CONSTRAINED, MULTIPOINT SHAPE OPTIMIZATION FOR COMPLEX 3D CONFIGURATIONS

Jonathan Elliott and Jaume Peraire
Fluid Dynamics Research Laboratory
MIT Department of Aeronautics and Astronautics

Abstract

A method for performing three-dimensional, multipoint, lift-constrained drag minimization for complex aircraft configurations is presented. Parameters representing the aircraft geometry are the design variables used in the solution of an optimization problem. The compressible Euler equations for the flow are discretized on automatically generated unstructured meshes, and the sensitivities of the objective function and the constraints with respect to the design parameters are efficiently calculated using the discrete adjoint method. In addition, the solution algorithm has been parallelized making the approach feasible for practical applications. Several constrained minimization strategies are discussed and some numerical tests are carried out for a lift-constrained two-dimensional problem. The strategy found to perform best is demonstrated for the three-dimensional optimization of a wing–body configuration operating at transonic and supersonic conditions. It is concluded that Euler based optimization can be useful as a first step in the design process but most applications involving transonic flows do require viscous flow modelling to avoid unrealistic pressure distributions.

Nomenclature

$\mathbf{a}_j$	jth constraint gradient, $(\partial c_j/\partial\boldsymbol{\beta})^T$
$\mathbf{A}_k$	constraint jacobian with rows, $\mathbf{a}_j^T$
b	wing span
$\mathbf{B}_k$	approximation to Hessian $\mathbf{G}_k$ at iteration k
$\mathbf{B}_Z$	approximation to projected Hessian $\mathbf{G}_Z$
c	chord
c_j	constraint associated with condition j
$\tilde{c}_k$	linear model of constraint at iteration k
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
$f_j(x/c)$	component of mode h_{ij} dependent on x
$\mathcal{F}$	objective function
$\tilde{\mathcal{F}}$	quadratic model for objective function
$\mathbf{F}^j$	jth flux vector
$\mathbf{g}_k$	cost function gradient, $(\partial\mathcal{F}/\partial\boldsymbol{\beta})^T$
$\mathbf{g}_Z$	gradient in subspace Z, $\mathbf{Z}_k\mathbf{g}_k$
$h_i(x,y)$	design mode associated with β_i
P_Q	cost function with quadratic penalty term
$\mathbf{q}_k$	step to return design to feasible space
$\mathbf{R}$	vector of nodal residuals
$\mathbf{s}_k$	step size for iteration k
t	time
$\mathbf{U}$	nodal values of $\mathbf{V}$
$\mathbf{V}$	vector of conservative variables
x_1,x_2,x_3	Cartesian coordinates
$\mathbf{y}_k$	iteration change in $\mathbf{g}$, $\mathbf{g}_{k+1} - \mathbf{g}_k$
$\mathbf{Y}_k$	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 $\mathbf{a}_j$
α_j	angle of attack at condition j
$\boldsymbol{\beta}$	design variable vector
$\overline{\boldsymbol{\beta}}$	scaled design variable vector
δ	line search parameter
Λ_{LE}	leading edge sweep angle
Λ_{TE}	trailing edge sweep angle
μ	Mach wave angle

$\mathcal{L}$	Lagrangian	μ_k	Lagrange multiplier associated with c_k
M	Mach number	η	nondimensional spanwise coordinate, $\frac{2y}{b}$
M	number of constraints	ρ	penalty parameter associated with penalty term $\mathbf{c}^T\mathbf{c}$
N	number of design variables		
$\mathbf{p}_k$	search direction for iteration k	ϕ	twist
$\mathbf{p}_Y$	step in range space of $\mathbf{A}_k^T$ to ensure feasibility in line search k	ω_j	cost function weight for condition j
		$\boldsymbol{\psi}$	Lagrange multiplier associated with $\mathbf{R}$
$\mathbf{p}_Z$	search direction in subspace $\mathbf{Z}_k$, $\mathbf{Z}_k\mathbf{p}_k$	θ	local angle between meanline and x-axis

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, 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 [20] which offer the potential for aircraft designs with significantly improved fuel burn, through a combination of lower aircraft weight and reduced drag.

The main obstacle to formal 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 [12], but there have been other significant contributions [2, 11, 10, 19]. Adjoint methods for the calculation of sensitivities can be formulated in two different forms. In the first approach, called the variational sensitivity analysis approach, 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. In the second approach, called the discrete sensitivity analysis approach, 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 [14] 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 single-block structured 3D Navier-Stokes equations. 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 [18] 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 underway to incorporate viscous effects.

Newman et al [17] 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 [16] and aeroelastic flow analysis [15]. The large storage and computational cost requirements associated with the implicit solution limited the usefulness of the approach in 3D. In a previous paper [5], the authors 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 [6].

This paper reports on the further development of the approach presented in [6] to incorporate a 3D lift-constrained, multipoint drag minimization capability based on unstructured grids. Solution of the problem using these particular features 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; (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 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. Of course, inclusion of viscous effects, in transonic flows, is vital for many of these tasks and this could be done either using a coupled viscous/inviscid method or by solving the full Navier-Stokes equations.

In the present study we neglect viscous effects for the time being and focus on the development of a drag minimization capability based on the 3D Euler equations for unstructured grids with specific focus given to the means by which the constraints are incorporated. The main conclusion of this study 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.

The basic unconstrained algorithm is described in detail elsewhere [5, 6]. Here, we shall focus on the more realistic constrained problem. However, we will also give brief descriptions of the flow analysis algorithm (Section 2.1), the optimization problem (Section 2.2), and the sensitivity analysis calculation (Section 2.3). The constrained minimization algorithm (Section 2.4) and some implementation issues are also described (Section 2.5). The algorithms presented are illustrated with some representative examples (Section 3), and finally, some conclusions and recommendations for future work are also presented (Section 4).

2 Algorithms

2.1 Governing Equations

The Euler equations, governing the flow field, are solved using an unstructured grid finite volume algorithm. Steady state solutions are obtained by solving the time dependent problem with an arbitrary initial condition and advancing the solution until convergence. Details of the discretization process in two and three dimensions can be found in [6]. Here, only a brief description of the algorithm is given for notational purposes. The Euler equations written in conservative form read

$$\frac{\partial \mathbf{V}}{\partial t} + \sum_{j=1}^{n_d} \frac{\partial \mathbf{F}^j}{\partial x_j} = 0 \tag{1}$$

where $\mathbf{V}$ is the vector of conserved quantities (mass, momentum and energy), $\mathbf{F}^j$ are the associated fluxes in the x_j coordinate direction, and n_d is the number of spatial dimensions (2 or 3). The spatial discretization of the above equations is carried out on a grid of triangular/tetrahedral elements. The unknown vector $\mathbf{V}$ is assumed to vary linearly over each element and the finite volume method, with control volumes made up of all the elements that share a common vertex, is used to produced the final space discretized system. The vector of nodal unknowns $\mathbf{U}$, containing the approximations to the values of $\mathbf{V}$ at each of the vertices, satisfies a semi–discrete equation set of the form

$$\frac{d\mathbf{U}}{dt} + \mathbf{R}(\mathbf{U}) = \mathbf{0} \tag{2}$$

where, through the use of an edge-based formulation [9], the residual at node i can be written as

$$\mathbf{R}_i = \sum_{ik=1}^{K_i} [\mathbf{R}]_{ik}. \tag{3}$$

Here ik represents the edge connecting nodes i and k, and K_i is the total number of nodes connected to node i. Finally, the time discretization of (2) is performed using an explicit multi-stage Runge-Kutta scheme. Since only the steady state solution satisfying $\mathbf{R}(\mathbf{U}) = \mathbf{0}$ is of interest here, local timestepping, whereby each equation is advanced at the maximum allowable timestep for stability, is employed.

The residual $\mathbf{R}(\mathbf{U})$ is made up of a central differencing, or Galerkin, approximation to the spatial derivative terms of the governing equations, and a stabilizing dissipative term. The dissipative term is designed to be of the same magnitude as the truncation error in smooth regions of the flow and sufficiently large near discontinuities to allow for effective shock capturing. For a given node, the central difference term depends only on the solution at the nearest neighbours, whereas the dissipative term has a larger stencil and its computation requires the solution at all points connected to the nearest neighbours. This additional information is required to monitor the smoothness of the solution [1] and determine the proper amount and type of dissipation required. In the current algorithm, dissipation is added selectively for each characteristic variable. The overall scheme is conservative, second order accurate for smooth flows, and can be shown to possess the local extremum diminishing property [13].

2.2 Optimization Problem

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,

$$\boldsymbol{\beta}^{opt} = \arg\min_{\boldsymbol{\beta}} \mathcal{F}(\boldsymbol{\beta}) \tag{4}$$

subject to M constraints of the form

$$c_j(\boldsymbol{\beta}) = 0, \qquad j = 1 \dots M \tag{5}$$

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

$$\mathcal{F} = \sum_{j=1}^{M} \omega_j C_{D_j}(\boldsymbol{\beta}) \tag{6}$$

and the N components of the design vector $\boldsymbol{\beta}$ are the parameters used to represent the aircraft geometry and the M angles of attack corresponding to each flow condition. The constraints considered are the lift at each operating point. Thus,

$$c_j(\boldsymbol{\beta}) \equiv C_{L_j}(\boldsymbol{\beta}) - C_{L_j}^{targ} = 0, \qquad j = 1 \dots M. \tag{7}$$

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 (4) 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 (7). 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 equations. One obvious possibility is to approximate the sensitivities using finite differences which require the computation of flowfields, satisfying the constraints, 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

$$\mathbf{R}_j(\boldsymbol{\beta}, \mathbf{U}_j) = 0, \qquad j = 1 \dots M. \tag{8}$$

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

$$\boldsymbol{\beta}^{opt} = \arg\min_{\boldsymbol{\beta}} \mathcal{F}(\boldsymbol{\beta}, \mathbf{U}_1(\boldsymbol{\beta}), \mathbf{U}_2(\boldsymbol{\beta}), \dots \mathbf{U}_M(\boldsymbol{\beta})) \tag{9}$$

subject to the M original constraints of equation (5) plus the M additional constraints of equation (8).

2.3 Sensitivity Calculation

2.3.1 Direct and Adjoint Methods

The procedure employed to evaluate the sensitivities is best illustrated by considering the unconstrained problem (4) which can be written as problem (9) subject to only M constraints given by (8). Procedures for incorporating the original M constraints (7) are discussed afterwards in Section 2.3.2.

Since the drag coefficient at flow condition j, C_{D_j}, only depends on $\boldsymbol{\beta}$ and $\mathbf{U}_j$, we write

$$\frac{d\mathcal{F}}{d\boldsymbol{\beta}} = \sum_{j=0}^{M} \omega_j \frac{d}{d\boldsymbol{\beta}} C_{D_j}(\boldsymbol{\beta}, \mathbf{U}_j(\boldsymbol{\beta})). \tag{10}$$

and the sensitivities of C_{D_j} are therefore given by

$$\frac{dC_{D_j}}{d\boldsymbol{\beta}} = \frac{\partial C_{D_j}}{\partial \boldsymbol{\beta}} + \frac{\partial C_{D_j}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \boldsymbol{\beta}}. \tag{11}$$

The first term on the right-hand side of (11) is relatively easy to compute. This term represents the variation of C_{D_j} when $\mathbf{U}_j$ is kept fixed. For instance, if the geometry or the angle of attack are changed, the surface integrations required to compute the drag coefficient and the decomposition of the total force into lift and drag components will also change. To calculate the second term, we utilize the implicit definition of $\mathbf{U}(\boldsymbol{\beta})$ given by (8) and write, for $j = 1, \dots M$,

$$\frac{\partial \mathbf{R}_j}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \boldsymbol{\beta}} = -\frac{\partial \mathbf{R}_j}{\partial \boldsymbol{\beta}} \tag{12}$$

with all the terms evaluated at $\mathbf{U} = \mathbf{U}_j$.

Two approaches referred to as the direct and adjoint methods can be followed at this point. The direct method involves *direct* solution of equation (12) and substitution of the resulting vector $\partial \mathbf{U}/\partial \beta$ into equation (11). Note that equation (12) represents a set of N decoupled problems corresponding to each of the components of $\boldsymbol{\beta}$. The means by which equation (12) is solved is discussed below.

In the adjoint method, equations (12) and (11) are combined to give:

$$\frac{dC_{D_j}}{d\beta} = \frac{\partial C_{D_j}}{\partial \beta} - \left[\frac{\partial C_{D_j}}{\partial \mathbf{U}}\right]^T \left[\frac{\partial \mathbf{R}_j}{\partial \mathbf{U}}\right]^{-1} \left[\frac{\partial \mathbf{R}_j}{\partial \beta}\right] \tag{13}$$

and an intermediate variable, or *adjoint* , vector $\boldsymbol{\psi}_j$ is computed by solving the following problem

$$\left[\frac{\partial \mathbf{R}_j}{\partial \mathbf{U}}\right]^T \boldsymbol{\psi}_j = -\left\{\frac{\partial C_{D_j}}{\partial \mathbf{U}}\right\}^T. \tag{14}$$

This allows the sensitivities to be calculated as

$$\frac{dC_{D_j}}{d\beta} = \frac{\partial C_{D_j}}{\partial \boldsymbol{\beta}} + \boldsymbol{\psi}_j^T \left\{\frac{\partial \mathbf{R}_j}{\partial \boldsymbol{\beta}}\right\} \tag{15}$$

It is noted that equations (8) and (14) correspond to the stationarity conditions of a Lagrangian formed by augmenting the cost function with the constraint terms

$$\mathcal{L}(\boldsymbol{\beta}, \mathbf{U}_1, \dots, \mathbf{U}_M, \boldsymbol{\psi}_1, \dots, \boldsymbol{\psi}_2) = \mathcal{F}(\boldsymbol{\beta}, \mathbf{U}_1, \dots, \mathbf{U}_M) + \sum_{j=0}^{M} \boldsymbol{\psi}_j^T \mathbf{R}_j(\mathbf{U}, \boldsymbol{\beta}). \tag{16}$$

With this interpretation, the intermediate variables ψ_j play the role of Lagrange multipliers [10]. The third optimality condition, requiring the variations of $\mathcal{L}$ with respect to β to be zero, leads to

$$\frac{d\mathcal{L}}{d\boldsymbol{\beta}} = \sum_{j=0}^{M} \omega_j \frac{\partial C_{D_j}}{\partial \boldsymbol{\beta}} + \sum_{j=0}^{M} \boldsymbol{\psi}_j^T \left\{ \frac{d\mathbf{R}_j}{d\boldsymbol{\beta}} \right\} = 0. \tag{17}$$

The equation systems (12) and (14) are linear and of the same size as the flow solution problem (8). A convenient way to solve these equations is to use the same relaxation scheme (2) used to solve (8). This is done by introducing an artificial unsteady term into, for example, the adjoint equation for flow condition j

$$\frac{d\boldsymbol{\psi}_j}{dt} + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \boldsymbol{\psi}_j - \left[\frac{\partial C_{D_j}}{\partial \mathbf{U}}\right]^T = 0 \tag{18}$$

It is apparent that the stability characteristics of this modified equation are identical to those of the linearized version of (2) and therefore both equations can be integrated in time using the same timstepping.

The advantage of the adjoint method over the direct method is immediately apparent. The solution of equations (14) is independent of $\boldsymbol{\beta}$. Therefore solving this problem *just once* gives the adjoint solution which can be readily used in (15) to provides sensitivities to all components of $\boldsymbol{\beta}$. However, it should be noted that for some multi–disciplinary optimization problems the number of cost functions required may exceed the number of design variables. In this case, the adjoint method would require an adjoint solution for each objective function, and therefore the direct method, which requires one solution per design variable, is advantageous. This situation is conceivable, for instance, when the Global Sensitivity matrix approach [20] is employed.

In this work, we have used the adjoint approach and the equations for the adjoint variables have been solved using exactly the same multi-stage explicit time-stepping employed for the flow analysis.

2.3.2 Incorporation of Constraints

The obvious treatment of the additional constraints (7) is to further augment the Lagrangian given in Equation (16) with the additional constraints:

$$\mathcal{L}(\boldsymbol{\beta}, \mathbf{U}_1, \ldots, \mathbf{U}_M, \boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_M, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_M) = \mathcal{F}(\boldsymbol{\beta}, \mathbf{U}_1, \ldots, \mathbf{U}_M) + \sum_{j=0}^{M} \boldsymbol{\psi}_j^T \mathbf{R}_j(\mathbf{U}, \boldsymbol{\beta}) + \sum_{j=0}^{M} \boldsymbol{\mu}_j^T \mathbf{c}_j(\mathbf{U}, \boldsymbol{\beta}) \tag{19}$$

In this case, making the Lagrangian stationary with respect to the newly introduced Lagrange multipliers $\boldsymbol{\mu}_j$ will produce the desired constraint equations. The main difficulty with this approach is its numerical implementation. The equations that need to be solved are fully coupled and their direct solution is impractical. An alterantive approach is to ensure that the constraints are satisfied by simultaneously solving equations (8) and (7) for $\mathbf{U}_j$ and the corresponding angle of attack at each operating point. In this way, the original form of the Lagrangian (16) is kept and β is effectively shortened to include only the variables representing the geometry. This approach is certainly more attractive but it no longer allows, in a straightforward manner, to use the relaxation scheme of equation (2) since there is no obvious way to augment the constraint equations (7) with a time dependent term.

A more expedient method to incorporate the constraints is to use a penalty method. This approach introduces a modified cost function which incorporates a quadratic penalty term [8] of the form

$$P_Q(\boldsymbol{\beta}, \rho) = \mathcal{F}(\boldsymbol{\beta}) + \rho \mathbf{c}(\boldsymbol{\beta})^T \mathbf{c}(\boldsymbol{\beta}), \tag{20}$$

where ρ is a scalar penalty parameter. The advantage of this method is that it is simple to implement. The disadvantage is that the optimum, $\boldsymbol{\beta}^{opt}$, depends on ρ, although

$$\lim_{\rho\to\infty} \boldsymbol{\beta}^{opt}(\rho) = \boldsymbol{\beta}^{opt} \tag{21}$$

as shown in [8]. Unfortunately, for large ρ, ill-conditioning is inevitable.

Another alternative is to work with an augmented Lagrangian whereby penalty terms similar to that in Equation (20) are added to the Lagrangian in an attempt to ensure that the Hessian of the augmented Lagrangian is positive definite [8],

$$P_Q(\boldsymbol{\beta}, \rho) = \mathcal{F}(\boldsymbol{\beta}) + \rho \mathbf{c}(\boldsymbol{\beta})^T \mathbf{c}(\boldsymbol{\beta}) + \boldsymbol{\mu}^T \mathbf{c}(\boldsymbol{\beta}). \tag{22}$$

This has the advantage (over augmenting just the cost function) that ill-conditioning is avoided for large ρ. A means for updating the Lagrange multiplier estimate $\boldsymbol{\mu}$ is provided in [8]. This approach requires the solution of a nested series of subproblems to be solved based on estimates of the Lagrange multipliers and until sufficiently large ρ is obtained.

The above approaches lead to sensitivities which already incorporate the constraints and therefore have the advantage that an unconstrained optimization algorithm can be used. In this work however, we have adopted the approach of incorporating the constraints (7) during the descent, or optimization, stage. As we shall see next, this requires the calculation of the sensitivity of the constraints with respect to the design parameters which is readily accomplished using the procedure presented above.

Consider the lift coefficient as a function of the design variables, directly and through the state vector, that is, $C_{L_j}(\boldsymbol{\beta}, \mathbf{U}_j)$, with $\mathbf{U}_j(\boldsymbol{\beta})$ defined implicitly by (8), and write

$$\frac{dC_{L_j}}{d\boldsymbol{\beta}} = \frac{\partial C_{L_j}}{\partial \boldsymbol{\beta}} + \frac{\partial C_{L_j}}{\partial \mathbf{U}_j}\frac{\partial \mathbf{U}_j}{\partial \boldsymbol{\beta}}$$

These sensitivities can be calculated using identical steps to those followed to calculate drag coefficient sensitivities (15).

2.4 Constrained Optimization

The approach taken in this work 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,

$$\tilde{\mathbf{c}}_k = \mathbf{c}_k + \mathbf{A}_k \boldsymbol{\beta} \tag{23}$$

where $\mathbf{A}_k$ is a matrix whose rows are the constraint gradients, $\mathbf{a}_j$, 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 [4, 8]. 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 constaints. The search direction thus obtained can be expressed as

$$\mathbf{p}_k = -\mathbf{g}_k + \mathbf{A}_k^T[\mathbf{A}_k\mathbf{A}_k^T]^{-1}\mathbf{A}_k\mathbf{g}_k \tag{24}$$

which can be alternatively written as

$$\mathbf{p}_k = -\mathbf{Z}_k\mathbf{Z}_k^T\mathbf{g}_k \tag{25}$$

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 sucessive quadratic models are generated following the quadratic/cubic backtracking algorithm described in Dennis et al [3].

The Constrained Steepest Descent algorithm used in this work varies from the "sequence-of-linear-subproblems" algorithm suggested in [8] 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 (23). The constraint gradients $\mathbf{a}_j$ are computed by solving the constraint adjoint equation — analogous to Equation (14) — 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,

$$\mathbf{q}_k = \mathbf{A}_k^T\mathbf{p}_Y \tag{26}$$

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

$$\tilde{\mathbf{c}}_{\text{feasible}} = \mathbf{c}_k + \mathbf{A}_k\mathbf{q}_k = 0. \tag{27}$$

Elimination produces

$$\mathbf{p}_Y = -[\mathbf{A}_k\mathbf{A}_k^T]^{-1}\mathbf{c}_k \tag{28}$$

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

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \delta_k\mathbf{p}_k + \mathbf{q}_k. \tag{29}$$

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. The details of the algorithm are described in [8].

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

$$\mathbf{B}_{Z_{k+1}} = \mathbf{B}_{Z_k} + \frac{1}{\mathbf{g}_{Z_k}^T \mathbf{p}_{Z_k}} \mathbf{g}_{Z_k} \mathbf{g}_{Z_k}^T + \frac{1}{\delta_k \mathbf{y}_{Z_k}^T \mathbf{p}_{Z_k}} \mathbf{y}_{Z_k} \mathbf{y}_{Z_k}^T \tag{30}$$

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

$$\mathbf{B}_{Z_k} \mathbf{p}_{Z_k} = -\mathbf{g}_{Z_k} \tag{31}$$

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, $\mathbf{G}$, 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 [8].) For the current algorithm, the fact that $\mathbf{Z}_k$ changes causes $\mathbf{B}_k$ to more slowly approach the actual Hessian $\mathbf{G}$, 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).

2.5 Implementation issues

The solution algorithms for the flow and adjoint variables have been parallelized using domain decomposition and explicit message passing which is supported using the MPI libraries [21]. The domain is decomposed using a coordinate bisection algorithm such that edges belong to a single subdomain whereas points can be shared between several subdomains. For the larger problems and domain decomposition strategy considered here the efficiency for the parallel algorithm has been found to be higher than 95% using up to twenty processors.

The computer implementation is such that the storage of $\partial\mathbf{R}/\partial\mathbf{U}$ on an edge basis, rather than having to calculate it at every stage of every Runge-Kutta iteration, is optional. In 3D, the storage of $\partial\mathbf{R}/\partial\mathbf{U}$, even on an edge basis, incurs a factor of 10 increase in memory requirements. However, this results in a factor of four speedup. Typically, when working with parallel machines having distributed memory, storage requirements are not a problem and therefore storing $\partial\mathbf{R}/\partial\mathbf{U}$ is the preferred option. For the wing-body geometry, which we optimized for this paper, the increased memory cost required 13 RS6000 nodes (with grids containing 211,000 nodes and 1.13 million elements) with 128MW of memory on the IBM SP2.

3 Results

Results have been obtained that demonstrate that the two descent algorithms described in the previous section 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.

2D optimization 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 (6)) 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 in Table 1 and the surface equation, $z(x)$, is given in terms of the baseline geometry, $z^0(x)$, and the shape design functions as

$$z(x) = z^0(x) + \sum_{i=3}^{6} \beta_i h_i(x) \tag{32}$$

for all points on surfaces affected by the design variables. It should be noted that all the shape variables

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 1: Airfoil Parameterization for 2D problem

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. This scaling is performed using

$$\overline{\beta}_{k_i} = g_{k=0_i} \beta_{k_i} \tag{33}$$

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 [4].

At each iteration a line search is performed. 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 1. Note that these are the values found at the end of each line search and after projection back into feasible space using Equations (28) and (29).

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 2-4. 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 airoil would remain shock-free for the grid-converged 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 5) 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), this strategy ends up costing less in terms of floating point operations. The geometry and pressure coefficient evolution shown in Figures 6-8 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.

3D optimization 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 $\phi = 0°$ and is uniformly uncambered across the span. The baseline grid is shown in Figure 9. It contains 1.13 million elements and 211,000 points.

The baseline transonic solution ($M_{\infty_1} = 0.9$) is shown in Figure 10. The sectional pressures are also plotted in Figure 15. To achieve the required C_{L_2} 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 "lambda" 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.

The baseline supersonic solution ($M_{\infty_1} = 1.6$) is shown in Figure 11. 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 $\mu = sin^{-1}(\frac{1}{M_\infty}) = 38.7°$ 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 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 $f_i(x/c)$ and spanwise $e_j(y)$ functions are summarized in Table 2. The construction of the actual design variables from $\mathbf{e}$ and $\mathbf{f}$ is also shown in Table 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 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 cleancut here as in the 2D case. The source of the problem was that the two flight conditions were on either side of the sonic 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 (33) resulted in movement in the wrong direction for some design variables. Replacement of Equation (33) 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 is the norm 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 12 and 13 for either condition while sectional geometry and C_p evolution are plotted in Figures 16-17. A summary of the optimization exercise is shown in Figure 14. After seven line searches, a 5.2% reduction in the cost function is found.

The cost function reduction breakdown is summarized in Table 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
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%	$+0.87\%$

Table 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 12 and 15. 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 15 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 may 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 11-13 and Figure 17. 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. However, the resultant C_D increase is evidently far lower than the decrease found at Condition 1 ($M_\infty = 0.9$).

4 Conclusion

The overall problem addressed was the extension of the optimization system reported in the authors' previous papers on modal inverse design optimization [5, 6], to the arguably more useful task of multipoint drag minimization at constrained lift. To this end, the required new features were developed and the system was tested on a wing/body configuration operating at transonic and supersonic flight conditions to determine whether the process would produce a credible optimal geometry in a timely manner. From the results presented herein, it is concluded that (1) lift-constrained drag minimization is a practical capability of the optimization system (2) viscous effects should be included in the optimization process in order that severe adverse pressure gradients be avoided in the optimal solution.

Additionally, it was found that (1) the subspace BFGS algorithm was superior to the subspace Steepest Descent algorithm; (2) full convergence of the adjoint calculations is not necessary until the latter stages of the optimization process; (3) reasonable scaling of the design variables is vital to ensure timely conclusion of the optimization process; and (4) regridding may be required if large design variable movements are demanded, although for the grid resolutions examined, this allowed seamless continuation of the optimization process.

As a direct result of this study, it now appears that 3D multipoint drag minimization on unstructured grids is a capability that is well in hand for compressible, inviscid flow. To allow the system to become a practical industrial tool for aerodynamic design, it seems that the next factors to be addressed are (1) viscous effects should be included and (2) a means for adjusting α to satisfy the C_L constraint should be incorporated directly into the flow analysis algorithm, which should result in cost function sensitivities in which the constraints are already included.

Acknowledgements

Thanks are in order to Professor Mark Drela for continuing invaluable assistance at various stages of this project. The authors would like to thank the Flight Test Information Systems Branch of the NASA Dryden Research Center (under grant NAG4-100) and the Aerothermodynamics Branch of the NASA Langley Research Center (under grant NAG-1-1587) for partially supporting this research.

References

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

[2] 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.

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

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

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

[6] J. Elliott and J. Peraire. Practical 3D Aerodynamic Design and Optimization using Unstructured Meshes. *AIAA Journal*, 35, vol. 9:1479-1486, 1997.

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

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

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

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

[11] 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.

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

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

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

[15] J. Newman, P.A.Newman, A.C.Taylor, and G.-W. Hou. Efficient Nonlinear Static Aeroelastic Wing Analysis. Submitted to Computers and Fluids, December, 1996.

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

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

[18] 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.

[19] G. Shubin and P. 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.

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

[21] Message Passing Interface Forum. MPI: A message-passing interface standard Interantional Journal of Supercomputer Applications 8(3/4), 1994.

iter	res(C_{D_1})	res(C_{L_1})	res(C_{D_2})	res(C_{L_2})
0	3.3e-13	4.6e-12	2.1e-13	6.2e-13
1	1.6e-13	3.0e-12	4.5e-14	2.5e-12
2	4.0e-14	1.3e-12	5.8e-15	1.8e-13
3	3.0e-14	7.0e-13	5.0e-14	2.0e-13
4	7.0e-14	7.0e-13	5.0e-15	2.0e-14
5	2.0e-14	1.0e-13	3.0e-16	2.0e-13
6	7.6e-15	4.1e-14	1.2e-15	8.0e-15

Table 4: Adjoint Convergence levels as a function of line search iteration

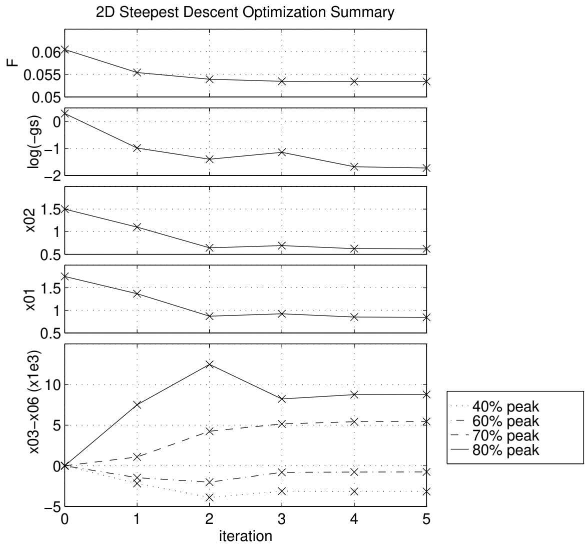

Figure 1: Cost function and design variable evolution for 2D steepest descent optimization exercise

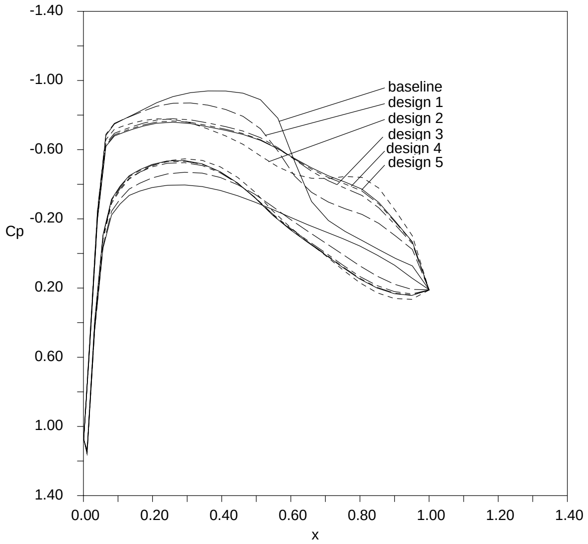

Figure 3: Condition 1 C_p evolution for 2D steepest descent optimization exercise

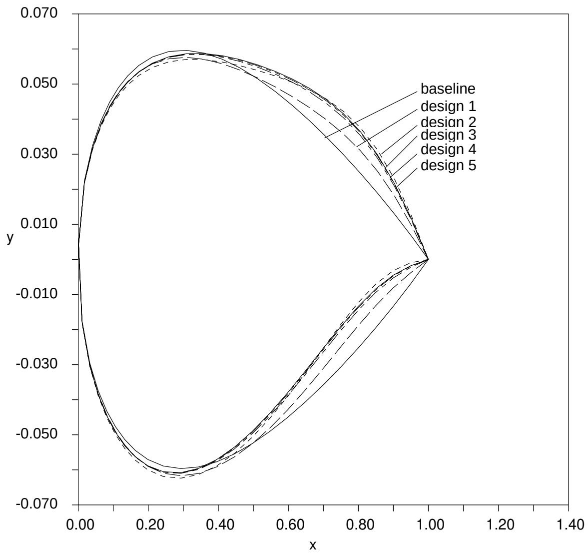

Figure 2: Geometry evolution for 2D steepest descent optimization exercise

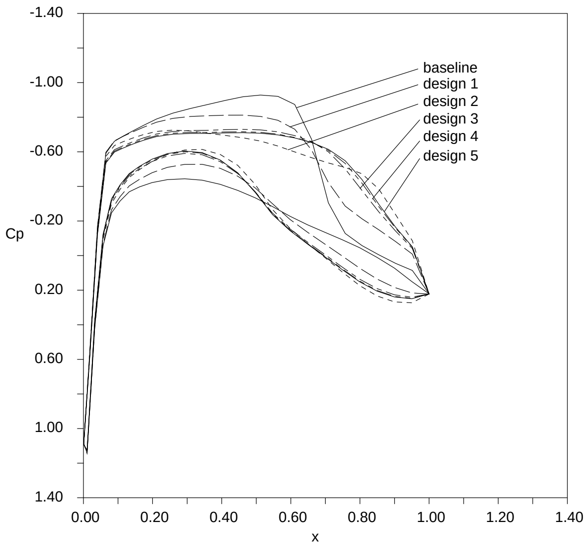

Figure 4: Condition 2 C_p evolution for 2D steepest descent optimization exercise

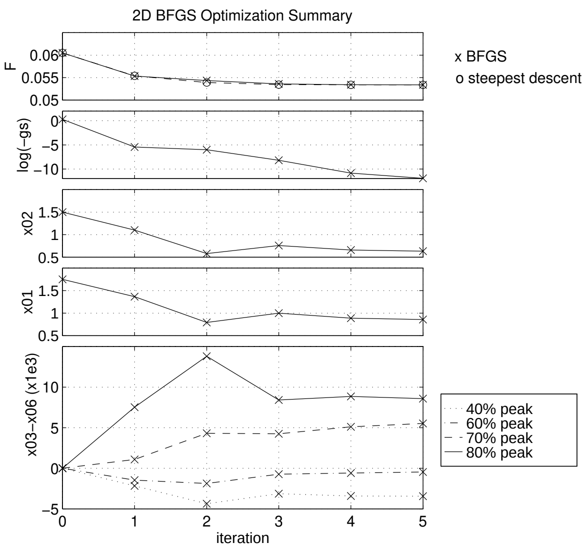

Figure 5: Cost function and design variable evolution for 2D BFGS optimization exercise

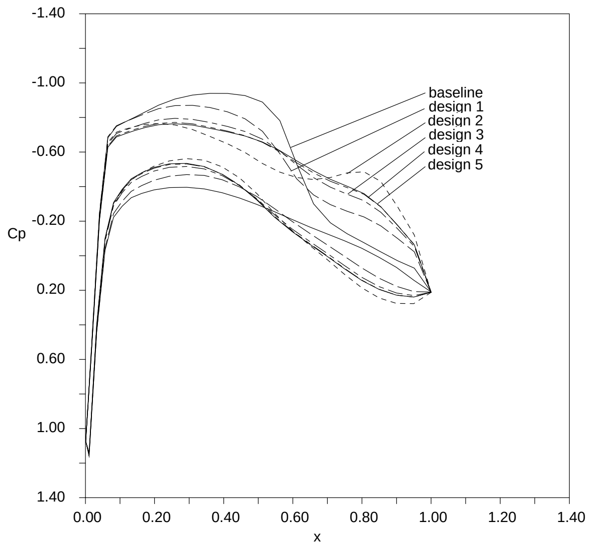

Figure 7: Condition 1 C_p evolution for 2D BFGS optimization exercise

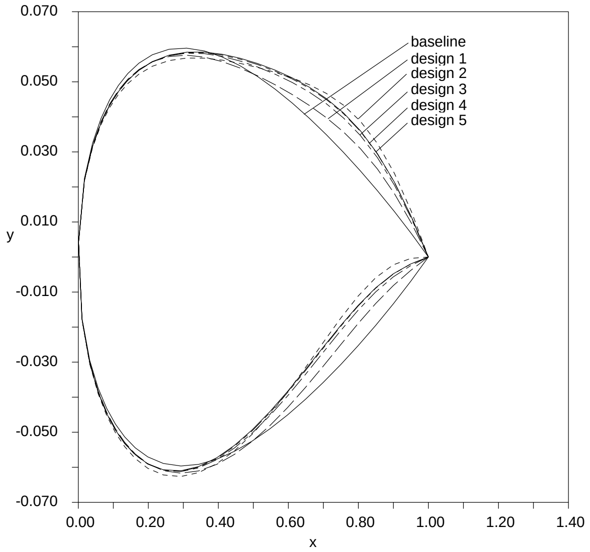

Figure 6: Geometry evolution for 2D BFGS optimization exercise

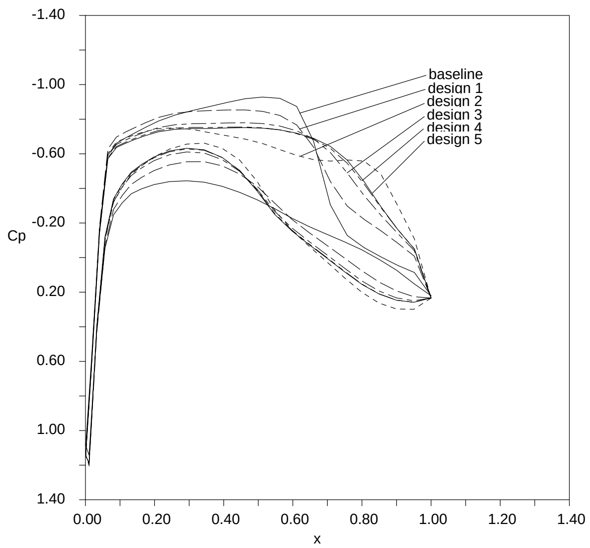

Figure 8: Condition 2 C_p evolution for 2D BFGS optimization exercise