JOURNAL OF COMPUTATIONAL PHYSICS **127,** 380–397 (1996)
ARTICLE NO. 0182

Implicit Method for the Computation of Unsteady Flows on Unstructured Grids

V. VENKATAKRISHNAN* AND D. J. MAVRIPLIS*

Institute for Computer Applications in Science and Engineering, MS 132C, NASA Langley Research Center, Hampton, Virginia 23681-0001

Received August 31, 1995; revised April 9, 1996

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.

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 [21, 30] or implicit schemes [37, 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 [19, 43], and temporal adaptation techniques [14], 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. The last method has the drawback of not being easily parallelizable. 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 [32, 31, 7, 34]. 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 [42] has applied this technique to compute flow

* 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.

0021-9991/96 $18.00

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 nonnested grids and using piecewise-linear transfer operators [21, 30]. An important development in the use of multigrid techniques for unstructured grids is the agglomeration multigrid algorithm [16, 38, 45] for the two- and three-dimensional Euler equations. This technique has since been extended to deal with viscous flows [15, 23]. 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 straightforward 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

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

$$W = [\rho, \rho\mathbf{V}, \rho e]^{\mathrm{T}}$$

$$F(W,\mathbf{n},\mathbf{s}) = (\mathbf{V}-\mathbf{s})\cdot\mathbf{n}W,$$

$$G(W,\nabla W,\mathbf{n}) = [0, \mathbf{t}, \mathbf{t}\cdot\mathbf{V} - \mathbf{q}\cdot\mathbf{n}]^{\mathrm{T}},$$

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

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

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 Eq. (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 [21]. 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. Following [21], all the loops are recast as loops over edges for inviscid, as well as viscous, terms. The edge-coefficients are precomputed since they only depend on the geometry. For the inviscid terms, the edge-coefficients are the projections of the control volume faces onto the coordinate planes.

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 Eq. (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 ordinary differential equations in Eq. (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'' [39], 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 few 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 [20, 9] is not efficient, especially when larger grids are used. Miller [25] and Wathen [46] 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 Eq. (3) by the identity matrix and making a 3-point backward-difference approximation for the time derivative yields

$$\frac{3}{2\,\Delta t} V^{n+1}W^{n+1} - \frac{2}{\Delta t} V^nW^n + \frac{1}{2\,\Delta t} V^{n-1}W^{n-1} + R(W^{n+1}) = 0. \tag{4}$$

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]. Equation (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 multistage Runge–Kutta scheme applied to solve Eq. (6) performs the role of a smoother in the multigrid process. A low-storage, second-order accurate, m-stage Runge–Kutta scheme to advance U is given by

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

Starting with $U^1 = W^n$, the sequence of iterates U^l, $l = 1, 2, 3, \ldots$ converges to W^{n+1}. A five-stage Runge–Kutta scheme with three evaluations of dissipation given in [11] is used. In combination with the technique of residual averaging, this allows a non-dimensional time step (CFL) of 6 to be used.

This formulation has been observed by Arnone *et al.* [3] to be unstable for small physical time steps, Δt. This is counterintuitive 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.* [24] showed that the problem is due to 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 $(3/2\ \Delta t)$ 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. Using Eq. (7), the Runge–Kutta scheme now becomes

$$Q_0 = U^l$$

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

$$k = 1, m,$$

$$U^{l+1} = Q_m. \tag{10}$$

With the modified scheme, Melson *et al.* [24] have shown that arbitrarily large or small Δt may be employed.

As in [11, 24], 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 adopt 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 edge-coefficients for the coarse grids are computed from those of the fine grid. Following [15, 23], the edge-coefficients for the viscous terms are scaled so as to yield a consistent discretization on the coarse grids. Also, as long as no grid points are added or removed, the triangulation remains valid, and the grid connectivity remains unchanged, the interpolation operators remain the same. Multigrid schemes based on nonnested 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 overhead.

A few observations about the dual time stepping procedure are in order. The maximum outer (physical) time step is determined by the physics of the problem. Assuming that the spatial accuracy is adequate, temporal convergence needs to be established by performing a uniform refinement in time. The convergence rate of the multigrid procedure for the inner problem varies inversely with the size of the physical time step: the smaller the physical time step, the faster the convergence. We find that typically it is sufficient to obtain about two orders of magnitude reduction in the unsteady residual. The number of multigrid cycles required to achieve this is problem-dependent.

4. TREATMENT OF THE MASS MATRIX

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

$$\frac{3}{2\,\Delta t}V^{n+1}M^{n+1}W^{n+1} - \frac{2}{\Delta t}V^nM^nW^n + \frac{1}{2\,\Delta t}V^{n-1}M^{n-1}W^{n-1} + R(W^{n+1}) = 0. \tag{11}$$

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

$$\frac{\partial VU}{\partial t^*} + R^*(U) = 0, \tag{12}$$

where U is the approximation to W^{n+1}, and $R^*(W)$ now includes the mass matrix terms. Notice that the first term, $(\partial/\partial t^*)\,VU$, 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.* [24] poses a serious problem. Their modification would require the term $(3/2\ \Delta t)\ VMU$, 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

$$Q_0 = U^l$$

$$\left[I + \frac{3}{2\,\Delta t}\alpha_k\,\Delta t^*M^{n+1}\right] V^{n+1}Q_k = V^{n+1}Q_0 - \alpha_k\,\Delta t^*[R(Q_{k-1}) - S], \quad k = 1, m, \tag{13}$$

$$U^{l+1} = Q_m, \tag{14}$$

where the source term S is now given by

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

If we simply replace the mass matrix M by the identity on the left-hand side of Eq. (14), we have observed that the instability at small time steps persists. In our modification, we first add and subtract $(3/2\ \Delta t)\ \alpha_k\ \Delta t^*M^{n+1}V^{n+1}Q_{k-1}$ on the right-hand side of Eq. (13) to obtain

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

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, $(3/2\ \Delta t)\ \alpha_k\ \Delta t^* M\ VQ$, appears on the left- and the right-hand sides of Eq. (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

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

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 an adaptive h-refinement technique in order to create a fine mesh. In the present work, we only investigate mesh point movement strategies. Tension spring analogy [5, 29, 36] or other physical analogies, such as incompressible flow [13], are typically used to move the mesh points. In the former case, 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 is 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 [26], where evolution equations are derived for grid point motion from the governing equations.

We generate a triangular mesh with a smooth point distribution using the advancing-front Delaunay algorithm [22]. We then 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,

$$\sum_j K_{ij}(\mathbf{x_i} - \mathbf{x_j}) = S_{ij}, \tag{19}$$

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 reconnect the edges by the Delaunay criterion, which states that the circumcircle of each $(n+1)$-simplex in an n-dimensional triangulation contains no other vertex in its interior. In two dimensions, this can be enforced by means of a local swapping procedure [18]. Examining each convex quadrilateral, of the two possible triangulations (Fig. 3), the one that meets the circumcircle test is chosen. Since the grid quality must necessarily deteriorate before grid lines can cross and render the grid invalid, the Delaunay criterion determines the best possible triangulation for the given point distribution. In two dimensions, the circumcircle criterion can be shown to be equivalent to maximizing the minimum angle [18] and, thus, the Delaunay triangulation represents the ''most equilateral'' triangulation of the given point set. 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:

$$(I + \varepsilon N_i)\mathbf{x}_i^{\text{new}} = \mathbf{x}_i^{\text{old}} + \varepsilon \sum_j \mathbf{x}_\mathbf{j}^{\text{new}}, \tag{20}$$

where N_i is the degree of node i. Typically, we find that 100–200 steps are required to solve Eq. (19), while only about 3–4 iterations of Eq. (20) are sufficient. The value of ε is taken to be 0.25 in Eq. (20). The number of Jacobi iterations to solve Eq. (19) may appear to be large, but this is mainly due to the large displacements arising from using the 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, reagglomeration, and generation of the edge-coefficients consume about a third of the time required for the flow solver, mainly because of the swapping procedure not being vectorized. The swapping procedure has since been vectorized and the entire sequence of steps consumes about one-eighth the time required for the flow solver (with 20 multigrid cycles per time step).

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) [40, 48, 28] formalizes this concept. It can be derived from the continuity equation in Eq. (1) by first assuming the control volumes to be the simplices themselves. Assuming a uniform velocity field and a constant density field, we obtain

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

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

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

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

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

where $\mathscr{S}_I(t) = \sum_j \Omega_{I,j}(t)$ is the surface enclosing the volume $\mathscr{V}_I(t)$ 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 $\mathbf{s}$ and the grid normal $\mathbf{n}$ to be determined. Since simplices are convex, the volumes $\mathscr{V}^n$, $\mathscr{V}^{n+1}$ are uniquely determined by the positions of the points at time levels n and $n + 1$. In two dimensions, it can be shown [48] that the change in volume can be expressed as

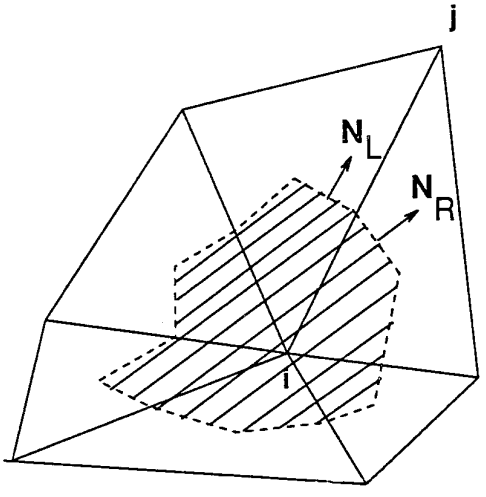

FIG. 1. Control volume for vertex i and edge normals.

$$\mathscr{V}_I^{n+1} - \mathscr{V}_I^n = \Delta t \sum_j \mathbf{s}_j^{n+1/2} \cdot \mathbf{N}_j^{n+1/2}, \tag{24}$$

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

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

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

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

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

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 [28], 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 (Fig. 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 Eqs. (24)–(26) 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 expression for the change in V_i, the volume associated with vertex i,

$$V_i^{n+1} - V_i^n = \Delta t \sum_j \mathbf{N}_{j,L}^{n+1/2} \cdot \mathbf{s}_{j,L}^{n+1/2} + \mathbf{N}_{j,R}^{n+1/2} \cdot \mathbf{s}_{j,R}^{n+1/2}, \quad (27)$$

where the summation is over the edges meeting at vertex i. Also, $\mathbf{N}_{j,L}^{n+1/2}$ and $\mathbf{N}_{j,R}^{n+1/2}$, 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, $\mathbf{s}_{j,L}^{n+1/2}$ and $\mathbf{s}_{j,R}^{n+1/2}$, 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 (Eq. (1)) for a two-level explicit or implicit scheme that obeys the GCL is performed as

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

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 Eq. (1) and makes use of the scaled normal for the edge, $\mathbf{N}_j^{n+1/2}$, defined as

$$\mathbf{N}_j^{n+1/2} = \mathbf{N}_{j,L}^{n+1/2} + \mathbf{N}_{j,R}^{n+1/2}. \quad (29)$$

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

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

It is easy to see that this formulation satisfies the GCL and that it reduces to the standard three-point formula (Eq. (11)) in the absence of any grid motion. We mention here that for fixed grid applications it is not necessary to compute $\mathbf{N}_{j,L}$ and $\mathbf{N}_{j,R}$ 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 $\mathbf{N}_{j,L}$ and $\mathbf{N}_{j,R}$ need to be computed separately for use in Eqs. (28), (30).

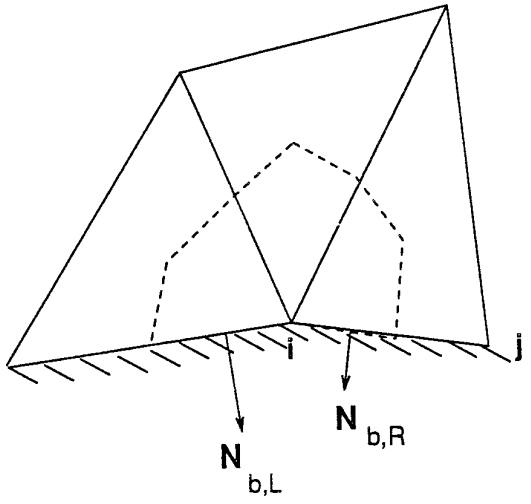

FIG. 2. Control volume for vertex boundary vertex i and boundary edge normals.

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 Eq. (29) for the two-level explicit or implicit scheme. For the three-level scheme, the scaled normal is defined as

$$\overline{\mathbf{N}}_j = \tfrac{3}{2}[\mathbf{N}_{j,L}^{n+1/2} + \mathbf{N}_{j,R}^{n+1/2}] - \tfrac{1}{2}[\mathbf{N}_{j,L}^{n-1/2} + \mathbf{N}_{j,R}^{n-1/2}]. \quad (31)$$

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 wall boundary conditions. The control volume for a boundary vertex is illustrated in Fig. 2. The summation over the edges in Eqs. (28), (30) is augmented with contributions from the boundary edges that delimit the control volume. For the two-level scheme, the following terms are added to Eq. (28),

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

where W_L and W_R are the values of W on the left and right

side of vertex i, $\mathbf{N}_{b,R}$ and $\mathbf{N}_{b,L}$ are the outward normals to the boundary edges of the control volumes, and $\mathbf{s}_{b,L}$ and $\mathbf{s}_{b,R}$ are the grid velocities associated with these edges. For an inviscid or a viscous wall, since the $\mathbf{N}\cdot[\mathbf{V}-\mathbf{s}]=0$, only the last two terms are retained. Likewise for the three-point scheme (Eq. (30)), the additional terms are

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

At the wall, only the last four terms are retained, implying the following boundary condition:

$$\begin{aligned} \mathbf{V}^{n+1}\cdot[\tfrac{3}{2}\mathbf{N}_b^{n+1/2}-\tfrac{1}{2}\mathbf{N}_b^{n-1/2}]\\ =\tfrac{3}{2}\mathbf{N}_b^{n+1/2}\cdot\mathbf{s}_b^{n+1/2}-\tfrac{1}{2}\mathbf{N}_b^{n-1/2}\cdot\mathbf{s}_b^{n-1/2}. \end{aligned} \tag{34}$$

6. MESH RESTRUCTURING AND INTERPOLATION

The procedure for restructuring the mesh was discussed earlier. At the beginning of every time step, the edges of the triangulation are swapped by some criterion to improve grid quality. When using the three-point difference formula (Eq. (11)), 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. Recall that in a finite volume scheme for a conservation law, the rate of change of the conserved variable is given by the net efflux through the faces. This principle holds true for volume as well (Eqs. (24)); i.e., the same shape has to evolve in time. Therefore, when the triangulation changes at a particular time step n, the volumes need to be recalculated at the previous time step, $n-1$, using the same connectivity, thus requiring a valid grid. This additional criterion is easily incorporated into the swapping procedure, i.e., swapping is performed only if the grid would also remain valid at the previous time step, $n-1$, with the new connectivity. Using the (two-level) 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].

At each time step, the mesh restructuring procedure is followed by presmoothing of the grid-point distribution, a mechanism for moving grid points in response to the motion of the boundary points and, finally, the evolution of the solution from the old grid to the new grid in a time-accurate manner with the grid movement terms present.

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, Eq. (4), the requirement for conservation is that $\Sigma_{\text{Vertices}}\, WV$ 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 3 shows a four-point quadrilateral subset of a triangulation. The initial triangulation is given by the triangles 125 and 126 and the triangulation after swapping is 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-constant 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 [33]. 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

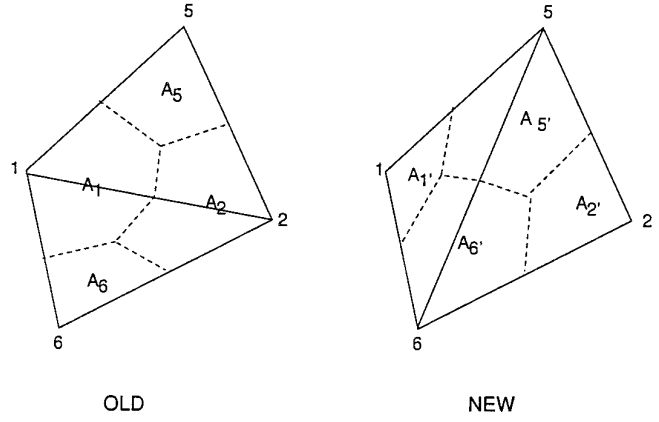

FIG. 3. Two possible triangulations of four points and control volumes of vertices.

to first order. For example, swapping back to the initial configuration and repeating the procedure in Fig. 3 will result in equal values at the four 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 identity

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

holds, where V is the area of the control volume, $\mathscr{V}$ is the area of the triangle, and $\overline{W}$, the average value in a triangle, is given by the average of the vertex values, W. Referring to Fig. 3, 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 Eq. (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 tilings 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 Eq. (35) from the two triangulations as

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

Changes are now made to the vertex values in a conservative manner by distributing the difference, $T_{\text{old}} - T_{\text{new}}$, to the nodes. In principle, the changes can be made to all the four vertex values and will result in a linearity-preserving, conservative scheme, but in inspecting Fig. 3 we notice that with swapping, the control volumes associated with vertices 1 and 2 (connected by the old edge) can only shrink, whereas those associated with vertices 5 and 6 (connected by the new edge) can only grow. Therefore changes need to be made only to vertices 5 and 6. We apportion the difference, $T_{\text{old}} - T_{\text{new}}$, equally:

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

We now show that the interpolation formulas given by Eq. (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

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

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

$$\begin{aligned} 3V_5^{\text{new}} &= v_5^{\text{new}} \\ &= v_5 - \mathscr{V}_{125} + \mathscr{V}_{156} + \mathscr{V}_{256} = v_5 + \mathscr{V}_{126}. \end{aligned} \tag{39}$$

We can thus derive the relations:

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

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

$$\begin{aligned} &W_5^{\text{new}} V_5^{\text{new}} + W_6^{\text{new}} V_6^{\text{new}} + W_1 V_1^{\text{new}} + W_2 V_2^{\text{new}} \\ &\quad = W_5 V_5 + W_6 V_6 + W_1 V_1 + W_2 V_2, \end{aligned} \tag{40}$$

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

In the formulas given by Eq. (37), the difference $T_{\text{old}} - T_{\text{new}}$ has been apportioned equally to the two vertices. This step 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

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

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

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

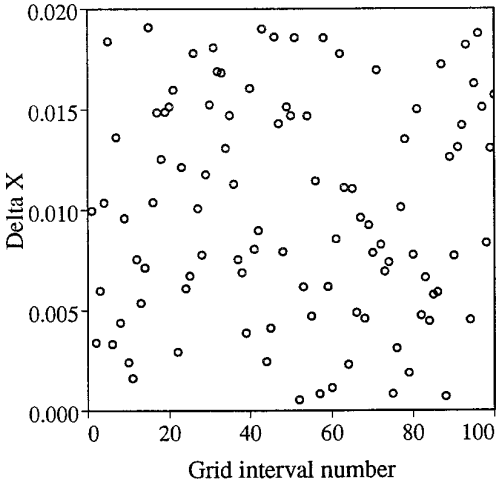

FIG. 4. Distribution of cell sizes for a grid with 101 points.

These 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. Although this paper only addresses the issues with two-dimensional triangular grids, the swapping procedure and 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,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \tag{42}$$

is solved on a random grid. The distribution of the mesh widths, $x_{i+1} - x_i$, is shown in Fig. 4 for a grid with 101 points. 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 time integration is performed with a second-order accurate Runge–Kutta scheme. The spatial derivative is approximated in a MUSCL scheme [41] as

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

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

$$-\frac{x_{i-1} - 2x_i + x_{i+1}}{4(x_{i+1} - x_{i-1})}u_{i-1} + u_i + \frac{x_{i-1} - 2x_i + x_{i+1}}{4(x_{i+1} - x_{i-1})}u_{i+1}. \tag{47}$$

Figure 5 compares the errors in L_2 norm with the mass

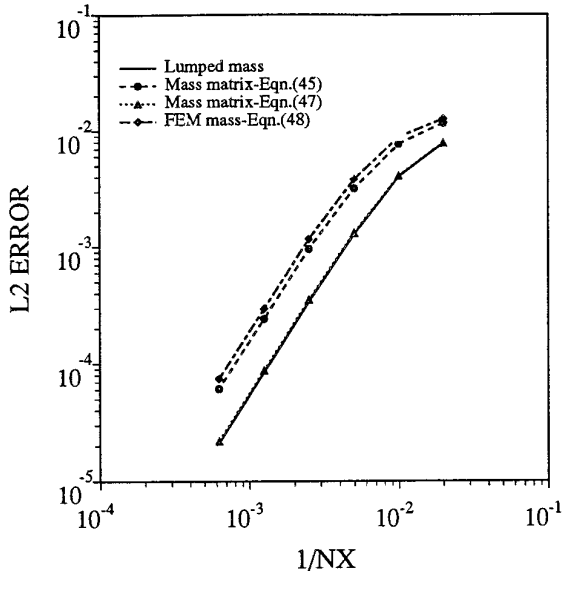

FIG. 5. L_2 norms of the errors with various schemes on a random grid in one dimension.

matrices given by Eq. (45) and Eq. (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 Eq. (45). The results obtained with the lumped mass matrix are almost identical to those obtained with Eq. (47). Taylor series expansion would imply a first-order error with the lumped mass matrix on a random grid, whereas Fig. 5 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 $h_i = x_i - x_{i-1}$,

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

are also shown in Fig. 5 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 (as also Eq. (47)) gives the lowest errors with this particular spatial discretization.

It is well known in finite element literature [39] 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

$$u^L_{i+1/2} = -\frac{h^2_{i+1}}{4h_i(h_i + h_{i+1})} u_{i-1} + \left(\frac{1}{2} + \frac{h_{i+1}}{4h_i}\right) u_i + \frac{h_{i+1} + 2h_i}{4(h_{i+1} + h_i)} u_{i+1}. \qquad (49)$$

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

$$a_1 u_{i-1} + a_2 u_i + a_3 u_{i+1},$$

$$a_1 = \frac{-2h^2_{i+1} + 2h_i h_{i+1} + h^2_i}{12h_i(h_i + h_{i+1})},$$

$$a_3 = \frac{h^2_{i+1} + 2h_i h_{i+1} - 2h^2_i}{12h_{i+1}(h_i + h_{i+1})},$$

$$a_2 = 1 - a_1 - a_3.$$

The standard Runge–Kutta scheme which is fourth-order accurate in time is used for the higher order computations.

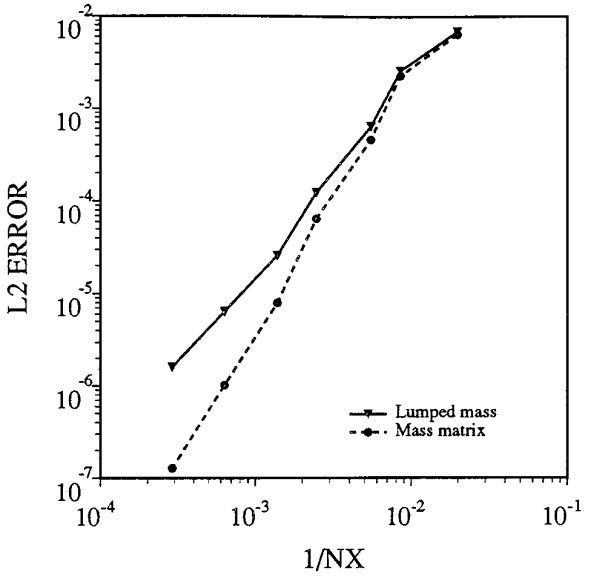

FIG. 6. L_2 norms of the errors with quadratic reconstruction on a random grid in one dimension.

Figure 6 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 (or less) 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 may 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 a 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

$$\alpha(t) = \alpha_m + \alpha_0 \sin(\omega t). \qquad (50)$$

For the test case chosen, $\alpha_m = 0.016°$, $\alpha_0 = 2.51°$, the reduced frequency $\kappa = \omega c/2U_\infty = 0.0814$ and the free-stream Mach number, $M_\infty = 0.755$. Computing this flow

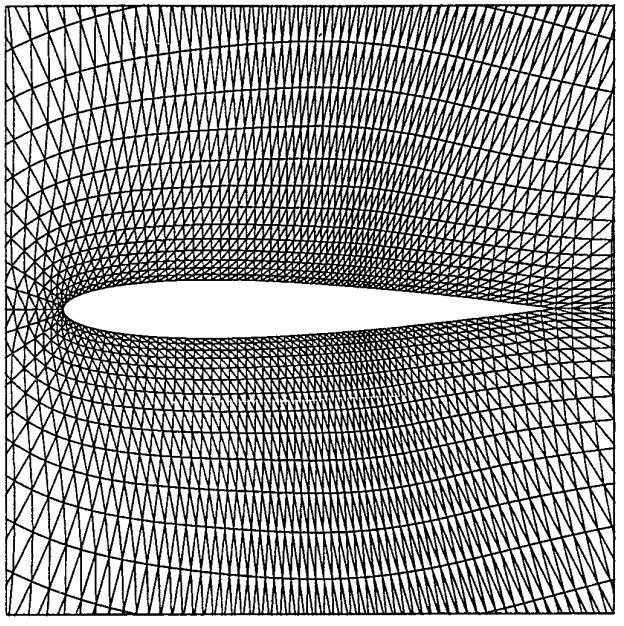

FIG. 7. GRID1 about an NACA0012 airfoil with 6336 vertices.

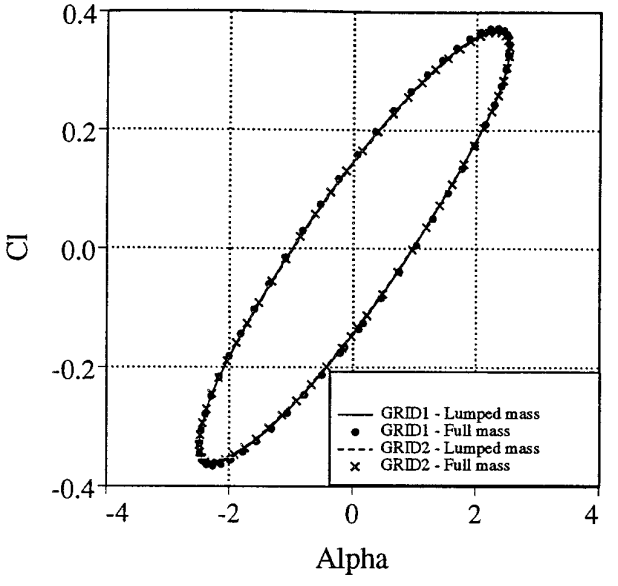

FIG. 9. Lift histories during the third cycle of motion.

using an explicit scheme is time-consuming because of the low frequency. The flow is computed using two meshes, referred to as GRID1 and GRID2, each having 6336 vertices. These are shown in Figs. 7 and 8, respectively. GRID1 is generated by drawing diagonals in a structured C-mesh and is fairly uniform. GRID2 is generated by symmetric random perturbations on GRID1. In GRID2, the centroids of the control volumes formed by the median dual are not represented well by the vertices. The grid moves rigidly with the airfoil. Figure 9 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 Eq. (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 10 shows the convergence of four-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. 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 in this paper have been computed with the lumped mass matrix.

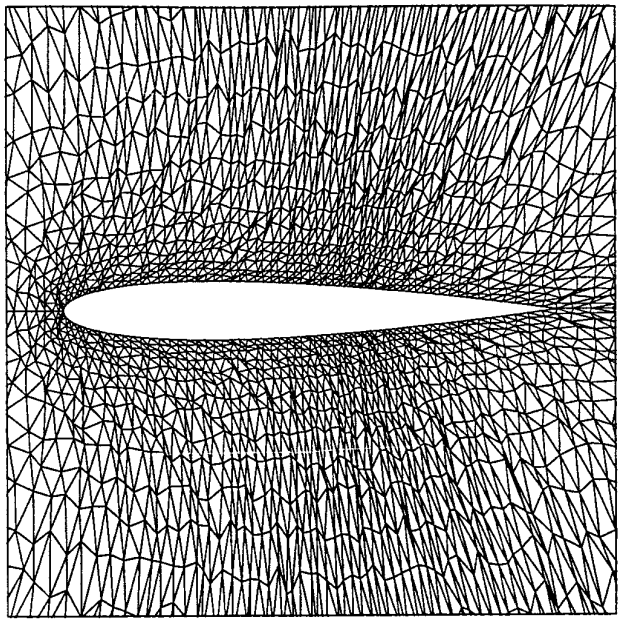

FIG. 8. GRID2 about an NACA0012 airfoil with 6336 vertices.

Finally, Figs. 11 and 12 show the effect of the physical time step size. Three 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 Eq. (27). The nodes are repositioned by the procedure described earlier. No swapping of edges occurred in this computation. The number of

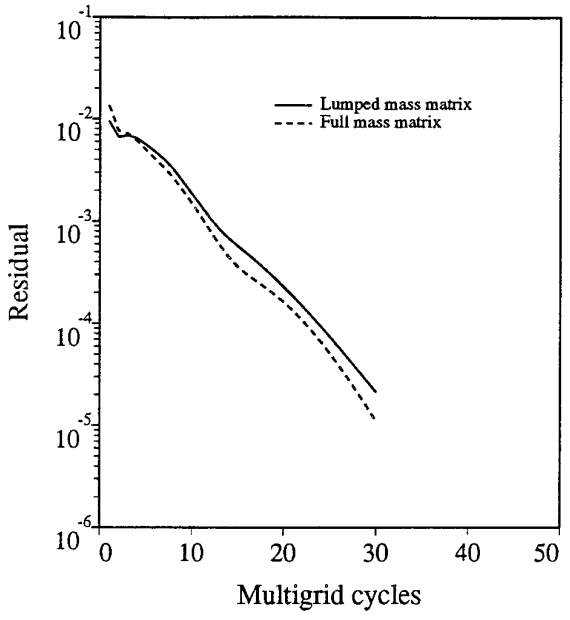

FIG. 10. Histories of the unsteady residual at a particular time step.

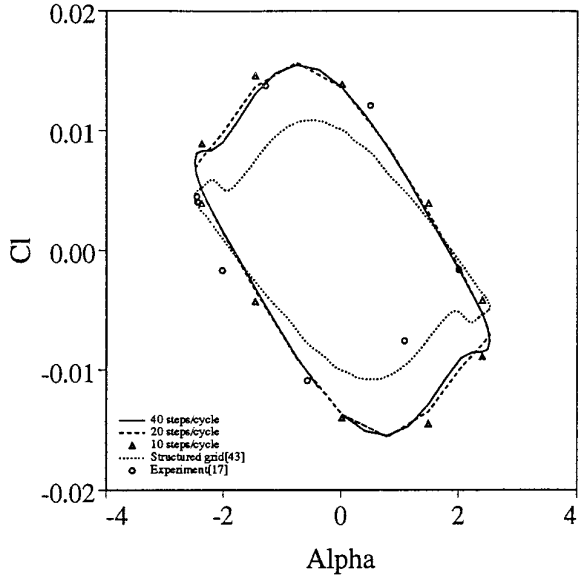

FIG. 12. Moment histories during the third cycle of motion with 40, 20, and 10 steps per cycle.

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 not improve the solution. Also shown is a comparison with the experimental data of Landon [17] as well as with a structured grid computation employing a TVD scheme on the same grid [43]. 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 a 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 Fig. 13. Figure 14 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 15 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 introduce errors of $O(\Delta x^2)$ which decrease as the grid is refined.

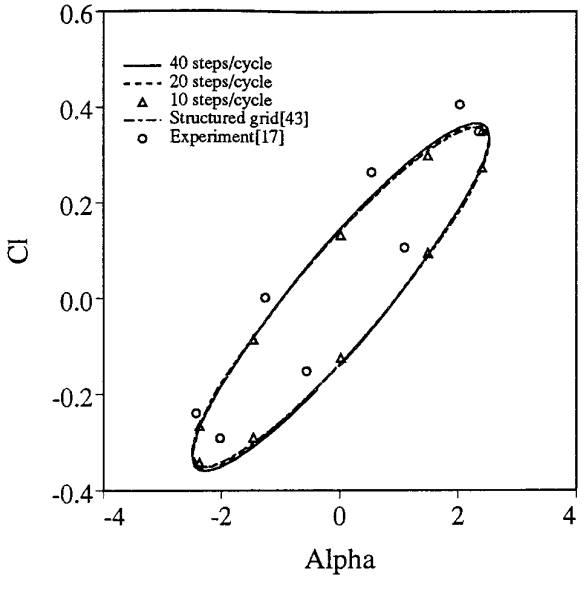

FIG. 11. Lift histories during the third cycle of motion with 40, 20, and 10 steps per cycle.

We next present a laminar unsteady calculation of impulsive start-up flow over a cylinder at a Reynolds number of 1200. Rumsey [35] has computed this flow using a structured grid code and has made detailed comparisons with experimental data [27]. We employ the same grid (192×64) as in [35], but divide the quadrilaterals into triangles.

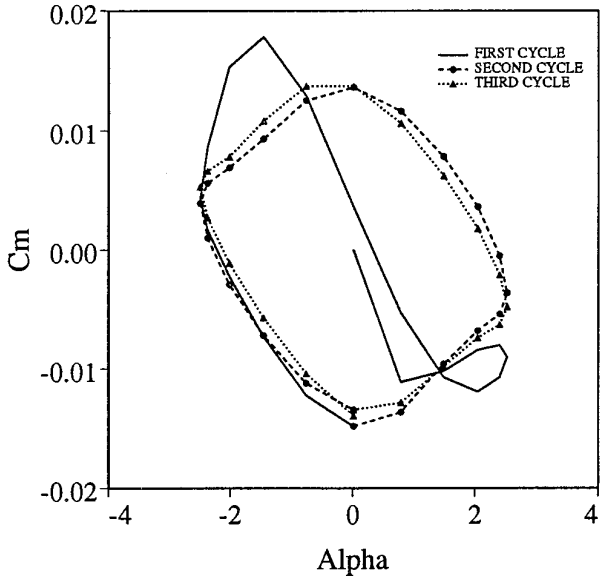

FIG. 13. Moment histories on the coarse O-mesh without swapping of edges.

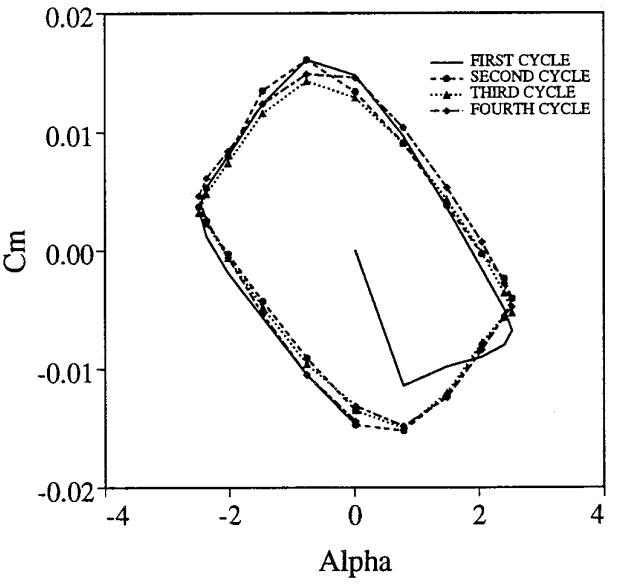

FIG. 15. Moment histories oan the fine O-mesh with forced swapping of edges.

Comparable accuracies are to be expected from the two approaches since the node-based unstructured grid solver has the same number of degrees of freedom as the structured grid. After an initial bubble-growth phase, the flow separates, and eventually, vortices are shed. 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 nondimensional time steps $\Delta\bar{t}$ is chosen as in [35]: $\Delta\bar{t} = 0.01$ 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 [23]. Seven multigrid iterations were used at each time step yielding about 2–3 orders of reduction in the unsteady residual. The centerline velocity at a time $\bar{t} = 2.9$ is plotted in Fig. 16, and the maximum reverse flow velocity is plotted

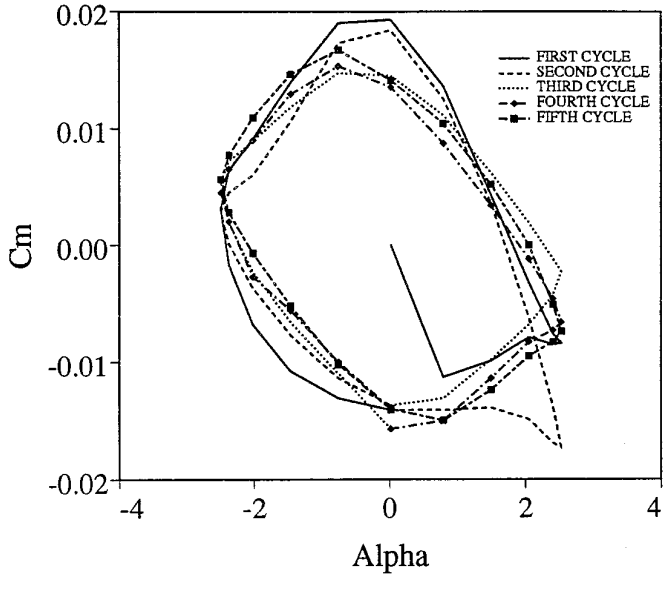

FIG. 14. Moment histories on the coarse O-mesh with forced swapping of edges.

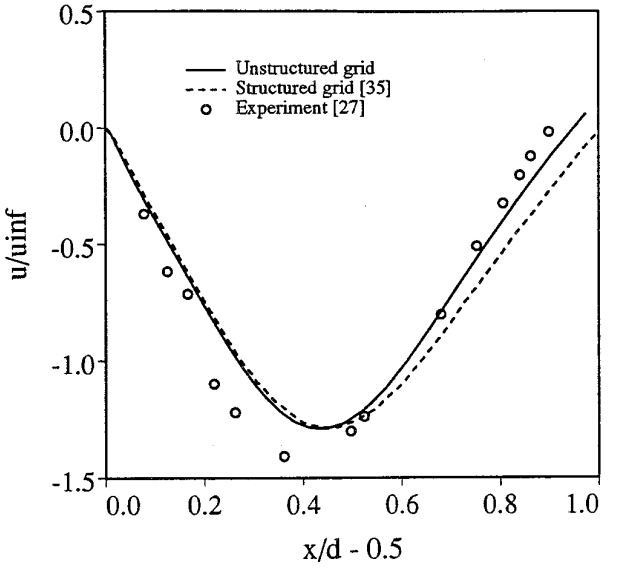

FIG. 16. Velocity distribution at $\bar{t} = 2.9$ on the symmetric axis of the wake.

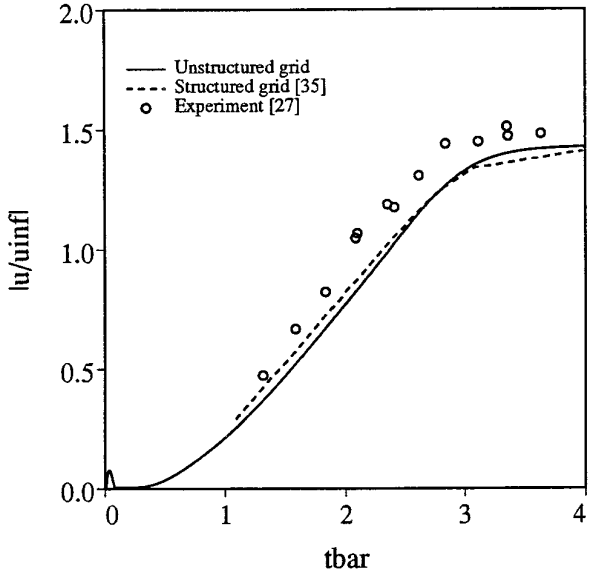

FIG. 17. Time history of the maximum reverse flow velocity on the symmetric axis of the wake.

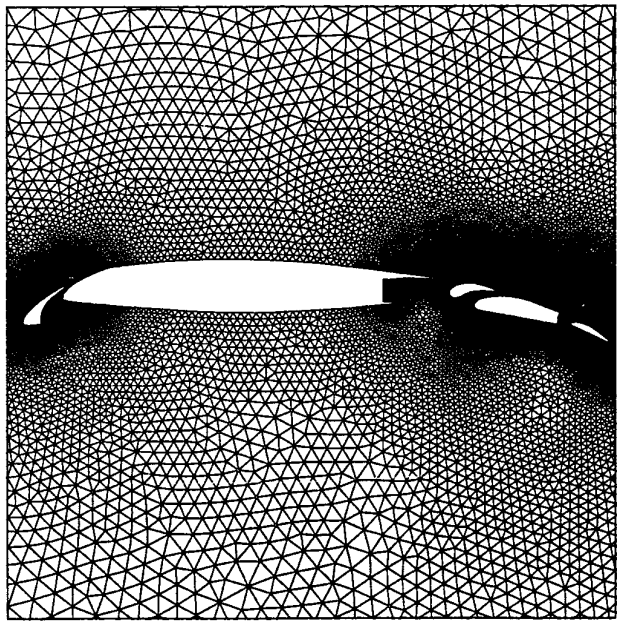

FIG. 19. Initial grid at the semi-deployed position (15° flap deflection).

in Fig. 17 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 because of the fact that it did not employ inner iterations. Thus, errors arising from factorization, linearization, and the mismatch of operators may be present in the structured grid solution. Figure 18 depicts the computed time histories of the total lift and the form drag coefficients; the oscillatory pattern corresponds to the shedding of vortices. 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.

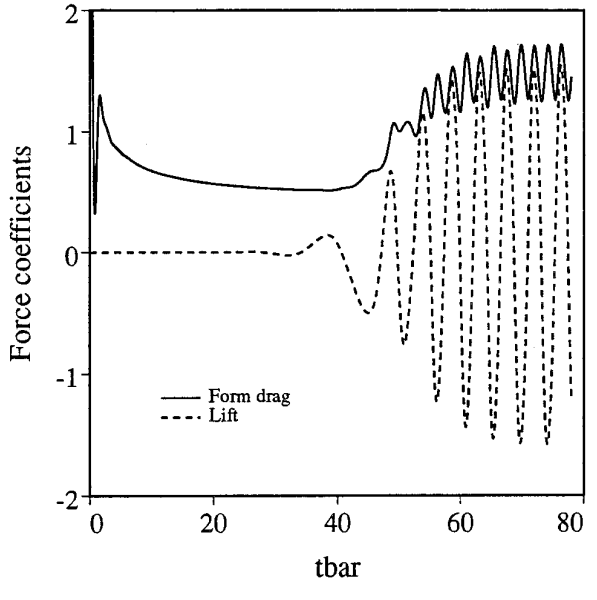

FIG. 18. Time histories of force coefficients.

The final test case represents an exploratory inviscid computation of flow over a multielement airfoil system undergoing deployment, as an example of flow over 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) [47]. The initial position corresponds to the 15° flap setting, while the final position 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 [22] at the 15° flap setting and is displayed in Fig. 19. The grid has 26,191 vertices. Figure 20 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. The time for deployment, defined in terms of nondimensional time unit $\bar{t} = l/\sqrt{p_\infty/\rho_\infty}$, is taken as 400 units. For the Boeing geometry, this corresponds to a deployment time of 5 s. Here l is the chord of the main element and p_∞, ρ_∞ are the freestream pressure and density. The flow is computed in 200 time steps. Thus, the size of each time step is 2 time units and corresponds to a CFL number of about 40,000. Twenty

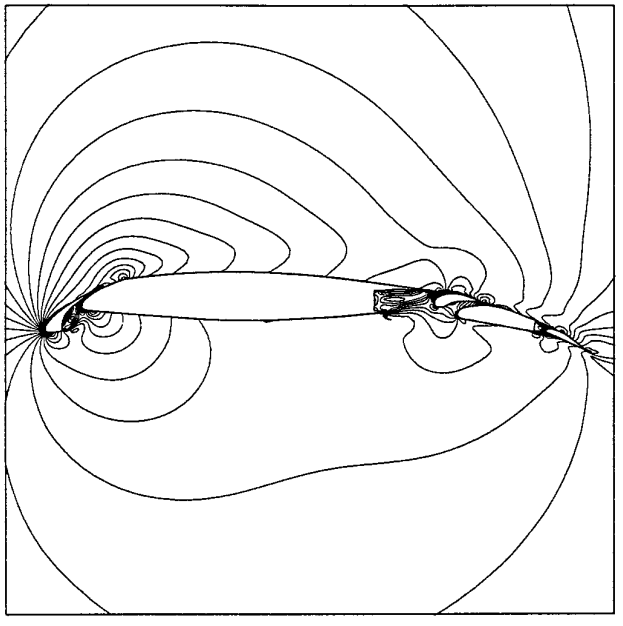

FIG. 20. Mach contours for steady flow at the initial position (15° flap deflection).

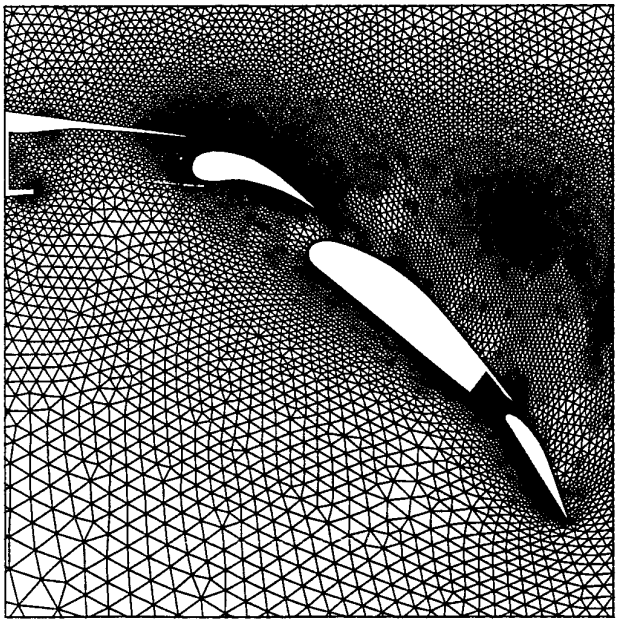

FIG. 21. Closeup view of the grid at full deployment (40° flap deflection).

multigrid cycles are employed at each time step. Each multigrid cycle is equivalent to about 1.33 iterations on the finest grid. Assuming a CFL number of 5 for an explicit scheme, the implicit scheme, for this choice of time step size, results in a savings of a factor of nearly 300.

A time-accurate solution is computed for another 50 time steps after the high-lift system reaches its final configuration. The grid restructuring involved presmoothing, spring analogy, and edge swapping. Figures 21 and 22 depict a closeup of the grid in the flap region and the instantaneous flow solution at full deployment at a nondimensional time of 400. Grid quality, while not as good as with the initial mesh, is acceptable, considering the large-scale motion. Figure 23 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 the 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 the 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.

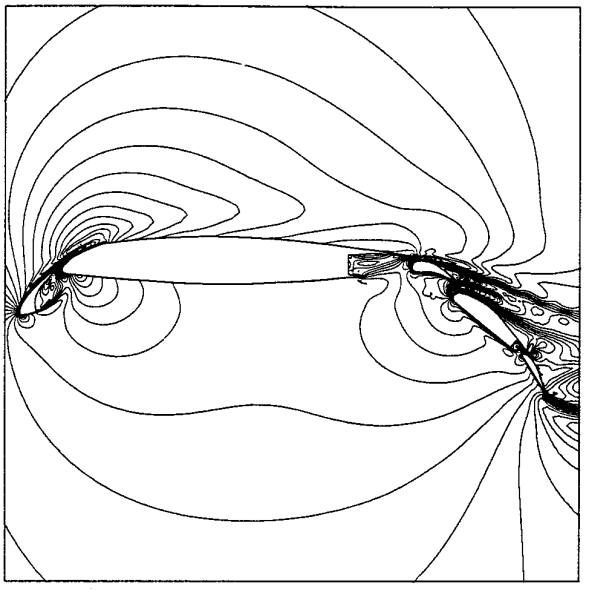

FIG. 22. Mach contours of the instantaneous solution at full deployment (40° flap deflection).

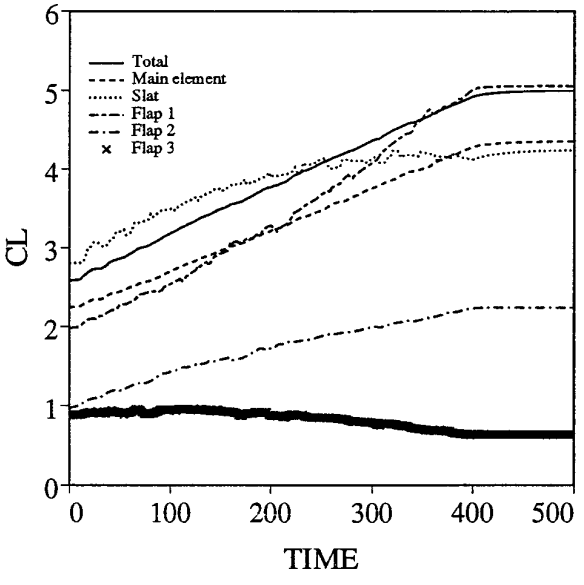

FIG. 23. Time history of the lift coefficients.

APPENDIX

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

1. four tetrahedra sharing a common interior vertex.
2. two tetrahedra sharing a common triangular face.
3. three 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 tetrahedrons formed [12]. Figure 24 illustrates the 2- and 3-tetrahedra configurations.

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

$$\begin{aligned} T_2 &= \tfrac{1}{4}[(W_1 + W_2 + W_3 + W_4)\mathcal{V}_{1234} \\ &\quad + (W_1 + W_2 + W_3 + W_5)\mathcal{V}_{1235}], \\ T_3 &= \tfrac{1}{4}[(W_1 + W_2 + W_4 + W_5)\mathcal{V}_{1245} \\ &\quad + (W_2 + W_3 + W_4 + W_5)\mathcal{V}_{2345} \\ &\quad + (W_3 + W_1 + W_4 + W_5)\mathcal{V}_{3145}]. \end{aligned}$$

Changes are now made to vertices 4 and 5 in a conservative manner by distributing the difference $T_2 - T_3$, using formulas similar to Eq. (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 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

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

and define

$$r = \min[r_1, r_2, r_3, \tfrac{1}{3}]$$

Now assume $r = r_1$. Then

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

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

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

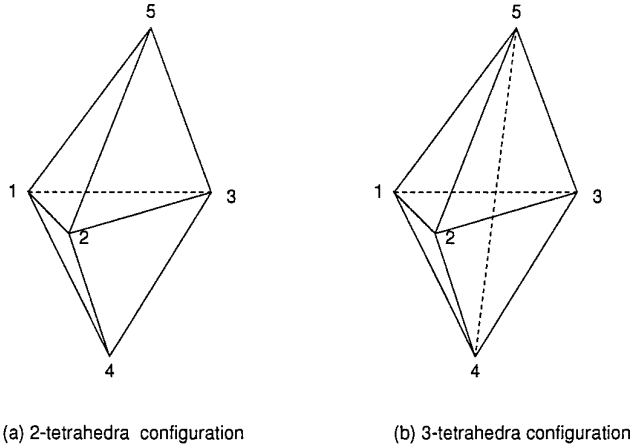

FIG. 24. Two possible tetrahedralizations of five vertices.

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

ACKNOWLEDGMENTS

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

REFERENCES

1. J. Alonso and A. Jameson, AIAA Paper 94-0056, Jan. 1994 (unpublished).
2. W. K. Anderson and D. L. Bonhaus, *Comput. Fluids* **23,** 1 (1994).
3. A. Arnone, M. Liou, and L. A. Povinelli, AIAA Paper 93-3361CP, June 1993 (unpublished).
4. T. J. Barth and S. W. Linton, AIAA Paper 95-0221, Jan. 1995 (unpublished).
5. J. T. Batina, *AIAA J.* **29,** 1836 (1991).
6. J. D. Baum, H. Luo, and R. Löhner, AIAA Paper 94-0414, Jan. 1994 (unpublished).
7. A. Brennis and A. Eberle, *Trans. ASME J. Fluids Engrg.* **112,** 510 (1990).
8. G. Dahlquist and A. Bjork, *Numerical Methods* (Prentice–Hall, Englewood Cliffs, NJ, 174).
9. G. A. Davis and O. O. Bendiksen, *AIAA J.* **31,** 1051 (1993).
10. T. J. R. Hughes, *Intl. J. Numer. Methods Fluids* **7,** 1261 (1987).
11. A. Jameson, AIAA Paper 91-1596, July 1991 (unpublished).
12. B. Joe, *SIAM J. Sci. Stat. Comput.* **10,** 718 (1989).
13. S. R. Kennon, J. M. Meyering, C. W. Berry, and J. T. Oden, AIAA Paper 92-4575, Aug. 1992 (unpublished).
14. W. L. Kleb, J. T. Batina, and M. H. Williams, *AIAA J.* **30,** 1980 (1992).
15. B. Koobus, M. H. Lallemand, and A. Dervieux, INRIA Report 1946, June 1993 (unpublished).
16. M. Lallemand, H. Steve, and A. Dervieux, *Comput. Fluids* **21,** 397 (1992).
17. R. H. Landon, Data Set 3 in AGARD-R-702, Compendium of Unsteady Aerodynamic measurements, Aug. 1982 (unpublished).
18. C. L. Lawson, "Software for C^1 Surface Interpolation," in *Mathematical Software III*, edited by J. R. Rice (Academic Press, New York, 1977), p. 161.
19. A. Lerat, J. Sides, and V. Daru, "Efficient Computation of Steady and Unsteady Transonic Flows by an Implicit Solver," in *Advances in Computational Transonics*, edited by W. G. Habashi, Recent Advances in Numerical Methods in Fluids, Vol. 4 (Pineridge Press, Swansea, 1985), p. 545.
20. R. Löhner, K. Morgan, and O. Zienkiewicz, *Comput. Methods Appl. Mech. Engrg.* **51,** 441 (1985).
21. D. J. Mavriplis, *AIAA J.* **30,** 1753 (1992).
22. D. J. Mavriplis, *J. Comput. Phys.* **117,** 90 (1995).
23. D. J. Mavriplis and V. Venkatakrishnan, *Comput. Fluids* **24,** 553 (1995).
24. 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*, p. 423.
25. K. Miller, *SIAM J. Numer. Anal.* **29,** 89 (1992).
26. K. Miller and R. N. Miller, *SIAM J. Numer. Anal.* **18,** 1019 (1981).
27. H. Nagata, H. Funada, and T. Matsui, JSME, **28,** 2608 (1985).
28. B. Nkonga and H. Guillard, *Comput. Methods Appl. Mech. Engrg.* **113,** 183 (1994).
29. B. Palmerio, *Comput. Fluids* **23,** 487 (1994).
30. J. Peraire, J. Peirö, and K. Morgan, AIAA Paper 92-0449, Jan. 1992 (unpublished).
31. T. H. Pulliam, AIAA Paper 93-3360CP, July 1993 (unpublished).
32. M. M. Rai, AIAA Paper 87-0543, Jan. 1987 (unpublished).
33. J. D. Ramshaw, *J. Comput. Phys.* **59,** 193 (1985).
34. S. E. Rogers, D. Kwak, and C. Kiris, *AIAA J.* **29,** 603 (1991).
35. C. L. Rumsey, *J. Fluids Engrg.* **110,** 447 (1988).
36. K. P. Singh, J. C. Newman, and O. Baysal, *AIAA J.* **33,** 641 (1995).
37. D. C. Slack, D. L. Whitaker, and R. W. Walters, *AIAA J.* **32,** 1158 (1994).
38. W. A. Smith, "Multigrid Solution of Transonic Flow on Unstructured Grids," in *Recent Advances and Applications in Computational Fluid Dynamics, Nov. 1990. Proceedings, ASME Winter Annual Meeting*, eidted by O. Baysal.
39. G. Strang and G. J. Fix, *An Analysis of the Finite Element Method* (Prentice–Hall, Englewood Cliffs, NJ, 1973).
40. P. D. Thomas and C. K. Lombard, *AIAA J.* **17,** 1030 (1979).
41. B. van Leer, *J. Comput. Phys.* **32,** 101 (1979).
42. J. C. Vassberg, AIAA Paper 92-2693, 1992 (unpublished).
43. V. Venkatakrishnan and A. Jameson, *AIAA J.* **26,** 974 (1988).
44. V. Venkatakrishnan and D. J. Mavriplis, *J. Comput. Phys.* **105,** 83 (1993).
45. V. Venkatakrishnan and D. J. Mavriplis, *AIAA J.* **33,** 633 (1995).
46. A. Wathen, *SIAM J. Numer. Anal.* **23,** 797 (1986).
47. L. P. Yip, P. Vijgen, J. D. Hardin, and C. P. van Dam, AIAA Paper 92-4103, Aug. 1992 (unpublished).
48. H. Zhang, M. Reggio, J. Y. Trepanier, and R. Camarero, *Comput. Fluids* **22,** 9 (1993).