Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 201 (2004) 119–147
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A gridless boundary condition method for the solution of the Euler equations on embedded Cartesian meshes with multigrid

D.J. Kirshman *, F. Liu

Department of Mechanical & Aerospace Engineering, University of California, Irvine, CA 92697, USA

Received 8 September 2003; received in revised form 15 March 2004; accepted 6 May 2004
Available online 24 June 2004

Abstract

The solution of the Euler equations using a "gridless" boundary condition treatment on a patched, embedded Cartesian field mesh is described. The gridless boundary treatment is implemented by means of a least squares fitting of the conserved flux variables using a cloud of nodes in the vicinity of the body. The method allows for accurate treatment of the surface boundary conditions without the need for excessive refinement of the Cartesian mesh. Additionally, the method does not suffer from problems associated with thin body geometry or extremely fine cut cells near the body. Unlike some methods that consider a gridless (or "meshless") treatment throughout the entire domain, the use of a Cartesian field mesh allows for effective implementation of multigrid acceleration, and issues associated with global conservation are alleviated. Results are presented for transonic flow about single and dual NACA 0012 airfoil configurations including a convergence study indicating the effectiveness of multigrid acceleration within the construct of the gridless boundary treatment. Where applicable, comparisons to the FLO52 body-fitted Euler code are presented to gauge the accuracy of the method.

Keywords: Cartesian mesh; Gridless; Meshless; Embedded mesh; Euler equations, numerical boundary conditions; Transonic flow

1. Introduction

Solution methods using a Cartesian mesh in computational fluid dynamics simulations are popular with many investigators [1–17], and have numerous inherent advantages. These include simple and efficient field mesh generation, superior implementation of high order discretization schemes, minimal phase error associated with shock-capturing calculations, and an absence of issues associated with mesh skewness and distortion. However, an obvious difficulty with the Cartesian approach is the implementation of solid wall boundary conditions. Boundary treatment can be especially problematic for "thin body" regions such as airfoils, turning vanes, or fins, in which the body geometry is much thinner than the local cell size, such that

* Corresponding author.
E-mail addresses: dkirshman@socal.rr.com (D.J. Kirshman), fliu@uci.edu (F. Liu).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.05.006

Nomenclature

c	acoustic speed
E	total energy
$\mathbf{f}$	convective flux vector in x-direction
f	component of flux vector in x-direction
$\mathbf{g}$	convective flux vector in y-direction
g	component of flux vector in y-direction
H	total enthalpy
i, j	mesh indices
n	time level index
p	static pressure
$\mathbf{p}$	polynomial basis vector
$\mathbf{Q}_F$	multigrid forcing function
$\mathbf{R}$	residual vector
$\bar{R}$	ideal gas constant
t	time
T	temperature
$\mathbf{u}$	vector of conserved variables
u, v	Cartesian velocity components
x, y	Cartesian coordinates
α	Runge–Kutta stage coefficients
Δx	local grid spacing
Δt	time step
ε	smoothing coefficient
γ	ratio of specific heats
ϕ_n	least squares shape function at node n
ρ	density
σ	CFL number

multiple flow regions are formed within a single cell (see Fig. 1). Additionally, boundary treatments that incorporate "cut cells" near the body/grid interface can suffer significant convergence problems when very small cells are formed. Problematic occurrences such as these are depicted in Fig. 1, which is typical near the trailing edge of an airfoil.

Clarke et al. [2] performed Euler flow calculations for multi-element airfoils using Cartesian grids. They incorporated a finite volume procedure in which fractional cells are formed at the intersection of the mesh and the body geometry. Flow properties in the fractional cells are extrapolated from values in adjacent uncut cells. The method suffers in instances when the cut cells become very small, and merging of adjacent cells near the body was incorporated to alleviate the problem, should the cut cell size fall below 25% (to 50%) of the Cartesian cell size, complicating the representation of the cells near the boundary. Gaffney et al. [3] conducted similar work considering Euler simulation about aircraft wings.

Instead of a finite-volume approach, Tidd et al. [4] used a finite difference formulation to treat the boundary conditions. They introduced the concept of a "multi-valued dummy point". These are nodes that belong to the finite difference stencil of another node, but are hidden from the other node by the surface geometry. Because of the complexities associated with the need for multi-valued nodes using finite difference methods, most current research considers finite volume approaches.

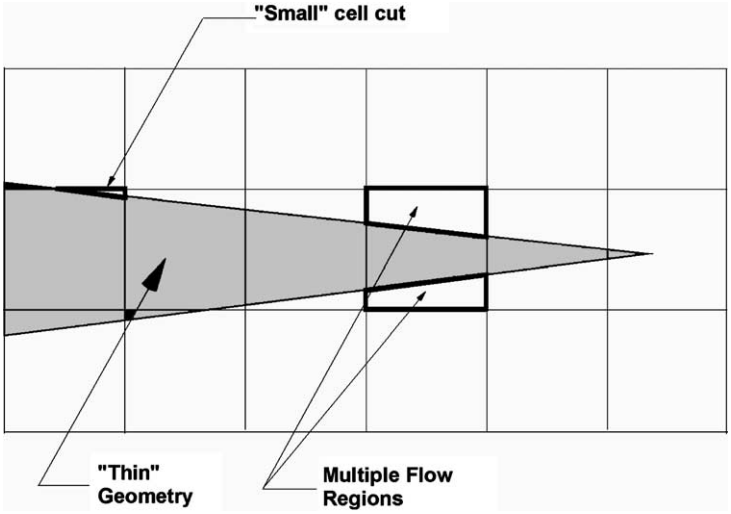

Fig. 1. Problematic thin body geometry.

One of the more prominent methods is that of Leveque [5] and Leveque and Berger [6,7]. The primary objective of their work was to provide a means of alleviating the restrictive time step limitations associated with small cut cells. Namely, they developed a technique in which the cut cell time step is governed by the size of the uncut cells. In the earlier work of Leveque [5], this was achieved by extending the domain of dependence of the cell adjacent to the wall. That is, the effects of waves reflecting off the wall were allowed to penetrate cells other than those directly adjacent to the wall during one time step. Extension of this method from one to multiple dimensions, however, resulted in stability issues in some cases. As a result, the approach was revised by Leveque and Berger [6,7], in which the surface properties are evaluated by means of a rotated Riemann problem in order to provide flux cancellation. The approach is quite involved requiring the application of two separate Riemann problems using ghost cells identical is size to the uncut cells, one parallel and one normal to the wall, in which reflected properties are used to form the fluxes along each of the cut faces. A similar method using rotated cells for a Riemann solution near the boundary was also proposed by Forrer [8].

De Zeeuw and Powell [9] considered a method using a Gauss–Green linear reconstruction in conjunction with Roe's approximate Riemann solver and local time stepping. They reported that diminutive cut cells near the boundary did not significantly degrade the accuracy or stability of the scheme. In instances when the sizes of the cut cells are of the same order of machine accuracy, the body is "pulled out" to the node thus eliminating the cell. Coirier and Powell [12] investigated the accuracy of such methods with comparison to exact solutions of a transonic expansion. Also investigated was the effect of cell merging near the wall to reduce stiffness of the solution. Similar efforts using cell merging were performed by Quirk [13] in which cut cells near the boundaries with areas less than half of uncut cells were merged with a suitable neighboring cell.

In an attempt to achieve higher order solutions at the fluid boundary interface, Forrer and Jeltsch [14] proposed a revision to their earlier treatment [8]. That is, instead of using rotated cells for evaluation of the corrective Riemann flux, the geometry of the cut cell (including the local velocity field) is reflected across the intersection of the surface and Cartesian mesh. Another technique using reflected ghost cells is that of Dadone and Grossman [16,17]. In their work, flow properties at the ghost cells are determined based on an assumed flow field model consisting of an isentropic vortex of constant total enthalpy, satisfying the normal momentum equation such that normal velocity at the wall vanishes. The solution proceeds adopting a finite

volume approach using fluxes at surrounding cell faces, with cells near the surface using the ghost nodes without any other boundary condition required. As in other methods that require reflected cells at the boundary, the method is problematic when considering thin geometry.

An alternative approach to reducing the effort associated with mesh generation is the use of a purely "gridless" approach for the entire computational domain [18–24]. Such methods typically incorporate either a direct least squares fitting of the flux variables, or may employ a moving least squares fitting to establish trial and test functions in a finite element formulation. Such formulations have been implemented successfully within the field of fracture mechanics and crack propagation [25]. For compressible fluid dynamic applications, however, there are typically issues associated with global conservation of mass, momentum, and energy. Another disadvantage of purely gridless schemes is difficulties in efficiently incorporating acceleration techniques such as multigrid.

A method is presented in this paper that incorporates a compromise between the above-mentioned strategies. Namely, the use of a Cartesian field mesh while incorporating a gridless treatment for the surface boundary conditions is proposed. Application of the gridless boundary treatment requires the evaluation of shape functions with least squares approximations of flow field gradients in the vicinity of the surface. To form the shape functions, a "cloud" of nodes is chosen which may or may not be directly associated with the Cartesian grid. That is, surface nodes beyond those formed by the intersection of the body geometry and Cartesian mesh may be included. Furthermore, additional adaptive cloud nodes, unassociated with either the surface or field mesh, may also be introduced to facilitate shape function evaluation in regions where the nodal density is insufficient. Though nodes associated with the field grid are incorporated into the boundary treatment, their geometric connectivity with the field is not retained for purposes of obtaining surface properties.

Within the construct of the gridless boundary condition treatment, the method presented here incorporates the following capabilities and features: (1) embedded grid patches, which may intersect the body, to add local mesh refinement, (2) multiple body geometry, (3) surface discontinuity (airfoil trailing edge) treatment, (4) incorporation of adaptive cloud nodes unassociated with the surface or field mesh, and (5) the ability to use an unaligned Cartesian mesh in which the airfoil trailing edge does not directly lie on a grid line. In addition to discussions of these features, a detailed description of the multigrid implementation for the embedded grids is discussed, along with presentation of the convergence characteristics. For the single airfoil cases, comparisons with a body fitted solution using Jameson's FLO52 code are provided to gauge the accuracy of the method.

2. Governing equations

In this study, the two-dimensional flow of an inviscid, compressible gas is considered. Such flows are governed by the Euler equations, which provide for the conservation of mass, momentum, and energy, and are given by

$$\frac{\partial \mathbf{u}}{\partial t}+\frac{\partial \mathbf{f}}{\partial x}+\frac{\partial \mathbf{g}}{\partial y}=0, \tag{1}$$

where

$$\mathbf{u}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\end{bmatrix}\quad \mathbf{f}=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uH\end{bmatrix}\quad \mathbf{g}=\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vH\end{bmatrix}. \tag{2}$$

For an ideal gas, the total energy and total enthalpy can be written

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

$$H = E + \frac{p}{\rho}. \tag{4}$$

Additionally, the equation of state is given by

$$p = \rho \bar{R} T. \tag{5}$$

3. Flow field discretization

In this work, the entire flow domain is discretized using a purely Cartesian mesh, which is generated independently of the body. In order to solve for the flow field, a finite difference scheme using Van Leer flux vector splitting is performed [26]. In this scheme, the convective flux vectors **f** and **g** are decomposed into "upwind" and "downwind" components based on the sign of the eigenvalues of the system of governing equations. Namely, at each grid point, the convective flux vectors are written as

$$\begin{aligned} \mathbf{f} &= \mathbf{f}^+ + \mathbf{f}^-, \\ \mathbf{g} &= \mathbf{g}^+ + \mathbf{g}^-, \end{aligned} \tag{6}$$

where the positive and negative superscripts indicate the portion of the total flux that travels in the positive and negative coordinate directions, respectively. For supersonic flow, there is no splitting and the entire flux quantity travels in the downstream direction. For subsonic flow (in a given coordinate direction), the flux vectors are given by

$$\mathbf{f}^{\pm} = \pm \frac{\rho}{4c(u \pm c)^2} \begin{bmatrix} 1 \\ \frac{1}{\gamma}[u(\gamma - 1) \pm 2c] \\ v \\ \frac{[(\gamma-1)u \pm 2c]^2}{(\gamma^2-1)} + \frac{u^2+v^2}{2} \end{bmatrix}, \tag{7}$$

$$\mathbf{g}^{\pm} = \pm \frac{\rho}{4c(v \pm c)^2} \begin{bmatrix} 1 \\ u \\ \frac{1}{\gamma}[v(\gamma - 1) \pm 2c] \\ \frac{[(\gamma-1)v \pm 2c]^2}{(\gamma^2-1)} + \frac{u^2+v^2}{2} \end{bmatrix}. \tag{8}$$

The governing equations can then be re-written as

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}^+}{\partial x} + \frac{\partial \mathbf{f}^-}{\partial x} + \frac{\partial \mathbf{g}^+}{\partial y} + \frac{\partial \mathbf{g}^-}{\partial y} = 0, \tag{9}$$

which can be discretized into a finite difference formulation. Considering the grid location defined by the nodal indices i and j in the x- and y-directions, respectively, a three point second order upwind discretization is written

$$\begin{aligned}\frac{\partial \mathbf{f}^+}{\partial x} &= \frac{1}{2\Delta x}\left(3\mathbf{f}_i^+ - 4\mathbf{f}_{i-1}^+ + \mathbf{f}_{i-2}^+\right),\\ \frac{\partial \mathbf{f}^-}{\partial x} &= \frac{1}{2\Delta x}\left(-3\mathbf{f}_i^- + 4\mathbf{f}_{i+1}^- - \mathbf{f}_{i+2}^-\right).\end{aligned} \tag{10}$$

Generalizing to two dimensions, the semi discrete form of Eq. (9) can be written in terms of a first order difference and a limited second order correction as

$$\begin{aligned}\frac{\mathrm{d}\mathbf{u}_{i,j}}{\mathrm{d}t} = &-\frac{1}{2}\left[\left(2+\Psi\left(r_{i+\frac{1}{2}}^+\right)\right)\delta_x^+\mathbf{f}_{i,j}^+ - \Psi\left(r_{i-\frac{3}{2}}^+\right)\delta_x^+\mathbf{f}_{i-1,j}^+\right] - \frac{1}{2}\left[\left(2+\Psi\left(r_{i-\frac{1}{2}}^-\right)\right)\delta_x^-\mathbf{f}_{i,j}^- - \Psi\left(r_{i+\frac{3}{2}}^-\right)\delta_x^-\mathbf{f}_{i+1,j}^-\right]\\ &-\frac{1}{2}\left[\left(2+\Psi\left(r_{j+\frac{1}{2}}^+\right)\right)\delta_y^+\mathbf{g}_{i,j}^+ - \Psi\left(r_{j-\frac{3}{2}}^+\right)\delta_y^+\mathbf{g}_{i,j-1}^+\right] - \frac{1}{2}\left[\left(2+\Psi\left(r_{j-\frac{1}{2}}^-\right)\right)\delta_y^-\mathbf{g}_{i,j}^- - \Psi\left(r_{j+\frac{3}{2}}^-\right)\delta_y^-\mathbf{g}_{i,j+1}^-\right],\end{aligned} \tag{11}$$

where the δ terms are first order upwind fluxes

$$\delta_x^+ f_{i,j}^+ = \frac{f_{i,j}^+ - f_{i-1,j}^+}{\Delta x} \quad \delta_x^- f_{i,j}^- = \frac{f_{i+1,j}^- - f_{i,j}^-}{\Delta x} \tag{12}$$

and similarly for the y-direction. The flux limiter, $\Psi(r)$ in the above is implemented to suppress oscillations of the second order solution in high gradient regions (i.e. shocks). In the work presented here, the standard minmod limiter is incorporated in which

$$\Psi(r) = \begin{cases} 1 & r \geqslant 1,\\ r & 0 \leqslant r \leqslant 1,\\ 0 & r < 0,\end{cases} \tag{13}$$

where

$$\begin{aligned} r_{i+\frac{1}{2}}^+ &= \frac{f_{i+2,j}^+ - f_{i+1,j}^+}{f_{i+1,j}^+ - f_{i,j}^+} \quad r_{i+\frac{1}{2}}^- = \frac{f_{i,j}^- - f_{i-1,j}^-}{f_{i+1,j}^- - f_{i,j}^-}\\ r_{j+\frac{1}{2}}^+ &= \frac{g_{i,j+2}^+ - g_{i,j+1}^+}{g_{i,j+1}^+ - g_{i,j}^+} \quad r_{j+\frac{1}{2}}^- = \frac{g_{i,j}^- - g_{i,j-1}^-}{g_{i,j+1}^- - g_{i,j}^-}.\end{aligned} \tag{14}$$

4. Gridless surface boundary treatment

4.1. "Gridless" node selection and definition

In order to characterize the region of the flow field in the vicinity of the body, the locations where the surface geometry intersects the Cartesian grid must first be evaluated. To facilitate this process, the body is described by a series of surface definition nodes through which a spline curve-fit is created. Once the intersection of the body with the Cartesian grid has been determined, the field nodes are then categorized into three types: those far from the body which can be treated as field points using the finite difference expression, Eq. (11); those embedded within the body, or located very close to the surface (within say $0.3\Delta x$), which are removed from the calculation; and those near the body which do not have a complete computational stencil to apply Eq. (11). This last category of field nodes is to be solved with the gridless method

using least squares approximation. Also, all nodes associated with the body geometry are treated using the gridless method. Namely, the surface nodes formed by the intersection of the body and field mesh, as well as any independent surface definition nodes that may be desired to ensure that intricacies in the body geometry are retained in the solution. Finally, additional adaptive "cloud" nodes may be introduced into the solution to facilitate the least squares approximation or to increase the local grid resolution. Fig. 2 provides a graphical representation of the various types of nodes described above.

4.2. Derivation of gridless shape functions

Implementation of the gridless boundary method is facilitated by the use of nodal shape functions. Namely, the effect of each node in the vicinity of a particular location is weighted by the value of its shape function. These shape functions are determined using a weighted least squares approximation in each of the coordinate directions. Flow properties at a particular location are then evaluated by the sum of the products of the shape function and flow properties at the surrounding nodes. Derivation of these shape functions follows. Belytschko et al. [25] presents a similar derivation for application to structures analysis using element-free Galerkin methods.

Considering a gridless node p, either on the body or in the field, a "cloud" of N nodes in the vicinity of p are chosen so that a weighted least squares approximation can be made. In order to establish the approximation, a polynomial basis vector is first chosen so that any particular component of a conserved flux variable, say f, can be approximated by a least squares interpolant f^h of M terms given by

$$f^h(x,y) = \sum_{m=1}^{M} p_m(x,y)a_m = \mathbf{p}^{\mathrm{T}}(x,y)\mathbf{a}, \tag{15}$$

where each p_m is a monomial in the space coordinates providing a basis vector given by

$$\mathbf{p}^{\mathrm{T}}(x,y) = [1, x, y, xy, x^2, y^2, \ldots] \tag{16}$$

and $\mathbf{a}$ is the vector of polynomial coefficients. The basis vector $\mathbf{p}$ can have as many terms as desired. Increasing the number of terms yields higher order approximations at the expense of an increased number of

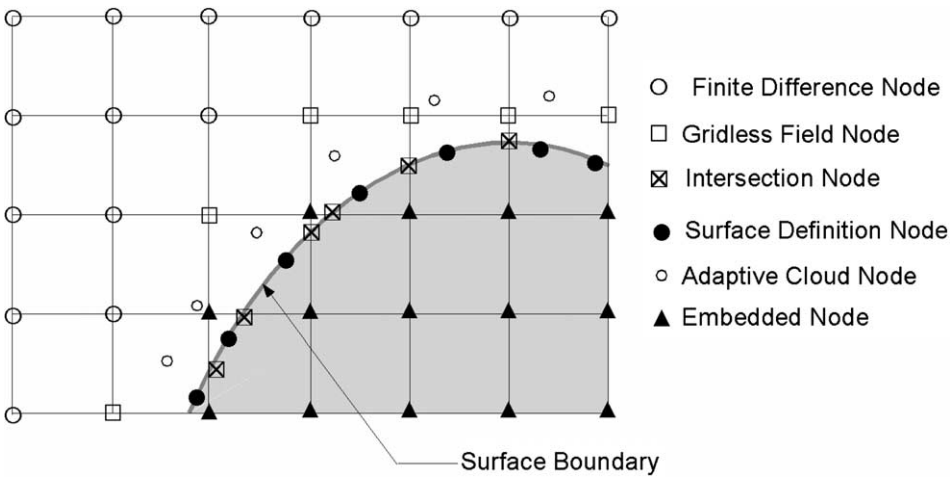

Fig. 2. Nodal assignments for gridless boundary treatment.

cloud nodes required to perform the least squares fit, as well as an increased computational effort. At a minimum, one must have as many cloud nodes as there are terms in **p**. It is also required that the points not be collinear, in which case the least squares matrix becomes singular. Increasing the number of cloud nodes beyond that of terms in **p** yields introduces extra diffusion into the solution. However, this is sometimes necessary in order to establish a non-singular least squares matrix.

A weighted least squares interpolant is found at the location of any pth node by minimizing

$$I = \sum_{n=1}^{N} w(x_n - x_p, y_n - y_p)[\mathbf{p}^{\mathrm{T}}(x_n, y_n)\mathbf{a} - f_n]^2, \tag{17}$$

where N is the number of nodes in the cloud of points considered, and f_n is the value of the flux variable at node n. The weight function, w, is of arbitrary form, though it should decrease in magnitude with the relative distance between the location of the node of interest (x_p, y_p) and the location of any given cloud node (x_n, y_n). For w of unity, one retains a standard least squares formulation.

The stationarity of I with respect to the coefficient vector **a** leads to a set of M equations given by

$$\frac{\partial I}{\partial a_m} = \sum_{n=1}^{N} w_n[\mathbf{p}^{\mathrm{T}}(x_n, y_n)\mathbf{a} - f_n]p_m(x_n, y_n) = 0, \tag{18}$$

where $w_n \equiv w(x_n - x_p, y_n - y_p)$ decreases with the distance from n to p. The solution for the coefficient vector is then,

$$\mathbf{a} = \mathbf{A}^{-1}\mathbf{B}\mathbf{f}, \tag{19}$$

where

$$\begin{aligned}
&\mathbf{a}^{\mathrm{T}} = [a_1 \cdots a_M] \\
&\mathbf{A} = \sum_{n=1}^{N} w_n \mathbf{p}^{\mathrm{T}}(x_n, y_n)\mathbf{p}(x_n, y_n), \\
&\mathbf{B} = [w_1\mathbf{p}(x_1, y_1) \cdots w_N\mathbf{p}(x_N, y_N)], \\
&\mathbf{f}^{\mathrm{T}} = [f_1 \cdots f_N].
\end{aligned} \tag{20}$$

The interpolant can then be written at location (x_p, y_p) as

$$f^h(x_p, y_p) = \mathbf{p}^{\mathrm{T}}(x_p, y_p)\mathbf{a} = \mathbf{p}^{\mathrm{T}}(x_p, y_p)\mathbf{A}^{-1}\mathbf{B}\mathbf{f} = \sum_{n=1}^{N}\sum_{m=1}^{M} p_m(x_p, y_p)[\mathbf{A}^{-1}\mathbf{B}]_{mn} f_n, \tag{21}$$

or in terms of a shape function ϕ as

$$\begin{aligned}
&f^h(x_p, y_p) = \sum_{n=1}^{N} \phi_n(x_p, y_p) f_n, \\
&\phi_n(x_p, y_p) \equiv \sum_{m=1}^{M} p_m(x_p, y_p)[\mathbf{A}^{-1}\mathbf{B}]_{mn}.
\end{aligned} \tag{22}$$

Finally, the spatial derivative of the flux in the kth direction is written in terms of a derivative shape function as

$$\begin{aligned}\frac{\partial f^h(x_p,y_p)}{\partial x_k} &= \sum_{n=1}^{N} \frac{\partial \phi_n(x_p,y_p)}{\partial x_k} f_n,\\ \frac{\partial \phi_n(x_p,y_p)}{\partial x_k} &\equiv \sum_{m=1}^{M} \frac{\partial p_m(x_p,y_p)}{\partial x_k} [\mathbf{A}^{-1}\mathbf{B}]_{mn}.\end{aligned} \tag{23}$$

It is pointed out that the shape functions have the following property

$$\sum_{n=1}^{N} \phi_n = 1; \quad \sum_{n=1}^{N} \frac{\partial \phi_n}{\partial x_k} = 0. \tag{24}$$

In deriving the shape functions, the N cloud nodes are selected from the appropriate flux direction consistent with the split flux component under consideration. Calculation of the shape functions as a preprocessing activity allows for fast evaluation of the flux gradients without having to perform least squares fitting at every time step.

4.3. Gridless node discretization

For field nodes near the body designated as "gridless", since they may not have a complete computational stencil in certain flux directions to apply Eq. (11), flux discretization in those directions are simply substituted with expressions based on the shape functions given by

$$\begin{aligned}\frac{\partial \mathbf{f}^{\pm}}{\partial x} &= \sum_{n=1}^{N_x^{\mp}} \frac{\partial \phi_n^{\pm}}{\partial x} \mathbf{f}_n^{\pm},\\ \frac{\partial \mathbf{g}^{\pm}}{\partial y} &= \sum_{n=1}^{N_y^{\mp}} \frac{\partial \phi_n^{\pm}}{\partial y} \mathbf{g}_n^{\pm},\end{aligned} \tag{25}$$

where the upper summation index indicates that gridless clouds are formed from the appropriate flux direction. For instance, N_x^+ indicates that gridless shape functions are formed from nodes located in the positive x-direction, relative to the point of interest.

For nodes on the surface, a local wall normal coordinate system $(\tilde{x},\tilde{y})$ rotated from the general coordinate system, in which the $+\tilde{y}$ direction is oriented out of the surface. One can specify gradients of flow properties normal to the wall as boundary conditions. Namely,

$$\begin{aligned}&\frac{\partial \rho}{\partial \tilde{y}} = 0,\\ &\frac{\partial \tilde{u}}{\partial \tilde{y}} = 0,\\ &\tilde{v} = 0,\\ &\frac{\partial p}{\partial \tilde{y}} = \frac{\rho \tilde{u}^2}{R},\end{aligned} \tag{26}$$

where the first three equations are reflection conditions, and the last equation represents a balance between the pressure in the fluid and the centrifugal force associated with the fluid motion along a curved path defined by the local surface radius of curvature, R. The tildes indicate that the velocity components are in the local (surface normal) coordinate system. The surface boundary conditions of Eq. (26) can be written in terms of gridless shape functions at any surface node p by

$$
\begin{aligned}
\rho_p &= -\frac{1}{\partial\phi_p^-/\partial\tilde{y}}\left(\sum_{\substack{n=1\\ n\neq p}}^{N_y^+}\frac{\partial\phi_n^-}{\partial\tilde{y}}\rho_n\right),\\
\tilde{u}_p &= -\frac{1}{\partial\phi_p^-/\partial\tilde{y}}\left(\sum_{\substack{n=1\\ n\neq p}}^{N_y^+}\frac{\partial\phi_n^-}{\partial\tilde{y}}\tilde{u}_n\right),\\
\tilde{v}_p &= 0,\\
p_p &= \frac{1}{\partial\phi_p^-/\partial\tilde{y}}\left(\frac{\rho_p\tilde{u}_p^2}{R_p}-\sum_{\substack{n=1\\ n\neq p}}^{N_y^+}\frac{\partial\phi_n^-}{\partial\tilde{y}}p_n\right).
\end{aligned}
\tag{27}
$$

The above equations are evaluated subsequent to evaluation of the field nodes after each time step. Note that, in the boundary formulation of Eq. (27), only shape functions for the flux direction normal to the wall ($\tilde{y}$-direction) are required since the boundary condition is based solely on wall normal derivatives.

In forming the shape functions for the gridless field nodes, three points are fit in each flux direction using a polynomial basis given by $[1, x, y]$. For wall points, a six-term basis, given by $[1, x, y, xy, x^2, y^2]$ is used for the wall normal ($\tilde{y}$-direction). Nominally, six nodes are used to fit the six-term basis. However, in some instances, seven nodes are required to preclude a near-zero pivot in the least squares matrix.

4.4. Trailing edge surface discontinuity

Special treatment is required for a surface discontinuity such as that existing at the trailing edge of an airfoil. This is due to the fact that both the surface normal and radius of curvature are ill defined. Here, a formulation of the gridless boundary condition treatment is considered in which surface properties are evolved by time marching the solution in a similar manner to that of the field nodes with Eq. (11), using gridless flux contributions from both the normal and tangential directions. A two-step approach is considered by making preliminary predictions of the surface properties (conserved variable vector) using the geometry of the "lower" side of the trailing edge, followed by a similar prediction using the geometry of the "upper" side. The two preliminary sets of surface values are then averaged to provide the final value of the conserved variable vector at the trailing edge. Consider Fig. 3, which depicts the gridless flux contributions with respect to the lower surface geometry; $\tilde{\mathbf{f}}_{\mathrm{L}}^+$ and $\tilde{\mathbf{f}}_{\mathrm{L}}^-$ for the tangential directions, and $\tilde{\mathbf{g}}_{\mathrm{L}}^-$ for the normal direction. Similarly, one could consider analogous contributions resulting from the upper surface geometry, $\tilde{\mathbf{f}}_{\mathrm{U}}^+$, $\tilde{\mathbf{f}}_{\mathrm{U}}^-$, and $\tilde{\mathbf{g}}_{\mathrm{U}}^-$. Since the negative $\tilde{y}$-direction is oriented into the body surface, there is no flux contribution from that direction. Thus, in semi-discrete form, the time variation of flow properties at the surface discontinuity is written (using either the "upper" or "lower" fluxes independently)

$$
\frac{\mathrm{d}\tilde{\mathbf{u}}}{\mathrm{d}t} = -\sum_{n=1}^{N_{\tilde{x}}^-}\frac{\partial\phi_n^+}{\partial\tilde{x}}\tilde{\mathbf{f}}_n^+ - \sum_{n=1}^{N_{\tilde{x}}^+}\frac{\partial\phi_n^-}{\partial\tilde{x}}\tilde{\mathbf{f}}_n^- - \sum_{n=1}^{N_{\tilde{y}}^+}\frac{\partial\phi_n^-}{\partial\tilde{y}}\tilde{\mathbf{g}}_n^-. \tag{28}
$$

In the local wall coordinate system, flow tangency requires that the velocity normal to wall, $\tilde{v}$, must vanish. As a result, the above equation is only integrated for three components of the local conserved variable vector, while the normal component of velocity, $\tilde{v}$, is set to zero. Subsequent to each time step, the conserved variable vector is rotated back into the global coordinate orientation so that their flux contributions to other nodes may properly be evaluated. After considering both the upper and lower surfaces of the trailing edge using Eq. (28), the final value of the conserved variable at the trailing edge is taken to be

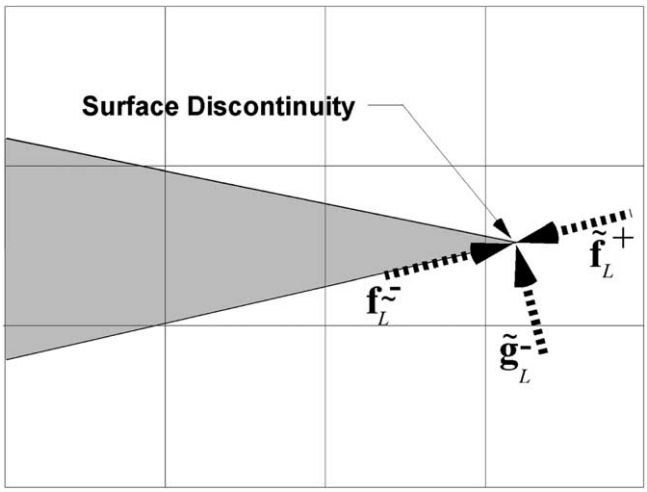

Fig. 3. Gridless flux contributions for airfoil "lower" trailing edge.

$$\mathbf{u}_{\mathrm{TE}} = \tfrac{1}{2}(\mathbf{u}_{\mathrm{L}} + \mathbf{u}_{\mathrm{U}}). \tag{29}$$

It is pointed out that a surface boundary treatment using Eq. (28) along the entire body has been considered. However, numerical results indicate that the incorporation of the surface radius of curvature, as in Eq. (27), yields smoother, and more accurate results.

4.5. Adaptive cloud nodes

To assess the potential for increased accuracy and convergence, the concept of "adaptive cloud nodes" is introduced. These are nodes which are adapted to the local grid prior to the flow field simulation. That is, they are used to fill sparse regions of the Cartesian mesh formed as a result of the intersection with the body geometry. Adaptive cloud nodes, when incorporated, are positioned at 0.5 times the local grid spacing, Δx, normal to the wall at each surface node. Cloud nodes that fall within $0.3\Delta x$ of a Cartesian field node are omitted from the solution. The adaptive cloud nodes incorporate a local $(\tilde{x}, \tilde{y})$ coordinate system based on the surface node from which they are generated. A six term polynomial basis is used in the $+\tilde{y}$ direction, while a three term basis for the other three directions.

As discussed below, the use of the adaptive cloud nodes revealed only small enhancements in convergence and accuracy. This is attributed to their lower order of discretization in the direction toward and tangential to the wall. It is noted, however, that these nodes are sometimes very useful in facilitating the generation of the gridless shape functions in cases where a collection of only Cartesian mesh nodes result in singular least squares matrices.

5. Multistage time stepping

Time integration (evolution) of the flow field is achieved using Runge–Kutta time stepping, as first presented by Jameson et al. [27]. This explicit multi-stage scheme is ideally suited for multigrid acceleration and provides an increased (though limited) allowable time step compared to computational work. Considering separate temporal and spatial discretization (i.e. method of lines), the semi-discrete form of the governing equations is written

$$\frac{d\mathbf{u}}{dt} = -\mathrm{Res}_j, \tag{30}$$

where the residual at node j, Res_j, is a discretized representation of the spatial derivatives (e.g. Eqs. (11) and (28)). The Runge–Kutta integration is a sequence of updates which takes the solution from time level n to time level $n+1$. For k stages, the scheme reads

$$\begin{aligned}
u_j^{(0)} &= u_j^n, \\
u_j^{(1)} &= u_j^{(0)} - \alpha_1 \Delta t_j \mathrm{Res}_j^{(0)}, \\
u_j^{(2)} &= u_j^{(0)} - \alpha_2 \Delta t_j \mathrm{Res}_j^{(1)}, \\
&\vdots \\
u_j^{(k)} &= u_j^{(0)} - \alpha_k \Delta t_j \mathrm{Res}_j^{(k-1)}, \\
u_j^{n+1} &= u_j^{(k)},
\end{aligned} \tag{31}$$

where α_k are the stage coefficients, and Δt is the local allowable time step at node j. The stage coefficients can be optimized for various spatial discretizations. For instance, Van Leer et al. [28], provides stage coefficients for 3, 4, and 5 stage schemes optimized for first order and second order upwind schemes. For a 4 stage first order upwind scheme, the coefficients read

$$\alpha_k = \{0.0833, 0.2069, 0.4265, 1.0000\}. \tag{32}$$

It is noted that in solutions containing shocks, stage coefficients optimized for first order schemes are preferred regardless of the spatial discretization [29]. The optimal CFL number for the stage coefficients presented above is reported to be 2.0 [28]. For time marching of the field nodes with constant grid spacing Δx in both coordinate directions, the CFL number, σ, is defined by

$$\sigma = \frac{\Delta t}{\Delta x}[(|u| + c) + (|v| + c)]. \tag{33}$$

6. Multigrid for embedded meshes

A full approximation storage (FAS) multigrid scheme based on the work of Jameson [30] has been incorporated for communication and convergence acceleration of the flow solution on an embedded mesh hierarchy. In the FAS method, the simulation is accelerated to convergence by transferring (restricting) both the solution and residual vector to successively coarser meshes. After performing iterations on the coarser grids, corrections to the fine grid solution are transferred (prolonged) back to the fine grid. The multigrid method accelerates the solutions to convergence primarily by means of two effects. First, larger time steps can be applied on a coarser grid with reduced numerical effort. More importantly, iterative schemes are most efficient at reducing the high frequency components of the solution error, whereas low frequency error is barely affected. Thus, upon transferring the solution from a fine grid to a coarse grid, the low frequency components of error become high frequency components and are therefore more effectively damped. Additionally, first order spatial discretization may be used on the coarse grids since their contribution does not affect the accuracy of the fine grid, only the convergence rate.

The implementation of multigrid acceleration presented here is different than that typically considered on traditional computational grids. Namely, a traditional computational grid typically spans the entire domain whereupon successively coarser grids are formed by coalesce of neighboring cells. This process is

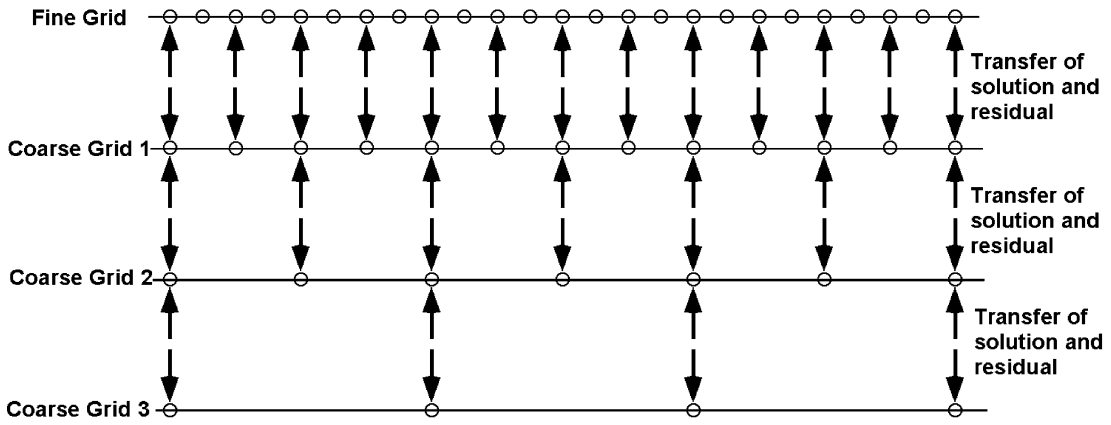

Fig. 4. Traditional mesh coarsening for three layer multigrid.

exemplified in Fig. 4, which shows a coarsening process for a one-dimensional domain. The process then proceeds as follows. The solution at time step $n+1$ on the fine grid, $\mathbf{u}_h^{n+1}$, is transferred to the first coarse grid using an interpolation (restriction) operator I_h^{2h} providing an initial solution of the coarse grid given by

$$\mathbf{u}_{2h}^{(0)} = I_h^{2h}\mathbf{u}_h^{n+1}. \tag{34}$$

In order to maintain the accuracy of the fine grid solution, the residual vector, $\mathbf{R}$, is also transferred from the fine grid to the coarse grid using a forcing function defined by

$$(\mathbf{Q}_{\mathrm{F}})_{2h} = I_h^{2h}\mathbf{R}_h^{n+1} - \mathbf{R}_{2h}^{(0)}, \tag{35}$$

which is the difference of the residual transferred from the fine grid and that calculated on the coarse grid after one stage of the Runge–Kutta integration using the transferred solution, $\mathbf{u}_{2h}^{(0)}$. Iteration(s) are then carried out on the coarse grid using the Runge–Kutta multi-stage process defined in Eq. (31) in the same manner as the fine grid, but with the forcing function added to the residual at each stage. Thus,

$$\begin{aligned} &\mathbf{u}_{2h}^{(k)} = \mathbf{u}_{2h}^{(0)} - \alpha_k \Delta t_j \left[\mathbf{R}_{2h}^{(k-1)} + (\mathbf{Q}_{\mathrm{F}})_{2h}\right] \quad k = 1, \ldots, m, \\ &\mathbf{u}_{2h}^{n+1} = \mathbf{u}_{2h}^{(m)}. \end{aligned} \tag{36}$$

In this manner, the residual after the first stage on the coarse grid is identical to that transferred from the fine grid thus retaining the fine grid accuracy. The process can be carried down to several coarse grids, each transferring the forcing function from the finer grid above. For instance a second coarsening would transfer the forcing function given by

$$(\mathbf{Q}_{\mathrm{F}})_{4h} = I_{2h}^{4h}[\mathbf{R}_h^{n+1} + (\mathbf{Q}_{\mathrm{F}})_{2h}] - \mathbf{R}_{4h}^{(0)}. \tag{37}$$

Once the coarsest grid has been reached, a coarse grid correction can be defined as the difference between the transferred solution and that obtained after performing iteration(s) on the coarse grid. Thus, for a three layer multigrid, the corrected solution on the fine grid, $\mathbf{u}_h^+$, would follow from

$$\begin{aligned}
\delta\mathbf{u}_{4h} &= \mathbf{u}_{4h}^{n+1} - \mathbf{u}_{4h}^{(0)}, \\
\mathbf{u}_{2h}^{+} &= \mathbf{u}_{2h}^{n+1} + I_{4h}^{2h}\delta\mathbf{u}_{4h}, \\
\delta\mathbf{u}_{2h} &= \mathbf{u}_{2h}^{+} - \mathbf{u}_{2h}^{(0)}, \\
\mathbf{u}_{h}^{+} &= \mathbf{u}_{h}^{n+1} + I_{2h}^{h}\delta\mathbf{u}_{2h},
\end{aligned} \tag{38}$$

where I_{4h}^{2h} and I_{2h}^{h} are interpolation (prolongation) operators. Variations in the process can be made by changing the number of iterations completed before restriction and/or after prolongation.

In contrast to the above, the approach taken here is for an embedded mesh hierarchy, which differs somewhat from the traditional coarsening approach. Here, grids of various resolution are generated prior to flow field evolution, and finer grids are "patched" above coarser ones. Instead of coarsening a fine grid, the region of coarse grid that extends beneath a finer one is used for multigrid communication. This process is depicted in Fig. 5. As shown, only the grid nodes that are depicted as solid in the figure carry a forcing function for multigrid acceleration of embedded meshes located above. Nodes on the perimeter of an embedded mesh do not transfer a forcing function, but are instead prescribed with a Dirichlet boundary condition obtained from the coarser mesh below. These boundary conditions are applied during prolongation of the coarse mesh along with the transfer of the coarse grid corrections. Embedded meshes with a locally rectangular domain have the significant advantage of low storage requirement per cell since any grid point in the domain can be located by the cell index (i, j), and the grid spacing Δx of the particular mesh. Actual coordinates of grid points and face dimensions of each cell need not be stored.

In this study, five strategies for flow field evolution are considered and their convergence properties are demonstrated. These five strategies are listed in Table 1. All multigrid implementations incorporate a V-cycle approach, but the number of iterations prior to prolongation is varied. Also considered is Laplacian smoothing of the coarse grid corrections. This is achieved by solving the tridiagonal system given by

$$\begin{aligned}
&-\varepsilon\delta\hat{\mathbf{u}}_{i-1,j} + (1+2\varepsilon)\delta\hat{\mathbf{u}}_{i,j} - \varepsilon\delta\hat{\mathbf{u}}_{i+1,j} = \delta\mathbf{u}_{i,j}, \\
&-\varepsilon\delta\bar{\mathbf{u}}_{i,j-1} + (1+2\varepsilon)\bar{\mathbf{u}}_{i,j} - \varepsilon\delta\bar{\mathbf{u}}_{i,j+1} = \delta\hat{\mathbf{u}}_{i,j},
\end{aligned} \tag{39}$$

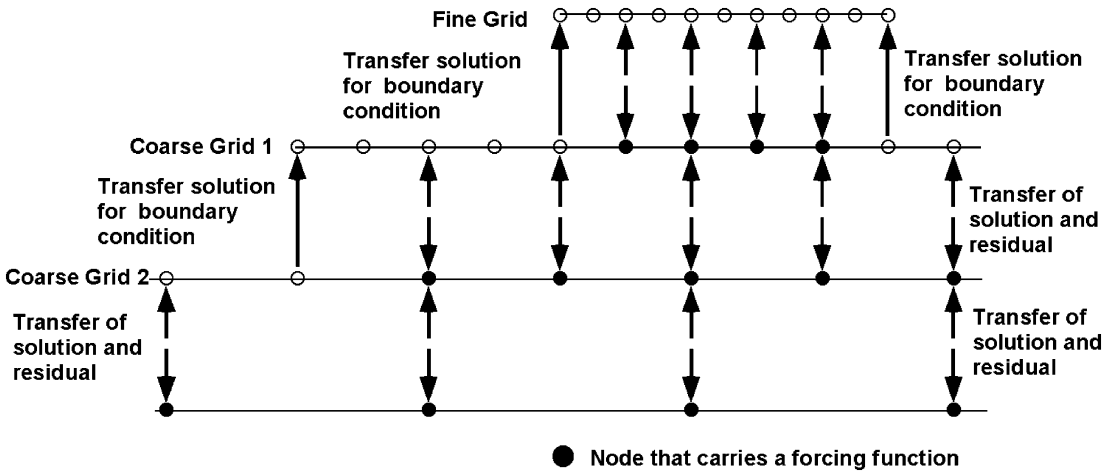

Fig. 5. Multigrid strategy for embedded mesh.

Table 1
Flow field evolution strategies used for convergence assessment

1	No multigrid acceleration
2	One Runge–Kutta iteration prior to restriction only
3	One Runge–Kutta iteration prior to restriction with Laplacian smoothed corrections prior to prolongation
4	One Runge–Kutta iteration prior to restriction and one Runge–Kutta iteration prior to prolongation
5	One Runge–Kutta iteration prior to restriction and one Runge–Kutta iteration prior to prolongation with Laplacian smoothed corrections

where the caret indicates smoothed corrections in i direction, the over bar indicates corrections smoothed in both i and j directions, and ε is a smoothing coefficient, taken here to be 2.0, unless otherwise mentioned. Convergence properties of the five various evolution methods are presented below.

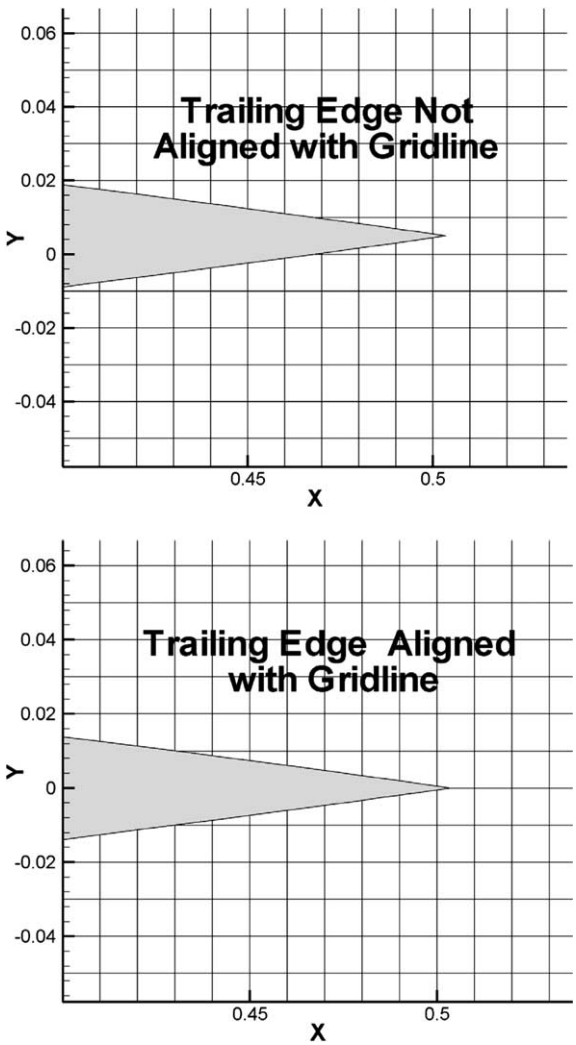

Fig. 6. Comparison of NACA 0012 trailing edge region for aligned and unaligned mesh.

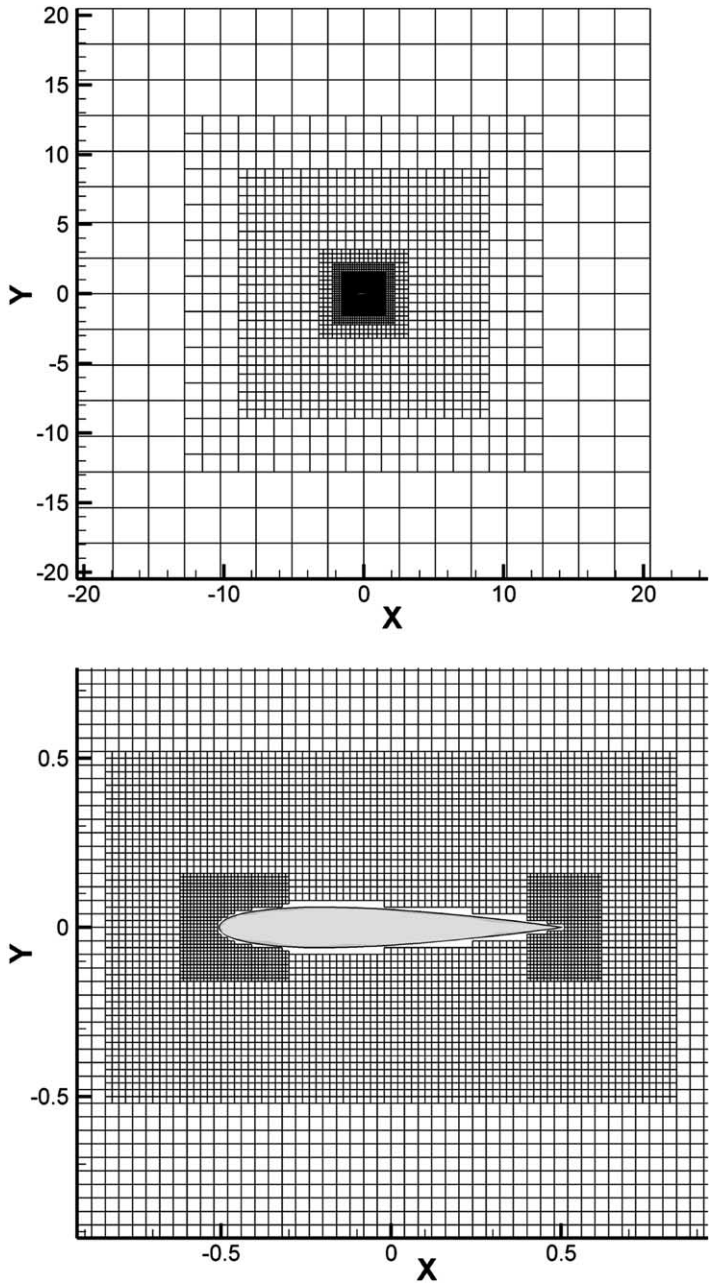

Fig. 7. Cartesian mesh for single NACA 0012 airfoil configuration ($M = 0.5$, $\alpha = 3.0$).

It is noted that, in performing the multigrid acceleration, only field nodes incorporate coarse grid corrections. For gridless field nodes (located near the body), shape functions are calculated at all grid levels and used for multigrid. Typically, as the mesh hierarchy is traversed from fine to coarse, the location of the gridless field node changes depending on the coarseness of the grid. This is due to the fact that, as the grid

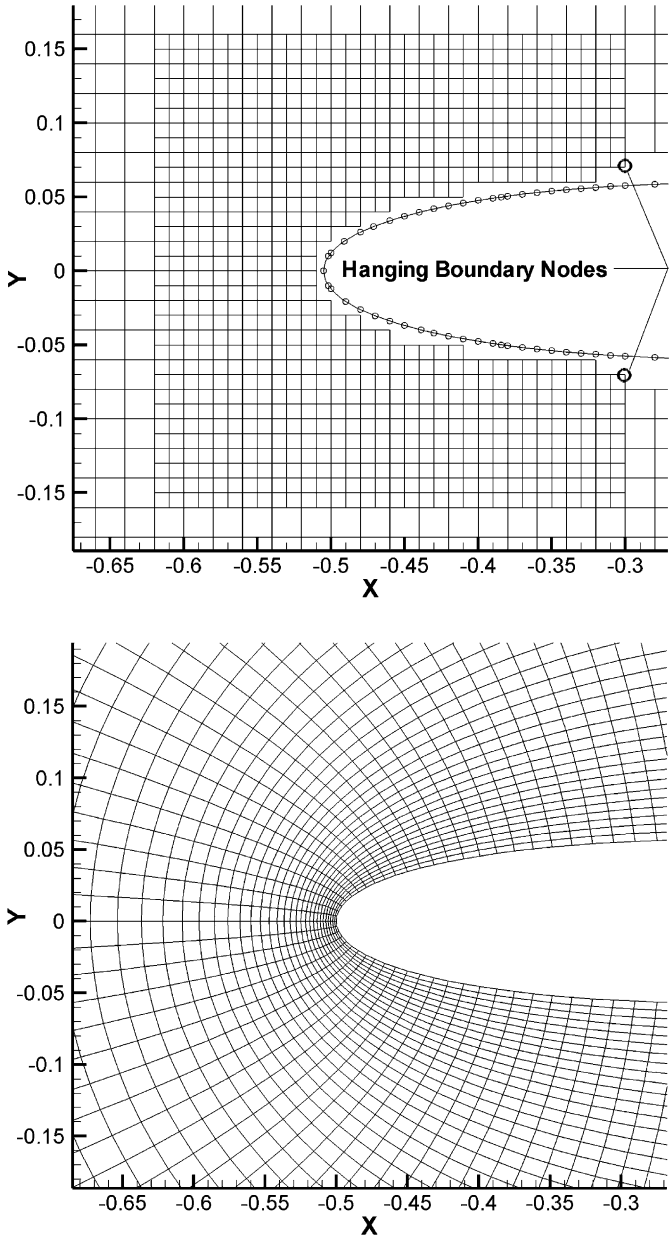

Fig. 8. Comparison of FLO52 NACA 0012 nose region mesh with Cartesian mesh ($M = 0.5$, $\alpha = 3.0$).

spacing increases, the minimum allowable distance between the body and gridless field nodes, $0.3\Delta x$, increases. Surface nodes and adaptive cloud nodes do not receive coarse grid corrections through multigrid.

Finally, for flow field evolution without multigrid acceleration, the solution from a fine mesh is transferred to a solid node (see Fig. 5), but no Runge–Kutta iterations are performed on those nodes, since there values are strictly imposed by the solution on the finer grid above. Here, the Dirchlet boundary conditions are imposed after the Runge–Kutta iteration on the fine grid.

7. Results and discussion

One advantage of using the gridless boundary treatment for a Cartesian mesh is that significant features of the surface geometry (i.e. the trailing edge of an airfoil) need not necessarily be aligned with the mesh. However, alignment with the mesh would be beneficial should it be feasible, as is the case for many

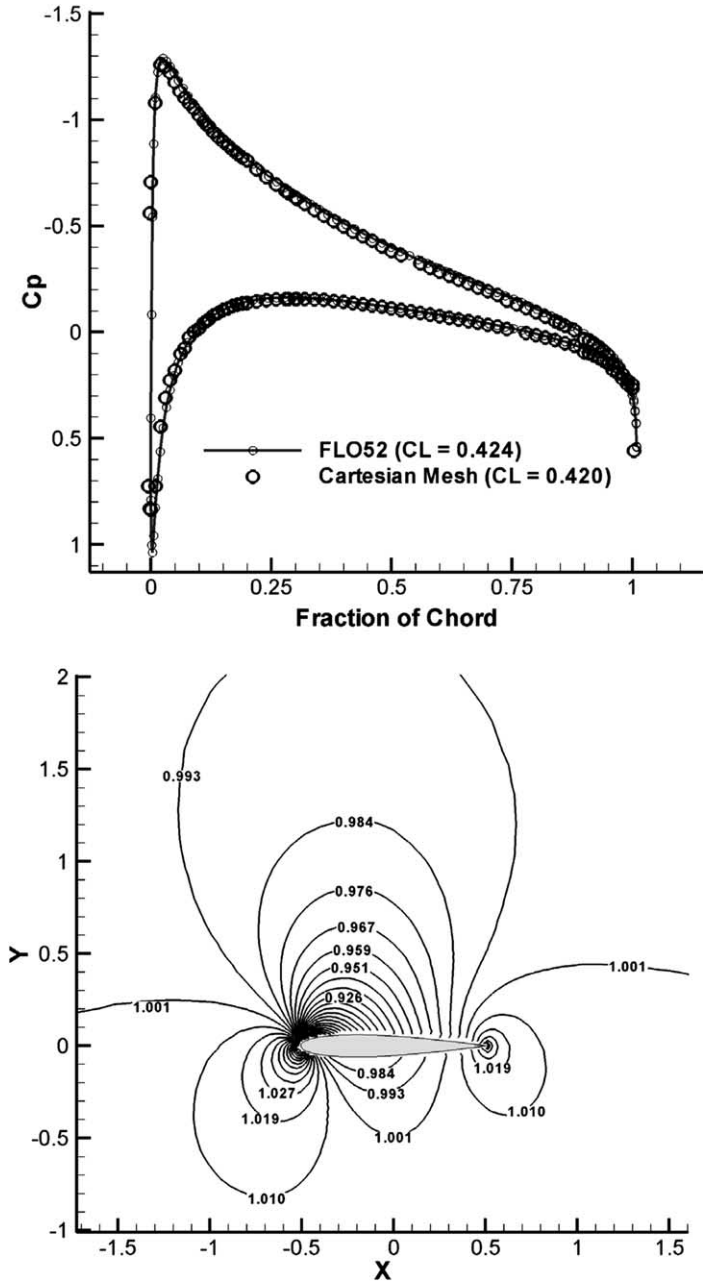

Fig. 9. Comparison of NACA 0012 pressure coefficient for ($M = 0.5, \alpha = 3°$) and flow field pressure distribution (P/P_{inf}) for Cartesian mesh.

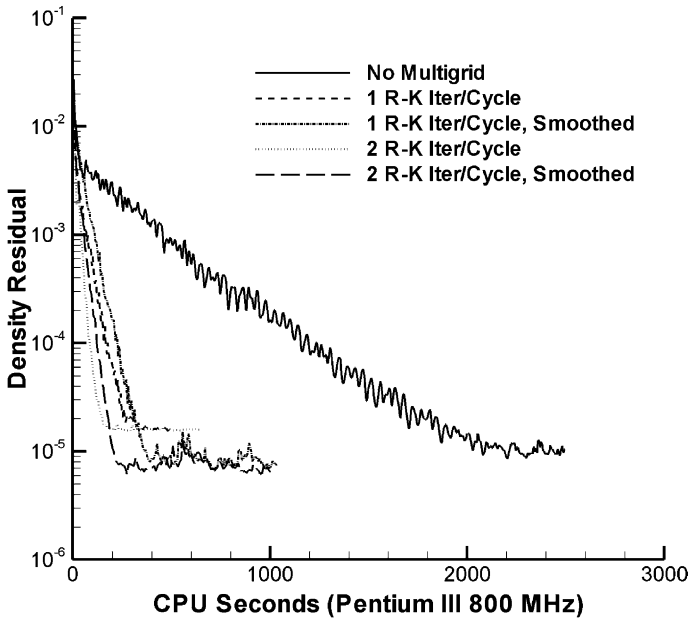

Fig. 10. Convergence history for NACA 0012 ($M = 0.5$, $\alpha = 3°$) for Various multigrid acceleration techniques.

2-dimensional considerations. Fig. 6 is provided to clarify the distinction between aligned and unaligned grids in which the trailing edge of the NACA 0012 is shown. For the unaligned case, the trailing edge lies within the middle of a grid cell and is not directly seen by the mesh. However, results below indicate that the boundary treatment presented here has no trouble with such an occurrence. Convergence examples for both aligned and unaligned geometry are presented below. For the aligned cases, surface nodes can be well established by only considering the intersection points between the body and Cartesian grid. However, for an unaligned airfoil case, there is no intersection of the trailing edge with the mesh so the trailing edge point would be omitted from the solution. To rectify such an occurrence, incorporating some of the surface definition nodes into the solution is required. At a minimum, the surface definition node associated with the trailing edge stagnation point must be incorporated. This is easily accomplished in the present gridless boundary-condition method.

In addition to aligned and unaligned mesh cases, results for an unaligned mesh with the addition of adaptive cloud nodes are also presented. The adaptive cloud nodes are inserted in the large empty spaces formed between the body and the Cartesian mesh. Results below indicate that these cloud nodes have a small effect on convergence, but can facilitate fitting of gridless shape functions. Both single and dual NACA 0012 airfoil configurations are considered, and the effectiveness of multigrid acceleration and adaptive cloud nodes is investigated.

It is noted that the far field (vortex) correction for lifting bodies of [31] was used for the single airfoil cases. Also all cases were run on a Pentium III 800 MHz with single precision arithmetic. For all cases, the finest grid level has a spacing of 0.01 chord lengths.

7.1. Case 1: Subsonic flow over NACA 0012

Case 1 considers subsonic flow at Mach 0.5, and 3° angle of attack around the NACA 0012 airfoil. The computational mesh for this case is presented in Fig. 7, which is aligned with the airfoil trailing edge and

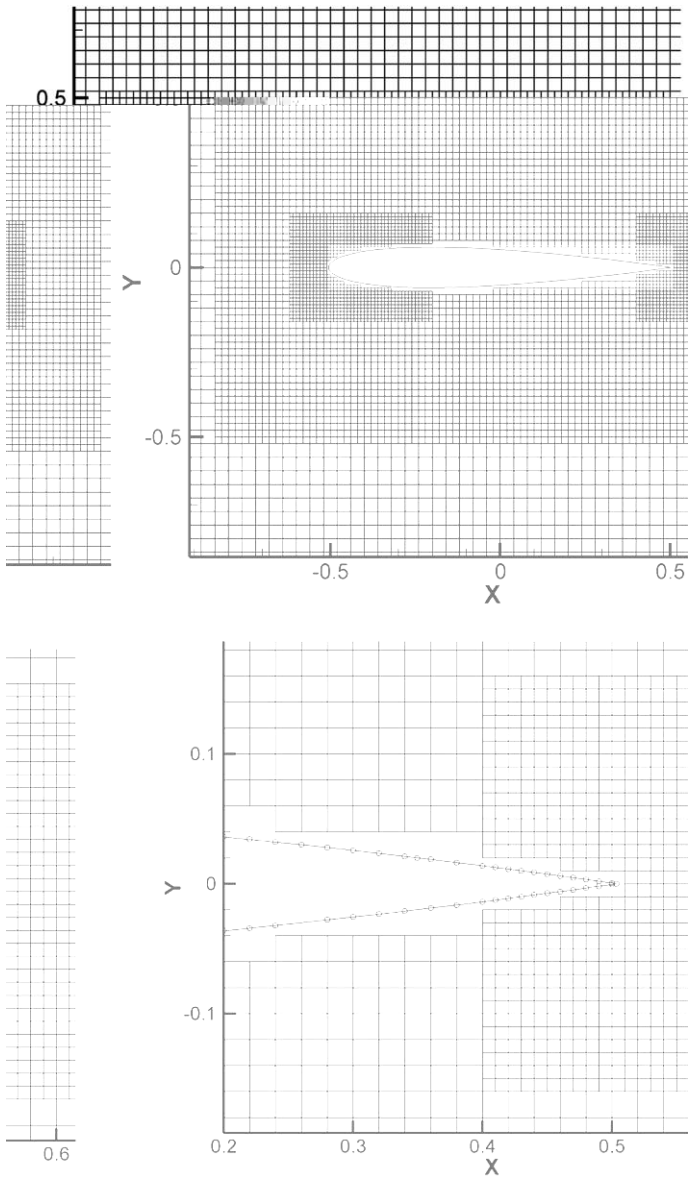

Fig. 11. Cartesian mesh for single NACA 0012 airfoil configuration ($M = 0.8$, $\alpha = 1.25$).

has 10,418 top-level grid points in the field, and 137 body nodes. Pre-processing of this grid required 0.611 CPU seconds to evaluate all grid intersections and evaluate the gridless shape functions.

To assess the accuracy of the method, a comparison is made with results of a FLO52 simulation. Fig. 8 presents both the Cartesian mesh and the FLO52 body-fitted mesh. Note that the intersection of the body with the Cartesian mesh has resulted in hanging boundary nodes. Essentially, these are nodes that cannot be set using a Dirichlet condition prescribed from the course mesh below. Instead, they must be treated as gridless field nodes and solved using the least squares fitting of the gridless method.

Although the body-fitted mesh allows for better clustering of grid cells near the nose, comparisons of the pressure coefficient distribution on the surface of the airfoil are in excellent agreement, as shown in the top portion of Fig. 9. Predicted lift coefficient is within 1% of the FLO52 value. Note that peak pressure at the leading edge was missed to the lack of grid clustering near the nose. The flow field pressure distribution is

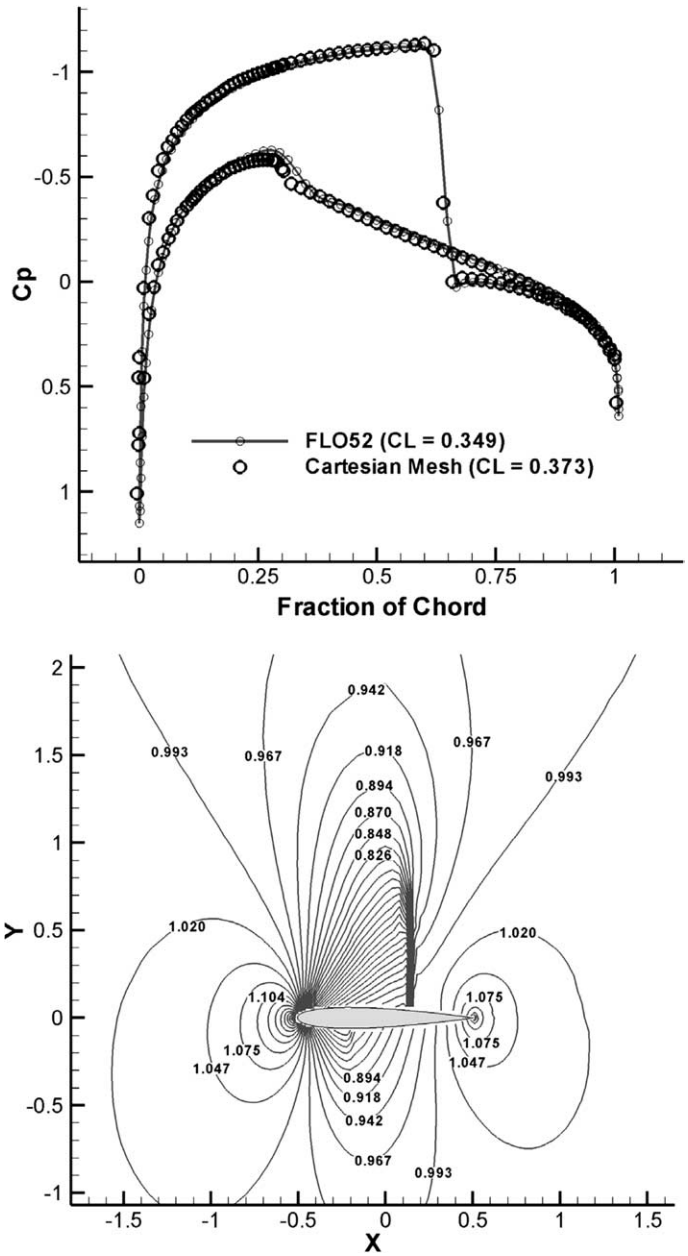

Fig. 12. Comparison of NACA 0012 pressure coefficient for ($M = 0.8$, $\alpha = 1.25°$) and flow field pressure distribution (P/P_{inf}) for Cartesian mesh.

presented in the bottom portion of Fig. 9. The convergence history for this case is presented in Fig. 10 in terms of density residual. It is seen here that all multigrid strategies considered are very effective in increasing convergence of the solution. Although the smoothing of coarse grid corrections provides a lower final residual, the convergence rate is slightly longer. For this case, using two Runge–Kutta (R–K) iterations per cycle with no smoothing has the best convergence, offering roughly a 12-fold increase in speed.

7.2. Case 2: Transonic flow over NACA 0012

The computational mesh used for the transonic NACA 0012 airfoil at Mach 0.8, $\alpha = 1.25°$, is presented in Fig. 11. This mesh is nearly identical to that of Case 1 with the exception of a larger grid patch near the nose in anticipation of the weak shock formed there under the considered free stream conditions. As in Case 1, the domain extends 20 chord lengths from the body. The surface pressure coefficient and flow field pressure distribution are provide in Fig. 12 which show good agreement with the FLO52 solution, particularly with respect to the location of the shocks, captured with a single grid point. The predicted lift coefficient is found to be within roughly 6% of the value predicted by FLO52. The convergence rate here is not as good as Case 1 due to the presence of shocks in the flow field. For this case, performing smoothing of the coarse grid corrections significantly helped the one R–K iteration per cycle multigrid method. Without the smoothing, persistent oscillations in the lift coefficient were prominent, and the density residual remained elevated, as shown in Fig. 13. By far the best method here is with two R–K iterations per cycle and no smoothing providing roughly a 3-fold increase in convergence speed.

Next, a comparison is made between solutions on an aligned mesh with that on an unaligned mesh. Also considered is the effect of adaptive cloud nodes used in conjunction with an unaligned mesh. With the exception of misalignment of the airfoil, the grid is unchanged. Total pre-processing time required for this case is 0.961 CPU seconds when incorporating the adaptive cloud nodes, and 0.671 CPU seconds when considering the unaligned mesh without the adaptive cloud nodes. A close up view of the nose and tail for

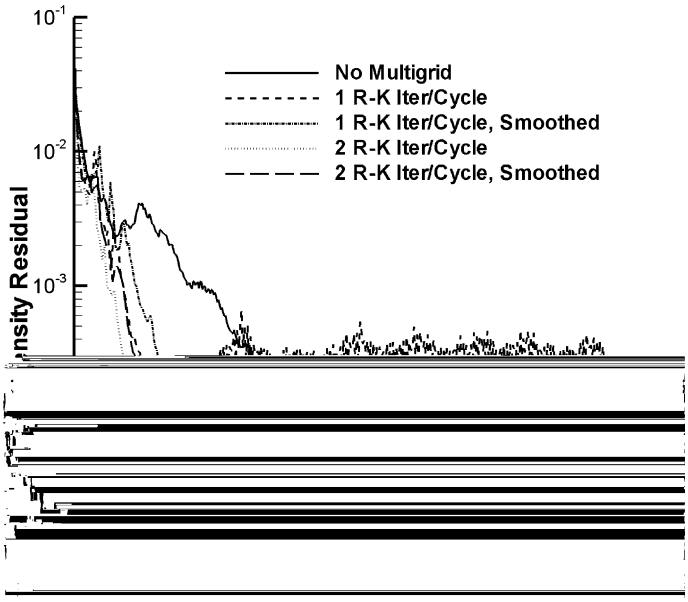

Fig. 13. Convergence history for NACA 0012 ($M = 0.8$, $\alpha = 1.25°$) for various multigrid acceleration techniques.

the unaligned grid with a total of 77 adaptive cloud nodes is provided in Fig. 14. In addition to using the grid intersection nodes for the boundary solution, a point for the trailing edge has been incorporated. Additionally, three points at the nose have also been added to help capture the leading edge stagnation pressure.

In Fig. 15, a comparison is made between the surface pressure coefficient obtained using the aligned mesh (taken from Fig. 12), on the unaligned mesh, on the unaligned mesh with the addition of adaptive

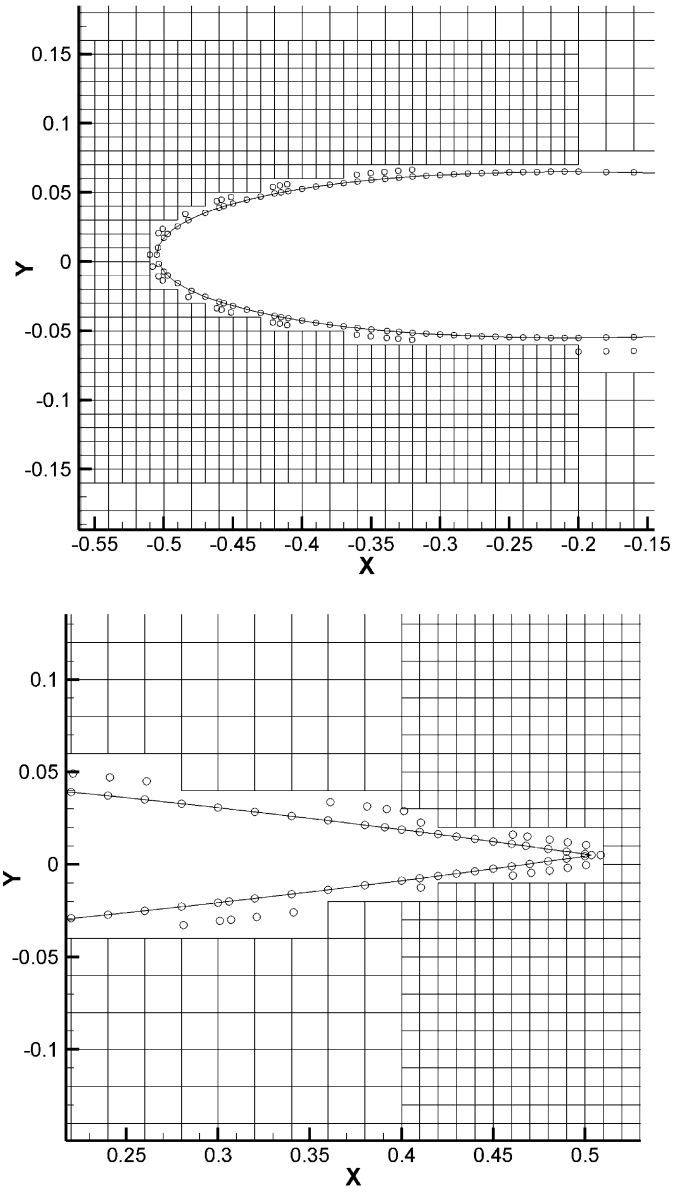

Fig. 14. Nose and tail region for unaligned mesh for single NACA 0012 airfoil configuration ($M = 0.8$, $\alpha = 1.25$) showing cloud and surface nodes.

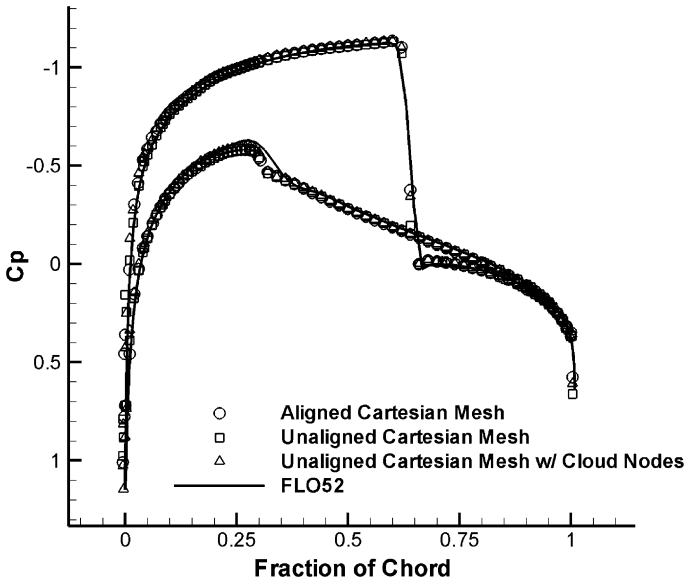

Fig. 15. Comparison of NACA 0012 pressure coefficient for ($M = 0.8$, $\alpha = 1.25°$) for aligned and unaligned Cartesian mesh.

cloud nodes, and that obtained using FLO52. As shown, all cases are in close agreement indicating that the solution is insensitive to the mesh misalignment. Furthermore the introduction of the adaptive cloud nodes in conjunction with the additional surface nodes near the nose has resulted in a more accurate prediction of the leading edge stagnation pressure.

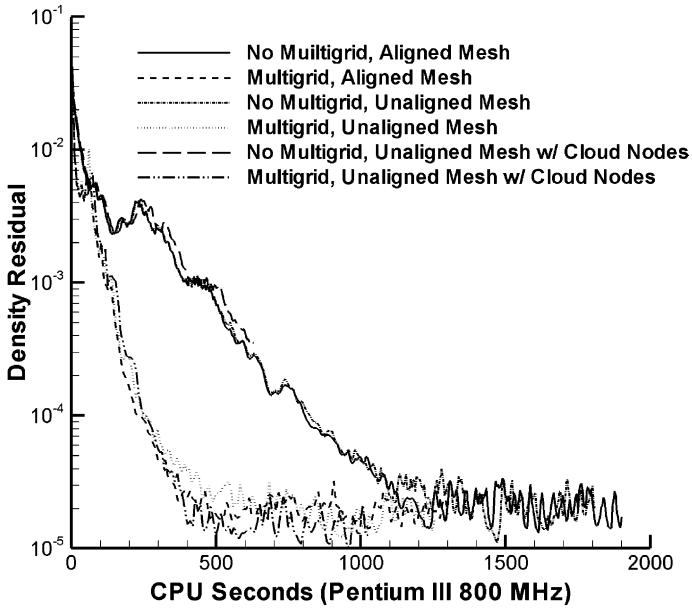

Fig. 16. Convergence history for NACA 0012 ($M = 0.8$, $\alpha = 1.25°$) for unaligned mesh.

The convergence history for this case is provided in Fig. 16, which considers either no multigrid, or 2 R–K iterations per cycle. As shown, convergence history is very similar for all mesh configurations, with multigrid offering a speed increase of roughly a factor of 3, the case with adaptive cloud nodes being the fastest. Thus introduction of the adaptive cloud nodes has been shown to increase accuracy and, to a small degree, the convergence rate.

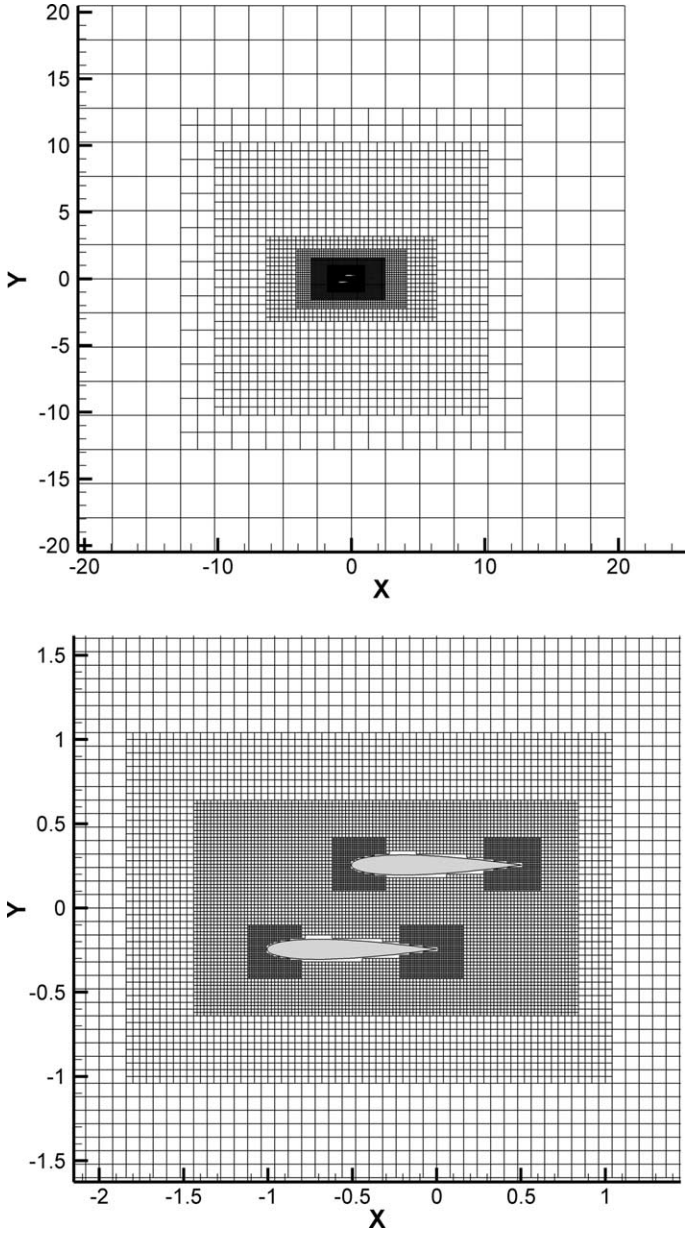

Fig. 17. Unaligned Cartesian mesh for staggered NACA 0012 airfoil configuration ($M = 0.7$, $\alpha = 0°$).

7.3. Case 3: Transonic flow through two staggerd NACA 0012 airfoils

This final case presented considers a staggered NACA 0012 configuration at Mach 0.8. The computational grid for this case is presented in Fig. 17. As shown, this case incorporates four embedded meshes within a single host grid. Again, considered here will be the case of the aligned and unaligned mesh with

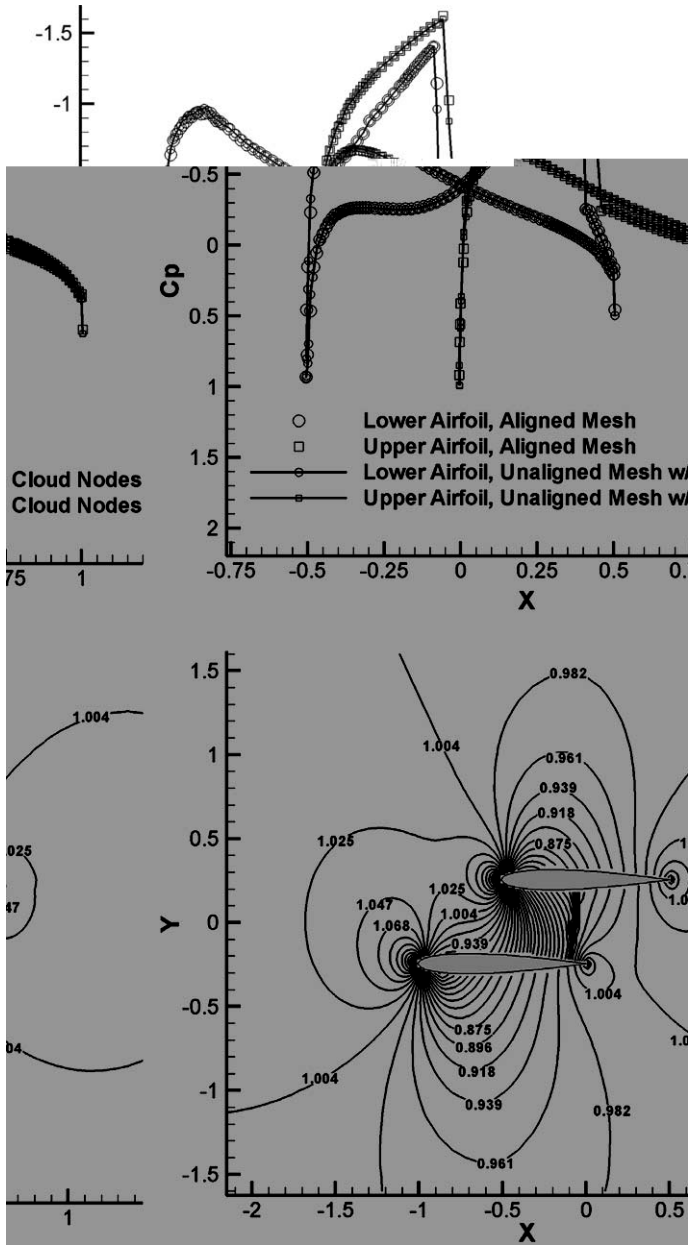

Fig. 18. Comparison aligned and unaligned surface pressure coefficient for ($M = 0.7$, $\alpha = 0°$) and flow field pressure distribution (P/P_{inf}) for staggered NACA 0012 configuration.

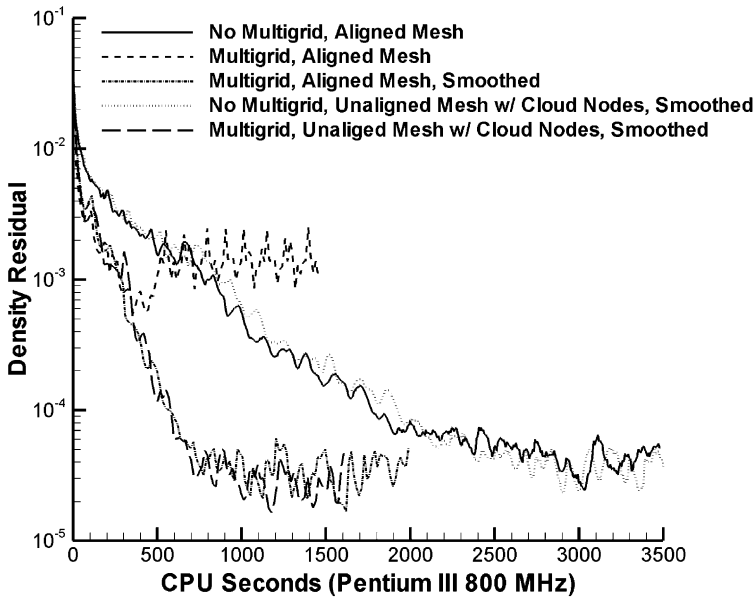

Fig. 19. Convergence history of staggered NACA 0012 configurations ($M = 0.7$, $\alpha = 0°$).

adaptive cloud nodes, establishing insensitivity to mesh alignment. There are 17,233 top-level cells in this mesh with 149 surface nodes on each airfoil for the aligned case. For the unaligned case, there are 157 surface nodes on each with an additional 92 cloud nodes per airfoil. The pre-processing time for this grid requires 1.793 CPU seconds for the aligned mesh, and 3.014 CPU seconds for the unaligned mesh with adaptive cloud nodes.

The surface pressure coefficient for this case is presented at the top of Fig. 18. As shown, the solutions are nearly identical for both aligned and unaligned meshes. The bottom of the figure provides the flow field pressure distribution for the unaligned case. The convergence history is presented in Fig. 19. As shown, two R–K iterations per cycle multigrid requires smoothing of the coarse grid corrections in order to establish convergence. Both the aligned and unaligned grids (with cloud nodes) exhibited similar convergence properties, with roughly a 4-fold increase in speed using multigrid.

To summarize the qualities of the various multigrid strategies, performing a Runge–Kutta iteration after prolongation enhances convergence, especially for cases in which shocks are present. The Laplacian smoothing of the coarse grid corrections in the manner considered here did not typically help convergence, except in cases in which overall convergence was problematic (such as the transonic airfoil case with one R–K iteration per cycle, and the dual airfoil case). It is hypothesized that the gridless boundary treatment, for some mesh configurations, introduces high frequency components that are not sufficiently damped during the multigrid cycle. Smoothing the coarse grid corrections, however, provides enough damping such that the overall convergence of the solution can be effectively accelerated with multigrid.

8. Conclusions

A gridless boundary treatment has been introduced for application of the Euler equations on Cartesian meshes for arbitrary body geometries. The method has been shown to be insensitive to mesh positioning relative to the body, and provides for accurate results without the need for excessive mesh refinement. As a

natural extension of the gridless formulation, the method allows for the inclusion of adaptive cloud nodes, which can be placed in sparse regions of the mesh, adding resolution, and facilitating implementation of the method. Unlike many Cartesian methods, there are no issues associated with thin body geometry where small cut cells may be formed by intersection of the mesh.

For simple geometries, such as a single airfoil, a body-fitted grid method may be easily applied. Results presented here show that the proposed Cartesian method is well suited for both single- and multiple-body geometries, and an excellent candidate for extension into three-dimensional simulations where grid generation is more complicated. For the two-body airfoil cases, mesh pre-processing is shown to take less than a few seconds. The method does not require extensive grid generation effort and provides very reasonable solutions, making an excellent and efficient tool for analyses in a design environment.

A convergence study has been presented which indicated that multigrid can accelerate convergence of the solution by a factor of 3 or 4 in cases with shock waves present, and greater than a factor of 10 for cases without shocks. As a result, the method has advantages over purely gridless algorithms in which multigrid is not easily incorporated, and issues with global conservation are present.

It must be pointed out that the proposed embedded mesh refinement method in this paper can be easily modified to work in an adaptive mesh refinement framework. Furthermore, the approach of the proposed gridless boundary condition treatment method is in principle equally applicable to the solution of the Navier–Stokes equations. The effectiveness of this method for the Navier–Stokes equations for high Reynolds number flows, however, is subject to further research.

Acknowledgements

The authors would like to thank Dr. Her-Mann Tsai from Temasek Laboratory, National University of Singapore, for many fruitful discussions on the gridless method.

References

[1] P.R. Lahur, Y. Nakamura, A New Method for Thin Body Problem in Cartesian Grid Generation, AIAA Paper 99-0919, 1999.
[2] D.K. Clarke, M.D. Salas, H.A. Hassan, Euler calculations for multielement airfoils using Cartesian grids, AIAA Journal 24 (March) (1986) 353–358.
[3] R.L. Gaffney, H.A. Hassan, M.D. Salas, Euler Calculations for Wings Using Cartesian Grids, AIAA Paper 87-03563, January 1987.
[4] D.M. Tidd, D.J. Strash, B. Epstein, A. Luntz, A. Nachshon, T. Rubin, Multigrid Euler calculations over complete aircraft, Journal of Aircraft 29 (November–December) (1992) 1080–1085.
[5] R.J. Leveque, High Resolution Finite Volume Method on Arbitrary Grids Via Wave Propagation, AIAA 89-1930, 1989.
[6] M.J. Berger, R. Leveque, Stable boundary conditions for Cartesian grid calculations, Computing Systems in Engineering 1 (1990) 305–311.
[7] R.J. Leveque, M.J. Berger, A rotated difference scheme for Cartesian grids in complex geometries, AIAA Paper CP-91-1602, 1991.
[8] H. Forrer, Boundary treatment for a Cartesian grid method, ETH Report 96-04, April 1996.
[9] D. De Zeeuw, K.G. Powell, An adaptively refined Cartesian mesh solver for he Euler equations, Journal of Computational Physics 104 (1993) 56–68.
[10] R.B. Pember, J.B. Bell, P. Colella, W.Y. Crutchfield, M.L. Welcome, An adaptive Cartesian grid method for unsteady compressible flow in irregular regions, Journal of Computational Physics 1209 (1995) 278–304.
[11] J.E. Melton, M.J. Berger, M.J. Aftosmis, M.D. Wong, 3D applications of a Cartesian grid Euler method, AIAA Paper 95-0853, 1995.
[12] W.J. Coirier, K.G. Powell, An accuracy assessment of Cartesian-mesh approaches for the Euler equations, Journal of Computational Physics 177 (1995) 1995.
[13] J.J. Quirk, An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies, Computer & Fluids 23 (1) (1994) 125–142.

[14] H. Forrer, R. Jeltsch, A higher order boundary treatment for Cartesian-grid methods, Journal of Computational Physics 140 (1998) 259–277.
[15] M.J. Aftosmis, M.J. Berger, G. Adomavicius, A parallel multilevel method for adaptively refined Cartesian grids with embedded boundaries, AIAA Paper 2000-0808, 38th Aerospace Sciences Meeting and Exhibit, January 2000.
[16] A. Dadone, B. Grossman, An immersed body methodology for inviscid flows on Cartesian grids, AIAA Paper 2002-1059, January 2002.
[17] A. Dadone, B. Grossman, Surface boundary conditions for the numerical solution of the Euler equations, AIAA Journal 32 (1995) 285–293.
[18] J.T. Batina, A gridless Euler/Navier–Stokes solution algorithm for complex two-dimensional applications, NASA-TM-107631, June 1992.
[19] S.-C. Shih, S.-Y. Lin, A weighted least squares method for Euler and Navier–Stokes equations, AIAA Paper 94-0522, January 1994.
[20] J.L. Liu, S.J. Su, A potentially gridless solution method for the compressible Euler/Navier–Stokes equations, AIAA Paper 96-0526, 1996.
[21] S.P. Vanka, N. Ploplys, Meshless methods for Navier–Stokes equations using radial basis functions, Advances in Computational Engineering and Sciences 2 (2000).
[22] R. Subrata, M. Fleming, Nonlinear subgrid embedded element-free-Galerkin method for monotone CFD solutions, in: Proc. Int. Mechanical Engineering Cong. and Expo., November 2000.
[23] H. Lin, S.N. Atluri, The meshless local Petrov–Galerkin (MLPG) method for solving incompressible Navier–Stokes equations, Computer Modeling in Engineering & Sciences 2 (2001) 117–142.
[24] D. Sridar, N. Balakrishnan, An upwind finite difference scheme for meshless solvers, Journal of Computational Physics 189 (2003) 1–29.
[25] T. Belytschko, Y.Y. Lu, L. Gu, Element-free Galerkin methods, International Journal of Numerical Methods in Engineering 37 (1994) 229–256.
[26] B. Van Leer, Flux-vector splitting for the Euler equations, in: Proc. Eighth International Conference on Numerical Methods in Fluid Dynamics, Springer Verlag, Berlin, 1982, pp. 507–512.
[27] A. Jameson, W. Schmidt, E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping, AIAA Paper 81-1259, AIAA 14th Fluid and Plasma Dynamics Conference, June, 1981.
[28] B. Van Leer, C.-H. Tai, K.G. Powell, Design of optimally smoothing multi-stage schemes for the Euler equations, AIAA Paper 89-1933, 1989.
[29] J. Blazek, Computational Fluid Dynamics: Principles and Applications, Elsevier Science Ltd, Amsterdam, 2001.
[30] A. Jameson, Solution of the Euler equations for two dimensional transonic flow by a multigrid method, Applied Mathematics and Computation 13 (1983) 327–356.
[31] W.J. Usab, E.M. Murman, Embedded mesh solution of the Euler equations using a multiple-grid-method, AIAA Paper 83-1946, 1983.