PRACTICAL 3D AERODYNAMIC DESIGN AND OPTIMIZATION USING UNSTRUCTURED MESHES

Jonathan Elliott and Jaime Peraire

Department of Aeronautics and Astronautics
Massachusetts Institute of Technology

77 Massachusetts Avenue
Cambridge, MA 02139
Room 37-471

PH: (617) 253-5850
FAX: (617) 258-5143
`jkelliot@mit.edu`

Abstract

A methodology for performing optimization on 3D unstructured grids based on the Euler equations is presented. The same, low-memory-cost explicit relaxation algorithm is used to resolve the discrete equations which govern the flow, linearized direct and adjoint problems. The analysis scheme is a high resolution Local-Extremum-Diminishing (LED) type scheme which uses Roe decomposition for the dissipative fluxes. Mesh movement is performed in such a way that optimization of arbitrary geometries is allowed. The parallelization of the algorithm, which permits its extension to optimization of realistic, complete aircraft geometries is presented. Two sample optimizations are performed. The first is the inverse design of a transonic wing/body configuration using a surface target pressure distribution found by analyzing the geometry for known design variable deflections. The second exercise is the inverse design of a business jet configuration consisting of wing, body, strut, nacelle, horizontal fin and vertical fin. The surface target pressure distribution in this case is provided by an analysis of the configuration with no strut-nacelle.

1 Introduction

Advances in both computer technology and algorithms have allowed CFD to occupy a vital part of the aerospace, marine and automotive vehicle design processes, significantly reducing the costs associated with testing of physical models. However, it is possible that even greater dividends are being missed due to the absence, to a large extent, of formal aerodynamic/hydrodynamic optimization in these design processes. Indeed the coupling of aerodynamic sensitivities with, for example, structural weight and propulsion sensitivities in a Multidisciplinary

Design Optimization (MDO) process, promises even richer rewards, and is thought to be vital to ensure the viability of future vehicles such as the High Speed Commercial Transport (HSCT).

The main obstacle to aerodynamic optimization has been the high cost associated with the calculation of sensitivities. However, in recent years impressive efforts have been made using structured grids that allow for efficient sensitivity calculation [1, 2].

For the solution of flowfields over complex domains, unstructured grid methods have several advantages over their alternative structured counterparts. For example, the time associated with grid generation for multiblock schemes and the errors associated with lack of conservation and grid generation problems (to a lesser extent) for Chimera-type schemes limit their usefulness. Unstructured mesh methods using tetrahedra offer the ability to model flows around complex geometries and can incorporate mesh adaptive procedures. Weaknesses associated with resolving boundary layers and viscous wakes are currently being tackled and it is believed that unstructured methods will have viscous resolution capabilities matching those of structured grids in the not too distant future.

Transonic/supersonic wing design in the presence of strut-nacelles and aircraft high lift system design are examples of tasks associated with inherently complex geometric domains for which unstructured flow solvers, with automatic mesh generation, have a decided advantage. Furthermore, the capability to refine and derefine unstructured meshes, introduces the possibility of distributing the mesh degrees of freedom more efficiently and therefore concentrating elements where they are required to reduce the local error to a desired level.

Hence a capability to perform optimization using unstructured flow solvers is perceived to be an important one. We present herein a capability for performing practical 3D optimization on unstructured grids in which the cost of the gradient calculation is practically *independent of the number of design variables.* This is done using the discrete adjoint method [12, 10, 9, 8, 20] and solving the resulting system with the same explicit multistage timestepping algorithm used to solve the flow equations.

For structured grids, generation of analytic sensitivities based on the Euler equations in 2D and 3D has recently been performed using the analytic adjoint approach [12, 17, 18], in which flow and adjoint equations are discretized separately. The same relaxation algorithm was used to solve flow and adjoint equations. In [12] an alternating direction method with upwind biasing and multigrid was used to drive the residuals to zero. In [17] an explicit multistage relaxation scheme was used in a scheme incorporating scalar dissipation with pressure switching to allow shock capturing.

Newman, Taylor and Burgreen presented a capability to generate 2D [14], and recently 3D [13], unstructured sensitivities for a 2nd order scheme using Van Leer flux vector splitting. This was done using an incremental iterative implicit time-marching scheme. They used the discrete sensitivity analysis approach in which differentiation of the discrete flow analysis scheme is performed directly. They extended this capability to 2nd order by using the 1st order Jacobian on the left hand side with its lower required core memory (due to the smaller stencil) and the second order Jacobian on the right hand side.

A 3D unstructured sensitivity capability based on the Euler equations was also reported in [5] by the authors. The extension of this capability to the optimization of realistic, complete aircraft geometries is discussed in this paper. This extension has been made possible (for practical design times) by the parallelization of the algorithm.

All sensitivity analyses reported herein use the discrete sensitivity analysis approach. Introduction of an artificial time dependent term into the adjoint and sensitivity equations permits their efficient solution by using an identical relaxation scheme to the flow analysis scheme. The flow variables are discretized on 2D and 3D unstructured grids. LED properties are obtained by introducing just enough dissipation to upwind each characteristic component through the use of a Roe-averaged flux jacobian [11]. Minmod-limited antidiffusion is added to improve the accuracy of the scheme to 2nd order in smooth regions of the solution. Furthermore the mesh movement algorithm can handle arbitrary geometries and consists of finding the equilibrium position of the unstructured mesh considered as a system of interconnected springs.

Section 2 presents the details of the flow and sensitivity analysis algorithms. Section 3 presents applications of these algorithms to sample optimization exercises. Validation of the sensitivities produced by the algorithm has been presented in [5] and here we focus on the the details of the optimization exercises. We summarize the capabilities developed and indicate future research directions in Section 4. It should also be noted that the algorithm extends trivially to optimization of other more general cost functions such as drag.

2 Flow and Sensitivity Analysis Algorithms

Flow Analysis

For illustration purposes, the algorithm is described based on the 2D algorithm. The 3D algorithm is a direct extension thereof. The integral form of the Euler equations reads

$$\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t} dx dy = - \int_{\Omega} \nabla \cdot \vec{\mathbf{F}} dx dy = - \oint_{\Gamma} \vec{\mathbf{F}} \cdot \hat{n} ds \tag{1}$$

where,

$$\mathbf{U} = \left[\ \rho, \rho u, \rho v, \rho E \ \right]^T \tag{2}$$

and Ω is a closed control volume with boundary, Γ and $\vec{\mathbf{F}}$ is the usual flux tensor. The discretization of $\mathbf{U}$ and $\vec{\mathbf{F}}$ 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 consisting of all triangles having vertex i and performing exact integration around the outer boundary of this control volume. For illustrative purposes, the semidiscrete form is shown here to be advanced in time using a Forward Euler scheme. A multistage Runge-Kutta was used in the reported computations. After discretization, the following set of equations results.

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n + \sigma(\frac{\Delta t}{A})_i \mathbf{R}_i(\mathbf{U}^n) \tag{3}$$

where a cell-dependent timestep Δt_i has been used for convergence acceleration, A_i is the area of the control volume and σ represents the local CFL number. The residual at a point can be calculated as a sum over all the edges that contain that point [16].

$$\mathbf{R}_i = \sum_{ik=1}^{K} [\mathbf{R}]_{ik} \tag{4}$$

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

$$[\mathbf{R}]_{ik} = -\frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_k}{2} \cdot \vec{S}_{ik} \tag{5}$$

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

$$\vec{S}_{ik} = (y_{k+1} - y_{k-1}, x_{k-1} - x_{k+1}) \tag{6}$$

The residual contribution for a wall boundary edge is

$$[\mathbf{R}_i^b]_{il} = -\{\frac{p_i + p_l}{2}(\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}) + \vec{\mathbf{F}}_i\} \cdot \vec{S}_b - \frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_l}{2} \cdot \vec{S}_{l,l+1} \tag{7}$$

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

$$\vec{S}_b = (y_l - y_i, x_i - x_l) \tag{8}$$

For a farfield boundary edge the contribution to the residual becomes

$$[\mathbf{R}_i^b]_{il} = -\frac{\vec{\mathbf{F}}_i^b + \vec{\mathbf{F}}_l^b + 2\vec{\mathbf{F}}_i}{2} \cdot \vec{S}_b - \frac{\vec{\mathbf{F}}_i + \vec{\mathbf{F}}_l}{2} \cdot \vec{S}_{l,l+1} \tag{9}$$

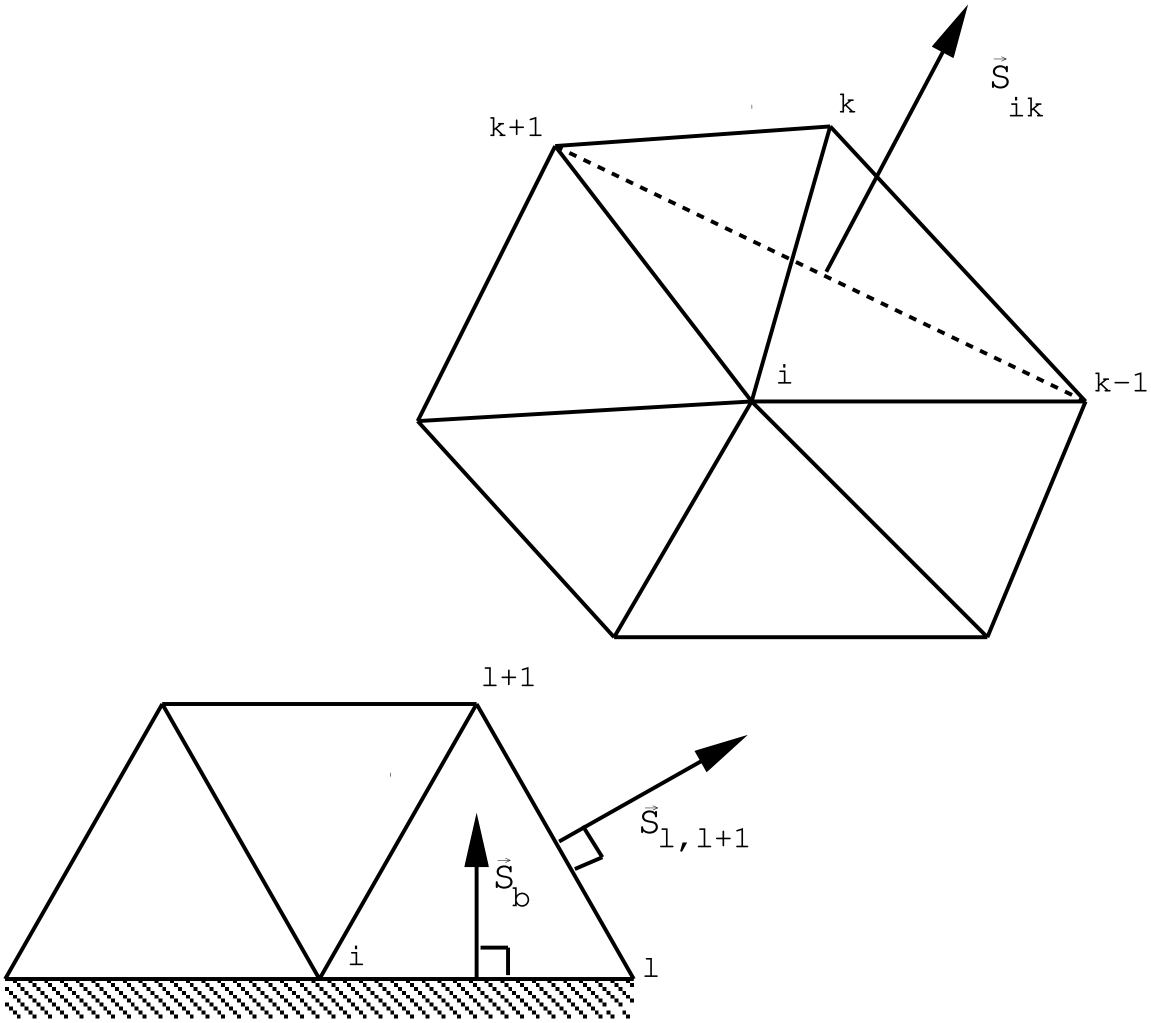

Figure 1: Area vectors for boundary and interior edges

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 U_∞.

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

where $\mathbf{A} = \partial(\vec{\mathbf{F}} \cdot \hat{n}_b)/\partial\mathbf{U}$ is the standard Roe matrix [19] 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 have $\mathbf{R}(\mathbf{U}) = 0$ at steady state everywhere in the domain including the nodes on the boundary. This is important for it guarantees that the implicit function theorem holds and that correct sensitivities will be calculated (as discussed in the Sensitivity Analysis section below).

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

$$\mathbf{D}_{ik} = \frac{S}{2}|\mathbf{A}(\mathbf{U}_i, \mathbf{U}_k)|(\mathbf{U}_i - \mathbf{U}_k) = \frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|\mathbf{P}^{-1}(\mathbf{U}_i - \mathbf{U}_k) \tag{10}$$

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, $\Delta\mathbf{W}_{ik} = \mathbf{P}^{-1}\Delta\mathbf{U}_{ik}$. 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. Hence the diffusive flux takes the form:-

$$\mathbf{D}_{ik} = \frac{S}{2}\mathbf{P}|\mathbf{\Lambda}|(\mathbf{P}^{-1}\Delta\mathbf{U}_{ik} - L(\mathbf{P}^{-1}\Delta\mathbf{U}^{-}, \mathbf{P}^{-1}\Delta\mathbf{U}^{+})) \tag{11}$$

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

$$L(u, v) = minmod(u, v) \tag{12}$$

and $\Delta\mathbf{U}^{-}$, $\Delta\mathbf{U}^{+}$ are formed exactly from the gradients in the cells that either end of the edge points into [3]. For example,

$$\Delta\mathbf{U}^{+} = (\nabla\mathbf{U})^{+} \cdot (\vec{x}_i - \vec{x}_k) = \epsilon_{ir}\Delta\mathbf{U}_{ir} + \epsilon_{is}\Delta\mathbf{U}_{is} \tag{13}$$

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. The addition of antidiffusion causes the scheme to now resemble a high order MUSCL scheme. For further details of similar algorithms, the reader is referred to [16, 15, 4].

Timestep calculation

It was found that use of a linearized Fourier stability analysis to determine the allowable timestep [6], allows convergence to machine accuracy only for very low CFL numbers. For practical CFL levels, limit-cycling was observed. It is found that for 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 TVD CFL-like condition. For a scalar equation, TVD conditions for the semidiscrete form of a scheme that can be expressed as

$$\frac{du_o}{dt} = \sum_{i=1}^{K} c_{oi}(u_i - u_o) \tag{14}$$

are

$$c_{oi} > 0 \quad i = 1, \cdots, K \tag{15}$$

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

$$u_o^{n+1} = u_o^n + \Delta t \sum_{i=1}^{K} c_{oi}(u_i - u_o) = \sum_{i=1}^{K} d_{oi} u_i + \mu_o u_o \tag{16}$$

In this case, the TVD conditions become

$$d_{oj} = \Delta t c_{oj} > 0 \quad j = 1, \cdots, K \tag{17}$$

$$\mu_o = 1 - \Delta t \sum_{i=1}^{K} c_{oi} > 0 \tag{18}$$

The latter is a multidimensional TVD CFL-like condition. For our scheme this takes the form

$$\mu_i = 1 - \Delta t_i \left(\sum_{ki}^{edges} (C_{ki}^{int} + C_{ki}^{upw} + C_{ki}^{ad}) + \sum_{kip}^{bou} C_{kip}^{bou} \right) \tag{19}$$

where the four coefficients in the inner parenthesis represent maximum multipliers of characteristic differences, $\Delta \mathbf{W} = \mathbf{P}^{-1} \Delta \mathbf{U}$. They correspond to interior "physical" fluxes, 1st order diffusive fluxes, antidiffusive fluxes and boundary fluxes, respectively, and are given by,

$$C_{ki}^{int} = C_{ki}^{upw} = \frac{\sigma}{A_i} S \lambda_{max} \tag{20}$$

$$C_{ki}^{ad} = \frac{\sigma}{A_i} (\epsilon_{ir} + \epsilon_{is}) S \lambda_{max} \phi(r)|_{max} \tag{21}$$

where $\phi(r)$ is the limiter function and is related to $L(u, v)$ in (12) by $L(u, v) = u\phi(v/u)$. For the present case, with the minmod limiter, $\phi_{max} = 1$.

The supplemental TVD 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. An example presented in [5] shows dramatically improved convergence properties with $\mu_t = -1$ for an analysis case. Furthermore, the sensitivity calculation, with an identical Δt, was found to be stable for the CFL number the analysis scheme had been run at.

Sensitivity Analysis

Gradient-based optimization methods require the gradient, or sensitivity, of a cost function, $I(\mathbf{U},\boldsymbol{\beta})$, with respect to the vector of design variables, $\boldsymbol{\beta}$. Sensitivities can be calculated using either the direct or the adjoint method. In either case, the implicit function theorem is used which results from direct differentiation of the steady state discrete residual equation,

$$\mathbf{R}(\mathbf{U},\boldsymbol{\beta}) = \mathbf{0} \tag{22}$$

with respect to the design variables, producing,

$$\frac{d\mathbf{R}}{d\boldsymbol{\beta}} = \mathbf{0} \Rightarrow \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \boldsymbol{\beta}} = -\frac{\partial \mathbf{R}}{\partial \boldsymbol{\beta}} \tag{23}$$

We can calculate the cost function sensitivity as follows.

$$\frac{dI}{d\boldsymbol{\beta}} = \frac{\partial I}{\partial \boldsymbol{\beta}} + \frac{\partial I}{\partial \mathbf{U}}\frac{\partial \mathbf{U}}{\partial \boldsymbol{\beta}} \tag{24}$$

With the direct method one solves (23) directly for $\mathbf{U}_\beta = \partial\mathbf{U}/\partial\boldsymbol{\beta}$ whereas with the adjoint method one solves for the adjoint variables $\boldsymbol{\psi}$ which satisfy

$$(\frac{\partial \mathbf{R}}{\partial \mathbf{U}})^T\boldsymbol{\psi} = (\frac{\partial I}{\partial \mathbf{U}})^T \tag{25}$$

The cost function sensitivities are then calculated according to

$$\frac{dI}{d\boldsymbol{\beta}} = \frac{\partial I}{\partial \boldsymbol{\beta}} - \frac{\partial I}{\partial \mathbf{U}}(\frac{\partial \mathbf{R}}{\partial \mathbf{U}})^{-1}(\frac{d\mathbf{R}}{d\boldsymbol{\beta}}) = \frac{\partial I}{\partial \boldsymbol{\beta}} - \boldsymbol{\psi}^T(\frac{d\mathbf{R}}{d\boldsymbol{\beta}}) \tag{26}$$

It is possible to adopt exactly the same relaxation algorithm to solve equations (22), (23) and (25). Hence to find $\mathbf{U}_\beta$ or $\boldsymbol{\psi}$, we can drive the following linear equations to steady state.

$$\mathbf{U}_{\beta_i}^{n+1} = \mathbf{U}_{\beta_i}^{n} + \sigma(\frac{\Delta t}{A})_i(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\boldsymbol{U}_\beta + (\frac{d\mathbf{R}}{d\boldsymbol{\beta}})_U) \tag{27}$$

$$\boldsymbol{\psi}_i^{n+1} = \boldsymbol{\psi}_i^{n} + \sigma(\frac{\Delta t}{A})_i(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}^T\boldsymbol{\psi} - \frac{\partial I}{\partial \mathbf{U}}^T) \tag{28}$$

For the direct method, equation (23) is solved for each design variable while for the adjoint method equation (25), requiring a similar computational effort, needs to be solved only once, assuming that only one objective function is considered. This represents a large cost savings when the number of design variables is large.

These schemes are stable because the error modes are subject to the same amplification matrix as those for the linearized flow analysis scheme (3):

$$\Delta\mathbf{U}_i^{n+1} \approx \Delta\mathbf{U}_i^{n} + \sigma(\frac{\Delta t}{A})_i\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Delta\mathbf{U}^n \tag{29}$$

The stability characteristics of equations (27) and (28) are the same since transposing a matrix does not change its eigenvalues. Indeed for all calculations performed, the asymptotic convergence rate of analysis, direct and adjoint schemes was found to be very close.

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 (4) for sensitivities is

$$\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}\mathbf{U}_{\beta_j} = \sum_{ik=1}^{K}[\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}]_{ik}\mathbf{U}_{\beta_j} \tag{30}$$

where, for example, for an interior edge,

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

For the adjoint problem the transposed Jacobian is required.

$$(\frac{\partial\mathbf{R}_j}{\partial\mathbf{U}_i})^T\boldsymbol{\psi}_i = \sum_{ik=1}^{K}[\frac{\partial\mathbf{R}_i}{\partial\mathbf{U}_j}]_{ik}\boldsymbol{\psi}_i \tag{31}$$

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

$$\frac{\partial\mathbf{R}}{\partial\boldsymbol{\beta}} = \frac{\partial\mathbf{R}}{\partial\mathbf{X}}\frac{d\mathbf{X}}{d\boldsymbol{\beta}} \tag{32}$$

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.

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

$$\vec{\boldsymbol{\delta}}_i^{\,n+1} = \frac{\sum k_{ij}\vec{\boldsymbol{\delta}}_j^{\,n}}{\sum k_{ij}} \tag{33}$$

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

$$\mathbf{X}^{n+1} = \mathbf{X}^n + \boldsymbol{\delta}^{n+1} \tag{34}$$

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

$$\frac{d\mathbf{X}}{d\boldsymbol{\beta}} = \frac{\partial \boldsymbol{\delta}^N}{\partial \boldsymbol{\beta}} = \frac{\partial \boldsymbol{\delta}^N}{\partial \boldsymbol{\delta}^{N-1}} \frac{\partial \boldsymbol{\delta}^{N-1}}{\partial \boldsymbol{\delta}^{N-2}} \cdots \frac{\partial \boldsymbol{\delta}^1}{\partial \boldsymbol{\delta}^0} \frac{\partial \boldsymbol{\delta}^0}{\partial \boldsymbol{\beta}} \tag{35}$$

where we have performed N Jacobi iterations of the grid relaxation scheme. The factors of the form $\partial \boldsymbol{\delta}^i / \partial \boldsymbol{\delta}^{i-1}$ in the above product are found from equation (33) while $\partial \boldsymbol{\delta}^0 / \partial \boldsymbol{\beta}$ is determined from the expression for the wing surface design variable perturbation. It is simple and inexpensive to check the correctness of these sensitivity calculations versus finite difference and this has become our standard practice.

Much discussion of boundary conditions for the adjoint equation has taken place in the literature [12, 10, 9, 8]. We take the approach that since we have implicitly included the boundary conditions in the expression for the flow equation residual, that the grid and flow derivatives of these residuals will automatically include the boundary conditions for both adjoint and flow sensitivity variables.

Implementation issues 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 partioning 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 that 95% for the simulation of practical problems. With this implementation, the flow analysis code requires 0.32 microseconds per time step per mesh point while the adjoint requires 2.42 microseconds.

In [5], it was found that the edge contributions to $\partial \mathbf{R} / \partial \mathbf{U}$ matrix could be stored easily for the 2D case. Eliminating the cost of forming this matrix every iteration lowers the per-iteration cost significantly. It was found that in 2D the sensitivity calculation took approximately half that of the flow analysis while it was a factor of eight larger (than flow analysis) in 3D where saving these terms on an edge basis was not done originally due to the overwhelmingly large memory requirements for a serial implementation. However for small 3D cases it was found that saving the $\partial\mathbf{R}/\partial\mathbf{U}$ matrix reduced the per-iteration cost by a factor of about four. Furthermore, the parallel implementation allows storing of $\partial\mathbf{R}/\partial\mathbf{U}$ for much larger systems since only those values corresponding to the local domain and interface regions need to be stored.

3 Results

Wing/body optimization 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 2 and 3. The mesh contains 65,000 nodes and 350,000 elements. This optimization case was performed using the serial algorithm. Four design variables were used. The associated mode shapes were chosen to be products of linear functions, $g_j(y)$, in the spanwise direction, and NACA 4-series meanline ($f_1(x/c)$ with maximum at 40% chord) and thickness ($f_2(x/c)$) functions in the chordwise direction. Specifically, the wing upper and lower surface parameterizations are

$$\begin{aligned} z_{upper}(x,y) &= z^0_{upper}(x,y) + \sum_{i=1}^{N_{des}} \beta_i h_i(x,y) &&= z^0_{upper}(x,y) + \sum_{i=1}^{N_i}\sum_{j=1}^{N_j} \beta_{k_{ij}} f_i(x/c) g_j(y) \\ z_{lower}(x,y) &= z^0_{lower}(x,y) + \sum_{i=1}^{N_{des}} \varphi_i \beta_i h_i(x,y) &&= z^0_{lower}(x,y) + \sum_{i=1}^{N_i}\sum_{j=1}^{N_j} \varphi_{k_{ij}} \beta_{k_{ij}} f_i(x/c) g_j(y) \end{aligned} \tag{36}$$

where $k_{ij} = i + 2(j-1)$, φ_k is -1 and 1 for thickness and camber design variables, respectively and z^0_{upper}, for example, represents the baseline upper surface. The functions f_i and g_j for this case are summarized in Table 1.

function	definition	range
f_1	$(0.8(x/c) - (x/c)^2)/0.16$	$0 < x/c < 0.4$
	$(0.2 + 0.8(x/c) - (x/c)^2)/0.7056$	$0.4 < x/c < 1$
f_2	$0.2969\sqrt{x/c} - .12810(x/c) - .35160(x/c)^2 + .28430(x/c)^3 - .1015(x/c)^4$	$0 < x/c < 1$
g_1	$(y_{midspan} - y)/(y_{midspan} - y_{root})$	$y_{root} < y < y_{midspan}$
	$(y - y_{midspan})/(y_{tip} - y_{midspan})$	$y > y_{midspan}$
g_2	0	$y_{root} < y < y_{midspan}$
	$(y - y_{midspan})/(y_{tip} - y_{midspan})$	$y > y_{midspan}$

Table 1: Surface Parameterization for wing-body problem

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_{targ}, 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. The BFGS method is a quasi-Newton method in which an increasingly accurate estimate of the Hessian matrix, H, is made based on changes in the gradient vector along the optimization path [7]. Resulting geometry and surface pressure evolutions are shown in Figures 4 and 5 while the evolution of design variables and cost function is shown in Figure 6. 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 performed in [5]) provides confidence that the sensitivities are being accurately calculated by the current scheme.

Business jet optimization The second optimization example presented herein is that 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 12 and 7, respectively. The grid contains 160,000 nodes and 860,000 elements. The target pressure distribution 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 8. 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. Six design variables were used whose associated mode shapes were combinations of 3 chordwise functions and 2 spanwise functions. The selected design variables result in a wing parameterization given by equation (36) and the functions f_i and g_j for this case are listed in Table 2. 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. The spanwise functions have maxima at the wing-body intersection (y_{root}) and at the nacelle centerline (y_{engine}). They 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.

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

function	definition	range
f_1	x/c	$0 < x/c < 1$
f_2	$\sin(\pi(x/c)^{2.40942})$	$0 < x/c < 1$
f_3	$\sin^2(\pi(1-x/c))$	$0 < x/c < 1$
g_1	$\cos^2(\frac{\pi}{2}(y-y_{root})/(y_{engine}-y_{root}))$ 0	$0 < y < y_{engine}$ $y > y_{engine}$
g_2	$\sin^2(\frac{\pi}{2}(y-y_{root})/(y_{engine}-y_{root}))$ $\cos^2(\frac{\pi}{2}(y-y_{engine})/(y_2-y_{engine}))$ 0	$0 < y < y_{engine}$ $y_{engine} < y < y_2$ $y > y_2$

Table 2: Surface Parameterization for business jet problem

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, 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.

$$\Delta\theta(x,\theta(z)) = \Delta\theta_{w/b}(x)g(\theta(z)) \tag{37}$$

The linear function in θ 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 tactic 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 9 and 10. The final pressure distribution is shown in Figure 11. Cost function and design variable evolutions are plotted in Figure 13. 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%.

Various partitioning strategies of the domain were used in this exercise, with the number of domains used varying from 6 to 20.

4 Conclusion

In [5], the authors demonstrated for the discrete 2D and 3D Euler equations, the capability to accurately calculate "exact" sensitivities. Per-iteration times for the sensitivity analysis were found to be a factor of 2 lower than those for the analysis in 2D. In 3D, the corresponding per-iteration times were found to be a factor of slightly less than 8 *higher*. However, when the $\partial\mathbf{R}/\partial\mathbf{U}$ matrix is stored, the *increase* in cost is reduced to a factor of about 2. The extension of the 3D capability to practical optimization of complete aircraft configurations by parallelization of the scheme has been presented and found to be accurate and efficient. Furthermore, use of the adjoint method practically *eliminates* the dependence of the cost function gradient calculation on the number of design variables. However, the number of iterations required to reach a minimum for optimization exercises based on the BFGS strategy will tend to depend on N_{des} since a good estimate of the Hessian, H, is not typically found until after N_{des} iterations.

Acknowledgements

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] AGARD. Computational Methods for Aerodynamic Design (Inverse) and Optimization. Report No. 463, 1989.

[2] AGARD. Special Course on Inverse Methods for Airfoil Design for Aeronautical and Turbomachinery Applications. Report No. 780, 1990.

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

[4] J. Elliott. *Aerodynamic Optimization based on the Euler and Navier-Stokes equations using Unstructured Grids.* PhD thesis, Dept. of Aeronautics and Astronautics, MIT, 1997. To be published.

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

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

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

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

[9] 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.

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

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

[12] A. Jameson. Optimum Aerodynamic Design via Boundary Control. Princeton University, MAE Report 1996, 1994.

[13] 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.

[14] 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.

[15] 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*, 1992.

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

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

[18] J. Reuther and A. Jameson. Supersonic Wing and Wing-Body Shape Optimization using an Adjoint Formulation. In *CFD for Design and Optimization: ASME International Mech. Eng. Congress and Exposition*, November 1995.

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

[20] 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.

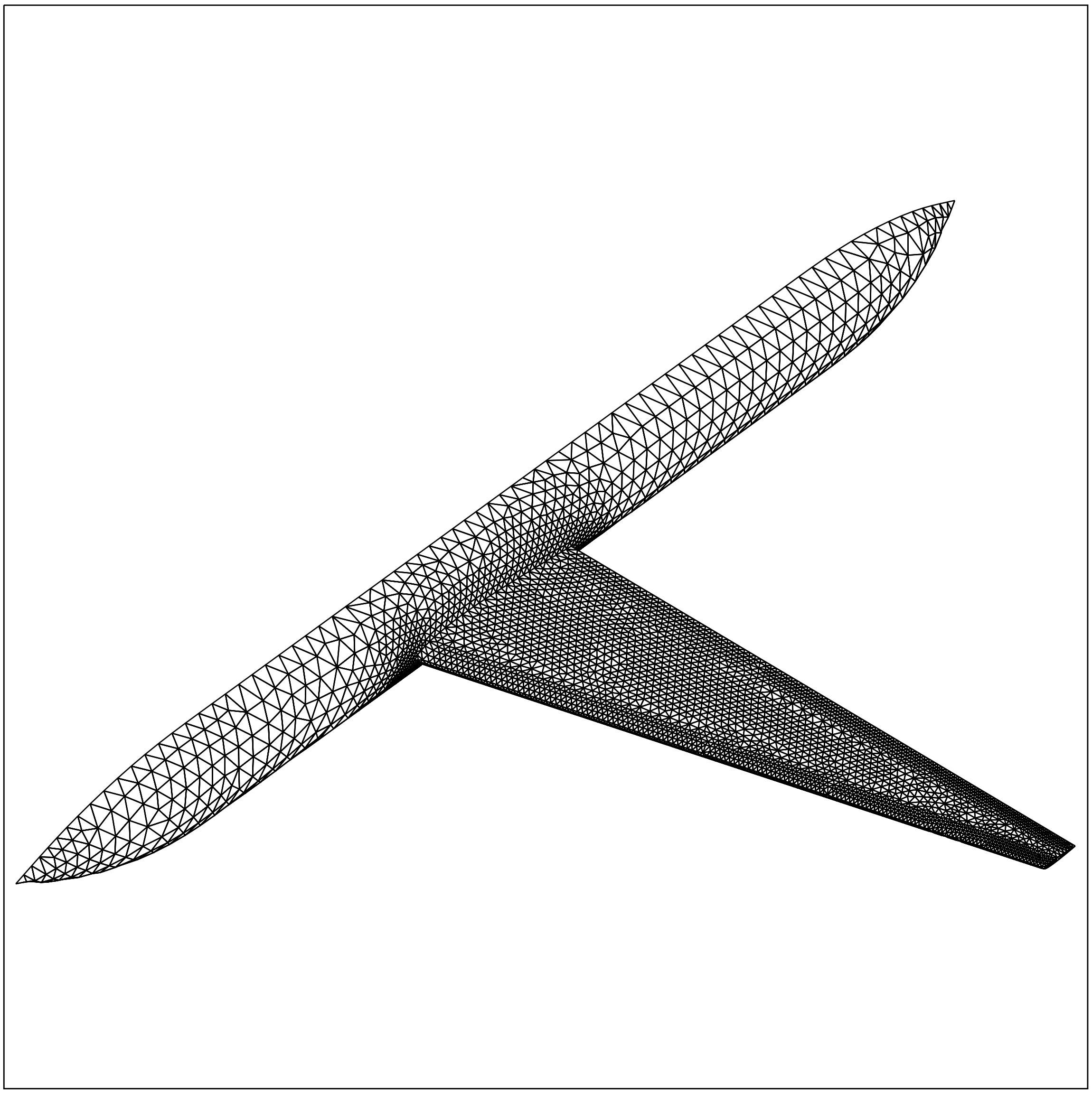

Figure 2: Surface grid for baseline 3D wing/body calculation

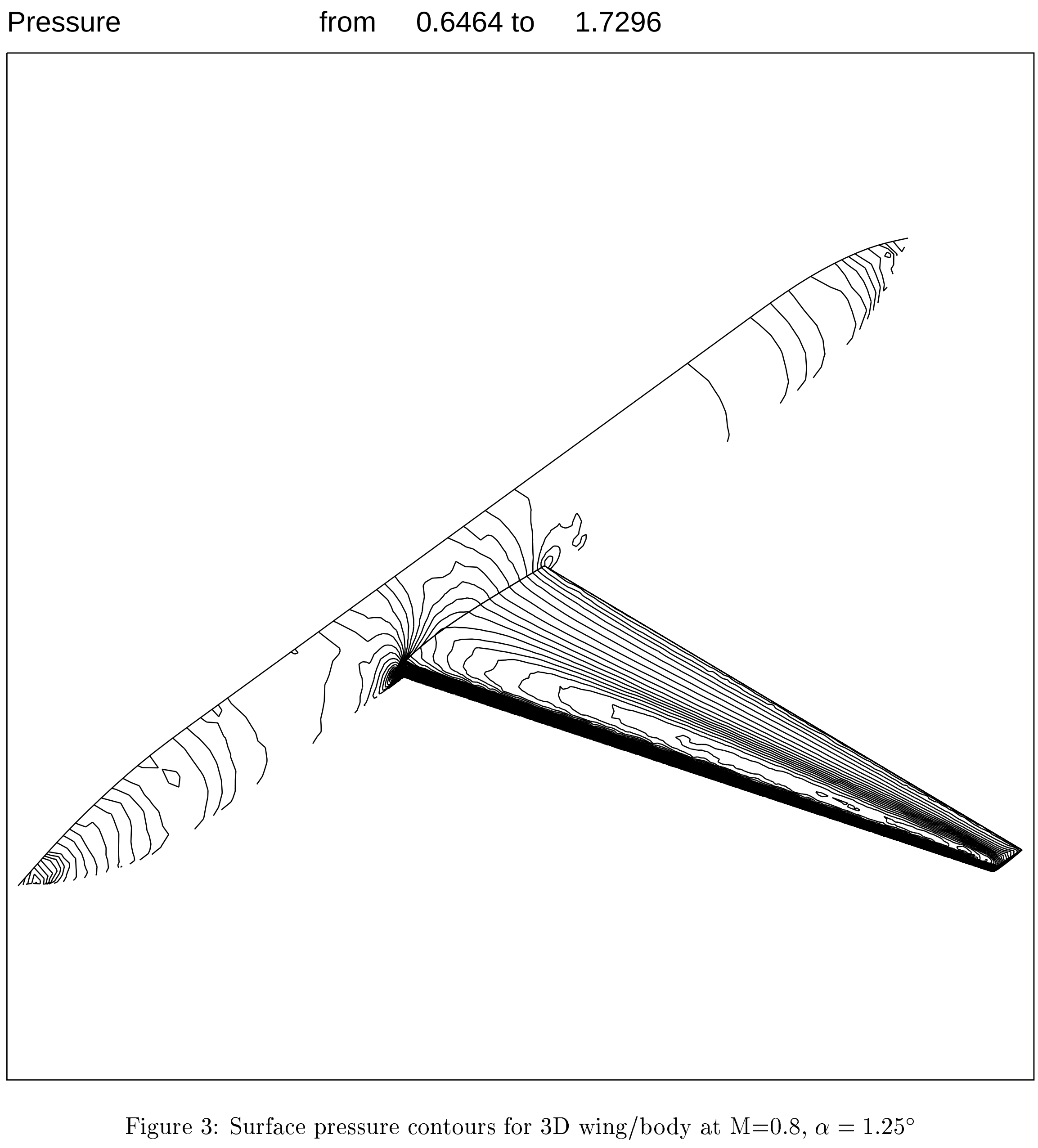

Figure 3: Surface pressure contours for 3D wing/body at M=0.8, $\alpha = 1.25°$

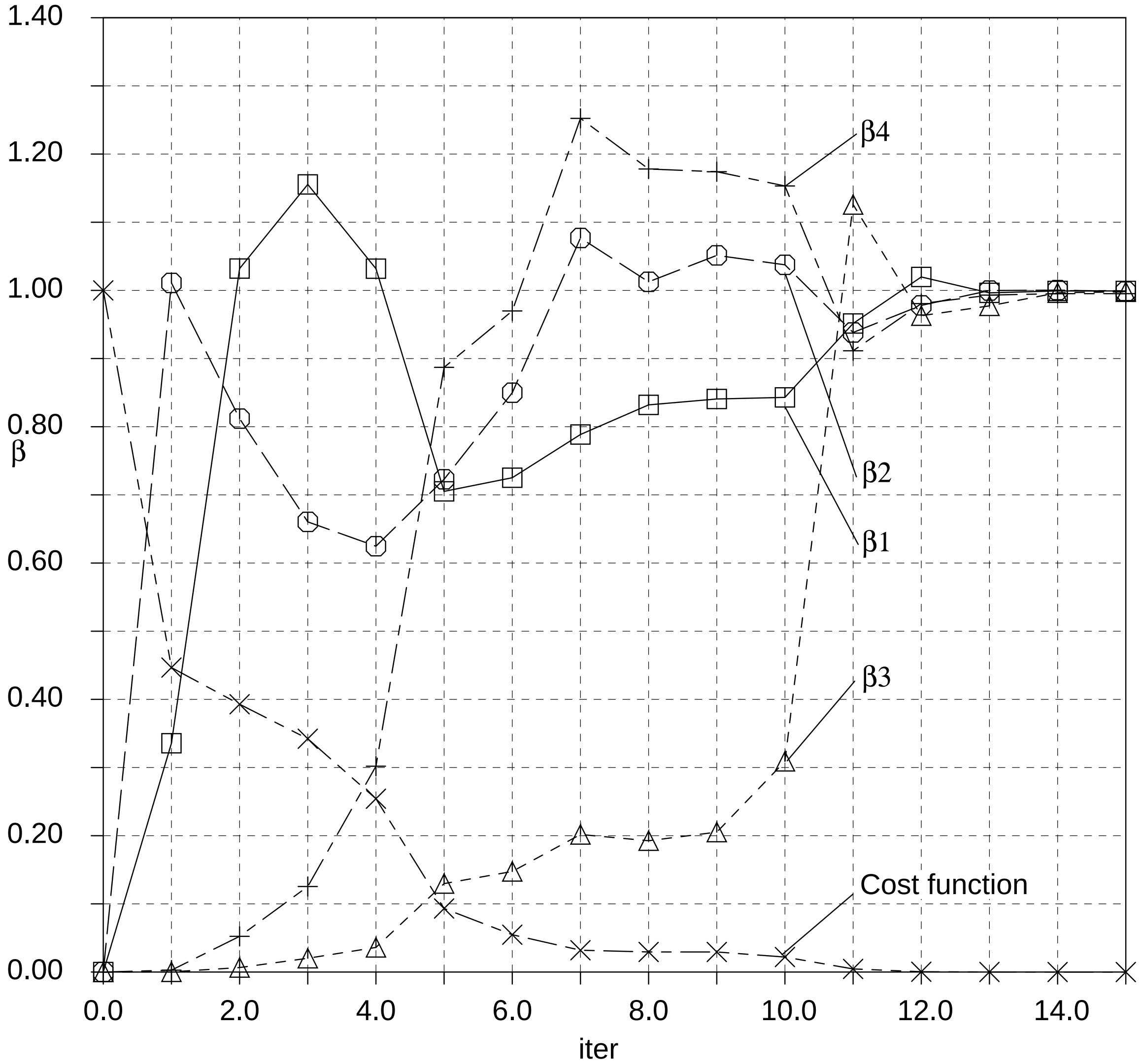

Figure 4: Cost function and design variable evolution for wing/body optimization exercise

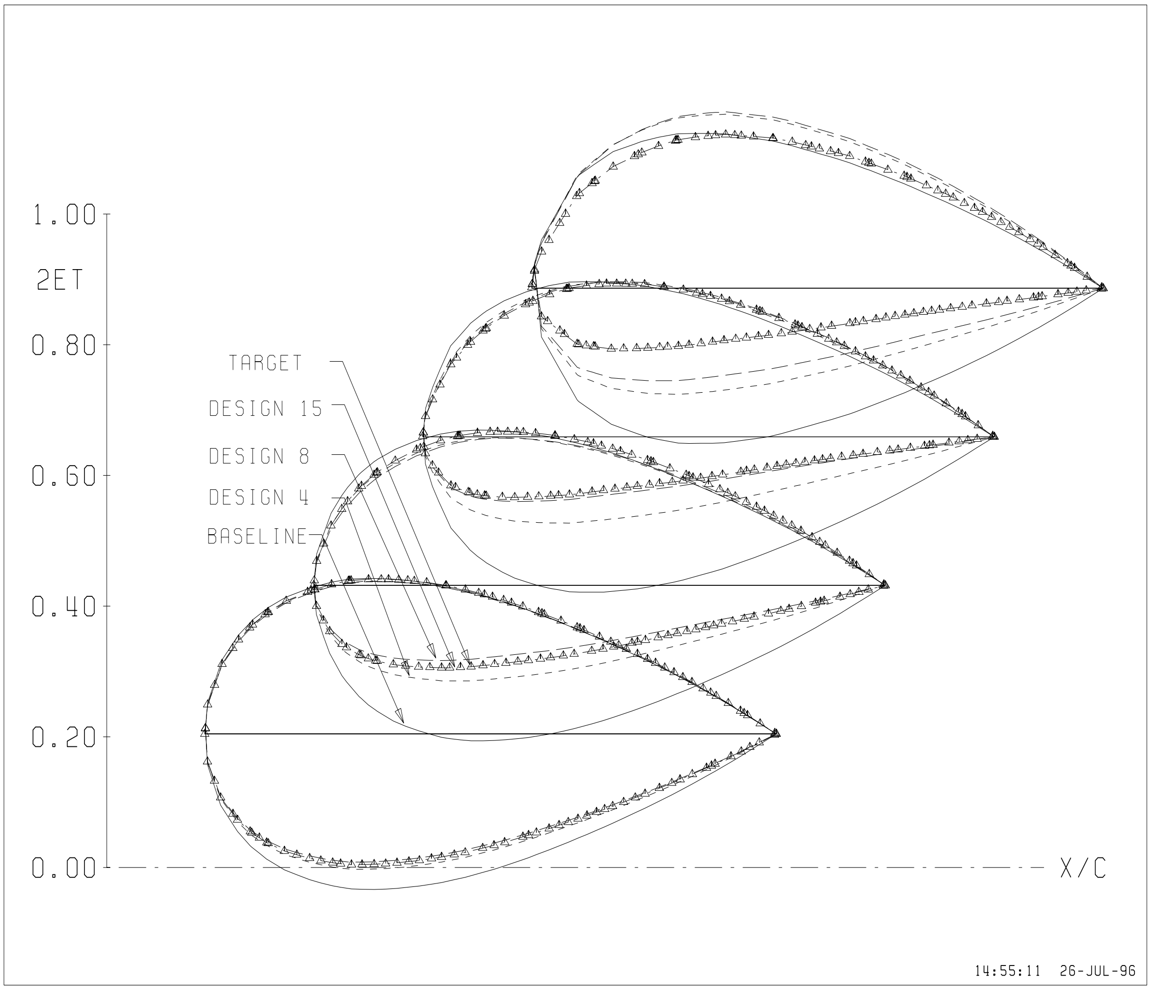

Figure 5: Geometry evolution for wing/body optimization exercise

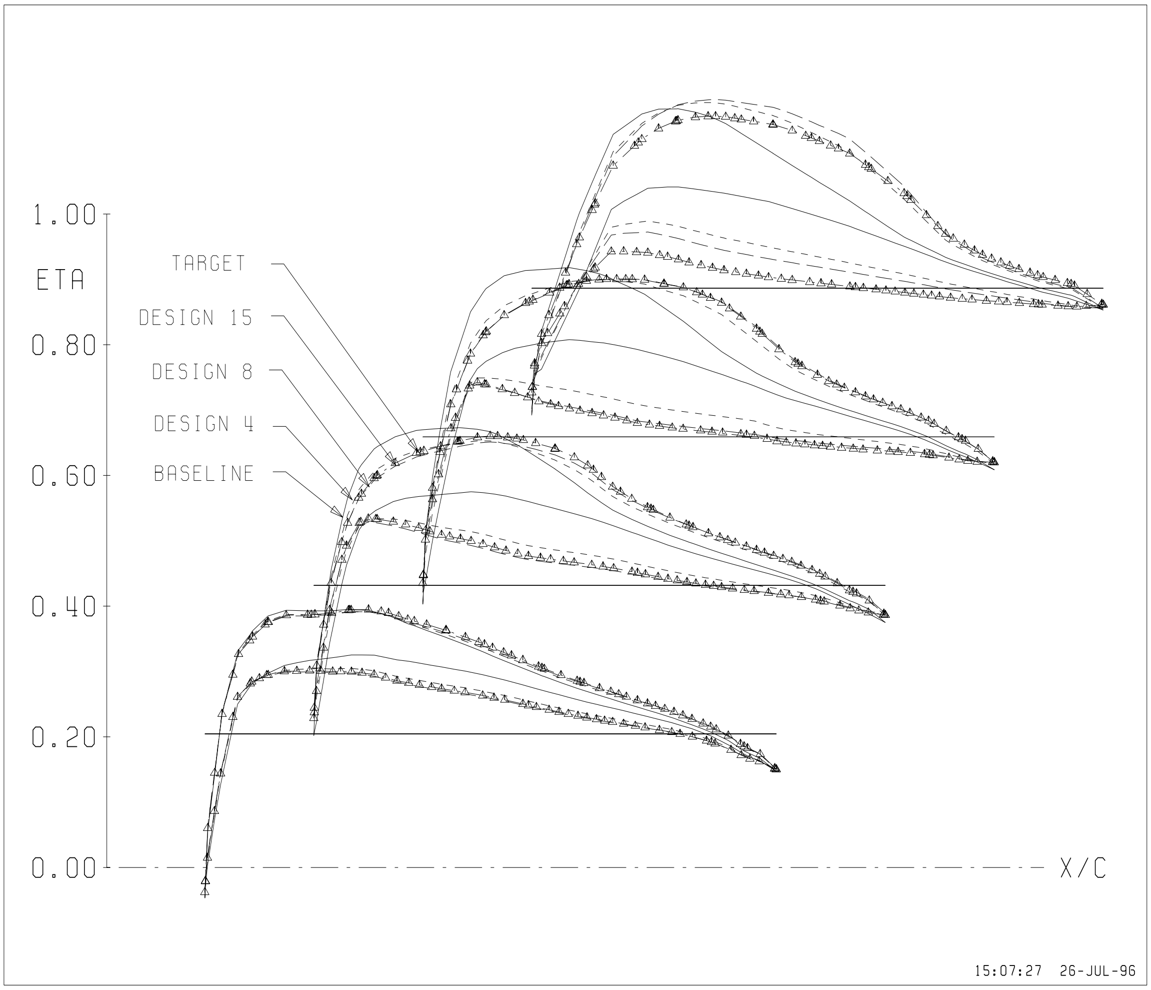

Figure 6: Pressure evolution for wing/body optimization exercise

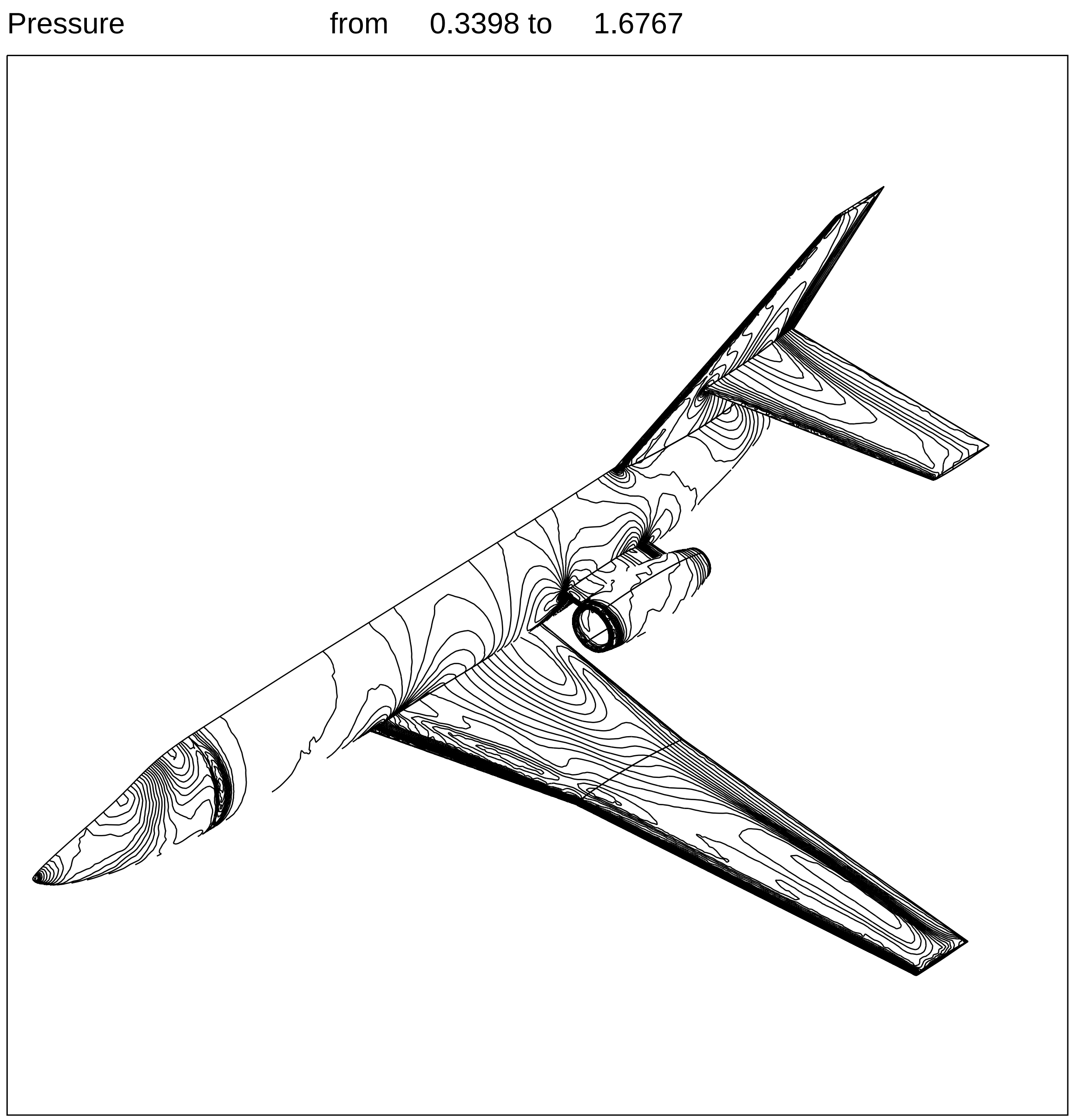

Figure 7: Baseline surface pressure contours for business jet at M=0.85, $\alpha = 2°$

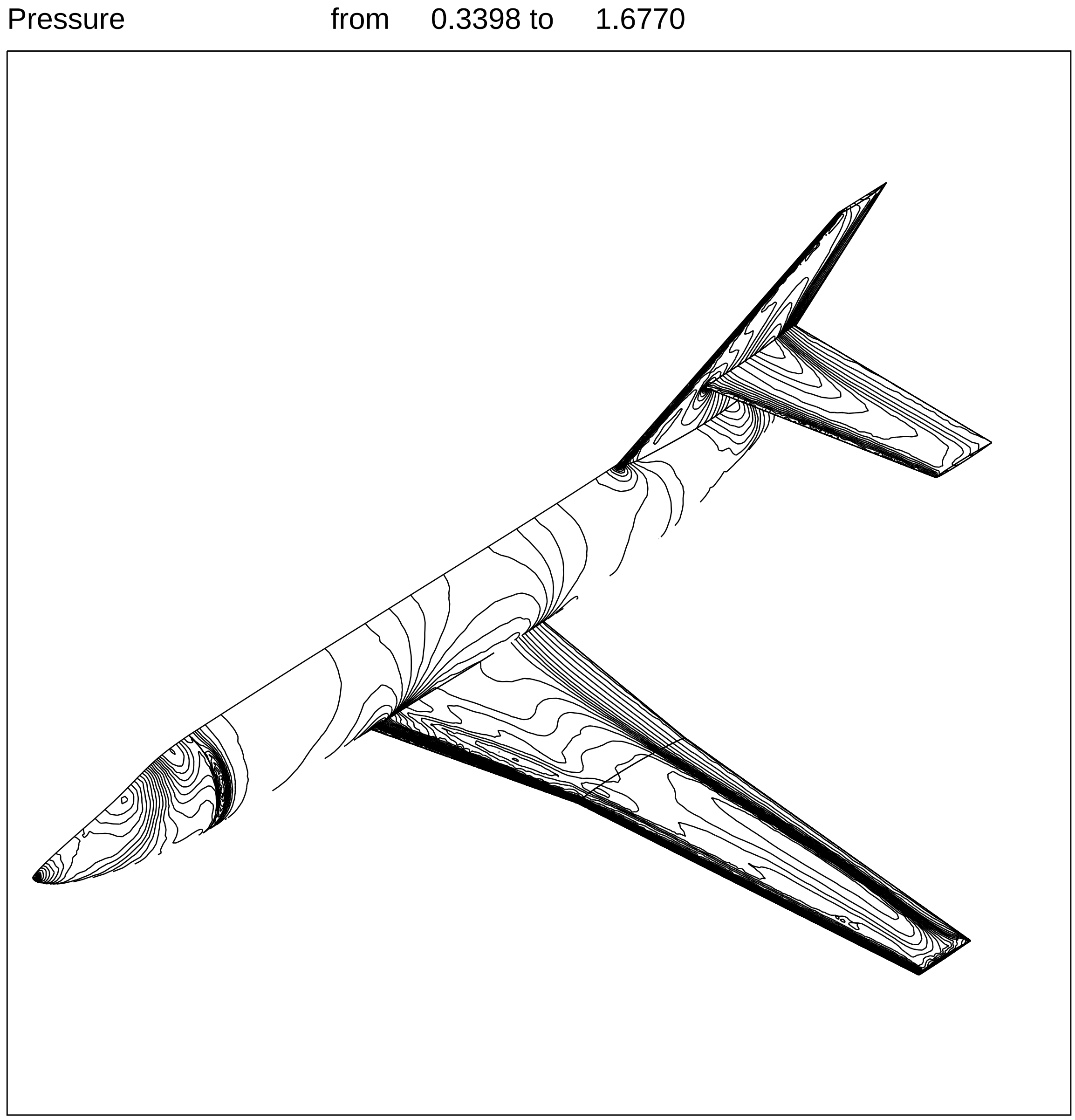

Figure 8: Target surface pressure contours for business jet at M=0.85, $\alpha = 2°$

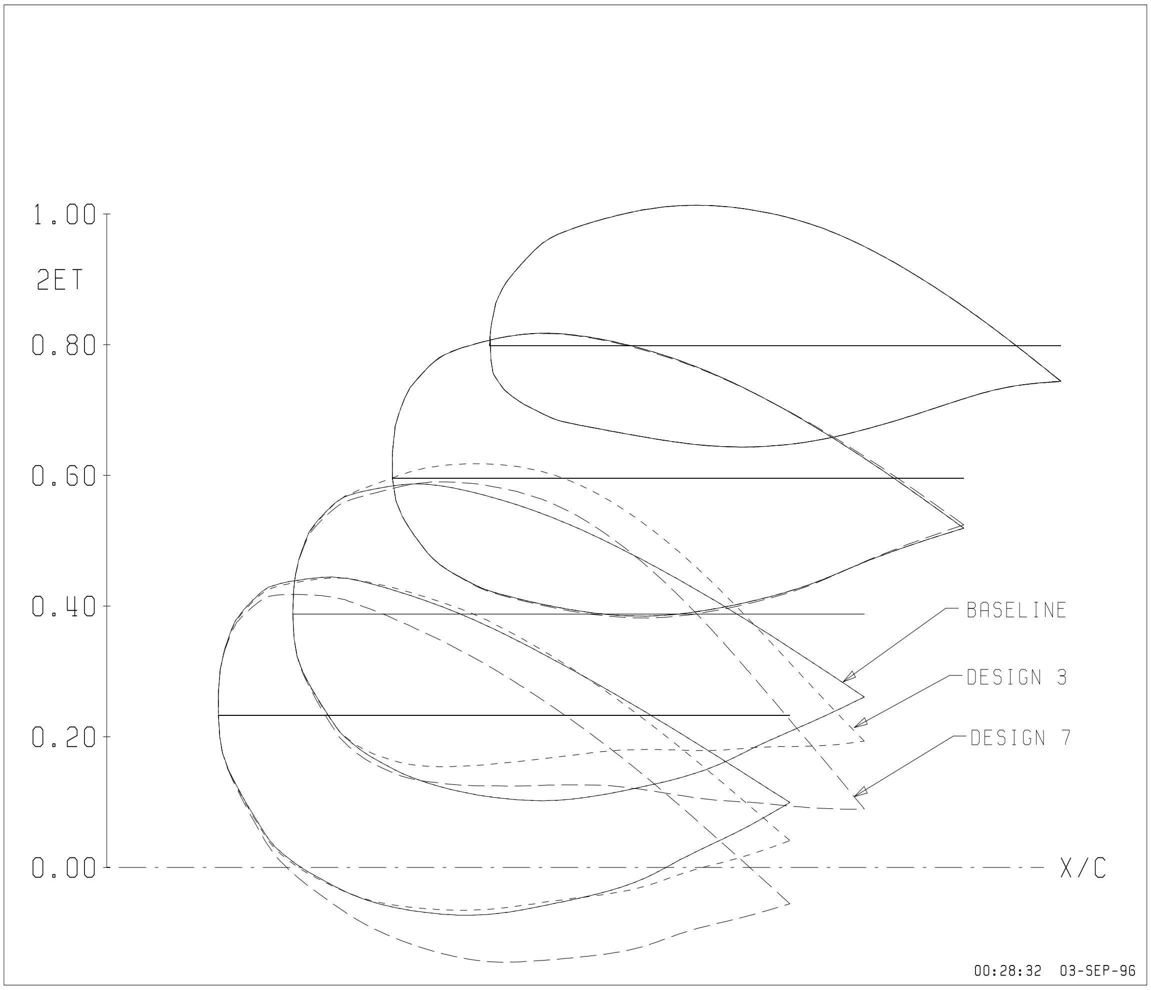

Figure 9: Geometry evolution for business jet optimization exercise

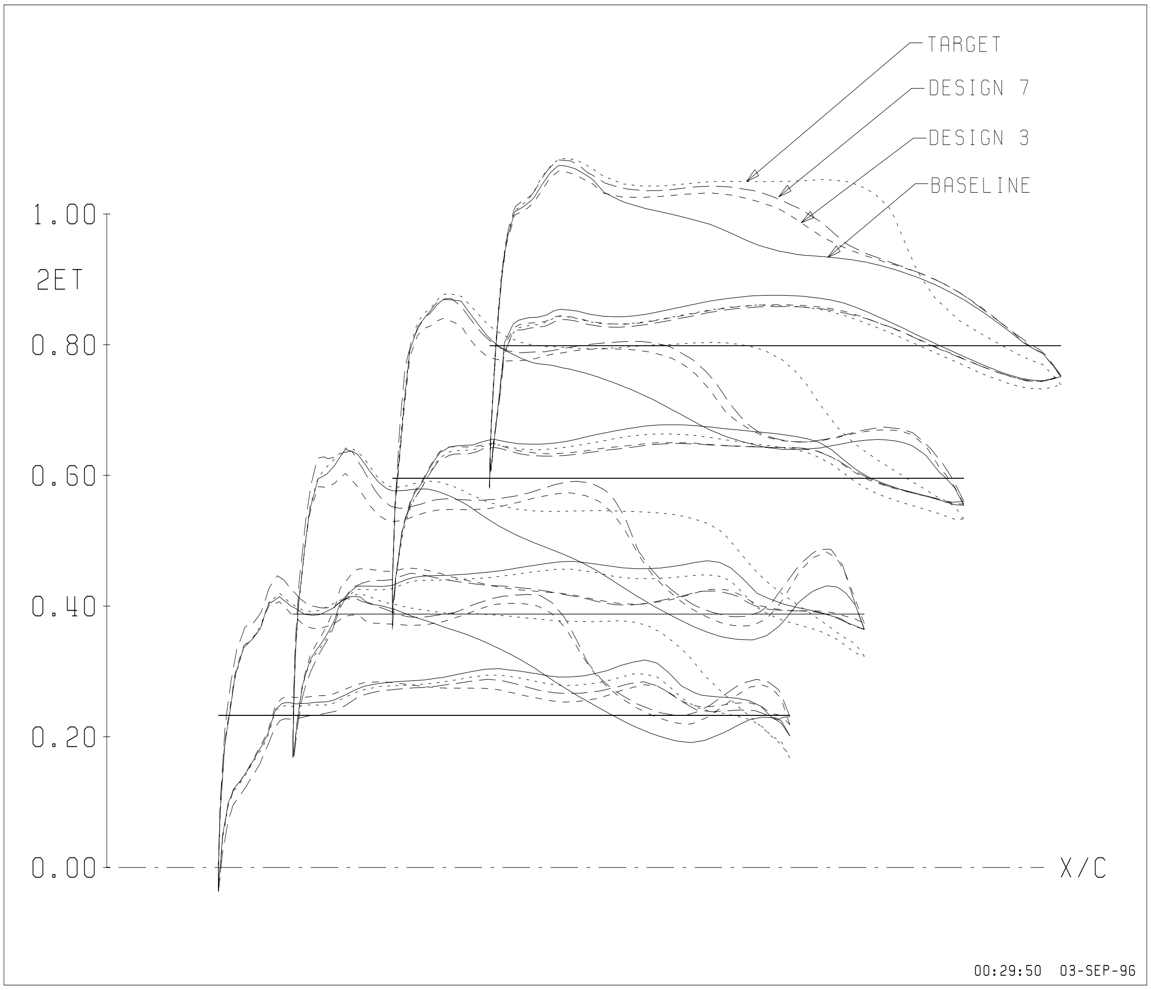

Figure 10: Pressure evolution for business jet optimization exercise

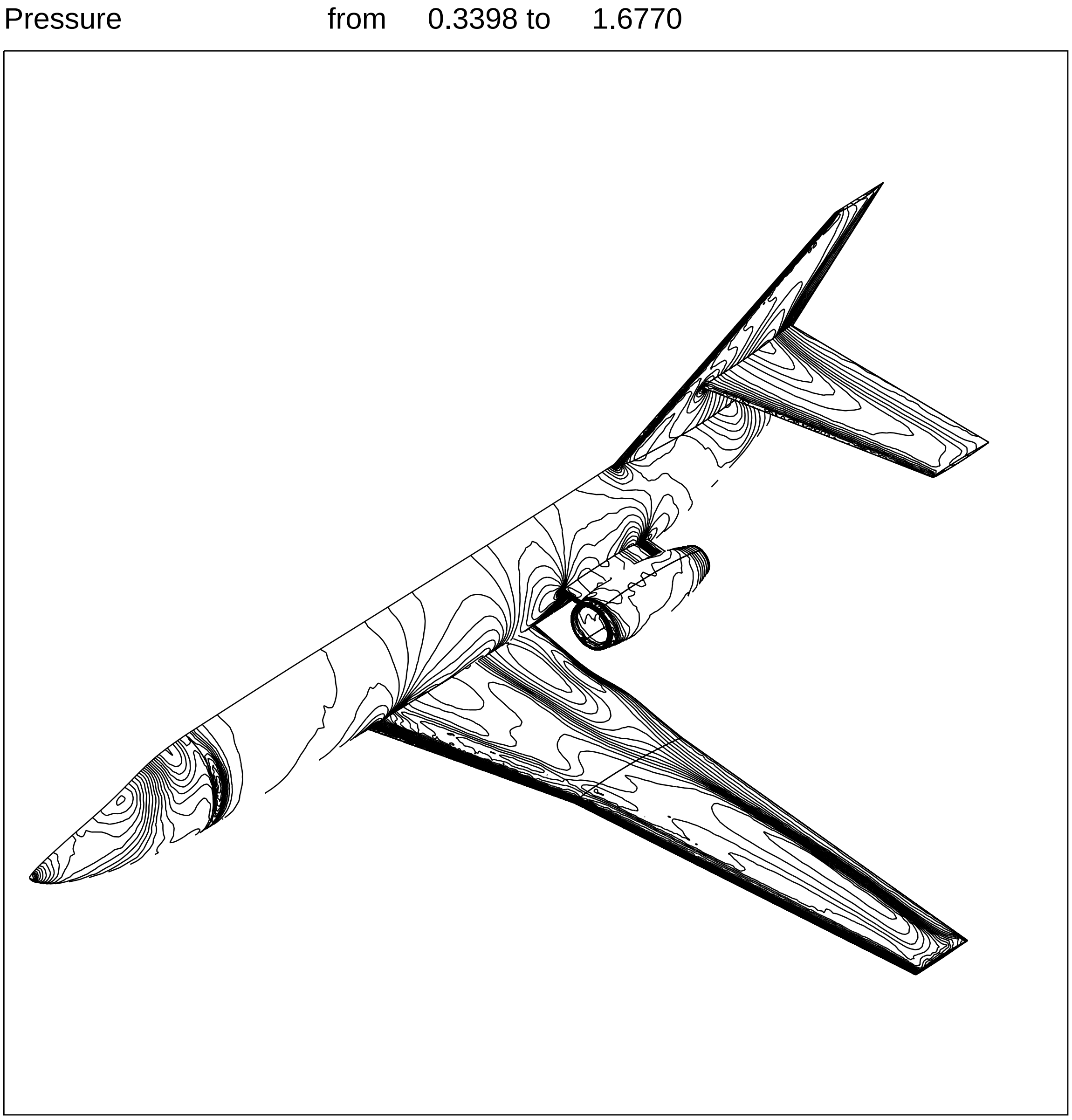

Figure 11: Surface pressure contours for business jet at M=0.85, $\alpha = 2°$ after 7 design iterations

Figure 12: Surface grid for baseline business jet calculation

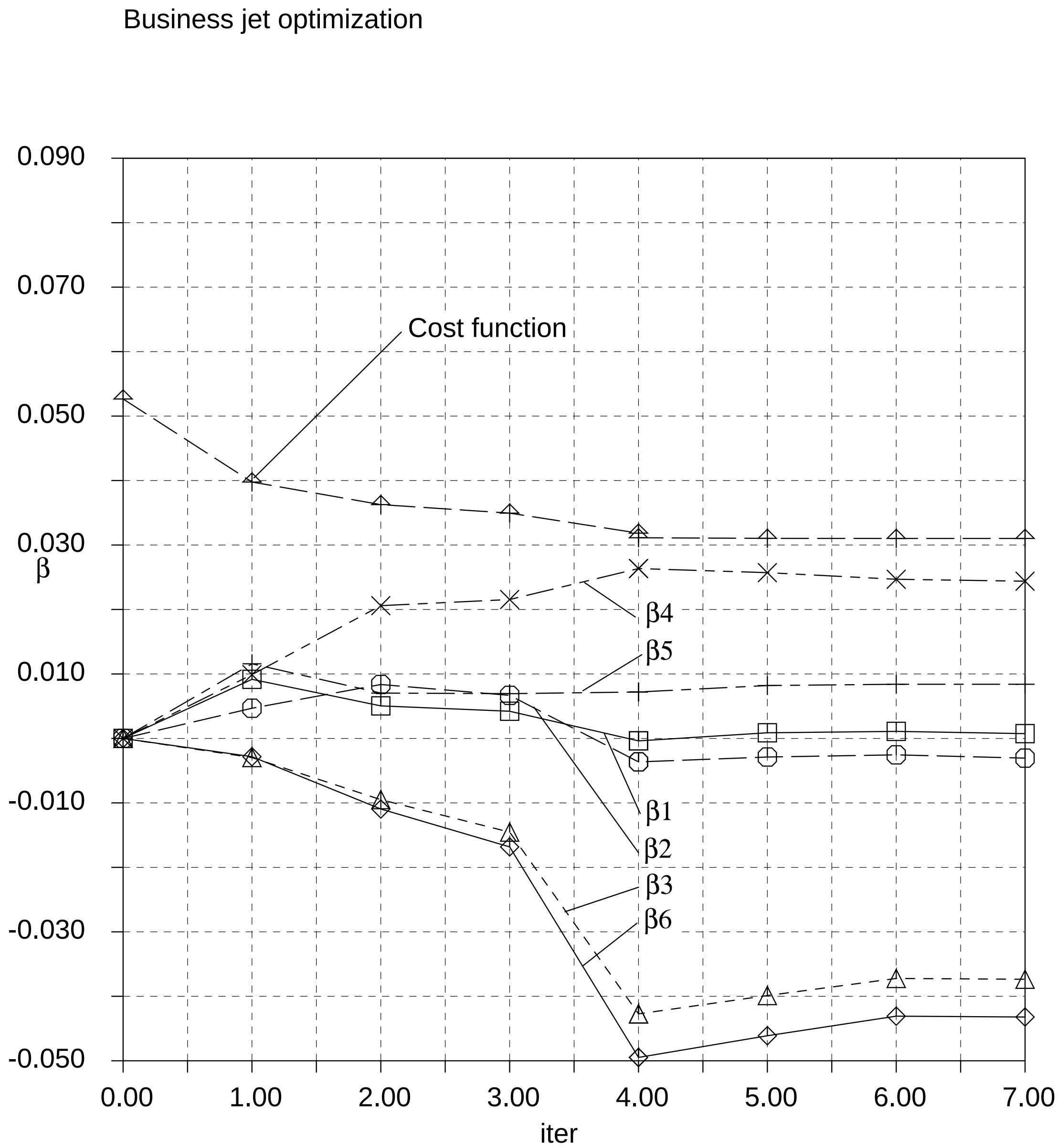

Figure 13: Cost function and design variable evolution for business jet optimization exercise