AEROELASTIC COMPUTATIONS IN THE TIME DOMAIN USING UNSTRUCTURED MESHES

K. Willcox and J. Peraire
Computational Aerospace Sciences Laboratory
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology

Abstract

A strategy for computing aeroelastic solutions is proposed. An implicit LU factorisation scheme for solving the time dependent Euler equations on unstructured triangular meshes is presented and coupled with a typical section aeroelastic wing model. Efficiency is improved by coupling the LU factorisation scheme with a GMRES algorithm. In this case the LU scheme plays the role of a preconditioner. The fluid and structural models are simultaneously integrated in time in a fully coupled manner. The response of structural sections in different flow regimes is determined and flutter boundaries are computed. In the transonic regime and beyond the region of linear stability, the section is found to exhibit limit cycle behaviour.

1 Introduction

In recent years, Computational Fluid Dynamics has reached a considerable level of maturity and has become an essential tool for aerodynamic analysis. With the latest developments in automatic unstructured mesh generation and efficient multigrid flow solution algorithms, it is now possible to routinely obtain steady Euler solutions about complete aircraft configurations [7] [13] and quickly assess the effect of design modifications.

Unsteady flows are of importance for a variety of practical problems, such as flutter or gust alleviation. In particular, aeroelastic modelling becomes essential in the transonic regime where nonlinear effects are important. A fully coupled solution of the nonlinear problem must be considered to predict phenomena such as transonic dip [5]. Kousen and Bendiksen found that the movements of shocks on the surface of an airfoil interact dynamically with the motion of the airfoil, a factor closely related to the process of limit cycle formation [9]. Obtaining an accurate time response of these shock motions is crucial in obtaining design criteria in the transonic regime, since small disturbance codes predict only the onset of growing oscillations and cannot model limit cycle behaviour.

The computation of time dependent flows requires significantly more resources than are required for steady state problems. If explicit timestepping methods are used, the allowable timestep is determined by stability considerations and is a function of the smallest grid size used to discretise the spatial domain. In some cases this timestep can be several orders of magnitude smaller than that required to obtain accurate solutions. When using unstructured meshes, it is in general very difficult to carefully control the size and shape of the elements generated, especially in three dimensions. The appearance of distorted elements, even if they occur in very small numbers, may severely limit the timestep size.

The alternative is to use an implicit formulation so that the timestep can be based on accuracy considerations only. In this case, it is necessary to devise a procedure for solving the resulting sparse equation system at each timestep. A possible approach is to discretise the equations in time using backward differencing formulae and to extend the multigrid techniques developed for the solution of steady flows to solve the resulting non-linear system of equations. This approach was proposed in [6] and has been extended to the solution of the two dimensional time dependent Navier-Stokes equations [2]. This method can also be used in the context of unstructured meshes and preliminary computations in three dimensions were reported in [12]. For unstructured meshes however, the automatic generation of coarse grids may prove difficult, especially if complex geometries are involved. This may present a limitation, since the ability to create very coarse grids is critical to the performance of the multigrid approach. An alternative strategy is to use a recursive algorithm for automatically generating coarser grids by agglomeration of fine grid cells [10] [16].

In this paper we investigate a different approach which utilises a single grid and which is effective for both steady and unsteady computations. The idea presented is an extension of the LU factorisation schemes proposed in [8] to unstructured meshes. For unstructured meshes, the algorithm presented is non-vectorisable but can be parallelised efficiently. The basic algorithm can be enhanced with the addition of a non-linear GMRES scheme as described in [17]. In this case the LU scheme plays the role of a preconditioner. The combined algorithm approaches a Newton method [3] when the number of search directions in the GMRES procedure is chosen to be sufficiently large.

2 Governing Equations

Consider a two dimensional time dependent control volume $\Omega(t)$, with boundary $\Gamma(t)$. The Euler equations governing the unsteady two-dimensional flow of an inviscid compressible fluid can be written in integral form as

$$\frac{\partial}{\partial t} \int_{\Omega} U dx dy + \oint_{\Gamma} (F n_x + G n_y) \, d\Gamma = 0 \tag{1}$$

where n_x and n_y are unit vectors pointing out of Ω, U is the unknown vector of conserved variables and F and G are the inviscid flux vectors given by

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ e \end{pmatrix} \quad F = \begin{pmatrix} \rho(u - x_t) \\ p + \rho u(u - x_t) \\ \rho v(u - x_t) \\ pu + e(u - x_t) \end{pmatrix} \quad G = \begin{pmatrix} \rho(v - y_t) \\ \rho u(v - y_t) \\ p + \rho v(v - y_t) \\ pv + e(v - y_t) \end{pmatrix} \tag{2}$$

Here ρ, u, v, p and e denote density, cartesian velocity components, pressure and total energy respectively. x_t and y_t are the speeds in the x and y directions with which the boundary $\Gamma(t)$ moves. Also, for an ideal gas the equation of state becomes

$$e = \frac{p}{\gamma - 1} + \frac{1}{2}\rho(u^2 + v^2) \tag{3}$$

These equations are discretised on an unstructured triangular grid covering the computational domain and approximations to the unknown vector U are sought at the vertices of that grid.

3 Unstructured Mesh Algorithm

3.1 Finite Volume Formulation

By considering equation (1) for each volume V_j consisting of all the triangles having vertex j as shown in figure 1, the following set of ordinary differential equations may be written

$$\frac{d}{dt}(V_j U_j) + \int_{\Gamma_j} (F n_x + G n_y) d\Gamma = 0 \tag{4}$$

where Γ_j is the boundary of V_j and U_j represents the average value of U over volume V_j. The above equation may be approximated as

$$\frac{d}{dt}(V_j U_j) + \sum_{i' \varepsilon \Gamma_j} \left[\frac{1}{2}(F_i + F_{i+1}) n_x^{i'} + \frac{1}{2}(G_i + G_{i+1}) n_y^{i'} \right] l^{i'} = 0 \tag{5}$$

where the summation extends over all the edges on Γ_j. In the above expression $n_x^{i'}$, $n_y^{i'}$ and $l^{i'}$ are the cartesian components of the outward unit normal and length of a generic edge i' connecting vertices i and $i+1$.

An edge based data structure can be used to express (5) in terms of a weighted summation of flux differences. Grouping together all the contributions from node i to node j we can define weights as

$$W_x^{ij} = \frac{1}{2}(n_x^{i'-1} l^{i'-1} + n_x^{i'} l^{i'}) \;, \quad W_y^{ij} = \frac{1}{2}(n_y^{i'-1} l^{i'-1} + n_y^{i'} l^{i'}) \tag{6}$$

when ij is an interior edge and

$$W_x^{ij} = \frac{1}{4} n_x^{ij} l^{ij} + \frac{1}{2} n_x^{i'} l^{i'} \;, \quad W_y^{ij} = \frac{1}{4} n_y^{ij} l^{ij} + \frac{1}{2} n_y^{i'} l^{i'}$$

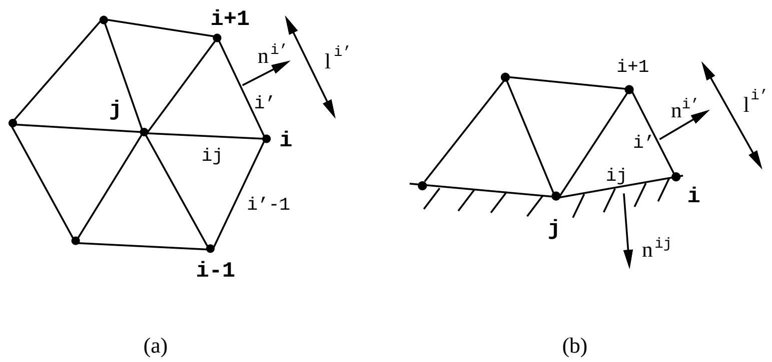

Figure 1: Control volume V_j associated to a generic node j of an unstructured grid (a) Interior node, (b) Boundary node.

$$\overline{W}_x^{ij} = \frac{1}{4} n_x^{ij} l^{ij} \ , \ \ \overline{W}_y^{ij} = \frac{1}{4} n_y^{ij} l^{ij} \tag{7}$$

when ij is a boundary edge. Noting that

$$\sum_{ij \varepsilon E_j} W_x^{ij} + 3 \sum_{ij \varepsilon B_j} \overline{W}_x^{ij} = 0 \ \ , \ \sum_{ij \varepsilon E_j} W_y^{ij} + 3 \sum_{ij \varepsilon B_j} \overline{W}_y^{ij} = 0 \tag{8}$$

where E_j denotes the set of edges meeting at point j and B_j the subset of these edges which also lie on the boundary on the domain Ω, the following equation can be written

$$\frac{d}{dt}(V_j U_j) + \sum_{ij \varepsilon E_j} \left[W_x^{ij}(F_i - F_j) + W_y^{ij}(G_i - G_j) \right]$$

$$+ \sum_{ij \varepsilon B_j} \left[\overline{W}_x^{ij}(F_i - F_j) + \overline{W}_y^{ij}(G_i - G_j) \right] = 0 \tag{9}$$

It is easily verified that the weights defined in (6) and (7) satisfy the following relations

$$W_x^{ij} = -W_x^{ji} \ , \ \ W_y^{ij} = -W_y^{ji}$$

$$\overline{W}_x^{ij} = \overline{W}_x^{ji} \ , \ \ \overline{W}_y^{ij} = \overline{W}_y^{ji} \tag{10}$$

By further defining the fluxes associated to an edge ij

$$H_i^{ij} = W_x^{ij} F_i + W_y^{ij} G_i = -H_i^{ji}$$

$$\overline{H}_i^{ij} = \overline{W}_x^{ij} F_i + \overline{W}_y^{ij} G_i = \overline{H}_i^{ji} \tag{11}$$

the semi-discrete scheme for a generic point j may be written as

$$\frac{d}{dt}(V_j U_j) + \sum_{ij \varepsilon E_j} (H_i^{ij} - H_j^{ij}) + \sum_{ij \varepsilon B_j} (\overline{H}_i^{ij} - \overline{H}_j^{ij}) = 0 \tag{12}$$

Finally, the following matrices are introduced

$$A_{ij} = W_x^{ij} \mathcal{A}(U_i, U_j) + W_y^{ij} \mathcal{B}(U_i, U_j)$$

$$\overline{A}_{ij} = \overline{W}_x^{ij} \mathcal{A}(U_i, U_j) + \overline{W}_y^{ij} \mathcal{B}(U_i, U_j) \tag{13}$$

where $\mathcal{A}$ and $\mathcal{B}$ are the Jacobian matrices $\frac{\partial F}{\partial U}$ and $\frac{\partial G}{\partial U}$ evaluated at the Roe average between U_i and U_j [14], and equation (12) is written as

$$\frac{d}{dt}(V_j U_j) + \sum_{ij \varepsilon E_j} A_{ij}(U_i - U_j) + \sum_{ij \varepsilon B_j} \overline{A}_{ij}(U_i - U_j) = 0 \tag{14}$$

3.2 Geometric Conservation Law

The scheme must also satisfy a geometric conservation law to avoid errors induced by the moving mesh and ensure that a uniform flow remains an exact solution of the numerical algorithm. We write the semi-discrete scheme (12) as

$$\frac{d}{dt}(V_j U_j) + \mathcal{F}_j(U, \mathbf{x}, \mathbf{x}_t) = 0 \tag{15}$$

where $\mathcal{F}_j$ represents all the flux summations applying to node j and is a function of the flow solution U, the weights W and $\overline{W}$ and thus the mesh position $\mathbf{x}$, and the mesh velocity $\mathbf{x}_t$. Integrating from time t^n to time t^{n+1} we obtain

$$V_j^{n+1} U_j^{n+1} - V_j^n U_j^n + \int_{t^n}^{t^{n+1}} \mathcal{F}_j \, dt = 0 \tag{16}$$

A required property of a conservative finite volume scheme is that a uniform flow is an exact solution of the scheme. If we consider the uniform flow solution $U_j^n = U_j^{n+1} = U$ for all nodes j, (16) reduces to

$$V_j^{n+1} - V_j^n = \int_{t^n}^{t^{n+1}} \int_{\Gamma_j} \mathbf{x}_t . \mathbf{n} \, d\Gamma \, dt \tag{17}$$

where spatial integration is now over Γ_j, the boundary of the volume V_j. Equation (17) defines the geometric conservation law which states that the change in area of each control volume over a given time interval must be equal to the area swept by the cell boundary during that interval. The left hand side of this equation can be computed exactly from the mesh updating scheme, so for the flow solver to satisfy the GCL, the integral $\int_{t^n}^{t^{n+1}} \int_{\Gamma_j} \mathbf{x}_t \, d\Gamma \, dt$ must also be exactly computed. For a two dimensional scheme, it can be shown [11] that the GCL will be satisfied if the weights W and $\overline{W}$ and velocities x_t and y_t are evaluated at the midpoint of the time interval where the positions and velocities of the grid are given by

$$\mathbf{x}^{n+\frac{1}{2}} = \frac{\mathbf{x}^n + \mathbf{x}^{n+1}}{2} \qquad \mathbf{x}_t^{n+\frac{1}{2}} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t} \tag{18}$$

3.3 Artificial Dissipation

In order to maintain a stable algorithm, the above scheme requires the addition of some numerical dissipation. A positive scheme [7] can be obtained by adding to the edge fluxes a diffusive term of the form $d^{ij} = |A_{ij}|\,(U_i - U_j)$. This gives

$$\frac{d}{dt}(V_j U_j) + \sum_{ij\varepsilon E_j} [A_{ij} - |A_{ij}|]\,(U_i - U_j) + \sum_{ij\varepsilon B_j} [\overline{A}_{ij} - |\overline{A}_{ij}|]\,(U_i - U_j) = 0 \tag{19}$$

where $|A| = X|\Lambda|X^{-1}$. X is the matrix whose columns are the right eigenvectors of A and $|\Lambda|$ is a diagonal matrix whose entries are the absolute values of the corresponding eigenvalues. This scheme is stable and can handle shock waves and discontinuities but introduces excessive dissipation for practical applications. A more accurate scheme can be obtained by introducing anti-diffusive terms in a controlled manner. This can be done by writing the diffusive fluxes as

$$d^{ij} = |A_{ij}| \left[\Delta U - \frac{1}{2}[L(\Delta U^+, \Delta U) + L(\Delta U, \Delta U^-)]\right] \tag{20}$$

where ΔU^+ and ΔU^- are differences associated to edge ij and computed as

$$\Delta U^+ = \left[\left(\frac{\partial U}{\partial x}\right)_j (x_i - x_j) + \left(\frac{\partial U}{\partial y}\right)_j (y_i - y_j)\right] \tag{21}$$

$$\Delta U^- = \left[\left(\frac{\partial U}{\partial x}\right)_i (x_i - x_j) + \left(\frac{\partial U}{\partial y}\right)_i (y_i - y_j)\right] \tag{22}$$

In (20) $L(u,v)$ represents a limited average of u and v. Several options are possible but the expression adopted for the results reported here is a differentiable function given in [15]

$$L(u,v) = \frac{1}{2}(u+v)\frac{2uv + \epsilon}{u^2 + v^2 + \epsilon} \tag{23}$$

where ϵ is a small number proportional to the cube of the length of the edge ij. Clearly the dissipative flux in (20) reduces to that in (19) when the the limiter function is zero. It is noted that if U is locally a linear function of x and y and the computed gradients are exact for linear data then the dissipative terms become zero and the scheme reduces to that of expression (14). A suitable expression for the calculation of the gradients at the nodes is obtained by employing an analogous algorithm to that used for the Euler fluxes

$$V_j \left(\frac{\partial U}{\partial x} \right)_j = \sum_{ij \varepsilon E_j} W_x^{ij} (U_i - U_j) + \sum_{ij \varepsilon B_j} \overline{W}_x^{ij} (U_i - U_j) \tag{24}$$

$$V_j \left(\frac{\partial U}{\partial y} \right)_j = \sum_{ij \varepsilon E_j} W_y^{ij} (U_i - U_j) + \sum_{ij \varepsilon B_j} \overline{W}_y^{ij} (U_i - U_j) \tag{25}$$

In practice best results are obtained if the limiting process is performed on the characteristic variables $\Delta S = X^{-1} \Delta U$. Then (20) is replaced by

$$d^{ij} = X |\Lambda| \left[\Delta S - \frac{1}{2} [L(\Delta S^+, \Delta S) + L(\Delta S^-, \Delta S)] \right] \tag{26}$$

with $\Delta S^+ = X^{-1} \Delta U^+$ and $\Delta S^- = X^{-1} \Delta U^-$.

4 Time Accurate Discretisation

4.1 LU Implicit Scheme

The scheme of equation (14) can be discretised in time using a backward Euler formula to give an unconditionally stable implicit algorithm. For second order accuracy in time, the fully discrete system is

$$\frac{3(V_j U_j)^{n+1} - 4(V_j U_j)^n + (V_j U_j)^{n-1}}{2\Delta t} + \tilde{R}_j^{n+1} = 0 \tag{27}$$

where the residual $\tilde{R}$ is given by

$$\tilde{R}_j^n = \sum_{ij \varepsilon E_j} \left[A_{ij} (U_i - U_j) - d^{ij} \right]^n + \sum_{ij \varepsilon B_j} \left[\overline{A}_{ij} (U_i - U_j) - \overline{d^{ij}} \right]^n \tag{28}$$

To solve (27) the following system is solved iteratively at every timestep until a steady state solution is obtained

$$V_j \frac{dU_j}{dt} + R_j = 0 \tag{29}$$

where the modified residual R is given by

$$R_j = \tilde{R}_j + \frac{3(V_j U_j) - 4(V_j U_j)^n + (V_j U_j)^{n-1}}{2\Delta t} \tag{30}$$

Equation (29) is discretised in time using a first order backward Euler formula to give

$$V_j^n \Delta U_j + \Delta \tau R_j^{p+1} = 0 \tag{31}$$

where $\Delta U_j = U_j^{p+1} - U_j^p$. The steady state solution is most efficiently attained by using local timestepping. The timestep is chosen independently for each node according to

$$\Delta\tau_j = \frac{2CV_j}{\sum_{ij\varepsilon E_j}\left(|\tilde{u}W_x^{ij} + \tilde{v}W_y^{ij}| + c \;\; l^{ij}\right) + \sum_{ij\varepsilon B_j}\left(3|\tilde{u}\overline{W}_x^{ij} + \tilde{v}\overline{W}_y^{ij}| + c \;\; \overline{l}^{ij}\right)} \tag{32}$$

where C is the CFL number which for an unconditionally stable scheme can be taken to be as large as desired, c is the speed of sound and $l^{ij} = [(W_x^{ij})^2 + (W_y^{ij})^2]^{\frac{1}{2}}$. The velocities $\tilde{u}$ and $\tilde{v}$ include the velocity of the moving mesh and are related to the local flow velocity by $\tilde{u} = u - x_t$, $\tilde{v} = v - y_t$. The nodal velocities and volume are evaluated at time level p.

A diagonally dominant linearised version of the scheme (31) can be written as

$$\Delta U_j + \frac{\Delta\tau_j}{V_j^n}\sum_{ij\varepsilon E_j}\tilde{A}_{ij}(\Delta U_i - \Delta U_j) + \frac{3\Delta\tau_j}{2\Delta t}\Delta U_j = -\frac{\Delta\tau_j}{V_j^n}R_j^n \tag{33}$$

where

$$\tilde{A} = A_{ij}^p - |A_{ij}^p|$$

when ij is an interior edge and

$$\tilde{A} = A_{ij}^p - |A_{ij}^p| + \overline{A}_{ij}^p - |\overline{A}_{ij}^p|$$

when ij is a boundary edge.

To avoid a direct solution of the sparse equation system that results from considering (33) for each node in the grid, an approximate factorisation can be derived. Consider a sweep direction defined by a vector (s_x, s_y), and renumber the nodes in the grid according to their s coordinate defined as

$$s_j = x_j s_x + y_j s_y \tag{34}$$

Further, the edges meeting at node j are subdivided into two sets E_j^+ and E_j^- according to

$$\begin{array}{lll} ij \;\; \varepsilon \;\; E_j^+ & \text{if } (x_i - x_j)s_x + (y_i - y_j)s_y \geq 0 \\ ij \;\; \varepsilon \;\; E_j^- & \text{if } (x_i - x_j)s_x + (y_i - y_j)s_y < 0 \end{array}$$

as shown in figure 2. Thus, (33) can be rewritten

$$\left[\mathcal{I} + \mathcal{C}_j\frac{\Delta\tau_j}{V_j^n}\sum_{ij\varepsilon E_j^+}\tilde{A}_{ij}\delta_{ij} + \mathcal{C}_j\frac{\Delta\tau_j}{V_j^n}\sum_{ij\varepsilon E_j^-}\tilde{A}_{ij}\delta_{ij}\right]\Delta U_j = -\mathcal{C}_j\frac{\Delta\tau_j}{V_j^n}R_j^n \tag{35}$$

where $\mathcal{I}$ is the 4x4 identity matrix and $\delta_{ij} U_j = U_i - U_j$. $\mathcal{C}_j$ is a constant for each node which depends on the CFL number C used in (32) to choose the timestep and is given by

$$\mathcal{C}_j = \frac{1}{1 + \frac{3\Delta\tau_j}{2\Delta t}} \tag{36}$$

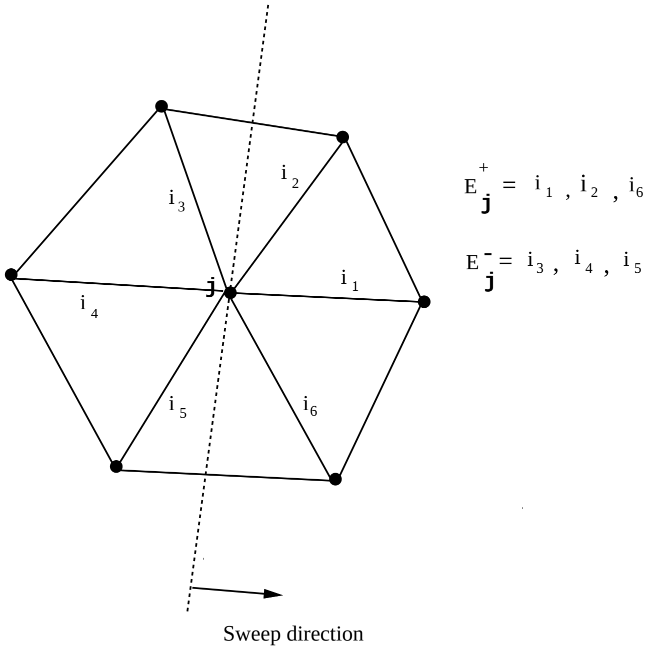

Figure 2: Splitting of sides according to chosen sweep direction.

An approximate LU factorisation analogous to that proposed in [8] for structured meshes is obtained by replacing (35) by

$$\left[\mathcal{I} + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^+} \tilde{A}_{ij} \delta_{ij} \right] \left[\mathcal{I} + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^-} \tilde{A}_{ij} \delta_{ij} \right] \Delta U_j = -\mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} R_j^p \tag{37}$$

It is easily verified that when the above equation is applied to every grid node, the first factor in the left hand side leads to a block lower triangular matrix, whereas the second factor leads to a block upper triangular matrix. Thus, (37) results in an equation system of the form

$$\left[\mathcal{L} + \mathcal{D}^+ \right] \left[\mathcal{U} + \mathcal{D}^- \right] \Delta U = R \tag{38}$$

where $\mathcal{L}$ and $\mathcal{U}$ are strictly block lower and upper triangular matrices, the $\mathcal{D}^+$ and $\mathcal{D}^-$ matrices contain the block diagonal terms, and ΔU and R are global vectors containing the increments of nodal unknowns and the right hand side terms of equation (37) respectively. The block entries of the $\mathcal{L}$ and $\mathcal{U}$ matrices are the edge matrices $\tilde{A}_{ij}$, and the block diagonal entries $\mathcal{D}_j^+$ and $\mathcal{D}_j^-$ are given by

$$\mathcal{D}_j^+ = \left[\mathcal{I} - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^+} \tilde{A}_{ij} \right] \qquad \mathcal{D}_j^- = \left[\mathcal{I} - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^-} \tilde{A}_{ij} \right] \tag{39}$$

The scheme (38) can be solved in two stages

$$\left[\mathcal{L} + \mathcal{D}^+\right] \Delta U^* = R$$

$$\left[\mathcal{U} + \mathcal{D}^-\right] \Delta U = \Delta U^* \tag{40}$$

each stage requiring the inversion of a 4x4 block matrix for each grid point. In order to efficiently implement this scheme in an unstructured mesh, it is convenient to create a sorted list containing all points and edges. This list is not unique but must satisfy two conditions:

i) The nodes in the list appear in the order implied by their s coordinate

ii) The edge connecting nodes i and j where $s_i < s_j$ is listed after node i and before node j as illustrated in figure 3.

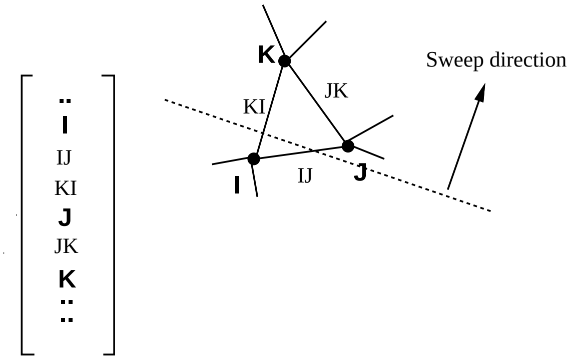

Figure 3: Structure of vector used to order sides and nodes.

The forward and back substitutions required to solve (40) can then be performed by looping over the elements of this vector in forward and reverse order. When a node is encountered a 4x4 matrix is inverted to obtain the increment of nodal unknowns. When an edge is found, the matrix $\tilde{A}_{ij}$ associated to that edge is computed and used to modify the equation for the corresponding downstream node.

4.2 LU-Gauss-Seidel Relaxation

The linearised implicit scheme (33) can be simulated with forward and backward Gauss-Seidel relaxation sweeps as discussed in [8, 18].

$$\Delta U_j^* + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^+} A_{ij}(\Delta U_i^* - \Delta U_j^*) - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^-} A_{ij}\Delta U_j^* + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} R_j^p = 0 \tag{41}$$

$$\Delta U_j + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^-} A_{ij}(\Delta U_i - \Delta U_j) + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^+} A_{ij}(\Delta U_i^* - \Delta U_j) + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} R_j^p = 0 \tag{42}$$

Thus, the forward sweep can be written

$$\left[\mathcal{I} - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^\pm} A_{ij}\right] \Delta U_j^* + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^+} A_{ij}\Delta U_i^* = -\mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} R_j^p$$

and the backward sweep can be rewritten, after subtracting (41) from (42), as

$$\left[\mathcal{I} - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^\pm} A_{ij}\right] \Delta U_j + \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^-} A_{ij}\Delta U_i = \left[\mathcal{I} - \mathcal{C}_j \frac{\Delta\tau_j}{V_j^n} \sum_{ij\varepsilon E_j^\pm} A_{ij}\right] \Delta U_j^*$$

Using the notation introduced in the last section, the combined scheme can be expressed as

$$(\mathcal{D} + \mathcal{L})\mathcal{D}^{-1}(\mathcal{D} + \mathcal{U})\Delta U = R \tag{43}$$

where $\mathcal{D} + \mathcal{I} = \mathcal{D}^+ + \mathcal{D}^-$. It is noted that (43) can be regarded as an LU implicit scheme and therefore can be implemented using the same data structure described in the previous section. In practice the scheme (43) performs better than the LU factorisation and furthermore has the advantage that only one block diagonal matrix needs to be inverted for each pair of forward and backward sweeps.

4.3 GMRES Algorithm

The convergence of the LU schemes can be further improved through use of a non-linear generalised minimal residual algorithm (GMRES) [17].

We begin by considering the steady state solution of (38) or (43) which satisfies

$$R(U) = 0 \tag{44}$$

Given an approximate solution U^p we then construct k orthonormal search directions, p_j, as follows

$$p_1 = \frac{R(U^p)}{||R(U^p)||} \tag{45}$$

where $||R||$ is the usual L_2 norm of R.
For $j = 1, 2, ...k - 1$ take

$$\tilde{p}_{j+1} = \overline{R}(U^p; p_j) - \sum_{i=1}^{j} b_{ij} p_i \tag{46}$$

and set

$$p_{j+1} = \frac{\tilde{p}_{j+1}}{||\tilde{p}_{j+1}||}$$

where

$$b_{ij} = \overline{R}(U^p; p_j).p_i$$

and $\overline{R}(U^p; p_j)$ denotes the directional derivative of R evaluated at U^p in the direction of p_j and is approximated as

$$\overline{R}(U^p; p_j) \sim \frac{R(U^p + \epsilon p_j) - R(U^p)}{\epsilon} \tag{47}$$

where ϵ is taken to be a small number proportional to $||U^p||$.

Once the k search directions are known, U^p is updated according to

$$U^{p+1} = U^p + \sum_{j=1}^{k} a_j p_j \tag{48}$$

where the coefficients a_j are evaluated by minimising $||R(U^{p+1})||^2$.

The performance of this algorithm is very dependent on the use of a suitable preconditioner. The LU factorisation algorithm described in the previous section can be used very effectively for that purpose. After performing one relaxation sweep with a method $\mathcal{M}$ such as that given by (38) or (43), an update to U^p will be obtained. This can be written symbolically as

$$U^{p+1} = \mathcal{M}(U^p) \tag{49}$$

Convergence is obtained when $U^{p+1} = U^p$, and hence we can use GMRES to obtain the solution of the preconditioned system

$$U - \mathcal{M}(U) = 0 \tag{50}$$

The nonlinear version of GMRES approximates Newton's method, and thus can diverge unless started sufficiently close to the solution. To avoid this problem, we follow [17] and introduce a damping factor λ into equation (50) and apply GMRES to

$$U^{p+1} - (1 - \lambda)\mathcal{M}(U^{p-1}) - \lambda\mathcal{M}(U^p) = 0 \tag{51}$$

The value of λ is initially taken to be close to 1 and is decreased to 0 as convergence is approached.

5 Aeroelastic Model

The wing section is modelled with two degrees of freedom as shown in figure 8. h is the plunging coordinate, which is positive down, while α is the angle of attack. The equations of motion can be written in non-dimensional form

$$[M]\{\ddot{q}\} + [K]\{q\} = \{F\} \tag{52}$$

where $[M]$ and $[K]$ are the non-dimensional mass and stiffness matrices given by

$$[M] = \begin{bmatrix} 1 & x_\alpha \\ x_\alpha & r_\alpha^2 \end{bmatrix} \qquad [K] = \begin{bmatrix} \left(\frac{\omega_h}{\omega_\alpha}\right)^2 & 0 \\ 0 & r_\alpha^2 \end{bmatrix} \tag{53}$$

Here ω_h and ω_α are the uncoupled natural frequencies of the section in plunge and pitch respectively. x_α and r_α^2 are section properties given by

$$x_\alpha = \frac{S_\alpha}{mb} \tag{54}$$

$$r_\alpha^2 = \frac{I}{mb} \tag{55}$$

Here S_α is the static inbalance and I is the section moment of inertia about the elastic axis. b is the airfoil semichord and m is the mass of the airfoil. The load and displacement vectors are

$$\{F\} = \frac{1}{\pi\mu k_c^2}\begin{Bmatrix} -C_l \\ 2C_m \end{Bmatrix} \qquad \{q\} = \begin{Bmatrix} \frac{h}{b} \\ \alpha \end{Bmatrix} \tag{56}$$

where k_c is the reduced frequency which is given by

$$k_c = \frac{1}{V_f\sqrt{\mu}} \tag{57}$$

Here V_f is the speed index which is defined as

$$V_f = \frac{U_\infty}{\omega_\alpha b\sqrt{\mu}} \tag{58}$$

C_l is the section lift coefficient and C_m is the moment coefficient about the elastic axis. μ is the airfoil mass ratio given by

$$\mu = \frac{m}{\pi\rho b^2} \tag{59}$$

(52) can be written as a system of two first order linear equations by letting $v = \dot{q}$. This leads to the following system of equations

$$\begin{Bmatrix} \dot{v} \\ \dot{q} \end{Bmatrix} + \begin{bmatrix} 0 & M^{-1}K \\ -I & 0 \end{bmatrix}\begin{Bmatrix} v \\ q \end{Bmatrix} = \begin{Bmatrix} M^{-1}F \\ 0 \end{Bmatrix} \tag{60}$$

The 4x4 matrix here is constant and needs only to be inverted once. At every timestep we can therefore solve (60) to find the new position of the airfoil. The fluid and structural equations are fully coupled and are advanced simultaneously in time.

6 Dynamic Mesh Algorithm

Once the new position of the airfoil has been determined from the structural equations, the grid must be deformed so that the fluid equations can be solved to find the resulting flow. This is done by modelling each edge in the grid as a spring with stiffness inversely proportional to the length of the edge. For an edge connecting nodes i and j

$$k_{ij} = \frac{1}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} \tag{61}$$

The position of points on the airfoil surface are prescribed by the calculated surface motion, while points on the outer farfield boundary are held fixed. At each timestep we must solve the static equilibrium equations which result from force summation for every node. Jacobi iterations of these equations are performed to obtain the displacements in the x and y directions. Generally, of the order of ten Jacobi iterations are required to obtain convergence.

$$\delta_{x_i}^{n+1} = \frac{\sum_j k_{ij} \delta_{x_j}^n}{\sum_j k_{ij}} \qquad \delta_{y_i}^{n+1} = \frac{\sum_j k_{ij} \delta_{y_j}^n}{\sum_j k_{ij}} \tag{62}$$

The new position of each node can then be determined

$$x_i^{n+1} = x_i^n + \delta_{x_i}^{n+1} \qquad y_i^{n+1} = y_i^n + \delta_{y_i}^{n+1} \tag{63}$$

7 Results

The first test case considered is that of inviscid transonic flow past a NACA 0012 airfoil. The freestream Mach number was 0.8 with an angle of attack of 1.25°. The farfield boundary was placed at a distance of 25 chord lengths from the airfoil, and a compressible vortex perturbation was used to improve the farfield boundary conditions. The solution was obtained using an explicit predictor/corrector scheme, the LU-Gauss-Seidel factorisation and the GMRES algorithm preconditioned with the LU-Gauss-Seidel scheme on an unstructured mesh containing 16634 nodes and 32908 elements. For the LU schemes, the sweep vector $\mathbf{s}$ was taken to be (1,1) and the timestep was chosen according to (32) with a CFL number of 10000. The value of ϵ used for the finite difference computation of the directional derivative in the GMRES algorithm was taken to be 10^{-8} times the modulus of the current solution vector U^p. Neither decreasing this parameter further nor altering the direction of the sweep vector was found to affect the performance of the algorithm significantly, however varying the number of search directions used in each cycle was found to alter the convergence properties. A steady state solution was computed by letting Δt in (27) become infinite and advancing the algorithm through one timestep to drive the residual to zero. The pressure contours of the steady state solution are plotted in figure 4.

Convergence histories for the three schemes are shown in figure 5. The residual is plotted as a function of the number of calls made to the solution routine. For the LU schemes, these calls require the inversion of 4 x 4 block matrices for each grid point, so the computational cost is greater than for the equivalent number of calls in the explicit predictor-corrector scheme. Two sets of results are presented for the GMRES scheme with LU-Gauss-Seidel as a preconditioner. In the first, the number of search directions is increased from 5 to a maximum of 30 in steps of 5 per cycle, while for the second we increase from 2 to 60 in steps of 2. The number of cycles required to reduce the maximum residual by six orders of magnitude for these cases was 46 and 34 respectively. The implicit schemes show a big improvement in convergence rate over the explicit predictor corrector scheme. GMRES can be seen to further improve the performance of the LU-Gauss-Seidel scheme.

The unsteady problem considered was the AGARD test case number 5 for the NACA 0012 airfoil [1]. A steady state solution is first computed then the airfoil is set in motion for unsteady calculations. The airfoil was forced to oscillate about its quarter chord point. The angle of attack is given by

$$\alpha = \alpha_m + \alpha_0 \sin(\omega t) \tag{64}$$

where ω is defined in terms of a reduced frequency

$$k_c = \frac{\omega c}{2V_\infty} \tag{65}$$

where c is the chord of the airfoil and V_∞ is the freestream velocity. For the case considered, $M_\infty = 0.755$, $\alpha_m = 0.016^\circ$, $\alpha_0 = 2.51^\circ$, $k = 0.0814$.

The GMRES algorithm with the LU-Gauss-Seidel scheme as a preconditioner was used to compute the solution. The grid used had 4258 nodes and 8320 triangular elements. First and second order accurate timestep schemes were considered. The number of timesteps per oscillation cycle was taken to be 36. Because the transient solution computed at each timestep is close to the previous solution, it is not necessary to use any damping in the GMRES algorithm. The resulting lift and moment coefficients for the second order time integration scheme are plotted in figure 6, along with the experimental data [1] and minor deviations are observed. The residuals are plotted in figure 7. The number of search directions was chosen to be 20, requiring approximately 12-15 GMRES cycles per timestep.

A fully coupled structural-fluid problem was considered next. This is the two-dimensional swept wing model (Case A of Isogai [5]). The structural parameters are $x_\alpha = 1.8, r_\alpha^2 = 3.48, \frac{\omega_h}{\omega_\alpha} = 1$ and $\mu = 60$ with the elastic axis positioned one chord length ahead of the leading edge. The steady state solution for flow about a NACA 64-A010 airfoil is first computed for a given Mach number. The airfoil is then forced sinusoidally in pitch about the elastic axis for three complete cycles at a frequency ω_f which was chosen in this case to be equal to ω_α. At the end of these forced cycles the system is allowed to respond to its self-induced loads. The response is computed for a number of different Mach numbers and different values of the speed index V_f. A noted difference in the characteristic behaviour of the airfoil response is observed between subsonic and transonic Mach numbers. The pitching and plunging responses of the airfoil are shown in figures 9 and 10 for a Mach number of 0.6. For this subsonic case, we see a clear transition between stable response at a speed index of 1.1 and unstable response at the only slightly higher value of 1.15. In the transonic regime however, nonlinear effects become important and the transition from stable to unstable response is not so sharp. Figure 11 shows the response obtained at a Mach number of 0.82 and a speed index of 0.71. For speed indices less than 0.7 the response is clearly stable, however there is a region above this where limit cycle behaviour is observed such as that shown in figure 11. The flutter boundary for the airfoil can be determined by finding the point at which unstable response occurs for several different Mach numbers. Figure 12 shows results from the present code and from references [2] and [4].

The results also demonstrate that for certain Mach numbers there is a phase difference between the pitching and plunging modes of response. Figure 13 shows these phase differences at the flutter point for a number of Mach numbers. The phase difference is largest for the subsonic case $M = 0.6$ and reduces to almost zero at a Mach number of 0.875. For the structural parameters considered, the non-dimensional natural frequencies of the section were calculated to be 0.71 and 5.34 with pivotal points at -3.87 and -0.13 semichords respectively. These points are measured aft of the leading edge, so that the first lies ahead of the elastic axis (postioned at $a =$ -2.0 semichords) while the second is just in front of the airfoil leading edge, but behind the elastic axis. Because the phase difference at $M = 0.875$ is almost zero, there is a pivotal point at

$$x_{pv} = a - \frac{h/b}{\alpha} \tag{66}$$

which in this case was calculated to be -3.96 semichords. This flutter mode is thus very close to the first natural mode which is pure pitching motion about the pivotal point $x_{pv} =$ -3.87 semichords.

Multiple flutter points are observed around Mach numbers of 0.875. Increase in V_f past the second flutter point once again results in a stable response. As figure 14 shows, flutter now occurs in the second mode of vibration with a higher frequency and in the antisymmetric mode. Because the airfoil is now responding with a much higher frequency, it was found necessary to decrease the size of the timestep in order to maintain accuracy.

8 Conclusions

The proposed schemes are shown to be a viable alternative for the solution of the time dependent Euler equations. The LU schemes are not vectorisable but can be parallelized efficiently. The fluid solver has been successfully coupled with a typical section aeroelastic model and used to predict airfoil response.

Finally, work is underway to extend this algorithm to three dimensions and viscous flows.

Acknowledgements

The authors would like to thank the Flight Test Information Systems Branch of the NASA Dryden Research Center for supporting this research under grant NAG2-4002 and in particular Dr K.K. Gupta for his continued interest and encouragement. Also thanks to Professor K. Morgan for his helpful comments and discussions.

References

[1] AGARD. Compendium of Unsteady Aerodynamic Measurements. Report No. 702, 1982.

[2] J. Alonso and A. Jameson. Fully-Implicit Time-Marching Aeroelastic Solutions. AIAA-94-0056, 1994.

[3] T. Barth and S. Linton. An Unstructured Mesh Newton Solver for Compressible Fluid Flow and Its Parallel Implementation. AIAA-95-0221, 1995.

[4] J. W. Edwards, R. Bennett, W. Whitlow Jr., and D. Seidel. Time-Marching Transonic Flutter Solutions Including Angle-of-Attack Effects. *Journal of Aircraft*, 20:899–906, November 1983.

[5] K. Isogai. Transonic Dip Mechanism of Flutter of a Sweptback Wing: Part II. *AIAA*, 19:1240–1242, September 1981.

[6] A. Jameson. Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows Past Airfoils and Wings. AIAA 91-1596, 1991.

[7] A. Jameson. Computational Algorithms for Aerodynamic Analysis and Design. *Applied Numerical Mathematics*, 13:383–422, 1992.

[8] A. Jameson and S. Yoon. LU Implicit Schemes with Multiple Grids for the Euler Equations. AIAA-86-0105, 1986.

[9] K. Kousen and O. Bendiksen. Nonlinear Aspects of the Transonic Aeroelastic Stability Problem. AIAA-88-2306, 1988.

[10] M. Lallemand, H. Steve, and A. Dervieux. Unstructured Multigridding by Volume Agglomeration: Current Status. *Computers and Fluids*, 21:397–433, 1992.

[11] M. Lesoinne and C. Farhat. Geometric Conservation Laws for Aeroelastic Computations using Unstructured Dynamic Meshes. AIAA-95-1709, 1995.

[12] K. Morgan, J. Peraire, and J. Peiro. Unstructured Grid Method for High Speed Compressible Flows. Int. Conference on Mathematics of Finite Elements and Applications, Academic Press, 1993.

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

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

[15] G. Van Albada, B. Van Leer, and W. Roberts. A Comparative Study of Computational Methods in Cosmic Gas Dynamics. *Astronomy and Astrophysics*, 108:76–85, 1982.

[16] V. Venkatakrishnan and D. Mavriplis. Agglomeration Multigrid for the 3D Euler Equations. AIAA-94-0069, 1994.

[17] L. Wigton, N. Yu, and D. Young. GMRES Acceleration of Computational Fluid Dynamics Codes. AIAA-85-1494, 1985.

[18] S. Yoon and A. Jameson. An LU-SSOR Scheme for the Euler and Navier-Stokes Equations. AIAA-87-0600, 1987.

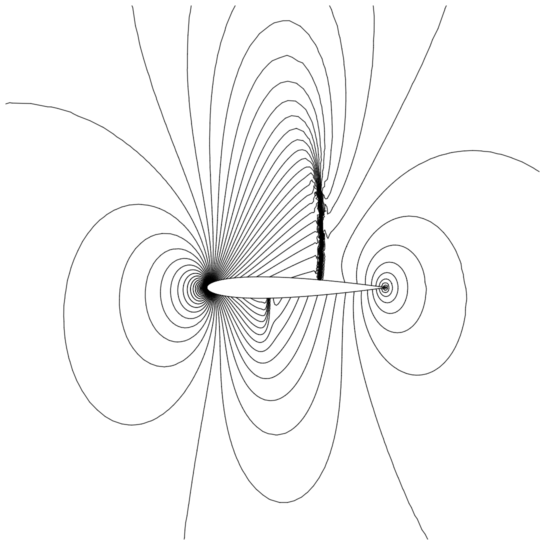

Figure 4: Pressure contours for steady inviscid transonic flow.

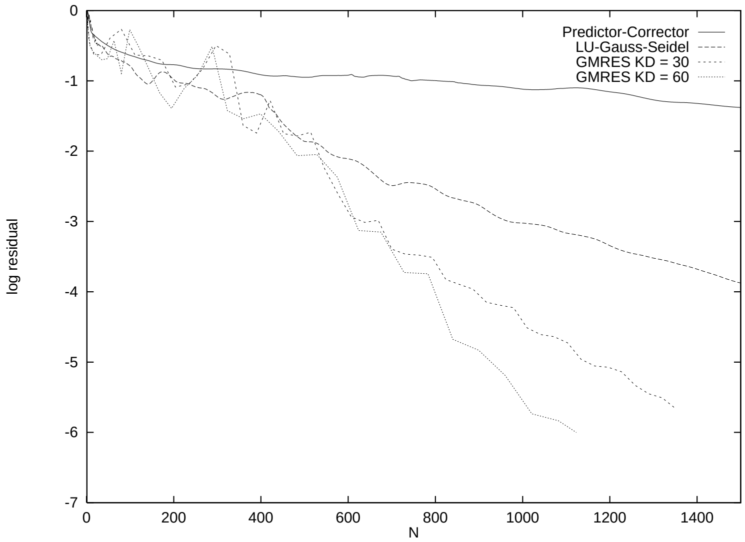

Figure 5: Residuals for steady inviscid transonic flow.

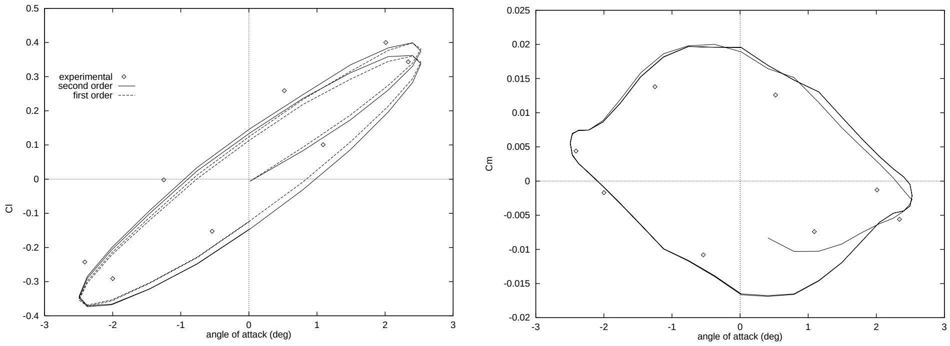

Figure 6: Lift coefficient and moment coefficient for unsteady inviscid transonic flow.

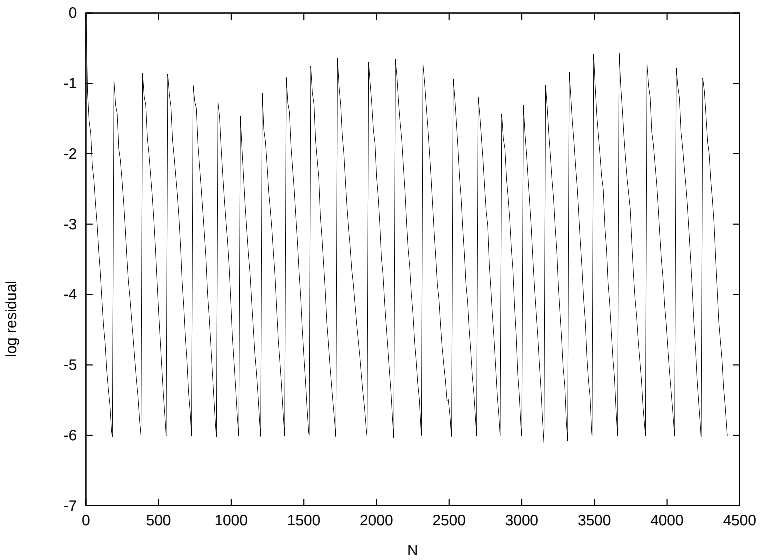

Figure 7: Residuals for unsteady inviscid transonic flow.

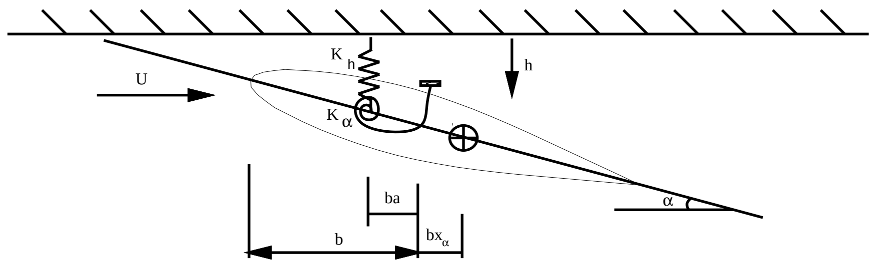

Figure 8: Typical section wing model geometry

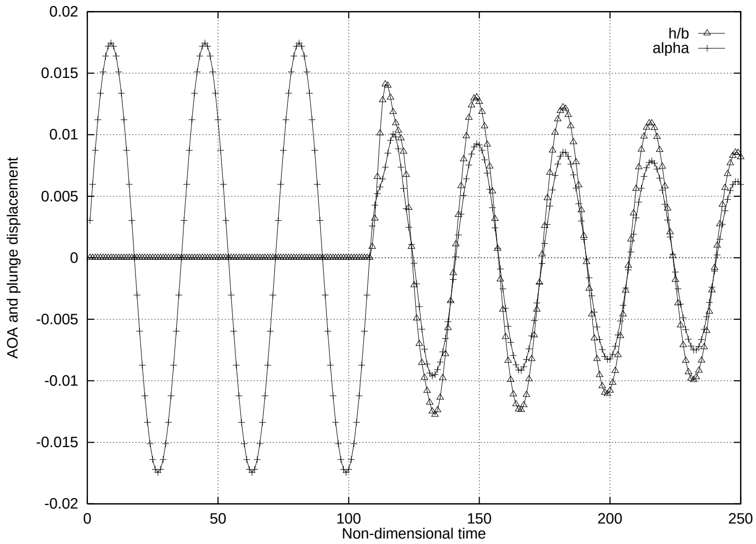

Figure 9: Damped Response. $M = 0.6$, $V_f = 1.1$, $\alpha_m = 0°$, $\alpha_o = 1°$, $\frac{\omega_f}{\omega_\alpha} = 1$

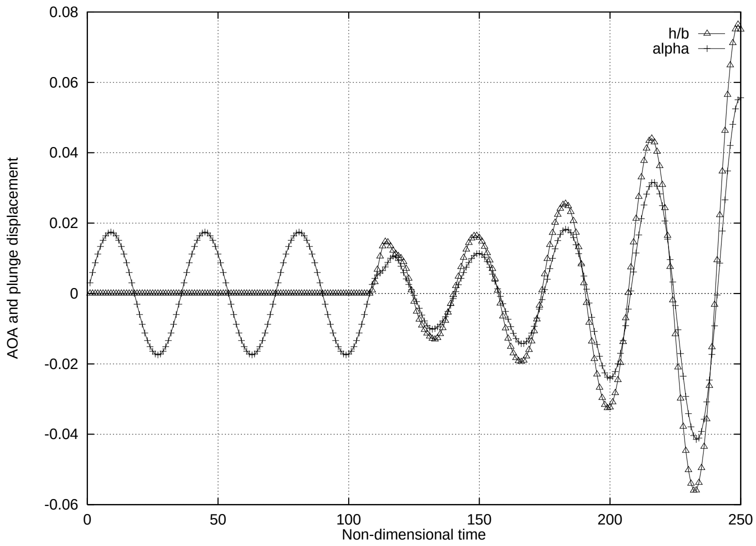

Figure 10: Unstable Response. $M = 0.6$, $V_f = 1.15$, $\alpha_m = 0°$, $\alpha_o = 1°$, $\frac{\omega_f}{\omega_\alpha} = 1$

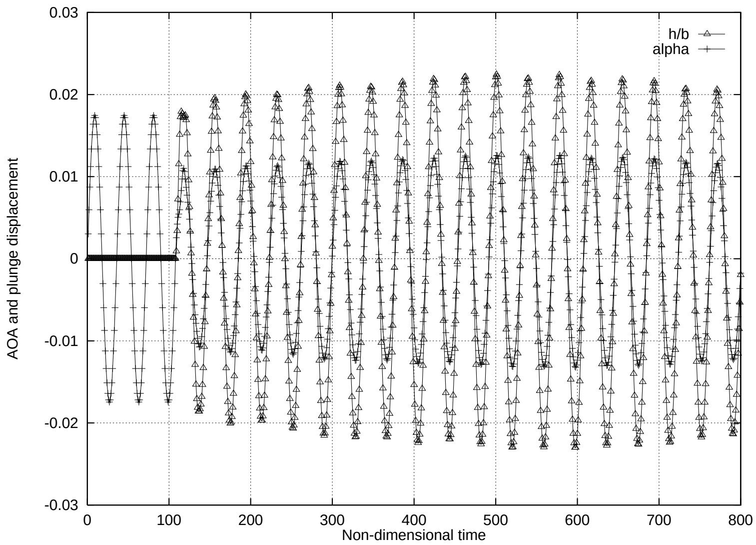

Figure 11: Limit Cycle Response. $M = 0.82$, $V_f = 0.71$, $\alpha_m = 0°$, $\alpha_o = 1°$, $\frac{\omega_f}{\omega_\alpha} = 1$

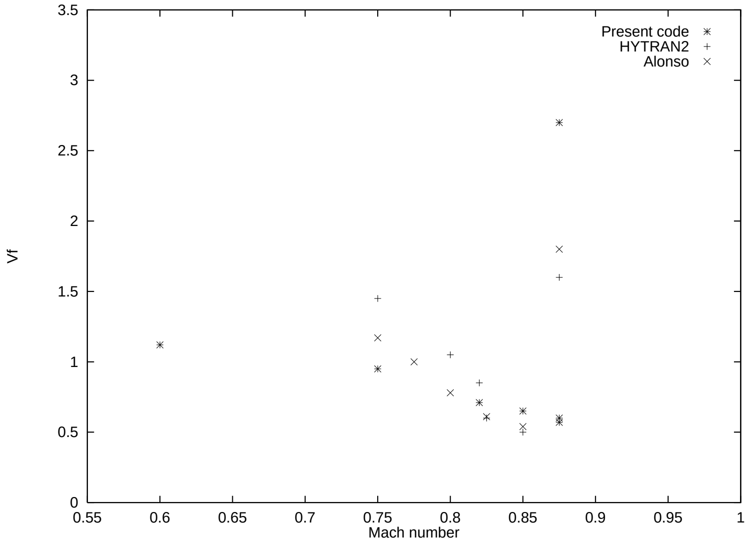

Figure 12: Flutter Boundary. $V_f = U_\infty / b\omega_\alpha \mu^{0.5}$

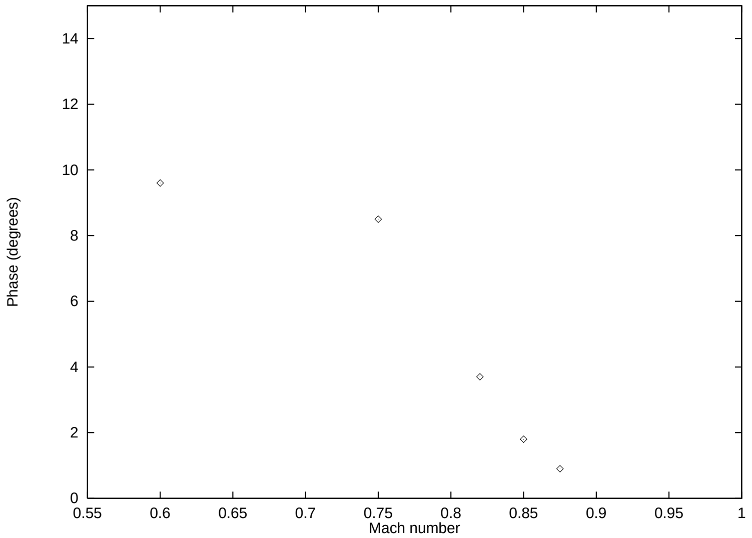

Figure 13: Flutter mode phase difference. $\mu = 60$

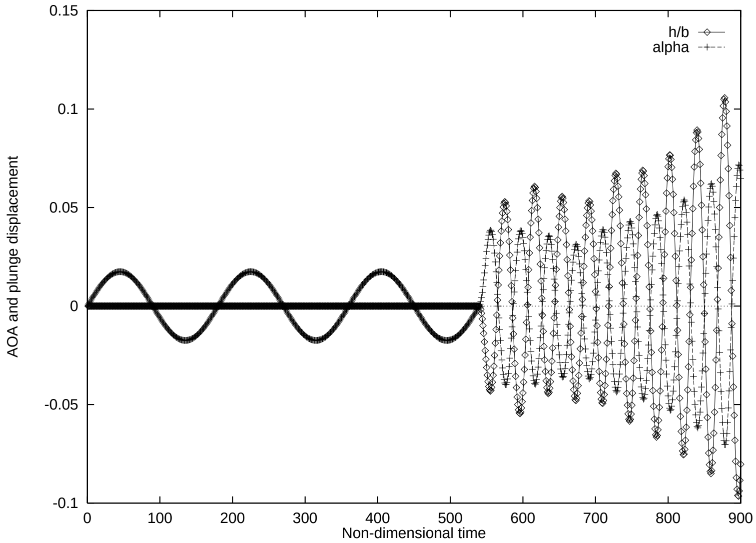

Figure 14: Second mode response. $M_\infty = 0.875$, $V_f = 2.6$, $\alpha_m = 0°$, $\alpha_o = 1°$, $\frac{\omega_f}{\omega_\alpha} = 1$