IMPLICIT METHOD FOR THE COMPUTATION OF UNSTEADY FLOWS ON UNSTRUCTURED GRIDS

V. Venkatakrishnan*
D. J. Mavriplis *
Institute for Computer Applications in Science and Engineering
MS 132C, NASA Langley Research Center
Hampton, VA 23681-0001

Abstract

An implicit method for the computation of unsteady flows on unstructured grids is presented. Following a finite difference approximation for the time derivative, the resulting nonlinear system of equations is solved at each time step by using an agglomeration multigrid procedure. The method allows for arbitrarily large time steps and is efficient in terms of computational effort and storage. Inviscid and viscous unsteady flows are computed to validate the procedure. The issue of the mass matrix which arises with vertex-centered finite volume schemes is addressed. The present formulation allows the mass matrix to be inverted indirectly. A mesh point movement and reconnection procedure is described that allows the grids to evolve with the motion of bodies. As an example of flow over bodies in relative motion, flow over a multi-element airfoil system undergoing deployment is computed.

*This research was supported by the National Aeronautics and Space Administration under the NASA contract No. NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681.

1 Introduction

Solution techniques for computing steady flows on unstructured grids have evolved to a high degree of sophistication. With explicit schemes, convergence to steady state is usually unacceptably slow, especially as the problem sizes and complexities grow. Therefore, either multigrid methods [20, 29] or implicit schemes [36, 44, 2, 4] are required to accelerate the convergence. On the other hand, solution techniques for dealing with unsteady flows have lagged behind. Explicit schemes, deemed to be too slow for obtaining steady state solutions, may be the schemes of choice for certain unsteady applications, when the time scales of interest are small, or more precisely, when they are comparable to the spatial scales. The grids should be clustered only in regions of interest; otherwise, the size of the explicit time step could become unnecessarily small. However, when dealing with many low frequency phenomena such as flutter, explicit schemes lead to large computing times. A time-accurate residual averaging formulation [18, 42], and temporal adaptation techniques [13], which enable different cells to take a varying number of local time steps to get to a particular time level, can be used to realize modest improvements in the performance of explicit methods, but the sizes of the time steps are still controlled by the spatial resolution. Therefore, it is desirable to develop a fully implicit method, where the time step is solely determined by the flow physics and is not limited by the cell sizes. Also, for many practical viscous flows, the time step restrictions imposed by small cells deep inside the boundary layer are excessively small. Since the boundary layer is quasi-steady, implicit methods that allow for larger time steps may be more suitable.

When an implicit scheme is used to compute unsteady flows, one has to drive the unsteady residual to zero (or at least to truncation error) at each time step. In the context of factored implicit schemes, this is usually done by employing inner iterations [31, 30, 7, 33]. It is the role of these inner iterations to eliminate errors, if any, due to factorization and linearization, and sometimes also errors arising from employing a lower order approximation on the implicit side. The number of inner iterations required may be large depending on the flow situation, the size of the time step employed and the extent of mismatch of the explicit and implicit operators. Jameson [11] has advocated the use of a full approximation storage multigrid procedure as a driver for a fully implicit scheme when using structured grids. The significant advantage of the approach when multigrid is used to solve the nonlinear problem is that it incurs no storage overheads plaguing traditional implicit schemes based on linearization. The method is therefore particularly attractive for unstructured grid computations in three dimensions. The method allows the time step to be determined solely based on flow physics. It has been used to compute two- and three-dimensional inviscid flows over airfoils and wings [11, 1] using structured grids. Vassberg [41] has applied this technique to compute flow solutions over oscillating airfoils using unstructured grids where a sequence of triangulations was generated by removing points from the fine grid triangulation.

Multigrid techniques have been successfully extended to deal with unstructured grids by generating a sequence of non-nested grids and using piecewise-linear transfer operators [20, 29]. An important development in the use of multigrid techniques for unstructured grids is the agglomeration multigrid algorithm [15, 37, 43] for the two- and three-dimensional Euler equations. This technique has since been extended to deal with viscous flows [14, 22]. The main advantage of the agglomeration multigrid algorithm is that it only requires the fine mesh to be generated; the coarse grids are automatically generated by fusing fine grid control volumes using an efficient graph-based algorithm, resulting in a fully nested sequence of coarse grid levels.

In this work, the agglomeration multigrid strategy is used to solve the nonlinear system of equations at each time step. The nested property of the agglomeration approach enables a straight-forward treatment of problems involving moving meshes. It is also shown that the mass matrix can be inverted indirectly during the multigrid process. The issue of the mass matrix is critically examined in both one and two dimensions. In order to allow the grids to conform to moving geometries, a technique is proposed and tested that attempts to maintain the validity and quality of the triangulation. Inviscid flow over a pitching airfoil and viscous flow over an impulsively started cylinder are computed and the results are compared with experiments and other computations. Finally, an exploratory computation is carried out that simulates the phenomenon of flap deployment.

2 Governing equations and discretization

The equations governing compressible fluid flow in integral form for a control volume $\mathcal{V}(t)$ with boundary $\mathcal{S}(t)$ are given by

$$\frac{\partial}{\partial t}\int_{\mathcal{V}(t)} W dv + \oint_{\mathcal{S}(t)} [F(W, \boldsymbol{n}, \boldsymbol{s}) - G(W, \nabla W, \boldsymbol{n})] da = 0, \tag{1}$$

$$\begin{aligned} W &= [\rho, \rho\boldsymbol{V}, \rho e]^T \\ F(W,\ \boldsymbol{n}, \boldsymbol{s}) &= \ (\boldsymbol{V} - \boldsymbol{s}).\boldsymbol{n}W, \\ G(W, \nabla W,\ \boldsymbol{n}) &= [0,\ \boldsymbol{t}, \boldsymbol{t}.\boldsymbol{V} - \boldsymbol{q}.\boldsymbol{n}]^T, \end{aligned}$$

where $\boldsymbol{t}$ and $\boldsymbol{q}$ are respectively, the stress and heat flux vectors. ρ is the density, $\boldsymbol{V}$ is the velocity vector with Cartesian components V_i, e is the specific total energy, $\boldsymbol{n}$ is the outward unit normal vector of the boundary $\mathcal{S}(t)$, and $\boldsymbol{s}$ is the velocity vector of the boundary. The equations are augmented by the equation of state, which for a perfect gas is

$$p = (\gamma - 1)(\rho e - \frac{\rho|\boldsymbol{V}|^2}{2}). \tag{2}$$

In the present scheme, the variables are stored at the vertices of a mesh composed of triangles. The control volumes are nonoverlapping polygons which surround the vertices of the mesh. The contour integrals in Eqn. (1) are replaced by discrete path integrals over the faces of the control volume which are computed using the trapezoidal rule. This technique can be shown to be equivalent to using a piecewise-linear finite-element discretization under certain conditions. For dissipative terms, a blend of Laplacian and biharmonic operators is employed [20]. The Laplacian term acts only in the vicinity of shocks and is inactive elsewhere, while the biharmonic term acts only in regions of smooth flow. Only the Laplacian dissipative term is used on the coarse grids.

After applying the finite volume procedure, the following system of coupled differential equations is obtained:

$$\frac{d}{dt}(VMW) + R(W) = 0 \tag{3}$$

Here W is the solution vector over the whole field, $R(W)$ is the residual vector approximating the second integral in Eqn. (1), V is the cell volume associated with the vertex and M is the mass matrix.

The mass matrix arises because the update indicated by the residual $R(W)$ should be made to the average value in the control volume. It thus relates the average value of a control volume associated with a vertex to the point values of the vertex and those of its immediate neighbors. This definition differs from the way the mass matrix is defined in finite element formulations, where the mass matrix arises naturally from requiring the residual of the discretized PDE to be orthogonal to a set of trial functions, with the solution expanded in a set of basis functions. If the dissipative terms are made proportional to the residuals, this definition of the mass matrix carries through, e.g. the Streamline Upwind Petrov Galerkin method [10]. For finite volume schemes employing a polynomial reconstruction procedure within a cell, we instead derive the mass matrix entries by computing the average of this polynomial over the control volume. The mass matrix M couples the system of ODE's in Eqn. (3). The effect is that even with an explicit scheme, one has to deal with the solution of a coupled linear system. A technique called "mass-lumping" [38], replaces the matrix M by the identity matrix. For second-order accurate cell-centered schemes, which employ the triangles as the control volumes and store the values at the centroids, mass-lumping does not compromise the accuracy, since the point value at the centroid matches the average value to second order. However, for cell-vertex schemes on nonuniform grids, the centroid of the control volume is not represented by the vertex in question. For time-accurate computations on such grids, mass lumping would appear to introduce locally a first order spatial error. This approximation is routinely adopted for unsteady flows as well, and does not appear to adversely affect the quality of the solutions obtained. Davis and Bendiksen [9] observed little discernible differences in the unsteady solutions when using the full and the lumped mass matrices. However, since they used an explicit scheme, the time steps were quite small and furthermore, the grids employed appeared to be fairly uniform. The technique employed to solve the mass matrix (a few Jacobi iterations) in [19, 9] is not efficient, especially when larger grids are used. However, Miller [24] and Wathen [45] have established mesh-independent bounds on the eigenvalues of the diagonally preconditioned mass matrix arising out of linear finite elements and have reported that a conjugate gradient method can be used to efficiently invert the diagonally preconditioned mass matrix in just a few iterations. When higher order spatial discretizations are employed, the mass matrix has to be reckoned with, even when using cell-centered discretizations. We note that the mass matrix can be avoided altogether if only cell averages are employed for the spatial discretization.

3 The implicit scheme

We first outline the implicit scheme as developed by Jameson [11] for cell-centered, structured grids, where mass lumping was used. Replacing the mass matrix in Eqn. (3) by the identity matrix and making a 3-point backward-difference approximation for the time derivative yields

$$\frac{3}{2\Delta t}V^{n+1}W^{n+1} - \frac{2}{\Delta t}V^{n}W^{n} + \frac{1}{2\Delta t}V^{n-1}W^{n-1}$$

$$+R(W^{n+1}) = 0. \tag{4}$$

When applied to a linear differential equation of the form,

$$\frac{dW}{dt} = \alpha W \tag{5}$$

this discretization is A-stable i.e., stable for all values of $\alpha\Delta t$ in the left-half of the complex plane [8]. Eqn. (4) is now treated as a steady state equation by introducing a pseudo-time variable t^*. The multigrid scheme then solves the following nonlinear system to steady state using local time steps Δt^*:

$$\frac{\partial VU}{\partial t^*} + R^*(U) = 0, \tag{6}$$

where U is the approximation to W^{n+1}. Here the *unsteady residual* $R^*(U)$ is defined as

$$R^*(U) = \frac{3}{2\Delta t}VU + R(U) - S(V^nW^n, V^{n-1}W^{n-1}) \tag{7}$$

with the source term

$$S(V^nW^n, V^{n-1}W^{n-1}) = \frac{2}{\Delta t}V^nW^n - \frac{1}{2\Delta t}V^{n-1}W^{n-1} \tag{8}$$

remaining fixed through the multigrid procedure.

A multi-stage Runge-Kutta scheme applied to solve Eqn. (6) performs the role of a smoother. A low-storage second order accurate m-stage Runge-Kutta scheme to advance U is given by

$$\begin{aligned} Q_0 &= U^l \\ &\ldots. \\ V^{n+1}Q_k &= V^{n+1}Q_0 - \alpha_k \Delta t^* R^*(Q_{k-1}) \\ &\ldots. \\ U^{l+1} &= Q_m \end{aligned} \tag{9}$$

Starting with $U^1 = W^n$, the sequence of iterates $U^l, l = 1, 2, 3....$ converges to W^{n+1}.

However, the way the scheme has been formulated has been observed by Arnone et al. [3] to be unstable for small physical time steps, Δt. This is counter-intuitive because when using a small Δt, the multigrid procedure should converge fast and ideally, in the limit of explicit time steps, the multigrid procedure should converge in just a few iterations. Melson et al. [23] showed that the problem is due an instability that arises when a small Δt is used. They modified the scheme to get rid of this instability. The source of the problem is that the unsteady residual $R^*(W)$ includes the term $\frac{3}{2\Delta t}VU$ and, is therefore, treated explicitly in the Runge-Kutta scheme. Their analysis showed that if this term were treated implicitly in the Runge-Kutta scheme, the stability region would grow as Δt is decreased. It is easy to treat the term implicitly since it is only a diagonal term. Splitting the residual $R^*(U)$ as

$$R^*(U) = \frac{3}{2\Delta t}VU + R(U) - S, \tag{10}$$

the Runge-Kutta scheme now becomes,

$$\begin{aligned}
Q_0 &= U^l \\
&\dots\dots \\
\left[I + \frac{3}{2\Delta t}\alpha_k \Delta t^*\right] V^{n+1} Q_k &= V^{n+1} Q_0 \\
&\quad - \alpha_k \Delta t^* \left[R(Q_{k-1}) - S\right] \\
&\dots\dots \\
U^{l+1} &= Q_m
\end{aligned} \tag{11}$$

With the modified scheme, Melson et al. [23] have shown that arbitrarily large or small Δt may be employed.

As in [11, 23], we employ a full approximation storage multigrid scheme. The source term is computed only on the fine grid and the coarse grid problems are driven by the fine grid residuals. For the generation of coarse grids we follow the agglomeration multigrid procedure. In this method, a sequence of coarse grids is generated using efficient graph-based algorithms. This method has certain advantages when dealing with rigidly moving or deforming meshes. Since the edges that comprise the coarse grid volumes are subsets of the fine grid control volume edges, when the grid moves rigidly or deforms, the projections of the control volume faces onto the coordinate directions are easily computed from those of the fine grid. Also, as long as no grid points are added or removed, and the triangulation remains valid, and the grid connectivity remains unchanged, the interpolation operators are unchanged. Multigrid schemes based on non-nested triangulations would require the recomputation of the interpolation operators when the grids deform. Even if the mesh connectivity changes, the agglomeration algorithm is efficient enough that it can be used to regenerate the coarse grids without incurring substantial overheads.

4 Treatment of the mass matrix

When employing a vertex-centered approximation, making a 3-point backward-difference approximation for the time derivative yields

$$\begin{aligned}
&\frac{3}{2\Delta t} V^{n+1} M^{n+1} W^{n+1} - \frac{2}{\Delta t} V^n M^n W^n \\
&+ \frac{1}{2\Delta t} V^{n-1} M^{n-1} W^{n-1} + R(W^{n+1}) = 0.
\end{aligned} \tag{12}$$

The multigrid scheme now solves the following system to steady state using local time steps Δt^*:

$$\frac{\partial V U}{\partial t^*} + R^*(U) = 0, \tag{13}$$

where U is the approximation to W^{n+1}, and $R^*(W)$ now includes the mass matrix terms. Notice that the first term $\frac{\partial}{\partial t^*} VU$ does not involve the mass matrix, thus uncoupling the system of equations. The explicit Runge-Kutta scheme can be applied exactly as before. The inversion of the mass matrix is thus accomplished indirectly during the multigrid procedure. However, the modified scheme of Melson et al. [23] poses a serious problem. Their modification would require the term $\frac{3}{2\Delta t}VMU$, which is no longer a diagonal term, to be treated implicitly. We have devised a modification that solves this problem which is detailed below. The implicit Runge-Kutta scheme that is stable for all Δt is given by

$$\begin{aligned} Q_0 &= U^l \\ &\ldots. \\ \left[I + \frac{3}{2\Delta t}\alpha_k \Delta t^* M^{n+1}\right] V^{n+1} Q_k &= V^{n+1} Q_0 - \\ &\quad \alpha_k \Delta t^* \left[R(Q_{k-1}) - S\right] \\ &\ldots. \\ U^{l+1} &= Q_m \end{aligned} \tag{14}$$

where the source term S is now given by

$$S = \frac{2}{\Delta t} V^n M^n W^n - \frac{1}{2\Delta t} V^{n-1} M^{n-1} W^{n-1} \tag{15}$$

If we simply replace the mass matrix M by the identity on the left hand side of Eqn. (14), we have observed that the instability at small time steps persists. In our modification, we first add and subtract $\frac{3}{2\Delta t}\alpha_k \Delta t^* M^{n+1} V^{n+1} Q_{k-1}$ on the right hand side of Eqn. (14) to obtain

$$\begin{aligned} \left[I + \frac{3}{2\Delta t}\alpha_k \Delta t^* M^{n+1}\right] V^{n+1} Q_k = V^{n+1} Q_0 - \\ \alpha_k \Delta t^* R^*(Q_{k-1}) + \frac{3}{2\Delta t}\alpha_k \Delta t^* M^{n+1} V^{n+1} Q_{k-1}, \end{aligned} \tag{16}$$

where use has been made of the equation

$$R^*(U) = \frac{3}{2\Delta t} VMU + R(U) - S. \tag{17}$$

Note that the same term $\frac{3}{2\Delta t}\alpha_k \Delta t^* MVQ$ appears on the left and the right hand sides of Eqn. (16), except that they are evaluated at the $k-1$ and k stages, and that $R^*(U)$ is being driven to zero. The mass matrix M can now be replaced by βI, where I is the identity matrix and β is a constant yielding the following equation:

$$\begin{aligned} \left[1 + \frac{3}{2\Delta t}\alpha_k \Delta t^* \beta\right] V^{n+1} Q_k = V^{n+1} Q_0 \\ -\alpha_k \Delta t^* R^*(Q_{k-1}) + \frac{3}{2\Delta t}\alpha_k \Delta t^* \beta V^{n+1} Q_{k-1} \end{aligned} \tag{18}$$

The method can always be stabilized by increasing β and is akin to using a damped Jacobi method. The Runge-Kutta scheme no longer requires a matrix inversion. For small time steps of the order permitted by the explicit scheme, we find that the choice of $\beta = 2$ stabilizes the scheme.

5 Mesh point movement strategies

In order to be able to perform unsteady flow simulations over moving geometries, a body-conforming mesh has to be regenerated either globally at each time step, or the existing grid can be allowed to deform. The former option is expensive, especially in three dimensions. However, Baum et al. [6] have proposed some simplifying strategies. Whenever the body movement becomes too severe, they regenerate a coarse mesh either locally or globally. This is followed by the use of adaptive h-refinement in order to create a fine mesh. In the present work, we only investigate mesh point movement strategies. Tension spring analogy [5, 28, 35] or other physical analogies, such as incompressible flow [12], are typically used to move the mesh points. In the former case, the distribution of the spring stiffnesses is crucial. Since the techniques try to maintain the connectivity of the grid at all costs, the grid lines may cross resulting in invalid triangulations. Nevertheless, for many simple configurations, such as isolated airfoils, no crossover occurs and the spring analogy has been demonstrated to work well e.g., [5]. For such cases however, the use of exponentially varying scaling factors [9] or rigid body motion are simpler. The real challenge for any mesh point movement strategy is when relative motion is present between bodies in close proximity and when fine grids are employed. Other possibilities for grid movement are methods in use in the moving finite element method [25], where evolution equations are derived for grid point motion from the governing equations.

In our approach, we use the tension spring analogy to allow the grid points to react to the motion of the geometries. The following linear system of equations is solved by a Jacobi method:

$$\sum_j K_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j) = S_{ij}, \tag{19}$$

where the source term S_{ij} is computed so as to maintain the initial grid in the absence of any displacements. The spring stiffness K_{ij} is taken to be l_{ij}^{-p} where l_{ij} is the length of the edge joining nodes i and j. Over a number of applications, we have found the value of $p = 2$ to work well. When the boundary motion is prescribed, this method does not guarantee that the grid lines will not cross. The next improvement is to make the spring system nonlinear i.e., the boundary motion is decomposed into smaller steps and the procedure is repeated at every step. When relative motion is present, this results in excessive skewing of grid lines and eventually the lines do cross. Thus some kind of reconnection procedure is unavoidable. We choose to swap the edges by the Delaunay criterion. However, even though the grid remains valid, the resulting point distribution is typically unsatisfactory. Therefore a presmoothing procedure is used to smooth the distribution of points. This involves using a Jacobi method on the following system:

$$(I + \epsilon N_i)\boldsymbol{x}_i^{new} = \boldsymbol{x}_i^{old} + \epsilon \sum_j \boldsymbol{x}_j{}^{new}, \tag{20}$$

where N_i is the degree of node i. Typically we find that of the order of 100–200 steps are required to solve Eqn. (19), while only about 3–4 iterations of Eqn. (20) are sufficient. The number of Jacobi iterations to solve Eqn. (19) may appear to be large, but this is mainly due to the large displacements arising from using large time steps permitted by the implicit scheme. The grid motion procedure described above also has applications in design optimization, where the surface geometry changes during the design cycle.

For the implicit flow solver, the coarse grids are rederived by using the agglomeration algorithm whenever the grid connectivity changes. An incremental agglomeration algorithm is possible that operates only on the affected region, but this is not currently done. Typically, for the time steps used with the implicit scheme, the grid reconnection and reagglomeration consume about a third of the time required for the flow solver, mainly on account of the swapping procedure not being vectorized.

The grid movement terms need to be discretized carefully so that freestream is preserved. In other words, simply moving the grid through the domain should not change the freestream solution. The Geometric Conservation Law (GCL) [39, 47, 27] formalizes this concept. It can be derived from the continuity equation in Eqn. (1) by first assuming the control volumes to be the simplices themselves. Assuming a uniform velocity field and a constant density field, we obtain

$$\frac{\partial \mathcal{V}}{\partial t} + \oint_{\mathcal{S}(t)} [\boldsymbol{V} - \boldsymbol{s}].\boldsymbol{n}\, da = 0, \tag{21}$$

where $\boldsymbol{V}$ is the velocity field and $\boldsymbol{s}$ is the velocity of the boundary $\mathcal{S}(t)$. Since $\boldsymbol{V}$ is constant and the control volume is assumed to be closed at all times so that $\oint_{\mathcal{S}(t)} \boldsymbol{n}\ da = 0$, the equation becomes

$$\frac{\partial \mathcal{V}}{\partial t} - \oint_{\mathcal{S}(t)} \boldsymbol{s}.\boldsymbol{n}\, da = 0. \tag{22}$$

The discrete form of this equation should hold at all time steps and for all the simplices and is called the GCL. Using a forward Euler approximation for the time derivative, we obtain

$$\begin{aligned} \mathcal{V}_I^{n+1} - \mathcal{V}_I^n &= \int_t^{t+\Delta t} \oint_{\mathcal{S}_I(t)} \boldsymbol{s}.\boldsymbol{n}\, da\ dt \\ &= \sum_j \int_t^{t+\Delta t} \int_{\Omega_{I,j}(t)} \boldsymbol{s}.\boldsymbol{n}\, da\ dt, \end{aligned} \tag{23}$$

where $\mathcal{S}_I(t) = \sum_j \Omega_{I,j}(t)$ is the surface enclosing the volume $\mathcal{V}_I(t)$ of simplex I. The term inside the summation represents the volume swept out by the boundary $\Omega_{I,j}$ as the grid points forming that segment move. If the grid points are allowed to move arbitrarily, the GCL enables the edge velocity $\boldsymbol{s}$ and the grid normal $\boldsymbol{n}$ to be determined. Since simplices are convex, the volumes $\mathcal{V}^n, \mathcal{V}^{n+1}$ are uniquely determined by the positions of the points at time levels n and $n+1$. In two dimensions, it can be shown [47] that the change in volume can be expressed as

$$\mathcal{V}_I^{n+1} - \mathcal{V}_I^n = \Delta t \sum_j \boldsymbol{s}_j^{n+1/2}.\boldsymbol{N}_j^{n+1/2}, \tag{24}$$

where the summation is over the edges forming the triangle I. Here the edge velocity $\boldsymbol{s}_j^{n+1/2}$ is given by the average of the velocities of the vertices connected by the edge j, and $\boldsymbol{N}_j$, the normal scaled by the edge length, is given by the average of the values at the old and new time level:

$$\boldsymbol{N}_j^{n+1/2} = 0.5(\boldsymbol{N}_j^n + \boldsymbol{N}_j^{n+1}). \tag{25}$$

The velocity is assumed to be constant within a time step. Therefore, the velocity of vertex i is given by

$$\boldsymbol{s}_i^{n+1/2} = \frac{\boldsymbol{X}_i^{n+1} - \boldsymbol{X}_i^n}{\Delta t}, \tag{26}$$

where $\boldsymbol{X}$ is the position vector of the vertex.

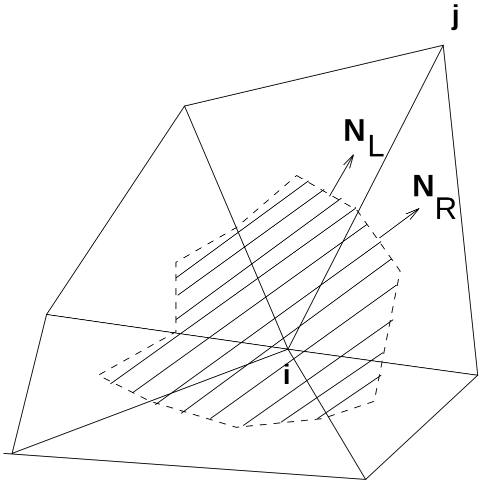


Figure 1: Control volume for vertex i and edge normals.

For a vertex-centered scheme, such as the one used in the present work, the control volumes are formed by the median dual. We closely follow the excellent development of Nkonga and Guillard [27], who have presented the steps involved in properly discretizing the flow equations in three dimensions so as to obey the GCL for a two-level scheme in time. The control volume edges are delimited by segments of the medians of the triangles (Figure 1). Inspecting one edge of the triangle, we notice that the control volume edge is comprised of two, generally non-collinear median segments. The results from Eqns.(24, 25) can be used to express the change in the volume of any polygon as a summation over the edges comprising the polygon of the volumes swept out. We obtain the following expression for the change in V_i, the volume associated with vertex i:

$$V_i^{n+1} - V_i^n = \Delta t \sum_j \boldsymbol{N}_{j,L}^{n+1/2}.\boldsymbol{s}_{j,L}^{n+1/2} + \boldsymbol{N}_{j,R}^{n+1/2}.\boldsymbol{s}_{j,R}^{n+1/2}, \tag{27}$$

where the summation is over the edges meeting at vertex i. Also, $\boldsymbol{N}_{j,L}^{n+1/2}$ and $\boldsymbol{N}_{j,R}^{n+1/2}$, the normals scaled by the lengths to the left and right of the edge, are obtained as averages of the values at n and $n+1$ time levels. The edge velocities, $\boldsymbol{s}_{j,L}^{n+1/2}$ and $\boldsymbol{s}_{j,R}^{n+1/2}$, are computed as the averages of the velocity of the midpoint of the edge and that of the left or right centroid, respectively.

In a vertex-centered finite-volume setting, the discretization of the governing equations (Eqn. (1)) for a two-level explicit or implicit scheme that obeys the GCL is performed as follows:

$$\begin{aligned}\frac{(MVW)^{n+1} - (MVW)^n}{\Delta t} &+ \sum_j W^\eta \left[\boldsymbol{N}_{j,L}^{n+1/2}.(\boldsymbol{V}^\eta - \boldsymbol{s}_{j,L}^{n+1/2}) + \boldsymbol{N}_{j,R}^{n+1/2}.(\boldsymbol{V}^\eta - \boldsymbol{s}_{j,R}^{n+1/2})\right] \\ &+ g_j(W^\eta, \boldsymbol{N}_j^{n+1/2}) = 0, \end{aligned} \tag{28}$$

where $\eta = n$ for forward and $\eta = n+1$ for backward Euler methods, and W is the average of the values at the vertices connected by the edge j. Also, g_j represents the discretization of the last term in Eqn. (1) and makes use of the scaled normal for the edge, $\boldsymbol{N}_j^{n+1/2}$, defined as

$$\boldsymbol{N}_j^{n+1/2} = \boldsymbol{N}_{j,L}^{n+1/2} + \boldsymbol{N}_{j,R}^{n+1/2}. \tag{29}$$

When a three point backward difference approximation is used for the time derivative, the spatial discretization is performed as follows:

$$\begin{aligned} & \frac{3(MVW)^{n+1} - 4(MVW)^n + (MVW)^{n-1}}{2\Delta t} \\ + & \sum_j W^{n+1}\{\frac{3}{2}[\boldsymbol{N}_{j,L}^{n+1/2}.(\boldsymbol{V}^{n+1} - \boldsymbol{s}_{j,L}^{n+1/2}) + \boldsymbol{N}_{j,R}^{n+1/2}.(\boldsymbol{V}^{n+1} - \boldsymbol{s}_{j,R}^{n+1/2})] \\ - & \frac{1}{2}[\boldsymbol{N}_{j,L}^{n-1/2}.(\boldsymbol{V}^{n+1} - \boldsymbol{s}_{j,L}^{n-1/2}) + \boldsymbol{N}_{j,R}^{n-1/2}.(\boldsymbol{V}^{n+1} - \boldsymbol{s}_{j,R}^{n-1/2})]\} \\ + & \frac{3}{2}g_j(W^{n+1}, \boldsymbol{N}_j^{n+1/2}) - \frac{1}{2}g_j(W^{n+1}, \boldsymbol{N}_j^{n-1/2}) = 0. \end{aligned} \tag{30}$$

It is easy to see that this formulation satisfies the GCL, and that it reduces to the standard three-point formula (Eqn. (12)) in the absence of any grid motion. We mention here that for fixed grid applications it is not necessary to compute $\boldsymbol{N}_{j,L}$ and $\boldsymbol{N}_{j,R}$ separately. Instead only the sum of the two is required, which can be computed easily from the the centroidal coordinates. On the other hand, for moving grid problems, the $\boldsymbol{N}_{j,L}$ and $\boldsymbol{N}_{j,R}$ need to be computed separately for use in Eqns. (28,30).

The spatial discretization given above only deals with the convective and diffusive fluxes. The dissipative fluxes are computed in the usual fashion using the scaled normal given by Eqn. (29) for the two-level explicit or implicit scheme. For the three-point scheme, the scaled normal is defined as follows:

$$\overline{\boldsymbol{N}}_j = 3/2\left[\boldsymbol{N}_{j,L}^{n+1/2} + \boldsymbol{N}_{j,R}^{n+1/2}\right] - 1/2\left[\boldsymbol{N}_{j,L}^{n-1/2} + \boldsymbol{N}_{j,R}^{n-1/2}\right]. \tag{31}$$

The discretization of the dissipative fluxes has no implications for the GCL as these fluxes vanish for a uniform field.

We have verified that the formulation given above preserves the freestream conditions to machine precision. With other treatments, such as simply modifying the fluxes at the vertices by using nodal velocities, freestream conditions are preserved to much less precision, which could be detrimental in some applications.

Finally, we address the issue of boundary conditions for inviscid flows. The control volume for a boundary vertex is illustrated in Figure 2. The summation over the edges in Eqns. (28,30) is augmented with the contributions from the the boundary edges that delimit the control volume. For the two-level scheme, the following terms are added to Eqn. (28):

$$\begin{aligned} & W_L^\eta[\boldsymbol{N}_{b,L}^{n+1/2}.(\boldsymbol{V}_L^\eta - \boldsymbol{s}_{b,L}^{n+1/2})] + W_R^\eta[\boldsymbol{N}_{b,R}^{n+1/2}.(\boldsymbol{V}_R^\eta - \boldsymbol{s}_{b,R}^{n+1/2})] \\ + & g(W_L^\eta, \boldsymbol{N}_L^{n+1/2}) + g(W_R^\eta, \boldsymbol{N}_R^{n+1/2}), \end{aligned} \tag{32}$$

where W_L and W_R are the values of W on the left and right side of vertex i, $\boldsymbol{N}_{b,R}$ and $\boldsymbol{N}_{b,L}$ are the outward normals to the boundary edges of the control volumes, and $\boldsymbol{s}_{b,L}$ and $\boldsymbol{s}_{b,R}$

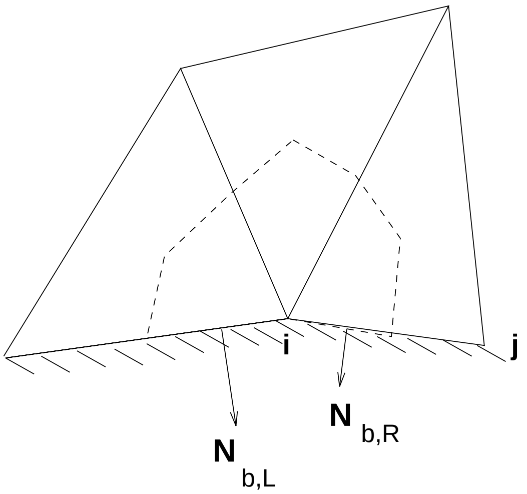


Figure 2: Control volume for vertex boundary vertex i and boundary edge normals.

are the grid velocities associated with these edges. For an inviscid all, since the $[\boldsymbol{V} - \boldsymbol{s}] = 0$, only the last two terms are retained. Likewise for the three-point scheme (Eqn. (30)), the additional terms are

$$\begin{aligned} W_L^{n+1} \quad [\quad & \frac{3}{2}\boldsymbol{N}_{b,L}^{n+1/2}.(\boldsymbol{V}_L^{n+1} - \boldsymbol{s}_{b,L}^{n+1/2}) - \frac{1}{2}\boldsymbol{N}_{b,L}^{n-1/2}.(\boldsymbol{V}_L^{n+1} - \boldsymbol{s}_{b,L}^{n-1/2}) \\ + \quad & \frac{3}{2}\boldsymbol{N}_{b,R}^{n+1/2}.(\boldsymbol{V}_R^{n+1} - \boldsymbol{s}_{b,R}^{n+1/2}) - \frac{1}{2}\boldsymbol{N}_{b,R}^{n-1/2}.(\boldsymbol{V}_R^{n+1} - \boldsymbol{s}_{b,R}^{n-1/2})] \\ + \quad & \frac{3}{2}[g(W_L^{n+1}, \boldsymbol{N}_{b,L}^{n+1/2}) + g(W_L^{n+1}, \boldsymbol{N}_{b,L}^{n+1/2})] \\ - \quad & \frac{1}{2}[g(W_R^{n+1}, \boldsymbol{N}_{b,R}^{n-1/2}) + g(W_R^{n+1}, \boldsymbol{N}_{b,R}^{n-1/2})]. \end{aligned} \tag{33}$$

For an inviscid wall, only the last four terms are retained, implying the following boundary condition:

$$\boldsymbol{V}^{n+1}.\left[\frac{3}{2}\boldsymbol{N}_b^{n+1/2} - \frac{1}{2}\boldsymbol{N}_b^{n-1/2}\right] = \frac{3}{2}\boldsymbol{N}_b^{n+1/2}.\boldsymbol{s}_b^{n+1/2} - \frac{1}{2}\boldsymbol{N}_b^{n-1/2}.\boldsymbol{s}_b^{n-1/2}. \tag{34}$$

6 Mesh restructuring and interpolation

The procedure for mesh restructuring was discussed earlier. This involved a mechanism for moving grid points in response to the motion of the boundary points, evolving the solution with the grid movement terms present, and swapping the edges of the triangulation to improve grid quality. When using the three point difference formula (Eqn.(12)), the swapping of edges at a particular time step has to be done in such a manner that the triangulation is also valid (i.e., no crossing of edges) at the previous time step. This additional criterion can easily be incorporated in the swapping procedure. Using the trapezoidal rule would circumvent this step and still ensure second order accuracy in time. However it is unattractive, because of the instabilities that occur as large time step sizes are used [8].

When the swapping of edges occurs, the solution, which is stored at the vertices of the triangulation, needs to be modified. For capturing weak solutions of hyperbolic conservation laws, conservation in time is a requirement. If conservation were not an issue, the solution need not be modified since it merely changes from one piecewise-linear representation to another and the two are acceptable second-order accurate solutions. Note that the swapping takes place at a fixed time step after the solution has already been evolved up to that time level. It is possible to carry out another iterative loop so that the unsteady residual is driven to zero in the new configuration. This would double the work done at each time step. Instead, we propose a noniterative conservative interpolation at each time step. With the lumped mass matrix, Eqn. (4), the requirement for conservation is that $\sum_{Vertices} WV$ be conserved before and after the swapping. When an edge is swapped, the control volumes change for all the four points forming the quadrilateral. Figure 15 shows a four point quadrilateral subset of a triangulation. The initial triangulation given by the triangles 125 and 126 and the triangulation after swapping given by the triangles 156 and 256. The portions of the control volumes associated with the four vertices that fall inside the quadrilateral region are illustrated as well. A conservative interpolation can be performed by treating the solution inside the control volumes as piecewise constants and computing geometrically the fractions of the old control volumes comprising the new ones. This would require complex intersection computations, especially for the vertex-centered scheme, where the control volume edges are segmented edges. This algorithm may be viewed as a particular application of the algorithm due to Ramshaw [32]. An algebraic approach that only involves areas $A_1, A_2, A_5, A_6, A_{1'}, A_{2'}, A_{5'}, A_{6'}$ is attractive, but the system becomes underdetermined if conservation is assumed. The drawbacks of these algorithms are algorithmic complexity and their diffusive character. The latter is particularly nettlesome because it degrades the accuracy to first order. For example, swapping back to the initial configuration and repeating the procedure in Figure 15, will result in equal values at the four vertices.

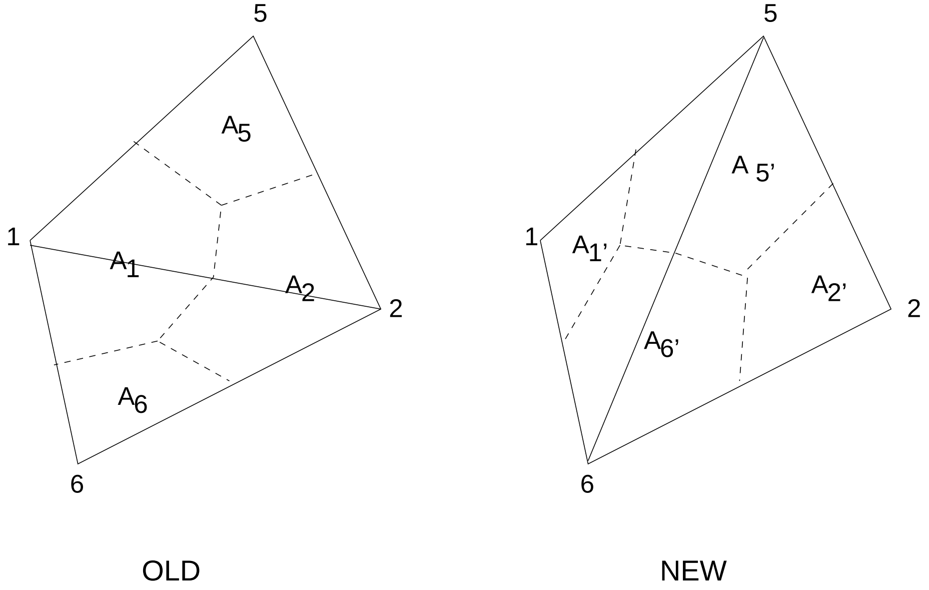


Figure 3: Two possible triangulations of four points and control volumes of vertices.

In the following, a conservative, linearity-preserving interpolation procedure is presented. The derivation is given assuming that the lumped mass approximation is adopted. We observe that the following identity holds:

$$\sum_{Vertices} WV \equiv \sum_{Triangles} \overline{W}\mathcal{V}, \tag{35}$$

where V is the area of the control volume and $\mathcal{V}$ is the area of the triangle, and $\overline{W}$, the average value in a triangle, is given by the average of the vertex values. Referring to Figure 15, when an edge is swapped, the contributions from the triangles, 125 and 126, change to the contributions from the two swapped triangles, 156 and 256, on the right hand side of Eqn. (35). We further note that each term in the right hand-side may be viewed as volume under a "roof" in the $x-y-W$ space. If the data were linear, the volumes under the two triangulations would be identical. In that case, no changes need to be made to the solution at the vertices. In the general case, we compute the contributions to the right-hand side of Eqn. (35) from the two triangulations as:

$$\begin{aligned} T_{old} &= \frac{1}{3}\left[(W_1+W_2+W_5)\mathcal{V}_{125}+(W_1+W_2+W_6)\mathcal{V}_{126}\right], \\ T_{new} &= \frac{1}{3}\left[(W_1+W_5+W_6)\mathcal{V}_{156}+(W_2+W_5+W_6)\mathcal{V}_{256}\right]. \end{aligned} \tag{36}$$

Changes are now made to the vertex values in a conservative manner by distributing the difference, $T_{old}-T_{new}$, to the nodes. In principle, the changes can be made to all the four vertex values and would still result in a linearity-preserving, conservative scheme, but in inspecting Figure 15 we notice that with swapping, the control volumes associated with vertices 1 and 2 (connected by the old edge) can only decrease, whereas those associated with vertices 5 and 6 (connected by the new edge) can only increase. Therefore changes need to be made only to vertices 5 and 6. We apportion the difference, $T_{old}-T_{new}$, equally:

$$\begin{aligned} W_5^{new}V_5^{new} &= W_5V_5^{new}+\frac{1}{2}(T_{old}-T_{new}), \\ W_6^{new}V_6^{new} &= W_6V_6^{new}+\frac{1}{2}(T_{old}-T_{new}). \end{aligned} \tag{37}$$

We now show that the interpolation formulas given by Eqn. (37) satisfy the conservation property. First note that the overlapping control volume v_i, given by the union of the triangles meeting at vertex i, is related to the nonoverlapping control volume V_i by the formula,

$$v_i = 3V_i \equiv \sum_j \mathcal{V}_j, \tag{38}$$

where the sum is over all the triangles meeting at vertex i. After swapping, we have

$$3V_5^{new} = v_5^{new} = v_5 - \mathcal{V}_{125}+\mathcal{V}_{156}+\mathcal{V}_{256} = v_5+\mathcal{V}_{126}. \tag{39}$$

We can thus derive the following relations:

$$\begin{aligned} V_5^{new} &= V_5+\mathcal{V}_{126}/3. \\ V_6^{new} &= V_6+\mathcal{V}_{125}/3, \\ V_1^{new} &= V_1-\mathcal{V}_{256}/3, \\ V_2^{new} &= V_2-\mathcal{V}_{156}/3. \end{aligned}$$

With the help of these relations, it is easy to show that the conservation property holds:

$$W_5^{new}V_5^{new} + W_6^{new}V_6^{new} + W_1V_1^{new} + W_2V_2^{new} = W_5V_5 + W_6V_6 + W_1V_1 + W_2V_2, \qquad (40)$$

where use has been made of the fact that the values at vertices 1 and 2 do not change.

In the formulas given by Eqn. (37), the difference $T_{old} - T_{new}$ has been apportioned equally to the two vertices. It is easy to see that this could introduce new extrema in the solution. The degree of freedom available in how this apportionment is made to the two vertices can be used to effectively prevent new extrema. Let

$$\begin{aligned} W_{max} &= \text{Max}(W_1, W_2, W_5, W_6) \\ W_{min} &= \text{Min}(W_1, W_2, W_5, W_6). \end{aligned}$$

For $i = 5$ or 6 compute,

$$r_i = \begin{cases} V_i^{new}\frac{W_{max}-W_i}{T_{old}-T_{new}}, & \text{if } T_{old} - T_{new} > 0 \\ V_i^{new}\frac{W_{min}-W_i}{T_{old}-T_{new}}, & \text{if } T_{old} - T_{new} < 0. \end{cases}$$

Let $r = \text{Min}[0.5, r_5, r_6]$. The interpolation formulas can be written in a compact form as follows:

$$\begin{aligned} W_5^{new} &= W_5 + \tfrac{T_{old}-T_{new}}{V_5^{new}}\left[\tfrac{(r-r_5)(r-r_6)}{(0.5-r_5)(0.5-r_6)}0.5 + \tfrac{(r-0.5)(r-r_6)}{(r_5-0.5)(r_5-r_6)}r + \tfrac{(r-0.5)(r-r_5)}{(r_6-0.5)(r_6-r_5)}(1-r)\right], \\ W_6^{new} &= W_6 + \tfrac{T_{old}-T_{new}}{V_6^{new}}\left[\tfrac{(r-r_5)(r-r_6)}{(0.5-r_5)(0.5-r_6)}0.5 + \tfrac{(r-0.5)(r-r_6)}{(r_5-0.5)(r_5-r_6)}(1-r) + \tfrac{(r-0.5)(r-r_5)}{(r_6-0.5)(r_6-r_5)}r\right]. \qquad (41) \end{aligned}$$

The formulas are conservative, linearity-preserving, and also do not introduce new extrema. In contrast to the conservative formulas that assume piecewise constant values and result in equal values at the vertices upon repeated application, the new formulas converge locally to a linear (possibly least squares) profile for the four data points upon repeated application. The swapping and the interpolation formulas generalize to three-dimensional tetrahedral tessellations and are discussed in the Appendix.

7 Results

First, results from a one-dimensional example are presented illustrating the role of the mass matrix. On a uniform mesh, since the vertex and the centroid of its control volume coincide, the mass matrix can be lumped without suffering any adverse consequences. The situation is different if a mesh with variable mesh widths is considered. The one-dimensional advection equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \qquad (42)$$

is solved on a random grid. The scheme stores the pointwise values u_i at locations x_i. The initial condition is a Gaussian and the profile is advected by marching to a fixed time. A grid refinement study is carried out using a constant CFL number of 0.3. The spatial derivative is approximated in a MUSCL scheme [40] as

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u^L_{i+1/2} - u^L_{i-1/2}}{\Delta x}, \qquad (43)$$

where $u^L_{i+1/2}$, the value interpolated to the left side of the face $i + 1/2$, is given by

$$u^L_{i+1/2} = u_i + (x_{i+1/2} - x_i)\frac{u_{i+1} - u_{i-1}}{x_{i+1} - x_{i-1}}. \tag{44}$$

The mass matrix, which is tridiagonal, is inverted using the Thomas algorithm. We have experimented with two definitions of the mass matrix. The first one derives the mass matrix by assuming a piecewise linear distribution of data between the grid points and computes the average over the control volume $[x_{i-1/2}, x_{i+1/2}]$:

$$\frac{1}{4}\frac{x_i - x_{i-1}}{x_{i+1} - x_{i-1}}u_{i-1} + \frac{3}{4}u_i + \frac{1}{4}\frac{x_{i+1} - x_i}{x_{i+1} - x_{i-1}}u_{i+1} \tag{45}$$

A second definition of the mass matrix is derived by computing the average of the reconstruction polynomial within a control volume. This polynomial is given by

$$u(x) = u_i + (x - x_i)\frac{u_{i+1} - u_{i-1}}{x_{i+1} - x_{i-1}}. \tag{46}$$

The average over the control volume is given by

$$-\frac{x_{i-1} - 2x_i + x_{i+1}}{8(x_{i+1} + x_{i-1})}u_{i-1} + u_i + \frac{x_{i-1} - 2x_i + x_{i+1}}{8(x_{i+1} + x_{i-1})}u_{i-1} \tag{47}$$

Figure 1 compares the errors in L_2 norm with the mass matrices given by Eqn. (45) and Eqn. (47), and with the lumped mass matrix. All the schemes exhibit second order accuracy but the errors are larger with the mass matrix given by Eqn. (45). The results obtained with the lumped mass matrix are almost identical to those obtained with Eqn. (47). Taylor series expansion would imply a first order error with the lumped mass matrix on a random grid, whereas Figure 1 clearly indicates second order accuracy. The results therefore reveal the inadequacy of local analysis. The results obtained with the usual finite element mass matrix, with $h_i = x_i - x_{i-1}$,

$$\frac{2h_i}{6(h_i + h_{i+1})}u_{i-1} + \frac{4}{6}u_i + \frac{2h_{i+1}}{6(h_i + h_{i+1})}u_{i+1}, \tag{48}$$

are also shown in Figure 1 and again display larger errors compared to the lumped mass approximation. The reason for this is that the finite element mass matrix is consistent with a Galerkin method which corresponds to a central difference discretization, whereas the spatial differencing employed here is upwind-biased. After experimenting with a one-parameter family of mass matrices, we have found that the lumped mass matrix gives the lowest errors with this particular spatial discretization.

It is well known in finite element literature [38] that in some cases the lumping of the mass matrix does not compromise the solution accuracy, but that the mass matrix may play a crucial role when higher-order discretizations are considered. To examine this, we employ a quadratic reconstruction procedure utilizing point values. With $h_i = x_i - x_{i-1}$, we obtain,

$$\begin{aligned} u^L_{i+1/2} &= -\frac{h^2_{i+1}}{4h_i(h_i + h_{i+1})}u_{i-1} \\ &+ \left(\frac{1}{2} + \frac{h_{i+1}}{4h_i}\right)u_i + \frac{h_{i+1} + 2h_i}{4(h_{i+1} + h_i)}u_{i+1} \end{aligned}$$

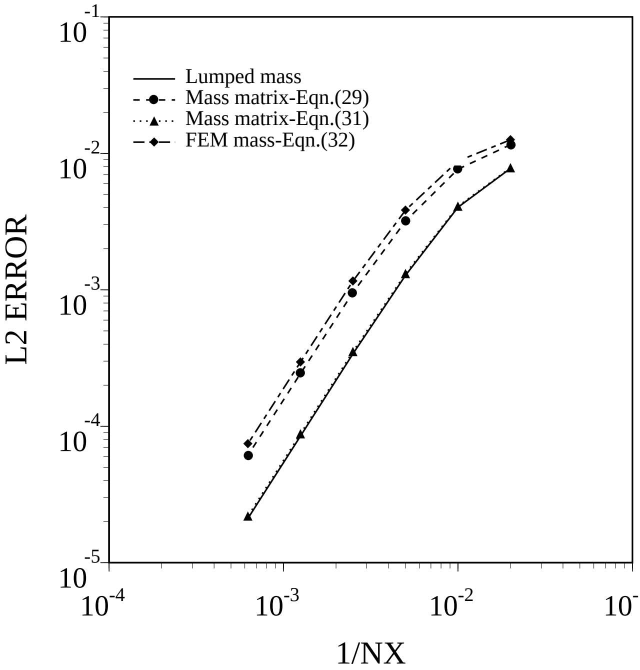


Figure 4: L_2 norms of the errors with various schemes on a random grid.

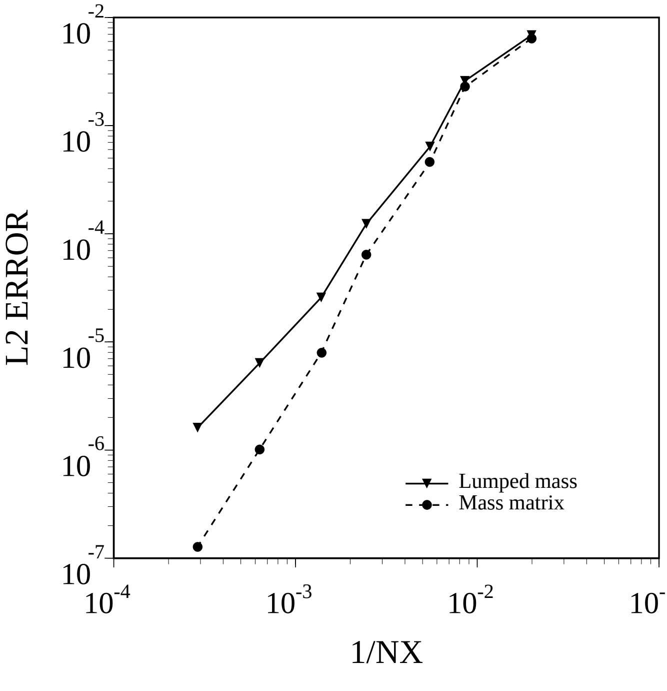


Figure 5: L_2 norms of the errors with quadratic reconstruction on a random grid.

The finite volume mass matrix is derived by determining the average of the quadratic distribution and is given by:

$$
\begin{aligned}
& a_1 u_{i-1} + a_2 u_i + a_3 u_{i+1}, \\
a_1 &= \frac{-2h_{i+1}^2 + 2h_i h_{i+1} + h_i^2}{12h_i(h_i + h_{i+1})}, \\
a_3 &= \frac{h_{i+1}^2 + 2h_i h_{i+1} - 2h_i^2}{12h_{i+1}(h_i + h_{i+1})}, \\
a_2 &= 1 - a_1 - a_3.
\end{aligned}
$$

The standard Runge-Kutta scheme which is fourth order accurate in time is used for the higher order computations. Figure 2 shows the error plots using the lumped mass matrix and the full mass matrix on the random grid. It shows that with the lumped mass matrix, the accuracy eventually degrades to second order as the grid is refined, whereas using the full matrix yields the third order accuracy of the spatial discretization. We have observed that using any other definition for the mass matrix degrades the accuracy to second order.

The implications for the scheme in multiple dimensions are clear. As long as only a second order accurate scheme is used and we operate with either cell-vertex or cell-centered data, the mass matrix may be lumped without any loss of order of accuracy. The mass matrix can also be ignored for second (and higher) order accurate schemes if a strict cell-average interpretation is employed. If point values are used to construct third and higher order accurate schemes, the accuracy will degrade if the mass matrix is lumped. For higher order accurate schemes based on point values, the indirect mass matrix inversion technique discussed earlier will help preserve the order of accuracy of the scheme.

We next present results from two-dimensional inviscid calculation over a pitching airfoil. The transonic flow is over a sinusoidally oscillating NACA0012 airfoil where the angle of attack $\alpha(t)$ varies according to the formula

$$\alpha(t) = \alpha_m + \alpha_0 sin(\omega t) \tag{49}$$

For the test case chosen, $\alpha_m = 0.016°$, $\alpha_0 = 2.51°$, $\kappa = \frac{\omega c}{2U_\infty} = 0.0814$ and the freestream Mach number, $M_\infty = 0.755$. Computing this flow using an explicit scheme is time-consuming because of the low frequency. The flows are computed using two meshes, referred to as GRID1 and GRID2, each having 6336 vertices. These are shown in Figures 3 and 4, respectively. GRID1 is generated by drawing diagonals in a structured C-mesh and is fairly uniform. GRID2 is generated by random perturbations on GRID1. In GRID2, the centroids of the control volumes formed by the median dual are not represented well by the vertices. Figure 5 shows the lift histories during the third cycle of oscillation. Four curves are shown, namely, the histories with the lumped and full mass matrices for GRID1 and GRID2. Since a projection-evolution approach is not followed, the mass matrix is derived by using a definition similar to Eqn. (45). As expected, the mass matrix has little impact on the integrated quantities even in the random mesh. The differences in the solutions between the two grids are likewise insignificant. The CPU time increases by about 15% when the full mass matrix is included. These examples have been run with a maximum physical CFL number of 500, corresponding to using 54 time steps per sinusoidal oscillation of the airfoil. The number of iterations for the inner multigrid procedure is fixed at 30. Figure 6 shows the convergence

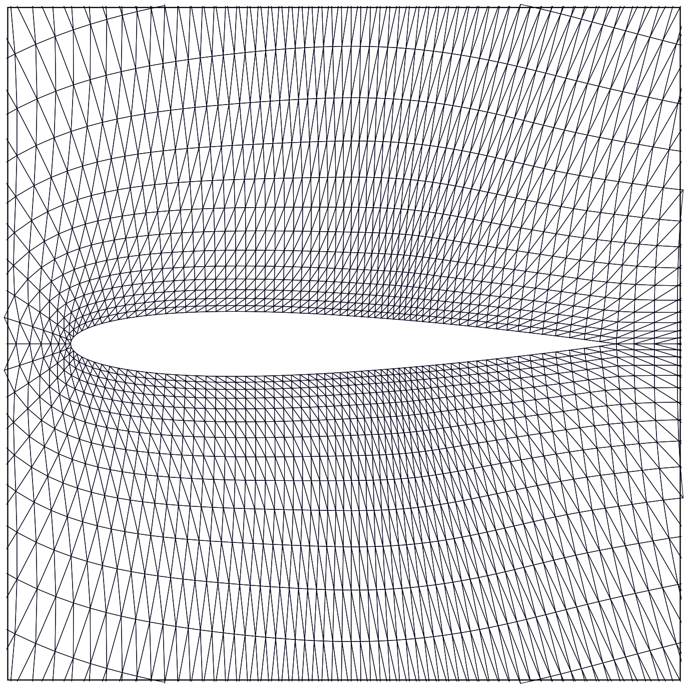


Figure 6: GRID1 about an NACA0012 airfoil with 6336 vertices.

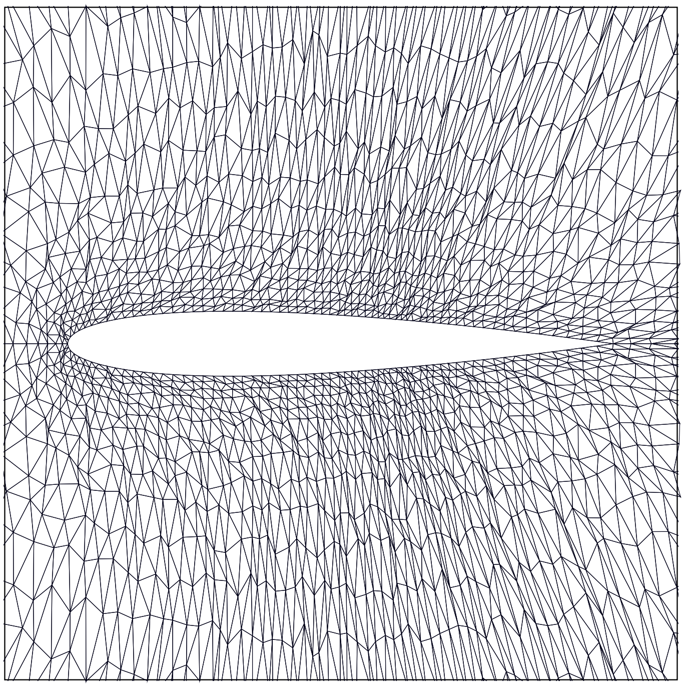


Figure 7: GRID2 about an NACA0012 airfoil with 6336 vertices.

of 4-grid agglomeration multigrid procedure during a particular time step with the lumped and the full matrices where the L_2 norm of the unsteady residual R^* is plotted as a function of the multigrid cycles using 4 grid levels. The convergence improves slightly when the mass matrix is included. The reason for this is that the mass matrix can be recast as an implicit smoothing operator. The remaining examples have been computed with the lumped mass matrix. Finally, Figures 7 and 8 shows the effect of the physical time step size. Three

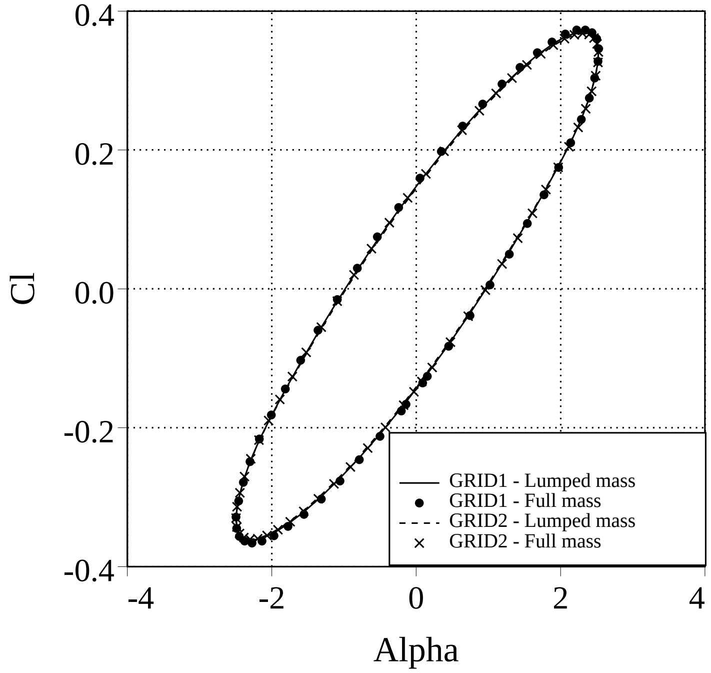


Figure 8: Lift histories during the third cycle of motion.

lift and moment histories are shown, employing 40, 20 and 10 steps per pitch cycle using the lumped mass matrix and deforming grids. The grid velocities are computed by using Eqn. (27). The nodes are repositioned by the procedure described earlier. No swapping of edges occurred in this computation. The number of multigrid cycles used at each time step are respectively, 15, 20 and 20. With 10 steps per pitch cycle, some discrepancy may be observed; increasing the number of multigrid cycles (thus solving the nonlinear problem better at each time step) did improve the solution. Also shown is a comparison with the experimental data of Landon [16] as well as with a structured grid computation employing a TVD scheme on the same grid [42]. The differences in the moment histories between the structured and unstructured grid computations may be attributed to the differences in the spatial discretization. To assess the effects of swapping, we modify the Delaunay criterion to force the swapping of edges. Given a quadrilateral with two possible triangulations, rather than swap to the new configuration if it has a larger minimum angle, we swap if some multiple (1.1 in this example) of the minimum angle in the new configuration is larger than the minimum angle in the old configuration; only one pass of this algorithm is performed. On a coarse triangular grid generated from a structured 128 × 32 grid, the moment histories during the first three pitch cycles without the swapping are shown in Figure 9. Figure 10 shows the histories during the first five pitch cycles with the swapping of edges. It is seen that the transient response is indeed quite different especially during the second cycle, although the response does become periodic during the fifth cycle. Figure 11 shows the moment histories on a triangular mesh generated from a 256×64 grid with the swapping of edges. The transient response is not nearly as erratic as on the 128×32 grid with swapping and more in line with the response on the coarse grid without swapping. Note that the swapping and subsequent conservative interpolation introduces errors proportional to Δx^2, which decrease as the grid is refined.

We next present a laminar unsteady calculation of impulsive start-up flow over a cylinder at a Reynolds number of 1200. Rumsey [34] has computed this flow using a structured grid code and has made detailed comparisons with experimental data [26]. We employ the same grid (192×64) as in [34], but divide the quadrilaterals into triangles. The flow separates and eventually vortices are shed. The computed Strouhal number is 0.225 as compared to the experimental value of 0.215 and the computed value of 0.222 using the structured grid. The time is nondimensionalized as $\bar{t} = t/(d/U_\infty)$ where d is the diameter of the cylinder and U_∞ is the freestream velocity. The sequence of non-dimensional time steps $\Delta\bar{t}$ is chosen as in [34]: $\Delta\bar{t} = 0.01$ for $\bar{t}$ up to 6.0, $\Delta\bar{t} = 0.02$ between $\bar{t} = 6.0$ and 9.0 and $\Delta\bar{t} = 0.05$ above $\bar{t} = 9.0$. We use the agglomeration multigrid strategy with six grids where the edge-coefficients needed for computing the viscous and inviscid terms on the coarse grids are precomputed as in [22]. Seven multigrid iterations were used at each time step yielding about of 2–3 orders of reduction in the unsteady residual. The centerline velocity at a time $\bar{t} = 2.9$ is plotted in Figure 8, and the maximum reverse flow velocity is plotted in Figure 9 as a function of $\bar{t}$ during the initial bubble growth phase. Good agreement may be observed with experiment. There is some discrepancy with the structured grid code which may be on account of the fact that it did not employ inner iterations. Thus, errors arising from factorization, linearization and and mismatch of operators may be present in the structured grid solution.

The final test case represents an exploratory inviscid computation of flow over a multi-element airfoil system undergoing deployment, as an example of bodies in relative motion. The geometry represents a sectional cut of the wing of the NASA Langley Transport System Research Vehicle (TSRV), and was obtained by direct measurement of the full scale aircraft (Boeing 737-100) [46]. The initial position, depicted in Figure 10, corresponds to the 15° flap setting, while the final condition represents the 40° flap setting, which is the approach configuration for this high-lift system. The initial grid about the 5-element airfoil is generated using the advancing front Delaunay triangulation method [21] at the 15° flap setting and is displayed in Figure 10. The grid has 26,191 vertices. Figure 11 displays the Mach contours for steady flow at this setting at an angle of attack of 5° and a freestream Mach number of 0.2.

We compute the time-accurate flow solution to full deployment using a five-grid agglomeration multigrid procedure. The motion is prescribed as a linear variation of both translation and rotation of the various airfoil elements, and is computed in 300 time steps. The non-dimensional time for deployment, defined as $\bar{t} = l/\sqrt{p_\infty/\rho_\infty}$, is taken as 10. Here l is the chord of the main element and p_∞, ρ_∞ are the freestream pressure and density. The grid restructuring involved presmoothing, spring analogy and edge swapping. Figures 12 and 13 depict a closeup of the grid in the flap region and the instantaneous flow solution at full deployment. Grid quality, while not as good as with the initial mesh, is acceptable considering the large scale motion. For this case, rather than the 3-point backward approximation, we have chosen to use the implicit midpoint (trapezoidal) rule:

$$\frac{V^{n+1}W^{n+1} - V^n W^n}{\Delta t} + \frac{R(W^{n+1}) + R(W^n)}{2} = 0.$$

When the connectivity of the mesh changes, the solution changes from one piecewise linear representation to another and the new control volumes are used to evolve the solution to the next time step. Using a 3-point backward rule would require that the volumes at the previous time step be changed as well. Figure 14 plots the total lift history as well as the lift histories of the elements as a function of physical time. The total lift, based on the chord length in the fully nested position, increases from an initial value of 2.53 to a final value of 3.10.

8 Conclusions

An efficient implicit time integration procedure has been developed. The implicit system is solved by using the agglomeration multigrid procedure. The issue of the mass matrix which arises in cell-vertex methods is addressed. It is shown that lumping of the mass matrix may be done for second-order accurate schemes without any degradation in order of accuracy. For higher order methods based on point values, it is shown that the lumping of the mass matrix degrades the order of accuracy of the scheme. Inviscid and viscous calculations have been presented for flow over a pitching airfoil and an impulsively started cylinder and the results have been compared with experiments and other computations. A mesh point movement strategy has been proposed and tested. This has been used to compute inviscid flow over a multi-element airfoil system undergoing deployment.

Acknowledgements

The authors thank Harold Atkins of NASA Langley Research center and Gordon Erlebacher of ICASE for stimulating discussions on conservative interpolation.

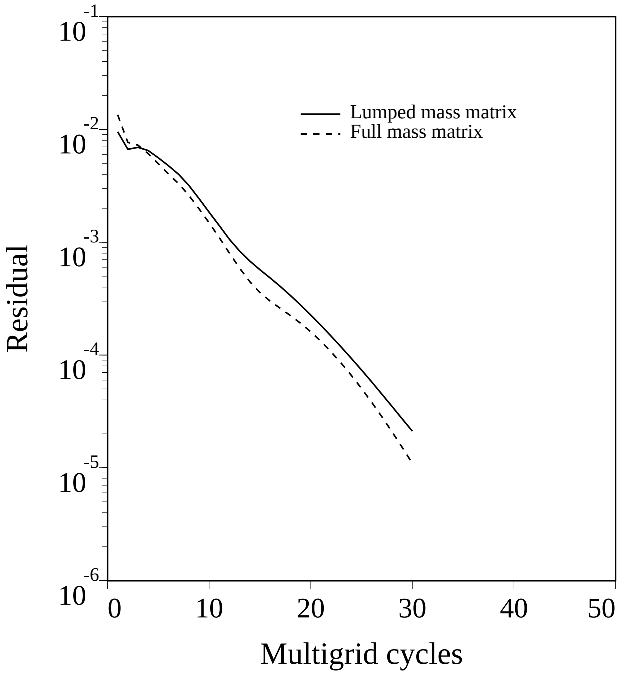


Figure 9: Histories of the unsteady residual at a particular time step.

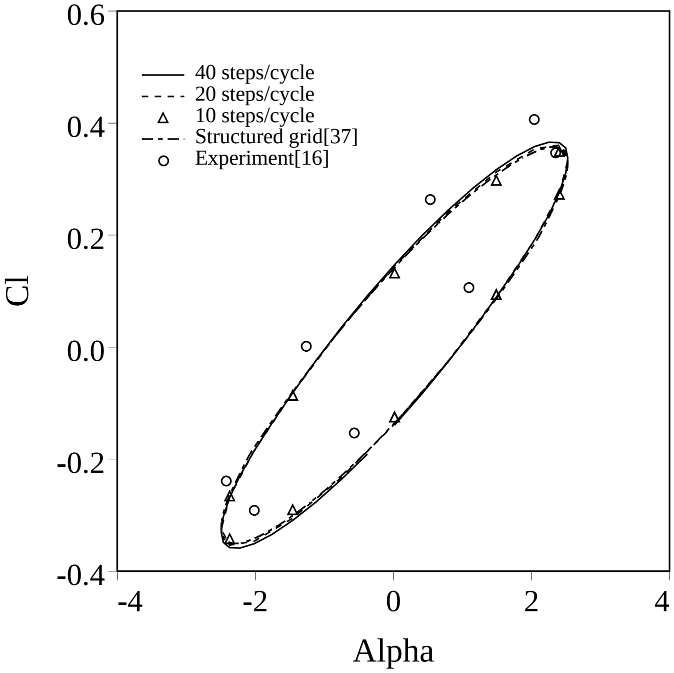


Figure 10: Lift histories during the third cycle of motion with with 40, 20 and 10 steps per cycle.

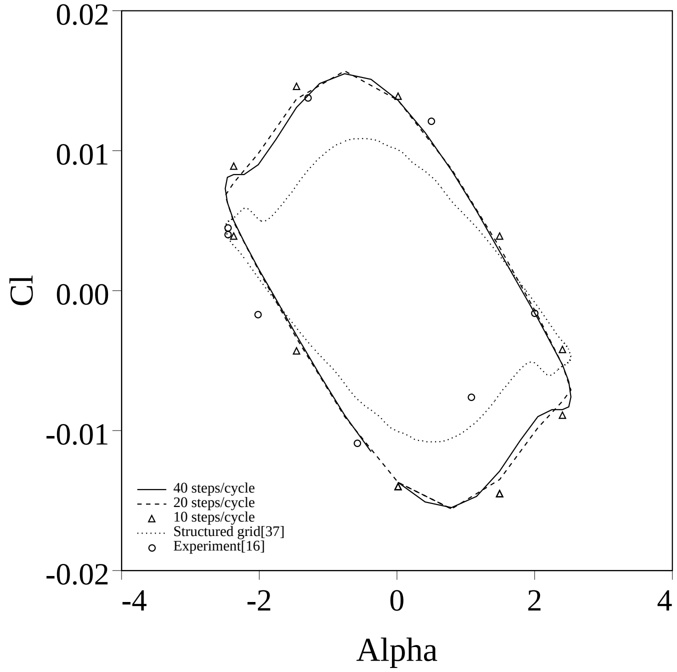


Figure 11: Moment histories during the third cycle of motion with with 40, 20 and 10 steps per cycle.

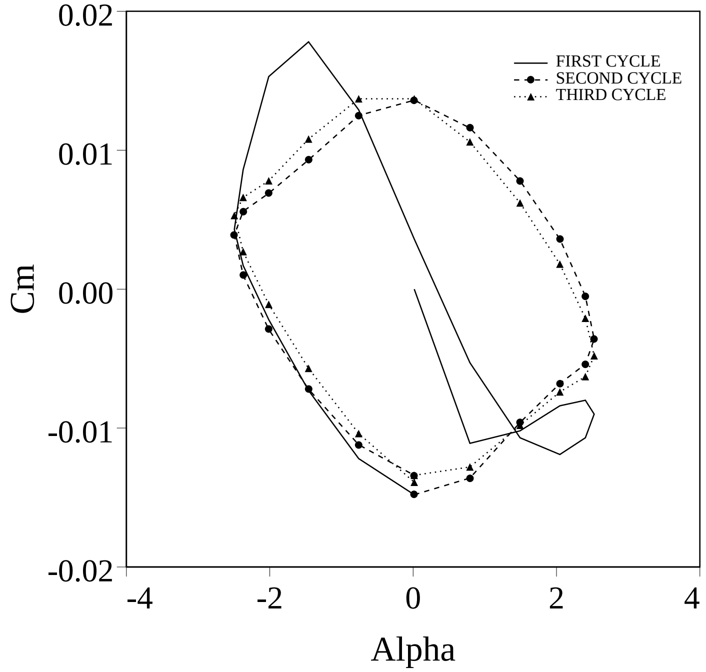


Figure 12: Moment histories on the coarse O-mesh without swapping of edges.

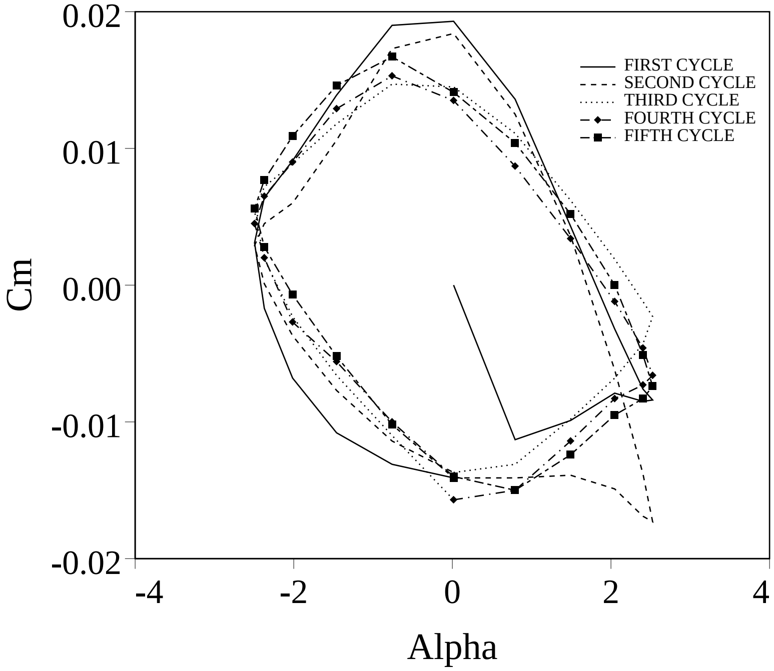


Figure 13: Moment histories on the coarse O-mesh with forced swapping of edges.

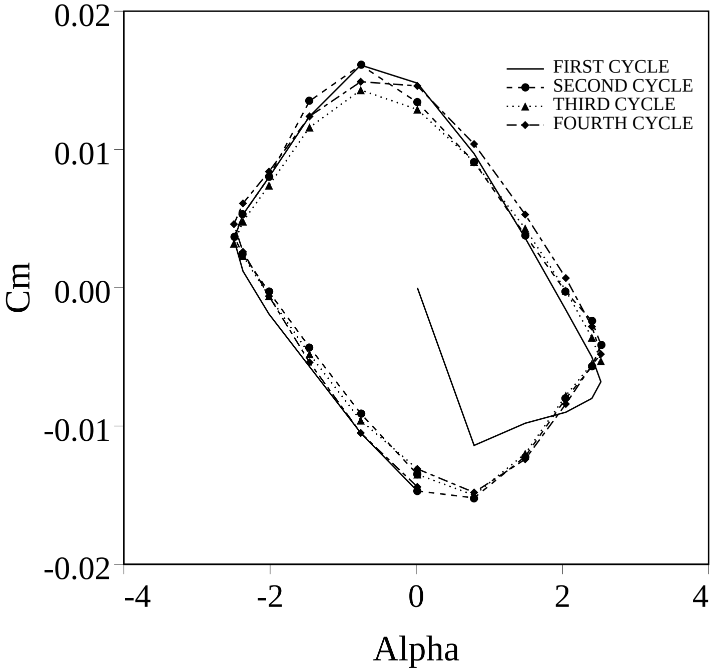


Figure 14: Moment histories on the fine O-mesh with forced swapping of edges.

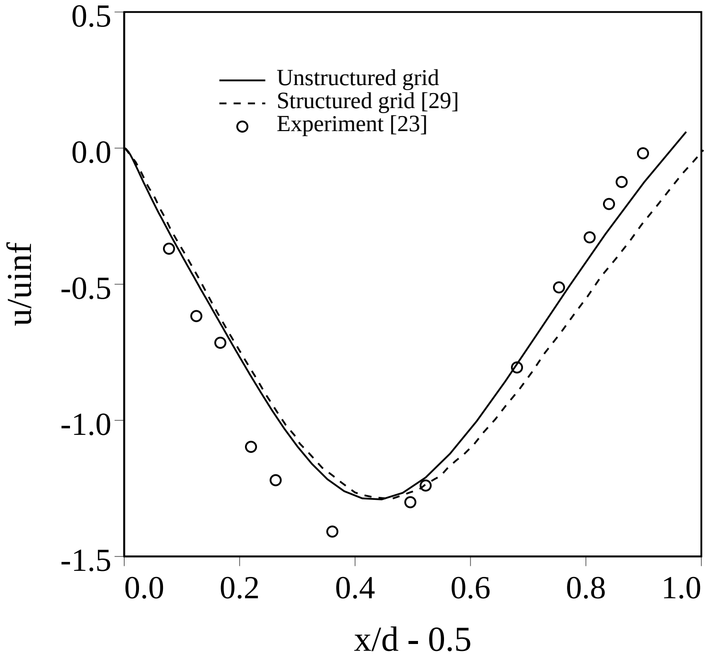


Figure 15: Velocity distribution at $\bar{t} = 2.9$ on the symmetric axis of the wake.

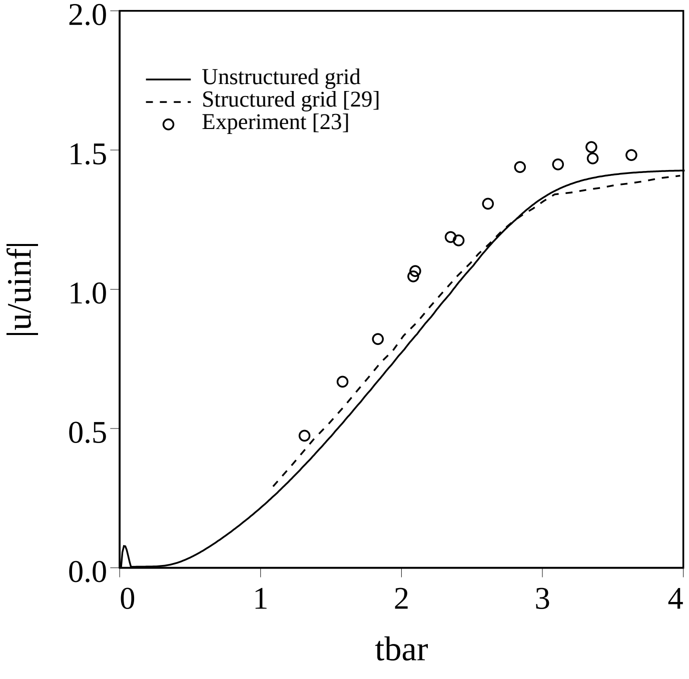


Figure 16: Time history of the maximum reverse flow velocity on the symmetric axis of the wake.

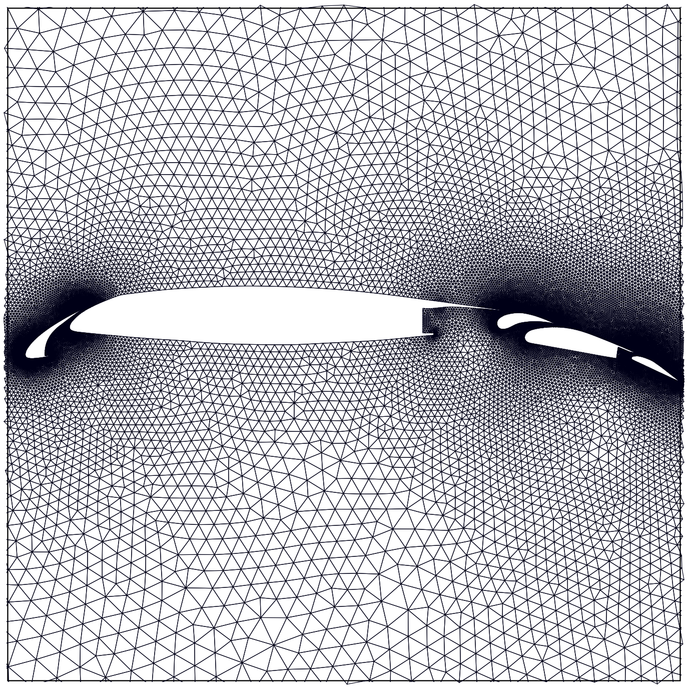


Figure 17: Initial grid at the semi-deployed position (15° flap deflection).

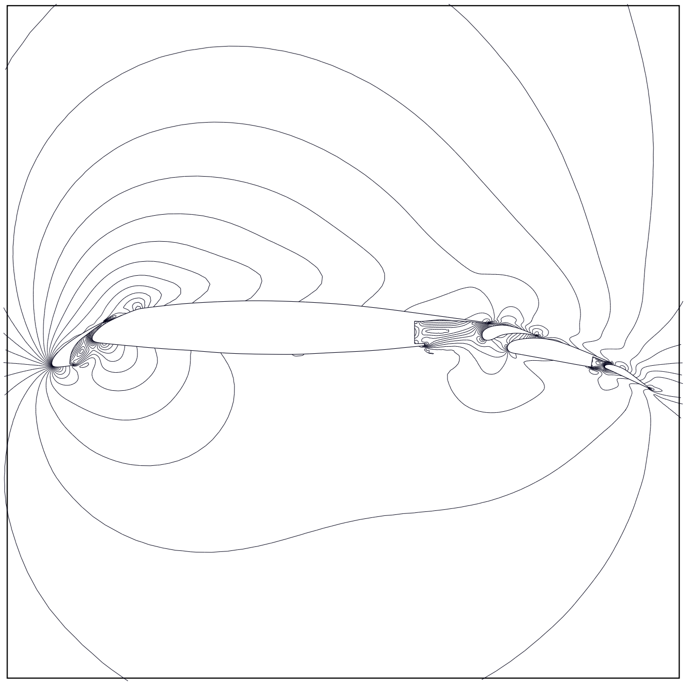


Figure 18: Mach contours for steady flow at the initial position (15° flap deflection).

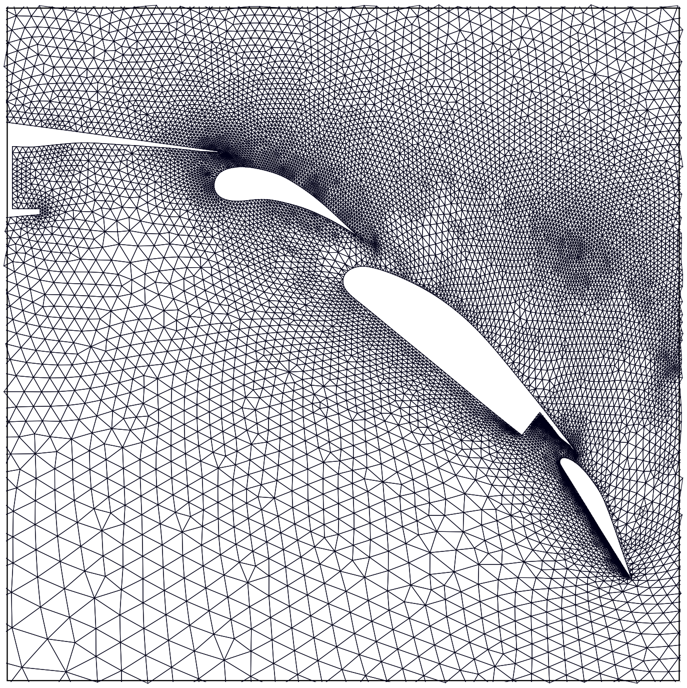


Figure 19: Closeup view of the grid at full deployment (40° flap deflection).

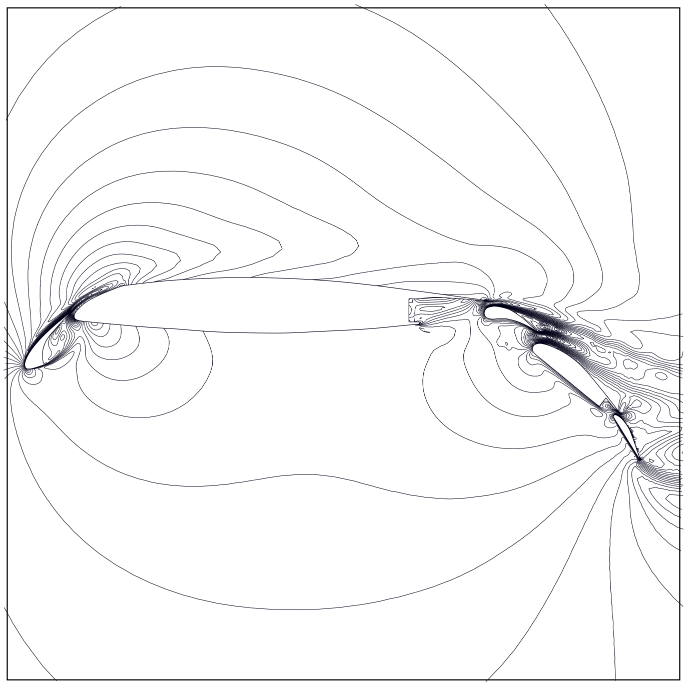


Figure 20: Mach contours of the instantaneous solution at full deployment (40° flap deflection).

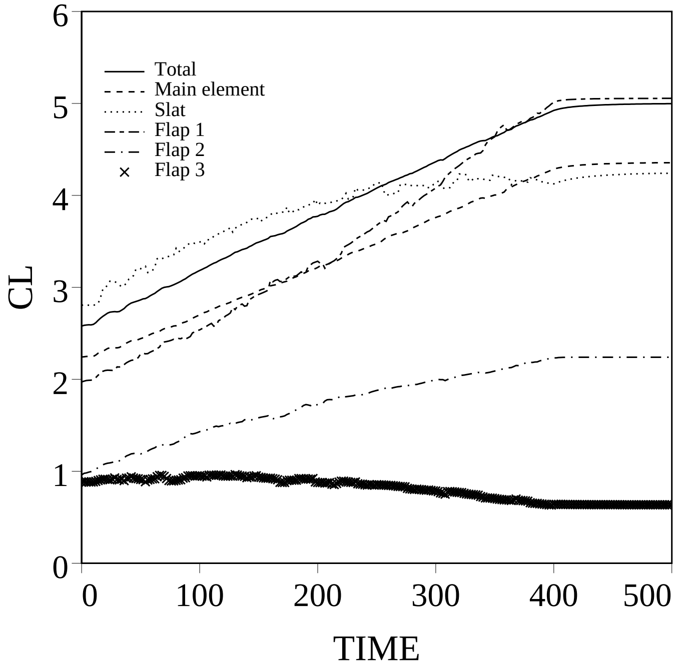


Figure 21: Time history of the lift coefficients.

Appendix

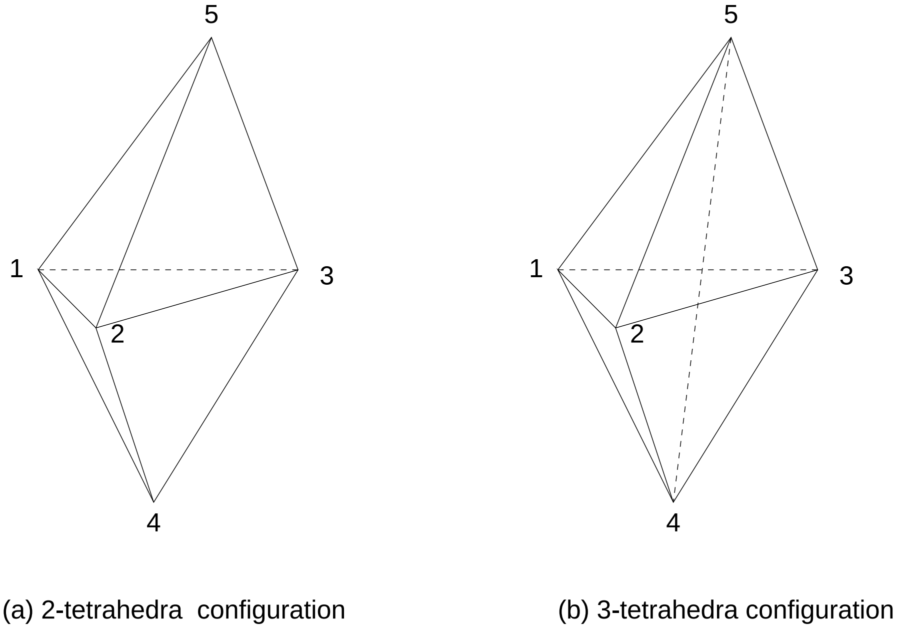
We discuss the swapping and interpolation procedures in three dimensions. Given 5 points in three space, the region enclosed by the convex hull can be tetrahedralized in one of three ways:

1. 4 tetrahedra sharing a common interior vertex.

2. 2 tetrahedra sharing a common triangular face.

3. 3 tetrahedra sharing a common edge.

Swapping in three dimensions consists of switching between the second and the third choices to improve the quality of the tetrahedralization [17]. Figure 25 illustrates the 2- and 3-tetrahedra configurations.



(a) 2-tetrahedra configuration

(b) 3-tetrahedra configuration

Figure 22: Two possible tetrahedralizations of five vertices.

Conservative, linearity-preserving interpolation, when switching from the 2-tetrahedra configuration to the 3-tetrahedra configuration, is similar to the two-dimensional situation, since only the values at the vertices joined by the new edge need to be changed. Define

$$\begin{aligned} T_2 &= \frac{1}{4}\left[(W_1+W_2+W_3+W_4)\mathcal{V}_{1234}+(W_1+W_2+W_3+W_5)\mathcal{V}_{1245}\right], \\ T_3 &= \frac{1}{4}\left[(W_1+W_2+W_4+W_5)\mathcal{V}_{1245}+(W_2+W_3+W_4+W_5)\mathcal{V}_{2345}+(W_3+W_1+W_4+W_5)\mathcal{V}_{3145}\right]. \end{aligned}$$

Changes are now made to the vertices 4 and 5 in a conservative manner by distributing the difference $T_2 - T_3$ using formulas similar to Eqn. (41), where the W_{max} and W_{min} are taken as the maximum and minimum over the five points.

Swapping from the 3- to the 2-tetrahedra configuration is different, however. Here changes are made to the vertices $1, 2$ and 3 by distributing the difference $T_3 - T_2$. Thus, there are two degrees of freedom, which are needed to enforce the condition that no new extrema be created. For $i = 1, 2, 3$ compute

$$r_i = \begin{cases} V_i^{new} \frac{W_{max} - W_i}{T_3 - T_2}, & \text{if } T_3 - T_2 > 0 \\ V_i^{new} \frac{W_{min} - W_i}{T_3 - T_2}, & \text{if } T_3 - T_2 < 0. \end{cases}$$

and define

$$r = \mathrm{Min}[r_1, r_2, r_3, \frac{1}{3}].$$

Now assume $r = r_1$. Then

$$W_1^{new} V_1^{new} = W_1 V_1^{new} + r(T_3 - T_2)/V_1^{new}.$$

Define $s = \mathrm{Min}[r_2, r_3, 0.5]$. Changes are made to vertices 2 and 3 as follows:

$$\begin{aligned} W_2^{new} &= W_2 + \tfrac{T_3 - T_2}{V_2^{new}}(1 - r)\left[\tfrac{(s-r_2)(s-r_3)}{(0.5-r_2)(0.5-r_3)}0.5 + \tfrac{(s-0.5)(s-r_3)}{(r_2-0.5)(r_2-r_3)}s + \tfrac{(s-0.5)(s-r_2)}{(r_3-0.5)(r_3-r_2)}(1-s)\right], \\ W_3^{new} &= W_3 + \tfrac{T_3 - T_2}{V_6^{new}}(1 - r)\left[\tfrac{(s-r_2)(s-r_3)}{(0.5-r_2)(0.5-r_3)}0.5 + \tfrac{(s-0.5)(s-r_3)}{(r_2-0.5)(r_2-r_3)}(1-s) + \tfrac{(s-0.5)(s-r_2)}{(r_3-0.5)(r_3-r_2)}s\right]. \end{aligned}$$

Formulas for the remaining cases, when $r = \frac{1}{3}, r_2, r_3$, can be derived in a similar manner.

References

[1] J. Alonso and A. Jameson, *Fully-implicit time-marching aeroelastic solution.* AIAA Paper 94-0056, Jan. 1994.

[2] W. K. Anderson and D. L. Bonhaus, *An implicit upwind algorithm for computing turbulent flows on unstructured grids*, Computers Fluids, 23 (1994), pp. 1–21.

[3] A. Arnone, M. Liou, and L. A. Povinelli, *Multigrid time-accurate integration of navier-stokes equations.* AIAA Paper 93-3361CP, June 1993.

[4] T. J. Barth and S. W. Linton, *An unstructured mesh Newton solver for compressible fluid flow and its parallel implementation.* AIAA Paper 95-0221, Jan. 1995.

[5] J. T. Batina, *Implicit flux-split Euler scheme for unsteady aerodynamic analysis involving unstructured dynamic meshes*, AIAA J., 29 (1991), pp. 1836–1843.

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

[7] A. Brennis and A. Eberle, *Application of an implicit relaxation method solving the Euler equations for time-accurate unsteady problems*, Transactions of ASME - J. Fluids Engrg., 112 (1990), pp. 510–520.

[8] G. Dahlquist and A. Bjork, *Numerical methods*, Prentice Hall, Inc, Englewood Cliffs, NJ, 1974.

[9] G. A. Davis and O. O. Bendiksen, *Unsteady transonic two-dimensional Euler solutions using finite elements*, AIAA J., 31 (1993), pp. 1051–1059.

[10] T. J. R. Hughes, *Recent progress in the development and understanding of SUPG methods with special reference to the compressible Euler and Navier-Stokes equations*, Intl. J. for Numer. Meth. in Fluids, 7 (1987), pp. 1261–1275.

[11] A. Jameson, *Time-dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings.* AIAA Paper 91-1596, July 1991.

[12] S. R. Kennon, J. M. Meyering, C. W. Berry, and J. T. Oden, *Geometry based Delaunay tetrahedralization and mesh movement strategies for multi-body CFD.* AIAA Paper 92-4575, Aug. 1992.

[13] W. L. Kleb, J. T. Batina, and M. H. Williams, *Temporal adaptive Euler/Navier-Stokes algorithm involving unstructured dynamic meshes*, AIAA J., 30 (1992), pp. 1980–1985.

[14] B. Koobus, M. H. Lallemand, and A. Dervieux, *Unstructured volume-agglomeration MG: Solution of the poisson equation.* INRIA Report 1946, June 1993.

[15] M. Lallemand, H. Steve, and A. Dervieux, *Unstructured multigridding by volume agglomeration: Current status*, Computers and Fluids, 21 (1992), pp. 397–433.

[16] R. H. Landon, *NACA0012. Oscillating and trandient pitching.* Data Set 3 in AGARD-R-702, Compendium of Unsteady Aerodynamic measurements, Aug. 1982.

[17] C. L. Lawson, *Properties of n-dimensional triangulations*, CAGD, 3 (1986), pp. 231–246.

[18] A. Lerat, J. Sides, and V. Daru, *Efficient computation of steady and unsteady transonic flows by an implicit solver*, in Advances in computational transonics, W. G. Habashi, ed., vol. 4 of Recent advances in Numerical Methods in Fluids, Swansea, U.K., 1985, Pineridge Press, pp. 545–576.

[19] R. Löhner, K. Morgan, and O. Zienkiewicz, *An adaptive finite element procedure for compressible high speed flows*, Comput. Methods Appl. Mech. Engrg, 51 (1985), pp. 441–465.

[20] D. J. Mavriplis, *Three-dimensional multigrid for the Euler equations*, AIAA Journal, 30 (1992), pp. 1753–1761.

[21] ——, *An advancing front Delaunay triangulation algorithm designed for robustness*, J. Comp. Phys., 117 (1995), pp. 90–102.

[22] D. J. Mavriplis and V. Venkatakrishnan, *Agglomeration multigrid for viscous turbulent flows.* AIAA Paper 94-2332, June 1994; to appear in Computers and Fluids.

[23] N. D. Melson, M. D. Sanetrik, and H. L. Atkins, *Time-accurate Navier-Stokes calculations with multigrid acceleration*, in 6th Copper Mountain Conf. on Multigrid Methods, 1993, pp. 423–439.

[24] K. Miller, *On the mass matrix spectrum bounds of Wathen and the local moving finite elements of Baines*, SIAM J. Numer. Anal., 29 (1992), pp. 89–106.

[25] K. Miller and R. N. Miller, *Moving finite elements. i*, SIAM J. Numer. Anal., 18 (1981), pp. 1019–1032.

[26] H. Nagata, H. Funada, and T. Matsui, *Unsteady flows in the vortex region behind a circular cylinder started implusively, 2nd report, velocity fields and circulations*, JSME, 28 (1985), pp. 2608–2616.

[27] B. Nkonga and H. Guillard, *Godunov-type method on non-structured meshes for three-dimensional moving boundary problems*, Comput. Methods Appl. Mech. Engrg, 113 (1994), pp. 183–204.

[28] B. Palmerio, *An attraction-repulsion mesh adaption model for flow solution on unstructured grids*, Computers and Fluids, 23 (1994), pp. 487–506.

[29] J. Peraire, J. Peirö, and K. Morgan, *A 3d finite-element multigrid solver for the Euler equations.* AIAA Paper 92-0449, Jan. 1992.

[30] T. H. Pulliam, *Time accuracy and the use of implicit schemes.* AIAA Paper 93-3360CP, July 1993.

[31] M. M. Rai, *Navier-Stokes simulations of blade-vortex interaction using high-order accurate upwind schemes.* AIAA Paper 87-0543, Jan. 1987.

[32] J. D. Ramshaw, *Conservative rezoning algorithm for generalized two-dimensional meshes*, J. Comp. Phys., 59 (1985), pp. 193–199.

[33] S. E. Rogers, D. Kwak, and C. Kiris, *Steady and unsteady solutions of the incompressible Navier-Stokes equations*, AIAA J., 29 (1991), pp. 603–610.

[34] C. L. Rumsey, *Details of the computed flowfiedl over a circular cylinder at Reynolds number 1200*, J. Fluids Engrg., 110 (1988), pp. 447–452.

[35] K. P. Singh, J. C. Newman, and O. Baysal, *Dynamic unstructured method for flows past multiple objects in relative motion*, AIAA J., 33 (1995), pp. 641–649.

[36] D. C. Slack, D. L. Whitaker, and R. W. Walters, *Time integration algorithms for the two-dimensional Euler equations on unstructured meshes*, AIAA J., 32 (1994), pp. 1158–1166.

[37] W. A. Smith, *Multigrid solution of transonic flow on unstructured grids*, in Recent Advances and Applications in Computational Fluid Dynamics, Nov. 1990. Proceedings of the ASME Winter Annual Meeting, Ed. O. Baysal.

[38] G. Strang and G. J. Fix, *An analysis of the finite element method*, Prentice Hall, New Jersey, 1973.

[39] P. D. Thomas and C. K. Lombard, *Geometric conservation law and its applications to flow computations on moving grids*, AIAA J., 17 (1979), pp. 1030–1037.

[40] B. van Leer, *Towards the ultimate conservative difference scheme V. A second order sequel to Godunov's method*, J. Comp. Phys., 32 (1979), pp. 101–136.

[41] J. C. Vassberg, *A fast, implicit unstructured-mesh Euler method.* AIAA Paper 92-2693, 1992.

[42] V. Venkatakrishnan and A. Jameson, *Computation of unsteady transonic flows by the solution of Euler equations*, AIAA J., 26 (1988), pp. 974–981.

[43] V. Venkatakrishnan and D. J. Mavriplis, *Agglomeration multigrid for the three-dimensional Euler equations.* AIAA Paper 94-0069, June 1994; to appear in AIAA J.

[44] ——, *Implicit solvers for unstructured meshes*, Journal of Computational Physics, 105 (1993), pp. 83–91.

[45] A. Wathen, *Mesh-independent spectra in the moving finite element equations*, SIAM J. Numer. Anal., 23 (1986), pp. 797–814.

[46] L. P. Yip, P. Vijgen, J. D. Hardin, and C. P. van Dam, *Subsonic high-lift flight research on the NASA transport systems.* AIAA Paper 92-4103, Aug. 1992.

[47] H. Zhang, M. Reggio, J. Y. Trepanier, and R. Camarero, *Discrete form of the GCL for moving meshes and its implementation in CFD schemes*, Computers and Fluids, 22 (1993), pp. 9–23.