October 7, 1997

REVIEW ARTICLE

Unstructured Grid Finite Element Methods for Fluid Mechanics

K Morgan† and J Peraire‡

†Department of Civil Engineering, University of Wales, Swansea SA2 8PP, UK

‡Department of Aeronautics and Astronautics, MIT, Cambridge MA 02139, USA

Abstract. The development of unstructured grid based finite element methods for the simulation of fluid flows is reviewed. The review concentrates on solution techniques for the compressible Euler and Navier Stokes equations, employing methods which are based upon a Galerkin discretisation in space together with an appropriate finite difference representation in time. It is assumed that unstructured assemblies of triangles are used to achieve the spatial discretisation in two dimensions, with unstructured assemblies of tetrahedra employed in the three dimensional case. Adaptive grid procedures are discussed and methods for accelerating the iterative solution convergence are considered. The areas of incompressible flow modelling and optimisation are also included.

1. Introduction

Over the past thirty years, there has been an intense research activity in the area of computational fluid dynamics. A large proportion of this activity has been driven by the aerospace industry, with its requirements for highly accurate solutions at minimum computational cost. Initially, it was potential flow problems which formed the main topic of interest but, as solution techniques have matured and computational resources have improved, attention has now moved to problems governed by the compressible Euler and Navier Stokes equations.

The numerical methods which have been employed in practice have generally been based upon finite difference or finite volume techniques [1]. The finite element method is a relative newcomer to this area and, indeed, by the early 1980s, its penetration was fairly small [2]. Methods based upon the finite element approach are generally attractive because they lead to general purpose computer codes. In addition, finite element methods may be rigorously formulated for elliptic and parabolic problems [3], while much is now also understood about the application to hyperbolic equations [4]. However, it is not these features which have been largely responsible for the recent interest shown in the finite element method but the fact that it is based upon an integral formulation and, hence, is readily implemented on arbitrary discretisations.

The significance of this property becomes apparent when industrial applications are considered, where the computational simulations frequently involve domains which are of complex geometrical shape. For such domains, the grid generation process is difficult and can prove to be time–consuming, so that cartesian based methods have attracted considerable recent attention [5, 6]. By relaxing the requirement for the use of a totally structured grid, methods have been developed which attempt to overcome this difficulty, while still retaining a great deal of the grid structure necessary for many solution algorithms. Typical examples would be the use of an unstructured assembly of structured grids, as in the multiblock method [7], or the use of overlapped structured grids, as in the chimera [8] or the feature adapted mesh embedding [9] approaches. Another alternative, is to allow the grid to be totally unstructured. This is attractive as fully automatic unstructured tetrahedral grid generators have now been developed and these can rapidly and efficiently discretise domains of arbitrary geometrical shape [10, 11]. These unstructured grid generators can be directly coupled to CAD systems [12] and incorporated within graphical user interfaces [13]. This approach is attractive to industry as it appears to offer the possibility of significantly reducing the number of man hours spent on the task of grid generation. A further alternative is to attempt to solve the complex geometry issue by the generation of hybrid grids, in which the domain of interest is discretised by using an assembly of both structured and unstructured grid regions [14]. The objective is to attempt to gain the best features of both the structured and the unstructured grid methods, but it is not yet clear that this can be generally achieved.

With this background, this review will concentrate upon the development of finite element methods for the solution of the compressible Euler and Navier Stokes equations on unstructured grids. Although the use of higher order elements offers many potential advantages for future developments, it is only the use of simple elements, employing linear interpolation, which is considered here. The initial discussion is directed towards the solution of convection problems in one space dimension and the procedures which are introduced are then employed as the basis for solution methods in higher dimensions. For both steady and unsteady problems, the use of explicit timestepping provides a convenient implementation strategy. However, explicit methods require the use of small timesteps, which can result in excessively long computation times and in convergence difficulties. To illustrate how these drawbacks may be overcome, techniques such as multigrid and implicit timestepping are also included. The unstructured grid approach provides a natural environment for the incorporation of an adaptive grid strategy, designed to improve the quality of a computed solution in an optimum manner, and a brief review is made of the current status of adaptive grid methods. In conclusion, consideration is given to the development of algorithms designed for the modelling of incompressible flow and an illustration is included of recent progress in the field of formal shape optimisation methods.

2. Conservation Equations in One Space Dimension

A study of the scalar conservation equation, in the absence of diffusion, can be adopted as the starting point for the development of solution algorithms for general problems in fluid mechanics. In this section, things are simplified further in that consideration is restricted to examples involving only one spatial dimension, x_1. The problem requires the determination of the distribution of a conserved quantity $U(x_1, t)$ at all points in a finite spatial domain Ω, for all values of the time $t > 0$. This distribution is assumed to be known at the initial time, $t = 0$, for all values of x_1 in Ω. If $U(x_1, t)$, is transported, without diffusion, with a flux $F^1 = F^1(U)$, this leads to the following classical formulation of the problem: find $U(x_1, t)$ such that

$$\frac{\partial U}{\partial t} + \frac{\partial F^1}{\partial x_1} = 0 \qquad x_1 \in \Omega;\ t > 0 \tag{1}$$

subject to the initial condition

$$U(x_1, 0) = U^0(x_1) \qquad x_1 \in \Omega \tag{2}$$

and to appropriate conditions on the boundary Γ of Ω. Here U^0 is a prescribed function. For this one dimensional hyperbolic problem, Γ will consist of just two points, say $x_1 = 0$ and $x_1 = \ell$, and characteristic theory [15] gives information on the type of boundary conditions which are appropriate to ensure that the problem is correctly defined.

2.1. Variational Formulation

Finite element methods of solution are based upon approximations to a variational formulation of this problem [4]. A variational formulation requires the introduction of a space of trial functions, $\mathcal{T}$, and a space of weighting functions, $\mathcal{W}$. For present purposes, it is sufficient to define these spaces as containing all suitably smooth functions and to be such that

$$\mathcal{T} = \{U(x_1, t)\ |U(x_1, 0) = U^0(x_1) \text{ for } x_1 \in \Omega\} \tag{3}$$

$$\mathcal{W} = \{W(x_1)\} \tag{4}$$

A variational formulation of the problem is then: find $U(x_1, t)$ in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_\Omega W \left(\frac{\partial U}{\partial t} + \frac{\partial F^1}{\partial x_1} \right) dx_1 = 0 \tag{5}$$

for all $W(x_1)$ in $\mathcal{W}$ and for all $t > 0$. The equivalence of this formulation and the classical formulation follows from the fundamental lemma of the variational calculus [16].

It is well known that the linear scalar conservation equation will propagate any discontinuities, which may be present in the initial solution or in the boundary condition, through the solution domain, without change of form. In addition, the non–linear scalar conservation equation can, under certain circumstances, produce discontinuous solutions, even though both the initial conditions and the boundary conditions are smooth. Discontinuous solutions do not satisfy the conservation law of equation (1), or the variational statement of equation (5), in the classical sense, as they are not differentiable. Methods for dealing with discontinuous solutions will be considered further in Section 2.5.

2.2. Galerkin Approximation

To produce an approximate solution to the variational problem, a grid of finite elements is constructed on the domain Ω. It will be assumed that the discretisation employs p nodes, which need not be numbered sequentially and are such that node J is located at $x_1 = x_{1J}$. Elements, joining neighbouring nodes, are also numbered and the standard linear finite element shape function associated with node J is denoted by $N_J(x_1)$ [17]. Finite dimensional subspaces $\mathcal{T}^{(p)}$ and $\mathcal{W}^{(p)}$ of the trial and weighting function spaces respectively are defined by

$$\mathcal{T}^{(p)} = \{U^{(p)}(x_1, t)\ |U^{(p)}(x_1, t) = \sum_{J=1}^{p} U_J(t) N_J(x_1); U_J(0) = U^0(x_{1J})\} \tag{6}$$

$$\mathcal{W}^{(p)} = \{W(x_1)\ |W(x_1) = \sum_{J=1}^{p} a_J N_J(x_1)\} \tag{7}$$

Here $U_J(t)$ is the value of $U^{(p)}$ at node J and $a_1, a_2, \cdots, a_p$ are constants. The question of boundary condition imposition should also be addressed at this point but, to avoid notational complexity, the manner of implementation is not considered further here or in the remainder of this review. The Galerkin approximate solution follows by using the variational statement of equation (5) in the form: find $U^{(p)}$ in $\mathcal{T}^{(p)}$, satisfying the problem boundary conditions, such that

$$\int_\Omega W \frac{\partial U^{(p)}}{\partial t} dx_1 = - \int_\Omega W \frac{\partial F^1}{\partial x_1} dx_1 \tag{8}$$

for every W in $\mathcal{W}^{(p)}$ and for all $t > 0$. Here F^1 is now to be interpreted as $F^1(U^{(p)})$. Since the functions $N_1, N_2, \cdots, N_p$ form a basis for $\mathcal{W}^{(p)}$, it is clear that the Galerkin approximation $U^{(p)}$ is, alternatively, defined by the requirement that

$$\int_\Omega N_I \frac{\partial U^{(p)}}{\partial t} dx_1 = - \int_\Omega N_I \frac{\partial F^1}{\partial x_1} dx_1 \tag{9}$$

for $I = 1, 2, \cdots, p$ and for all $t > 0$. The form assumed for $U^{(p)}$ in equation (6) can now be inserted into the left hand side of this equation and the result written as

$$\sum_{J=1}^{p} \left(\int_{\Omega} N_I N_J dx_1 \right) \frac{\mathrm{d}U_J}{\mathrm{d}t} = - \int_{\Omega} N_I \frac{\partial F^1}{\partial x_1} dx_1 \tag{10}$$

Both integrals are evaluated using the standard finite element approach of summing the contributions from the individual elements [17]. On the left hand side, the resulting element integrals are evaluated exactly. The integral on the right hand side has to be evaluated approximately. Here numerical quadrature may be used or, alternatively, the assumption that $F^1(U^{(p)})$ also varies linearly over each element may be adopted. In the latter case, the result at a typical interior node I is that

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = - \sum_{E \in I} \sum_{J \in E} \left[\int_{\Omega_E} N_I \frac{\mathrm{d}N_J}{\mathrm{d}x_1} dx_1 \right] F_J^1 \tag{11}$$

The first summation on the right hand side extends over the two elements E in the grid which contain node I and the second summation extends over both nodes J of element E. A suitable data structure for use with this equation is then the prescription of the nodes which belong to each element, E. In equation (11), Ω_E is that portion of Ω which is represented by element E, M_{IJ} denotes the entry in row I and column J of the non–diagonal finite element consistent mass matrix, $\mathbf{M}$, while

$$F_J^1 = F^1(U^{(p)}) \Big|_J = F^1(U_J) \tag{12}$$

It should be noted that the numerical scheme represented by equation (11) is conservative, in the sense that summing all such equations shows that the change in the total amount of $U^{(p)}$ which is present in Ω is due solely to inflow and outflow at the boundaries.

Alternatively, integration by parts can be employed in the right hand side of equation (9), and in this case the Galerkin equation corresponding to a typical interior node I, becomes

$$\int_{\Omega} N_I \frac{\partial U^{(p)}}{\partial t} dx_1 = \int_{\Omega} \frac{\mathrm{d}N_I}{\mathrm{d}x_1} F^1 dx_1 \tag{13}$$

When the form assumed for $U^{(p)}$ in equation (6) is now inserted into this equation, the result is that

$$\sum_{J=1}^{p} \left(\int_{\Omega} N_I N_J dx_1 \right) \frac{\mathrm{d}U_J}{\mathrm{d}t} = \int_{\Omega} \frac{\mathrm{d}N_I}{\mathrm{d}x_1} F^1 dx_1 \tag{14}$$

Again the left hand side integral is evaluated exactly, while an approximate evaluation is made of the right hand side integral. For example, with the integrand on each element approximated by a trapezoidal rule, the result is that

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = \sum_{e=1}^{2} F_{II_e}^1 \tag{15}$$

where

$$F^1_{II_e} = C^1_{II_e}\left(F^1_I + F^1_{I_e}\right) \tag{16}$$

In forming equation (15), it has been assumed that node I in the grid is connected to the 2 nodes I_1 and I_2 and the summation on the right hand side extends over the two elements e in the grid which contain node I. The coefficient $C^1_{II_e}$ denotes the weight which must be applied to the sum of the nodal fluxes on element e to obtain the contribution made by this element to node I. With this notation, $C^1_{I_eI}$ will denote the weight which must be applied to the same quantity to obtain the contribution made by the flux and the element e to the node I_e. If h_{II_e} denotes the length of element e, the values of these weight coefficients are computed as

$$C^1_{II_e} = \frac{h_{II_e}}{2}\left.\frac{\mathrm{d}N_I}{\mathrm{d}x_1}\right|_e \qquad\qquad C^1_{I_eI} = \frac{h_{II_e}}{2}\left.\frac{\mathrm{d}N_{I_e}}{\mathrm{d}x_1}\right|_e \tag{17}$$

and it is readily verified that

$$\sum_{e=1}^{2} C^1_{II_e} = 0 \tag{18}$$

Since $N_I + N_{I_e} = 1$ on element e, it follows, from equation (17), that the numerical procedure in this form is conservative, in the sense that the total contribution made by any element is zero.

It should be noted that, at a typical interior node I, the formulations of equation (11) and equation (15) lead, in this case, to exactly the same equations. A study of these equations shows that the application of the Galerkin method results in a central difference type approximation to the first order spatial derivative.

In the variational formulation which has been followed here, the time and the space dimensions have been accorded a different treatment. The result is a semi–discrete formulation in which only the space dimension has been discretised by the finite element method, with the time dimension remaining undiscretised. This review will concentrate on methods which are based upon following this approach. It should be noted, however, that alternative finite element based solution strategies are also possible. For example, practical solution algorithms for fluid mechanics have been developed based upon either partial or complete least squares [18, 19, 20, 21, 22, 23], or characteristic formulations [24, 25]. Some methods also employ discontinuous Galerkin formulations [26, 4, 27, 28].

2.3. Stabilization

It is well known that the use of the Galerkin method employed here for the discretisation of the spatial derivative, leads to a so–called non–coercive approximation which allows for the appearance of spurious modes in the solution [29, 30]. This is illustrated by considering the application of the procedure to the equation of pure convection, where $F^1 = A^1 U$ and A^1 is a constant. Now, using equation (15), the equation corresponding to a typical interior node I of the grid takes the form

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = A^1 \sum_{e=1}^{2} C^1_{II_e} (U_I + U_{I_e}) \tag{19}$$

It is apparent that this equation is unaltered if a mode, with alternate values of $+a$ and $-a$ at successive nodes, is added to the solution. The implication of this observation is that the numerical scheme needs to be spatially stabilized, to ensure the elimination of such modes, before it can be used for practical computations. In practice, general stabilization is accomplished by adding some form of diffusion to the basic procedure [31].

2.3.1. Explicit Addition of Diffusion Diffusion may be added to the numerical procedure to provide stability while, at the the same time, ensuring that the accuracy of the original approximation is maintained. In the finite difference approach, this is readily accomplished for the general form of equation (1) by applying the spatial discretisation directly to the modified equation

$$\frac{\partial U}{\partial t} + \frac{\partial \mathcal{F}}{\partial x_1} = 0 \tag{20}$$

where $\mathcal{F}$ is a numerical flux function, which satisfies a consistency requirement of the form $\mathcal{F} \to F^1$ as the grid size $h \to 0$. In the present context, an appropriate choice for such a numerical flux function is

$$\mathcal{F} = F^1 + \varepsilon_4^* h^3 |A^1| \frac{\partial^3 U}{\partial x_1^3} \tag{21}$$

where ε_4^* is a positive dimensionless user–specified diffusion coefficient [32] and

$$A^1 = \frac{\mathrm{d}F^1}{\mathrm{d}U} \tag{22}$$

At a typical interior node I of a grid, on which the nodes are consecutively numbered and equally spaced by an amount h, application of a centered finite difference method to the modified equation produces the semi–discrete form

$$\begin{aligned} h\frac{\mathrm{d}U_I}{\mathrm{d}t} = & -\frac{1}{2}\left(F^1_{I+1} - F^1_{I-1}\right) - \varepsilon_4^* \Big[|A^1|_{I+1/2} (U_{I+2} - 3U_{I+1} + 3U_I - U_{I-1}) \\ & - |A^1|_{I-1/2} (U_{I+1} - 3U_I + 3U_{I-1} - U_{I-2})\Big] \end{aligned} \tag{23}$$

This form maintains the accuracy of the original central difference approximation and achieves the necessary stabilization.

It is apparent that the Galerkin method, on a grid of linear finite elements, cannot be used directly to discretise a third or a fourth order derivative. This means that it will not be possible to extend this finite difference approach into the finite element framework by just using the numerical flux function $\mathcal{F}$ from equation (21) to replace the actual flux function F^1 in the Galerkin variational statement of equation (9) or equation (13). A slightly more complicated method is required to achieve the same effect. The actual flux function $F^1_{II_e}$ in equation (15) is replaced by a numerical flux function, defined according to

$$\mathcal{F}_{II_e} = C^1_{II_e}\left(F^1_I + F^1_{I_e}\right) + \varepsilon_4|\lambda_{II_e}|d_{II_e} \tag{24}$$

where

$$d_{II_e} = U_{I_e} - U_I - \left(h\frac{\partial U}{\partial \sigma}\right)_{II_e} \qquad\qquad \lambda_{II_e} = C^1_{II_e}A^1_{II_e} \tag{25}$$

In this expression, σ denotes a local element coordinate, with σ taking the value zero at node I and the value h_{II_e}, the element length, at node I_e. Nodal values of the solution gradient are obtained from a Galerkin procedure as

$$\sum_{J=1}^{p} M_{IJ}\left.\frac{\partial U}{\partial x_1}\right|_J = -\sum_{e=1}^{2} C^1_{II_e}\left(U_I + U_{I_e}\right) \tag{26}$$

For computational convenience, this equation set is frequently solved approximately by replacing the consistent mass matrix on the left hand side by a lumped diagonal matrix, with diagonal entries

$$M_I = \int_\Omega N_I dx_1 \tag{27}$$

and, in this case, the nodal gradients are directly established as

$$M_I\left.\frac{\partial U}{\partial x_1}\right|_I = -\sum_{e=1}^{2} C^1_{II_e}\left(U_I + U_{I_e}\right) \tag{28}$$

With the nodal gradients determined, the gradient at II_e is evaluated as the mean of the gradients at the nodes I and I_e.

The modified semi–discrete finite element equation is expressed as

$$\sum_{J=1}^{p} M_{IJ}\frac{\mathrm{d}U_J}{\mathrm{d}t} = \sum_{e=1}^{2}\mathcal{F}_{II_e} \tag{29}$$

and spatial stabilization is again achieved. It is observed that the right hand side of this equation is exactly equivalent, for a grid of elements of equal length h, to the right hand side of equation (23), if $\varepsilon_4^* = \varepsilon_4/8$. It should be noted that the construction of the numerical flux of equations (24) and (25) is such that it results in the addition of zero diffusion, even on a non–uniform grid, if $U^{(p)}$ varies linearly with x_1. This feature ensures that the accuracy of the original approximation on a uniform grid is maintained when it is implemented on a non–uniform grid [33, 34].

2.3.2. Residual Methods An alternative method of achieving stabilization involves the direct addition of appropriate terms to the approximate variational statement. To illustrate how this may be accomplished, the requirement of equation (9) is replaced by

$$\int_\Omega N_I \frac{\partial U^{(p)}}{\partial t} dx_1 = -\int_\Omega N_I \frac{\partial F^1}{\partial x_1} dx_1 - \int_\Omega \mathcal{P}(N_I)\tau\mathcal{R}(U^{(p)}) d\Omega \tag{30}$$

for $I = 1, 2, \cdots, p$ and for all $t > 0$. Here $\mathcal{P}$ represents a differential operator, $\mathcal{R}$ is the residual of the differential equation and τ is a positive element intrinsic time scale which, for element E is chosen to be of order $h_E/|A^1|$, where h_E is the element length.

With the definitions

$$\mathcal{R}(U^{(p)}) = \frac{\partial U^{(p)}}{\partial t} + \frac{\partial F^1}{\partial x_1} \qquad\qquad \mathcal{P}(N_I) = A^1(U^{(p)})\frac{\mathrm{d}N_I}{\mathrm{d}x_1} \tag{31}$$

the resulting approximation procedure is equivalent to a method of streamline upwind Petrov–Galerkin (SUPG) type [35]. Alternative algorithms may be produced by different choices of the operators $\mathcal{P}$ and $\mathcal{R}$ [36]. The approximation $U^{(p)}$ is now required to satisfy the statement

$$\int_\Omega \left(N_I + \tau\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)\frac{\partial U^{(p)}}{\partial t} dx_1 = -\int_\Omega \left(N_I + \tau\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)\frac{\partial F^1}{\partial x_1} dx_1 \tag{32}$$

for $I = 1, 2, \cdots, p$ and for all $t > 0$. Inserting the assumed form for $U^{(p)}$, and evaluating the resulting integrals by summing individual element contributions, this equation can be written as

$$\begin{aligned}&\sum_{E\in I}\sum_{J\in E}\left[\int_{\Omega_E}\left(N_I + \tau\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)N_J dx_1\right]\frac{\mathrm{d}U_J}{\mathrm{d}t}\\ &= -\sum_{E\in I}\sum_{J\in E}\left[\int_{\Omega_E}\left(N_I + \tau\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)\frac{\mathrm{d}N_J}{\mathrm{d}x_1}dx_1\right]F^1_J\end{aligned} \tag{33}$$

It is readily demonstrated that this modification to the Galerkin statement results in a stabilized numerical procedure. It should also be noted that, as the additional term in equation (30) vanishes when $U^{(p)}$ is equal to the exact solution U, the consistency of the solution is not compromised in this modified formulation.

2.4. Discretisation in Time

The stabilized semi–discrete nodal equations which have been produced following the spatial discretisation will be of the form of equation (29) or equation (33). A typical stabilized equation, corresponding to an interior node I, is therefore expressed as

$$\sum_{J=1}^{p} \mathcal{M}_{IJ}\frac{\mathrm{d}U_J}{\mathrm{d}t} = f_I \tag{34}$$

where $\mathcal{M}_{IJ}$ and f_I denote entries in the left hand side matrix and right hand side vector respectively, corresponding to the form of the approximation which has been adopted.

Equation (34) may be integrated immediately via standard explicit or implicit finite difference time stepping procedures.

2.4.1. Methods of Explicit Type The solution may be advanced, over a time interval Δt from time $t = t_n$ to time $t = t_{n+1}$, by a k–stage scheme [31] of the form

$$\begin{aligned}
\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(U_J^{\{1\}} - U_J^n \right) &= \alpha_1 \Delta t f_I^n \\
&\vdots \\
\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(U_J^{\{\jmath\}} - U_J^n \right) &= \alpha_\jmath \Delta t f_I^{\{\jmath-1\}} \\
&\vdots \\
\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(U_J^{\{k\}} - U_J^n \right) &= \Delta t f_I^{\{k-1\}}
\end{aligned} \tag{35}$$

In these equations, the superscript n denotes an evaluation at time $t = t_n$, the superscript $\{\jmath\}$ denotes the solution obtained following stage $\jmath$ of the scheme and $U_I^{n+1} = U_I^{\{k\}}$. The values given to the constants $\alpha_1, \alpha_2, \cdots, \alpha_{k-1}$ determine the characteristics of the scheme. The one–stage form of this procedure is

$$\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(U_J^{n+1} - U_J^n \right) = \Delta t f_I^n \tag{36}$$

which is just the forward Euler method. These are not fully explicit schemes, as the left hand side matrices are not diagonal. Truly explicit schemes are produced by lumping the matrix $\mathcal{M}$ and this process does not change the steady state solution of the equation set. Following the application of lumping, the one–stage scheme of equation (36), for example, becomes

$$\mathcal{M}_I \left(U_I^{n+1} - U_I^n \right) = \Delta t f_I^n \tag{37}$$

where

$$\mathcal{M}_I = \sum_{J=1}^{p} \mathcal{M}_{IJ} \tag{38}$$

It should be observed that the term involving τ in the left hand side of equation (33) is skew–symmetric and, consequently, it disappears when lumping is invoked.

The Lax–Wendroff method [37] is a successful stabilized explicit finite difference method for the solution of the scalar convection equation. The method is second order accurate in both space and time. In the present context, if all the elements in the grid are assumed to be of equal length, the Lax–Wendroff method follows from equation (33) by lumping the left hand side matrix, replacing the time derivative by an explicit forward Euler finite difference approximation and defining the parameter τ according to

$$\tau = \frac{\Delta t}{2} \tag{39}$$

If the left hand side matrix is not lumped, but the term involving τ on the left hand side is neglected, the result is the simplest member of the Taylor–Galerkin family of methods, developed originally by Donéa [38] using a different derivation. At a typical interior node I, this Taylor–Galerkin scheme is expressed as

$$\sum_{J=1}^{p} M_{IJ}\left(U_J^{n+1} - U_J^n\right) = f_I^n \tag{40}$$

where

$$f_I = -\Delta t \sum_{E\in I}\sum_{J\in E}\left[\int_{\Omega_E}\left(N_I + \frac{\Delta t}{2}\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)\frac{\mathrm{d}N_J}{\mathrm{d}x_1}dx_1\right]F_J^1 \tag{41}$$

These timestepping procedures will be conditionally stable. The size of the timestep which must be employed to maintain stability can be investigated by applying the algorithms to the equation of pure convection. For the multistage procedure of equation (35), the allowable timestep size will depend upon the number of stages k and the values adopted for the constants $\alpha_1, \alpha_2, \cdots, \alpha_{k-1}$ [39, 40]. For the Taylor–Galerkin method of equations (40) and (41), the stability criterion takes the form

$$\Delta t \le \min_E\left[\frac{h_E}{\sqrt{3}|A^1|}\right] \tag{42}$$

while the stability range is extended, according to the criterion

$$\Delta t \le \min_E\left[\frac{h_E}{|A^1|}\right] \tag{43}$$

when the Lax–Wendroff method is employed.

In computational fluid dynamics, steady state solutions are frequently obtained by explicit time marching of the unsteady governing equation. In this case, the rate of convergence to the steady solution may be increased by using the stability criterion at each node separately, and then advancing each node at its own local time step. For example, with the Taylor–Galerkin method, the right hand side of equation (41) will be modified in this case to give

$$f_I = -\Delta t_I \sum_{E\in I}\sum_{J\in E}\left[\int_{\Omega_E}\left(N_I + \frac{\Delta t_E}{2}\frac{\mathrm{d}N_I}{\mathrm{d}x_1}A^1\right)\frac{\mathrm{d}N_J}{\mathrm{d}x_1}dx_1\right]F_J^1 \tag{44}$$

where, for the lumped mass form,

$$\Delta t_E = \frac{h_E}{A^1} \tag{45}$$

denotes a local timestep for element E and

$$\Delta t_I = \min_{E \in I}(\Delta t_E) \tag{46}$$

is a local timestep for node I, which is computed as the minimum of the surrounding element timesteps.

2.4.2. Implicit Schemes Backward difference formulae are employed to generate truly implicit schemes which can be used to advance the solution of equation (34). If steady state solutions are of interest, the first order formula

$$\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(U_J^{n+1} - U_J^n \right) = \Delta t f_I^{n+1} \tag{47}$$

may be used, while the second order formula

$$\sum_{J=1}^{p} \mathcal{M}_{IJ} \left(3U_J^{n+1} - 4U_J^n + U_J^{n-1} \right) = 2\Delta t f_I^{n+1} \tag{48}$$

is preferable if time accuracy is important. Both these formulae lead to discretizations which are unconditionally stable for the linear problem.

2.5. Treatment of Discontinuities

To allow for the presence of discontinuities, the concept of a solution has to be extended to include the so–called weak solutions. For an infinite spatial domain, a weak solution will satisfy the conservation law and the initial condition in the integral form

$$\int_{t=0}^{\infty} \int_{\Omega} \left(U \frac{\partial \phi}{\partial t} + F^1 \frac{\partial \phi}{\partial x_1} \right) dx_1 dt + \int_{\Omega} U^0 \left. \phi \right|_{t=0} dx_1 = 0 \tag{49}$$

for all continuously differentiable test functions, $\phi(x_1, t)$, which have compact support [15]. The corresponding Rankine–Hugoniot condition will relate the jump in U across any discontinuity to the speed at which the discontinuity propagates.

Unfortunately, weak solutions are not unique and it is essential to work with numerical schemes which only select the physically relevant solution. This is defined as the solution of the associated viscous equation

$$\frac{\partial U}{\partial t} + \frac{\partial F^1}{\partial x_1} = \varepsilon \frac{\partial^2 U}{\partial x_1^2} \tag{50}$$

which is obtained in the limit $\varepsilon \to 0$. Schemes which have this property are said to be entropy satisfying.

If discontinuities were to be exactly resolved in the approximate solution, the Rankine–Hugoniot condition would need to be used. Generally, discontinuous solutions obtained in the absence of diffusion represent an approximation to a physical continuous

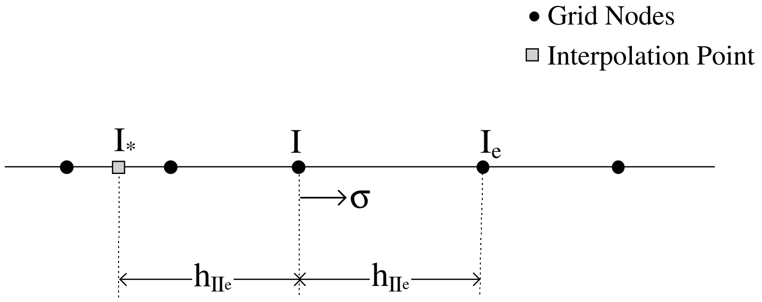

Figure 1. Stencil of points employed in the construction of the discontinuity sensor for node I and element e

viscous solution, involving high gradients over very narrow transition regions. In the discontinuity capturing approach which is followed here, diffusion is added to the original formulation so that the high gradient regions are smeared over two or three grid cells. In this way, the regions of high gradient may be adequately resolved by the grid. The correct smearing is achieved by adding to the numerical procedure a diffusion term with a coefficient which is proportional to the mesh size. It is apparent that this will reduce the basic accuracy of the procedures which have been discussed. Therefore, the added diffusion is constructed in a manner which is designed to produce a diffusion coefficient which is significant near any discontinuity but whose effect is negligible in smooth regions of the flow. Stable consistent finite element schemes of the type that have been discussed, can be modified to automatically capture discontinuities. The fact that the captured discontinuities will be of the correct strength and will propagate at the correct speed is a consequence of the the Lax–Wendroff theorem [37], which states that a conservative numerical scheme, which converges as $h \to 0$ with $\Delta t/h$ fixed, converges to a weak solution of the conservation law.

2.5.1. Explicit Addition of Diffusion Discontinuity capturing may be achieved by the direct addition of diffusion to the existing schemes. A scheme, in the form of equation (29), is produced by adding a suitably scaled second order diffusion operator to the flux function of equation (24). The resulting semi–discrete finite element equation is expressed as

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}U_J}{\mathrm{d}t} = \sum_{e=1}^{2} \tilde{\mathcal{F}}_{II_e} \tag{51}$$

where the modified numerical flux function is defined by

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 |\lambda_{II_e}| (U_{I_e} - U_I) \tag{52}$$

and ε_2 is a user–specified constant. This scheme is, in general, too diffusive for practical computation. However, the scheme can be modified, by the introduction of a discontinuity sensor, so that the accuracy of the original centered scheme is retained in the smooth regions of the flow. In this case, the resulting flux function is written as

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} |\lambda_{II_e}| \left(U_{I_e} - U_I \right) \tag{53}$$

where χ is the discontinuity sensor. This sensor should be designed in such a way that χ is significant only in the vicinity of the discontinuities. Assuming a grid with equally spaced nodes, a scaled approximation to the second derivative of the solution is used in the definition [41]

$$\chi_I = \frac{\left| \sum_{e=1}^{2} (U_{I_e} - U_I) \right|}{\sum_{e=1}^{2} (|U_{I_e}| + |U_I|)} \tag{54}$$

This can be modified, to take account of a non–uniform grid, by computing a sensor at node I, for element e, according to

$$\chi_I = \frac{|U_{I_e} - 2U_I + U_{I_*}|}{|U_{I_e}| + 2\,|U_I| + |U_{I_*}|} \tag{55}$$

In this equation, the point I_* is located such that the element coordinate σ, defined in equation (25), takes the value $-h_{II_e}$ at I_*, as illustrated in Figure 1. The value of U_{I_*} is computed, using a central difference approximation and the solution gradient at node I obtained from equation (28), from the relation

$$\left. \frac{\partial U}{\partial x_1} \right|_I = \frac{(U_{I_e} - U_{I_*})}{2h_{II_e}} \tag{56}$$

The value χ_{II_e} which is required in the definition of equation (53), is computed as the mean of the values of the sensor at nodes I and I_e. A number of alternative methods for specifying the sensor have also been proposed [42, 43].

It should be noted that the full form of the numerical flux function, including the discontinuity capturing capability, is

$$\tilde{\mathcal{F}}_{II_e} = C^1_{II_e} \left(F^1_I + F^1_{I_e} \right) + |\lambda_{II_e}| \left\{ \varepsilon_2 \chi_{II_e} (U_{I_e} - U_I) + \varepsilon_4 d_{II_e} \right\} \tag{57}$$

To avoid the appearance of oscillations in the numerical solution, only the discontinuity capturing diffusion should be added in regions where the solution exhibits strong gradients [44, 45]. This is achieved by user specification of the constant values ε_2 and e_4 and the use of an expression such as

$$\varepsilon_4 = \max \left\{ e_4 - \min \left(\chi_{II_e} \varepsilon_2, 1.0 \right), 0.0 \right\} \tag{58}$$

to determine the appropriate value for the coefficient ε_4.

The multistage timestepping procedure of equation (35) provides a readily implementable method for advancing the solution of equation (51) in time.

2.5.2. Residual Methods For any stabilized scheme, discontinuity capturing can also be achieved by the addition of an artificial diffusion term based upon the equation residual [46]. For example, when discontinuity capturing of this type is used in conjunction with the Taylor–Galerkin method, the timestepping scheme of equation (40) is modified, by the addition of an explicit form of the diffusion to give [47],

$$\sum_{J=1}^{p} M_{IJ}\left(U_J^{n+1} - U_J^n\right) = \tilde{f}_I^n \tag{59}$$

where

$$\tilde{f}_I^n = f_I^n - \Delta t \sum_{E \in I} \sum_{J \in E} \left[\int_{\Omega_E} \delta_E^n \frac{\mathrm{d}N_I}{\mathrm{d}x_1} \frac{\mathrm{d}N_J}{\mathrm{d}x_1} dx_1 \right] U_J^n \tag{60}$$

In this equation,

$$\delta_E = \varepsilon_2 h_E \frac{|\mathcal{R}_E|}{|\mathcal{U}_E|} \qquad \mathcal{U} = \frac{\partial U^{(p)}}{\partial x_1} \tag{61}$$

where ε_2 is a user specified constant, and the approximation

$$\mathcal{R}^n = \frac{1}{\Delta t}\left(U^{(p)n} - U^{(p)n-1}\right) + \left.\frac{\partial F^1}{\partial x_1}\right|^n \tag{62}$$

is adopted. Since $\mathcal{R}$ is the residual of the actual, not the modified, equation, it can be expected that a diffusion operator of this form will prove to be successful since the value of $\mathcal{R}$ will be small in smooth regions of the flow and larger in the vicinity of discontinuities.

A computationally convenient method of implementing equation (59) is to employ k passes of the explicit iteration [48]

$$M_I U_I^{\{\jmath\}} = \sum_{J=1}^{p} M_{IJ}\left(U_J^n - U_J^{\{\jmath-1\}}\right) + M_I U_I^{\{\jmath-1\}} + \tilde{f}_I^n \qquad \jmath = 1, 2, \ldots, k \tag{63}$$

with $U_J^{\{0\}} = U_J^n$ and $U_J^{n+1} = U_J^{\{k\}}$. In practice, only a few iterations of this procedure are needed to obtain sufficient convergence.

2.5.3. Total Variation Diminishing Schemes With sufficient addition of diffusion, these numerical schemes will smear discontinuities and produce oscillation free solutions. However, this may be achieved at the expense of reducing the overall accuracy of the solution. The ideas of flux corrected transport (FCT) represented the first attempt at minimising the amount of added diffusion, while maintaining oscillation free solutions [49]. The FCT approach employs a smooth low accuracy solution together with a higher accuracy solution and blends these together in such a way that the solution at time t_{n+1} contains no extrema which are not present in either the solution at time t_n or

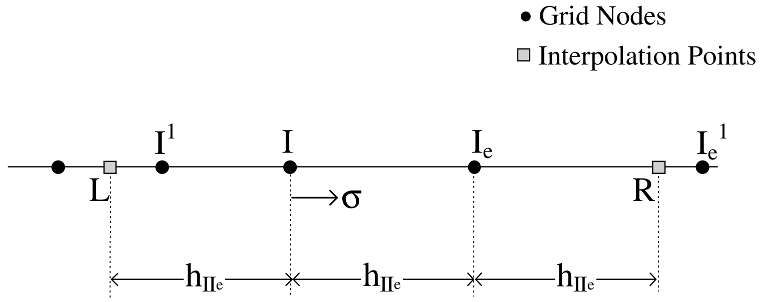

Figure 2. Stencil of points employed in the construction of the higher accuracy TVD algorithm

in the low accuracy solution at time t_{n+1}. A multidimensional extension [50] forms the basis of an unstructured grid finite element implementation [51, 52].

Total variation diminishing algorithms provide a more sophisticated procedure for minimising the amount of added diffusion. These algorithms are based on the fact that for general non–linear scalar equations of the form of equation (1), the total variation of the solution, defined as

$$TV(U) = \int \left| \frac{\partial U}{\partial x_1} \right| dx_1 \tag{64}$$

is a non–increasing function of time i.e.

$$\frac{\mathrm{d}}{\mathrm{d}t} \left[TV(U) \right] \leq 0 \tag{65}$$

Here, it is intended that the integral in equation (64) should be performed over a domain which is bounded by any two characteristic curves. In a similar fashion, for an equally spaced grid on which the nodes are sequentially numbered, the total variation of a discrete numerical solution, at time level t_n, is defined as

$$TV(U^n) = \sum_J \left| U^n_{J+1} - U^n_J \right| \tag{66}$$

Total variation diminishing (TVD) numerical schemes [53] are defined to be algorithms for which the computed solution satisfies

$$TV(U^{n+1}) \leq TV(U^n) \tag{67}$$

A semi–discrete numerical scheme written in the form

$$M_I \frac{\mathrm{d}U_I}{\mathrm{d}t} = \sum_{e=1}^{2} D_{II_e}(U_{I_e} - U_I) \tag{68}$$

will be TVD provided that all the coefficients $D_{II_e} \geq 0$. Clearly, a modified lumped Galerkin finite element scheme of the form

$$M_I \frac{\mathrm{d}U_I}{\mathrm{d}t} = \sum_{e=1}^{2} \mathcal{F}_{II_e} = \sum_{e=1}^{2} \left[C^1_{II_e} \left(F^1_I + F^1_{I_e} \right) + |C^1_{II_e} A^1_{II_e}| (U_{I_e} - U_I) \right] \tag{69}$$

with $A^1_{II_e}$ determined here as

$$A^1_{II_e} = \begin{cases} (F^1_{I_e} - F^1_I)/(U_{I_e} - U_I) & U_{I_e} \neq U_I \\ A^1_I & U_{I_e} = U_I \end{cases} \tag{70}$$

will be TVD since, in this case, the corresponding coefficients

$$D_{II_e} = C^1_{II_e} A^1_{II_e} + |C^1_{II_e} A^1_{II_e}| \tag{71}$$

are clearly non–negative.

This scheme, which is a fully upwind method for determining the flux associated with the element e, is too diffusive for practical computation and a more sophisticated approach is required if a higher accuracy TVD scheme is to be produced. To achieve this, consider the modified flux

$$\tilde{\mathcal{F}}_{II_e} = C^1_{II_e} \left(F_I + F_{I_e} \right) + |C^1_{II_e} A^1_{II_e}| \left[\Delta U_{II_e} - \mathcal{L}(\Delta U^-_{II_e}, \Delta U^+_{II_e}) \right] \tag{72}$$

where, for the element e, on a general grid,

$$\begin{aligned} \Delta U_{II_e} &= U_{I_e} - U_I \\ \Delta U^+_{II_e} &= U_R - U_{I_e} &&= \varepsilon^+_{II_e} \left(U_{I^1_e} - U_{I_e} \right) \\ \Delta U^-_{II_e} &= U_I - U_L &&= \varepsilon^-_{II_e} \left(U_I - U_{I^1} \right) \end{aligned} \tag{73}$$

In this notation, the element coordinate, σ, takes the value $-h_{II_e}$ at the point L, while σ takes the value $2h_{II_e}$ at the point R. The points I^1 and I^1_e are the other grid nodes connected to nodes I and I_e respectively, as illustrated in Figure 2. The coefficients in equation (73) are defined as

$$\varepsilon^+_{II_e} = \frac{x_R - x_{I_e}}{x_{I^1_e} - x_{I_e}} \qquad \varepsilon^-_{II_e} = \frac{x_L - x_I}{x_{I^1} - x_I} \tag{74}$$

It should be noted that, on a uniform grid, where the point L coincides with node I^1 and the point R coincides with the node I^1_e, these coefficients are equal to unity. In equation (72), $\mathcal{L}$ denotes a limiter function and, by correct choice of this limiter, the resulting scheme may be shown to be TVD and to retain the accuracy of the original centered scheme in smooth regions of the flow. A possible definition of the limiter function is

$$\mathcal{L}(a,b) = \begin{cases} \mathrm{sign}(a) \min(|a|, |b|) & \text{if } \mathrm{sign}(a) = \mathrm{sign}(b) \\ 0 & \text{otherwise} \end{cases} \tag{75}$$

and, in this case, it can be shown that

$$D_{II_e} = C^1_{II_e} A^1_{II_e} + \left| C^1_{II_e} A^1_{II_e} \right| + \left| C^1_{II^1} A^1_{II^1} \right| \varepsilon^-_{II^1} \mathcal{L} \left(1, \frac{\Delta U^+_{II^1}}{\Delta U^-_{II^1}} \right) \tag{76}$$

which is clearly positive, since the coefficients $\varepsilon^+_{II_e}$ and $\varepsilon^-_{II_e}$ are both positive. Other definitions for allowable limiter functions may be found in the literature [1].

Additional constraints are necessary if the TVD property is to be maintained in a fully discrete scheme. For example, forward Euler timestepping provides a simple method of advancing the resulting equations and, in this case, the additional constraint takes the form

$$1 - \Delta t \sum_{e=1}^{2} D_{II_e} > 0 \tag{77}$$

It is apparent that this constraint places a limit on the allowable timestep size.

2.6. Extension to Equation Systems

The ideas which have been developed for the one dimensional scalar conservation equation need to be extended to handle problems involving coupled systems of conservation laws, before they can begin to be applied in the area of general computational fluid mechanics. To illustrate how this can be achieved, consider the one–dimensional compressible Euler equations. Following a non–dimensionalisation, based upon a representative length and the velocity and the density of the free stream, these equations may be written in the conservation form

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^1}{\partial x_1} = 0 \tag{78}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho \epsilon \end{bmatrix} \qquad \mathbf{F}^1 = \mathbf{F}^1(\mathbf{U}) = \begin{bmatrix} \rho u_1 \\ \rho u_1^2 + p \\ (\rho\epsilon + p) u_1 \end{bmatrix} \tag{79}$$

In these equations, ρ, p and ϵ represent the fluid density, pressure and total specific energy respectively and u_1 is the fluid velocity. The equation set is completed by the addition of the perfect gas state equation

$$p = (\gamma - 1)\rho \left(\epsilon - 0.5 u_1^2 \right) \tag{80}$$

where γ denotes the ratio of the specific heats of the fluid. Equation (78) may be expressed in the alternative form

$$\frac{\partial \mathbf{U}}{\partial t} + \mathbf{A}^1 \frac{\partial \mathbf{U}}{\partial x_1} = 0 \tag{81}$$

where the 3×3 matrix $\mathbf{A}^1$ is determined as

$$\mathbf{A}^1 = \mathbf{A}^1(\mathbf{U}) = \frac{\mathrm{d}\mathbf{F}^1}{\mathrm{d}\mathbf{U}} \tag{82}$$

The eigenvalues, $\varpi^j, j = 1, 2, 3$, of the matrix $\mathbf{A}^1$ are given by

$$\varpi_1 = u_1 - c \qquad \varpi_2 = u_1 \qquad \varpi_3 = u_1 + c \tag{83}$$

where

$$c = \left(\frac{\gamma p}{\rho}\right)^{1/2} \tag{84}$$

is the local speed of sound in the fluid. Since these eigenvalues are real and distinct, it follows that the equation system (78) is totally hyperbolic.

The solution of this equation set is sought in a finite spatial domain Ω, for all values of the time $t > 0$, and is subjected to the initial condition

$$\mathbf{U}(x_1, 0) = \mathbf{U}^0(x_1) \qquad x_1 \in \Omega \tag{85}$$

where $\mathbf{U}^0$ is a prescribed function, and to boundary conditions at the points $x_1 = 0$ and $x_1 = \ell$. Again, the theory of characteristics [15] determines the form of the necessary boundary conditions which will ensure a well–posed problem.

2.6.1. Variational Formulation A variational formulation is developed following the introduction of a space of vector trial functions, $\mathcal{T}$, and a space of vector weighting functions, $\mathcal{W}$. These spaces contain all suitably smooth functions and, in addition, the space $\mathcal{T}$ will be such that

$$\mathcal{T} = \{\mathbf{U}(x_1, t) \; | \mathbf{U}(x_1, 0) = \mathbf{U}^0(x_1) \text{ for } x_1 \in \Omega\} \tag{86}$$

The problem may then be stated in the variational form: find $\mathbf{U}(x_1, t)$ in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_\Omega \mathbf{W}^T \left(\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}^1}{\partial x_1}\right) dx_1 = 0 \tag{87}$$

for all $\mathbf{W}(x_1)$ in $\mathcal{W}$ and for all $t > 0$.

Following the discretisation of the spatial domain Ω into a grid of linear finite elements, consisting of p nodes, finite dimensional subspaces $\mathcal{T}^{(p)}$ and $\mathcal{W}^{(p)}$ of the trial and weighting function spaces respectively are defined by

$$\mathcal{T}^{(p)} = \{\mathbf{U}^{(p)}(x_1, t) \; | \mathbf{U}^{(p)}(x_1, t) = \sum_{J=1}^{p} \mathbf{U}_J(t) N_J(x_1); \mathbf{U}_J(0) = \mathbf{U}^0(x_{1(J)})\} \tag{88}$$

$$\mathcal{W}^{(p)} = \{\mathbf{W}(x_1) \; | \mathbf{W}(x_1) = \sum_{J=1}^{p} \mathbf{a}_J N_J(x_1)\} \tag{89}$$

Here $\mathbf{U}_J(t)$ is the value of $\mathbf{U}^{(p)}$ at node J. The Galerkin approximate solution follows by using the variational statement of equation (87) in the form: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$, satisfying the problem boundary conditions, such that

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} dx_1 = -\int_\Omega N_I \frac{\partial \mathbf{F}^1}{\partial x_1} dx_1 \tag{90}$$

for $I = 1, 2, \cdots, p$ and for all $t > 0$. Here $\mathbf{F}^1$ is to be interpreted as $\mathbf{F}^1(\mathbf{U}^{(p)})$. In the notation of equation (11), the direct insertion of the form assumed for $\mathbf{U}^{(p)}$ in equation (88) leads, at a typical interior node I, to the equation

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = -\sum_{E \in I} \sum_{J \in E} \left[\int_{\Omega_E} N_I \frac{\mathrm{d}N_J}{\mathrm{d}x_1} dx_1 \right] \mathbf{F}^1_J \tag{91}$$

In the alternative formulation, in which integration by parts is used in the right hand side of equation (90), the Galerkin equation, corresponding to a typical interior node I, becomes

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} dx_1 = \int_\Omega \frac{\mathrm{d}N_I}{\mathrm{d}x_1} \mathbf{F}^1 dx_1 \tag{92}$$

Inserting the assumed for $\mathbf{U}^{(p)}$ produces, in the notation of equation (15),

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{2} \mathbf{F}^1_{II_e} \tag{93}$$

where

$$\mathbf{F}^1_{II_e} = C^1_{II_e} \left(\mathbf{F}^1_I + \mathbf{F}^1_{I_e} \right) \tag{94}$$

Here, the weight coefficients are exactly the same as those defined in equation (17). Following appropriate modification, to account for the fact that the unknown is now a vector, the Galerkin statement of equation (91) or equation (93) can again be made the basis of a solution algorithm, by using the stabilization techniques introduced in Section 2.3.

2.6.2. Added Diffusion Methods An algorithm for smooth flows is constructed by the explicit addition of diffusion, as in Section 2.3.1. The result is that equation (93) is replaced by

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{2} \mathcal{F}_{II_e} \tag{95}$$

where the numerical flux function is defined here, by direct extension from equations (24) and (25), as

$$\mathcal{F}_{II_e} = C^1_{II_e} \left(\mathbf{F}^1_I + \mathbf{F}^1_{I_e} \right) + \epsilon_4 |\lambda_{II_e}| \mathbf{d}_{II_e} \tag{96}$$

In this expression, λ is computed, using the maximum eigenvalue of $\mathbf{A}^1$ in absolute value, as

$$\lambda_{II_e} = C^1_{II_e}\left(|u_{1II_e}| + c_{II_e}\right) \tag{97}$$

and

$$\mathbf{d}_{II_e} = \mathbf{U}_{I_e} - \mathbf{U}_I - \left(h\frac{\partial \mathbf{U}}{\partial \sigma}\right)_{II_e} \tag{98}$$

In this form, it is apparent that the same scalar diffusion coefficient is applied to each equation. An alternative [43], is to employ a matrix diffusion coefficient and to express the numerical flux function as

$$\mathcal{F}_{II_e} = C^1_{II_e}\left(\mathbf{F}^1_I + \mathbf{F}^1_{I_e}\right) + \epsilon_4\left|\mathbf{A}_{II_e}\right|\mathbf{d}_{II_e} \tag{99}$$

where

$$\mathbf{A}_{II_e} = C^1_{II_e}\mathbf{A}^1_{II_e} \tag{100}$$

In equation (99), $|\mathbf{A}|$ denotes the matrix which has the same eigenvectors as $\mathbf{A}$ and whose eigenvalues are those of $\mathbf{A}$ in absolute value. For the element e, this matrix is evaluated at some average between the nodal states $\mathbf{U}_I$ and $\mathbf{U}_{I_e}$. A suitable average state is that introduced by Roe [54], which is defined by

$$\begin{aligned} \rho^R_{II_e} &= \sqrt{\frac{\rho_{I_e}}{\rho_I}} \\ u^R_{II_e} &= \frac{(u_1\sqrt{\rho})_{I_e} + (u_1\sqrt{\rho})_I}{\sqrt{\rho_{I_e}} + \sqrt{\rho_I}} \\ H^R_{II_e} &= \frac{(H\sqrt{\rho})_{I_e} + (H\sqrt{\rho})_I}{\sqrt{\rho_{I_e}} + \sqrt{\rho_I}} \end{aligned} \tag{101}$$

In these relations,

$$H = \frac{c^2}{(\gamma - 1)} + \frac{u_1^2}{2} \tag{102}$$

denotes the specific total enthalpy. It should be noted that this average also provides a means of linearizing equation (82) in the form

$$\mathbf{F}(\mathbf{U}_{I_e}) - \mathbf{F}(\mathbf{U}_I) = \mathbf{A}^1(\mathbf{U}^R_{II_e})(\mathbf{U}_{I_e} - \mathbf{U}_I) \tag{103}$$

for arbitrary $\mathbf{U}_I$ and $\mathbf{U}_{I_e}$.

The nodal values of the solution gradient are determined from

$$\sum_{J=1}^{p} M_{IJ}\left.\frac{\partial \mathbf{U}}{\partial x_1}\right|_J = -\sum_{e=1}^{2} C^1_{II_e}(\mathbf{U}_I + \mathbf{U}_{I_e}) \tag{104}$$

and the required gradient at II_e is obtained as the mean of the values computed at nodes I and I_e.

As in the case of the scalar conservation equation, additional diffusion has to be added, to ensure correct discontinuity capturing, before this type of scheme can be used for the simulation of general flows. In this context, following the approach adopted in Section 2.5.1, equation (95) is replaced by

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{2} \tilde{\mathcal{F}}_{II_e} \tag{105}$$

where the numerical flux function $\tilde{\mathcal{F}}_{II_e}$ is defined by

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} |\lambda_{II_e}| (\mathbf{U}_{I_e} - \mathbf{U}_I) \tag{106}$$

This form, with a scalar diffusion coefficient, would be appropriate for use with the flux function $\mathcal{F}_{II_e}$ of equation (96), while the form

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} \left|\mathbf{A}_{II_e}\right| (\mathbf{U}_{I_e} - \mathbf{U}_I) \tag{107}$$

with a matrix diffusion coefficient, could be employed with the flux function of equation (99). With either of these forms, the diffusion which has been added in equation (96) or equation (99) should be removed in the vicinity of discontinuities. This can again be accomplished using the approach of equation (58).

For an equation system, the computation of the discontinuity sensor, χ, causes a further difficulty. Generally, a key variable for the equation system is identified and the sensor is determined using an expression such as that appearing in equation (55), with the value of the selected key variable replacing the value of U. The pressure, p, is a natural choice for the key variable for the compressible Euler equations.

The equation system (105) may be conveniently advanced by the multistage timestepping procedure of equation (35).

2.6.3. A Taylor–Galerkin Scheme A Taylor–Galerkin scheme for the solution of equation (78) is developed by direct extension of equations (40) and (41) as

$$\sum_{J=1}^{p} M_{IJ} \left(\mathbf{U}_J^{n+1} - \mathbf{U}_J^n\right) = \mathbf{f}_I^n \tag{108}$$

where

$$\mathbf{f}_I = -\Delta t \sum_{E \in I} \sum_{J \in E} \left[\int_{\Omega_E} \left(N_I \mathbf{I} + \frac{\Delta t}{2} \frac{\mathrm{d}N_I}{\mathrm{d}x_1} \mathbf{A}^1 \right) \frac{\mathrm{d}N_J}{\mathrm{d}x_1} dx_1 \right] \mathbf{F}_J^1 \tag{109}$$

and $\mathbf{I}$ is the 3×3 unit matrix. To ensure smooth discontinuity capturing, an additional residual based diffusion term is added, in the form of equation (60). This is accomplished by replacing $\mathbf{f}_I^n$ on right hand side of equation (108) by $\tilde{\mathbf{f}}_I^n$, where

$$\tilde{\mathbf{f}}_I^n = \mathbf{f}_I^n - \Delta t \sum_{E \in I} \sum_{J \in E} \left[\int_{\Omega_E} \delta_E^n \frac{\mathrm{d}N_I}{\mathrm{d}x_1} \frac{\mathrm{d}N_J}{\mathrm{d}x_1} dx_1 \right] \mathbf{U}_J^n \tag{110}$$

In this case,

$$\delta_E = \varepsilon_2 h_E \frac{|\mathcal{R}|}{|\mathcal{U}|} \qquad\qquad \mathcal{U} = \frac{\partial \mathbf{U}^{(p)}}{\partial x_1} \tag{111}$$

and

$$\mathcal{R} = \frac{\partial \mathbf{U}^{(p)}}{\partial t} + \frac{\partial \mathbf{F}^1}{\partial x_1} \tag{112}$$

In the evaluation of the terms in equation (111), the approximation employed in equation (62) may again be adopted. A convenient computational implementation for equation (108) is the explicit iteration of equation (63), which is employed here in the form

$$M_I \mathbf{U}_I^{\{\jmath\}} = \sum_{J=1}^{p} M_{IJ} \left(\mathbf{U}_J^n - \mathbf{U}_J^{\{\jmath-1\}} \right) + M_I \mathbf{U}_I^{\{\jmath-1\}} + \tilde{\mathbf{f}}_I^n \qquad \jmath = 1, 2, \ldots, k \tag{113}$$

with $\mathbf{U}_J^{\{0\}} = \mathbf{U}_J^n$ and $\mathbf{U}_J^{n+1} = \mathbf{U}_J^{\{k\}}$.

Genuine SUPG schemes of the form introduced in equation (32) can also be formulated for the solution of conservative equation systems [55].

2.6.4. Total Variation Diminishing Schemes The concept of a TVD algorithm, introduced in context of the scalar conservation equation in Section 2.5.3, may be extended to produce a procedure for the solution of a system of conservation equations. The approach is best illustrated by considering a local decoupling of the equations into characteristic form and applying the previous approach to each of the resulting scalar equations separately. Upon recombination of the equations, this produces the numerical flux function

$$\tilde{\mathcal{F}}_{II_e} = C^1_{II_e} \left(\mathbf{F}_I + \mathbf{F}_{I_e} \right) + \mathbf{X}^{-1}_{II_e} \left| \mathbf{\Lambda}_{II_e} \right| \left[\Delta \mathbf{S}_{II_e} - \mathcal{L}(\Delta \mathbf{S}^+_{II_e}, \Delta \mathbf{S}^-_{II_e}) \right] \tag{114}$$

where

$$\Delta \mathbf{S}_{II_e} = \mathbf{X}_{II_e} \left(\mathbf{U}_{I_e} - \mathbf{U}_I \right) \tag{115}$$

Here $\mathbf{X}_{II_e}$ is the matrix of right eigenvectors of the matrix $\mathbf{A}_{II_e}$, evaluated at the Roe average state of equation (101), and $|\mathbf{\Lambda}_{II_e}|$ is the diagonal matrix whose entries are the eigenvalues of $\mathbf{A}_{II_e}$ in absolute value. In addition,

$$\Delta \mathbf{S}^+_{II_e} = \mathbf{X}_{II_e} \Delta \mathbf{U}^+_{II_e} \qquad\qquad \Delta \mathbf{S}^-_{II_e} = \mathbf{X}_{II_e} \Delta \mathbf{U}^-_{II_e} \tag{116}$$

with $\Delta \mathbf{U}^+_{II_e}$ and $\Delta \mathbf{U}^-_{II_e}$ defined using the notation of equation (73).

3. Domain Discretisation for Multi–Dimensional Problems

The discretisation of a multi–dimensional computational domain can prove to be a major obstacle to the analyst interested in simulating practical problems, where boundaries of complex geometrical shape are frequently encountered. This grid generation problem is simplified if unstructured grids are employed, as fully automatic methods are now available for subdividing general regions into unstructured assemblies of triangles in two dimensions and tetrahedra in three dimensions [10, 11].

3.1. Unstructured Grids

Galerkin finite element methods, which are integral based, are readily implemented on unstructured triangular or tetrahedral grids. The manner of the implementation, however, will depend upon the grid data structure which is employed. The grid may be represented in a number of different ways [56], but only two data structures will be of relevance to this review.

The first approach is the standard finite element representation of the grid, which provides the connectivity of each element, i.e.the numbers of the ordered nodes belonging to each element are prescribed. The discrete equations are then formed by looping over all the elements in the grid and sending element contributions to the respective nodes [17]. This grid data structure is widely employed in finite element solvers, over a wide range of disciplines, and is the format which has been assumed in the construction of the right hand sides in equations (60) and (110).

The second approach represents the grid in terms of an edge data structure. The edges are numbered, the numbers of the two nodes connected by each edge are prescribed and an orientation is associated with each edge. The discrete equations are now formed by looping over all the edges in the grid and sending edge contributions to the respective nodes [42]. This is the format which has been assumed in the construction of the right hand sides in equations (51) and (95). For a general grid of linear tetrahedral elements, the number of elements is approximately 5.5 times the number of nodes, while the number of edges is approximately equal to the number of elements plus the number of nodes. Using these relations, it is possible to estimate that, for three dimensional simulations, distinct computational advantages, in terms of reduced CPU and memory requirements, can be expected to follow when an edge data structure can be employed [57].

3.2. Unstructured Grid Generation

In practice, the methods which have been successfully employed to discretise general computational domains are based upon either the advancing front concept or the Delaunay approach. Both methods start from a triangulation of the boundaries of the computational domain.

The advancing front method [58, 59] discretises the volume by a marching approach in which points and elements are simultaneously created [60, 61]. The key to success is the implementation of an appropriate method of grid control. To this end, the grid is generated so as to meet, as closely as possible, the distribution of grid size defined by a user–specified grid control function. The grid control function is defined by linear interpolation between nodal values specified on a coarse background grid of tetrahdera which covers the computational domain. The background grid can be supplemented by the addition of sources which enable the grid control function to readily specify a requirement for the clustering of elements in the vicinity of points, lines or surfaces. Following the placement of a new point, the creation of an element can only proceed if the element is valid, in the sense that it does not intersect with any other element or face which has already been generated. Thus, an efficient implementation will require the use of a data structure which enables checking operations of this type to be rapidly performed. The alternating digital tree [62] proves to be particulary successful in this context. As a further refinement of the advancing front approach, anisotropy can be built into the generated grids, by an appropriate addition to the grid control function. However, implementation considerations imply that only a limited level of distortion can be achieved in this manner. Recent enhancements to the advancing front approach [63] produce a powerful grid generation capability.

The Delaunay method provides a systematic geometrical procedure for constructing a tetrahedral grid [64, 65]. Strict application of the Delaunay procedure may lead to a grid which violates the integrity of the computational boundary and, to correct this, boundary recovery procedures also need to be provided [66, 67]. A computationally efficient algorithm is produced by removing the requirement for an initial distribution of nodal points and coupling the Delaunay method with a method of nodal point creation [11]. As the quality of the grids produced is similar to that of the advancing front approach [68], it is the Delaunay method which is currently favoured by most analysts, because it requires significantly less computer time to generate a grid of a given size. It should be noted, however, that the computer memory requirements of the Delaunay method are significantly more than those of the advancing front approach.

In computational fluid mechanics, the efficient resolution of many flows requires the use of highly stretched anisotropic grids. Typical examples of flows in this category would be viscous flows involving boundary layers and wakes. From the observations which have been made above, it is apparent that either of the basic unstructured grid generation approaches will require modification before it can be employed to produce suitable grids for such problems. One possible modification is to split the domain into distinct regions in which either isotropic or anisotropic grids are required. The anisotropic region is generated first and, in early work, this was accomplished by using some form of structured mesh generation [69, 70]. Refinements to this approach have also been reported [71, 72, 73]. More recently, attempts have been made at generating an unstructured grid in the anisotropic region by using advancing front concepts to connect

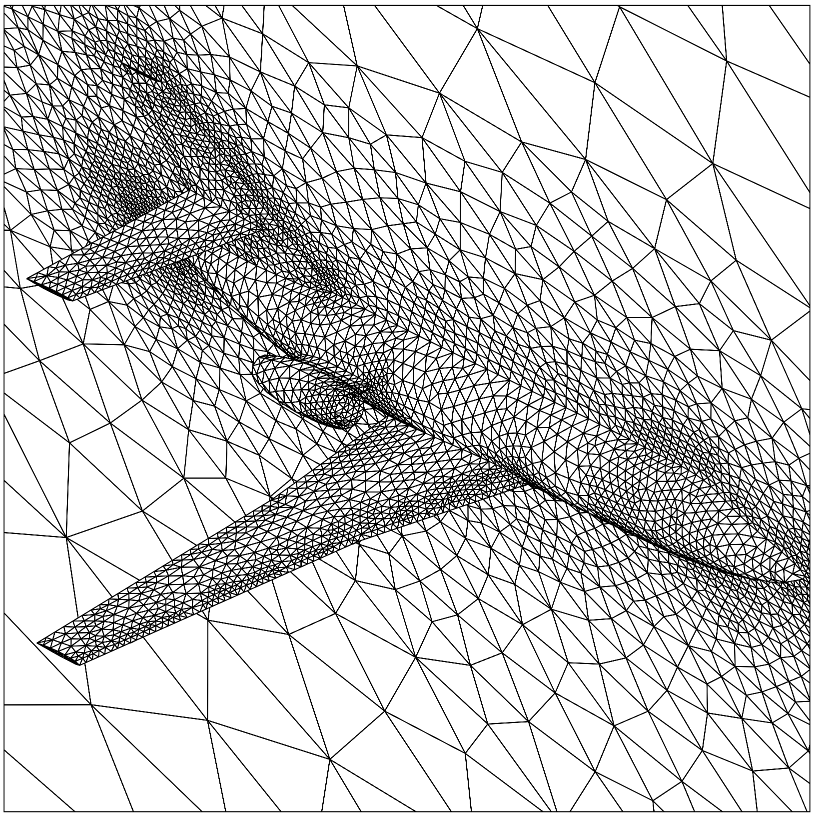

Figure 3. View of the surface discretisation for the initial grid produced for the analysis of viscous flow over a complete aircraft configuration

points which are placed, along computed surface normals, according to a user–specified distribution [74, 75, 76, 77]. In all these approaches, when the anisotropic region is complete, the remainder of the grid is generated by either of the standard methods.

Another alternative is to start from a valid isotropic grid for the computational domain of interest and to modify the grid until it meets user–specified requirements. The modification may be accomplished in terms of prescribed grid operations, such as point creation and reconnection [12, 56]. Figures 3 and 4 show views of a tetrahedral mesh, which has been produced by this approach, for the analysis of viscous flow over a complete aircraft configuration.

4. The Multi–Dimensional Euler Equations

The methods which have been described for the solution of the one dimensional compressible Euler equations in Section 2.6 can be extended to enable the simulation of multi–dimensional inviscid compressible flow. The starting point is to express the Euler equations, relative to a cartesian coordinate system $Ox_1x_2x_3$, in the non–dimensional conservation form

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} = \mathbf{0} \tag{117}$$

Here

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho \epsilon \end{bmatrix} \qquad \mathbf{F}^j = \begin{bmatrix} \rho u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ (\rho\epsilon + p)u_j \end{bmatrix} \tag{118}$$

In these equations, u_j is the component of the fluid velocity in the direction x_j and δ_{ij} is the Kronecker delta. The perfect gas equation of state takes the form

$$p = (\gamma - 1)\rho \left[\epsilon - 0.5(u_1^2 + u_2^2 + u_3^2)\right] \tag{119}$$

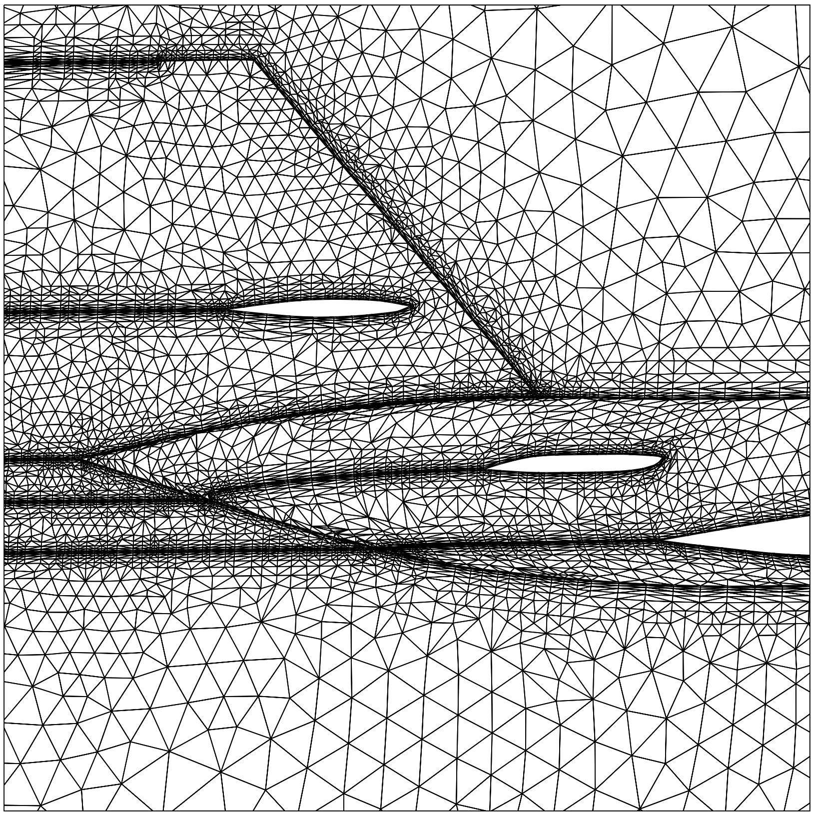

Figure 4. Detail of the surface discretisation on the final grid produced for the analysis of viscous flow over a complete aircraft configuration

The solution of this equation set is sought over a three dimensional spatial domain, Ω, with closed boundary surface Γ, for all $t > 0$. The solution is subject to the initial condition

$$\mathbf{U}(\mathbf{x}, 0) = \mathbf{U}^0(\mathbf{x}) \qquad \mathbf{x} \in \Omega \tag{120}$$

and to appropriate boundary conditions. Following the approach outlined in Section 2.6, relevant trial and weighting function spaces are introduced and the problem is expressed in the variational form: find $\mathbf{U}$ in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_\Omega \mathbf{W}^T \left(\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} \right) d\Omega = \mathbf{0} \tag{121}$$

for every $\mathbf{W}$ in $\mathcal{W}$ and for all $t > 0$. The spatial domain, Ω, is discretised into a general assembly of tetrahedral elements, with nodal points, numbered from 1 to p, located at the element vertices. An approximate solution $\mathbf{U}^{(p)}$ is sought in the finite dimensional trial function space

$$\mathcal{T}^{(p)} = \left\{ \mathbf{U}^{(p)}(\mathbf{x}, t) | \mathbf{U}^{(p)} = \sum_{J=1}^{p} \mathbf{U}_J(t) N_J(\mathbf{x}); \mathbf{U}_J(0) = \mathbf{U}^0(\mathbf{x}_J) \right\} \tag{122}$$

where N_J is the piecewise linear trial function associated with node J, located at $\mathbf{x} = \mathbf{x}_J$. The Galerkin approximation is determined by applying the variational formulation of equation (121) in the form: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$, satisfying the problem boundary conditions, such that

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega = - \sum_{j=1}^{3} \int_\Omega N_I \frac{\partial \mathbf{F}^j}{\partial x_j} d\Omega \tag{123}$$

for $I = 1, 2, \cdots, p$ and for all $t > 0$. Here $\mathbf{F}^j$ is to be interpreted here as $\mathbf{F}^j(\mathbf{U}^{(p)})$. Using the divergence theorem, this Galerkin statement can be replaced by the requirement that

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega = \sum_{j=1}^{3} \int_\Omega \frac{\partial N_I}{\partial x_j} \mathbf{F}^j d\Omega \tag{124}$$

at each interior node I in the grid.

4.1. Edge Based Method with Explicitly Added Diffusion

When the form assumed for $\mathbf{U}^{(p)}$ in equation (122) is substituted into equation (124), the left hand side integral may be evaluated exactly. Numerical quadrature [78], with the integrand over each element sampled at the four vertices with equal weight of 1/4, is used to evaluate approximately the right hand side integral. Suppose that the unstructured tetrahedral grid is represented in terms of an edge based data structure, according to which node I in the grid is directly connected to the μ_I nodes $I_1, I_2, \cdots, I_{\mu_I}$. Then, it can be shown that equation (124) may be expressed as

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{\mu_I} \sum_{j=1}^{3} C^j_{II_e} \left(\mathbf{F}^j_I + \mathbf{F}^j_{I_e} \right) \tag{125}$$

where the first summation on the right hand side extends over the edges e which contain node I. The coefficient $C^j_{II_e}$ denotes the weight which must be applied to the sum of the nodal fluxes, in the x_j direction, on the edge e to obtain the contribution made by this flux and this edge to node I. With this notation, the coefficient $C^j_{I_eI}$ is meant to denote the weight which must be applied to the same quantity in order to obtain the contribution made by the flux in the x_j direction and the edge e to node I_e. The coefficients can be determined as

$$C^j_{II_e} = \sum_{E \in II_e} \frac{\Omega_E}{4} \frac{\partial N_I}{\partial x_j}\bigg|_E \qquad C^j_{I_eI} = \sum_{E \in II_e} \frac{\Omega_E}{4} \frac{\partial N_{I_e}}{\partial x_j}\bigg|_E \tag{126}$$

where here the summation extends over all elements E which contain the edge II_e and Ω_E denotes the volume of element E. It can be verified that these coefficients satisfy the relations

$$\sum_{e=1}^{\mu_I} C^j_{II_e} = 0 \qquad\qquad \text{for } j = 1,2,3 \tag{127}$$

$$C^j_{II_e} + C^j_{I_eI} = 0 \qquad\qquad \text{for } e = 1,2,\cdots,\mu_I \text{ and } j = 1,2,3 \tag{128}$$

From the asymmetry of the edge coefficients, apparent in equation (128), the numerical discretisation is conservative in the sense that the total contribution made by any interior edge in the grid is zero.

Using equation (127), it follows that equation (125) is a central difference type discretisation for the spatial derivatives at the interior node I. As in the one dimensional case, the addition of an appropriate diffusion will be required in order to produce a practically useful scheme. To achieve this, in the current multidimensional context, the actual flux function

$$\mathbf{F}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \left(\mathbf{F}^j_I + \mathbf{F}^j_{I_e} \right) \tag{129}$$

associated with edge e in equation (125) is replaced by a numerical flux function $\mathcal{F}_{II_e}$ and the resulting algorithm is written as

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{\mu_I} \mathcal{F}_{II_e} \tag{130}$$

A simple method of constructing a suitable numerical flux function is to use one dimensional concepts along each edge, employing the ideas introduced in Section 2.

Following this approach, to produce an algorithm for the simulation of smooth flows, equations (96) and (98) suggest the choice

$$\mathcal{F}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \left(\mathbf{F}^j_I + \mathbf{F}^j_{I_e} \right) + \varepsilon_4 |\lambda_{II_e}| \mathbf{d}_{II_e} \tag{131}$$

and

$$\mathbf{d}_{II_e} = \mathbf{U}_{I_e} - \mathbf{U}_I - \left(h \frac{\partial \mathbf{U}}{\partial \sigma} \right)_{II_e} \tag{132}$$

Here σ denotes a local edge coordinate, which takes the value zero at node I and the value of the edge length ($= h_{II_e}$) at node I_e. In equation (131),

$$|\lambda_{II_e}| = \left| \sum_{j=1}^{3} C^j_{II_e} u_{jII_e} \right| + c_{II_e} \left[\sum_{j=1}^{3} \left(C^j_{II_e} \right)^2 \right]^{1/2} \tag{133}$$

is now the maximum eigenvalue, in absolute value, of the matrix $\mathbf{A}_{II_e}$, where

$$\mathbf{A}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \mathbf{A}^j_{II_e} \qquad\qquad \mathbf{A}^j_{II_e} = \left. \frac{d\mathbf{F}^j}{d\mathbf{U}} \right|_{II_e} \tag{134}$$

The alternative form

$$\mathcal{F}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \left(\mathbf{F}^j_I + \mathbf{F}^j_{I_e} \right) + \epsilon_4 \left| \mathbf{A}_{II_e} \right| \mathbf{d}_{II_e} \tag{135}$$

for the numerical flux function follows from the extension of equation (99) and involves a matrix diffusion coefficient. Here $|\mathbf{A}_{II_e}|$ is evaluated at the Roe state [54] between $\mathbf{U}_I$ and $\mathbf{U}_{I_e}$.

The solution gradients at the nodes are determined as

$$\sum_{J=1}^{p} M_{IJ} \left. \frac{\partial \mathbf{U}}{\partial x_j} \right|_I = - \sum_{e=1}^{\mu_I} C^j_{II_e} (\mathbf{U}_I + \mathbf{U}_{I_e}) \tag{136}$$

and the gradient at II_e, required in equation (132), is computed as

$$\left. \frac{\partial \mathbf{U}}{\partial \sigma} \right|_{II_e} = \frac{1}{2} \sum_{j=1}^{3} \left(\left. \frac{\partial \mathbf{U}}{\partial x_j} \right|_I + \left. \frac{\partial \mathbf{U}}{\partial x_j} \right|_{I_e} \right) \left. \frac{\partial x_j}{\partial \sigma} \right|_{II_e} \tag{137}$$

which is the mean of the computed gradients at the nodes I and I_e in the direction of the edge e.

One dimensional concepts may also be employed, along each edge in the grid, to construct a suitable form for the additional diffusion necessary to ensure smooth discontinuity capturing. Thus, following the approach adopted in Section 2.6.2, equation (130) is replaced by

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{II_e} \tag{138}$$

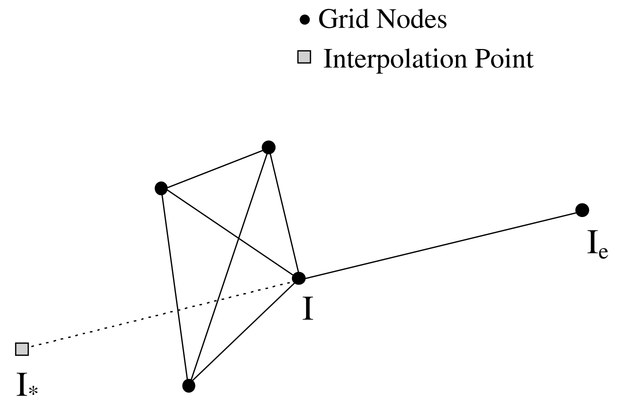

Figure 5. Construction of the stencil of points used to determine the value of the discontinuity sensor for node I and edge e

where, using a scalar diffusion coefficient,

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} \left| \lambda_{II_e} \right| (\mathbf{U}_{I_e} - \mathbf{U}_I) \tag{139}$$

or, in terms of a matrix diffusion,

$$\tilde{\mathcal{F}}_{II_e} = \mathcal{F}_{II_e} + \varepsilon_2 \chi_{II_e} \left| \mathbf{A}_{II_e} \right| (\mathbf{U}_{I_e} - \mathbf{U}_I) \tag{140}$$

The diffusion added in equation (131) or equation (135) should be removed in the vicinity of discontinuities, using the approach of equation (58).

In this multi–dimensional case, a value χ_I for the sensor at node I for edge e is again determined, using equation (55) with the pressure as the selected key variable, from

$$\chi_I = \frac{\left| p_{I_e} - 2p_I + p_{I_*} \right|}{\left| p_{I_e} \right| + 2\left| p_I \right| + \left| p_{I_*} \right|} \tag{141}$$

Now, as illustrated in Figure 5, the point I_* is located on the straight line which passes through the nodes I and I_e, such that the local edge coordinate, σ, takes the value $-h_{II_e}$ at I_*. The value of p_{I_*} is determined, using a central difference approximation and the computed pressure gradient at node I, from the relation

$$\left. \frac{\partial p}{\partial \sigma} \right|_I = \frac{(p_{I_e} - p_{I_*})}{2h_{II_e}} \tag{142}$$

The nodal pressure gradient required in this expression is computed for each edge e, by using the gradients of the conserved variables determined from equation (136) and the perfect gas relationship of equation (119), in the form

$$\left. \frac{\partial p}{\partial \sigma} \right|_I = (\gamma - 1) \left[\frac{\partial(\rho\epsilon)}{\partial \sigma} - \sum_{j=1}^{3} \left\{ u_j \frac{\partial(\rho u_j)}{\partial \sigma} - \frac{1}{2} u_j^2 \frac{\partial \rho}{\partial \sigma} \right\} \right]_I \tag{143}$$

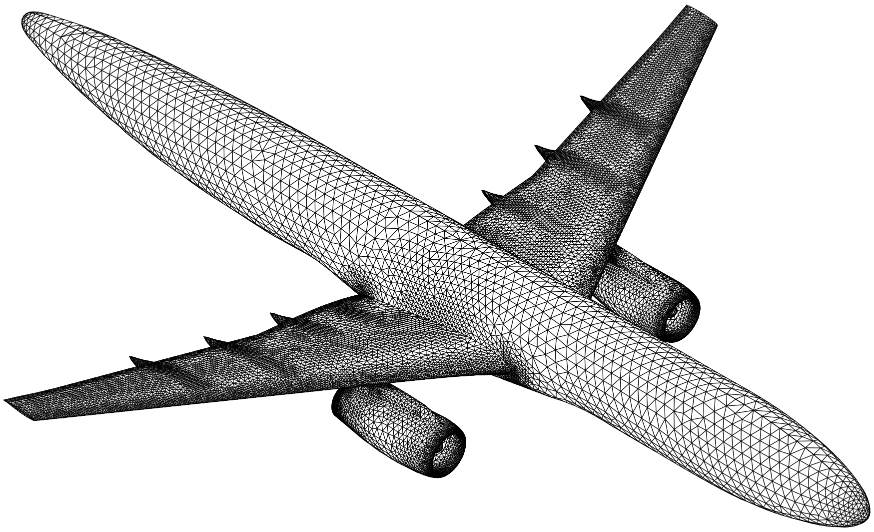

Figure 6. Compressible inviscid flow over an aircraft configuration. View of the surface discretisation

The value χ_{II_e} which is needed in the definition of equations (139) and (140), is obtained as the mean of the values computed for the sensor at nodes I and I_e.

If it is only the computation of steady state solutions which are of interest, the left hand side matrix in equation (138) may be lumped and the solution advanced by the explicit version of the multistage scheme of equation (35). This is a successful scheme for the simulation of steady transonic inviscid flows over general configurations. For example, using this scheme, flow over the aircraft geometry, whose surface discretisation is shown in Figure 6 is simulated at a free stream Mach number of 0.801 and an angle of attack of 2.74 degrees. The computed distribution of the pressure coefficient which is obtained for this flow is shown in Figure 7.

4.2. Element Based Taylor Galerkin Method

A Taylor–Galerkin method for the solution of equation (117) follows from the addition of appropriate terms to the right hand side of equation (123), by extension of the approach followed in equations (109) and (40). The result is the timestepping scheme

$$\sum_{J=1}^{p} M_{IJ}\left(\mathbf{U}_J^{n+1}-\mathbf{U}_J^{n}\right)=\mathbf{f}_I^{n} \tag{144}$$

where

$$\mathbf{f}_I=-\Delta t\sum_{E\in I}\sum_{J\in E}\sum_{j=1}^{3}\left[\int_{\Omega_E}\left(N_I\mathbf{I}+\frac{\Delta t}{2}\sum_{k=1}^{3}\frac{\partial N_I}{\partial x_k}\mathbf{A}^k\right)\frac{\partial N_J}{\partial x_j}d\Omega\right]\mathbf{F}_J^j \tag{145}$$

To ensure smooth discontinuity capturing, a diffusion term must again be added. If the form employed in equation (110) is extended to the multi–dimensional case, the result is that the term $\mathbf{f}_I^n$ on the right hand side of equation (144) is replaced by $\tilde{\mathbf{f}}_I^n$, where

$$\tilde{\mathbf{f}}_I^n=\mathbf{f}_I^n-\Delta t\sum_{E\in I}\sum_{J\in E}\sum_{j=1}^{3}\left[\int_{\Omega_E}\delta_E^n\frac{\partial N_I}{\partial x_j}\frac{\partial N_J}{\partial x_j}d\Omega\right]\mathbf{U}_J^n \tag{146}$$

In this case, δ_E should be chosen to be residual dependent and a simple choice is

$$\delta_E=\varepsilon_2 h_E\frac{|\mathcal{R}_E|}{|\mathcal{U}_E|}\qquad\qquad \mathcal{U}=\sum_{j=1}^{3}\frac{\partial \mathbf{U}^{(p)}}{\partial x_j} \tag{147}$$

and

$$\mathcal{R}=\frac{\partial \mathbf{U}^{(p)}}{\partial t}+\sum_{j=1}^{3}\frac{\partial \mathbf{F}^j}{\partial x_j} \tag{148}$$

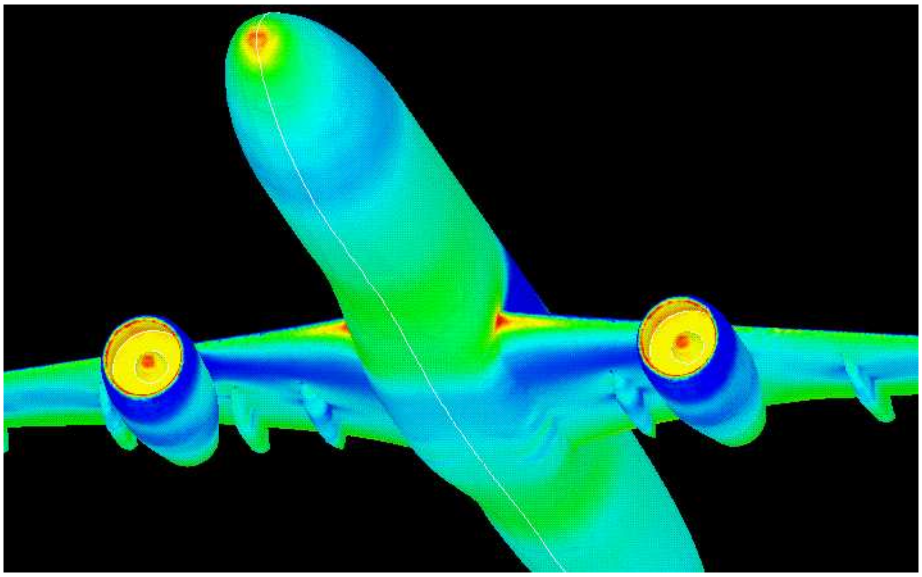

Figure 7. Compressible inviscid flow over an aircraft configuration. Computed distribution of pressure coefficient

In equation (147), h_E denotes a representative length for the element E. In equation (148), an approximation of the form employed in equation (62) may again be utilised. A more elaborate form for the discontinuity capturing diffusion attempts to build the grid metric information directly into the definition of δ [79, 80, 81]. If only steady state solutions are of interest, the left hand side matrix in equation (144) may be lumped and the result is then a truly explicit timestepping procedure.

Genuine SUPG schemes of the form introduced in equation (32) can also be formulated and applied to the solution of the compressible Euler equations, expressed in terms of both conservative and entropy variables [55].

4.3. Edge Based LED Methods

For multidimensional simulations, the TVD ideas introduced in Section 2.5.3 are not directly applicable. However, it is possible to design solution schemes which are based upon the closely related locally extremum diminishing (LED) concept [82]. LED schemes are designed to ensure that any local maximum in the solution will not increase and that any local minimum will not decrease. In one dimension, it is readily demonstrated that an LED scheme is also TVD [83].

To illustrate how LED schemes are constructed for multidimensional simulations, it is convenient, initially, to restrict consideration to the solution of the scalar conservation equation

$$\frac{\partial U}{\partial t} + \sum_{j=1}^{3} \frac{\partial F^j}{\partial x_j} = 0 \tag{149}$$

where $F^j = F^j(U)$. Semi–discrete LED schemes for this equation are defined as those which can be written, in the notation of the edge based data structure, in the form

$$M_I \frac{dU_I}{dt} = \sum_{e=1}^{\mu_I} D_{II_e} \left(U_{I_e} - U_I \right) \tag{150}$$

with all the coefficients $D_{II_e} \geq 0$. This is the direct multidimensional extension of the TVD definition of equation (68). Consider a modified Galerkin finite element scheme of the form

$$M_I \frac{dU_I}{dt} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{II_e} \tag{151}$$

where

$$\tilde{\mathcal{F}}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \left(F^j_I + F^j_{I_e} \right) + \left| \sum_{j=1}^{3} C^j_{II_e} A^j_{II_e} \right| \left(U_{I_e} - U_I \right) \tag{152}$$

When the quantity $A^j_{II_e}$ is determined from

$$A^j_{II_e} = \begin{cases} (F^j_{I_e} - F^j_I)/(U_{I_e} - U_I) & U_{I_e} \neq U_I \\ A^j_I & U_{I_e} = U_I \end{cases} \tag{153}$$

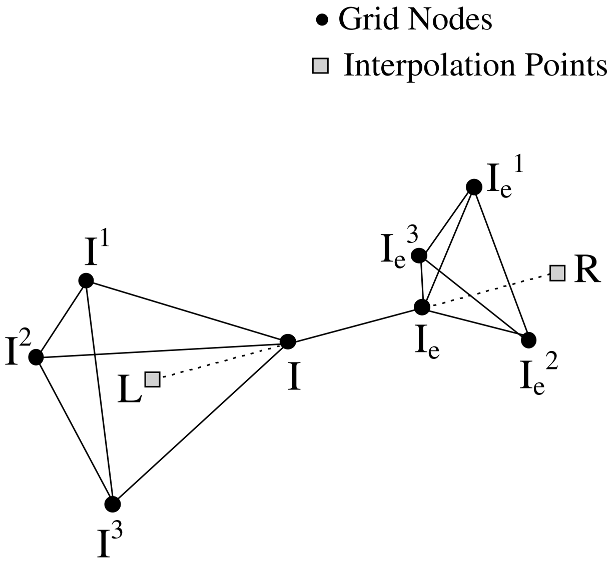

Figure 8. The stencil of points used in the construction of a flux function of higher order accuracy for edge e

the scheme is LED since, in this case,

$$D_{II_e} = \sum_{j=1}^{3} C^j_{II_e} A^j_{II_e} + |\sum_{j=1}^{3} C^j_{II_e} A^j_{II_e}| \tag{154}$$

which is clearly non–negative.

As in the one dimensional case, higher order accuracy is obtained by redefining the numerical flux function as

$$\tilde{\mathcal{F}}_{II_e} = \sum_{j=1}^{3} C^j_{II_e} \left(F^j_I + F^j_{I_e}\right) + |\sum_{j=1}^{3} C^j_{II_e} A^j_{II_e}| \left[\Delta U_{II_e} - \mathcal{L}\left(\Delta U^-_{II_e}, \Delta U^+_{II_e}\right)\right] \tag{155}$$

where, for edge e,

$$\begin{aligned} \Delta U^+_{II_e} &= U_R - U_{I_e} = \sum_{j=1}^{3} \varepsilon^{+j}_{II_e} \left(U_{I^j_e} - U_{I_e}\right) \\ \Delta U^-_{II_e} &= U_I - U_L = \sum_{j=1}^{3} \varepsilon^{-j}_{II_e} \left(U_I - U_{I^j}\right) \end{aligned} \tag{156}$$

In this notation, the point L is located on the straight line which passes through the nodes I and I_e such that the local edge coordinate, σ, takes the value $-h_{II_e}$ at L. Similarly, the point R is located on the same line such that σ takes the value $2h_{II_e}$ at R, as illustrated in Figure 8. The nodes I^j are the nodes of the element, connected to node I, which is used to interpolate or extrapolate for the value of U_L and for which all the coefficients $\varepsilon^{-j}_{II_e}$ are positive. Similarly, the nodes I^j_e are the nodes of the element, connected to node I_e, which is used to interpolate or extrapolate for the value of U_R and for which all the coefficients $\varepsilon_{II_e}^{+j}$ are positive. A limiter function of the form defined in equation (75) may be employed and, in this case, it is readily demonstrated that, when the scheme is expressed in the form of equation (150), the corresponding coefficients D_{II_e} will be non–negative [82].

For the Euler equation system, LED schemes are produced by formally extending the flux function of equation (155), to produce the modified Galerkin finite element scheme

$$M_I \frac{d\mathbf{U}_I}{dt} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{IIe} \tag{157}$$

with the numerical flux function defined by

$$\tilde{\mathcal{F}}_{II_e} = \sum_{j=1}^{3} C_{II_e}^{j} \left(\mathbf{F}_I^j + \mathbf{F}_{I_e}^j \right) + \mathbf{X}_{II_e}^{-1} \left| \mathbf{\Lambda}_{II_e} \right| \left[\Delta \mathbf{S}_{II_e} - \mathcal{L}(\Delta \mathbf{S}_{II_e}^{+}, \Delta \mathbf{S}_{II_e}^{-}) \right] \tag{158}$$

Here $\mathbf{X}_{II_e}$ is the matrix of right eigenvectors of $\mathbf{A}_{II_e}$, evaluated at the Roe average state of equation (101), $\mathbf{\Lambda}_{II_e}$ is the diagonal matrix whose entries are the absolute values of the corresponding eigenvalues and $\Delta\mathbf{S}_{II_e}$ is defined as in equation (115). In addition,

$$\Delta \mathbf{S}_{II_e}^{+} = \mathbf{X}_{II_e} \Delta \mathbf{U}_{II_e}^{+} \qquad\qquad \Delta \mathbf{S}_{II_e}^{-} = \mathbf{X}_{II_e} \Delta \mathbf{U}_{II_e}^{-} \tag{159}$$

with $\Delta\mathbf{U}_{II_e}^{+}$ and $\Delta\mathbf{U}_{II_e}^{-}$ obtained by extending the definition of equation (156) to this case. It is apparent, that this extension to deal with equation systems has been accomplished by applying the treatment outlined for the scalar conservation equation to each of the characteristic variables on every edge in the grid.

Forward Euler timestepping may be used to advance equation (157) in time. Schemes of this form are particularly well–suited to the simulation of hypersonic flows [84]. An example of such an application is given in Figure 9, which shows the computed distribution of the Mach number contours for a steady inviscid hypersonic flow over an aerospace vehicle at a free stream Mach number of 9.8 and an angle of attack of 40 degrees.

5. The Navier Stokes Equations

For laminar viscous compressible flows, following a non–dimensionalisation based upon a representative length and the free stream velocity, density and viscosity, the governing Navier Stokes equations may be written in the conservation form

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} = \sum_{j=1}^{3} \frac{\partial \mathbf{G}^j}{\partial x_j} \tag{160}$$

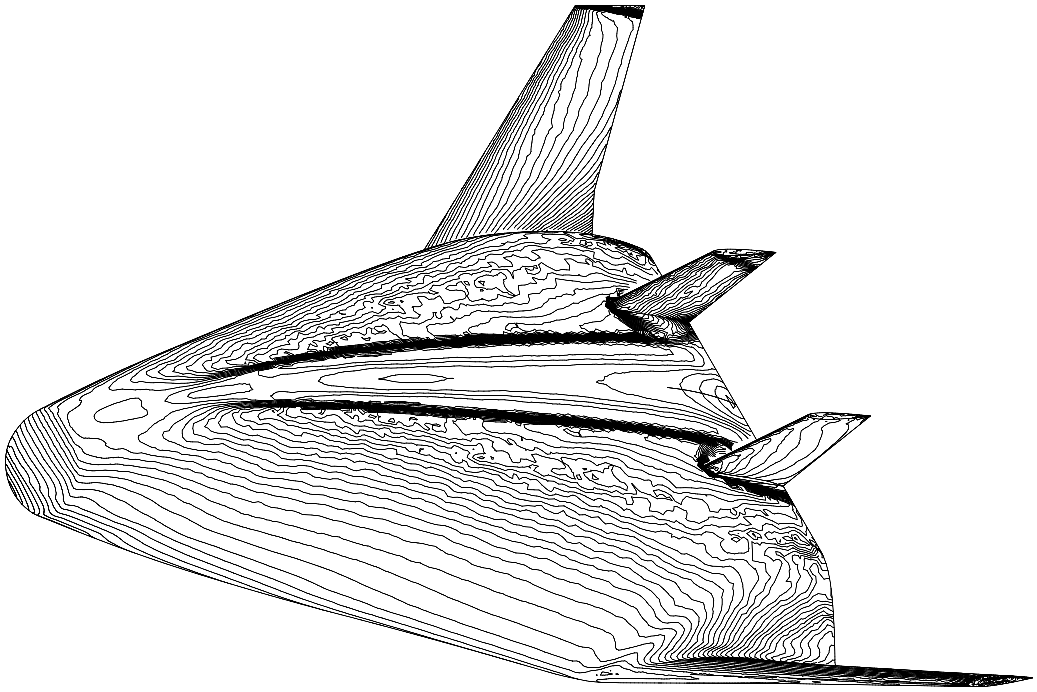

Figure 9. Compressible steady inviscid hypersonic flow over an aerospace vehicle. Computed distribution of the Mach number contours

where the unknown vector $\mathbf{U}$ and the inviscid flux vectors $\mathbf{F}^j$ are as defined previously in equation (118), while the viscous flux vectors are defined as

$$\mathbf{G}^j = \frac{1}{Re}\begin{bmatrix} 0 \\ \sigma_{1j} \\ \sigma_{2j} \\ \sigma_{3j} \\ u_k\sigma_{kj} + \dfrac{\mu}{Pr}\dfrac{\partial T}{\partial x_j} \end{bmatrix} \tag{161}$$

Here, Re is the flow Reynolds number and Pr is the Prandtl number, while

$$T = \frac{\gamma p}{(\gamma - 1)\rho} \tag{162}$$

is the fluid temperature. The components, σ_{ij}, of the fluid stress tensor are defined as

$$\sigma_{ij} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2\mu}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{163}$$

when the Stokes hypothesis is invoked. The fluid viscosity, μ, may be assumed to vary with temperature according to the Sutherland relation

$$\mu = \sqrt{(\gamma-1)}M_\infty \left[\frac{1+S_0/T_\infty}{T+S}\right] T^{3/2} \tag{164}$$

where S_0 is a reference temperature, M_∞ is the Mach number of the free stream and

$$S = \frac{S_0}{(\gamma-1)M_\infty^2 T_\infty} \tag{165}$$

In these equations, T_∞ represents the free stream temperature.

The solution of this equation set is sought in a spatial domain Ω, bounded by a closed surface Γ. The solution is subject to an initial condition, in the form of equation (120), and to suitable boundary conditions. With the introduction of appropriate trial and weighting function spaces, the problem is expressed in the variational form: find $\mathbf{U}$ in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_\Omega W\left(\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3}\frac{\partial \mathbf{F}^j}{\partial x_j}\right) d\Omega = -\sum_{j=1}^{3}\int_\Omega \frac{\partial W}{\partial x_j}\mathbf{G}^j d\Omega + \sum_{j=1}^{3}\int_\Gamma W\nu_j \mathbf{G}^j d\Gamma \tag{166}$$

for every W in $\mathcal{W}$ and for all $t > 0$, where $\nu = (\nu_1, \nu_2, \nu_3)$ denotes the unit outward normal vector to Γ. With Ω represented by an unstructured grid of linear tetrahedral elements, with p nodes, an approximate solution is again sought in the finite dimensional subspace $\mathcal{T}^{(p)}$, defined in equation (122), of the trial function space. The Galerkin approximation satisfies the statement: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$, satisfying the problem boundary conditions, such that

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega = -\sum_{j=1}^{3}\int_\Omega N_I \frac{\partial \mathbf{F}^j}{\partial x_j} d\Omega - \sum_{j=1}^{3}\int_\Omega \frac{\partial N_I}{\partial x_j}\mathbf{G}^j d\Omega \tag{167}$$

for each interior node I in the grid and for all $t > 0$. Here $\mathbf{F}^j$ and $\mathbf{G}^j$ are to be regarded as functions of $\mathbf{U}^{(p)}$. Applying the divergence theorem, this Galerkin statement may be alternatively replaced by the requirement that

$$\int_\Omega N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega = \sum_{j=1}^{3}\int_\Omega \frac{\partial N_I}{\partial x_j}\left(\mathbf{F}^j - \mathbf{G}^j\right) d\Omega \tag{168}$$

at any interior node I.

5.1. Edge Based Implementation

The edge based methods described for the solution of the compressible Euler equations may be directly extended to produce solution algorithms for the laminar Navier Stokes equations [85]. To illustrate this, it can be observed that the approximate discrete form of equation (168) may be expressed, using the edge based data structure, as

$$\sum_{J=1}^{p} M_{IJ} \frac{d\mathbf{U}_J}{dt} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{II_e} - \sum_{e=1}^{\mu_I} \sum_{j=1}^{3} C^j_{II_e} \left(\mathbf{G}^j_I + \mathbf{G}^j_{I_e} \right) \tag{169}$$

where $\tilde{\mathcal{F}}$ denotes either one of the numerical flux functions, introduced for stabilization and shock capturing, in equations (131) or (135). Since the viscous fluxes are functions of both $\mathbf{U}$ and the gradient of $\mathbf{U}$, the first order spatial derivatives of the solution need to be determined at each node before this form can be used within a computational procedure. These derivatives can be determined as in equation (104).

This method of discretising the viscous fluxes leads to a centered approximation, but the corresponding stencil is not compact. In fact, when the method is applied on a uniform grid, of size h, in one dimension, the discretisation of a second spatial derivative is accomplished via a five point stencil, which is seen to be equivalent to a central difference approximation on a grid of size $2h$.

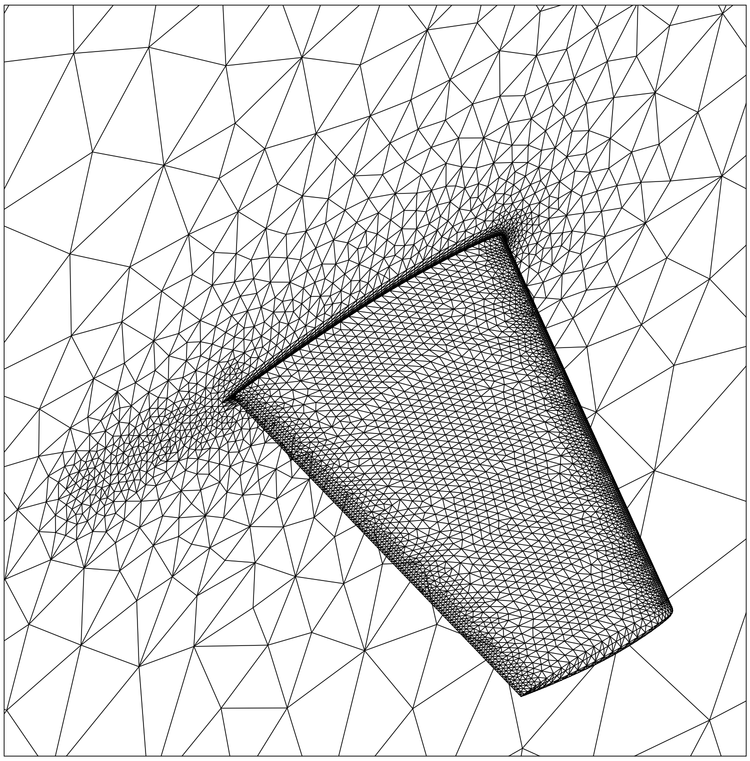

Figure 10. Compressible steady turbulent flow over an ONERA M6 wing. View of the surface discretisation

5.2. Element Based Implementation

The integral involving the viscous flux terms in the variational statement may also be approximately evaluated by employing an element based data structure [86]. In this case, the viscous fluxes are assumed to vary linearly, between their nodal values, over each element and the result is that

$$\sum_{J=1}^{p} M_{IJ} \frac{d\mathbf{U}_J}{dt} = \sum_{e=1}^{\mu_I} \tilde{\mathcal{F}}_{II_e} - \sum_{E\in I} \sum_{J\in E} \sum_{j=1}^{3} \left[\int_{\Omega_E} \frac{\partial N_I}{\partial x_j} N_J d\Omega \right] \mathbf{G}_J^j \tag{170}$$

where $\tilde{\mathcal{F}}$ denotes an appropriate numerical flux function. The advantage of this method is that it does lead to a compact discrete representation. This form is obviously well suited for use with the Taylor Galerkin method of equation (146) or with a genuine SUPG scheme [87], but it could also be used in conjunction with a method based upon an edge based evaluation of the inviscid fluxes. However, in this latter case, certain computational advantages of the edge based approach may be lost, as the corresponding computer code would need to work with both data structures simultaneously.

5.3. Turbulent Flows

With the adoption of an appropriate turbulence model [88], turbulent flows may be simulated by using straightforward extensions to the algorithms which have been outlined. One additional complication, which arises when algebraic models are implemented on unstructured grids, is the requirement for a method of determining the distance from any point in the grid to the nearest solid wall [89, 90]. One or two equation turbulence models, which involve additional differential equations, are readily implemented, using either edge based [91, 92, 93] or element based approaches [94].

Coupling such algorithms with the mesh generation approaches outlined in Section 3 provides a powerful capability for the analysis of turbulent aerodynamic flows over aircraft configurations. This is illustrated here by employing a two equation model to simulate the flow over an ONERA M6 wing at a free stream Mach number of 0.8447 and an angle of attack of 5.06 degrees [95]. The Reynolds number, based on the mean aerodynamic cord, is 11.6×10^6. The surface discretisation of the wing, consisting of 48 056 elements, is illustrated in Figure 10. Fourteen layers of elements are generated in the anisotropic region close to the wing surface. The complete mesh consists of 2 788 768 tetrahedral elements and 464 736 nodal points. The computed distribution of the pressure contours on the wing surface and on the symmetry plane is shown in Figure 11.

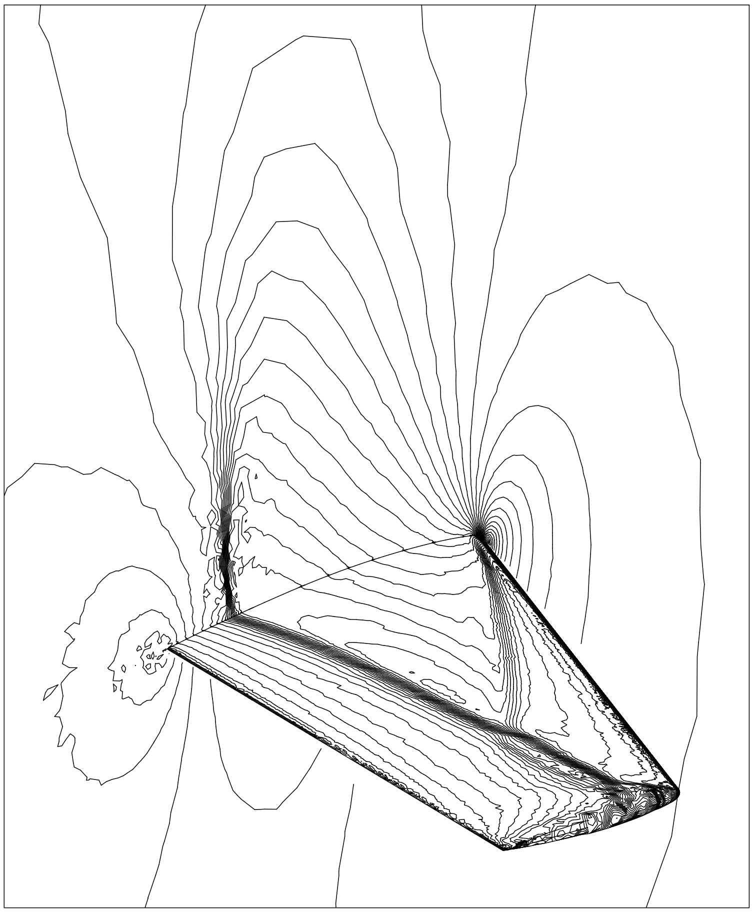

Figure 11. Compressible steady turbulent flow over an ONERA M6 wing. Computed distribution of the pressure contours

6. Adaptivity

The use of grid adaptivity is particularly attractive in computational fluid dynamics, as it offers the possibility of resolving localized flow features in the most efficient and cost–effective manner. Adaptive methods can be readily implemented within an unstructured grid environment and the early expectations of the impact of adaptivity in this area were considerable. In fact, many implementations have been developed and these have illustrated the benefits of the use of adaptive methods in particular applications. However, these implementations have been largely based on heuristic practices and the methods have not yet proved to be sufficiently reliable for general applicability in the area of computational aerodynamics. The two main ingredients for any successful adaptive method are an error estimation capability, which will dictate the required grid resolution, and a procedure which is capable of constructing a grid with the desired characteristics.

6.1. Error Estimation

To date, the process of error estimation has generally been accomplished by employing indicators which attempt to estimate the interpolation error. Consequently, these indicators ignore the nature of the underlying partial differential equation and are based only on the shape of the computed solution. In addition, for an equation system, such as the compressible Euler or Navier Stokes equations, estimation has been made on the basis of the behaviour of a key variable or key variables, selected in an ad hoc manner. Although such indicators have proved effective in enhancing solutions to particular problems, they can not be used with confidence to guarantee accurate solutions, when starting from arbitrary initial coarse grids.

Recently, more rigorous and sophisticated a–posteriori error estimation procedures have been developed for elliptic and parabolic problems [96]. These algorithms have proved to be very reliable and effective when used within an adaptive grid context. Methods based upon solution reconstruction have been proposed [97] and procedures based on the solution residual, often involving some stability constants for the exact problem, have also been developed [98]. In addition, techniques which require the solution of local problems have attracted attention [99, 100, 101, 102]. This latter class of techniques appears to be the most general and rigorous, as these methods do not require the estimation of any problem constants. These methods estimate the solution error in some mathematically defined norm, whose exact meaning is not always clear from an engineering standpoint. In addition, the application to any non–linear problem will generally involve some form of linearisation.

A more recent approach is to look at the solution error in terms of the engineering outputs, such as forces or heating rates, which are of interest for the problem under investigation. The methods which are being developed recognise that different accuracy requirements may be necessary to predict different quantities, e.g. in the computation of the flow over an aerofoil, the level of resolution required to predict the lift to within a certain accuracy is often much lower than that required to predict the drag to the same accuracy. Morover, the optimal grids which are required in these cases can also be significantly different. These approaches use adjoint solutions to quantify, or weigh, the effect of the local solution on the final output and employ the solution residual and some constants, which need to estimated from the continuous problems [103], or the solution of some local sub–problems [104]. As these methods are still in their early stages of development, it will be some time yet before they are applicable to problems governed by non–linear hyperbolic equation systems.

6.2. Grid Adaptation

There are a variety of different grid procedures designed to adapt a given grid so as to meet the requirements dictated by an error estimation process. Grid enrichment adds new nodes to prescribed regions and then reconnects them to produce a valid new grid [105]. Although this is an effective method of improving the resolution of a computed solution, the continued use of grid enrichment in three dimensional simulations can, very quickly, lead to very large grids [106]. Grid movement attempts to meet the requirements of the error indicator by moving the nodes on the grid into the regions where higher resolution is needed, without changing the grid topology [105]. The effectiveness of this method of increasing resolution is limited by the number of nodes present in the initial grid, especially for the solution of problems containing more than one significant feature [106]. Adaptive remeshing procedures aim to overcome the shortcomings of the enrichment and movement methods. The idea is to use the error indicator to define a new grid control function and then to generate a completely new grid for the problem [60, 61, 107, 108]. A further alternative is to provide a general procedure which combines all these methods in an appropriate manner and this approach has achieved a certain element of success [109, 110].

7. Unsteady Flows

The accurate simulation of time varying high speed compressible flows can require the use of significant computational resources. Although unsteady compressible viscous flows have been successfully modelled in two dimensions [111], evidence of the corresponding three dimensional extension is small [112, 113] and unstructured grid finite element developments in three dimensions have generally been restricted to inviscid simulations [114, 115]. Explicit timestepping techniques may be applied directly to the solution of unsteady flows but, for many practically interesting problems, an additional complication is the fact that the computational domain varies with time and this requires special treatment. Two–fluid or free surface simulations present additional interface tracking problems which will not be considered here [116].

Consider a problem involving three dimensional inviscid compressible flow in a spatial domain $\Omega(t)$ which varies with time. The solution of equation system (117) is sought, for all time $t > t_n$, subject to an initial condition which is expressed here as

$$\mathbf{U}(\mathbf{x}, t_n) = \mathbf{U}^n \qquad \text{for all } \mathbf{x} \text{ in } \Omega(t_n) \tag{171}$$

where $\mathbf{U}^n$ is a prescribed function. Appropriate boundary conditions also need to be provided to complete the problem definition.

7.1. *Variational Formulation*

Trial function and weighting function spaces are introduced, subject to the requirement that these spaces should consist of all suitably smooth functions, with the trial function space subjected to the additional requirement that

$$\mathcal{T} = \{\mathbf{U}(\mathbf{x},t) \mid \mathbf{U}(\mathbf{x},t_n) = \mathbf{U}^n \text{ for all } \mathbf{x} \text{ in } \Omega(t_n)\} \tag{172}$$

A variational formulation of the problem is then: find $\mathbf{U}$ in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_{t=t_n}^{T} \int_{\Omega(t)} \mathbf{W}^T \left(\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} \right) d\Omega dt = \mathbf{0} \tag{173}$$

for every member $\mathbf{W}(\mathbf{x},t)$ of the weighting function set $\mathcal{W}$ and for all values of $T > t_n$.

7.2. *Approximate Variational Formulation*

The domain $\Omega(t_n)$ is represented by an assembly of linear tetrahedral elements, with p nodes located at the element vertices. To accomodate the movement of the boundary, the location of the nodes in the grid must be allowed to vary with time. Finite dimensional subspaces $\mathcal{T}^{(p)}$ and $\mathcal{W}^{(p)}$, of the trial and weighting function spaces respectively, are defined by

$$\begin{aligned} \mathcal{T}^{(p)} &= \left\{ \mathbf{U}^{(p)}(\mathbf{x},t) \mid \mathbf{U}^{(p)} = \sum_{J=1}^{p} \mathbf{U}_J(t) N_J(\mathbf{x},t);\ \mathbf{U}_J(t_n) = \mathbf{U}_J^n \right\} \\ \mathcal{W}^{(p)} &= \left\{ \mathbf{W}(\mathbf{x},t) \mid \mathbf{W} = \sum_{J=1}^{p} \mathbf{a}_J N_J(\mathbf{x},t) \right\} \end{aligned} \tag{174}$$

Here N_J is the standard piecewise linear finite element shape function, which takes the value unity at node J, located at $\mathbf{x} = \mathbf{x}_J(t)$, and the value zero at every other node. A Galerkin approximate variational formulation of the problem is expressed in the form: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$, satisfying the problem boundary conditions, such that

$$\int_{t=t_n}^{T} \int_{\Omega(t)} N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega dt = -\sum_{j=1}^{3} \int_{t=t_n}^{T} \int_{\Omega(t)} N_I \frac{\partial \mathbf{F}^j}{\partial x_j} d\Omega dt \tag{175}$$

for $I = 1, 2, \cdots, p$. Following the application of the divergence theorem, the Galerkin statement corresponding to a typical interior node I of the grid is expressed alternatively as

$$\int_{t=t_n}^{T} \int_{\Omega(t)} N_I \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega dt = \sum_{j=1}^{3} \int_{t=t_n}^{T} \int_{\Omega(t)} \frac{\partial N_I}{\partial x_j} \mathbf{F}^j d\Omega dt \tag{176}$$

Further manipulation of this statement is essential, to convert it into a form which is useful for the solution of practical problems, in which portions of the boundary surface might be moving. To achieve this, suppose that the grid employed is moving with a prescribed velocity

$$\mathbf{V}^{(p)} = \left(v_1^{(p)}, v_2^{(p)}, v_3^{(p)}\right) \tag{177}$$

where the variation of each velocity component is approximated, over the grid, in the form

$$v_j^{(p)} = \sum_{J=1}^{p} v_{jJ} N_J \tag{178}$$

Here, v_{jJ} denotes the component of the velocity of node J in the direction x_j. It follows that

$$\frac{d}{dt}\int_{\Omega(t)} N_I \mathbf{U}^{(p)} d\Omega = \int_{\Omega(t)} \frac{\partial}{\partial t}\left(N_I \mathbf{U}^{(p)}\right) d\Omega + \int_{\Omega(t)} \frac{\partial}{\partial x_j}\left(v_j^{(p)} N_I \mathbf{U}^{(p)}\right) d\Omega \tag{179}$$

where the total time derivative implies differentiation following the moving grid. When this expression is used in equation (176), the approximate variational statement is expressed in the alternative form: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$ such that

$$\left[\int_{\Omega(t)} N_I \mathbf{U}^{(p)} d\Omega\right]_{t=t_n}^{t=T} = \int_{t=t_n}^{T}\int_{\Omega(t)} \frac{\partial N_I}{\partial x_j}\mathbf{F}^{j*} d\Omega dt \tag{180}$$

where

$$\mathbf{F}^{j*} = \mathbf{F}^{j}(\mathbf{U}^{(p)}) - v_j^{(p)}\mathbf{U}^{(p)} \tag{181}$$

The form assumed for $\mathbf{U}^{(p)}$ in equation (174) is substituted into equation (180) and the left hand side is evaluated exactly. As before, a numerical quadrature rule is employed to approximate the right hand side integral over the spatial domain. The result is that

$$\left[\sum_{J=1}^{p} M_{IJ}\mathbf{U}_J\right]_{t=t_n}^{t=T} = \int_{t=t_n}^{T} \mathbf{f}_I dt \tag{182}$$

where, for example, in terms of an edge based data structure,

$$\mathbf{f}_I = \sum_{e=1}^{\mu_I}\sum_{j=1}^{3} C_{II_e}^{j}\left(\mathbf{F}_I^{j*} + \mathbf{F}_{I_e}^{j*}\right) \tag{183}$$

Stabilisation of the algorithm of equation (182) is achieved by replacing the actual flux function by a numerical flux function. For example, when the flux function of equation (131) or (135) is adopted, the solution can be advanced in time by the $k-$stage timestepping procedure of equation (35). The geometric conservation law is a statement which ensures that the resulting procedure preserves a constant solution for an arbitrary grid deformation. When the solution is advanced over a timestep Δt from time t_n to time t_{n+1}, it can be shown [117] that this law is satisfied if the nodal velocities in equation (178) are computed as

$$v_{jJ} = \frac{x_{jJ}^{n+1} - x_{jJ}^{n}}{\Delta t} \tag{184}$$

and if the right hand side of equation (182) is evaluated with appropriate accuracy. For a three dimensional simulation, the accuracy requirement is that the integration rule being used integrates a quadratic function exactly

7.3. Grid Modification in the Presence of Moving Boundaries

When the simulation involves a moving boundary, the co–ordinates of the boundary nodes must be updated at each time step, according to the prescribed boundary displacement. It is apparent that the initial grid will, eventually, become excessively distorted and invalid, unless some method of modifying the grid is added to the solution algorithm.

7.3.1. A Deforming Grid Algorithm A deforming grid algorithm is readily coupled to the solution process [118, 119]. With the co–ordinates of nodes given at time $t = t_n$, the new co–ordinates, at time $t = t_{n+1}$ can be obtained by the addition of a displacement. Nodal points located on the far field boundary will be held fixed, with a zero nodal displacement specified. At nodal points which are located on the moving boundary component, the updated nodal displacement is prescribed by the motion. The new displacement at any interior node is found by averaging the displacements of the surrounding points and iterating to convergence. Sufficient convergence is normally obtained after 5 iterations. Alternative procedures involve the direct computation of the mesh velocities at each node in the grid as the product of the boundary velocity and a scalar function, based upon the distance from the node to the moving boundary [120].

These algorithms will change the location of nodes in the grid, but will not change the grid topology. Such approaches will, therefore, only prove to be effective for problems in which the boundary displacement is small. For large scale boundary displacements, elements in the deforming grid will eventually become distorted. This means that an additional capability which enables local regeneration of the grid is also required.

7.3.2. A Local Regridding Algorithm The removal of all the elements which are regarded as being distorted, according to some measure, will, in general, create a number of separate holes in the grid. Each of these holes will be bounded by either a collection of triangular faces or by a collection of triangular faces plus a portion of the computational boundary. Local regeneration of the grid follows the steps involved in the generation of the initial grid, with any boundary surfaces being triangulated first. Following this step, each hole in the grid will now be completely bounded by a collection of triangular faces. The holes are discretised using any standard grid generation approach, with automatic point creation. Following the construction of the grid for each hole, the value of the solution at the newly created nodal points is obtained by interpolation over the original

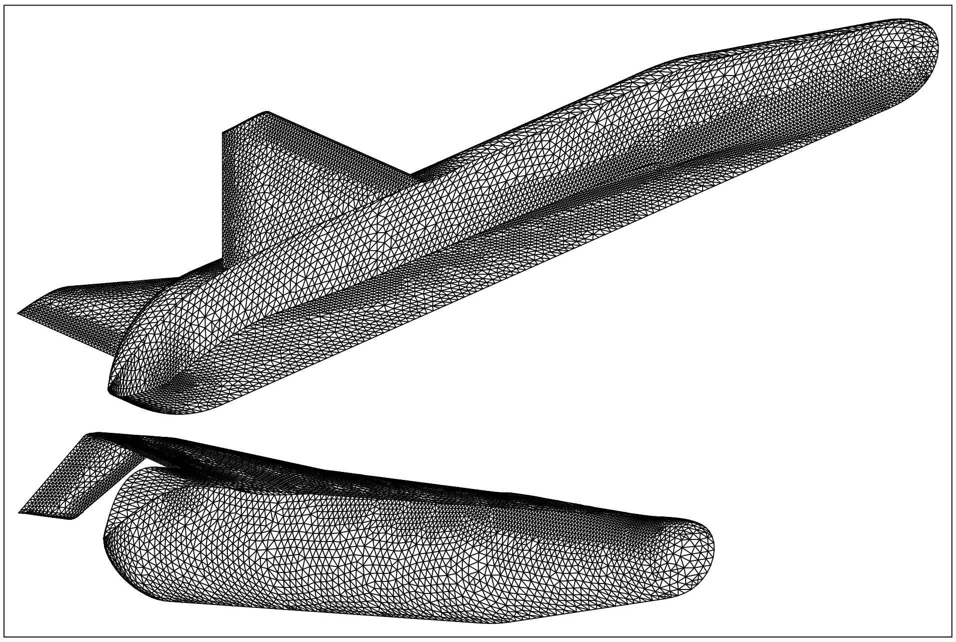

Figure 12. Final stage of the shuttle vehicle and booster simulation showing a view of the surface discretisation

grid. This can be accomplished using a searching process based upon the alternating digital tree [62]. In a practical implementation, the quality of the elements in the grid should be checked every time step. A quality test which can be adopted is that of comparing the current volume and the initial volume of each element. According to this test, when more than a certain percentage of the elements in the grid are found to have a volume change which is greater than a prescribed amount, these elements are removed and the grid is locally regenerated.

To illustrate the numerical performance of a procedure of this type, a problem involving the transient separation of a generic shuttle vehicle and a rocket booster has been simulated. Initially, a steady state solution for the configuration is obtained at a free stream Mach number of 0.9 and zero degrees angle of attack, on a grid consisting of 684 429 elements, 125 899 nodes and 810 570 edges. The two bodies now separate according to a prescribed translation, in the symmetry plane, together with a prescribed rotation of the shuttle about a point on its tail and the simulation proceeds until the shuttle rotates through 30 degrees. This amount of relative movement between the two bodies requires only 45 local regridding steps and the grid at this stage consists of 556 194 elements, 103 930 nodes and 685 072 edges. A view of the surface triangulation of the bodies at the end of the computation is given in Figure 12. The corresponding computed distribution of the pressure is displayed in Figure 13.

8. Improving Computational Performance

Although explicit timestepping procedures are readily implemented, they suffer from certain well–known computational drawbacks. When realistic steady problems are being simulated, explicit schemes frequently exhibit slow convergence rates while, for unsteady problems, explicit schemes may be subject to a stability limit on the timestep which is several orders of magnitude smaller than the limit required to produce accurate solutions. A variety of different approaches have been suggested to address these shortcomings and some of these are briefly considered in this section.

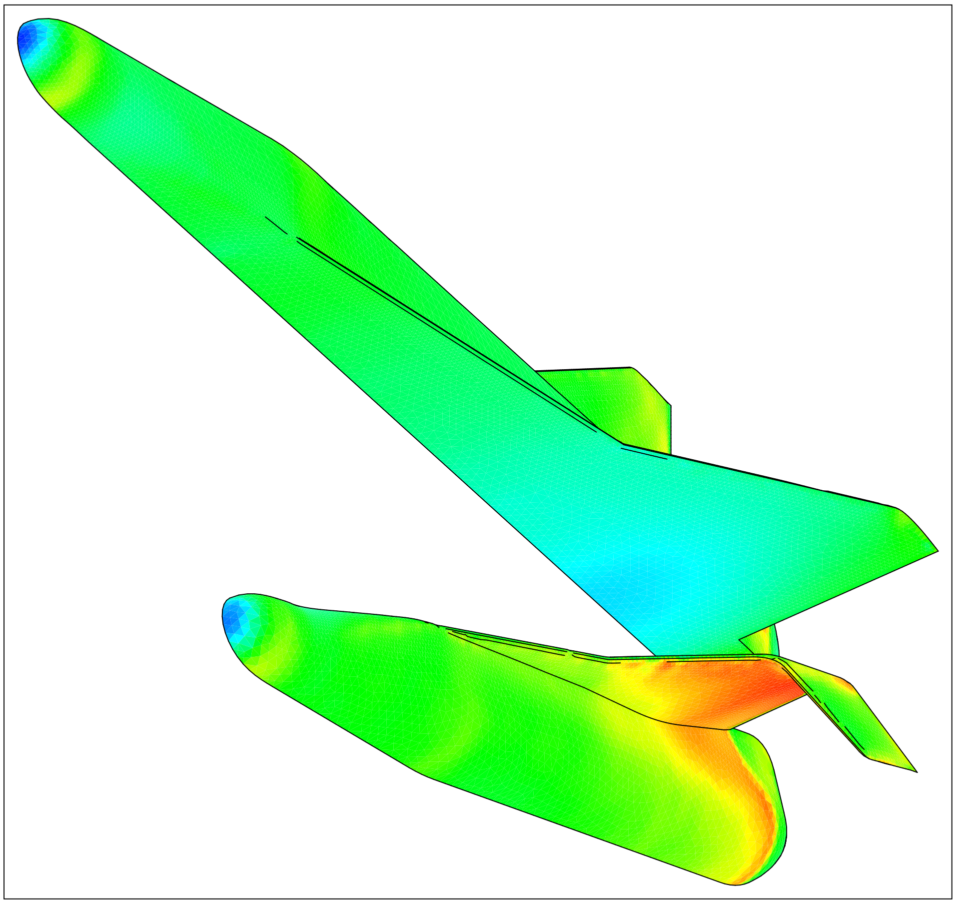

Figure 13. Shuttle vehicle and booster simulation showing the distribution of the pressure at the final stage of the computation

8.1. Domain Decomposition

Domain decomposition techniques may be incorporated to improve the efficiency of explicit methods used for the simulation of transient flows on grids exhibiting large variations in the element size. Methods have been proposed in which the decomposition is made into either groups of elements [121] or into groups of nodal points [111]. For illustration, consider an element decomposition, where the objective is to group the elements, according to their theoretical maximum allowable timestep size, and to advance the solution independently within each group. The time accuracy of the procedure is maintained by appropriate interchange of information across the boundaries of the groups. Algorithmically, the approach determines initially the minimum timestep, Δt_{min} allowed on the grid. An element E is then placed into group m if its allowable timestep, Δt_E, is such that

$$2^{m-1}\Delta t_{min} \leq \Delta t_E < 2^m \Delta t_{min} \tag{185}$$

The groups are overlapped, by adding two layers of elements to each group m from the groups q, where $q > m$. The solution is advanced through one global timestep $\Delta t = 2^{m_{max}-1}\Delta t_{min}$, where m_{max} is the maximum number of groups employed, by sequencing through the groups, and using the overlapping to provide the updated group interface values. The optimum value to be adopted for m_{max} may be determined by analysing the allowable timestep distribution on the grid [122].

8.2. Parallelisation

The introduction of parallel computers offers the promise of providing the analyst with access to large computational power [123]. Explicit timestepping schemes are readily implemented within a distributed computational environment, so that this provides an obvious method of reducing the required run times. As an example, a parallel implementation of unstructured grid algorithms may be achieved by employing a single program multiple data model [124] and using standard message passing routines. At present, grids are generally generated and decomposed in a serial manner, which means that computer memory requirements place restrictions on the size of the global grid which can be employed [125]. Such restrictions could be avoided by performing the grid generation in parallel, though this is a topic which has, only recently, been receiving attention [126, 127, 128].

A number of different methods have been proposed for partitioning unstructured grids [129]. Multilevel RSB is a method which produces the well–balanced subdomains, with minimal communication costs, which are required for the optimisation of parallel performance [130]. When the technique is used to colour the nodes, the decomposition splits the global grid into $\mathcal{N}$ subdomains, producing $\mathcal{N}$ data sets, with local numbering of

Table 1. Performance Statistics—Edge based Euler solver on a CRAY T3D

No. of Partitions	Edges		Points		Run Time (min) (1000 steps)	Relative Speed Up
	Min	Max	Min	Max		
4	469 771	484 465	66 523	70 130	650	1
8	230 459	244 827	33 262	37 357	346	1.88
16	114 139	125 040	16 631	19 674	180	4.09
32	55 144	64 926	8 316	10 609	96	7.07
64	26 760	33 475	4 158	5 930	53	17.77
128	12 826	16 980	2 079	3 239	29	24.07
256	6 189	8 762	1 040	1 822	16	46.43

points, elements, edges and boundary faces. If an edge based representation of the grid is employed, the edges within each subdomain are subdivided into two groups. One group consists of edges attached to interface nodes and, as such, will require communication. The second group contains edges connecting interior nodes only. The interface edges and the connecting nodes are stored in the subdomain which has the minimum partition number. This results in duplicated interface nodes in neighbouring subdomains, but interface edges are not duplicated. The communication arrays, which are necessary for the transfer of information between the subdomains, may be evaluated during the domain partitioning stage. At the first stage of the computation loop, interface nodes obtain contributions from interface edges. These partially updated contributions are then broadcast to the neighbouring subdomains. A loop over the interior edges of each partition, followed by the receiving and updating of the interface nodes completes the procedure. This procedure allows computation and communication to take place concurrently [131]. For a grid consisting of 1 612 174 elements, 266 092 nodes and 1 912 170 edges, Table 1 illustrates the load balancing capabilities of the RSB method and the resulting performance of an edge based parallel Euler flow solver on a CRAY T3D.

In this context, the introduction of special parallel libraries to aid the process of parallelisation of unstructured mesh based solvers should also be noted [132, 133].

8.3. Multigrid Methods

The multigrid method is an effective acceleration device, when it is coupled with an explicit multistage timestepping scheme [134]. The basic idea of the approach is to use solution residuals, computed on a given grid, to drive a time marching scheme on a coarser grid. The corrections to the unknowns computed on the coarser grid are then added to the fine grid solution. Advancing the solution on the coarse grid is computationally less expensive, because of the reduced number of unknowns and the larger allowable timestep size. The time marching scheme on the coarse grid may itself be accelerated by the use of an even coarser grid and, in this way, the multigrid concept can be extended to incorporate any number of coarse grids.

The inter–grid transfer operations, which this process requires, are readily performed in a structured grid environment, where the grids are generally nested. However, the situation is more complex when the grid upon which the solution is required is totally unstructured and, in this case, different approaches have been attempted [135]. In one approach, the coarse meshes are constructed by agglomerating cells on the fine mesh [136, 137, 138, 139]. With this method, the coarse mesh equations are formed directly by combining, in an appropriate manner, the contributions from the relevant fine mesh cells. A second approach employs a sequence of non–nested coarse unstructured grids [140, 141]. Here the grids employed are totally unrelated and, in general, will not have coincident nodes. In this case, the grid sequence is constructed by repeatedly using the grid generator, with appropriate redefinition of the grid control functions. Procedures for rapidly transferring information between the grids are an essential feature of a successful implementation.

8.3.1. Transferring Information Between Non–Nested Grids To illustrate how information can be transferred between two non–nested tetrahedral grids [142], let

$$\mathbf{U}^{(p_f)} = \sum_{J_f=1}^{p_f} \mathbf{U}_{J_f} N_{J_f} \qquad\qquad \mathbf{U}^{(p_c)} = \sum_{J_c=1}^{p_c} \mathbf{U}_{J_c} N_{J_c} \tag{186}$$

denote piecewise linear approximations to the solution on two grids f and c, having p_f and p_c nodes respectively. Suppose that the representation on grid f is known and that the representation on grid c is required. Starting from the Galerkin projection statement [17]

$$\int_\Omega \mathbf{U}^{(p_c)} N_{I_c} d\Omega = \int_\Omega \mathbf{U}^{(p_f)} N_{I_c} d\Omega \tag{187}$$

for $I_c = 1, 2, \cdots, p_c$, and inserting the assumed forms for $\mathbf{U}^{(p_f)}$ and $\mathbf{U}^{(p_c)}$ from equation (186), results in the requirement that

$$\sum_{J_c=1}^{p_c} \left[\int_\Omega N_{J_c} N_{I_c} d\Omega \right] \mathbf{U}_{J_c} = \sum_{J_f=1}^{p_f} \left[\int_\Omega N_{J_f} N_{I_c} d\Omega \right] \mathbf{U}_{J_f} \tag{188}$$

for $I_c = 1, 2, \cdots, p_c$. This is a system of p_c equations for the nodal values of the unknowns on grid c. In the present context, a projection procedure of this type will only be exact when the solution varies linearly. The sum of all the shape functions on a given grid is equal to unity everywhere in the domain, so that adding the equations in the set (187) produces

$$\int_\Omega \mathbf{U}^{(p_c)} d\Omega = \int_\Omega \mathbf{U}^{(p_f)} d\Omega \tag{189}$$

This result demonstrates that, in this form, the projection leads to a conservative transfer of the unknown between the two grids.

However, the exact evaluation of the integral on the right hand side of equation (188) proves difficult and expensive, as the functions N_{I_f} and N_{I_c} are defined on different supports. For a practical implementation, this integral may be approximated numerically and two different methods of achieving this are immediately apparent.

The first option is to express the integral, as the sum of individual element contributions from grid c, and to use a composite quadrature rule, with the integral over each element approximated by sampling the integrand at the nodes. If the left hand side matrix is also lumped, the result is that

$$\mathbf{U}_{I_c} = \sum_{J_f=1}^{p_f} N_{J_f}(\mathbf{x}_{I_c}) \mathbf{U}_{J_f} \qquad I_c = 1, 2, \cdots, p_c \tag{190}$$

In this form, the projection is seen to lead to a direct linear interpolation. This can be implemented when the four nodes of the element within grid f which contains a given node of grid c is known. This form also preserves the accuracy of the original projection, i.e. it is exact for linear data. In general, however, the interpolated values on grid c will not satisfy exactly the conservation property expressed by equation (189).

The second option is to evaluate the right hand side integral of equation (188) by summing element contributions from grid f. In this case, use of the same composite

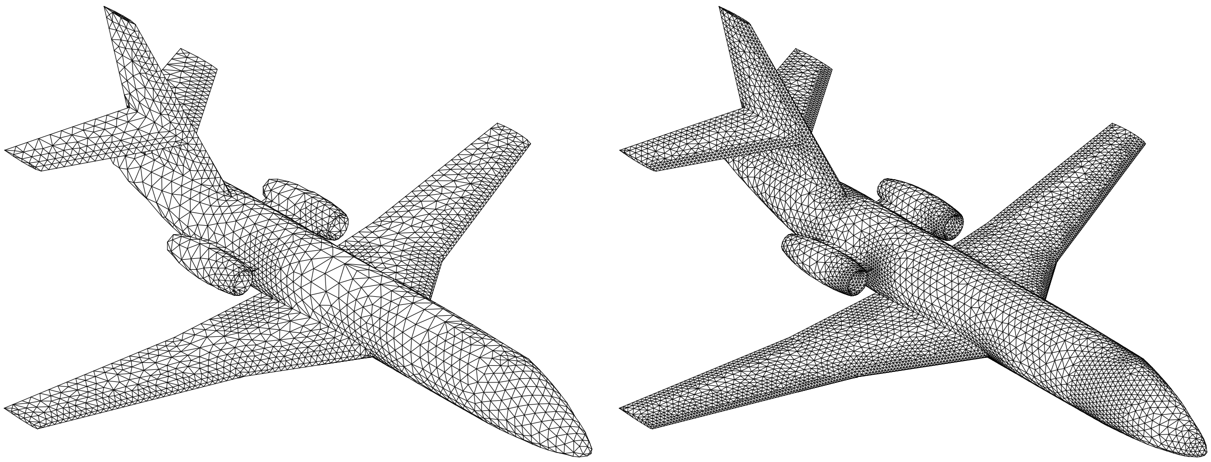

Figure 14. Compressible steady inviscid transonic flow over a business jet configuration. View of the coarser surface grids employed in the multigrid process

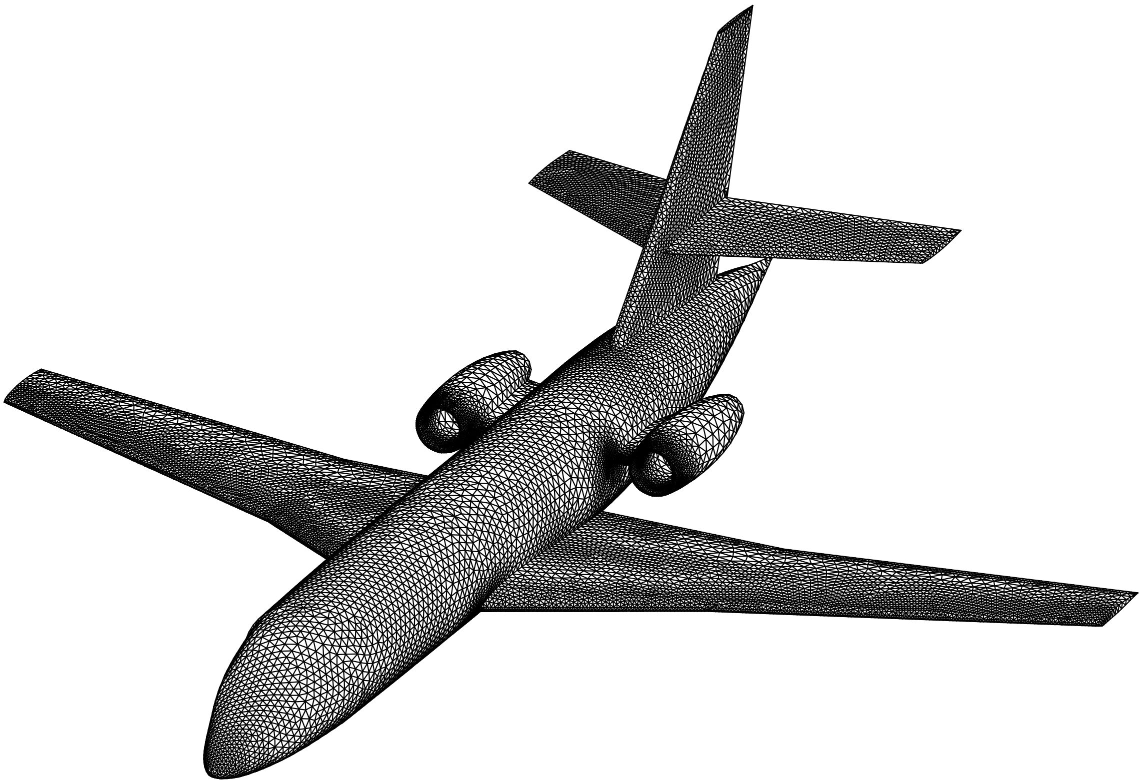

Figure 15. Compressible steady inviscid transonic flow over a business jet configuration. View of the finest surface grid employed in the multigrid procedure

quadrature rule and the lumping of the left hand side matrix produces

$$\mathbf{U}_{I_c} = M_{I_c}^{-1} \sum_{J_f=1}^{p_f} M_{J_f} N_{I_c}(\mathbf{x}_{J_f}) \mathbf{U}_{J_f} \tag{191}$$

This operation does not preserve a linear solution, but, by summing all the equations of the form of (191), a discrete analogue of equation (189) is obtained, so that this transfer process is conservative.

8.3.2. Implementation of the Multigrid Method The implementation of a multigrid solution algorithm is best described for a case involving just two grids, a fine grid f and a coarser grid c. The solution obtained after performing i_f iterations on grid f of the multi–stage timestepping scheme of equation (35), and its corresponding residual, computed using an appropriate numerical flux function, are transferred to grid c. At this stage, it is assumed that the residuals are sufficiently smooth so that they may be meaningfully represented on the coarser grid c. The solution on grid c is now advanced for i_c iterations, again using the time marching scheme of equation (35), but with a

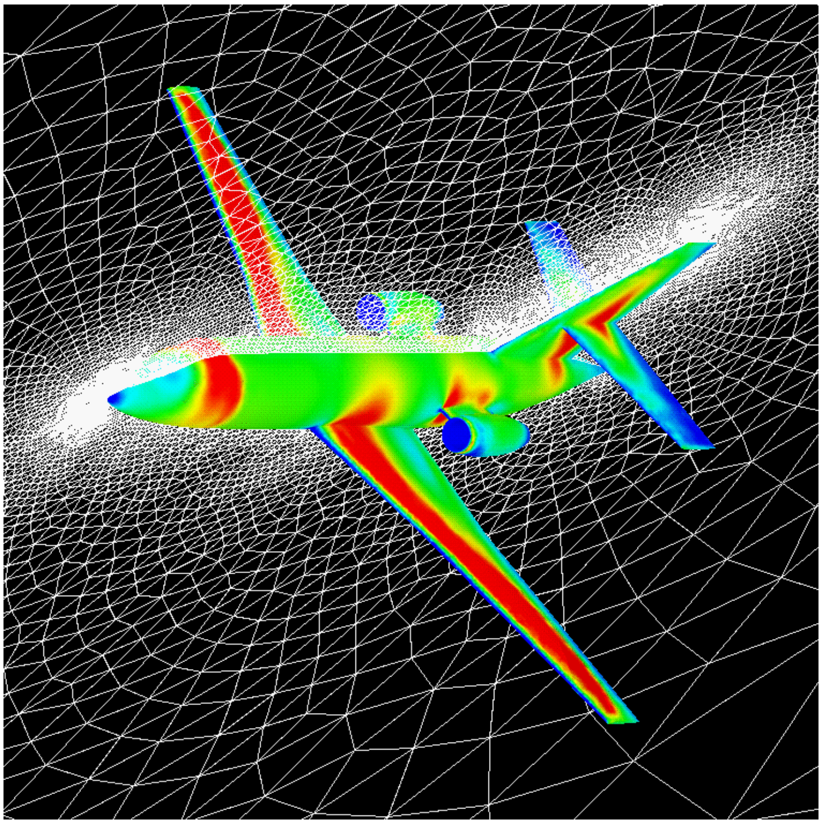

Figure 16. Compressible steady inviscid transonic flow over a business jet configuration. Computed distribution of the pressure coefficient on the finest grid

modified right hand side. This right hand side modification is essential if a converged solution on the fine grid is to remain unaltered by the multigrid process. The solution correction calculated on grid c is then transferred back to the fine grid f and added to the fine grid solution.

The solution computed on the coarse grid can intself be accelerated by using an even coarser grid. In this way, the above algorithm may be extended to more than two grids. At the same time, different methods of cycling between the grids may be devised. When more than two grids are employed, there exists the posssibility of advancing the solution for i_i timesteps on an intermediate grid, after transfer from a coarser grid and before transfer to the next finer grid. To illustrate the performance of the multigrid method, a procedure of this type has been applied to the simulation of a steady inviscid transonic flow over a business jet configuration. In this application, the unknowns and the corrections are transferred between grids using the direct interpolation of equation (190), while the residuals are transferred using the conservative form of equation (191). A sequence of three grids is employed. Figure 14 gives view of the surface discretisation for the coarse and intermediate grids, while Figure 15 displays a view of the surface discretisation for the finest grid. The computed distribution of the pressure coefficient contours on the finest grid is displayed in Figure 16. The effectiveness of the multigrid procedure is apparent from Figure 17, which shows a comparison of the convergence rates attained by the single grid algorithm, on the fine grid, and the multigrid algorithm, using the full sequence of grids.

8.4. Implicit Timestepping Methods

Implicit timestepping methods allow for the use of larger timesteps and, generally, result in reduced solution times. The drawback of implicit techniques is the requirement to solve a large sparse equation system at each time step. Direct methods of solving such equation systems are currently limited to two dimensional simulations, because of the computational costs and the memory requirements of the approach [143]. For this reason, alternative methods of approaching the problem of solving this equation system

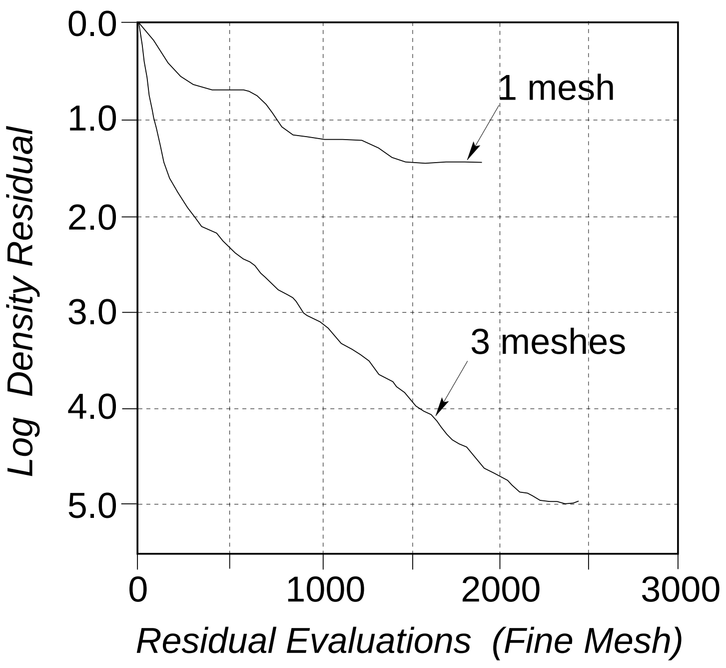

Figure 17. Compressible steady inviscid transonic flow over a business jet configuration. Comparison of the convergence rates of the single grid and the multigrid algorithms

have received attention.

8.4.1. Multigrid Methods The multigrid method may be extended and applied to the problem of solving an implicit equation system [144, 145]. To illustrate how this can be achieved, suppose that the equation corresponding to a typical node I of the grid, in the simulation of a transient flow in a fixed computational domain, can be written in the form

$$\sum_{J=1}^{p} M_{IJ} \frac{\mathrm{d}\mathbf{U}_J}{\mathrm{d}t} = \mathbf{f}_I \tag{192}$$

Using the three level implicit approximation of equation (48), leads to the timestepping scheme

$$\sum_{J=1}^{p} M_{IJ} \frac{\left(3\mathbf{U}_J^{n+1} - 4\mathbf{U}_J^{n} + \mathbf{U}_J^{n-1}\right)}{2\Delta t} = \mathbf{f}_I^{n+1} \tag{193}$$

which can be re–written in the form

$$\mathbf{R}_I^{n+1} + \mathbf{s}_I = \mathbf{0} \tag{194}$$

where

$$\mathbf{R}_I^{n+1} = \mathbf{f}_I^{n+1} - \frac{3}{2\Delta t} \sum_{J=1}^{p} M_{IJ} \mathbf{U}_J^{n+1} \tag{195}$$

and

$$\mathbf{s}_I = \frac{2\mathbf{U}_I^n}{\Delta t} - \frac{\mathbf{U}_I^{n-1}}{2\Delta t} \tag{196}$$

Equation (194) may now be solved by using an explicit multistage scheme to obtain the steady state solution of the equation

$$M_I \frac{\mathrm{d}\mathbf{U}_I^{n+1}}{\mathrm{d}t} = \mathbf{R}_I^{n+1} + \mathbf{s}_I \tag{197}$$

The convergence of the resulting procedure may be accelerated by using the multigrid concepts introduced in Section 8.3 [146, 57, 147]. It has been observed [148] that an explicit discretization of equation (197) may be unstable when the chosen timestep is larger than that employed in equation (193). This is easily remedied by either clipping the timestep used or simply treating the diagonal part of the second term on the right hand side of equation (195) implicitly when solving equation (197).

8.4.2. Standard Iterative Methods An alternative approach is to employ standard iterative methods to solve the matrix equation system arising from an implicit time discretisation. Suppose that a steady inviscid flow is being simulated, using an edge based LED algorithm of the type introduced in Section 4.3. The equation corresponding to a typical interior node I of the grid, may then be written as

$$M_I \frac{d\mathbf{U}_I}{dt} = \mathbf{f}_I \tag{198}$$

where, for convenience, $\mathbf{f}_I$ is now expressed as

$$\mathbf{f}_I = \sum_{e=1}^{\mu_I} [\mathbf{A}_{II_e}(\mathbf{U}_{I_e} - \mathbf{U}_I) + \mathbf{d}_{II_e}] \tag{199}$$

In equation (199), the added diffusion term is taken in the higher order form

$$\mathbf{d}_{II_e} = \mathbf{X}_{II_e}^{-1} \left|\mathbf{A}_{II_e}\right| \left[\Delta\mathbf{S} - \frac{1}{2}\left[\mathcal{L}(\Delta\mathbf{S}_{II_e}^{+}, \Delta\mathbf{S}_{II_e}) + \mathcal{L}(\Delta\mathbf{S}_{II_e}, \Delta\mathbf{S}_{II_e}^{-})\right]\right] \tag{200}$$

in the notation of equations (115) and (159). Equation (198) is discretised in time, using the first order method of equation (47), producing

$$M_I \Delta\mathbf{U}_I = \Delta t \mathbf{f}_I^{n+1} \tag{201}$$

where $\Delta\mathbf{U}_I = \mathbf{U}_I^{n+1} - \mathbf{U}_I^n$. A diagonally dominant linearised version of this equation is obtained, by employing a low order form on the left hand side, as

$$M_I \Delta\mathbf{U}_I + \Delta t \sum_{e=1}^{\mu_I} \tilde{\mathbf{A}}_{II_e}^n (\Delta\mathbf{U}_{I_e} - \Delta\mathbf{U}_I) = -\Delta t \mathbf{f}_I^n \tag{202}$$

where

$$\tilde{\mathbf{A}}_{II_e} = \mathbf{A}_{II_e} - \left|\mathbf{A}_{II_e}\right| \tag{203}$$

The fact that the left and right hand side operators are now of different order generally results in a conditionally stable procedure. When the complete equation set (202) is expressed in the matrix form

$$\mathbf{K}\Delta\mathbf{U} = \mathbf{f} \tag{204}$$

the matrix $\mathbf{K}$ may be decomposed as

$$\mathbf{K} = \mathbf{D} + \mathbf{E} \tag{205}$$

where $\mathbf{D}$ denotes the matrix which consists of only the block diagonal entries of $\mathbf{K}$. Equation (204) can then be solved by the block Jacobi iteration process [149, 91, 150]

$$\Delta\mathbf{U}^{(s+1)} = \mathbf{D}^{-1}\left(\mathbf{f} - \mathbf{E}\Delta\mathbf{U}^{(s)}\right) \qquad s = 1, 2, \cdots, s_{max} \tag{206}$$

where s_{max} denotes a prescribed number of iterations.

Line relaxation schemes have proved to be particularly successful when used in conjuction with numerical discretisations on structured grids [151]. On structured grids, the grid lines themselves serve as appropriate lines for the relaxation process.

On unstructured grids, a more elaborate procedure needs to be employed to identify suitable lines [152, 107]. An example of a line which may be constructed for use in a three dimensional simulation on an unstructured tetrahedral grid is shown in Figure 18. When the lines have been produced, the nodal equations forming the system (204) are renumbered according to the order implied by the line and the new system is written in the form

$$\mathbf{K}^{*}\Delta\mathbf{U}^{*} = \mathbf{f}^{*} \tag{207}$$

The matrix $\mathbf{K}^{*}$ is decomposed as

$$\mathbf{K}^{*} = \mathbf{D}^{*} + \mathbf{E}^{*} \tag{208}$$

where $\mathbf{D}^{*}$ is the block tridiagonal of $\mathbf{K}^{*}$, and this allows the use of the line relaxation procedure

$$\mathbf{D}^{*}\Delta\mathbf{U}^{*(s+1)} = \left(\mathbf{f}^{*} - \mathbf{E}^{*}\Delta\mathbf{U}^{*(s)}\right) \qquad s = 1, 2, \cdots, s_{max} \tag{209}$$

where s_{max} denotes the number of line iterations employed. For three dimensional simulations, a possible strategy is to use three different lines, with a fixed number of iterations being employed on each line within every timestep. As these lines frequently

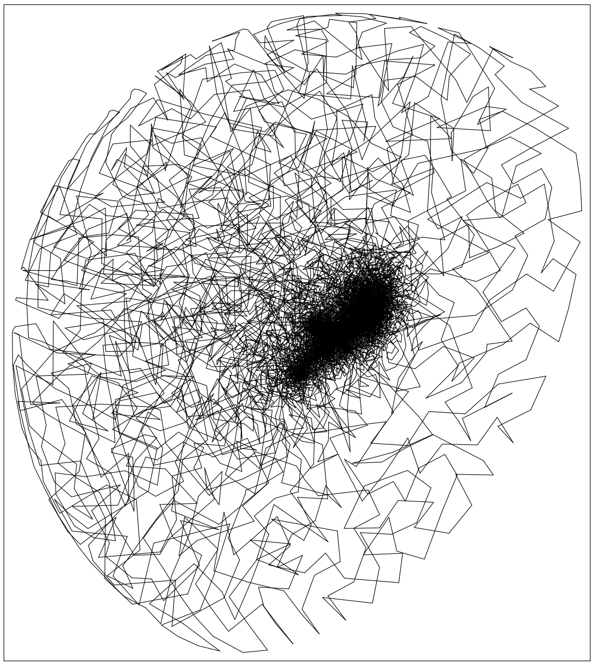

Figure 18. Continuous line constructed on a grid about a complete aircraft configuration.

exhibit significant folding, an alternative here is to break the lines into multiple linelets, which are roughly aligned with prescribed directions [153].

Relaxation methods based upon LU factorisation schemes have also been implemented on structured grids [154]. The manner in which this can be extended is outlined here for unstructured triangular meshes in 2D [155]. The corresponding 3D extension is direct. Consider a sweep direction, defined by a unit vector (s_{x_1}, s_{x_2}). The nodes in the grid are renumbered according to their s coordinate, with the coordinate for node I computed as

$$s_I = x_{1(I)} s_{x_1} + x_{2(I)} s_{x_2} \tag{210}$$

The edges e which meet at node I are subdivided into two sets, e^+ and e^-, with

$$II_e \in e^+ \quad \text{if} \quad (x_{1(I_e)} - x_{1(I)}) s_{x_1} + (x_{2(I_e)} - x_{2(I)}) s_{x_2} \geq 0 \tag{211}$$

$$II_e \in e^- \quad \text{if} \quad (x_{1(I_e)} - x_{1(I)}) s_{x_1} + (x_{2(I_e)} - x_{2(I)}) s_{x_2} < 0 \tag{212}$$

With this subdivision, equation (202) is rewritten as

$$\left(\mathbf{I} + \frac{\Delta t}{M_I} \sum_{II_e \in e^+} \tilde{\mathbf{A}}^n_{II_e} \mathcal{D}_{II_e} + \frac{\Delta t}{M_I} \sum_{II_e \in e^-} \tilde{\mathbf{A}}^n_{II_e} \mathcal{D}_{II_e} \right) \Delta \mathbf{U}_I = -\frac{\Delta t}{M_I} \mathbf{f}^n_I \tag{213}$$

where $\mathbf{I}$ is the identity matrix and

$$\mathcal{D}_{II_e} \mathbf{U}_I = \mathbf{U}_{I_e} - \mathbf{U}_I \tag{214}$$

An approximate LU factorisation for equation (213) is expressed as

$$\left(\mathbf{I} + \frac{\Delta t}{M_I} \sum_{II_e \in e^+} \tilde{\mathbf{A}}^n_{II_e} \mathcal{D}_{II_e} \right) \left(\mathbf{I} + \frac{\Delta t}{M_I} \sum_{II_e \in e^-} \tilde{\mathbf{A}}^n_{II_e} \mathcal{D}_{II_e} \right) \Delta \mathbf{U}_I = -\frac{\Delta t}{M_I} \mathbf{f}^n_I \tag{215}$$

It is readily verified that, when this equation is applied to every node, the first factor on the left hand side leads to a block lower triangular matrix, while the second factor leads to a block upper triangular matrix. This means that equation (215) results in an equation system of the form

$$\left(\Theta + \Pi^+ \right) \left(\Upsilon + \Pi^- \right) \Delta \mathbf{U} = \mathbf{f} \tag{216}$$

where Θ and Υ are strictly block lower and block upper triangular matrices respectively, the matrices Π^+ and Π^- contain block diagonal terms only, $\Delta \mathbf{U}$ is the global vector containing the increments of the nodal unknowns and $\mathbf{f}$ is the global vector of the right hand side terms of equation (215). This equation can then be solved in two stages as

$$\left(\Theta + \Pi^+ \right) \Delta \mathbf{U}^* = \mathbf{f} \tag{217}$$

and

$$\left(\Upsilon + \Pi^- \right) \Delta \mathbf{U} = \Delta \mathbf{U}^* \tag{218}$$

with each stage requiring the inversion of a block 4×4 matrix at each grid point.

A further alternative is to solve equation (202) by employing forward and backward Gauss–Seidel relaxation sweeps [156]. During the forward sweep, a nodal increment $\Delta\mathbf{U}_I^*$ is calculated from

$$\Delta\mathbf{U}_I^* + \frac{\Delta t}{M_I}\sum_{II_e\in e+}\tilde{\mathbf{A}}_{II_e}^n(\Delta\mathbf{U}_{I_e}^* - \Delta\mathbf{U}_I^*) - \frac{\Delta t}{M_I}\sum_{II_e\in e-}\tilde{\mathbf{A}}_{II_e}^n\Delta\mathbf{U}_I^* = \frac{\Delta t}{M_I}\mathbf{f}_I^n \tag{219}$$

The backward sweep computes the nodal increment $\Delta\mathbf{U}_I$ from

$$\Delta\mathbf{U}_I + \frac{\Delta t}{M_I}\sum_{II_e\in e-}\tilde{\mathbf{A}}_{II_e}^n(\Delta\mathbf{U}_{I_e} - \Delta\mathbf{U}_I) + \frac{\Delta t}{M_I}\sum_{II_e\in e+}\tilde{\mathbf{A}}_{II_e}^n(\Delta\mathbf{U}_{I_e}^* - \Delta\mathbf{U}_I) = \frac{\Delta t}{M_I}\mathbf{f}_I^n \tag{220}$$

The complete scheme may be written, using the notation of equation (216), in the form

$$(\Pi + \Theta)\,\Pi^{-1}\,(\Pi + \Upsilon)\,\Delta\mathbf{U} = \mathbf{f} \tag{221}$$

where $\Pi + \mathbf{I} = \Pi^+ + \Pi^-$ In this format it is apparent that this is an LU implicit scheme, which can be implemented using the sweeping strategy employed for the LU factorisation method. In practice, the scheme of equation (221) performs better than the LU factorisation method and has the advantage that only one block diagonal matrix needs to be inverted for each pair of forward and backward sweeps.

8.4.3. Advanced Iterative Methods Another alternative for solving the equation system (204) comes under the general heading of Krylov methods [143, 157, 158]. Consider initially the problem of solving a linear system in the form of equation (204), where $\mathbf{K}$ is a symmetric positive–definite matrix. It is readily shown that the solution of this equation system is the vector $\mathbf{z}$ which minimises the functional

$$\Xi(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T\mathbf{K}\mathbf{z} - \mathbf{z}^T\mathbf{f} \tag{222}$$

The steepest descent method defines an iterative procedure for obtaining the solution which uses this property. At stage k, with iterate $\Delta\mathbf{U}^{(k)}$, the next iterate, $\Delta\mathbf{U}^{(k+1)}$ is determined by a one–dimensional minimisation of Ξ in the direction of its gradient. This means that, with the definition

$$\mathbf{R}^{(k)} = \mathbf{K}\Delta\mathbf{U}^{(k)} - \mathbf{f} \tag{223}$$

the iterative process is described by the relation

$$\Delta\mathbf{U}^{(k+1)} = \min_{\alpha}\Xi(\Delta\mathbf{U}^{(k)} - \alpha\mathbf{R}^{(k)}) \tag{224}$$

The conjugate gradient method is a more elaborate iterative process, in which, at the equivalent stage, a $k+1$ dimensional minimisation is performed, according to the relation

$$\Delta\mathbf{U}^{(k+1)} = \min_{\alpha_0,\alpha_1,\cdots,\alpha_k}\Xi(\Delta\mathbf{U}^{(k)} - \alpha_0\mathbf{R}^{(0)} - \alpha_1\mathbf{R}^{(1)} - \cdots - \alpha_k\mathbf{R}^{(k)}) \tag{225}$$

This algorithm can be efficiently implemented [159] so as to avoid the requirement for storing every $\mathbf{R}^{(j)}, j = 1, \cdots, k$. The benefits of this method are that it overcomes certain convergence problems associated with the steepest descent method and that it can be proved that it converges in p operations, if exact arithmetic is employed.

The conjugate gradient method, however, needs further adaption before it can be successfully applied to a general equation system of the form of equation (204), in which the matrix $\mathbf{K}$ may be non–symmetric. The most widely used extension is the general minimum residual method (GMRES) [160]. To illustrate this method, suppose $\Delta\mathbf{U}^{(0)}$ is an approximate solution to equation (204). A new approximation $\Delta\mathbf{U}^{(k)}$ is computed via the GMRES(k) method by setting

$$\Delta\mathbf{U}^{(k)} = \Delta\mathbf{U}^{(0)} + \mathbf{z}^{(k)} \tag{226}$$

and finding a solution for $\mathbf{z}^{(k)}$ in the Krylov space $\mathcal{K}_k(\mathbf{R}^{(0)}, \mathbf{K})$, of dimension k, which is spanned by the vectors $\mathbf{R}^{(0)}, \mathbf{K}\mathbf{R}^{(0)}, \mathbf{K}^2\mathbf{R}^{(0)}, \cdots, \mathbf{K}^{k-1}\mathbf{R}^{(0)}$. The solution obtained is the best possible, in the sense that it satisfies the minimisation problem

$$\left\|\mathbf{R}^{(k)}\right\| = \min_{\mathbf{z}\in\mathcal{K}_k} \left\|\mathbf{R}^{(0)} + \mathbf{K}\mathbf{z}\right\| \tag{227}$$

where $\|\cdot\|$ denotes the L_2 norm of the vector. The implementation works with an orthogonal set of basis functions, or search directions, $\mathbf{v}_1(= \mathbf{R}_0), \mathbf{v}_2, \mathbf{v}_3, \cdots, \mathbf{v}_k$, for the Krylov space. The orthogonal set is constructed by a Gram–Schmidt process and these search directions need to be stored. As the value of k is increased, the storage requirements of the method increase linearly while the number of operations involved increases quadratically. These effects are mitigated with the restarted GMRES(k,m) algorithm, which discards and recomputes the search directions every m cycles.

The performance of these GMRES algorithms may be significantly improved by preconditioning the original system. The left preconditioned iterative method solves the equation system

$$\mathbf{P}^{-1}\mathbf{K}\Delta\mathbf{U} = \mathbf{y} = \mathbf{P}^{-1}\mathbf{f} \tag{228}$$

where the matrix $\mathbf{P}$ which is employed should be easy to compute and should represent a good approximation to the matrix $\mathbf{K}$. To illustrate this process, consider the application of BILU(0) preconditioning, which is block incomplete LU preconditioning with zero fill. In this method, $\mathbf{P}$ is constructed by an incomplete LU factorisation of $\mathbf{K}$, using Crout's in place method, with all entries $\mathbf{P}_{IJ}$, for which the pair of indices (I, J) does not belong to the graph $\mathcal{G}$ of the matrix $\mathbf{K}$, set to zero. This graph is defined as

$$\mathcal{G} = \{(I, J) | \mathbf{K}_{IJ} \neq 0\} \tag{229}$$

For a general unstructured mesh, with irregular node numbering, the matrix $\mathbf{K}$ will have a complicated sparsity pattern. By renumbering the nodes, to minimise the bandwidth, the BILU(0) process leads to an improved approximation to the matrix $\mathbf{K}$. Bandwidth minimisation may be achieved by a technique such as the standard reverse Cuthill McKee procedure [161]. The explicit computation of the inverse of $\mathbf{P}$ is not advisable, as it may be dense. In addition, $\mathbf{P}$ may be ill–conditioned, leading to error prone computations of its inverse. Instead of forming explicitly a matrix vector product of the form $\mathbf{P}^{-1}\mathbf{a}(=\mathbf{b})$, it is therefore preferable to look for $\mathbf{b}$, which is the solution of the linear system $\mathbf{Pb} = \mathbf{a}$. This can be efficiently computed by a forward and backward substitution when the LU decomposition for $\mathbf{P}$ is known.

The application of a fully implicit time stepping procedure to the solution of a typical semi–discrete system, such as equation (169) or (170), leads to a non linear, non symmetric equation system

$$\mathbf{M}\Delta\mathbf{U} = \Delta t\mathbf{f}^{n+1} \tag{230}$$

To produce a numerical solution via an iterative procedure of the GMRES type, this equation is first linearised about time level t_n, in the form

$$\mathbf{M}\Delta\mathbf{U} = \Delta t\left(\mathbf{f}^n + \left.\frac{\partial\mathbf{f}}{\partial\mathbf{U}}\right|^n \Delta\mathbf{U}\right) \tag{231}$$

Upon re–arrangement, this may be expressed as a linear equation system

$$\left(\frac{\mathbf{M}}{\Delta t} + \left.\frac{\partial\mathbf{f}}{\partial\mathbf{U}}\right|^n\right)\Delta\mathbf{U} = \mathbf{f}^n \tag{232}$$

which is in the form of equation (204), with

$$\mathbf{K} = \left(\frac{\mathbf{M}}{\Delta t} + \frac{\partial\mathbf{f}^n}{\partial\mathbf{U}}\right) \tag{233}$$

The solution may now be obtained using the GMRES approach outlined above [162, 163, 164].

An advantage of using this iterative technique is that the matrix $\mathbf{K}$ need not be computed explicitly. Instead, the required matrix vector products are computed directly as required, with consequent low storage requirements. This is particularly advantageous for non–compact solution algorithms, such as the method employed in equations (198)–(200), where the number of non–zero entries in $\mathbf{K}$ could be very large. In these cases, the preconditioning matrix often needs to be computed by factorizing some approximation to $\mathbf{K}$ which only involves nearest neighbours.

Additional computational improvement may result from implementing these procedures in a parallel framework [165, 166, 167].

9. Incompressible Flows

Incompressible flow modelling is an important area of general industrial interest which has received significant attention from researchers over a prolonged period. In this area, computational techniques have been developed which allow for the modelling of complex geometrical configurations by, generally, employing multiblock or unstructured grids consisting of hexahedral cells. A relatively small amount of effort has been devoted to the development of methods, based upon a velocity–pressure formulation and equal order interpolation, for implementation on general tetrahedral grids. This is mainly due to the stability difficulties which are associated with direct application of the Galerkin method to this class of problems [168, 25].

Here, attention is limited to the 3D incompressible laminar Navier Stokes equations, which can be expressed in the non–dimensional conservation form

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} = \frac{1}{Re} \sum_{j=1}^{3} \frac{\partial^2 \mathbf{U}}{\partial x_j^2} \tag{234}$$

where

$$\mathbf{U} = \begin{bmatrix} 0 \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \qquad \mathbf{F}^j = \begin{bmatrix} u_j \\ u_1 u_j + p\delta_{1j} \\ u_2 u_j + p\delta_{2j} \\ u_3 u_j + p\delta_{3j} \end{bmatrix} \tag{235}$$

and the solution is sought subject to appropriate initial and boundary conditions. Two different approaches to the problem of numerically simulating incompressible flows will be reviewed. One approach employs an extension of the techniques which have already been reviewed for compressible flows, while the other approach has been directly developed for incompressible flow modelling.

9.1. The Artificial Compressibility Approach

Solution techniques which have been developed for compressible high speed flows may be extended into the incompressible regime. The method by which this can be achieved is to employ the artificial compressibility concept [169]. This was initially demonstrated for the simulation of steady incompressible flows using quadrilateral and hexahedral grids [170, 171]. An extension of the approach allows its implementation on general tetrahedral grids [172, 173].

To illustrate this procedure, consider the computation of steady state solutions of equation (234). An artificial time dependent term is added to the equations, which are then expressed as

$$\mathbf{P}^{-1} \frac{\partial \mathbf{U}}{\partial t} + \sum_{j=1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} = \frac{1}{Re} \sum_{j=1}^{3} \frac{\partial^2 \mathbf{U}}{\partial x_j^2} \tag{236}$$

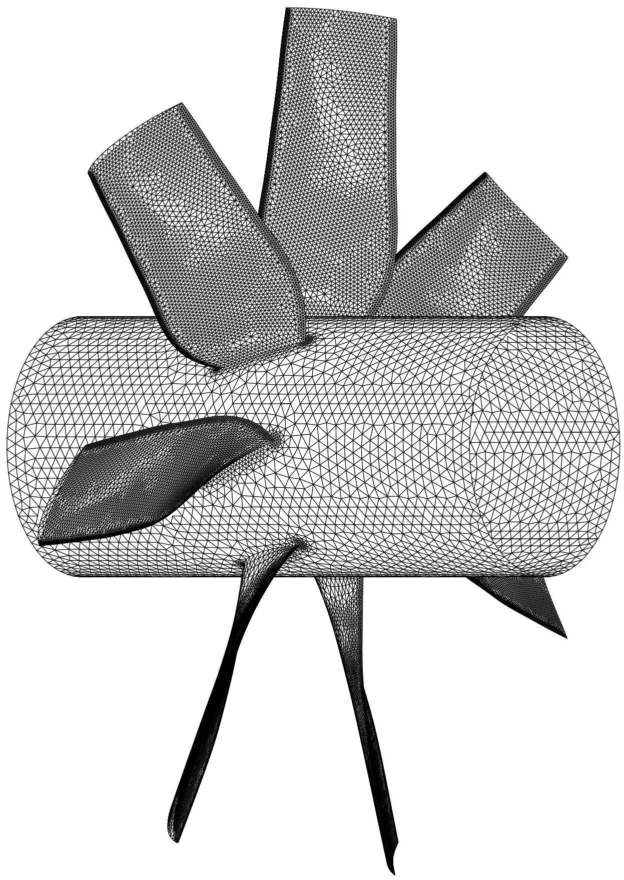

Figure 19. Incompressible inviscid flow through a rotating fan. View of the surface discretisation of the finest grid

where now

$$\mathbf{U} = \begin{bmatrix} p \\ u_1 \\ u_2 \\ u_3 \end{bmatrix} \qquad \mathbf{P} = \begin{bmatrix} \beta^2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{237}$$

The matrix $\mathbf{P}$ acts as a pre–conditioning matrix for the system and is designed to alleviate the difficulties associated with application of methods devised for the simulation of compressible flows in the incompressible limit. The value of the parameter β^2, which is the artificial compressibility, is selected to optimise the convergence to steady state [174]. It should be noted that, at steady state, the solution of the modified equation (236) is a solution of the original equation (234).

Following the approach outlined previously, relevant trial and weighting function spaces are introduced and the problem is expressed in the variational form: find $\mathbf{U}$ in

$\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\int_\Omega \mathbf{W}^T \left(\mathbf{P}^{-1} \frac{\partial \mathbf{U}^{(p)}}{\partial t} + \sum_{j+1}^{3} \frac{\partial \mathbf{F}^j}{\partial x_j} \right) d\Omega = -\frac{1}{Re} \sum_{j=1}^{3} \int_\Omega \frac{\partial \mathbf{W}^T}{\partial x_j} \frac{\partial \mathbf{U}^{(p)}}{\partial x_j} d\Omega + \frac{1}{Re} \int_\Gamma \mathbf{W}^T \frac{\partial \mathbf{U}^{(p)}}{\partial \nu} d\Gamma \tag{238}$$

for every $\mathbf{W}$ in $\mathcal{W}$ and for all $t > 0$.

Working with the subspace of the trial function space, defined in equation (122), leads to the construction of the Galerkin approximate solution via the statement: find $\mathbf{U}^{(p)}$ in $\mathcal{T}^{(p)}$, satifying the problem boundary conditions, such that

$$\int_\Omega N_I \mathbf{P}^{-1} \frac{\partial \mathbf{U}^{(p)}}{\partial t} d\Omega = -\sum_{j=1}^{3} \int_\Omega N_I \frac{\partial \mathbf{F}^j}{\partial x_j} d\Omega - \frac{1}{Re} \sum_{j=1}^{3} \int_\Omega \frac{\partial N_I}{\partial x_j} \frac{\partial \mathbf{U}^{(p)}}{\partial x_j} d\Omega \tag{239}$$

at each interior node I of the grid. If an edge based data structure is employed, the discrete form of this equation, following the use of the divergence theorem and, for

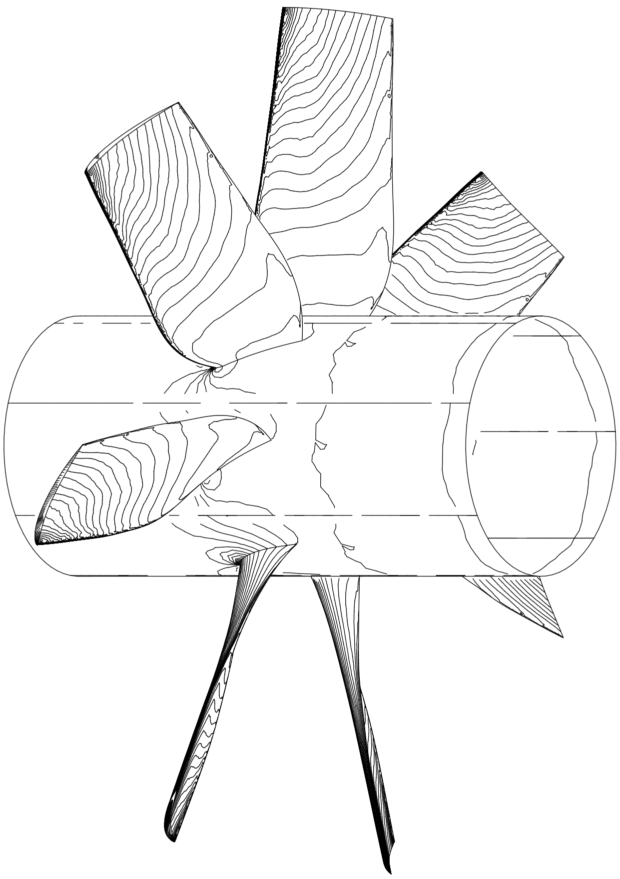

Figure 20. Incompressible inviscid flow through a rotating fan. Computed contours of pressure

example, the stabilization of equation (131), can be expressed as

$$M_I \frac{\mathrm{d}\mathbf{U}_I}{\mathrm{d}t} = \mathbf{P}_{II} \left\{ \sum_{e=1}^{\mu_I} \mathcal{F}_{II_e} - \sum_{e=1}^{\mu_I} D_e \left[\mathbf{U}_{I_e} - \mathbf{U}_I \right] \right\} \tag{240}$$

where

$$D_e = \frac{1}{Re} \sum_{E \in II_e} \sum_{j=1}^{3} \Omega_E \left. \frac{\partial N_I}{\partial x_j} \right|_E \left. \frac{\partial N_{I_e}}{\partial x_j} \right|_E \tag{241}$$

For this steady flow simulation, the left hand side matrix has been lumped.

The performance of this procedure is demonstrated by considering the simulation of steady inviscid incompressible flow through a rotating seven bladed fan. The flow is considered to be steady in a frame of reference which is rotating with the fan and, in this case, the governing equations have to be modified slightly [173]. In this simulation, the solution is advanced by the explicit multistage timestepping scheme of equation (35), together with multigrid acceleration. A view of the surface discretisation on the finest grid employed is given in Figure 19. The computed surface distribution of contours of pressure is displayed in Figure 20.

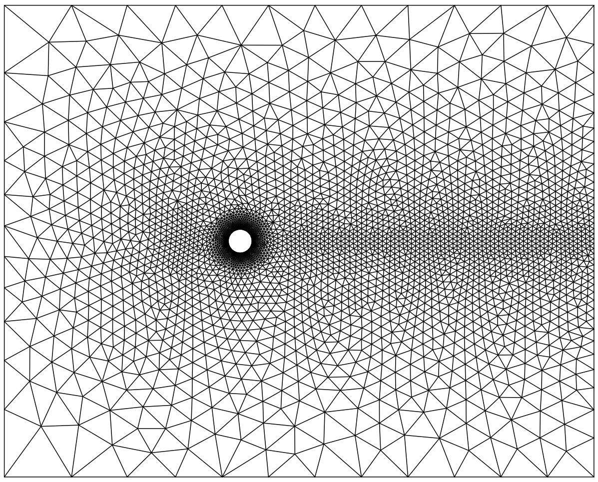

Figure 21. Incompressible viscous flow past a circular cylinder. View of the mesh employed.

9.2. Projection Methods

An alternative approach for the simulation of incompressible flows on unstructured meshes is based upon the use of a three step projection method in time [175, 176, 177].

In this method, the continuity and momentum equations are considered in the non–conservative dimensionless form

$$\sum_{j=1}^{3} \frac{\partial u_j}{\partial x_j} = 0 \tag{242}$$

$$\frac{\partial u_i}{\partial t} + \sum_{j=1}^{3} u_j \frac{\partial u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{1}{Re} \sum_{j=1}^{3} \frac{\partial^2 u_i}{\partial x_j^2} \qquad \text{for } i = 1, 2, 3 \tag{243}$$

The momentum equations are discretized in time, via an intermediate velocity $\mathbf{u}^*$, as

$$u_i^* - u_i^n = -\Delta t \sum_{j=1}^{3} u_j^n \frac{\partial u_i^n}{\partial x_j} + \frac{\Delta t}{Re} \sum_{j=1}^{3} \frac{\partial^2 u_i^n}{\partial x_j^2} \tag{244}$$

$$u_i^{n+1} - u_i^* = -\Delta t \frac{\partial p^{n+1}}{\partial x_i} \tag{245}$$

It is observed that, before $\mathbf{u}^{n+1}$ can be computed from equation (245), the value of the pressure at time level $n+1$ must be found. This is achieved by subjecting $\mathbf{u}^{n+1}$ to the mass conservation condition of equation (242), leading to

$$\sum_{j=1}^{3} \frac{\partial^2 p^{n+1}}{\partial x_j^2} = \frac{1}{\Delta t} \sum_{j=1}^{3} \frac{\partial u_j^*}{\partial x_j} \tag{246}$$

The solution at time level t_{n+1} is, therefore, achieved in three steps by first solving equation (244), then equation (246) and, finally, equation (245). This is accomplished by introducing relevant trial and weighting function spaces and expressing the problem in the variational form: find $u_i{}^*$, u_i^{n+1} and p^{n+1} in $\mathcal{T}$, satisfying the problem boundary conditions, such that

$$\begin{aligned} \int_\Omega W \left(u_i^* - u_i^n\right) d\Omega &= -\Delta t \sum_{j=1}^{3} \int_\Omega W u_j^n \frac{\partial u_i^n}{\partial x_j} d\Omega \\ -\frac{\Delta t}{Re} \sum_{j=1}^{3} \int_\Omega \frac{\partial W}{\partial x_j} \frac{\partial u_i^n}{\partial x_j} d\Omega &+ \frac{\Delta t}{Re} \sum_{j=1}^{3} \int_\Gamma W \frac{\partial u_i^n}{\partial \nu} d\Gamma \end{aligned} \tag{247}$$

$$\sum_{j=1}^{3} \int_\Omega \frac{\partial W}{\partial x_j} \frac{\partial p^{n+1}}{\partial x_j} d\Omega - \sum_{j=1}^{3} \int_\Gamma W \frac{\partial p^{n+1}}{\partial \nu} d\Gamma = -\frac{1}{\Delta t} \sum_{j=1}^{3} \int_\Omega W \frac{\partial u_j^*}{\partial x_j} d\Omega \tag{248}$$

$$\int_\Omega W \left(u_i^{n+1} - u_i^*\right) d\Omega = -\Delta t \int_\Omega W \frac{\partial p^{n+1}}{\partial x_i} d\Omega \tag{249}$$

for every W in the weighting function space. Working with a finite dimensional subspace $\mathcal{T}^{(p)}$ of the trial function space, which is such that the velocity, at any time, and the pressure distributions are approximated in the piecewise linear fashion

$$u_i^{(p)} = \sum_{J=1}^{p} u_{iJ} N_J \qquad\qquad p^{(p)} = \sum_{J=1}^{p} p_J N_J \tag{250}$$

the Galerkin approximate solution follows from the requirement that, at a typical interior node I on the grid,

$$\sum_{J=1}^{p} M_{IJ}\left(u_{iJ}^{*}-u_{iJ}^{n}\right)=f_{iI}^{1n} \tag{251}$$

$$\sum_{J=1}^{p} L_{IJ}p_{J}^{n+1}=f_{I}^{2n} \tag{252}$$

$$\sum_{J=1}^{p} M_{IJ}\left(u_{iJ}^{n+1}-u_{iJ}^{*}\right)=f_{iI}^{3n} \tag{253}$$

where

$$f_{iI}^{1}=-\Delta t\sum_{j=1}^{3}\sum_{J=1}^{p}\sum_{K=1}^{p}\left(\int_{\Omega}N_I N_J\frac{\partial N_K}{\partial x_j}d\Omega\right)u_{jJ}u_{iK}-\frac{\Delta t}{Re}\sum_{j=1}^{3}\sum_{J=1}^{p}\left(\int_{\Omega}\frac{\partial N_I}{\partial x_j}\frac{\partial N_J}{\partial x_j}d\Omega\right)u_{iJ} \tag{254}$$

$$L_{IJ}=\sum_{j=1}^{3}\int_{\Omega}\frac{\partial N_I}{\partial x_j}\frac{\partial N_J}{\partial x_j}d\Omega \qquad f_{I}^{2n}=-\frac{1}{\Delta t}\sum_{j=1}^{3}\sum_{J=1}^{p}\left(\int_{\Omega}N_I\frac{\partial N_J}{\partial x_j}d\Omega\right)u_{jJ}^{*} \tag{255}$$

$$f_{iI}^{3}=-\Delta t\sum_{J=1}^{p}\left(\int_{\Omega}N_I\frac{\partial N_J}{\partial x_i}d\Omega\right)p_J \tag{256}$$

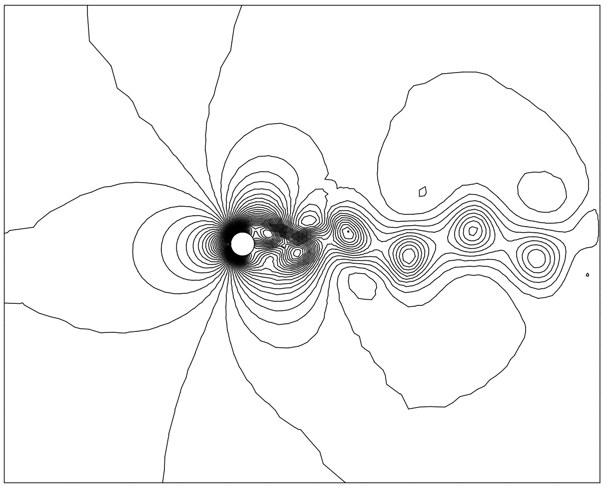

Figure 22. Incompressible viscous flow past a circular cylinder. Computed contours of pressure.

For convection dominated flows, equation (251) will need to be stabilized before it can be used in practice. For example, the stabilization can be accomplished by replacing

f_{iI}^1 on the right hand side of this equation by $\tilde{f}_{iI}^1$, where

$$\tilde{f}_{iI}^1 = f_{iI}^1 + \frac{\Delta t^2}{2} \sum_{j=1}^{3} \sum_{k=1}^{3} \sum_{J=1}^{p} \sum_{K=1}^{p} \sum_{L=1}^{p} \left[\int_\Omega \frac{\partial N_I}{\partial x_j} \frac{\partial N_J}{\partial x_k} N_L N_K d\Omega \right] u_{jK} u_{kL} u_{iJ} \tag{257}$$

When the corresponding boundary point equations are included, the numerical solution is obtained from the solution of three matrix equations. The matrices which appear in these equations are sparse and amenable to solution by iterative methods. A good choice here is the use of the preconditioned conjugate gradient algorithm, with a diagonal preconditioning matrix.

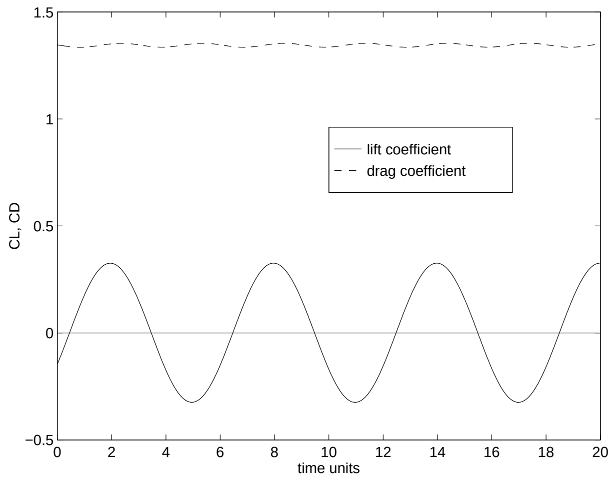

Figure 23. Incompressible viscous flow past a circular cylinder. Computed variation of lift and drag coefficients.

It should be noticed that, in this approach, the pressure is determined by solving a Poisson equation and, therefore, classical issues associated with unstable finite element velocity and pressure approximations do not arise. If the Galerkin approach is applied directly to equations (242) and (243), and the pressure equation is determined algebraically [176], then the Babuska–Brezzi condition on the allowable finite element approximation spaces must be respected [168, 25]. This approach has been applied to the simulation of incompressible flow over a circular cylinder, at a Reynolds number of one hundred [178]. The grid employed is shown in Figure 21 and consists of approximately 3000 triangular elements. A view of the computed distribution of the pressure contours at late stage of the computation is given in Figure 22. The computed histories of the lift and drag on the cylinder, after the effects of the initial transient have been removed from the system, are shown in Figure 23. The computed Strouhal number is 0.17, which is in good agreement with the results of other published numerical simulations for this problem.

10. Optimization

Computational methods have made important contributions in the area of aerodynamic design, where they have been used primarily in an analysis mode. The use of formal optimization procedures, that may guide the designer to perform the necessary changes to meet a given objective, have the potential of achieving much greater impact. To develop such methods, the problem to be optimized needs to be described in terms of a number of parameters and an appropriate cost function, which requires minimization, needs to be defined [179, 180]

For the purposes of this review, consideration is restricted to steady state problems only, in which the discrete solution $\mathbf{U}$ is such that the right hand sides of equations such as (144) or (151) become zero. Typically, the design variables $\beta = (\beta_1, \beta_2, \ldots, \beta_n)$ will affect either the geometry of the domain, i.e. the shape, or the boundary conditions, e.g. the free stream Mach number or the angle of attack. In either case, the flow equations will be written as

$$\mathbf{R}(\mathbf{U}, \beta) = \mathbf{0} \tag{258}$$

which implicitly defines $\mathbf{U}(\beta)$ as the discrete solution that satisfies the flow equations for a particular choice of the design parameters. Realistic aerodynamic optimisation exercises will require the sensitivity of a number of cost functions and constraints with respect to the design variables. A typical cost function, I, will depend upon $\mathbf{U}$ and, in some special cases, it may also have explicit dependence on β. Thus $I = I(\beta, \mathbf{U})$ and the sensitivities can then be written as

$$\frac{\mathrm{d}I}{\mathrm{d}\beta_j} = \frac{\partial I}{\partial \beta_j} + \frac{\partial I}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \beta_j} \qquad \text{for } j = 1, 2, \ldots, n \tag{259}$$

The implicit function theorem applied to equation (258) leads to

$$\frac{\partial \mathbf{R}}{\partial \mathbf{U}} \frac{\partial \mathbf{U}}{\partial \beta_j} = -\frac{\partial \mathbf{R}}{\partial \beta_j} \qquad \text{for } j = 1, 2, \ldots, n \tag{260}$$

Sensitivities may be computed using a direct or an adjoint method. With a direct method, equation (259) is solved to obtain $\partial \mathbf{U}/\partial \beta_j$ directly, while, with an adjoint method, adjoint variables ψ are defined which satisfy

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T \psi = \left[\frac{\partial I}{\partial \mathbf{U}}\right]^T \tag{261}$$

Using this result, the cost function sensitivities are then calculated according to

$$\frac{\mathrm{d}I}{\mathrm{d}\beta_j} = \frac{\partial I}{\partial \beta_j} - \psi^T \frac{\partial \mathbf{R}}{\partial \beta_j} \qquad \text{for } j = 1, 2, \ldots, n \tag{262}$$

It should be noted that, for the direct method, equation (260) needs to be solved once for each design variable. The calculated values of $\partial \mathbf{U}/\partial \beta_j$ will then be used to obtain the sensitivities with respect to all the required cost functions. On the other hand, equation (261) needs to be solved once for each cost function regardless of the number of design variables. The adjoint method is, therefore, preferred for problems in which the number of design variables is larger than the number of cost functions considered. In an alternative derivation, equation (262) is obtained directly by minimising the cost function subject to the constraint that the unknown $\mathbf{U}$ satisfies equation (258). In this case, the adjoint variables ψ may be regarded as Lagrange multipliers. Equations (260) and (261) are linear and can be solved by marching, to steady state, the modified equations

$$M_I \frac{d\mathbf{U}_{\beta I}}{dt} = -\sum_{J=1}^{p} \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)_{IJ} \mathbf{U}_{\beta J} - \frac{\partial \mathbf{R}_I}{\partial \beta} \tag{263}$$

and

$$M_I \frac{d\psi_I}{dt} = -\sum_{J=1}^{p} \left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)_{JI} \psi_J + \left(\frac{\partial I}{\partial \mathbf{U}}\right)_I^T \tag{264}$$

where $\mathbf{U}_\beta = \partial \mathbf{U}/\partial \beta$. Note that the stability limitations of equations (263) and (264) are identical to those of equations such as (144) or (151).

The matrix $\partial \mathbf{R}/\partial \mathbf{U}$ and the grid sensitivity matrix $\partial \mathbf{R}/\partial \mathbf{x}$, which are needed in the calculation of $\partial \mathbf{R}/\partial \beta$, are constructed in the same manner as $\mathbf{R}$, by looping over edges or elements of the grid. For shape optimization problems, requiring changes in the domain geometry, grid motion is accomplished by using iteration of the type described in Section 7.3.1. If δ^j denotes the grid displacement due to the jth iteration, i.e.

$$\mathbf{x}^{j+1} = \mathbf{x}^j + \delta^j \tag{265}$$

then the grid sensitivity is simply determined as

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\beta} = \frac{\partial \delta^N}{\partial \beta} = \frac{\partial \delta^N}{\partial \delta^{N-1}} \frac{\partial \delta^{N-1}}{\partial \delta^{N-2}} \cdots \frac{\partial \delta^1}{\partial \beta} \tag{266}$$

where N iterations of the grid relaxation scheme have been performed.

Once the sensitivities are calculated, the design variables are updated using a restarted BFGS method [181]. For problems involving constrained optimization, such as drag minimization at constant lift, the constraints can be built into the gradient calculation, when using the adjoint approach, by adding additional Lagrange multipliers. Alternatively, a constrained minimisation algorithm can be used, in which case, the sensitivity of the constraints with respect to the design variables needs to be computed using the procedures outlined above.

To illustrate the approach, the inviscid optimization of a business jet configuration, consisting of wing, body, fins, tails and fuselage mounted engines, is considered using the adjoint approach [182]. The tetrahedral grid employed consists of approximately

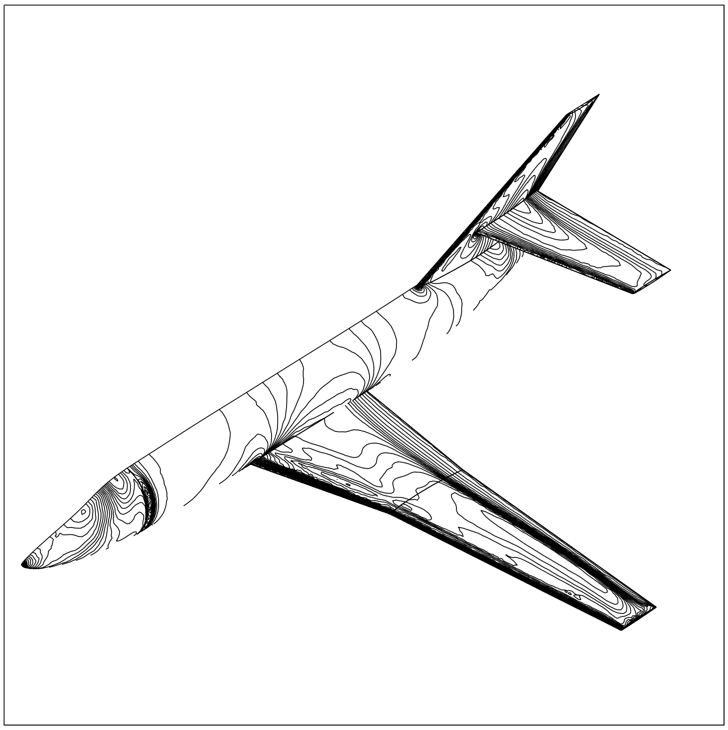

Figure 24. Target pressure distribution for the flow over a business jet configuration

160 000 nodes. Six design variables are employed to model shape changes in the inboard section of the wing and the computational implementation is undertaken in parallel. The objective function is the surface integral of the squares of the differences between the computed and some target pressure distributions. The far field conditions are taken to be a Mach number of 0.85 and an angle of attack of 2 degrees. The target pressure distribution is shown in Figure 24 and corresponds to the solution obtained when the engines and pylons are removed. The computed pressure distribution on the baseline configuration is shown in Figure 25 and the computed pressure distribution for the converged geometry, obtained after seven optimization iterations, is shown in Figure 26. It is apparent from this solution that much of the lift of the clean wing case target pressure distribution has been recovered.

It should be noted that, in the described approach, the sensitivities of the discrete solution with respect to the design variables are computed exactly. An alternative approach computes the sensitivities by a finite difference method [179], which leads to much simpler codes at the expense of robustness. A further alternative is that in which the sensitivity of the exact solution is formulated and the resulting problem is then approximated numerically [183, 184].

11. Conclusions

A review has been made of the current status of unstructured grid finite element methods for the solution of the equations of fluid flow. In the main, the review has reflected the experience of the authors and has concentrated solely upon methods which are based upon the use of a Galerkin approximation in space, with appropriate stabilization, together with the finite difference method in time. The examples which have been included demonstrate the power of an approach when it is employed in conjunction with a tetrahedral discretisation of the computational domain. The review has indicated that mesh adaptivity, improving computational performance and optimization will be important areas for future research.

Acknowledgments

The authors wish to express their appreciation to their colleagues J. Elliott, O. Hassan, J. Peiró, E. J. Probert and N. P. Weatherill for their contributions to the work which has been included in this review.

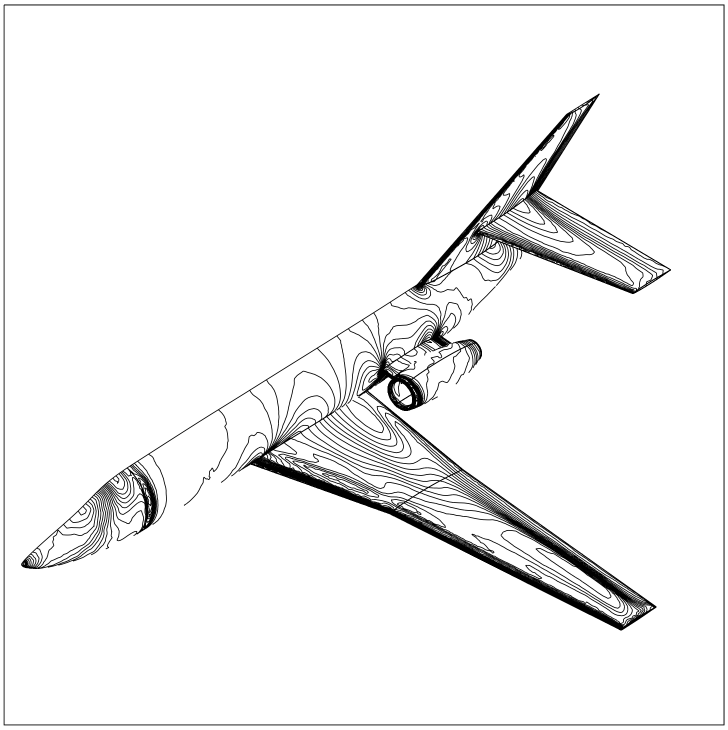

Figure 25. Computed pressure distribution for the baseline business jet configuration

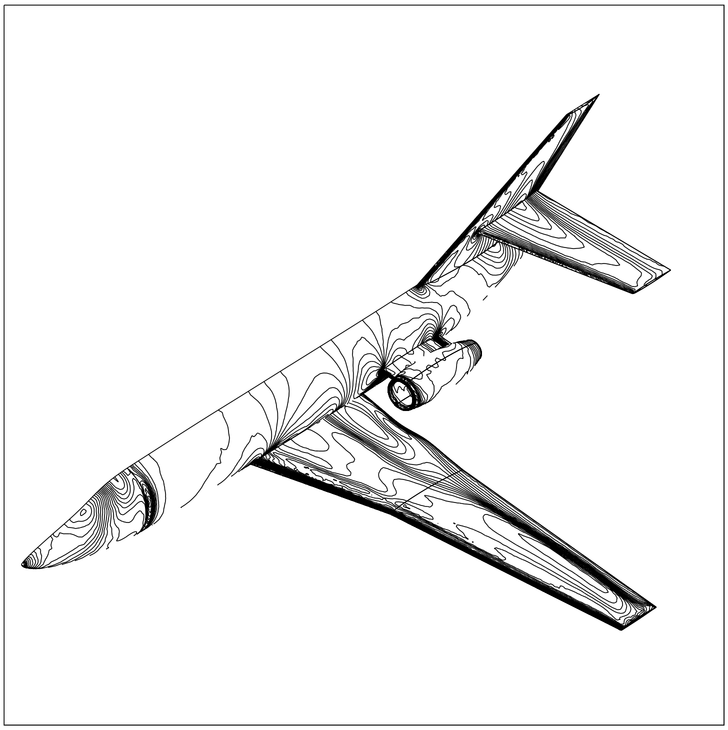

Figure 26. Computed pressure distribution for the optimized business jet configuration

References

[1] C. Hirsch, *Numerical Computation of Internal and External Flows. Volume 2: Computational Methods for Inviscid and Viscous Flows*, John Wiley, Chichester, 1990.

[2] A. J. Baker, *Finite Element Computational Fluid Mechanics*, McGraw–Hill, New York, 1983.

[3] P. G. Ciarlet, *The Finite Element Method for Elliptic Problems*, North–Holland, Amsterdam, 1980.

[4] C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, 1987.

[5] S. L. Karman, SPLITFLOW: a 3D unstructured cartesian/prismatic grid CFD code for complex geometries, *AIAA Paper 95–0343*, 1995.

[6] S. S. Samant, J. E. Bussoletti, F. T. Johnson, R. H. Burkhart, B. L. Everson, R. G. Melvin, D. P. Young, L. L. Erickson, M. D. Madson and A. C. Woo, TRANAIR: a computer code for transonic analyses of arbitrary configurations, *AIAA Paper 87–0034*, 1987.

[7] S. Allwright, Multiblock topology specification and grid generation for complete aircraft configurations, in *Conference Proceedings No. 464: Applications of Mesh Generation to Complex 3–D Configurations*, AGARD, Paris, 11.1–11.11, 1990.

[8] J. L. Steger and J. A. Benek, On the use of composite grid schemes in computational aerodynamics,

Computer Methods in Applied Mechanics and Engineering 64, 301–317, 1987.

[9] C. M. Albone and G. Joyce, Feature–associated mesh embedding for complex configurations, in *Conference Proceedings No. 464: Applications of Mesh Generation to Complex 3–D Configurations*, AGARD, Paris, 13.1–13.12, 1990.

[10] J. Peraire, K. Morgan and J. Peiró, Unstructured finite element mesh generation and adaptive procedures for CFD, in *Conference Proceedings No. 464: Applications of Mesh Generation to Complex 3–D Configurations*, AGARD, Paris, 18.1–18.12, 1990.

[11] N. P. Weatherill and O. Hassan, Efficient three–dimensional Delaunay triangulation with automatic point creation and imposed boundary constraints, *International Journal for Numerical Methods in Engineering 37*, 2005–2039, 1994.

[12] J. Peraire and K. Morgan, Unstructured mesh generation including directional refinement for aerodynamic flow simulation, *Finite Elements in Analysis and Design 25*, 343–356, 1997.

[13] N. P. Weatherill, O. Hassan, K. Morgan and M. J. Marchant, Large scale computations on unstructured grids, in : F. Benkhaldoun and R. Vilsmeier, editors, *Proceedings of the Conference on Finite Volumes for Complex Applications*, Hermes, Paris, 77–98, 1996.

[14] J. A. Shaw, J. M. Georgala and P. N. Childs, General procedures employed in the generation of three–dimensional hybrid structured/unstructured meshes, in N. P. Weatherill et al, editors, *Proceedings of the 4th International Conference on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*, Pineridge Press, Swansea, 297–303, 1994.

[15] A. J. Chorin and J. E. Marsden, *A Mathematical Introduction to Fluid Mechanics, 3rd Edition*, Springer–Verlag, New York, 1993.

[16] R. D. Milne, *Applied Functional Analysis. An Introductory Treatment*, Pitman, Boston, 1980

[17] O. C. Zienkiewicz and K. Morgan, *Finite Elements and Approximation*, John Wiley, New York, 1983

[18] T. J. R. Hughes and F. Shakib, Computational aerodynamics and the finite element method, *AIAA Paper 88–0031*, 1988.

[19] T. J. R. Hughes, New directions in computational mechanics, *Nuclear Engineering and Design 114*, 197–210, 1989.

[20] T. J. R. Hughes, G. Hauke, K. Jansen and Z. Johan, Current reflections on stabilized finite element methods for computational fluid mechanics, in: K. Morgan et al, editors, *Finite Elements in Fluids—New Trends and Applications. Part 1*, CIMNE, Barcelona, 44–63, 1993.

[21] M. Hafez and M. Soliman, Numerical solution of the incompressible Navier–Stokes equations in primitive variables on unstaggered grids, *AIAA Paper 91–1561–CP*, 1991.

[22] B. N. Jiang and L. A. Povinelli, Least–squares finite element method for fluid dynamics, *Computer Methods in Applied Mechanics and Engineering 81*, 13–37, 1990.

[23] D. Lefebvre, J. Peraire and K. Morgan, Finite element least squares solution of the Euler equations using linear and quadratic approximations, *International Journal of Computational Fluid Dynamics 1*, 1–23, 1993.

[24] O. C. Zienkiewicz and R. Codina, A general algorithm for compressible and incompressible flow. Part I: The split characteristic based scheme, *International Journal for Numerical Methods in Fluids 20*, 869–885, 1995.

[25] O. Pironneau, *Finite Element Methods for Fluids*, John Wiley, Chichester, 1989.

[26] P. Lesaint and P. A. Raviart, On the finite element method for solving the neutron transport equation, in C. de Boor, editor, *Mathematical Aspects of Finite Elements in Partial Differential*

Equations, Academic Press, London, 89–123, 1974.

[27] B. Cockburn and C. W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework, *Mathematics of Computation 52*, 411–435, 1989.

[28] C. E. Baumann and J. T. Oden, A discontinuous *hp* finite element method for the Euler equations of gas dynamics, *Computer Methods in Applied Mechanics and Engineering*, 1997 (in press).

[29] I. Christie, D. F. Griffiths, A. R. Mitchell and O. C. Zienkiewicz, Finite element methods for second order differential equations with significant first derivatives, *International Journal for Numerical Methods in Engineering 10*, 1389–1396, 1976.

[30] J. Donéa, Recent advances in computational methods for steady and transient transport problems, *Nuclear Engineering and Design 80*, 141–162, 1984.

[31] C. Hirsch, *Numerical Computation of Internal and External Flows. Volume 1: Fundamentals of Numerical Discretization*, John Wiley, Chichester, 1988.

[32] J. L. Steger, Implicit finite difference simulation of flow about two–dimensional geometries, *AIAA Journal 16*, 679–686, 1978.

[33] D. R. Lindquist and M. B. Giles, A comparison of numerical schemes on triangular and quadrilateral meshes, in D. L. Dwoyer, M. Y. Hussaini and R. G. Voigt, editors, *Lecture Notes in Physics, Volume 323: Proceedings of the 11th International Conference on Numerical Methods in Fluid Dynamics*, Springer–Verlag, 363–373, 1989.

[34] M. B. Giles, Accuracy of node based solutions on irregular meshes, in D. L. Dwoyer, M. Y. Hussaini and R. G. Voigt, editors, *Lecture Notes in Physics, Volume 323: Proceedings of the 11th International Conference on Numerical Methods in Fluid Dynamics*, Springer–Verlag, 272–277, 1989.

[35] A. N. Brooks and T. J. R. Hughes, Streamline upwind/Petrov–Galerkin formulations for convective dominated flows with particular emphasis on the incompressible Navier–Stokes equations, *Computer Methods in Applied Mechanics and Engineering 32*, 199–259, 1982.

[36] R. Codina, Comparison of some finite element methods for solving the diffusion–reaction equation, *Computer Methods in Applied Mechanics and Engineering*, 1997 (in press).

[37] P. D. Lax and B. Wendroff, Systems of conservation laws, *Communications on Pure and Applied Mathematics 13*, 217–237, 1960.

[38] J. Donéa, A Taylor–Galerkin method for convection transport problems, *International Journal for Numerical Methods in Engineering 20*, 101–119, 1984.

[39] M. Giles, Energy stability analysis of multi–step methods on unstructured meshes, *Technical Report CFDL–TR–87–1*, Massachusetts Institute of Technology, 1987.

[40] P. J. van der Howen, *Construction of Integration Formulas for Initial Value Problems*, North–Holland, Amsterdam, 1977.

[41] R. W. MacCormack and B. S. Baldwin, A numerical method for solving the Navier–Stokes equations with application to shock–boundary layer interaction, *AIAA Paper 75–1*, 1975.

[42] K. Morgan, J. Peraire and J. Peiró, Unstructured grid methods for compressible flows, in *Report No. 787: Unstructured Grid Methods for Advection Dominated Flows*, AGARD, Paris, 5.1–5.39, 1992.

[43] R. C. Swanson and E. Turkel, On central difference and upwind schemes, *ICASE Report No. 90–44*, NASA Langley Research Center, 1990.

[44] A. Jameson, W. Schmidt and E. Turkel, Numerical simulation of the Euler equations by finite

volume methods using Runge–Kutta time stepping schemes, *AIAA Paper 81–1259*, 1981.

[45] A. Jameson, Transonic aerofoil calculations using the Euler equations, in P. L. Roe, editor, *Numerical Methods in Aeronautical Fluid Dynamics*, Academic Press, New York, 1982.

[46] P. Hansbo and C. Johnson, Adaptive streamline diffusion methods for compressible flow using conservation variables, *Computer Methods in Applied Mechanics and Engineering 87*, 267–280, 1991.

[47] P. Hansbo, Explicit streamline diffusion finite element methods for the compressible Euler equations in conservation variables, *Journal of Computational Physics 109*, 272–288, 1993.

[48] J. Donéa and S. Giuliani, A simple method to generate high–order accurate convection operators for explicit schemes based on linear finite elements, *International Journal for Numerical Methods in Fluids 1*, 63–79, 1981.

[49] J. P. Boris and D. L. Book, Flux corrected transport: I. SHASTA, a fluid transport algorithm that works, *Journal of Computational Physics 11*, 38–69, 1973.

[50] S. T. Zalesak, Fully multidimensional flux–corrected transport, *Journal of Computational Physics 31*, 355–362, 1979.

[51] A. K. Parrott and M. K. Christie, FCT applied to the 2D finite element solution of tracer transport by single phase flow in a porous medium, in K. W. Morton and M. J. Baines, editors, *Numerical Methods for Fluid Dynamics*, Oxford University Press, 27–53, 1986.

[52] R. Löhner, K. Morgan, J. Peraire and M. Vahdati, Finite element flux–corrected transport (FEM–FCT) for the Euler and Navier–Stokes equations, *International Journal for Numerical Methods in Fluids 7*, 103–109, 1987.

[53] A. Harten, High resolution schemes for hyperbolic conservation laws, *Journal of Computational Physics 49*, 357–393, 1983.

[54] P. Roe, Approximate Riemann solvers, parameter vectors and difference schemes, *Journal of Computational Physics 43*, 357–372, 1981.

[55] G. J. Le Beau, S. E. Ray, S. K. Aliabadi and T. E. Tezduyar, SUPG finite element computation of compressible flows with the entropy and conservation variables formulation, *Computer Methods in Applied Mechanics and Engineering 104*, 397–422, 1993.

[56] T. J. Barth, Aspects of unstructured grids and finite–volume solvers for the Euler and Navier–Stokes equations, in *Lecture Series 1994–05: Computational Fluid Dynamics*, von Karman Institute for Fluid Dynamics, Brussels, 1994.

[57] K. Morgan, J. Peraire, J. Peiró and O. Hassan, Unstructured grid methods for high speed compressible flows, in J. R. Whiteman, editor, *The Mathematics of Finite Elements and Applications: Highlights 1993*, John Wiley, Chichester, 215–241, 1994.

[58] A. J. George, Computer implementation of the finite element method, *PhD Thesis STAN–CS–71–208*, Stanford University, 1971.

[59] S. H. Lo, A new mesh generation scheme for arbitrary planar domains, *International Journal for Numerical Methods in Engineering 21*, 1403–1426, 1985.

[60] J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz, Adaptive remeshing for compressible flow computations, *Journal of Computational Physics 72*, 449–466, 1987.

[61] J. Peraire, J. Peiró and K. Morgan, Adaptive remeshing for three–dimensional compressible flow computations, *Journal of Computational Physics 103*, 269–285, 1992.

[62] J. Bonet and J. Peraire, An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems, *International Journal for Numerical Methods in Engineering 31*, 1–17,

1991.

[63] R. Löhner, Extensions and improvements of the advancing front grid generation technique, *Communications in Numerical Methods in Engineering 12*, 683–702, 1996.

[64] A. Bowyer, Computing Dirichlet tessellations, *Computer Journal 24*, 162–166, 1981.

[65] D.F. Watson, Computing the n–dimensional Delaunay tessellation with application to Voronoi polytopes, *Computer Journal 24*, 167–172, 1981.

[66] P. L. George, F. Hecht and E. Saltel, Automatic mesh generator with specified boundary, *Computer Methods in Applied Mechanics and Engineering 92*, 269–288, 1991.

[67] N. P. Weatherill, The reconstruction of boundary contours and surfaces in arbitrary unstructured triangular and tetrahedral grids, *Engineering Computations 13*, 64–79, 1996.

[68] J. Peraire, K. Morgan and N. P. Weatherill, A comparison of Delaunay and advancing front methods of mesh generation for 3D configurations, in *Proceedings of the Second World Congress on Computational Mechanics*, University of Stuttgart, 1069–1072, 1990.

[69] K. Nakahashi, FDM–FEM zonal approach for viscous flow computations over multiple bodies, *AIAA Paper 87–0604*, 1987.

[70] O. Hassan, K. Morgan and J. Peraire, An implicit/explicit scheme for compressible viscous high speed flow, *Computer Methods in Applied Mechanics and Engineering 76*, 245–258, 1989.

[71] R. Löhner, Matching semi–structured and unstructured grids for Navier–Stokes calculations, *AIAA Paper 93–3348–CP*, 1993.

[72] Y. Kallinderis, A. Khawaja and H. McMorris, Hybrid prismatic/tetrahedral grid generation for complex geometries, *AIAA Paper 95–0211*, 1995.

[73] S. D. Connell and M. E. Braaten, Semi–structured mesh generation for 3D Navier–Stokes calculations, *Report 94CRD15*, Mechanical Systems Laboratory, General Electric R & D Center, 1994.

[74] S. Pirzadeh, Viscous unstructured three–dimensional grids by the advancing layers method, *AIAA Paper 94–0417*, 1994.

[75] S. Pirzadeh, Progress toward a user–oriented unstructured viscous grid generator, *AIAA Paper 96–0031*, 1996.

[76] O. Hassan, K. Morgan, E. J. Probert and J. Peraire, Unstructured tetrahedral mesh generation for three–dimensional viscous flows, *International Journal for Numerical Methods in Engineering 39*, 549–567, 1996.

[77] M. J. Marchant and N. P. Weatherill, Unstructured grid generation for viscous flow simulations, in N.P. Weatherill et al, editors, *Proceedings of the 4th International Conference on Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*, Pineridge Press, Swansea, 151–162, 1994.

[78] L. Lapidus and G. F. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering*, John Wiley, New York, 1982.

[79] T. J. R. Hughes, M. Mallet and A. Mizukami, A new finite element formulation for computational fluid dynamics: II. Beyond SUPG, *Computer Methods in Applied Mechanics and Engineering 54*, 341–355, 1986.

[80] T. J. R. Hughes and M. Mallet, A new finite element formulation for computational fluid dynamics: IV. A discontinuity–capturing operator for multidimensional advective–diffusive systems, *Computer Methods in Applied Mechanics and Engineering 58*, 329–336, 1986.

[81] R. Codina, A discontinuity–capturing crosswind–dissipation for the finite element solution of the

convection–diffusion equation, *Computer Methods in Applied Mechanics and Engineering 110*, 325–342, 1993.

[82] A. Jameson, Positive schemes and shock modelling for compressible flows, *International Journal for Numerical Methods in Fluids 20*, 743–776, 1995.

[83] B. Laney and D. A. Caughey, Extremum control II: Semi–discrete approximations to conservation laws, *AIAA Paper 91–0623*, 1991.

[84] K. L. Bibb, J. Peraire and C. J. Riley, Hypersonic flow computations on unstructured meshes, *AIAA Paper 97–0624*, 1997.

[85] P. R. M. Lyra, K. Morgan and J. Peraire, A high resolution flux splitting scheme for the solution of the compressible Navier–Stokes equations on triangular grids, in F.–K. Hebeker, R. Rannacher and G. Wittum, editors, *Numerical Methods for the Navier–Stokes Equations*, Vieweg, Braunschweig, 167–180, 1994.

[86] R. Löhner, K. Morgan, J. Peraire and O. C. Zienkiewicz, Finite element methods for high speed flows, *AIAA Paper 85–1531–CP*, 1985.

[87] S. K. Aliabadi, S. E. Ray and T. E. Tezduyar, SUPG finite element computation of viscous compressible flows based on the conservation and entropy variables formulation, *Computational Mechanics 11*, 300–312, 1993.

[88] D. C. Wilcox, *Turbulence Modelling for CFD*, DCW Industries, Inc., La Cañada, 1993.

[89] P. Rostand, Algebraic turbulence models for the computation of two–dimensional high–speed flows using unstructured grids, *International Journal for Numerical Methods in Fluids 9*, 1121–1143, 1989.

[90] D. J. Mavriplis, Algebraic turbulence modeling for unstructured and adaptive meshes, *AIAA Journal 29*, 2086–2093, 1991.

[91] M. Vahdati, K. Morgan and J. Peraire, Computation of viscous compressible flows using an upwind algorithm and unstructured meshes, in S. N. Atluri, editor, *Progress in Aeronautics and Astronautics Volume 146: Computational Nonlinear Mechanics in Aerospace Engineering*, AIAA, 479–505, 1992.

[92] D. J. Mavriplis, A three–dimensional multigrid Reynolds–averaged Navier–Stokes solver for unstructured meshes, *AIAA Paper 94–1878*, 1994.

[93] M. T. Manzari, O. Hassan, K. Morgan and N. P. Weatherill, The parallel computation of turbulent flow about aerospace configurations, in C. Taylor and J.T. Cross, editors, *Proceedings of the 10th International Conference on Numerical Methods in Laminar and Turbulent Flow*, Pineridge Press, Swansea, 1073–1084, 1997.

[94] D. L. Marcum and N. P. Weatherill, Turbulence models for unstructured finite element calculations, *International Journal for Numerical Methods in Fluids 20*, 803–817, 1995.

[95] M. T. Manzari, O. Hassan, K. Morgan and N. P. Weatherill, Turbulent flow computations on 3D unstructured grids, *Finite Elements in Analysis and Design*, 1997 (in press).

[96] M. Ainsworth and J. T. Oden, A posteriori error estimation in finite element analysis, *TICAM Report 96–19*, The University of Texas at Austin, 1996.

[97] O. C. Zienkiewicz and J. Z. Zhu, The superconvergent patch recovery and a posteriori error estimates. Part 2: Error estimates and adaptivity, *International Journal for Numerical Methods in Engineering 33*, 1365–1382, 1992.

[98] C. Johnson and P. Hansbo, Adaptive finite element methods in computational mechanics, *Computer Methods in Applied Mechanics and Engineering 101*, 143–181, 1992.

[99] I. Babuska and W. C. Rheinboldt, A posteriori error estimates for the finite element method, *International Journal for Numerical Methods in Engineering 12*, 1597–1615, 1978.

[100] R. E. Bank and A. Weiser, Some a posteriori error estimators for elliptic partial differential equations, *Mathematics of Computation 44*, 283–301, 1985.

[101] P. Ladeveze and D. Leguillon, Error estimation procedures in the finite element method and applications, *SIAM Journal of Numerical Analysis 20*, 485–509, 1983.

[102] M. Ainsworth and J. T. Oden, A unified approach to a posteriori error estimation using residual methods, *Numerische Mathematik 65*, 25–50, 1993.

[103] R. Becker and R. Rannacher, Weighted a posteriori error control in finite element methods, *IWR Preprint 96–1*, University of Heidelberg, 1996.

[104] M. Paraschivoiu, J. Peraire and A. T. Patera, A posteiriori finite element bounds for linear–functional outputs of elliptic partial differential equations, *Computer Methods in Applied Mechanics and Engineering*, 1997 (in press).

[105] R. Löhner, K. Morgan and O. C. Zienkiewicz, Adaptive grid refinement for the compressible Euler equations, in I. Babuska et al, editors, *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, John Wiley, Chichester, 281–297, 1986.

[106] O. Hassan, K. Morgan, J. Peraire, E. J. Probert and R. R. Thareja, Adaptive unstructured mesh methods for steady viscous flow, AIAA Paper 91–1538, 1991.

[107] O. Hassan, E. J. Probert, K. Morgan and J. Peraire, Mesh generation and adaptivity for the solution of compressible viscous high speed flow, *International Journal for Numerical Methods in Engineering 38*, 1123–1148, 1995.

[108] F. Ilinca, D. Pelletier and F. Arnoux–Guisse, An adaptive finite element scheme for turbulent free shear flows, *International Journal of Computational Fluid Dynamics 8*, 171–188, 1997.

[109] M. Fortin, M.–G. Vallet, J. Dompierre, Y. Bourgault and W. G. Habashi, Anisotropic mesh adaptation: theory, Validation and applications, in J.–A. Désidéri et al, editors, *Computational Fluid Dynamics '96—Proceedings of the 3rd ECCOMAS Computational Fluid Dynamics Conference*, John Wiley, Chichester, 174–180, 1996.

[110] M. J. Castro–Diaz, H. Borouchaki, P. L. George, F. Hecht and B. Mohammadi, Anisotropic adaptive mesh generation in two dimensions for CFD, in J.–A. Désidéri et al, editors, *Computational Fluid Dynamics '96—Proceedings of the 3rd ECCOMAS Computational Fluid Dynamics Conference*, John Wiley, Chichester, 181–186, 1996.

[111] R. Vilsmeier and D. Hänel, Adaptive solutions for unsteady laminar flows on unstructured grids, *International Journal for Numerical methods in Fluids 22*, 85–101, 1996.

[112] V. Kalro, S. Aliabadi, W. Garrard, T. Tezduyar, S. Mittal and K. Stein, Parallel finite element simulation of large ram–air parachutes, *International Journal for Numerical Methods in Fluids 24*, 1353–1369, 1997.

[113] A. A. Johnson and T. E. Tezduyar, 3D simulation of fluid–particle interactions with the number of particles reaching 100, *Computer Methods in Applied Mechanics and Engineering 145*, 301–321, 1997.

[114] J. D. Baum, H. Luo and R. Löhner, A new ALE adaptive unstructured methodology for the simulation of moving bodies, AIAA Paper 94–0414, 1994.

[115] K. Morgan, L. B. Bayne, O. Hassan, E. J. Probert and N. P. Weatherill, The simulation of 3D unsteady inviscid compressible flows with moving boundaries, in M.–O. Bristeau et al, editors, *Computational Science for the 21st Century*, John Wiley, Chichester, 347–356, 1997.

[116] T. Tezduyar, S. Aliabadi and M. Behr, Enhanced–discretization interface–capturing technique (EDICT) for computation of unsteady flows with interfaces, Preprint 97–027, Army High Performance Computing Center, Minneapolis, 1997.

[117] C. Fahrat, High performance simulation of coupled nonlinear transient aeroelastic problems, in *Report R–807: Special Course on Parallel Computing in CFD*, AGARD, Paris, 8.1–8.79, 1995.

[118] J. T. Batina, Three–dimensional flux–split Euler schemes involving unstructured dynamic meshes, AIAA Paper 90–1649, 1990.

[119] K. P. Singh and O. Baysal, 3–D unstructured method for flows past bodies in 6–DOF relative motion, in M. M. Hafez, editor, *Proceedings of the 6th International Symposium of Computational Fluid Dynamics*, University of California Davis, 1161–1168, 1995.

[120] A. Huerta and W. K. Liu, Viscous flow structure interaction, *Journal of Pressure Vessel Technology 110*, 15–21, 1988.

[121] R. Löhner, K. Morgan and O. C. Zienkiewicz, The use of domain splitting with an explicit hyperbolic solver, *Computer Methods in Applied Mechanics and Engineering 45*, 313–329, 1984.

[122] O. Hassan, E. J. Probert, K. Morgan and J. Peraire, Adaptive finite element methods for transient compressible flow problems, in C. A. Brebbia and M. H. Aliabadi, editors, *Adaptive Finite and Boundary Element Methods*, Elsevier Applied Science, London, 119–160, 1993.

[123] H. D. Simon, Seven years of parallel computing at NAS (1987–1994): What have we learned?, *AIAA Paper 95–0219*, 1995.

[124] J. Cabello, Parallel explicit unstructured grid solvers on distributed memory computers, *Advances in Engineering Software 26*, 189–200, 1996.

[125] K. Morgan, N. P. Weatherill, O. Hassan, P. J. Brookes, M. T. Manzari and R. Said, Aerospace engineering on parallel computers, in D. A. Caughey and M. M. Hafez, editors, *Frontiers of Computational Fluid Dynamics—1997*, John Wiley, Chichester, 1997 (in press).

[126] A. Shostko and R. Löhner, Three–dimensional parallel unstructured grid generation, *International Journal for Numerical Methods in Engineering 38*, 905–925, 1995.

[127] N. A. Verhoeven, R. Said, N. P. Weatherill and K. Morgan, Delaunay mesh generation on distributed parallel platforms, in B. H. V. Topping, editor, *Proceedings of EURO–PAR97, Parallel and Distributed Computing for Computational Mechanics: Pre–Processing and Solution Procedures*, Saxe–Coburg Press, Edinburgh, 1997 (in press).

[128] T. Okusanya and J. Peraire, Parallel unstructured mesh generation, in *Proceedings of McNU97: Joint ASME/ASCE/SES Summer Meeting*, Northwestern University, 1997.

[129] C. Greenhough and R. F. Fowler, Partitioning methods for unstructured finite element meshes, *Report RAL–94–092*, Rutherford Appleton Laboratory, 1994.

[130] H. D. Simon, Partitioning of unstructured problems for parallel processing, *Computational Systems Engineering 2*, 135–148, 1991.

[131] K. Morgan, N. P. Weatherill, O. Hassan, M. T. Manzari, L. B. Bayne and P. J. Brookes, Parallel processing for large scale aerospace engineering simulations, in A. Ecer et al, editors, *Parallel Computational Fluid Dynamics '97*, Elsevier Science, Amsterdam, 1997 (in press).

[132] P. I. Crumpton and M. B. Giles, Aircraft computations using multigrid and an unstructured parallel library, AIAA Paper 95–0210, 1995.

[133] G. Karypis and V. Kumar, METIS: unstructured graph partitioningand sparse matrix ordering system, *Technical Report*, Department of Computer Science, University of Minnesota, 1995.

[134] A. Jameson, Solution of the Euler equations by a multigrid method, *Applied Mathematics and*

Computations 13, 327–356, 1983.

[135] D. J. Mavriplis, Multigrid techniques for unstructured meshes, *ICASE Report No. 95–27*, NASA Langley Research Center, 1995.

[136] M. Lallemand, H. Steve and A. Dervieux, Unstructured multigridding by volume agglomeration: current status, *Computers and Fluids 21*, 397–433, 1994.

[137] V. Venkatakrishnan and D. J. Mavriplis, Agglomeration multigrid for the three–dimensional Euler equations, *AIAA Paper 94–0069*, 1994.

[138] S. R. Elias, G. D. Stubley and G. D. Raithby, An adaptive agglomeration method for additive correction multigrid, *International Journal for Numerical Methods in Engineering 40*, 887–903, 1997.

[139] P. I. Crumpton, P. Moinier and M. B. Giles, An unstructured algorithm for high Reynolds number flows on highly–stretched grids, in C. Taylor and J. T. Cross, editors, *Proceedings of the 10th International Conference on Numerical Methods in Laminar and Turbulent Flow*, Pineridge Press, Swansea, 561–572, 1997.

[140] D. J. Mavriplis, Three–dimensional multigrid for the Euler equations, *AIAA Journal 30*, 1753–1761, 1992.

[141] J. Peraire, J. Peiró and K. Morgan, Multigrid solution of the 3D compressible Euler equations on unstructured tetrahedral grids, *International Journal for Numerical Methods in Engineering 36*, 1029–1044, 1993.

[142] J. Peraire, J. Peiró and K. Morgan, Finite element multigrid solution of Euler flows past installed aero–engines, *Computational Mechanics 11*, 433–451, 1993.

[143] V. Venkatakrishnan, Implicit schemes and parallel computing in unstructured grid CFD, *ICASE Report No. 95–28*, NASA Langley Research Center, 1995.

[144] A. Brennis and A. Eberle, Application of an implicit relaxation method solving the Euler equations for time–accurate unsteady problems, *Transactions of ASME: Journal of Fluids Engineering 112*, 510–520, 1990.

[145] A. Jameson, Time–dependent calculations using multigrid, with applications to unsteady flow past airfoils and wings, *AIAA Paper 91–1596*, 1991.

[146] V. Venkatakrishnan and D. J. Mavriplis, Implicit method for the computation of unsteady flows on unstructured grids, *ICASE Report No. 95–60*, NASA Langley Research Center, 1995.

[147] P. I. Crumpton and M. B. Giles, Implicit time accurate solutions on unstructured dynamic grids, *AIAA Paper 95–1671*, 1995.

[148] N. D. Melson, M. D. Sanetrik and H. L. Atkins, Time–accurate Navier–Stokes calculations with multigrid acceleration, in *Proceedings of the 6th Copper Mountain Conference on Multigrid Methods*, 423–439, 1993.

[149] L. Fezoui and B. Stoufflet, A class of implicit upwind schemes for Euler simulations on unstructured meshes, *Journal of Computational Physics 84*, 174–206, 1989.

[150] N. T. Frink, Assessment of an unstructured–grid method for predicting 3–D turbulent viscous flows, *AIAA Paper 96–0292*, 1996.

[151] A. Lerat, Multidimensional centered schemes of the Lax–Wendroff type, in M. Hafez and K. Oshima, editors, *Computational Fluid Dynamics Review 1995*, John Wiley, Chichester, 125–140, 1995.

[152] O. Hassan, K. Morgan and J. Peraire, An implicit finite element method for high speed flows, *International Journal for Numerical Methods in Engineering 32*, 185–205, 1991.

[153] D. Martin and R. Löhner, An implicit linelet–based solver for incompressible flows, *AIAA Paper 93–0668*, 1993.

[154] A. Jameson and S. Yoon, LU implicit schemes with multiple grids for the Euler equations, *AIAA Paper 86–0105*, 1986.

[155] K. Willcox and J. Peraire, Aeroelastic computations in the time domain using unstructured meshes, *International Journal for Numerical Methods in Engineering 40*, 2413–2431, 1997.

[156] S. Yoon and A. Jameson, An LU–SSOR scheme for the Euler and Navier–Stokes equations, *AIAA Paper 87–0600*, 1987.

[157] M. D. Tidriri, Krylov methods for compressible flows, *ICASE Report No. 95–48*, NASA Langley Research Center, 1995.

[158] V. Venkatakrishnan and D. J. Mavriplis, Implicit solvers for unstructured meshes, *ICASE Report No. 91–40*, NASA Langley Research Center, 1991.

[159] G.H. Golub and C. F. van Loan, *Matrix Computations, 2nd Edition*, The Johns Hopkins University Press, Baltimore, 1989.

[160] Y. Saad and M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM Journal of Scientific and Statistical Computing 7*, 856–869, 1986.

[161] E. Cuthill and J. McKee, Reducing the bandwidth of sparse symmetric matrices, in *Proceedings of the ACM National Conference*, 157–172, 1969.

[162] J. Cabello, K. Morgan, R. Löhner, H. Luo and J. Baum, An implicit solver for laminar compressible flows on unstructured grids, *AIAA Paper 95–0344*, 1995.

[163] R. Choquet, P. Leyland and T. Tefy, GMRES acceleration of iterative implicit finite element solvers for compressible Euler and Navier Stokes equations, *International Journal for Numerical Methods in Fluids 20*, 957–967, 1995.

[164] T. Hellström and J. Chomiak, Turbulent flow computations on adaptive unstructured grids using a coupled discrete Newton GMRES solver, *AIAA Paper 96–0419*, 1996.

[165] V. Venkatakrishnan, Parallel implicit unstructured grid Euler solvers, *AIAA Paper 94–0759*, 1994.

[166] S. Mittal and T. E. Tezduyar, Massively parallel finite element computation of incompressible flows involving fluid–body interactions, *Computer Methods in Applied Mechanics and Engineering 112*, 253–282, 1994.

[167] T. E. Tezduyar, S. K. Aliabadi, M. Behr and S. Mittal, Massively parallel finite element simulation of compressible and incompressible flows, *Computer Methods in Applied Mechanics and Engineering 119*, 157–177, 1994.

[168] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method. Fourth Edition. Volume 2*, McGraw–Hill, Maidenhead, 1991.

[169] A. Chorin, A numerical method for solving incompressible viscous flow problems, *Journal of Computational Physics 2*, 12–26, 1967.

[170] A. Rizzi and L. Eriksson, Computation of inviscid incompressible flow with rotation, *Journal of Fluid Mechanics 153*, 275–312, 1985.

[171] J. Farmer, L. Martinelli and A. Jameson, A fast multigrid method for solving incompressible hydrodynamic problems with free surfaces, *AIAA Paper 93–0767*, 1993.

[172] J. Dreyer, Finite volume solutions to the unsteady incompressible Euler equations on unstructured triangular meshes, *MS Thesis*, MAE Department, Princeton University, 1990.

[173] J. Peraire, K. Morgan and J. Peiró, The simulation of 3D incompressible flows using unstructured

grids, in D. A. Caughey and M. M. Hafez, editors, *Frontiers of Computational Fluid Dynamics 1994*, John Wiley, Chichester, 281–301, 1994

[174] E. Turkel, Preconditioned methods for solving the incompressible and low speed compressible equations, *Journal of Computational Physics 72*, 277–298, 1987.

[175] A. J. Chorin, Numerical solution of the Navier–Stokes equations, *Mathematics of Computation 23*, 341–354, 1968.

[176] J. Donéa, S. Giuliani, H. Laval and L. Quartapelle, Finite element solution of the unsteady Navier–Stokes equations by a fractional step method, *Computer Methods in Applied Mechanics and Engineering 30*, 53–73, 1982.

[177] O. C. Zienkiewicz, B. V. K. Satya Sai, K. Morgan and R. Codina, Split, characteristic based semi–implicit algorithm for laminar/turbulent flows, *International Journal for Numerical Methods in Fluids 23*, 787–809, 1996.

[178] K. T. Tang, W. R. Graham and J. Peraire, Active flow control using a reduced order model and optimum control, *AIAA Paper 96–1946*, 1996.

[179] *Report No. 463: Computational Methods for Aerodynamic Design (Inverse) and Optimization*, AGARD, Paris, 1989.

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

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

[182] J. Elliott and J. Peraire, Practical 3D aerodynamic design and optimization using unstructured meshes, *AIAA Paper 96–4170*, 1996.

[183] J. Reuther, A. Jameson, J. Farmer, L. Martinelli and D. Saunders, Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation, *AIAA Paper 96–0094*, 1996.

[184] A. Jameson and J. J. Alonso, Automatic aerodynamic optimization on distributed memory architectures, *AIAA Paper 96–0409*, 1996.