (D.1) was used for the cost function to allow modal inverse design optimization. The target pressure distribution, p^*, for the inverse design exercise was generated by perturbing all three design variables. For example, the flap was rotated 5°.

Two inverse design optimization exercises were performed: one each based on the direct and adjoint sensitivities.

Firstly, based on the sensitivities found using the direct method, the geometry was perturbed to a new state. In this case, the Newton descent algorithm discussed in Section 2.6.2 was used. The evolutions of surface pressure distribution and geometry in the course of the direct-method-based optimization process are shown in Figures 4-2 and 4-4. Figure 4-5 shows the variation of cost function and design variables as a function of design iteration. Convergence to the target is reached in 3 steps.

Secondly, with the adjoint-method-based sensitivities, we can no longer make such a good initial estimate of the Hessian matrix as we do with the direct-method-based sensitivities. Therefore, we use the BFGS method described in Section 2.6.1. As expected, this required more iterations (10 in this case) to reach convergence. With only three design variables, the direct method is about twice as fast as the adjoint method. However for a larger number of design variables the adjoint is expected to be much more efficient. Furthermore, for drag optimization in which the direct method loses its advantage of an accurate Hessian [12], the adjoint method is far superior.

4.1.2 3D Single-element Airfoil

For this case, the baseline geometry was a straight wing formed from NACA0012 sections and of spanwise extent equal to 70% chord. The computational domain is bounded by eight surfaces including two symmetry planes at the spanwise extremities of the wing. Freestream conditions are $M_\infty = 0.8$ and $\alpha = 1.25°$. Figure 4-7 and 4-9 show baseline grid and C_p contours, respectively. Two design variables were chosen to be that of the NACA 4-series meanline — Equation (C.9) — and thickness distributions — Equation (C.10) — with the meanline maximum located at 40% of chord.

Next, modal design optimization was performed based on the sensitivities found using the adjoint method. For this case, the BFGS described in Section 2.6.1 was used. Resulting geometry and surface pressure evolutions are shown in Figures 4-8 and 4-10 while the evolution of design variables and cost function is shown in Figure 4-6. Convergence is found at about 6 design iterations.

4.1.3 3D High Aspect Ratio Wing/Body

This case was based on a seed geometry which included a fuselage with circular cross-sections and a wing lofted by placing NACA0012 sections into a planform with the following parameters: leading edge sweep, $\Lambda_{LE} = 25°$, aspect ratio, $A_r = 9$, taper ratio, $\lambda = 0.2$. Freestream conditions were chosen to be $M_\infty = 0.8$ and $\alpha = 1.25°$.

The baseline grid and pressure contours are shown in Figures 4-11 and 4-13. The mesh contains 65,000 nodes and 350,000 elements. This optimization case was performed using the serial flow and adjoint solvers. Four design variables were used. The first two design variables were camber design variables with a chordwise function given by Equation (C.9) and two spanwise functions given by Equation (C.13). Equation (C.13) is used with $y_{k-1} = y_{root}$, $y_k = y_{midspan}$ and $y_{k+1} = y_{tip}$ for the first spanwise function while the second uses $y_{k-1} = y_{midspan}$, $y_k = y_{tip}$ and $y_{k+1} = y_{tip}$. The second two design variables were thickness design variables with a chordwise function given by Equation (C.10) and the spanwise functions given by Equation (C.13) with the same values of y_{k-1}, y_k and y_{k+1} used for the pair of camber design variables.

A target pressure distribution was generated by perturbing the camber design variables by an amount corresponding to vertical movement of the mean line of $\Delta\frac{y_c}{c}(\frac{x}{c} = .4) = .02$ (at the spanwise design variable maxima) and to a change in maximum thickness of $-.04c$. Hence the geometry of the target corresponds closely to a linear loft into the baseline planform of NACA2408 sections from $\eta = 55\%$ to $\eta = 100\%$ and a NACA0012 section at the wing-body intersection. It should be realized that this represents a one-third reduction in thickness and a very large camber change, both of which are probably much larger than the magnitudes of thickness and camber changes that might typically be found in a transport wing design process. They therefore represent a more challenging target than would typically be found.

Using the resulting pressure distribution as the target pressure, p^*, in the expression for the cost function, cost function sensitivities were calculated by the adjoint method. Modal design optimization was performed using the BFGS method, described in Section 2.6.1. Resulting geometry and surface pressure evolutions are shown in Figures 4-12 and 4-14 while the evolution of design variables and cost function is shown in Figure 4-15. It should be noted that after the eighth iteration, the BFGS optimization process had to be restarted because it was found that the optimization had reached a local minimum. This is not to be unexpected considering the large design variable changes that were being demanded. Also, at this point the thickness design variables were rescaled such that unit thickness design variable change caused approximately the same magnitude of maximum airfoil surface movement as unit camber design variable change. It can be seen that after the restart/rescaling action, the design process rapidly drives geometry and pressure to the target distributions. The robust convergence to the target, in addition to validation exercises (such as those described in Chapter 3 and those in [14]) provides confidence that the sensitivities are being accurately calculated by the current scheme.

4.1.4 3D Business Jet

This optimization example is of a business jet configuration consisting of wing, body, horizontal and vertical fins and fuselage mounted engines. Freestream conditions are $M_\infty = 0.85$ and $\alpha = 2°$. Engine exhaust total pressure and temperature increases are not modelled, i.e. flow through conditions are assumed. The baseline grid and pressure contours are shown in Figures 4-23 and 4-19, respectively. The grid contains 160,000 nodes and 860,000 elements. The target pressure distribution, p^*, was chosen to be that found on the wing when the strut-nacelle assembly is removed at the same conditions. This pressure distribution is shown in Figure 4-20. It can be seen that the "clean wing" pressure distribution is more desirable as it carries more lift inboard due to the shock being much farther aft. The selected design variables are all six permutations of products of two spanwise functions given by Equation (C.14) with peaks at y_{root} and $y_{nacelle-centerline}$ amd three chordwise functions given by Equations (C.8), (C.4) and (C.7). Note that the chordwise functions are given by a shear function (which is similar to a twist variable for small geometry perturbations), and two Hicks-Henne functions. Note that the spanwise functions are both zero outboard of a location y_2 which lies halfway between the engine centerline and the planform break. Slope and function continuity in the final geometry is ensured with this spanwise variation albeit at the expense of having zero slope at the maxima of these functions.

The grid perturbation algorithm is more challenging for this case because movement of the wing-body intersection requires nodes lying on the fuselage surface to move also. This is to avoid generation of highly skewed elements or even elements with negative volume. This was done by taking advantage of the circular geometry of the fuselage sections. Points close to the wing/body intersection were moved circumferentially by an amount determined by the product of a function, $g(\theta)$, taken to be linear in θ (the angular coordinate for the circular section associated with the local value of x) and the circumferential movement of the wing/body intersection at that x-location.

$$\Delta\theta(x, \theta(z)) = \Delta\theta_{w/b}(x) g(\theta(z)) \tag{4.1}$$

The linear function $g(\theta)$ is constructed such that it tends to zero at the symmetry plane below and the circumferential coordinate corresponding to the lower limit of the strut which is a constrained obstacle. The chain rule is applied to find grid sensitivities associated with this grid movement. Ideally, a CAD tool should be used to find the new surfaces and intersection curves, along with a scheme to automatically relocate the grid points on these surfaces. This process should be readily differentiable which would retain the capability of calculating exact grid sensitivities. In the interest of expediency, this task has been left as future work.

Sensitivities were calculated using the adjoint method and optimization proceeded using the BFGS method. It should be noted that golden section line searches were used for the 2nd and 3rd design iterations to ensure adequate reduction in the cost function and hence global convergence. This is a common approach for quasi-Newton optimization strategies. Convergence to the minimum was found after 7 iterations. Also, it should be noted that regridding was performed for the modified geometry after 4 design iterations due to excessive stretching of elements at the wing-body intersection. Surface geometry and pressure evolutions are shown in Figures 4-16 and 4-18, respectively. The final pressure distribution is shown in Figure 4-21. Cost function and design variable evolutions are plotted in Figure 4-22. It can be seen that much of the lift found for the "clean wing" case has been recovered. As expected, some of this has been achieved by moving the wing away from the strut-nacelle assembly whose blockage is the source of the high pressure region. Unexpectedly, more leverage on reducing the cost function appears to have been found by cambering the wing and hence increasing the lift. At the minimum, the cost function has decreased by a factor of 43%.

This was the first example performed using parallel versions of flow and adjoint analysis solvers. Various partitioning strategies of the domain were used in this exercise, with the number of domains used varying from 6 to 20.

4.2 Lift-constrained Drag Minimization Examples

Results have been obtained that demonstrate that the descent algorithms described in Section 2.6.1 and 2.6.2 produce plausible optimal geometries in 2D and 3D. In order to determine the most effective non-linearly constrained optimization algorithm for the ultimate goal, the 3D problem, a series of tests were performed in 2D, including a variety of single- and double-point, lift-constrained drag minimization problems. We report herein a comparison of two candidate strategies in 2D — the subspace Steepest Descent and BFGS algorithms. These tests resulted in credible optimal geometries in spite of the coarseness of the 2D grids. The most appropriate algorithm was then selected to perform the 3D optimization exercise.

4.2.1 2D Two-point, Single-element Airfoil

The two 2D test cases both solve the same double-point, lift-constrained drag minimization problem. They minimize the average drag coefficient — i.e. $\omega_1 = 0.5$, $\omega_2 = 0.5$ in Equation (2.24) — for the following two conditions: ($M_{\infty_1} = 0.80$; $C_{l_1}^{targ} = 0.3438$) and ($M_{\infty_2} = 0.82$; $C_{l_2}^{targ} = 0.3348$). The baseline geometry is a NACA0012 airfoil. The grid is not shown but contains very few points (658 points and 1236 elements) for the purpose of fast turnaround time. The first two design variables are the angle of attack at either condition. The four shape design variables were Hicks-Henne functions, which are given by Equation (C.4) for which the exponents p_k are given in Table 4.1. It should

function	definition	x_{peak}/c
h_1	α_1	
h_2	α_2	
h_3	$\sin(\pi(x/c)^{0.7565})$	40%
h_4	$\sin(\pi(x/c)^{1.3569})$	60%
h_5	$\sin(\pi(x/c)^{1.9434})$	70%
h_6	$\sin(\pi(x/c)^{3.1063})$	80%

Table 4.1: Airfoil Parameterization for 2D problem

be noted that all the shape variables have non-zero trailing edge slope. This allows $\Delta\theta_{net}$, the meanline slope change between leading and trailing edges, to vary. This is an important property for the design variable set to possess since it provides a powerful means for changing the net airfoil circulation. This in turn allows the angle of attack variables to vary significantly since the lift lost due to a reduction in α_1, for example, can be compensated by a corresponding increase in $\Delta\theta_{net}$. This provides considerable leverage over the chordwise distribution of lift which is vital for approximating the truly optimal shock location. It also ensures that modern, high aft-loaded, supercritical airfoils fall into the space spanned by the design variable set.

The first optimization strategy tested was the Steepest Descent method. The search direction is simply the negative of the cost function gradient in a scaled design variable space, as discussed in Section 2.6.3. This scaling is performed using

$$\overline{\beta}_{k_i} = g_{k=0_i}\beta_{k_i} \tag{4.2}$$

This causes the gradient components in the transformed space to be initially unity which in turn causes the steepest descent method to be much better behaved, as reported in [12].

At each iteration a line search is performed. Note that rather accurate line searches were performed for this case. The line minimum was assumed to have been reached when the quadratic backtracking algorithm described in Section 2.6.3 resulted in the minimum being bracketed for the first time. However, if the constraint was violated by an amount greater than 0.001, then the line search was stopped and the next line search commenced at the last point with $|c_k| < 0.001$.

The evolution of the shape design variables (pane 1), angle of attack design variables (panes 2-3), cost function (pane 4) and projected gradient (pane 5) are plotted in Figure 4-24. Note that these are the values found at the end of each line search and after projection back into feasible space using Equations (2.79), (2.81) and (2.82).

It can be seen that after five iterations (line searches) little change in the cost function is observed and the design variables also have converged.

The geometry and pressure coefficient evolution are shown in Figures 4-25 to 4-27. Note that a highly aft-loaded supercritical airfoil has been obtained. The rooftop levels have been significantly reduced by moving much of the lift to the aft part of the airfoil. This allows the initial reduction in the shock strength. Subsequent optimization iterations result in a shock-free airfoil at either condition — although it is questionable whether the airfoil would remain shock-free for the grid-converged flow analysis solution. Furthermore, this is not a realistic pressure construction for typical jet transport conditions, since the presence of such a severe trailing edge adverse pressure gradient would surely cause separation if viscous effects were present. (This is, of course, entirely the behaviour we expected and lends weight to the argument that *inviscid* drag minimization exercises at transonic conditions are only of academic benefit or of use for debugging optimization strategies.)

The second optimization strategy tested was the subspace BFGS strategy. The summary plot (Figure 4-28) for this exercise shows that the cost function converges slightly more slowly than the Steepest Descent method. However, since only two function evaluations are required per line search (one Newton step and one constraint correction step), while up to four or five were performed with the Steepest Descent algorithm with the rather accurate line searches, this strategy ends up costing less in terms of floating point operations. The geometry and pressure coefficient evolution shown in Figures 4-29 to 4-31 show similar behaviour to that found for the Steepest Descent exercise. Also note that the final design variable states and, of course, cost function values agree closely for the two exercises.

4.2.2 3D Two-point, Low Aspect Ratio Wing/Body

Based on the superior performance of the BFGS optimization strategy on the 2D case, it was decided to use it for the 3D case.

The 3D case is a double point drag-minimization exercise with $\omega_1 = 0.5$ and $\omega_2 = 0.5$. The two flight conditions are $(M_{\infty_1} = 0.9, C_{L_1} = 0.450)$ and $(M_{\infty_2} = 1.6, C_{L_2} = 0.125)$. The baseline geometry is a wing-body configuration with an area-ruled body. The wing has a low aspect ratio ($AR = 2.67$), with leading edge sweep, $\Lambda_{LE} = 45°$, trailing edge sweep, $\Lambda_{TE} = 0°$. It has uniform airfoil maximum thickness of 5% of chord, has uniform twist of $\phi(y) = 0°$ and is uniformly uncambered across the span. The baseline grid is shown in Figure 4-32. It contains 1.13 million elements and 211,000 points. An optimization exercise based on the same wing and forward fuselage and freestream conditions but with different design variables has been reported in [28]

The baseline transonic solution ($M_{\infty_1} = 0.9$) is shown in Figure 4-33. The sectional pressures are also plotted in Figure 4-38. To achieve the required C_{L_1} with the untwisted, uncambered wing, quite a high angle of attack is required. Combined with the sharp leading edge, this results in a region of high suction at the leading edge terminated by a strong shock. However, this shock does not extend far into the flowfield compared to the aft shock. The leading edge shock is highly swept with sweep angle even higher than that of the planform leading edge.

Because of the obliqueness of the shock, the flow region aft of the leading edge shock remains supersonic. This region ends in an almost unswept aft shock which is quite strong and extends between quarter and half a chord into the flowfield. It is therefore the source of most of the transonic wave drag. Inboard (around $\eta = 35\%$), the aft shock is located at about $x/c = 65\%$, while outboard (around $\eta = 80\%$), it is located at about $x/c = 50\%$. The leading edge and aft shocks coalesce at around $\eta = 90\%$ to form a single strong shock suggesting possible off-design problems. The relatively far forward position of the shock invites the possibility of significant drag improvement through aft movement of the shock in the optimization process. A very similar transonic solution was found in [28] with shock structure very close and with the attack angle α required to achieve the target lift distribution agreeing to two significant digits. However, the baseline drag was off by a factor of three partially due to the quite large amount of artificial dissipation used in the present Scheme I (and possibly partially due to the different aft fuselages used).

The baseline supersonic solution ($M_{\infty_2} = 1.6$) is shown in Figure 4-34. Examination of Mach sectional contour plots (not shown) across the span reveals quite strong trailing edge shocks but very weak leading edge shocks. The cause of the latter phenomenon is found by comparison of the

Mach angle for the supersonic case $\mu = sin^{-1}(\frac{1}{M_\infty}) = 38.7^\circ$ with the leading edge sweep angle. Therefore, based on infinite swept wing theory, one would expect the oncoming flow component normal to the leading edge to be subsonic, and so, if even present, leading edge shocks should be weak. The supersonic solution appears to also agree with that reported in [28] since, once again, the attack angle α required to achieve the target lift agrees to within two significant digits.

The optimization was performed using twelve design variables. Two of these were α_1 and α_2, the angles of attack at each condition. The other ten were combinations of three chordwise functions and four spanwise functions. The chordwise functions, $f_i(x/c)$ — based on Equations (C.4) and (C.8) — and spanwise functions, $e_j(y)$ — based on Equation (C.14) — are summarized in Table 4.2. The construction of the actual design variables from **e** and **f** is also shown in Table 4.2. Only the twist design variable was used at the tip since the other two exhibited strange initial behaviour, and it was decided that the prudent path to take would be to initially exclude these design variables which might be affected by the high gradients in the region of the wing tip.

func	definition	range
f_1	$\sin(\pi(x/c)^{0.7565})$	$0 < x/c < 1$
f_2	$\sin(\pi(x/c)^{1.9434})$	$0 < x/c < 1$
f_3	x/c	$0 < x/c < 1$
e_1	$\sin^2(\frac{\pi}{2}(y-y_0)/(y_1-y_0))$	$y_0 < y < y_1$
	$\cos^2(\frac{\pi}{2}(y-y_1)/(y_2-y_1))$	$y_1 < y < y_2$
e_2	$\sin^2(\frac{\pi}{2}(y-y_1)/(y_2-y_1))$	$y_1 < y < y_2$
	$\cos^2(\frac{\pi}{2}(y-y_2)/(y_3-y_2))$	$y_2 < y < y_3$
e_3	$\sin^2(\frac{\pi}{2}(y-y_2)/(y_3-y_2))$	$y_2 < y < y_3$
	$\cos^2(\frac{\pi}{2}(y-y_3)/(y_4-y_3))$	$y_3 < y < y_4$
e_4	$\sin^2(\frac{\pi}{2}(y-y_3)/(y_4-y_3))$	$y_3 < y < y_4$

var	h_1	h_1	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}
defn	f_1e_1	f_2e_1	f_3e_1	f_1e_2	f_2e_2	f_3e_2	f_1e_3	f_2e_3	f_3e_3	f_1e_4

Table 4.2: Surface Parameterization for wing-body problem

It should be pointed out that the spanwise functions ensure slope continuity, but at the expense of requiring zero spanwise gradient at the maxima of those functions. Also it should be noted that as for the 2D cases, the chordwise functions all have non-zero trailing edge slope and are associated with quite high values of C_L sensitivities and therefore allow close tailoring of the chordwise variation of lift and therefore the shock position.

Scaling of the design variables was not as straightforward here as in the 2D case. The source of the problem was that the two flight conditions were on either side of the $M_\infty = 1.0$ point. As a result, the sensitivities to C_D and C_L associated with some of the design variables were of the opposite sign for either flight condition. Hence, direct use of Equation (4.2) resulted in movement in the wrong direction for some design variables. Replacement of Equation (4.2) with an expression like $\overline{\beta}_i = |g_{0_i}|\beta_i$ solves some of the problems. However, some other problems are still present. For example, if the drag sensitivities associated with a particular design variable for the two conditions were of the opposite sign but approximately equal in magnitude then, $g_i \ll \partial C_{D_1}/\partial\beta_i$ and $g_i \ll \partial C_{D_2}/\partial\beta_i$ resulting in the search direction being dominated by these design variables. It was decided that the best approach was to normalize design variables associated with each of the three chordwise functions by their counterparts at the $\eta = 38\%$ span station. Better behaviour was expected because at $\eta = 38\%$, the 3 design variables tended to have associated sensitivities whose magnitude and sign were representative of the behaviour across the wing. Also the magnitude of the cost function sensitivity at $\eta = 38\%$ is on the order of either drag sensitivity ($|g_i| \approx |\partial C_{D_j}/\partial\beta_i|$)

As usual for the BFGS algorithm, the first line search proceeded in an identical fashion to the Steepest Descent method. The above choice of normalization resulted in a quite reasonable shape for the wing, lending credence to the choice of design variables. The optimization process was thereafter continued as in 2D with just two iterations per line search, that is one Newton step followed by one constraint correction step. One exception should be mentioned. During the fourth line search, severe problems were encountered in obtaining a converged solution for both flow and adjoint analyses. Close examination of the grid in the region of the largest surface movement revealed that some of the tetrahedra were severely compressed on the upper surface and severely stretched on the lower surface resulting in very high aspect ratio elements. The net movement upward was because nose down twisting movement (balanced by positive cambering movements) were demanded by the optimization process. Regridding (based on a new surface definition perturbed by the current net design variable movements) resulted in a return to reliable robust convergence behaviour in flow and adjoint analyses and an almost seamless continuation of the optimization process.

Final C_p distributions are shown in Figures 4-35 and 4-36 for either condition while sectional geometry and C_p evolution are plotted in Figures 4-38 to 4-40. A summary of the optimization exercise is shown in Figure 4-37. After seven line searches, a 5.14% reduction in the cost function is found.

The cost function reduction breakdown is summarized in Table 4.3. It can be seen that all of the improvement has come from an 8.14% reduction in drag at condition 1, with the drag increase found at the supersonic condition merely acting as a penalty on the improvements found at the transonic condition.

	Condition 1	Condition 2	Total
M_∞	0.9	1.6	
C_L	0.450	0.125	
α_0	6.338°	2.565°	
α_7	5.628°	3.115°	
C_{D_0}	0.1290	0.0432	
C_{D_7}	0.1185	0.0447	
$\Delta C_D/C_{D_0}$	-8.14%	$+3.47\%$	
$\Delta\omega_i C_D/F_0$	-6.10%	$+0.87\%$	-5.14%

Table 4.3: Summary of wing-body design changes

The source of the drag improvement at condition 1 can be found by examination of the transonic solution based on the final geometry as shown in Figures 4-35 and 4-38. The first thing to note is that the shock has moved back from $x/c \approx 55\%$ to $x/c \approx 85\%$. This allows less lift to be required at the leading edge and so the region of highly expanded supersonic flow there has almost vanished. Furthermore, the aft shock has acquired a lower sweep angle which is closer to that of the local planform x/c lines. This results in lower wave drag (and better off-design performance). Examination of the sectional pressure plots in Figure 4-38 reveals that the aft shock upstream Mach number levels are far lower outboard than the baseline although slight increases are found inboard. This also, is a cause for the lower wave drag. However, there is also a new weak leading edge shock with negative sweep. This only extends over a region very close to the wing tip.

This favourable movement of the aft shock has been achieved by net design variable movements that are similar to those found in the 2D case. The angle of attack at condition 1, α_1 has decreased while the twist design variables have all increased, resulting in lower incidence angle, as in 2D. The net $x/c = 70\%$ camber variable changes have been positive as in 2D resulting in airfoils with the expected highly aft-loaded, supercritical characteristics. However, the net change in the $x/c = 40\%$ camber design variable components here have been positive whereas the corresponding (single) component change was negative in 2D. This is possibly due to the different impact of the supersonic second condition in 3D compared to a far different coupling of a transonic second condition in 2D. As in 2D, the need for inclusion of viscous effects is apparent from these results: the strong adverse pressure gradients would likely cause boundary layer separation.

Although detailed breakdowns of the drag contributions have not been performed, examination of the spanload for the baseline and optimized geometries revealed that the baseline actually appeared to have closer to an elliptic distribution. However, the difference was marginal. Nevertheless, this suggests that as other researchers have found, most of the optimization benefit has come from wave drag reduction. Of course this may be coincidental, and other exercises, may reveal that induced drag reduction can have a larger impact.

The source of the supersonic C_D increase is evident after examination of Figure 4-34, 4-36 and Figure 4-40. Comparison of baseline and designed pressures reveals higher levels of expansion at the aft end of the upper surface on the outboard part of the wing for the latter configuration causing a stronger trailing edge shock. However, the resultant C_D increase is evidently far lower than the decrease found at Condition 1 ($M_\infty = 0.9$).

The optimization exercise reported in [28] found a rather larger 140 count C_D reduction for the transonic case. This is partially due to the different set of design variables which are local in nature and appear to allow elimination of the shock albeit at the cost of a point-designed geometry with high chordwise curvature — and likely severe off-design problems.

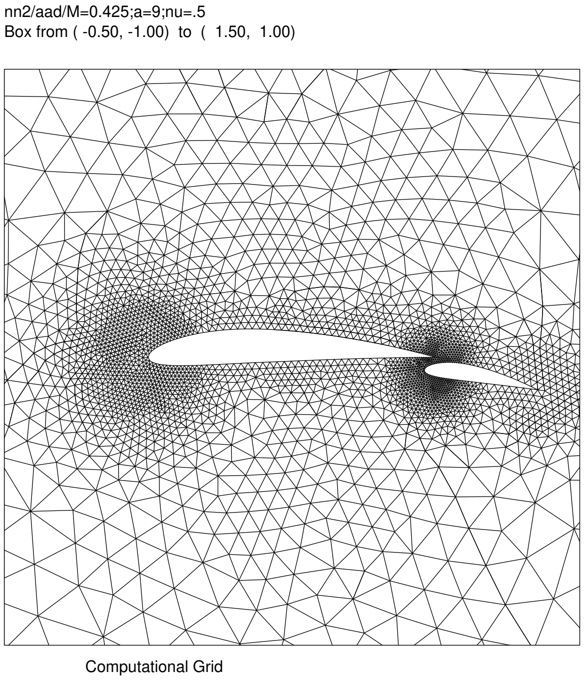

Figure 4-1: Grid used for seed 2D multielement airfoil calculation

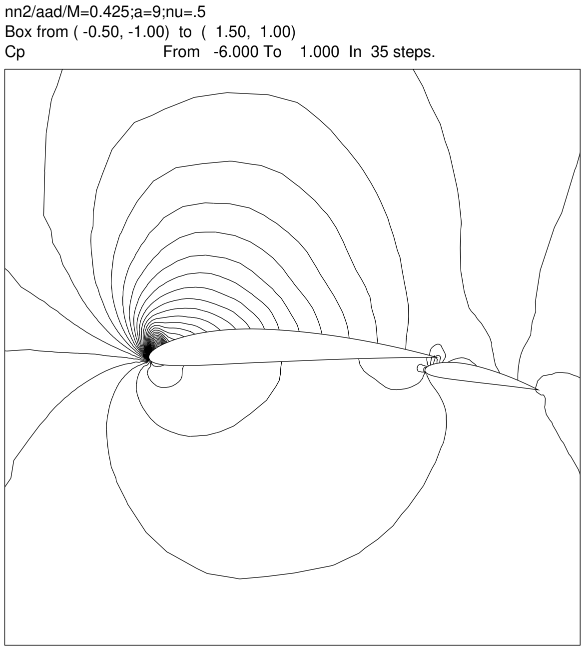

Figure 4-3: Pressure contours for 2D multielement airfoil at M=0.425, $\alpha = 9^{\circ}$

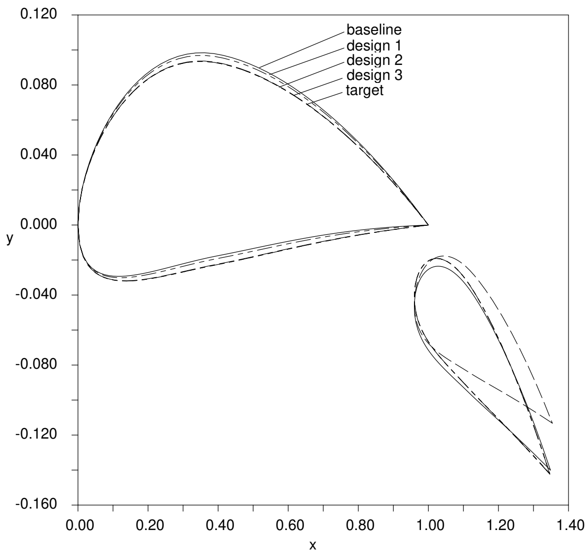

Figure 4-2: Geometry evolution for 2D multi element optimization exercise

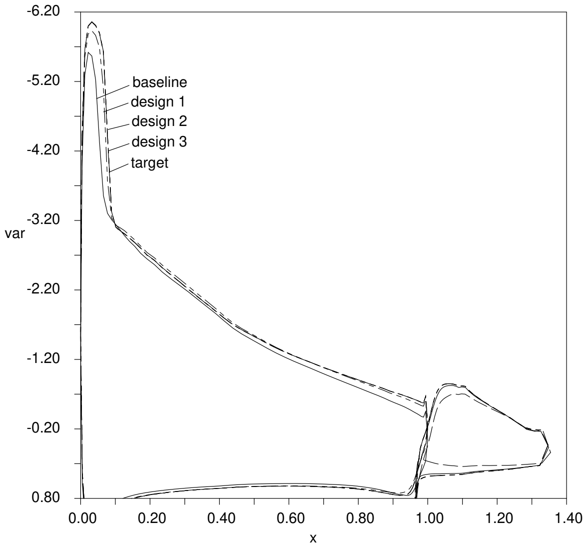

Figure 4-4: Pressure evolution for 2D multi element optimization exercise

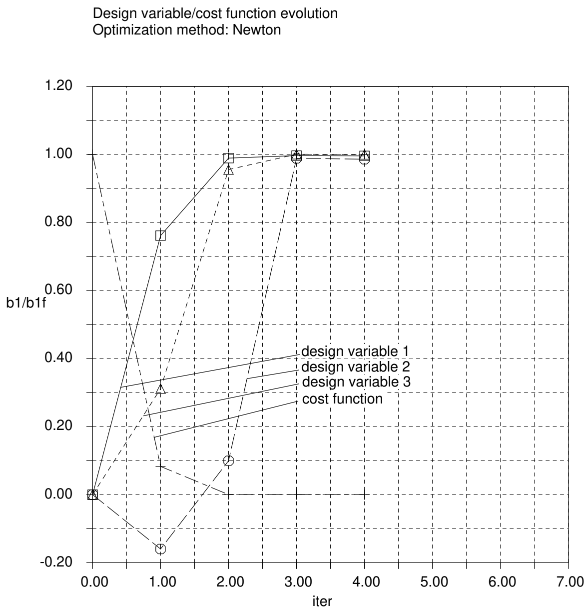

Figure 4-5: Cost function and design variable evolution for 2D multi element optimization exercise

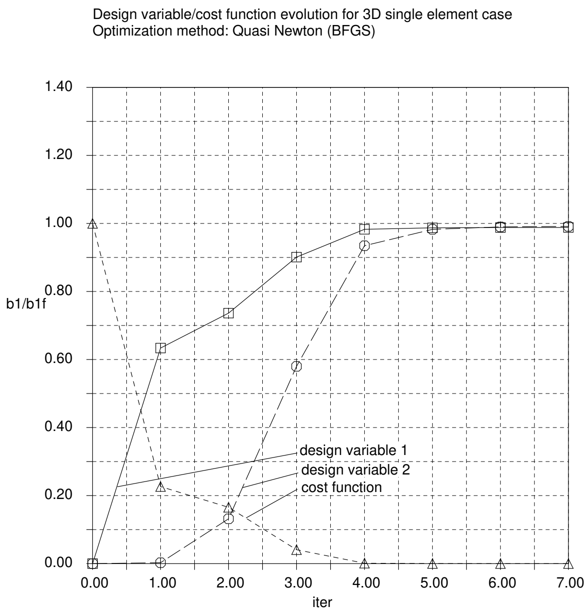

Figure 4-6: Cost function and design variable evolution for 3D infinite wing optimization exercise

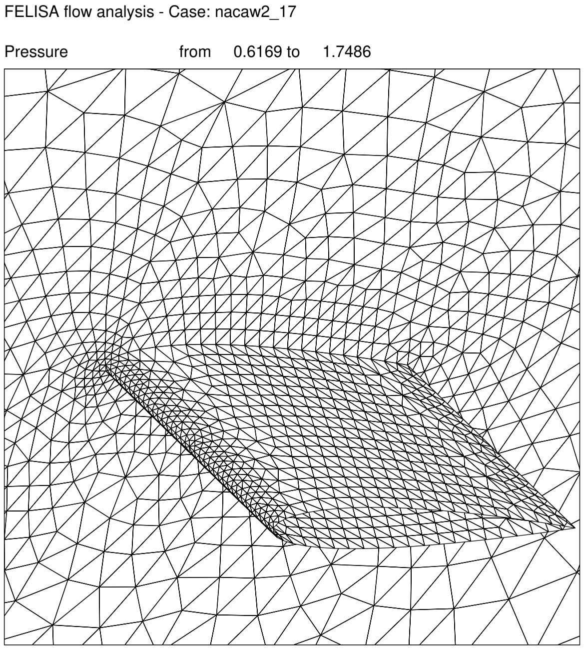

Figure 4-7: Grid used for baseline 3D infinite wing calculation

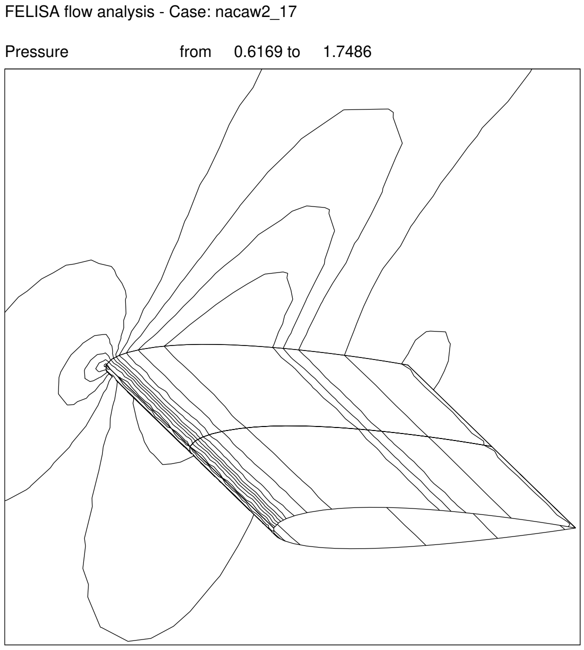

Figure 4-9: Pressure contours for 3D infinite wing at M=0.8, aoa=1.25deg

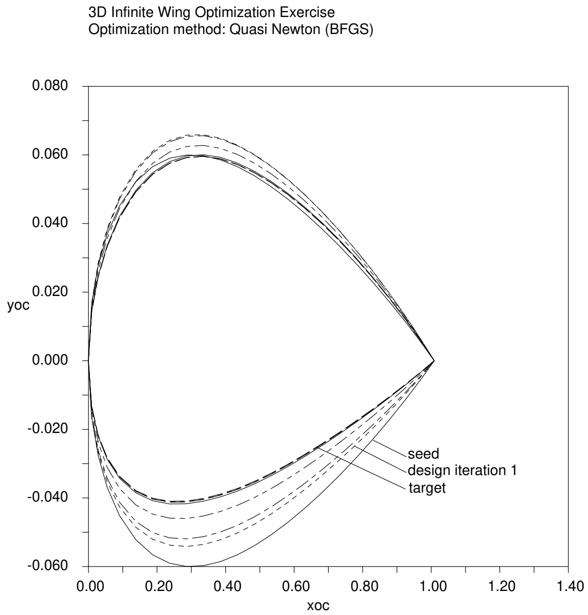

Figure 4-8: Geometry evolution for infinite wing optimization exercise

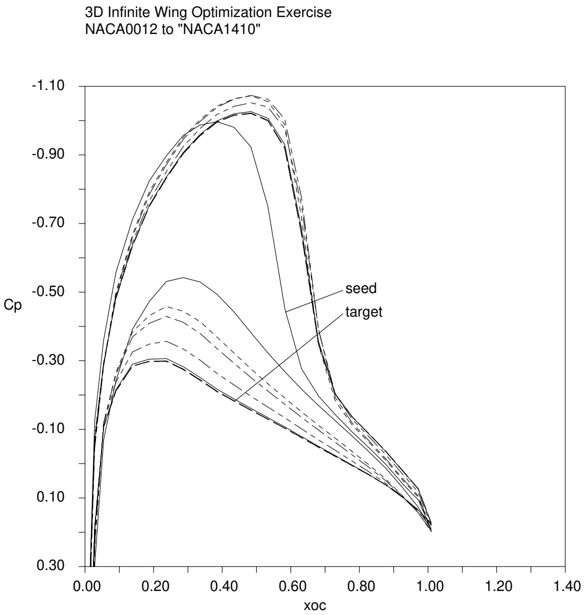

Figure 4-10: Pressure evolution for 3D infinite wing optimization exercise

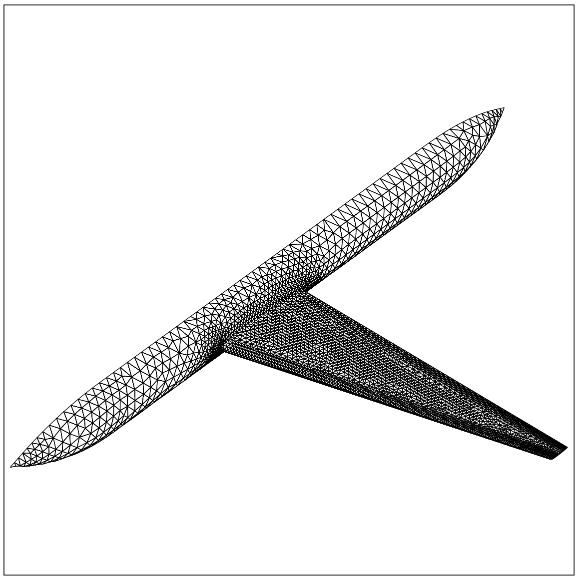

Figure 4-11: Surface grid for baseline 3D wing/body calculation

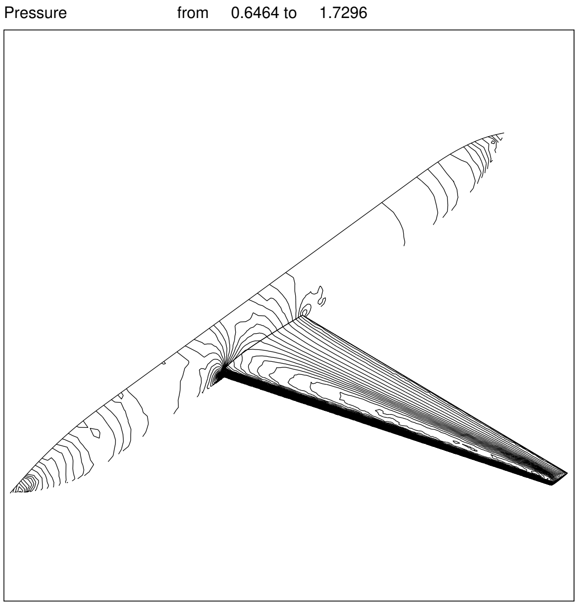

Figure 4-13: Surface pressure contours for 3D wing/body at M=0.8, $\alpha = 1.25°$

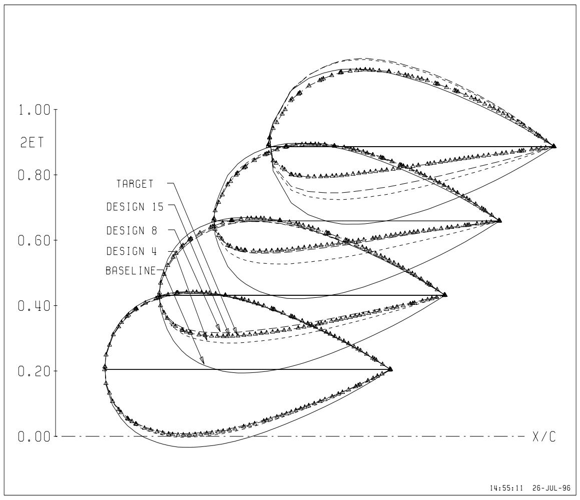

Figure 4-12: Geometry evolution for wing/body optimization exercise

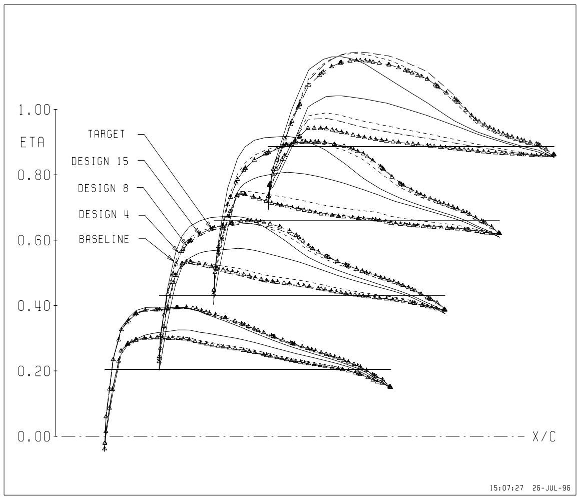

Figure 4-14: Pressure evolution for wing/body optimization exercise

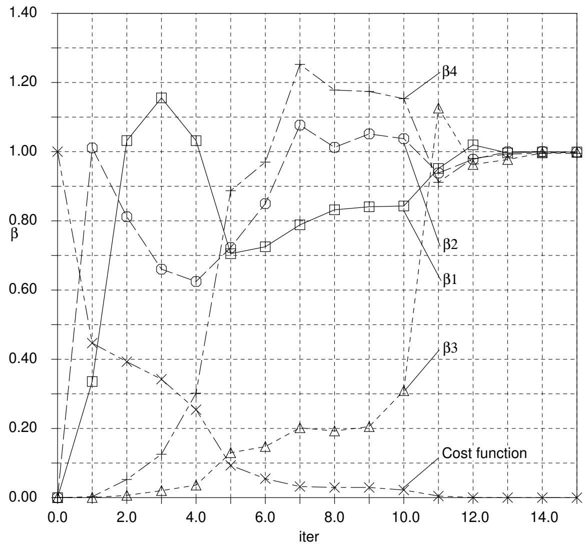

Figure 4-15: Cost function and design variable evolution for wing/body optimization exercise

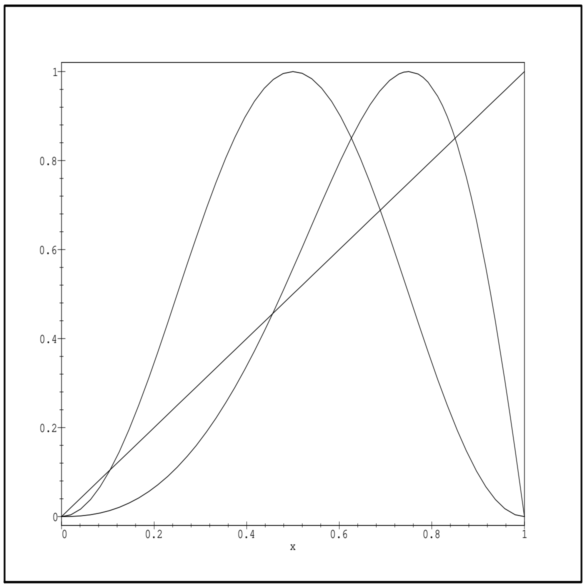

Figure 4-17: Chordwise components of design variables for 2nd wing/body optimization exervise

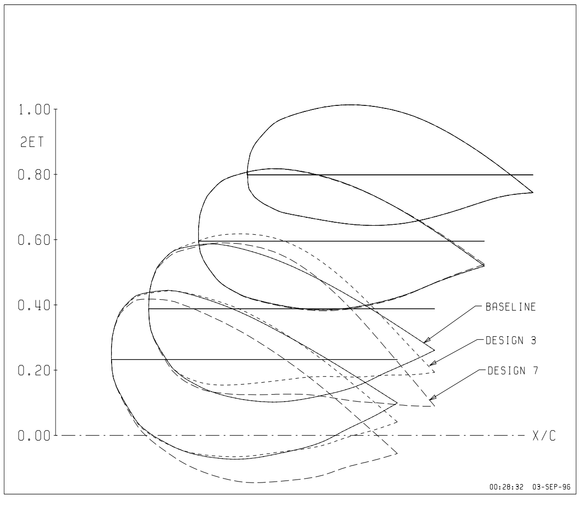

Figure 4-16: Geometry evolution for business jet optimization exercise

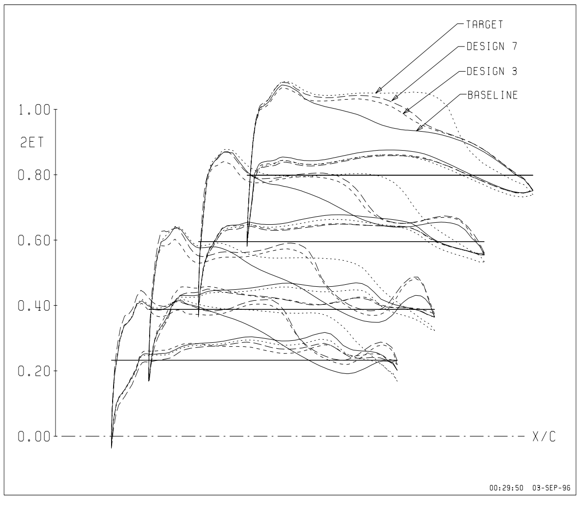

Figure 4-18: Pressure evolution for business jet optimization exercise

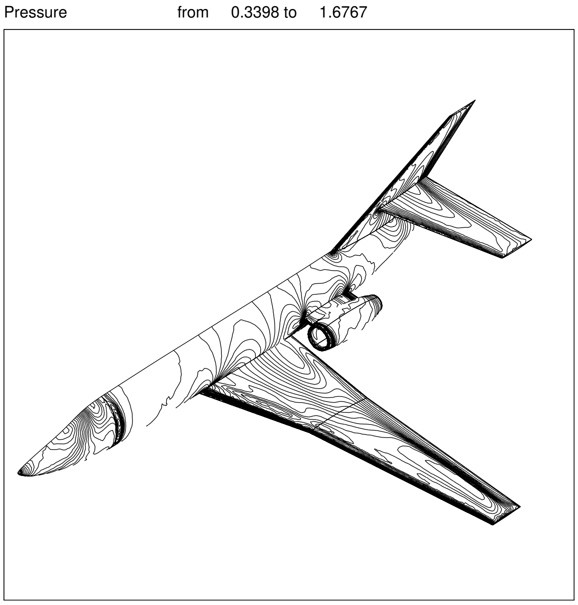

Figure 4-19: Baseline surface pressure contours for business jet at M=0.85, $\alpha = 2°$

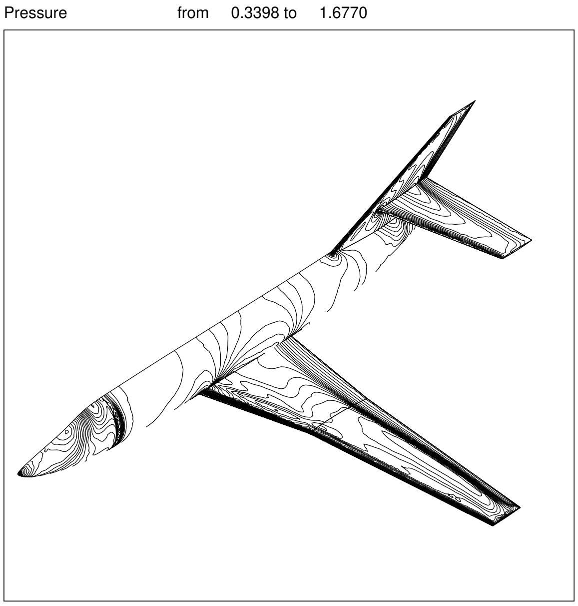

Figure 4-20: Target surface pressure contours for business jet at M=0.85, $\alpha = 2°$

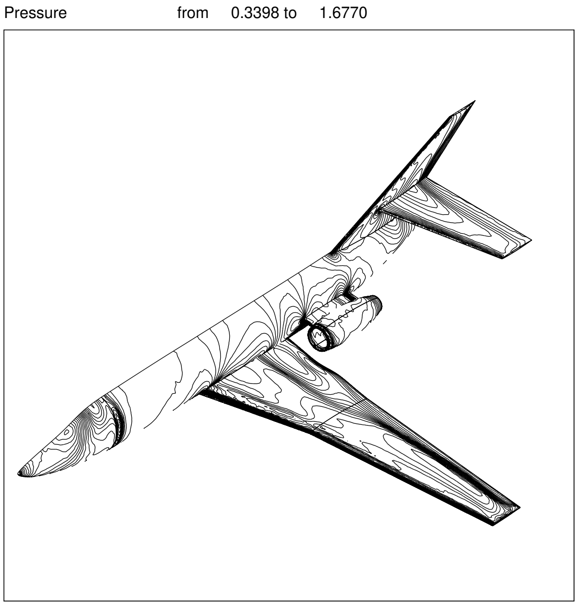

Figure 4-21: Surface pressure contours for business jet at M=0.85, $\alpha = 2°$ after 7 iterations

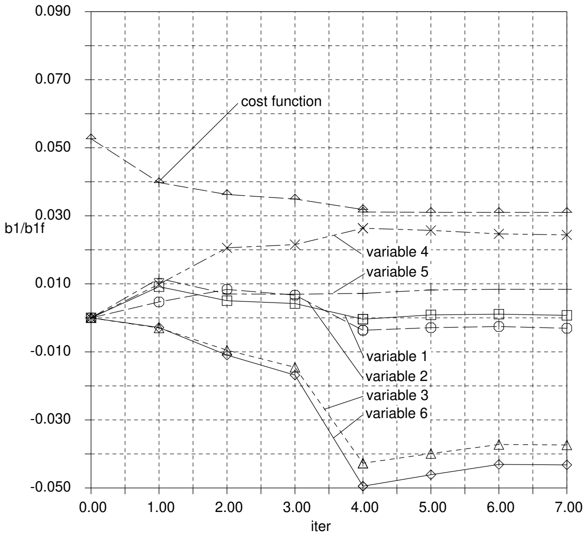

Figure 4-22: Cost function and design variable evolution for business jet optimization exercise

Figure 4-23: Surface grid for baseline business jet calculation

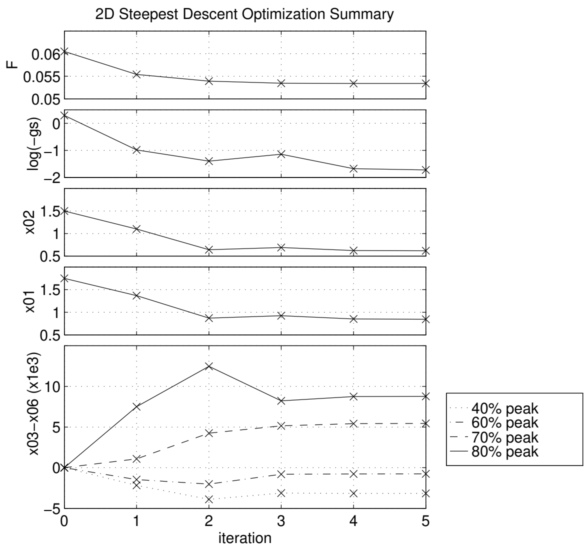

Figure 4-24: Cost function and design variable evolution for 2D steepest descent optimization exercise

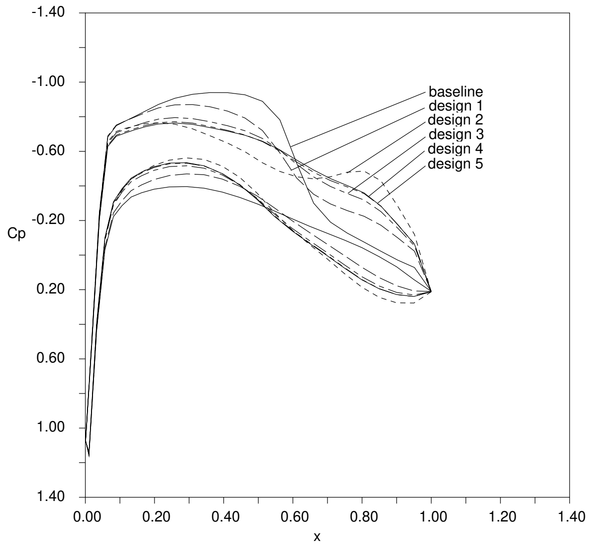

Figure 4-26: Condition 1 C_p evolution for 2D steepest descent optimization exercise

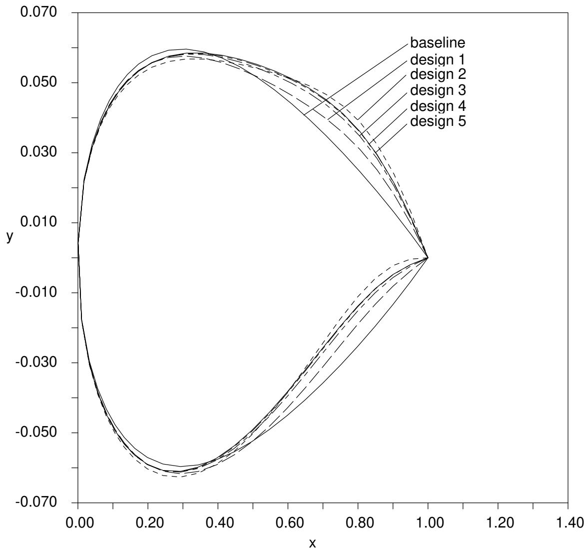

Figure 4-25: Geometry evolution for 2D steepest descent optimization exercise

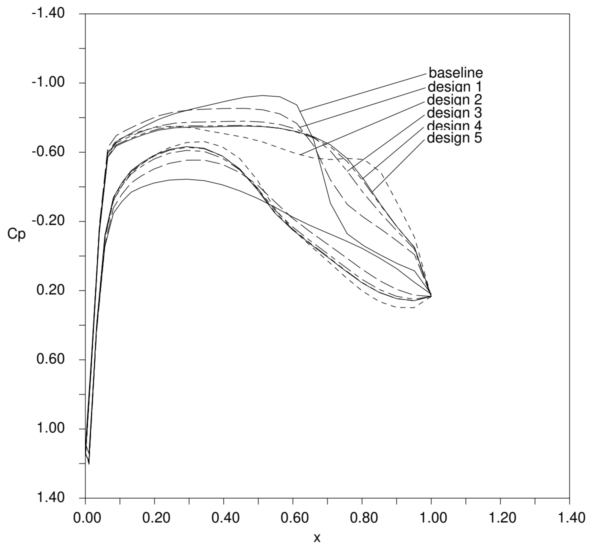

Figure 4-27: Condition 2 C_p evolution for 2D steepest descent optimization exercise

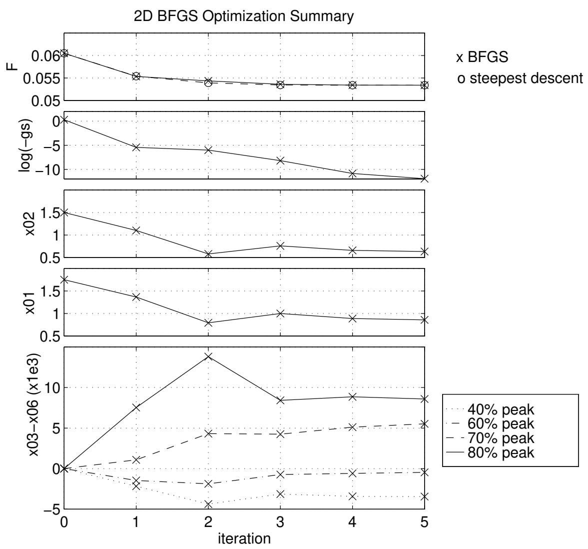

Figure 4-28: Cost function and design variable evolution for 2D BFGS optimization exercise

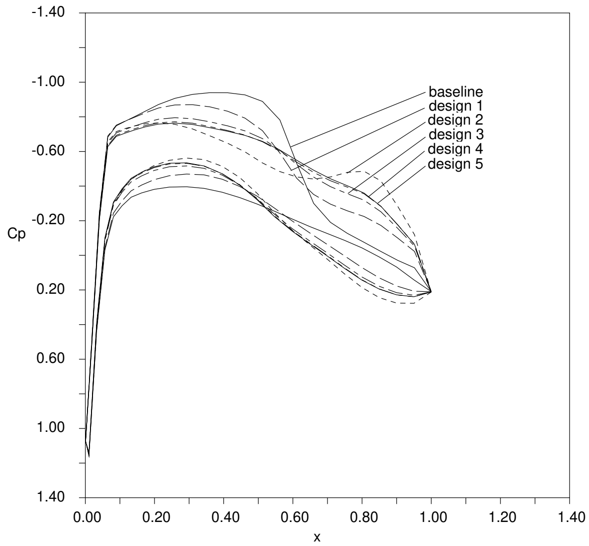

Figure 4-30: Condition 1 C_p evolution for 2D BFGS optimization exercise

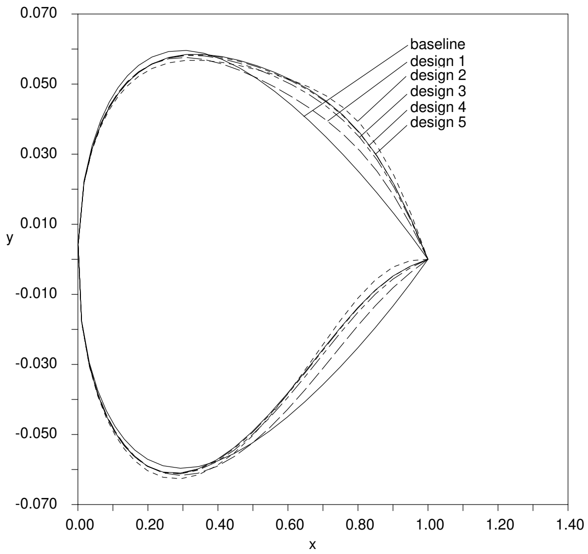

Figure 4-29: Geometry evolution for 2D BFGS optimization exercise

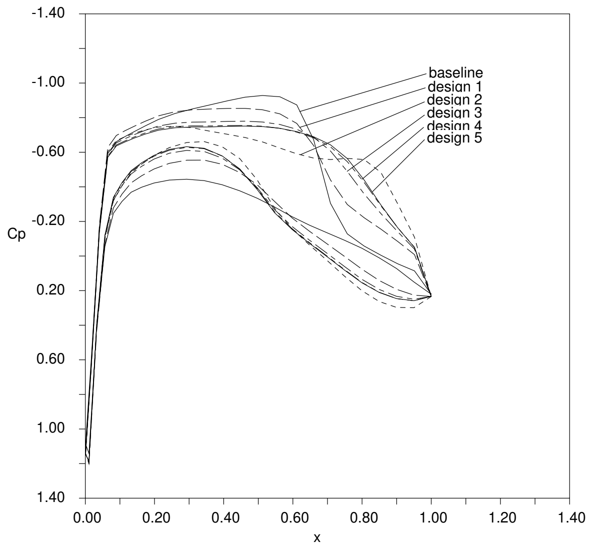

Figure 4-31: Condition 2 C_p evolution for 2D BFGS optimization exercise

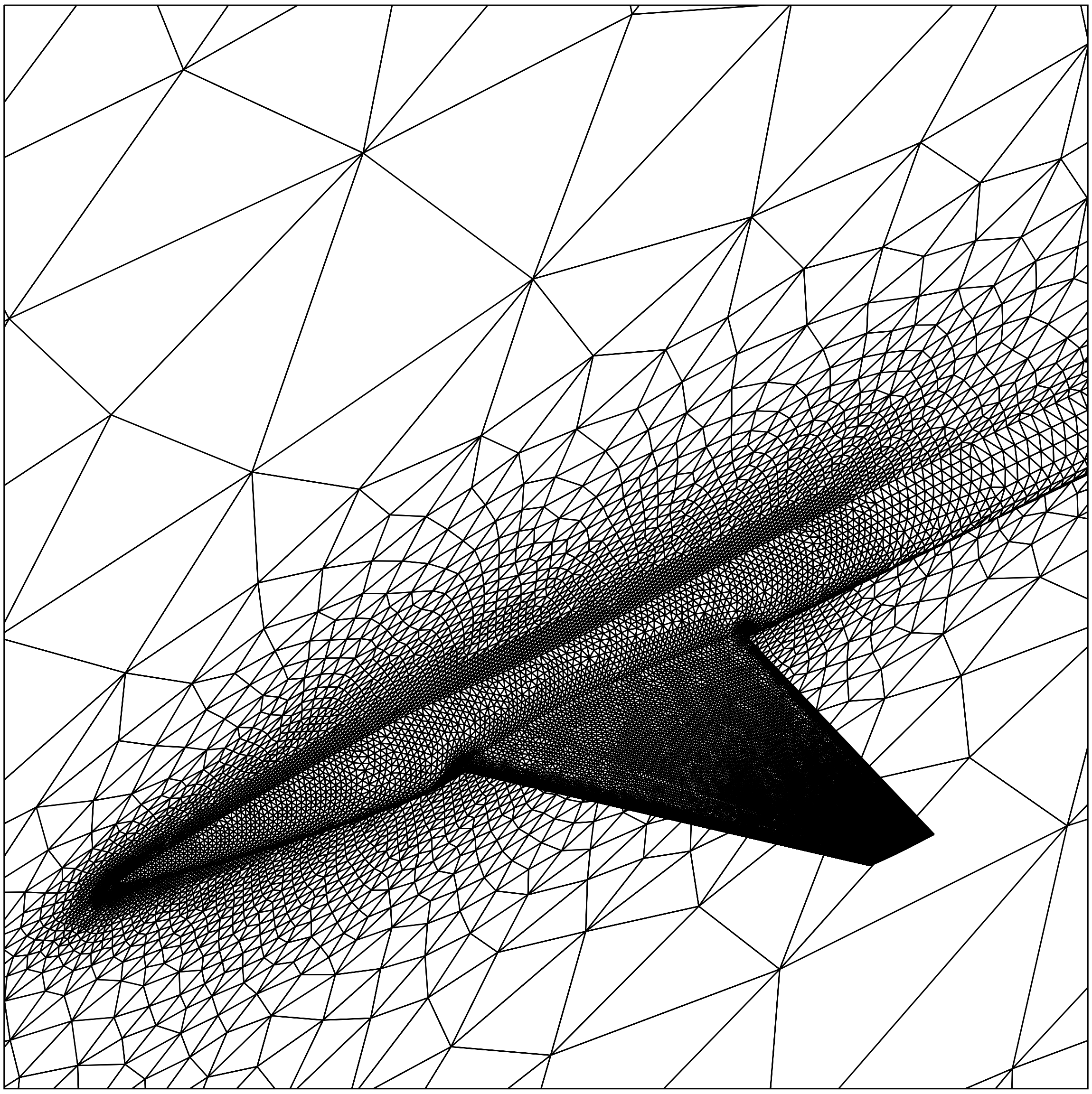

Figure 4-32: Baseline grid for 3D optimization exercise

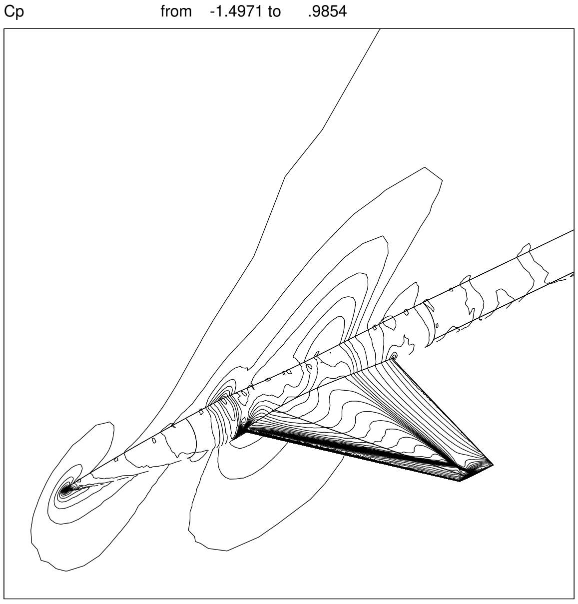

Figure 4-33: Condition 1 (M=0.9) baseline C_p distribution for 3D optimization exercise

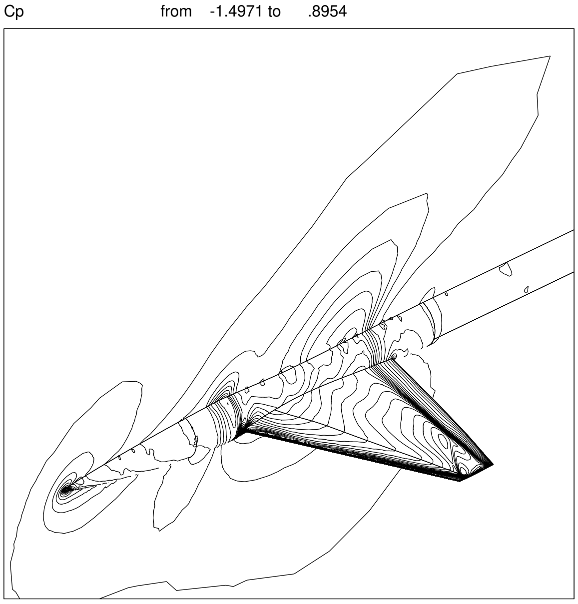

Figure 4-35: Condition 1 (M=0.9) "Design 7" C_p distribution for 3D optimization exercise

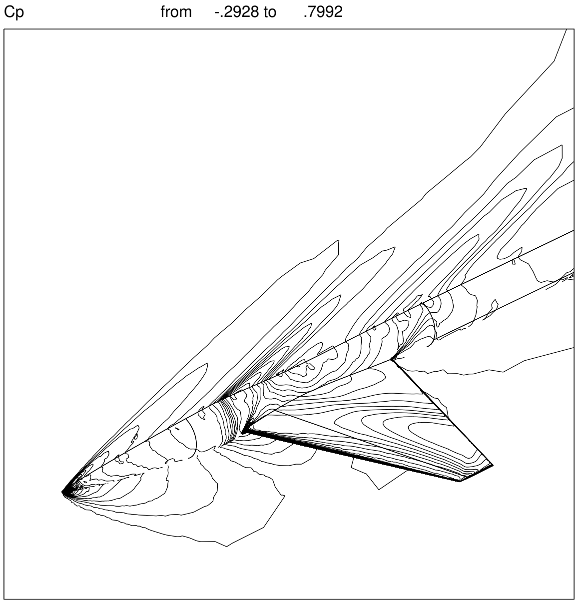

Figure 4-34: Condition 2 (M=1.6) baseline C_p distribution for 3D optimization exercise

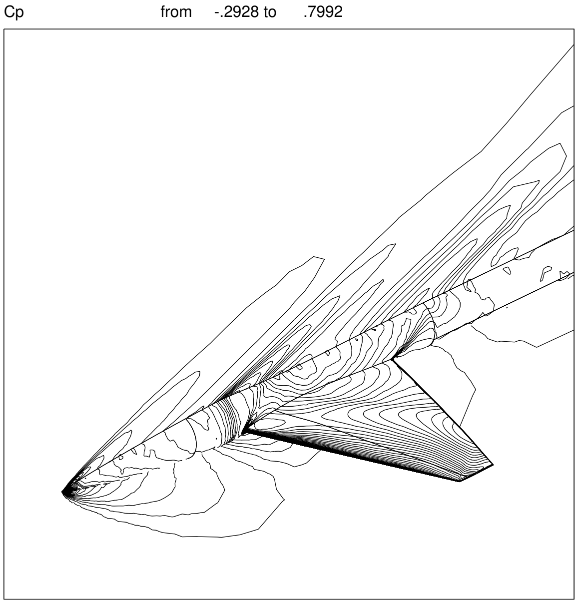

Figure 4-36: Condition 2 (M=1.6) "Design 7" C_p distribution for 3D optimization exercise

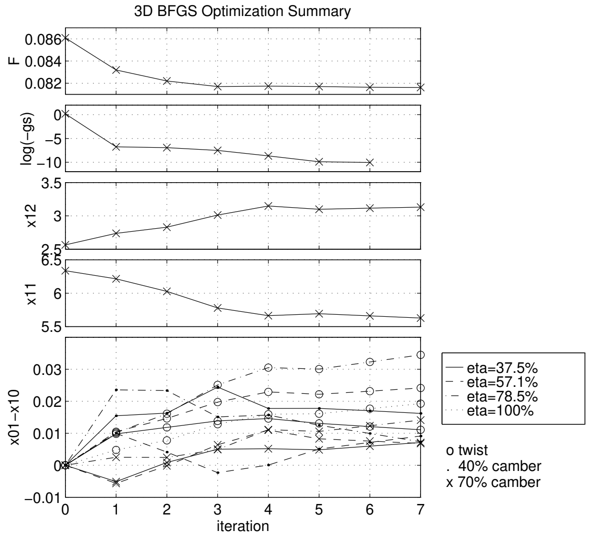

Figure 4-37: Cost function and design variable evolution for 3D BFGS optimization exercise

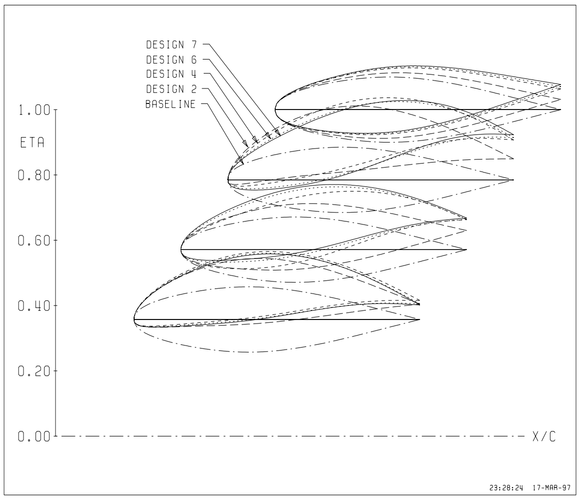

Figure 4-39: Geometry evolution for 3D BFGS optimization exercise

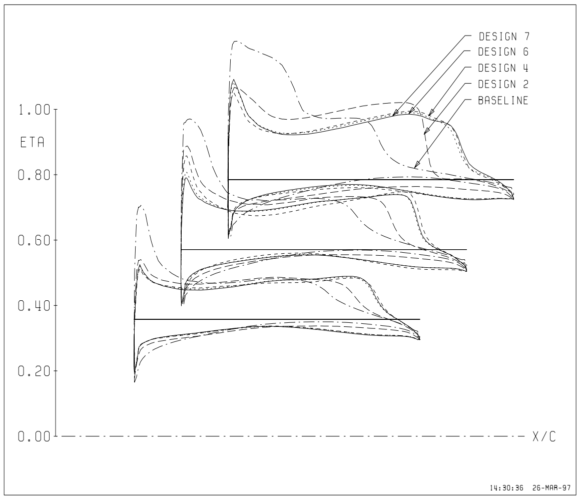

Figure 4-38: Condition 1 (M=0.9) C_p evolution for 3D optimization exercise

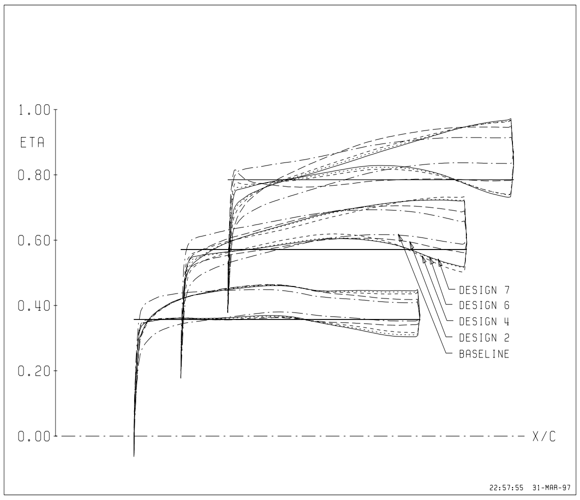

Figure 4-40: Condition 2 (M=1.6) C_p evolution for 3D optimization exercise

Chapter 5

Viscous Results

5.1 Lift-constrained Drag Minimization Examples

5.1.1 2D Attached Airfoil

Scheme II was used for flow analysis for this problem. The baseline geometry was a very thin airfoil (with about 1% maximum thickness), with a thickness distribution that had already undergone considerable design and optimization for low Reynolds number viscous flow, albeit at a slightly higher Re=10000. The original airfoil is under consideration for use in micro unmanned aerial vehicles (μ-UAVs). It has significant camber, but it was decided that a well behaved zero-camber airfoil was needed as an initial test for the optimization system, so the camber was removed from the airfoil by a simple geometric operation. At $\alpha = 3.5°$, $M = 0.8$, and $Re = 2000$ it was found that the boundary layer C_f distribution was well above zero everywhere. The high Mach number was chosen, not for its realism in representing the flight conditions of μ-UAVs, but because it allows quicker convergence of the flow analysis calculations. It was decided that drag minimization with a lift constraint was an appropriate initial test, even though not much of a drag reduction was expected. The design variables were chosen to be the modal amplitudes of Hicks-Henne camber functions given by Equations (C.4) and (C.5) with p_k chosen to be such that peaks occur at 20%, 40%, 60% and 80% of chord. In addition, the angle of attack was chosen as a design variable.

The baseline grid, pressure distribution and Mach distribution are shown in Figures 5-1 to 5-3. The baseline grid contains 8506 nodes and at the trailing edge there are approximately 20 points in the boundary layer. However for this case, the elements with "structure" in the boundary layer number about 16, which is the probable cause of the reduced presence of wiggles in the solution compared to the NACA0012 solution described in the previous section. In spite of the low level error associated with these wiggles, the surface C_p distribution was found to match that found by MSES even more closely than the match shown in Figures 3-11 and 3-12. Note that although there is locally supersonic flow, no shocks are present, although the ability to capture these shocks is an inherent part of the flow analysis algorithm.

The constrained BFGS algorithm described in Section 2.6.3 was used to minimize C_d at the baseline C_l. The final C_p distribution is shown in Figure 5-4 while the evolution of design variables, C_f distribution, C_p distribution and geometry are shown in Figures 5-5 to 5-8. It can be seen that the minimum appears to have been almost reached after about 4 line searches which is about what should be expected for a well behaved design surface. The cost function has only decreased by 18 counts or about 2.2% of the baseline C_d. This modest decrease was expected since the baseline solution exhibited quite healthy boundary layer behaviour. Note that the decrease has come about partially by a reduction of the leading edge pressure peak. The resulting reduction in the net pressure increase and adverse pressure gradient along the upper surface causes the momentum thickness to be lower at the upper surface trailing edge, indicating a more healthy boundary layer and lower drag.

Also it should be noted that this optimization exercise was reproduced using MSES and and LINDOP, its associated optimization driver [12]. A similar reduction in C_d was observed, although the final geometry was slightly different. It is believed that this is due to the relatively shallow minimum and obvious differences between the flow analysis algorithms.

5.1.2 2D Separated Airfoil

Scheme II is also used for flow analysis for this problem. This case is a more challenging one due to the presence of separation over the aft 55% of the upper surface. However, more of a drag reduction is expected due to the poor initial health of the upper surface boundary layer. It was felt that thickness design variables would possess powerful leverage over setting the separation point on the upper surface, so two thickness design variables with peaks at 20% and 40% were chosen as well as one camber design variable with a peak at 50%, along with the angle of attack. In addition to the lift constraint, for this case, it was decided that an area constraint should also be imposed. Otherwise, the airfoil would be driven uselessly to zero thickness. The baseline grid, C_p and M distributions are shown in Figures 5-9 to 5-11.

Once again, the constrained BFGS method was used to minimize C_d and the baseline C_l. It should be noted that this optimization exercise did not proceed nearly as smoothly as the one discussed in the previous paragraph. The presence of the separated flow seemed to cause two major undesirable effects. Firstly, as soon as the flow separates on both lower and upper surface, the flow becomes unsteady due to the onset of vortex shedding, and no fully converged flow or adjoint solution can be found. The way this undesirable feature was dealt with in the optimization algorithm was to assume a very large value of the objective function at that point and thereafter to trace back along the line search direction until a steady solution could be found, and then to search for a new direction from there. The second undesirable effect was that the presence of separation caused larger deviations from quadratic behaviour than usual. This appears to have slowed down the convergence of the approximate Hessian to the value found at the minimum. It is not clear that a good approximation has been made for the final design point depicted. The optimization process was stopped after 3 line searches when successive iterations resulted in the search direction not changing significantly with the design poised to enter the part of design space in which the above-mentioned vortex-shedding unsteadiness occurs.

The final C_p distribution is shown in Figure 5-12 while the evolution of design variables, C_f distribution, C_p distribution and geometry are shown in Figures 5-13 to 5-16. The cost function for this case has decreased a far larger 120 counts or about 10.6% of the baseline C_d. This larger decrease was expected due to the presence of separation in the baseline solution. Note that the decrease has been accomplished partially by a reduction of the leading edge pressure peak. Also the thickness distribution has been redistributed such that the maximum is about 15% further aft. Like the early natural laminar flow airfoils [49], this delays the start of the adverse pressure gradient to aft of that maximum thickness point. Consequently the upper surface separation point moves from 45% to about 65%. This movement appears to be limited by the appearance of separation on the lower surface and the resulting above-mentioned unsteadiness as can be seen in Figure 5-14.

Agreement with a similar optimization exercise performed using LINDOP and MSES revealed similar initial behaviour although the LINDOP geometry evolved to one with negative camber – a configuration not allowed by the current Navier-Stokes optimizer due to the intervening onset of vortex shedding as the maximum camber passes through zero.

5.1.3 3D ONERA M6 Wing

Scheme II was used for flow analysis for this problem. The baseline geometry was the ONERA M6 wing. This wing has no camber and is untwisted. The nondimensional airfoil sections that the wing is lofted from are uniform across the span and have quite a large maximum thickness of $t/c_{max} = 9.7\%$. Therefore some separation is likely and indeed is found on the upper surface at the baseline conditions of $\alpha = 3.5°$, $M = 0.8$, and $Re = 1600$ based on the root chord. However it is limited to the aft 10%-20% of the airfoil and therefore the design space was expected to be rather smooth. It was decided that drag minimization with a lift constraint was an appropriate test, even though, once again, not much of a drag reduction was expected. The design variables were combinations of two chordwise functions and five spanwise functions. The chordwise functions, $f_i(x/c)$ — based on Equations (C.4) and (C.8) — and spanwise functions, $e_j(y)$ — based on Equation (C.13) — are summarized in Table 5.1. In addition, the angle of attack was chosen as a design variable.

func	definition	range
f_1	$\sin(\pi(x/c))$	$0 < x/c < 1$
f_2	x/c	$0 < x/c < 1$
e_1	$(y_1 - y)/(y_1 - y_0)$	$y_0 < y < y_1$
e_2	$(y - y_0)/(y_1 - y_0)$	$y_0 < y < y_1$
	$(y_1 - y)/(y_2 - y_1)$	$y_1 < y < y_2$
e_3	$(y - y_1)/(y_2 - y_1)$	$y_1 < y < y_2$
	$(y_2 - y)/(y_3 - y_2)$	$y_2 < y < y_3$
e_4	$(y - y_2)/(y_3 - y_2)$	$y_2 < y < y_3$
	$(y_3 - y)/(y_4 - y_3)$	$y_3 < y < y_4$
e_5	$(y - y_3)/(y_4 - y_3)$	$y_3 < y < y_4$

var	h_2	h_3	h_4	h_5	h_6	h_7	h_8	h_9	h_{10}
defn	f_1e_1	f_1e_2	f_2e_2	f_1e_3	f_2e_3	f_1e_4	f_2e_4	f_1e_5	f_2e_5

Table 5.1: Surface Parameterization for viscous wing problem

The baseline grid, pressure distribution and Mach distribution are shown in Figures 5-17 to 5-19. The baseline grid contains 101,900 nodes and 578,880 tetrahedra. The grid was generated using HYPGEN [8], the grid generation program associated with OVERFLOW [7]. Each hexahedral cell in this structured grid was divided into six tetrahedra using a similar prismatic grid generation process used to generate the 3D tetrahedral grids from 2D triangular grids in Sections 3.2 and 3.3. At the trailing edge there are approximately 12 points in the boundary layer. It should be noted that the gradient contributions, via $\frac{\partial \mathbf{R}}{\partial \mathbf{G}} \frac{\partial \mathbf{G}}{\partial \mathbf{U}}$ were removed from $\mathbf{L}$ in Equation (2.56) since it was found that fluctuations in the result of the minmod limiter were destabilizing in the early stages of the calculation. Good convergence was found using $\sigma = 2.5$ and $\nu = 2.0$. Note that although there is locally supersonic flow, no shocks are present, although the ability to capture these shocks is an inherent part of the flow analysis algorithm, as before.

The constrained BFGS algorithm described in Section 2.6.3 was used to minimize C_D at the baseline C_L. The final C_p and M distributions are shown in Figures 5-20 and 5-21 while the evolution of geometry, C_p, C_f and θ distributions and design variables are shown in Figures 5-22 to 5-26. It can be seen that the minimum has been practically reached after 4 line searches.

Figures 5-27 to 5-29 summarize the first three line searches. All of these plots show the variation of the cost function as found by numerical analysis, $\mathcal{F}$, and the linear model, $\mathcal{F}^*$ which is based on the gradients of $\mathbf{g} = \frac{\partial \mathcal{F}}{\partial \beta}$, and the value of $\mathcal{F}$ at the start of the line search. Figures 5-27 and 5-29 also show $\mathcal{F}^{**}$, the quadratic fit to $\partial \mathcal{F}/\partial \delta$, $\mathcal{F}(\delta = 0)$, and $\mathcal{F}(\delta_1)$ as discussed in Section 2.6.3. Furthermore, Figure 5-29 shows another quadratic model which is available at the start of the line search. This uses the BFGS prediction, $\mathbf{B}_k$, of the Hessian to estimate the variation along the search direction. This is included to show why a slightly different approach was taken during this line search. Basically $\mathbf{B}_k$ was suspected as being inaccurate for this line search, and therefore it was felt that the full Newton step should not be taken. Therefore the initial step was $\delta = 0.1$. Another case was simultaneously run at $\delta = 0.0318$, making use of the golden section ratio. In the course of running these cases, another quadratic model $\mathcal{F}^{***}$ was fit based on $\mathcal{F}(\delta = 0)$, $\mathcal{F}(\delta = 0.1)$, $\mathcal{F}(\delta = 0.0318)$, using unconverged cost function values. Then another case was run at δ corresponding to the minimum of $\mathcal{F}^{***}$, which is also plotted in Figure 5-29. Since this resulted in a large increase in the cost function, the minimum was taken to be $\delta = 0.1$, for which the fully converged point is plotted in Figure 5-29. The final line search brought a much better estimate of the Hessian and the full Newton step was taken, resulting in minimal change in F and the optimization process was considered to be converged.

Much of the behaviour observed for this case is similar to that observed for the 2D case discussed in Section 5.1.1. This might have been expected since similar design variables were utilized and minimal separation is present. Firstly, the cost function has only decreased by a small amount —- in this case, 4 counts or about 0.4% of the baseline C_D, whereas the 2D case resulted in an 18-count reduction. The smaller decrease may be due to the lower number of camber design variables per section for the 3D case — just one was used per span station, whereas in the 2D case, four were used. As for the 2D case, this modest decrease was expected since the baseline solution exhibited quite healthy boundary layer behaviour. Another similarity to the 2D case is that the decrease has come about partially by a reduction of the leading edge pressure peak. The resulting reduction in the net pressure increase and adverse pressure gradient along the upper surface causes the momentum thickness to be lower at the upper and lower surface trailing edge — as can be seen in Figure 5-25 — indicating a more healthy boundary layer and lower sectional drag for each section. Indeed analysis using the Squire-Young formula [42] based on boundary layer quantities at the trailing edge at four sample span stations indicated profile drag decreases at each section. Induced drag was not examined. Unlike the 2D NACA0012 case, little change in the separation point at any section was observed.

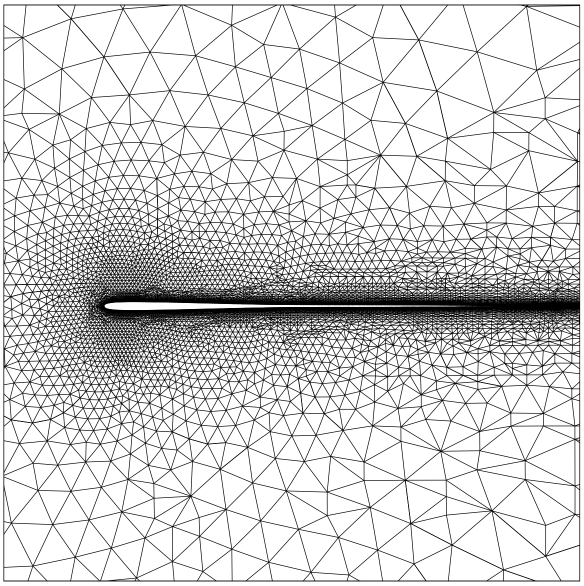

Figure 5-1: μ-UAV airfoil grid

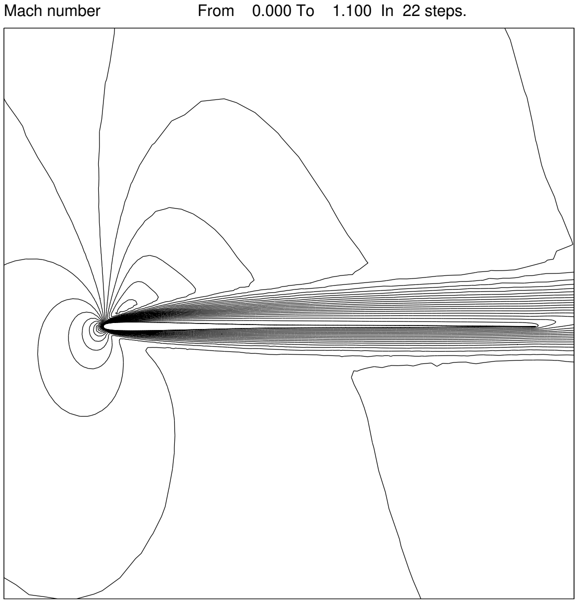

Figure 5-3: Baseline M distribution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\deg$

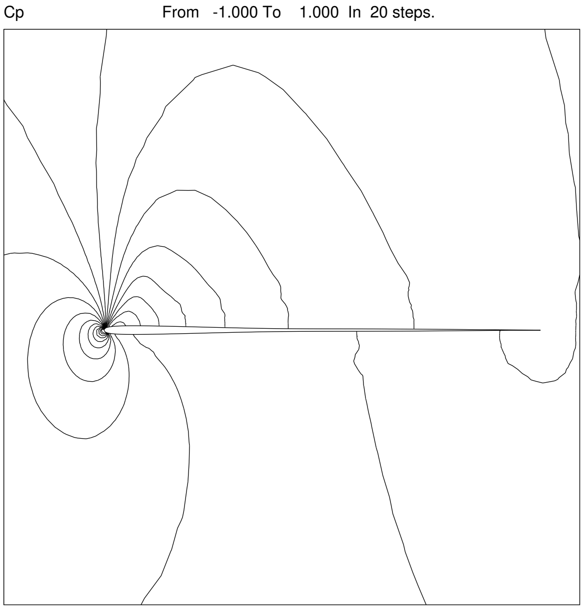

Figure 5-2: Baseline C_p distribution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\deg$

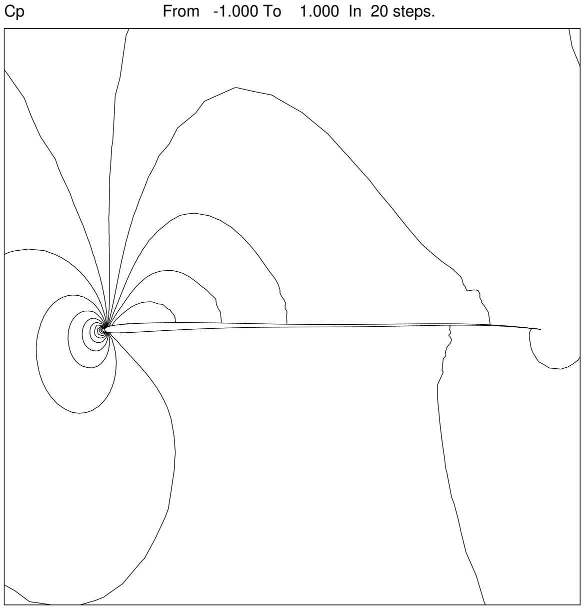

Figure 5-4: Final C_p distribution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\deg$

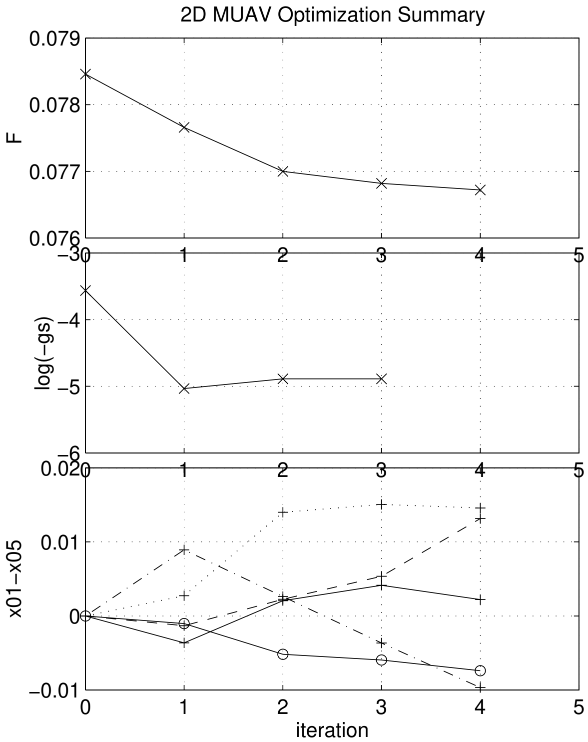

Figure 5-5: Design variable evolution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

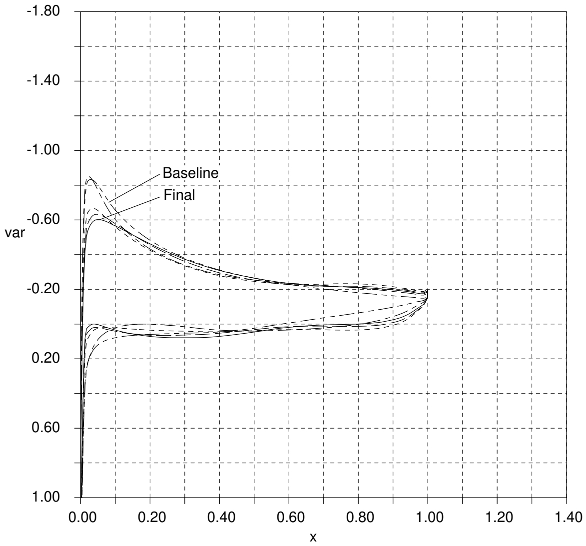

Figure 5-7: Surface C_p evolution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

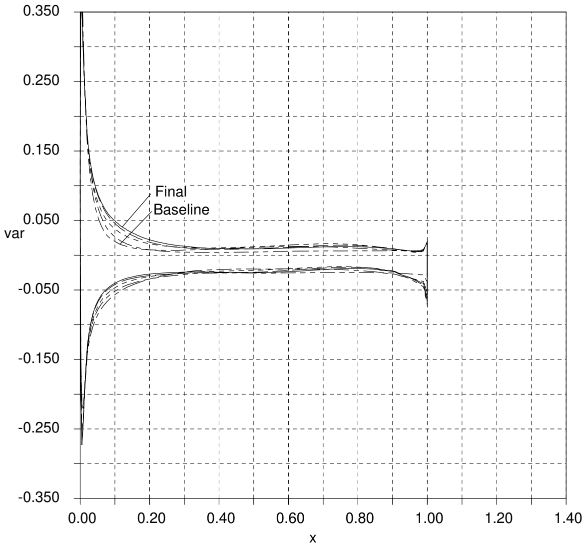

Figure 5-6: Surface C_f evolution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

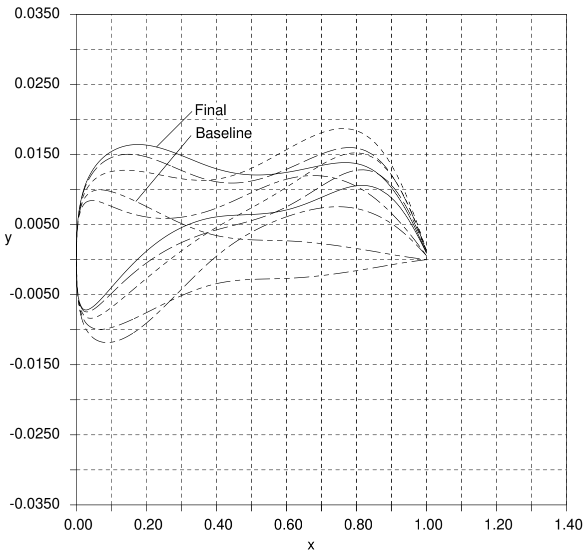

Figure 5-8: Geometry evolution for μ-UAV airfoil, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

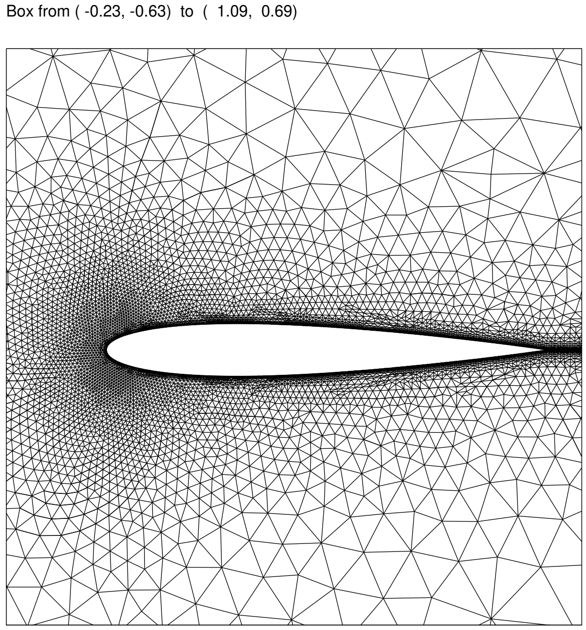

Figure 5-9: Re=2000 NACA0012 grid

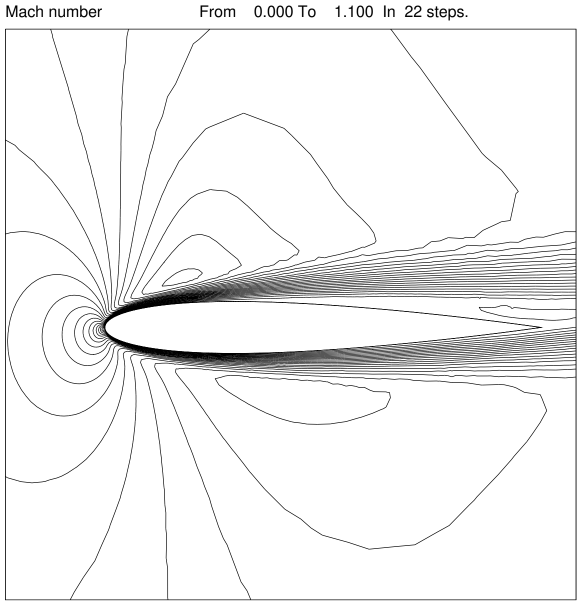

Figure 5-11: Baseline M distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

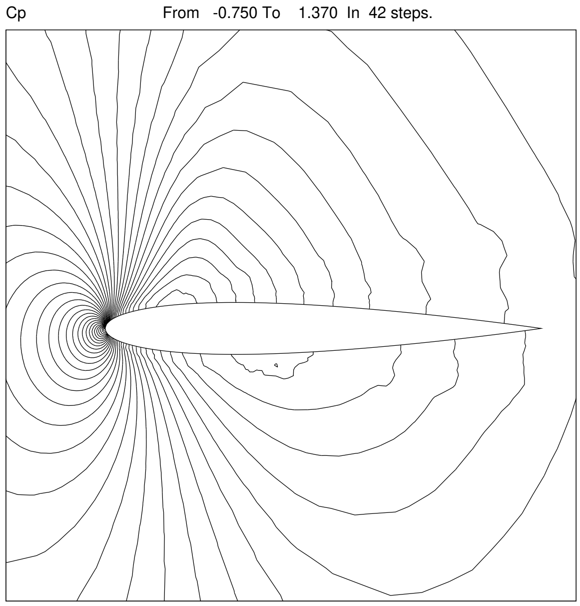

Figure 5-10: Baseline C_p distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

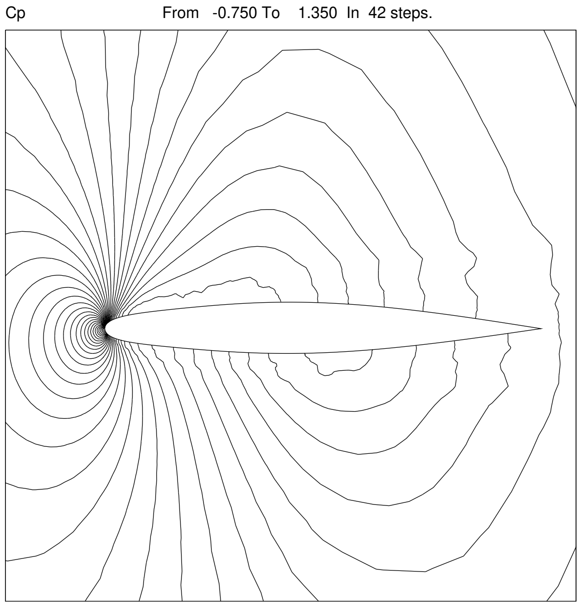

Figure 5-12: Final C_p distribution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\text{deg}$

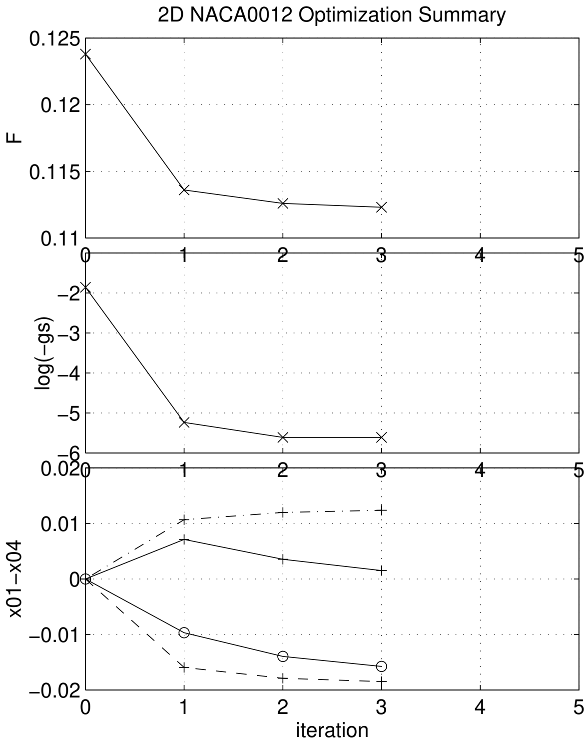

Figure 5-13: Design variable evolution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

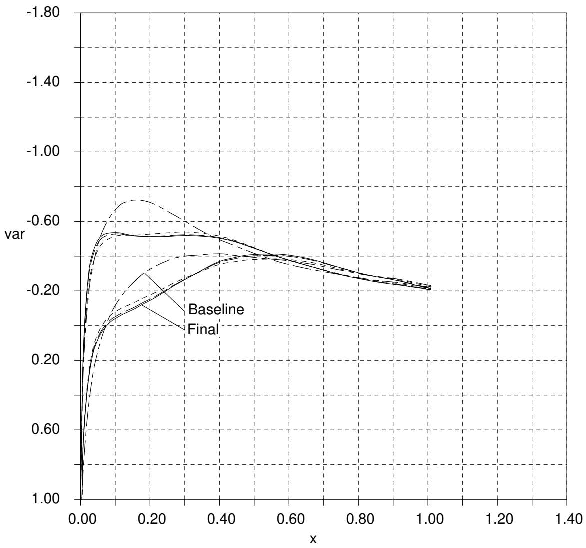

Figure 5-15: Surface C_p evolution for NACA0012, Re = 2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

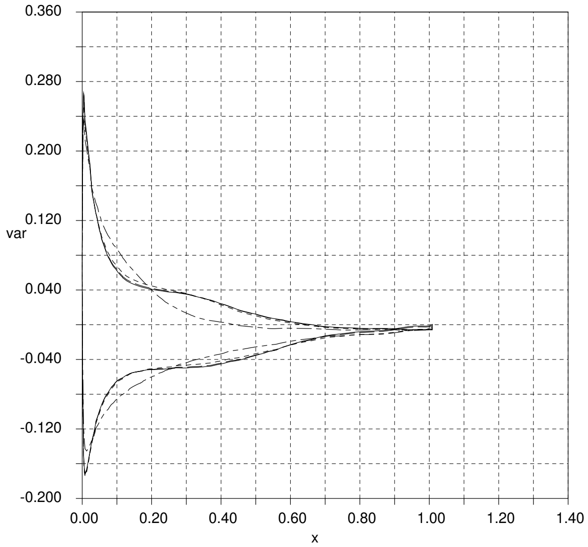

Figure 5-14: Surface C_f evolution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

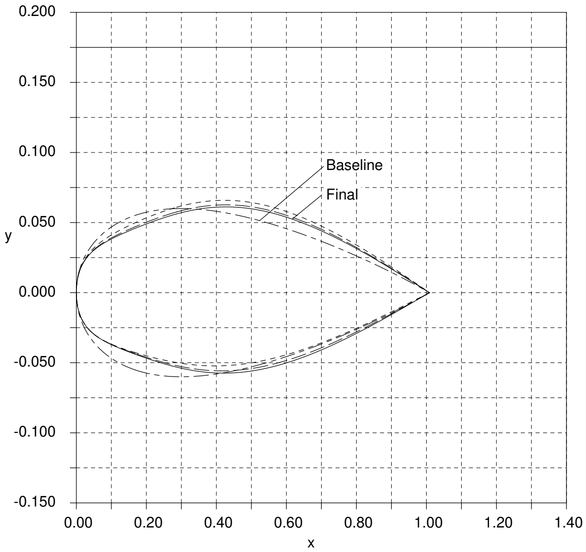

Figure 5-16: Geometry evolution for NACA0012, Re=2000, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

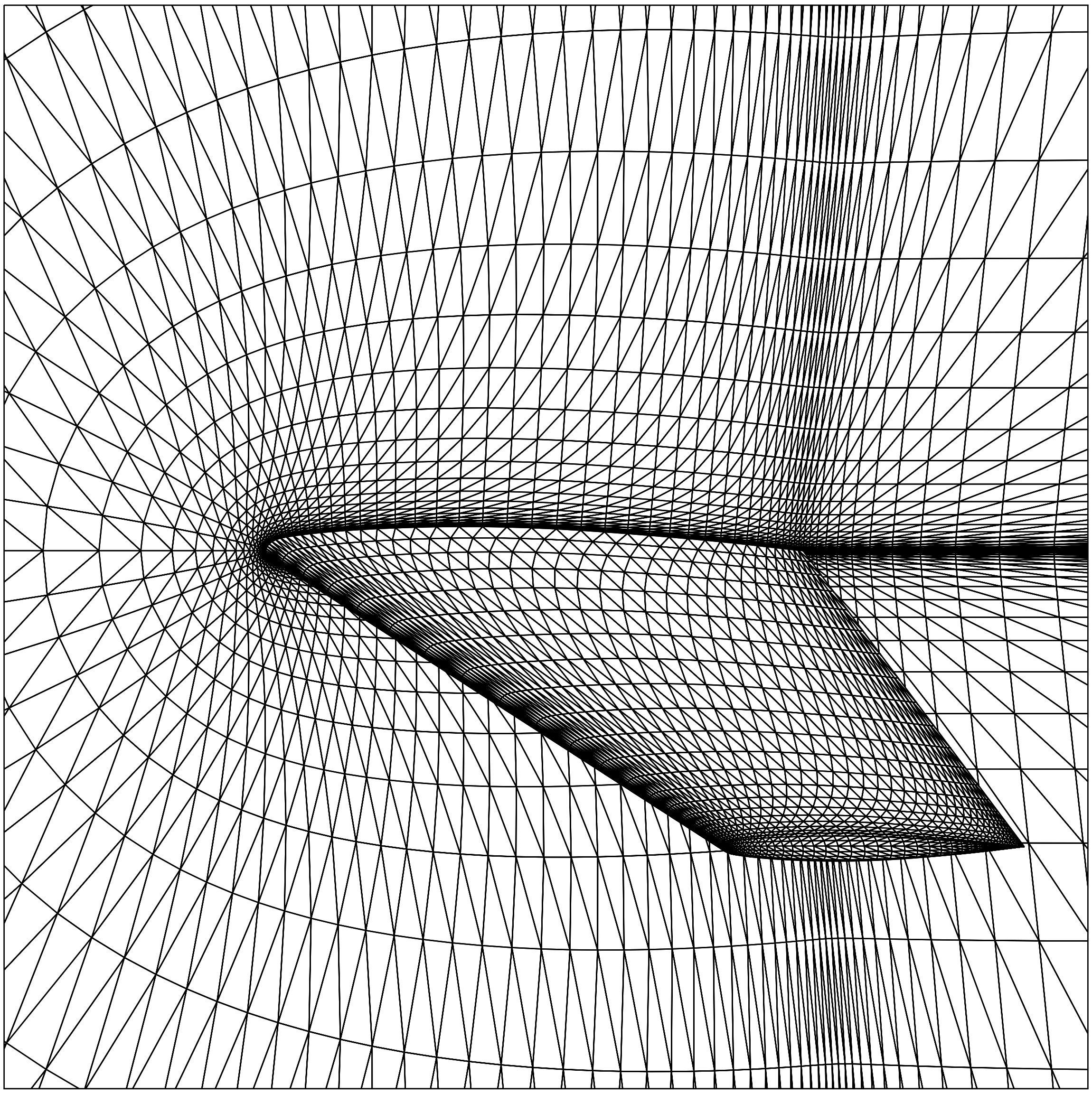

Figure 5-17: Surface grid for ONERA M6 calculation

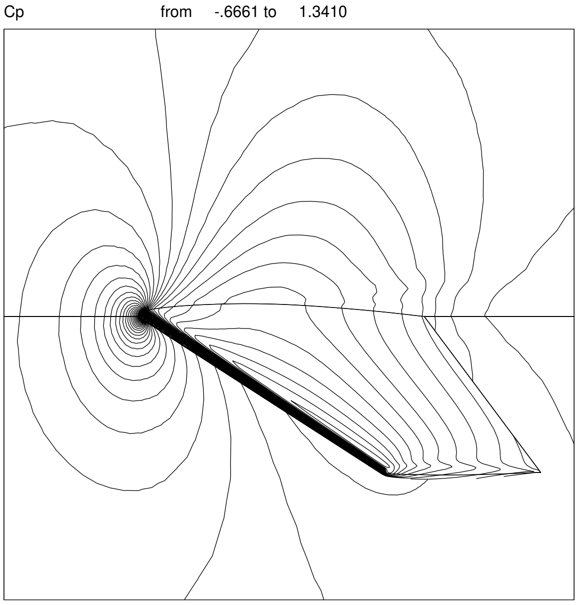

Figure 5-18: Baseline C_p distribution for ONERA M6 optimization exercise

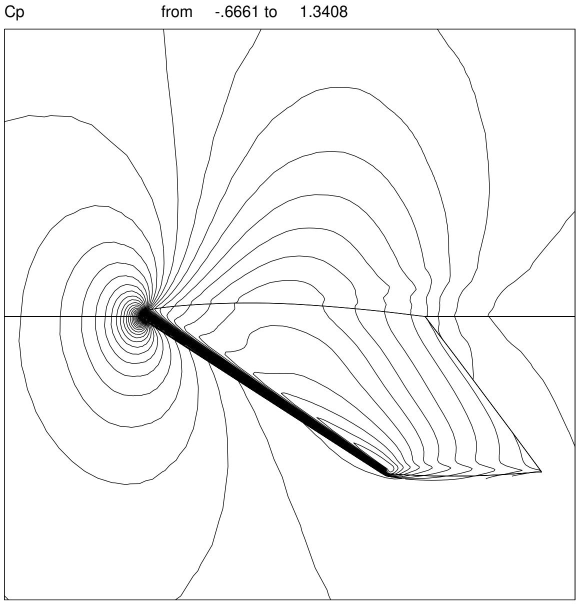

Figure 5-20: “Design 4” C_p distribution for ONERA M6 optimization exercise

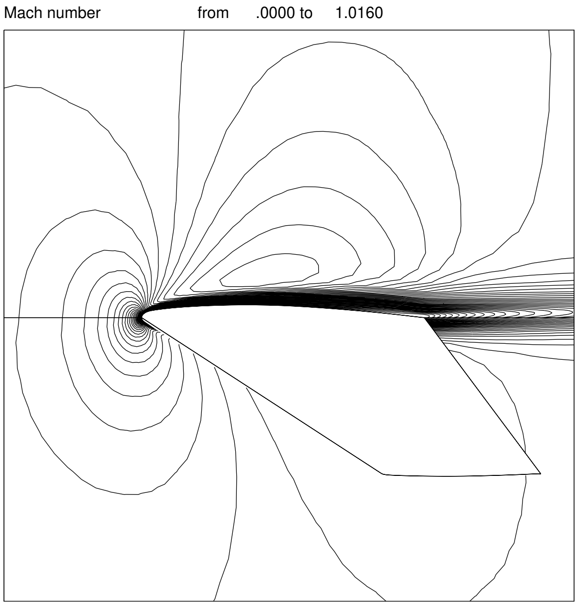

Figure 5-19: Baseline M distribution for ONERA M6 optimization exercise

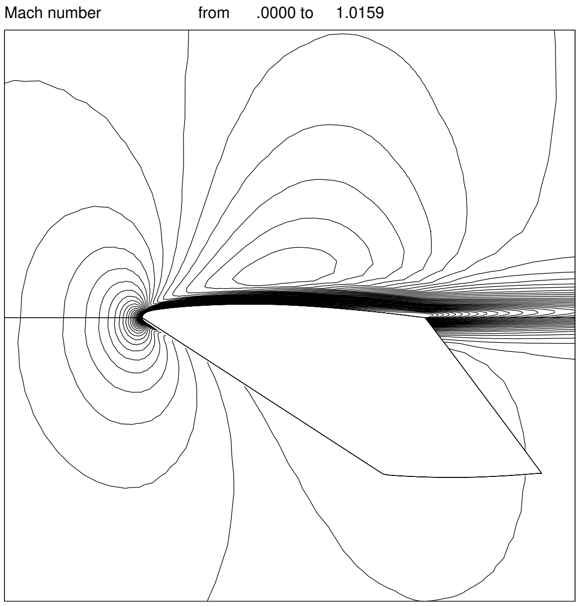

Figure 5-21: “Design 4” M distribution for ONERA M6 optimization exercise

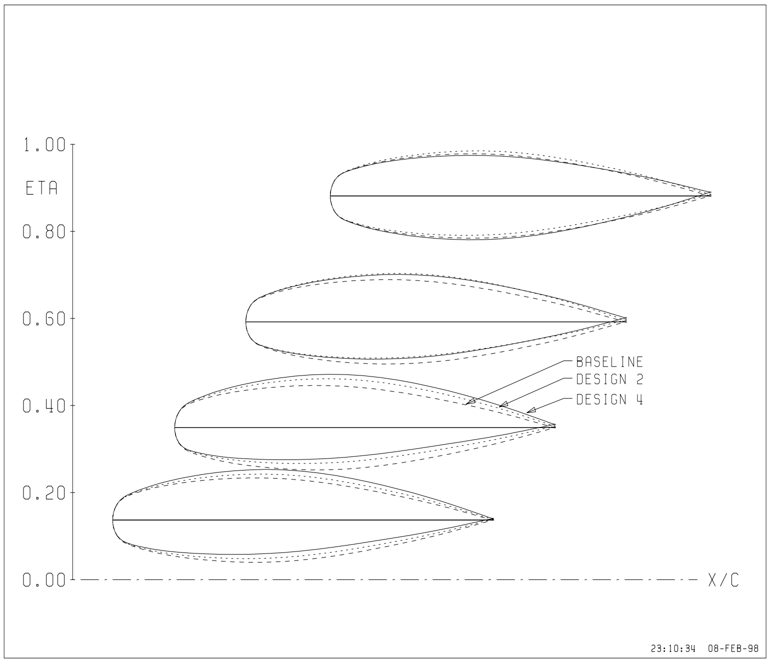

Figure 5-22: Geometry evolution for ONERA M6, Re=1600, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

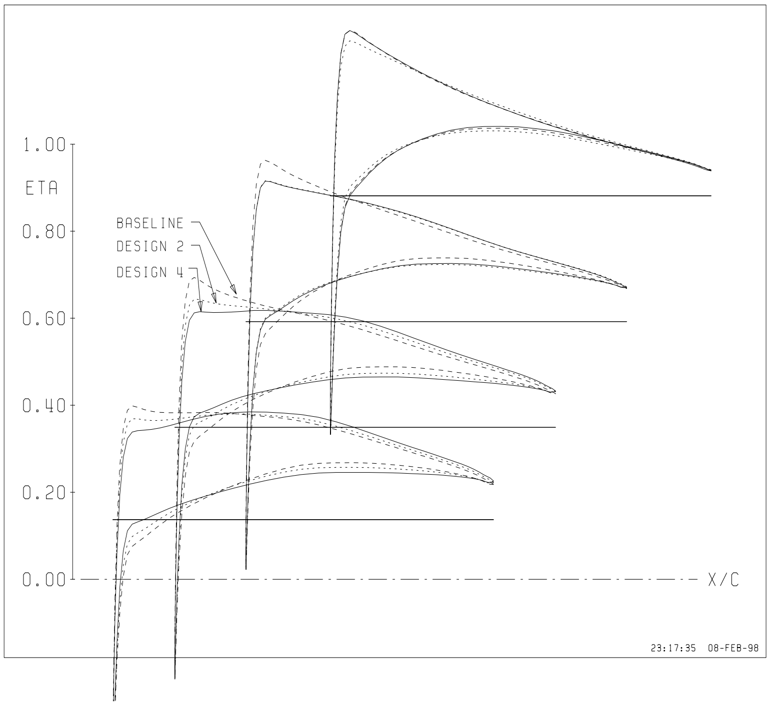

Figure 5-23: C_p evolution for ONERA M6, Re=1600, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

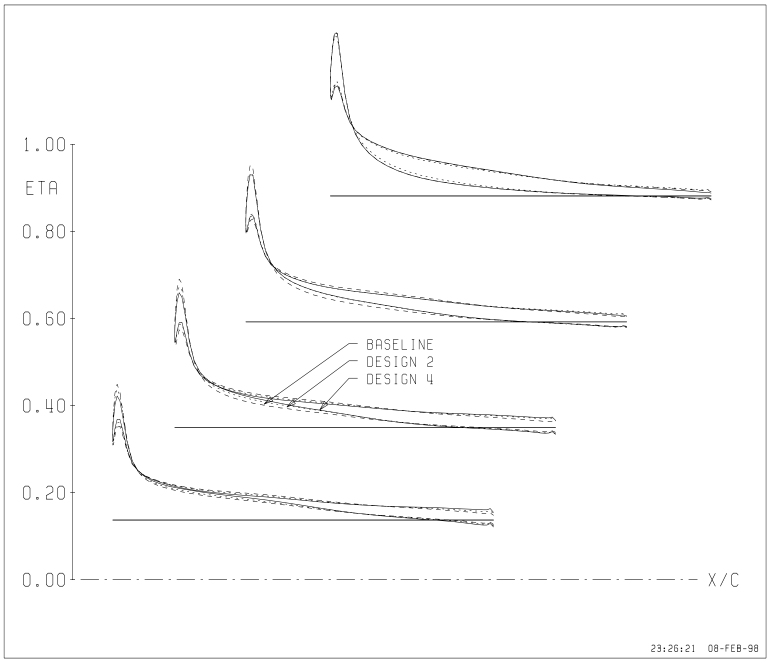

Figure 5-24: C_f evolution for ONERA M6, Re=1600, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

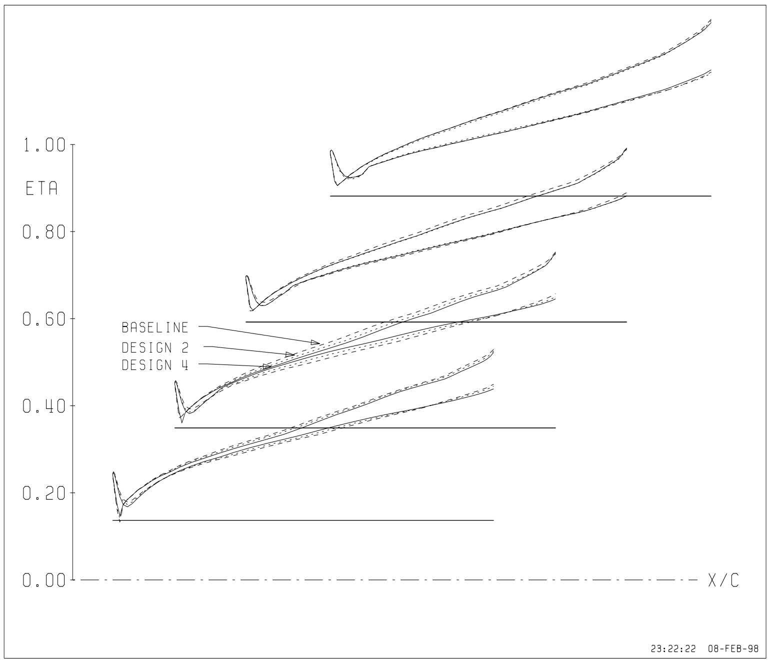

Figure 5-25: θ evolution for ONERA M6, Re=1600, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

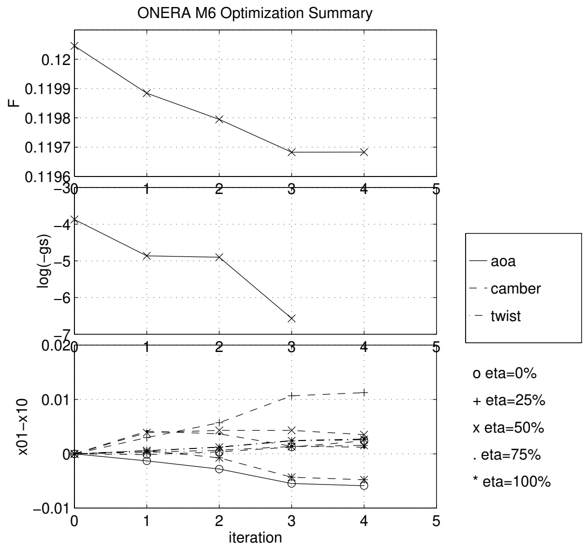

Figure 5-26: Design variable evolution for ONERA M6, Re=1600, $M_\infty = 0.8$, $\alpha = 3.5\,\mathrm{deg}$

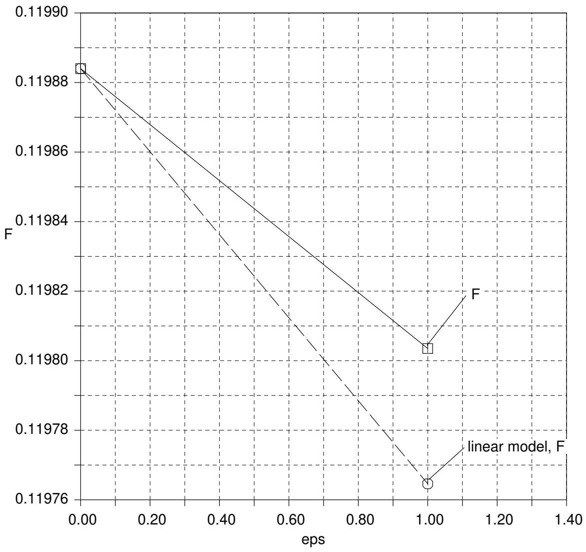

Figure 5-28: Line Search 2

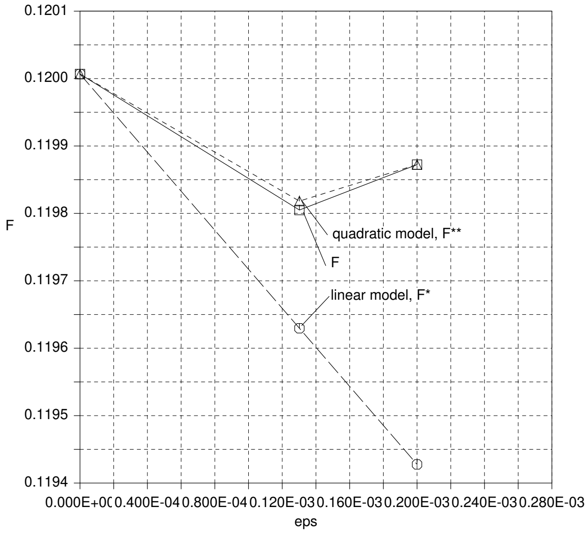

Figure 5-27: Line Search 1

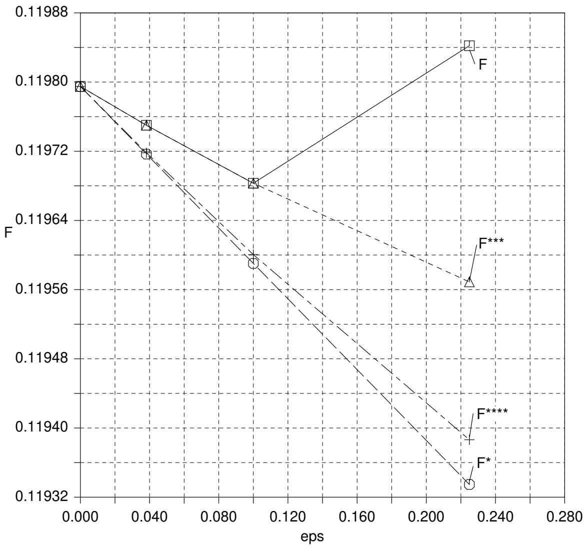

Figure 5-29: Line Search 3

Chapter 6

Conclusion

Schemes have been developed that allow efficient optimization based on the two- and three-dimensional Euler equations and the two- and three-dimensional Navier-Stokes equations. Both modal inverse pressure design and modal lift-constrained drag minimization have been demonstrated.

Three components are required to allow these types of optimization: (1) an efficient flow analysis scheme, (2) an efficient sensitivity analysis scheme and (3) an efficient optimization driver, which are discussed in the following sections.

6.1 Flow Analysis Scheme

For the Euler equations, a flow analysis scheme based on the Finite Volume method (or developed through use of a Galerkin weighted residual statement using piecewise linear trial functions) was used to perform the spatial discretization. Explicit multistage time-marching schemes were used for the Euler calculations discussed herein. A limited fourth order dissipative term, premultiplied by the Roe matrix, was used to stabilize the scheme and allow shock capturing. The gradients used for this dissipative scheme were calculated using a method that allows efficient storage of the residual jacobian — required for efficient sensitivity analysis — and preserves the LED properties of the underlying 1st order scheme.

On the other hand, for the scheme based on the Navier-Stokes equations, the gradient calculation method was based on a finite volume statement of the divergence theorem applied to an appropriate form of the gradient vector. This was complemented by an innovative scheme for efficient storage of the residual jacobian matrix — including the viscous terms. Although the rigorous LED character of the scheme is lost, this scheme tends to produce smoother and less dissipative solutions than the scheme described in the previous paragraph for transonic and low supersonic flows. Otherwise the spatial discretization of the two schemes are identical. It was also found necessary to use a point implicit time integration algorithm for the Navier-Stokes scheme to allow practical convergence to sufficiently low levels. The 3D versions were fully parallelized. Validation of the 2D and 3D Navier-Stokes schemes was performed by comparing with Blasius flat plate solutions and with NACA0012 solutions calculated using a coupled Euler/integral boundary layer solver (MSES).

6.2 Sensitivity Analysis Scheme

Use of the adjoint method based on the discrete sensitivity analysis approach allowed the cost of the sensitivity calculation to be independent of the number of design variables. The sensitivity analysis scheme incorporated an artificial unsteady term similar to that used for flow analysis, and was based on the recognition that such an approach would result in a stable scheme since it would have the same errormode amplification matrix as the flow analysis scheme. To eliminate severe convergence problems in the adjoint solver associated with low-level limit-cycling in the flow analysis calculations, an innovative check on local timestep satisfaction of the LED CFL-like condition was used. With this approach, near identical convergence rates were found for flow and both adjoint and direct sensitivity analysis schemes. The 3D versions were fully parallelized. Validation of viscous and inviscid sensitivity calculations was performed by comparing with finite difference sensitivities.

6.3 Optimization Algorithms

Algorithms were implemented to allow robust convergence of both modal inverse design and lift-constrained drag minimization processes. The best method for the former problem was found to be a BFGS algorithm. The best method for the latter was found to be through use of a BFGS algorithm implemented in the subspace orthogonal to the constraint gradients. This scheme incorporated a QR method to allow projection into this design subspace and a constraint enforcement step to return the step based on the linear model of the lift constraint back to feasible space from a point that was unfeasible due to the nonlinear nature of the actual lift variation.

6.4 Peripheral Points

Two pitfalls associated with viscous optimization problems jump to mind. The first was that for some cases with high levels of separation in the baseline solution, unsteadiness in the course of the optimization march impeded progress. This typically occured when there was separation on both surfaces and in some cases, even when it occured at low levels of residual convergence, caused the sensitivity analysis calculation to be unstable. However, a somewhat satisfactory workaround was developed which involved considering these design points as being of very high drag, and thereby rejecting those points in the course of a typical line search.

The second pitfall was that for some cases, separation tended to result in sudden large increases in drag — for significant movement of the separation point — and this resulted in nonsmoothness in the design space and therefore poor convergence of the BFGS algorithm to a good approximation of the Hessian. The workaround for this pitfall was just to use more careful line searches.

Fears expressed in the literature that the discontinuous nature of the nonlinear flux-limiting functions used in the flow analysis scheme would not allow accurate sensitivity calculation proved to be unfounded for the solver used herein. Although the limiting functions are discontinuous with respect to $\mathbf{U}$ at some points, the functions are piecewise differentiable and so in the limit of a finite difference step size tending to zero, the sensitivity calculations are exact. Although the limiting functions do result in some non-smoothness in the design space, this was found to be at a sufficiently low level that the optimization march was not impeded.

6.5 Future Work

The viscous calculations presented herein have been for low Reynolds numbers and laminar flow. A more useful capability would be to allow accurate modelling for higher Reynolds numbers. This requires incorporation of an accurate turbulence model suitable for use with unstructured grids. Possible options are the Spalart-Allmaras model [45] and the $k-\omega$ model of Wilcox [51]. The extension to higher Reynolds number also requires development of unstructured grid generation methods which provide sufficient resolution in the laminar sublayer and the log-law layer and improvement to the time-marching scheme to allow quick convergence for these grids with control volumes of very small dimension in the direction normal to the wall.

Incorporation of the C_L constraint directly into the flow analysis scheme would remove the requirement of performing an adjoint calculation for C_L at every iteration. The savings would be only about 10% per line search based on this factor since the adjoint calculations were found to be about one quarter the cost of the flow analysis calculations. The real savings would be in removing the requirement to incorporate a constraint correction step and perform the associated flow analysis. Hence incorporation of the C_L constraint directly into the flow analysis scheme would reduce the cost per line search by on the order of 50%, assuming there was no increase in flow analysis cost.

Bibliography

[1] Steven R. Allmaras. Analysis of a Local Matrix Preconditioner for the 2D Navier-Stokes Equations. AIAA-93-3330CP, 1993.

[2] Steven R. Allmaras. Contamination of Laminar Boundary Layers by Artificial Dissipation in Navier-Stokes Solutions. In *Proc. of the ICFD Conference on Numerical Methods for Fluid Dynamics*, Reading, England, UK, 1992.

[3] W.K. Anderson and V.Venkatakrishnan. Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation. AIAA-97-0643, 1997.

[4] P. Arminjon and A. Dervieux. Construction of TVD-like artificial viscosities on 2-dimensional arbitrary FEM grids. INRIA Report 1111, 1989.

[5] R.M. Beam and R.F. Warming. An Implicit Factored Scheme for the Compressible Navier-Stokes Equations. *AIAA Journal*, pages 393–402, April 1978.

[6] F. Beux and A. Dervieux. Exact-gradient shape optimization of a 2-D Euler flow. *Finite Elements in Analysis and Design*, 12:281–302, 1992.

[7] P. Buning, Chan W.M., Renze K.J., D. Sondat, I. Chiu, and Slotnick J.P. OVERFLOW User's Manual Version 1.6. Technical report, NASA Ames.

[8] W.M. Chan, Chiu I., and P. Buning. User's Manual for the HYPGEN Hyperbolic Grid Generator and the HGUI Graphical User Interface. TM 108791, NASA, October 1993.

[9] Roger L. Davis, Ron-Ho Ni, and James E. Carter. Cascade Viscous Flow Analysis using the Navier-Stokes Equations. AIAA-86-0033, 1986.

[10] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations.* Prentice-Hall, Inc, Englewood Cliffs, NJ, 1983.

[11] M. Drela. Newton Solution of Coupled Viscous/Inviscid Multielement Airfoil Flows. AIAA-90-1470, 1990.

[12] M. Drela. Design and Optimization Method for Multi-Element Airfoils. *AIAA*, 93-0969, 1993.

[13] M. Drela and M.B. Giles. Viscous-inviscid analysis of transonic and low Reynolds number airfoils. *AIAA Journal*, 25(10):1347–1355, 1987.

[14] J.K. Elliott and J. Peraire. Aerodynamic Design using Unstructured Meshes. AIAA-96-1941, 1996.

[15] J.K. Elliott and J. Peraire. Aerodynamic Optimization using Unstructured Meshes with Viscous Effects. *AIAA*, 97-1849, 1997.

[16] J.K. Elliott and J. Peraire. Practical 3D Aerodynamic Design and Optimization using Unstructured Meshes. *AIAA Journal*, 35(9):1479–1485, 1997.

[17] J.K. Elliott and J. Peraire. Three-Dimensional, Multipoint, Lift-Constrained Drag Minimization using Unstructured Meshes. *Aeronautical Journal of the Royal Aeronautical Society*, accepted for publication, 1998.

[18] M. Giles. An Energy Stability Analysis of Multi-step Methods on Unstructured Meshes. Technical Report 87-1, MIT CFD Lab, 1987.

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

[20] P. A. Gnoffo. An Upwind-Biased Point-Implicit Relaxation Algorithm for Viscous, Compressible Perfect Gas Flows. TP 2953, NASA, February 1990.

[21] P.A. Gnoffo, R.S. Candless, and H.C. Yee. Enhancements to Program Laura for Computation of Three-Dimensional Hypersonic Flow. AIAA Paper 87-0280, January 1987.

[22] Charles Hirsch. *Numerical Computation of Internal and External Flows, Vol.2*. John Wiley and Sons, Chichester, England, 1990.

[23] A. Iollo, G. Kuruvila, and S. Taasan. Pseudo-Time Method for Optimal Shape Design using the Euler Equations. Technical Report 95-59, ICASE, 1995.

[24] A. Iollo and M. D. Salas. Contribution to the Optimal Shape Design of Two-Dimensional Internal Flows with Embedded Shocks. Technical Report 95-28, ICASE, 1995.

[25] A. Jameson. Aerodynamic Design via Control Theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[26] A. Jameson. Artificial Diffusion, Upwind Biasing, Limiters and their Effect on Accuracy and Multigrid Convergence in Transonic and Hypersonic Flows. AIAA-93-3359, 1993.

[27] A. Jameson, N.A.Pierce, and L.Martinelli. Optimum Aerodynamic Design Using the Navier-Stokes Equations. AIAA-97-0110, January 1997.

[28] D.A. Lovell and J.J. Doherty. Aerodynamic Design of Aerofoils and Wings using a Constrained Optimisation Method. In *19th Congress of the International Council of Aeronautical Sciences*, Anaheim, California, USA, September 1994.

[29] Bilal H. Mughal. A Calculation Method for the Three-Dimensional Boundary-Layer Equations in Integral Form. Master's thesis, Dept. of Aeronautics and Astronautics, MIT, September 1992.

[30] J.C. Newman and A.C. Taylor. Three-dimensional Aerodynamic Shape Sensitivity Analysis and Design Optimization Using the Euler Equations on Unstructured Grids. AIAA-96-2464, 1996.

[31] J.C. Newman, A.C. Taylor, and G.W. Burgreen. An Unstructured Grid Approach to Sensitivity Analysis and Shape Optimization Using the Euler Equations. AIAA-95-1646, 1995.

[32] Brian A. Nishida. *Fully Simultaneous Coupling of the Full Potential Equation and the Integral Boundary Layer Equations in Three Dimensions.* PhD thesis, Dept. of Aeronautics and Astronautics, MIT, 1995.

[33] J. Peraire and K. Morgan. Unstructured Mesh Generation Including Directional Refinement for Aerodynamic Flow Simulation. *Finite Elements in Analysis and Design*, 25:343–356, 1997.

[34] J. Peraire, K. Morgan, M. Vahdati, and J. Peiro. The Construction and Behaviour of Some Unstructured Grid Algorithms for Compressible Flows. In *Proc. of the ICFD Conference on Numerical Methods for Fluid Dynamics*, Reading, England, UK, 1992.

[35] J. Peraire, J. Peiro, and K. Morgan. Finite Element Multigrid Solution of Euler Flows past Installed Aero-engines. *Computational Mechanics*, 11:433–451, 1993.

[36] J. Peraire, J. Peiro, and K. Morgan. Multigrid Solutions of the 3D Compressible Euler Equations on Unstructured Tetrahedral Grids. *International Journal for Numerical Methods in Engineering*, 36:1029–1044, 1993.

[37] W.H. Press, S. Teukolsky, W. Vettering, and B. Flannery. *Numerical Recipes in C.* Cambridge University Press, UK, 2nd ed. edition, 1992.

[38] J. Reuther and A. Jameson. Aerodynamic Shape Optimization of Wing and Wing-Body Configurations Using Control Theory. AIAA-95-0123, January 1995.

[39] J. Reuther, A. Jameson, J.J.Alonso, M.J.Rimlinger, and D.Saunders. Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers. AIAA-97-0103, January 1997.

[40] James Reuther. Aerodynamic Shape Optimization Using Control Theory. Technical Report 96-09, RIACS, 1996.

[41] P.L. Roe. Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[42] H. Schlichting. *Boundary Layer Theory.* McGraw–Hill, New York, 7th ed. edition, 1987.

[43] G.R. Shubin and P.D. Frank. A Comparison of Two Closely-Related Approaches to Aerodynamic Design Optimization. In *3rd International Conference on Inverse Design Concepts and Optimization in Engineering Sciences*, October 1991.

[44] J. Sobieski. Sensitivity of complex , internally coupled systems. *AIAA Journal*, 28:1, 1990.

[45] P.R. Spalart and S.R. Allmaras. A One-Equation Turbulence Model for Aerodynamic Flows. AIAA Paper 92-0439, 1992.

[46] John C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations.* Wadsworth & Brooks/Cole, Pacific Grove, CA, 1989.

[47] S. Tatsumi, L. Martinelli, and A. Jameson. Flux-Limited Schemes for the Compressible Navier-Stokes Equations. *AIAA Journal*, 33:252–261, 1995.

[48] George B. Thomas and Ross L. Finney. *Calculus and Analytic Geometry.* Addison-Wesley Pub. Co., Reading, Mass., 1988.

[49] A.E. Von Doenhoff and A.H. Abbot. *Theory of Wing Sections.* Dover Publications, Inc, Mineola, NY, 1959.

[50] F. M. White. *Viscous Fluid Flow.* McGraw–Hill, New York, NY, 2nd edition, 1991.

[51] David C. Wilcox. *Turbulence Modelling for CFD.* DCW Industries, Palm Drive, CA, 1st edition, 1993.

Appendix A

Weights for Edge Data Structure

The weights associated with each edge are derived through the use of the finite volume method. Discretization of $\mathbf{U}$ and $\mathbf{F}^j$ on a 2D unstructured triangulation is accomplished through the use of piecewise linear trial functions. Referring to Figure A-1, an exact evaluation of the line integral in Equation (2.1) for the control volume associated with node a results in,

$$\Omega_o \frac{d\mathbf{U}_o}{dt} = -\left\{ \frac{\vec{\mathbf{F}}_1 + \vec{\mathbf{F}}_2}{2} \cdot \vec{S}_{21} + \frac{\vec{\mathbf{F}}_2 + \vec{\mathbf{F}}_3}{2} \cdot \vec{S}_{32} + \frac{\vec{\mathbf{F}}_3 + \vec{\mathbf{F}}_4}{2} \cdot \vec{S}_{43} + \right.$$
$$\left. \frac{\vec{\mathbf{F}}_4 + \vec{\mathbf{F}}_5}{2} \cdot \vec{S}_{54} + \frac{\vec{\mathbf{F}}_5 + \vec{\mathbf{F}}_6}{2} \cdot \vec{S}_{65} + \frac{\vec{\mathbf{F}}_6 + \vec{\mathbf{F}}_1}{2} \cdot \vec{S}_{16} \right\} \tag{A.1}$$

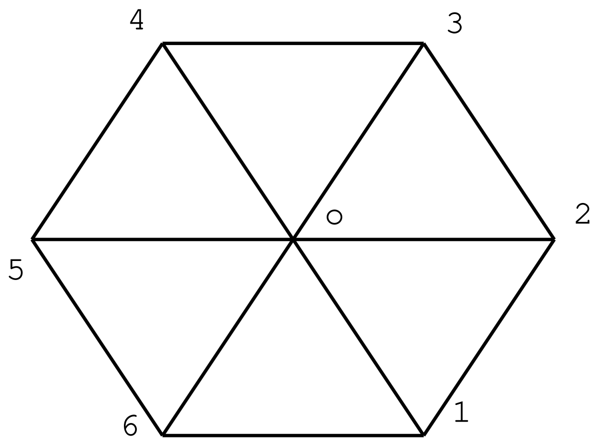

Figure A-1: Typical Triangulation

where, for example, $\vec{S}_{21} = (y_2 - y_1, x_1 - x_2)$. Simple algebraic manipulation results in

$$
\begin{aligned}
\Omega_o \frac{d\mathbf{U}_o}{dt} &= -\left\{ \frac{\vec{\mathbf{F}}_1}{2} \cdot \vec{S}_1 + \frac{\vec{\mathbf{F}}_2}{2} \cdot \vec{S}_2 + \frac{\vec{\mathbf{F}}_3}{2} \cdot \vec{S}_3 + \frac{\vec{\mathbf{F}}_4}{2} \cdot \vec{S}_4 + \frac{\vec{\mathbf{F}}_5}{2} \cdot \vec{S}_5 + \frac{\vec{\mathbf{F}}_6}{2} \cdot \vec{S}_6 \right\} \\
&= -\left\{ \frac{\vec{\mathbf{F}}_1 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_1 + \frac{\vec{\mathbf{F}}_2 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_2 + \frac{\vec{\mathbf{F}}_3 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_3 + \right. \\
&\qquad \left. \frac{\vec{\mathbf{F}}_4 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_4 + \frac{\vec{\mathbf{F}}_5 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_5 + \frac{\vec{\mathbf{F}}_6 + \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_6 \right\}
\end{aligned} \tag{A.2}
$$

where $\vec{S}_k = (y_{k+1} - y_{k-1}, x_{k-1} - x_{k+1})$ (with $k+1$, for example, representing the node at the other end of the next edge in the counterclockwise direction). Note that the relation $\sum_k \vec{S}_k = (0,0)$ has been used. Each term in (A.2) is recognizable as a contribution from an individual edge. A similar analysis for the control volume centered at node 1, for example, reveals that the contribution from the edge connecting nodes o and 1 is

$$
[\mathbf{R}]_{ko} = \frac{\vec{\mathbf{F}}_o + \vec{\mathbf{F}}_k}{2} \cdot \vec{S}_k. \tag{A.3}
$$

with $k = 1$. Hence if a loop over edges is performed, the residual contributions to nodes o and 1 corresponding to a given edge can be calculated at the same time, resulting in a factor of two savings over algorithms which loop over the elements or control volumes. In summary, the residual increment for an interior edge is simply

$$
[\mathbf{R}]_{ok} = -\frac{\vec{\mathbf{F}}_o + \vec{\mathbf{F}}_k}{2} \cdot \vec{S}_k. \tag{A.4}
$$

Special treatment is required along the boundary integration paths in the edge formulation. This is because typically boundary information is applied to some of the state vector components there. This requires adaptation of the contributions from boundary nodes for some residual increments while other contributions, associated with the closure of the control volume, have no boundary-associated adaptation.

Referring to Figure A-2, the application of a similar procedure to the interior edge-based approach described above results in the following expression for the time rate of change for the state vector at a.

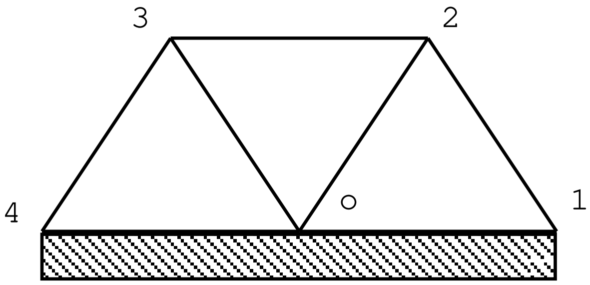

Figure A-2: Typical Triangulation at boundary

$$\begin{aligned}\Omega_o \frac{d\mathbf{U}_o}{dt} &= -\left\{ \frac{\vec{\mathbf{F}}_1+\vec{\mathbf{F}}_2}{2}\cdot\vec{S}_{21} + \frac{\vec{\mathbf{F}}_2+\vec{\mathbf{F}}_3}{2}\cdot\vec{S}_{32} + \frac{\vec{\mathbf{F}}_3+\vec{\mathbf{F}}_4}{2}\cdot\vec{S}_{43} + \frac{\vec{\mathbf{F}}_4^b+\vec{\mathbf{F}}_o^b}{2}\cdot\vec{S}_{o4} + \frac{\vec{\mathbf{F}}_o^b+\vec{\mathbf{F}}_1^b}{2}\cdot\vec{S}_{1o} \right\} \\ &= -\left\{ \frac{\vec{\mathbf{F}}_2+\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_2 + \frac{\vec{\mathbf{F}}_3+\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_3 + \frac{\vec{\mathbf{F}}_4^b+\vec{\mathbf{F}}_o^b+2\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_{o4} + \right. \\ &\quad \left. \frac{\vec{\mathbf{F}}_1^b+\vec{\mathbf{F}}_o^b+2\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_{1o} + \frac{\vec{\mathbf{F}}_4+\vec{\mathbf{F}}_o}{2}\cdot\vec{S}_{43} + \frac{\vec{\mathbf{F}}_o+\vec{\mathbf{F}}_1}{2}\cdot\vec{S}_{21} \right\} \end{aligned} \tag{A.5}$$

where the relation $\sum_k \vec{S}_k = (0,0)$ has once again been used and where the superscript b indicates that the flux is formed based on information known about the boundary state vector, such as zero normal wall velocity, or the direction of propagation of characteristic quantities in the farfield. In this equation the first two terms are recognizable as edge contributions, and the last four terms represent the boundary contributions for two boundary edges where the boundary contribution from a given edge il to node i can be written as

$$[\mathbf{R}_i^b]_{il} = -\frac{\vec{\mathbf{F}}_i^b+\vec{\mathbf{F}}_l^b+2\vec{\mathbf{F}}_i}{2}\cdot\vec{S}_b - \frac{\vec{\mathbf{F}}_i+\vec{\mathbf{F}}_l}{2}\cdot\vec{S}_{l,l+1} \tag{A.6}$$

where the subscripts are now based on Figure 2-1.

In 3D, the edge based data structure results in the following

$$\Omega_I \frac{d\mathbf{U}_I}{dt} = -\sum_{II_e=1}^{\mu_i} \frac{\vec{S}_{II_e}}{2}\cdot(\vec{\mathbf{F}}_I+\vec{\mathbf{F}}_{I_e}) + \left\langle \sum_{f=I_1}^{I_\nu} D_f(4\bar{\mathbf{F}}_I^n + 2\bar{\mathbf{F}}_{I_1 f}^n + 2\bar{\mathbf{F}}_{I_2 f}^n + 2\mathbf{F}_I^n - \mathbf{F}_{I_1 f}^n - \mathbf{F}_{I_2 f}^n) \right\rangle_I \tag{A.7}$$

where the first summation extends over all the edges e which contain node I. Here the term $\langle\cdots\rangle_I$ is included only when the node I lies on the boundary. $\vec{S}_{II_e} = (S_{II_e}^1, S_{II_e}^2, S_{II_e}^3)$ and can be calculated as follows

$$S^j_{II_e} = -\sum_{E\epsilon II_e} \frac{\Omega_E}{2}\frac{\partial N_I}{\partial x_j}\Bigg]_E + \left\langle \sum_{f\epsilon II_e} \frac{\Gamma_f}{12} n_j \right\rangle_{II_e} \quad , \quad D_f = -\frac{\Gamma_f}{24}, \tag{A.8}$$

$$\mathbf{F}^n_I = \vec{\mathbf{F}} \cdot \hat{n} \tag{A.9}$$

where N_I is a piecewise linear, FEM-type, trial function with maximum at node I, while the overbar in $\bar{\mathbf{F}}^n_I$ signifies that boundary information is used in the flux calculation. The first summation here extends over all elements E which contain the edge II_e. The term $\langle \cdots \rangle_I$ is included only if the edge e is a boundary edge. For further details, the reader is referred to [35].

Appendix B

Local Extremum Diminishing (LED) Schemes

LED schemes are ones that have the properties that maxima should not increase while minima should not decrease. The theory for these schemes is briefly developed here for one-dimensional scalar schemes, multidimensional scalar schemes and finally multidimensional schemes based on systems of equations. The reader is referred to [26] for a more comprehensive treatment of the subject.

For a scalar equation, LED conditions for the semidiscrete form of a scheme that can be expressed as

$$\frac{du_o}{dt} = \sum_{i=1}^{K} c_{oi}(u_i - u_o) \tag{B.1}$$

are

$$c_{oi} > 0 \quad i = 1, \cdots, K \tag{B.2}$$

where K represents the total number of nodes that contribute to the residual at node i. This can be seen by considering the cases where u_o represents (a) a local maximum and (b) a local minimum. For case (a), $u_i - u_o \leq 0$ implying that if the TVD conditions hold, then $du_o/dt \leq 0$ and the maxima must decrease. It is straightforward to see that the opposite is true for case (b). It is noted that in 1D, these conditions also imply that the scheme is total variation decreasing (TVD).

B.1 One-dimensional Scalar Schemes

Consider a one-dimensional scalar conservation law.

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{B.3}$$

If the scheme is approximated in semi-discrete form as

$$\frac{du_j}{dt} + \frac{h_{j+1/2} - h_{j-1/2}}{\Delta x} = 0 \tag{B.4}$$

where $h_{j+1/2}$ is a combination of a physical flux and a diffusive flux:

$$h_{j+1/2} = \frac{f_{j+1} + f_j}{2} - \alpha_{j+1/2}(u_{j+1} - u_j) \tag{B.5}$$

then the scheme can be rewritten as

$$\begin{aligned}\Delta x \frac{du_j}{dt} &= -h_{j+1/2} + h_{j-1/2} \\ &= \left(-\frac{a_{j+1/2}}{2} + \alpha_{j+1/2}\right) \Delta u_{j+1/2} - \left(\frac{a_{j-1/2}}{2} + \alpha_{j-1/2}\right) \Delta u_{j-1/2}\end{aligned} \tag{B.6}$$

where $a_{j+1/2}$ is an approximation to the wavespeed $a = \partial f/\partial u$ and satifies

$$f_{j+1} - f_j = a_{j+1/2}(u_{j+1} - u_j) \tag{B.7}$$

The LED condition is satisfied if

$$\alpha_{j+1/2} \geq \frac{|a_{j+1/2}|}{2}. \tag{B.8}$$

It is noted that the least diffusive first order scheme which satisfies the LED condition — taking the equality in Equation (B.8) — produces the first order upwind scheme.

Higher-order accurate LED schemes with much lower diffusion can be constructed by including antidiffusion in the diffusive flux which is limited in such a way that the scheme may still be written in the form of (B.1) with positive coefficients. For example, by extending the stencil one node to the left and one node to the right, we could use,

$$h_{j+1/2} = f_{j+1/2} - \alpha_{j+1/2} \left\{ \Delta u_{j+1/2} - L(\Delta u_{j+3/2}, \Delta u_{j-1/2}) \right\} \tag{B.9}$$

where $L(u, v)$ is a limited average such as the minmod function This results in

$$\begin{aligned}\Delta x \frac{du_j}{dt} &= \left(-\frac{a_{j+1/2}}{2} + \alpha_{j+1/2} + \alpha_{j-1/2}\phi(r^-)\right) \Delta u_{j+1/2} + \\ &\quad \left(-\frac{a_{j-1/2}}{2} - \alpha_{j-1/2} - \alpha_{j+1/2}\phi(r^+)\right) \Delta u_{j-1/2}\end{aligned} \tag{B.10}$$

where $L(u, v) = u\phi(v/u) = v\phi(u/v)$ and

$$r^+ = \frac{\Delta u_{j+3/2}}{\Delta u_{j-1/2}} \quad , \quad r^- = \frac{\Delta u_{j-3/2}}{\Delta u_{j+1/2}}. \tag{B.11}$$

Hence the scheme is LED if $\alpha_{j+1/2} \geq \frac{|a_{j+1/2}|}{2}$ and $\phi(r) \geq 0$. The latter is satisfied by many nonlinear flux-limiting functions including the minmod limiter. The reader is referred to [26] and [22] for a comprehensive list of candidate flux limiting functions which could be used.

B.2 Multi-dimensional Scalar Schemes

Consider a scalar version of Equation (2.1).

$$\int_\Omega \frac{\partial u}{\partial t} d\Omega = \oint_\Gamma (-F^j) n_j d\Gamma. \tag{B.12}$$

Using the same method of discretization as that described in Appendix A results in the following semi-discrete expression for interior nodes.

$$\Omega_o \frac{du_o}{dt} + \sum_k \frac{\vec{F}_k - \vec{F}_o}{2} \cdot \vec{S}_k = 0 \tag{B.13}$$

This can be rewritten as

$$\Omega_o \frac{du_o}{dt} + \sum_k a_{ko} \Delta u_{ko} = 0 \tag{B.14}$$

where a_{ko} satifies

$$(\vec{F}_k - \vec{F}_o) \cdot \vec{S}_k = a_{ko}(u_k - u_o). \tag{B.15}$$

Finally to produce a scheme satisfying the positivity requirements of a LED scheme, we add a dissipative flux to produce:-

$$\Omega_o \frac{du_o}{dt} + \sum_k \left\{ a_{ko} - \alpha_{ko} \right\} \Delta u_{ko} = 0 \tag{B.16}$$

where now $\alpha_{ko} \geq |a_{ko}|$ is required.

B.3 Multi-dimensional System Schemes

Following the LED ideas for the one-dimensional and multidimensional scalar cases, we can construct a scheme for multidimensional systems of equations whereby the contributions to the rate of change of the local characteristic variables — *based on the direction associated with the the current edge* — satisfy the positivity condition. This treatment has been found by many researchers to capture shocks over very few points with little or no overshoot. Indeed, in one dimension, shocks are captured with just one interior point [26].

The governing equations are taken to be of the form given in Equation (2.1), but with $\mathbf{G} = 0$. The first order scheme is written as

$$\Omega_o \frac{d\mathbf{U}_o}{dt} + \sum_k \left\{ \frac{\vec{\mathbf{F}}_k - \vec{\mathbf{F}}_o}{2} \cdot \vec{S}_k - \alpha \frac{|\mathbf{A}_{ko}|}{2} \left(\mathbf{U}_k - \mathbf{U}_o \right) \right\} = 0 \tag{B.17}$$

where $\mathbf{A}_{ko}$ is a mean value Jacobian following Roe's derivation which satisfies

$$\mathbf{A}_{ko} \left(\mathbf{U}_k - \mathbf{U}_o \right) = (\vec{\mathbf{F}}_k - \vec{\mathbf{F}}_o) \cdot \vec{S}_k,$$

$$|\mathbf{A}_{ko}| = \mathbf{P}_{ko} |\mathbf{\Lambda}_{ko}| \mathbf{P}_{ko}^{-1}$$

and $\mathbf{P}_{ko}$ is the matrix of eigenvectors of $\mathbf{A}_{ko}$ and $\mathbf{\Lambda}_{ko}$ is a diagonal matrix containing the corresponding eigenvalues. Isolating the contribution from one edge to $\Omega_o d\mathbf{U}_o/dt$, and premultiplying by $\mathbf{P}_{ko}^{-1}$ produces

$$(\mathbf{\Lambda} - \alpha|\mathbf{\Lambda}|)\mathbf{P}_{ko}^{-1}(\mathbf{U}_k - \mathbf{U}_o)$$

or in terms of the local characteristic variables,

$$(\mathbf{\Lambda} - \alpha|\mathbf{\Lambda}|)\Delta\mathbf{W}_{ko}$$

So coefficients multiplying characteristic increments are positive on a *local* basis as long as $\alpha \geq 1$.

Appendix C

Sample Shape Design Variables

In this research, most design changes to the surfaces (except for the design variable corresponding to Equation C.12) were implemented using the following expression.

$$\begin{aligned} z_{upper}(x,y) &= z^0_{upper}(x,y) &+ \textstyle\sum_{i=1}^{N_{des}} \beta_i h_i(x,y) \\ z_{lower}(x,y) &= z^0_{lower}(x,y) &+ \textstyle\sum_{i=1}^{N_{des}} \varphi_i \beta_i h_i(x,y) \end{aligned} \tag{C.1}$$

where φ_k is -1 and 1 for thickness and camber design variables, respectively and z^0_{upper}, for example, represents the baseline upper surface. In 2D, we set

$$h_i(x,y) = f_i(x) \tag{C.2}$$

and select from the functions listed in Section A.1 below. In 3D, we set the total change to be a function that is separable into functions of x and y as follows.

$$h_i(x,y) = f_j(x) g_k(y) \tag{C.3}$$

and select $g_k(y)$ from the functions listed in Section A.2 below.

C.1 Chordwise Functions

Hicks-Henne: non-zero trailing edge slope

$$f_k(x) = \sin(\pi(\frac{x}{c})^{p_k}) \tag{C.4}$$

Hicks-Henne: non-zero leading edge slope

$$f_k(x) = \sin(\pi(1 - \frac{x}{c})^{p_k}) \tag{C.5}$$

Hicks-Henne: zero slope; forward peak

$$f_k(x) = \sin^2(\pi(\frac{x}{c})^{p_k}) \tag{C.6}$$

Hicks-Henne: zero slope; aft peak

$$f_k(x) = \sin^2(\pi(1 - \frac{x}{c})^{p_k}) \tag{C.7}$$

Twist/shear

$$f_k(x) = \frac{x}{c} \tag{C.8}$$

4-Series meanline

$$f_k(x) = \begin{cases} (0.8(x/c) - (x/c)^2)/0.16 & 0 < x/c < 0.4 \\ (0.2 + 0.8(x/c) - (x/c)^2)/0.7056 & 0.4 < x/c < 1 \end{cases} \tag{C.9}$$

4-Series thickness

$$f_k(x) = 0.2969\sqrt{x/c} - .12810(x/c) - .35160(x/c)^2 + .28430(x/c)^3 - .1015(x/c)^4 \tag{C.10}$$

Vertical Deflection

$$f_k(x) = 1 \tag{C.11}$$

C.2 Rotation about a point

$$\left\{ \begin{array}{c} x_i \\ \\ y_i \end{array} \right\} = \left\{ \begin{array}{c} x_c \\ \\ y_c \end{array} \right\} + \left[\begin{array}{cc} cos\beta & -sin\beta \\ \\ sin\beta & cos\beta \end{array} \right] \left\{ \begin{array}{c} x_i^0 - x_c \\ \\ y_i^0 - y_c \end{array} \right\} \tag{C.12}$$

C.3 Spanwise Functions

In general, the spanwise functions were defined to have maxima at y_k and to only be nonzero between y_{k-1} and y_{k+1}, where $y_{k-1} < y_k < y_{k+1}$. Two types were used: one based on the linear trial function concept borrowed from the finite element method, and one based on similar piecewise sinusoidal trial functions for which there is slope continuity — although at the price of zero slope at the extremities of the "element".

<u>Linear</u>

$$g_j(y) = \begin{cases} (y_k - y)/(y_k - y_{k-1}) & y_{k-1} < y < y_k \\ \\ (y - y_{k+1})/(y_{k+1} - y_k) & y_k < y < y_{k+1} \end{cases} \tag{C.13}$$

<u>Sinusoidal</u>

$$g_j(y) = \begin{cases} \sin^2(\frac{\pi}{2}(y - y_{k-1})/(y_k - y_{k-1})) & y_{k-1} < y < y_k \\ \\ \cos^2(\frac{\pi}{2}(y - y_k)/(y_{k+1} - y_k)) & y_k < y < y_{k+1} \end{cases} \tag{C.14}$$

Appendix D

Sample Cost Functions

D.1 Cost Function Definition for Inverse Design

$$\mathcal{F} = \sum_{k=1}^{N_{wing}} (p_k(\mathbf{U}_k) - p_k^*)^2 \Delta s_k \tag{D.1}$$

where p_k^* is the target pressure at node k, and N_{wing} is the number of points on the wing or airfoil surface.

D.2 Force Definitions

$$C_{L_i} = \frac{-F_x \sin\alpha_i + F_z \cos\alpha_i}{q_{\infty_i} S_{ref}} \tag{D.2}$$

$$C_{D_i} = \frac{F_{x_i} \cos\alpha_i + F_z \sin\alpha_i}{q_{\infty_i} S_{ref}} \tag{D.3}$$

In these equations, F_x and F_z are the axial and vertical forces which can be estimated from the numerical flow solution $\mathbf{U}$ by using a piecewise linear approximation to the pressure at the wall. In 2D, for example, we obtain,

$$F_x = \oint_{wall} p(1,0)^T \hat{n}\, ds = \sum_k \frac{\Delta z^k}{2} \sum_{j=1}^{2} p_j^k \tag{D.4}$$

while in 3D, we obtain,

$$F_x = \oint_{wall} p(1,0,0)^T \hat{n}\, dS = \sum_k \int_{\Omega^k} p\, dy\, dz = \sum_k \frac{A_x^k}{3} \sum_{j=1}^{3} p_j^k$$

where Ω^k are the triangular wall faces (3D) or line segments (2D) and j is an index for the vertices of Ω^k.