AIAA JOURNAL
Vol. 42, No. 9, September 2004

Adjoint-Based, Three-Dimensional Error Prediction and Grid Adaptation

Michael Andrew Park*
NASA Langley Research Center, Hampton, Virginia 23681

Engineering computational fluid dynamics analysis and design applications often focus on output functions, such as lift or drag. Errors in these output functions are generally unknown, and conservatively accurate solutions may be computed. Computable error estimates can offer the possibility to minimize computational work for a prescribed error tolerance. Such an estimate can be computed by solution of the flow equations and the linear adjoint problem for the functional of interest. The computational mesh can be modified to minimize the uncertainty of a computed error estimate. This robust mesh-adaptation procedure automatically terminates when the simulation is within a user-specified error tolerance. This procedure for estimation and adaptation to error in a functional is demonstrated for three-dimensional Euler problems. An adaptive mesh procedure that links to a CAD surface representation is demonstrated for wing, wing–body, and extruded high lift airfoil configurations. The error estimation and adaptation procedure yielded corrected functions that are as accurate as functions calculated on uniformly refined grids with many more grid points.

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

Nomenclature

AR	=	tetrahedral aspect ratio
E	=	total energy per unit volume
f	=	functional, such as lift and drag
h	=	representative element length
I	=	adaptation intensity
n	=	total number of mesh nodes
p	=	pressure
$\boldsymbol{Q}$	=	vector of state variables
$\boldsymbol{R}$	=	flow equation residual
$\boldsymbol{R}_\lambda$	=	adjoint equation residual
s	=	primal and dual solution variables
t	=	user-specified error tolerance
u	=	x component of velocity
v	=	y component of velocity
w	=	z component of velocity
x	=	grid-point location
λ	=	adjoint variable
ρ	=	density
∇s	=	solution gradient

Subscripts

est	=	estimated (error-corrected)
g	=	global sum
s	=	scaled
0	=	normalizing value

Superscripts

H	=	higher-order interpolated
L	=	linear interpolated
0	=	original mesh
$*$	=	infinitely refined mesh

Presented as Paper 2002-3286 at the 32nd Fluid Dynamics Conference, St. Louis, MO, 24–27 June 2002; received 26 August 2002; revision received 8 July 2003; accepted for publication 5 April 2004. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0001-1452/04 $10.00 in correspondence with the CCC.

*Research Scientist, Computational Modeling and Simulation Branch, Mail Stop 128; Michael.A.Park@NASA.Gov. Member AIAA.

Introduction

ENGINEERING problems commonly require computational fluid dynamics (CFD) solutions with functional outputs of specified accuracy. The computational resources available for these solutions are often limited, and errors in solutions and outputs are often unknown. CFD solutions may be computed with an unnecessarily large number of grid points (and associated high cost) to ensure that the outputs are computed to within a required accuracy. A method to estimate the error present in a computed functional offers the possibility to avoid the use of overly refined grids to guarantee accuracy.

Unstructured grid technology promises easier initial grid generation for novel complex three-dimensional configurations compared with structured grid techniques. The use of unstructured grid technology for CFD simulations allows more freedom in adaptation of the discretization of the meshes to improve the fidelity of the simulation. Many previous efforts have attempted to tailor the discretizations of unstructured meshes to increase solution accuracy while reducing computational cost.[1–8]

Most of these adaptive methods focus on modifying discretizations to reduce local equation errors. These local errors are not guaranteed to impact error in global output functions directly. These adaptive methods, often referred to as feature-based adaptation, focus on resolving discontinuities or strong gradients in the flowfield. Flow features, such as shocks, can be in the incorrect location due to errors elsewhere in the flowfield. Also, resolution of the flow in a location may have a minimal effect on the output function, such as a downstream shock. The finite element community has developed methods to adapt a mesh to target the calculation of a functional of interest instead of a local equation error estimate.[9–12]

Pierce and Giles[13] have developed an adjoint-based error correction for functionals computed with finite volume discretizations. If the flow equations are linearized about the flow solution, the solution of a linear dual problem can yield a direct measure of the impact of local primal (flow equation) residual on a selected functional output. Venditti and Darmofal[14,15] Venditti,[16] and Venditti and Darfomal[17] have demonstrated these methods for compressible two-dimensional inviscid and viscous flow solutions.

If a specified error tolerance in an output function is required, the cost of computing a CFD solution can be minimized by adaptation of the discretization of the problem to minimize uncertainties directly in the corrected output function of interest. Also, the entire adaptive simulation can be terminated when the predicted error is equal to a specified tolerance, preventing the waste of computational work on an overly large mesh. The present study is essentially

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

a three-dimensional extension of the method of Venditti and Darmofal[15] that adapts the mesh to reduce uncertainty in an error correction. Müller and Giles[18] have also presented results for a similar approach utilizing a different adaptation parameter that directly targets the error correction.

The use of adjoint variables (solution of the dual problem) is an efficient method for computation of derivatives of a functional of interest for gradient-based design methods. Some examples of discrete adjoint design methods are given by Nielsen and Anderson.[19,20] The discrete dual problem for adjoint variables can be expensive (approximately as much work as the primal problem). However, the adjoint solution is already available for use during the aerodynamic design process, and so it could be employed for simultaneous design, error prediction, and grid adaptation.

The combination of adjoint-based grid adaptation and design techniques can yield an attractive tool for the aerodynamic design of new configurations. Adjoint-based error prediction and adaptation can yield smaller meshes than traditional feature-based schemes with computable error estimates on output functions. Design processes require analysis and derivative evaluation tools that operate with minimal human interaction. Robust, automatic adaptation techniques enable the increased use of nonlinear flow calculations in larger multidisciplinary design frameworks. These new techniques will enable efficient analysis for existing configurations and expanded exploration of design spaces for new configurations.

Flow Equations

The FUN3D[21–23] (http://fun3d.larc.nasa.gov) suite of codes is employed in this study. The compressible flow solver employs an unstructured finite volume tetrahedral method for conserved variables, $\boldsymbol{Q}$, that is,

$$\boldsymbol{Q} = [\rho \quad \rho u \quad \rho v \quad \rho w \quad E]^T \tag{1}$$

The incompressible flow solver employs the following state variables:

$$\boldsymbol{Q} = [p \quad u \quad v \quad w]^T \tag{2}$$

The node-based variables $\boldsymbol{Q}$ are computed by driving the flow equation residual $\boldsymbol{R}$ to steady state with an implicit point-iterative method. The inviscid mode of FUN3D was used in this study. The solution of $\boldsymbol{Q}$ allows the calculation of integral quantities f, such as lift and drag. To speed execution, the problem domain is decomposed and the flow and the adjoint problems are solved with a parallel execution scheme that utilizes the message passing interface (MPI) standard.

Adjoint Equations

After the flow solution is known, the discrete adjoint equations[19,22,23] are solved to complete the dual problem. The first step is to linearize the flow equation residual $\boldsymbol{R}$ and output function f with respect to the flow solution $\boldsymbol{Q}$. After this linearization, an adjoint variable λ is solved for each of the flow equations.

An abbreviated derivation, adapted from Taylor et al.,[24] is now used. The chain rule for the linearized output function is

$$\frac{\partial f}{\partial \boldsymbol{Q}} = \left(\frac{\partial f}{\partial \boldsymbol{R}}\right)^T \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}} \tag{3}$$

The adjoint variable λ is defined as the effect of a flow residual source term on the output function:

$$\frac{\partial f}{\partial \boldsymbol{R}} = \lambda \tag{4}$$

A set of linear equations is solved to find λ:

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right)^T \lambda = \left(\frac{\partial f}{\partial \boldsymbol{Q}}\right)^T \tag{5}$$

After the flow solution is known, this set of linear equations is solved with GMRES.[25] See Refs. 19, 20, and 23 for details. A implicit point-iterative time-marching method is employed to compute the adjoint solution for the high lift configuration.[26,27]

Error Correction

The error prediction and correction scheme is taken from Ref. 15. With a solution on a mesh of reasonable size $\boldsymbol{Q}^0$, it is desirable to predict the value of an output function evaluated with a solution on an infinitely refined mesh $f(\boldsymbol{Q}^*)$. This prediction can be computed without the solution on this infinitely refined mesh when the adjoint solution on this reasonable mesh λ^0 is utilized. The full derivation of the error correction term is available in Refs. 13, 15, and 18. An abbreviated derivation is presented by expansion of the Taylor series about $f(\boldsymbol{Q}^0)$, that is,

$$f(\boldsymbol{Q}^*) = f(\boldsymbol{Q}^0) + \left(\left.\frac{\partial f}{\partial \boldsymbol{R}}\right|_0\right)^T [\boldsymbol{R}(\boldsymbol{Q}^*) - \boldsymbol{R}(\boldsymbol{Q}^0)] + \cdots \tag{6}$$

Employment of Eq. (4) and the assumption that $\boldsymbol{Q}^*$ satisfies the residual on the infinitely refined mesh yields an improved estimate for the functional of interest f_{est}:

$$\left.\frac{\partial f}{\partial \boldsymbol{R}}\right|_0 = \lambda^0 \tag{7}$$

$$\boldsymbol{R}(\boldsymbol{Q}^*) = 0 \tag{8}$$

$$f(\boldsymbol{Q}^*) \approx f_{\text{est}} = f(\boldsymbol{Q}^0) - (\lambda^0)^T \boldsymbol{R}(\boldsymbol{Q}^0) \tag{9}$$

To improve the prediction of the functional f_{est}, $\boldsymbol{Q}^0$ and λ^0 can be interpolated to an embedded mesh. Interpolation is performed in two ways for this study: a linear interpolation ($\boldsymbol{Q}^L$, λ^L) and a higher-order interpolation ($\boldsymbol{Q}^H$, λ^H). Details of this interpolation and the construction of this embedded mesh are in the Interpolation Techniques and Embedded Mesh sections. Substitution of these interpolated quantities into Eq. (9) yields the linear and higher-order functional estimates f_{est}^L and f_{est}^H:

$$f_{\text{est}}^L = f(\boldsymbol{Q}^L) - (\lambda^L)^T \boldsymbol{R}(\boldsymbol{Q}^L) \tag{10}$$

$$f_{\text{est}}^H = f(\boldsymbol{Q}^H) - (\lambda^H)^T \boldsymbol{R}(\boldsymbol{Q}^H) \tag{11}$$

Adaptation Parameter

The adaptation parameter, also from Ref. 15, is intended to specify a grid spacing modification to reduce the uncertainty in the computed error prediction. The grid is not modified to reduce the computed error prediction directly (as in Ref. 18) because an estimate for the functional including this error term can be computed with Eq. (9). Instead, targeting the uncertainty in this computed quantity is more effective and improves the robustness of the adaptive process. The adaptation parameter is formulated to estimate the distribution of the remaining error throughout the mesh, not an exact value. One source of uncertainty in the error correction is due to errors in the original mesh dual solution. The error correction [Eq. (9)], including the uncertainty in the dual solution, is

$$f(\boldsymbol{Q}^0) - f(\boldsymbol{Q}^*) \approx (\lambda^0)^T \boldsymbol{R}(\boldsymbol{Q}^0) + (\lambda^* - \lambda^0)^T \boldsymbol{R}(\boldsymbol{Q}^0) \tag{12}$$

The uncertainty in the computed error correction is

$$f_{\text{est}} - f(\boldsymbol{Q}^*) \approx (\lambda^* - \lambda^0)^T \boldsymbol{R}(\boldsymbol{Q}^0) \tag{13}$$

The relation of the primal and dual problems[13,15] yields another expression for the error correction uncertainty:

$$(\lambda^* - \lambda^0)^T \boldsymbol{R}(\boldsymbol{Q}^0) \approx \boldsymbol{R}_\lambda(\lambda^0)(\boldsymbol{Q}^* - \boldsymbol{Q}^0)^T \tag{14}$$

where $\boldsymbol{R}_\lambda(\lambda)$ is the residual of the dual problem:

$$\boldsymbol{R}_\lambda(\lambda) = \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{Q}}\right)^T \lambda - \left(\frac{\partial f}{\partial \boldsymbol{Q}}\right)^T \tag{15}$$

A computable term is found by the use of the interpolation error of λ to replace $(\lambda^* - \lambda^0)$ and the interpolation error of $\boldsymbol{Q}$ to replace $(\boldsymbol{Q}^* - \boldsymbol{Q}^0)$. The higher-order interpolate for $\boldsymbol{Q}^0$ and λ^0 is employed to improve prediction in place of the linear interpolate in Ref. 15. The interpolation error is expressed as the difference in the high-order and linear interpolated values:

$$(\lambda^* - \lambda^0) \approx (\lambda^H - \lambda^L) \tag{16}$$

$$(\boldsymbol{Q}^* - \boldsymbol{Q}^0) \approx (\boldsymbol{Q}^H - \boldsymbol{Q}^L) \tag{17}$$

Interpolation error is commonly used as an adaptive indicator to target regions of the domain where a linear approximation yields a poor representation of a continuous solution.

The average of the absolute values of the two uncertainty terms in Eq. (14) yields the adaptation intensity $\boldsymbol{I}$, which is computed for each equation on each embedded node:

$$\boldsymbol{I} = \left[\left|(\lambda^H - \lambda^L)^T \boldsymbol{R}(\boldsymbol{Q}^H)\right| + \left|\boldsymbol{R}_\lambda(\lambda^H)(\boldsymbol{Q}^H - \boldsymbol{Q}^L)^T\right|\right]/2 \tag{18}$$

The intensity $\boldsymbol{I}$ is, therefore, the average of the absolute values of two terms. The first term is a dual solution interpolation error weighted with the primal residual. The second term is the dual problem residual weighted with a primal solution interpolation error. This form of the adaptation intensity (which includes weighed interpolation errors) focuses on the nonlinear contributions to the function error, which increases robustness of the adaptation method.

Error Correction and Adaptation Process

The error correction and adaptation process begins with an initial tetrahedral mesh, which can come from any mesh generation system. The state variables are computed as the nonlinear solution to the flow equations on the initial mesh. The adjoint variables are then computed with the linearized flow equations at the flow solution. These flow and adjoint solution procedures employ a parallel execution scheme. Then the global problem domain is reconstructed to facilitate the creation of a finer, embedded grid with interpolated primal and dual solutions.

Embedded Mesh

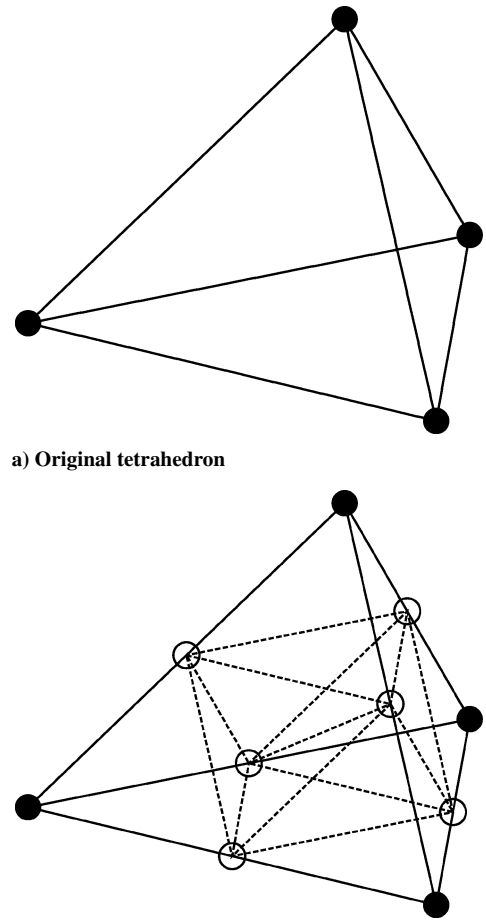
To compute the error prediction and the adaptation parameter, a globally embedded or h refined mesh is created. To construct the embedded mesh, a new node (open circle) is inserted at the midpoint of each existing edge that connects existing nodes (closed circles); see Figs. 1a and 1b. Each existing tetrahedron is subdivided to reconnect these new nodes with eight interior tetrahedra. (Each of the existing boundary faces is also divided into four triangles.)

The four new tetrahedra constructed at the corners of the existing tetrahedron have the same shape as the original tetrahedron but are smaller in size. The construction of the four corner tetrahedra leaves an interior volume with eight faces, which is subdivided into four tetrahedra. The four interior tetrahedra have three unique configurations. The configuration with the lowest maximum dihedral angle is selected.

The new nodes are placed at the midpoints of edges during the mesh embedding process. The embedded nodes on the boundaries of the mesh may no longer remain on the original surface definition of the model. When the grid is adapted to improve the discretization, the surface fidelity of the mesh is maintained with boundary node projection.

Interpolation Techniques

The primal and dual solution variables s, which are all elements of $\boldsymbol{Q}$ and λ, are now interpolated to this embedded mesh. The value of the solution at each of the existing nodes is directly copied into the corresponding nodes of the embedded mesh. Each of the solution variables is interpolated in two ways to form the linear and the higher-order reconstruction for the new nodes. The higher-order reconstruction of the solution for the new nodes requires the computation of least-squares gradients[22] at the existing nodes by the use of the existing mesh and solution. To simplify the three-dimensional interpolation, the interpolation is performed by independently examining each existing edge in the original mesh; thus, the interpolation problem becomes one-dimensional along each existing edge. Each existing edge has an existing node at each edge endpoint and a new node at the edge midpoint (Fig. 2).



a) Original tetrahedron

b) Embedded tetrahedron

Fig. 1 Tetrahedron embedding process.

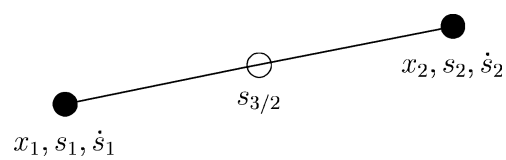


Fig. 2 New node and existing edge.

An edge has two three-dimensional endpoints: $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. The vector that represents the length and direction of the edge is $\Delta\boldsymbol{x} = \boldsymbol{x}_2 - \boldsymbol{x}_1$. The three-dimensional least-squares gradient of the solution ∇s can be projected to a total derivative along the edge to facilitate interpolation by

$$\dot{s} = \nabla s^T \Delta\boldsymbol{x} \tag{19}$$

The new node interpolation $s_{3/2}$ can be expressed as a combination of the solution values and the derivatives at each endpoint (s_1, $\dot{s}_1$, s_2, and $\dot{s}_2$). The linear interpolation $s^L_{3/2}$ is the average of the two end nodes

$$s^L_{\frac{3}{2}} = (s_1 + s_2)/2 \tag{20}$$

The higher-order interpolation $s^H_{3/2}$ is found with a cubic fit of the endpoint data that is evaluated at the midpoint:

$$s^H_{\frac{3}{2}} = (s_1 + s_2)/2 + (\dot{s}_1 - \dot{s}_2)/8 \tag{21}$$

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

Equation (21) is equivalent to a least-squares quadratic fit of the endpoint data that is evaluated at the edge midpoint.

At the completion of the grid embedding and interpolation step, the linear and higher-order interpolated solutions to the primal and dual problems $\boldsymbol{Q}^L$, λ^L, $\boldsymbol{Q}^H$, and λ^H are available to compute the error correction and adaptation parameter.

Error Correction and Adaptation Parameter

Once the new embedded grid is constructed with $\boldsymbol{Q}^L$, λ^L, $\boldsymbol{Q}^H$, and λ^H, it is partitioned to allow parallel calculation of the functional and flow and adjoint equation residuals. The flow and the adjoint equations are not iterated or solved on this embedded grid; the flow state and adjoint variables are interpolated from the original mesh. Therefore, the only computational costs on this larger embedded grid are function evaluations, flow and adjoint residual evaluations, and dot products of vectors. The linear and higher-order error correction term, Eqs. (10) and (11), is computed at each node on the embedded mesh and summed over the entire mesh for all flow equations.

The adaptation intensity, Eq. (18), is also computed at each embedded mesh node. At each node that is also present in the original mesh, the computed intensity is zero due to the chosen interpolation technique. The values of the lower- and higher-order interpolation schemes are the same at these existing nodes; thus, $(\boldsymbol{Q}^H - \boldsymbol{Q}^L)$ and $(\lambda^H - \lambda^L)$ are exactly zero.

To specify the grid adaptation on the original mesh, the adaptation intensities must be reduced from the embedded mesh to the original mesh ($\boldsymbol{I}^0$). The new nodes on the embedded mesh all lie on existing edges of the original mesh (Fig. 1b). Therefore, to construct $\boldsymbol{I}^0$, the original mesh is examined one edge at a time (Fig. 2). One-half of the intensity computed at each new node (which is at the midpoint of these original edges) is added to each existing node at the endpoints of these edges. The intensities are also summed over the equations at this point, resulting in one intensity value for each original node.

The adaptation parameter, which has been reduced to the original mesh, is summed to find the global intensity $I_g = \sum \boldsymbol{I}^0$. The number of nodes in the original mesh n and the user-specified error tolerance t are combined to scale the adaptation intensity; that is,

$$\boldsymbol{I}_s^0 = (I_g/t)(n/t)\boldsymbol{I}^0 \tag{22}$$

To perform the grid adaptation, the mesh is locally enriched in the location of nodes where the scaled intensity $\boldsymbol{I}_s^0$ is greater than a value, that is, unity. Reference 15 demonstrates a way to specify a new element spacing function. The adaptation procedure self-terminates as all elements of $\boldsymbol{I}_s^0$ become less than unity, that is, no nodes are flagged for adaptation.

Adaptation Mechanics

The adaptation mechanics utilize three independent modules. The first module inserts new nodes into the existing mesh and locally reconnects tetrahedra and boundary faces to maintain a valid tessellation. The second module employs face and edge swapping to improve the mesh quality. The final module performs grid smoothing and boundary node projection operations.

Node Insertion

The node insertion method is currently one level of selective h refinement. To start the refinement, all of the edges surrounding nodes on the original mesh that have a scaled intensity $\boldsymbol{I}_s^0$ greater than unity are flagged for h refinement. The set of flagged edges is examined tetrahedron by tetrahedron and additional edges are flagged in an attempt to maintain grid quality, that is, low maximum dihedral angles and smooth element size transition. The final set of flagged edges results in tetrahedra with one edge, three edges on one face, or all six edges flagged. A tetrahedron with all six edges flagged is shown in Fig. 1b. The mesh is then h refined by insertion of new nodes on the midpoint of the flagged edges and reconnection of these nodes into new tetrahedra and boundary faces.

Face and Edge Swapping

The current postadaptation grid-improvement scheme employs face and edge swapping.[28] The swapping algorithm minimizes a shape (cost) function, such as aspect ratio or dihedral angle. This study sought to reduce only the cell aspect ratio $\mathcal{A}\mathrm{R}$:

$$\mathcal{A}\mathrm{R} = \frac{1}{3}\frac{\text{tetrahedral circumsphere radius}}{\text{tetrahedral in-sphere radius}} \tag{23}$$

Reconnections of tetrahedra with undesirable shape measures are investigated and new local tetrahedra configurations with more desirable shape measures are selected. Edges on boundary faces can also be swapped. To simplify and speed up the edge swapping routine, the boundary face information is discarded and reconstructed at the end of the swapping process. Smart–Laplacian smoothing (see Ref. 29) is used on the interior nodes. The actual locations of the boundary nodes is not modified in this module; that modification is performed by the grid smoothing and projection module.

Grid Smoothing and Projection

The inserted boundary nodes may not be located on the surface geometry of the model to be simulated because they were inserted at the midpoints of existing edges. A CAD model is employed to describe the actual model surface. To regain the surface fidelity of the mesh, the newly inserted boundary nodes are projected to the model surface with a CAD interface package, CAPRI.[30] The projection of these new nodes to their location on the CAD surface can result in inverted, invalid tetrahedral elements.

A grid node-smoothing algorithm is employed to facilitate boundary projection without generation of invalid elements and to improve the overall quality (shape measure) of the mesh. This postadaptation smoothing is similar to that by Freitag.[29] Boundary node smoothing is coupled to CAPRI, which utilizes native CAD point projection routines. Node locations on the boundary are adjusted by evaluating points on CAD surfaces to improve the shape measure of adjacent tetrahedra.

As the nodes are projected, the neighboring tetrahedra are tested for validity. If invalid tetrahedra resulted from the projection, the projection distance of the boundary nodes is reduced until the neighboring tetrahedra are valid. Then the nodes in the neighborhood of the projected node are smoothed to improve a quality measure of the adjacent tetrahedra. The boundary points are then moved into the fully projected position in a number of iterative cycles.

It is anticipated that grid smoothing in the neighborhood of projected nodes may not adequately regain surface fidelity of anisotropic meshes. A grid-movement scheme may be required, as in Ref. 20. Another possibility is a three-dimensional version of mesh restructuring, as in Ref. 31.

Adaptation Module Interaction

The current selective h refinement technique often creates high-degree nodes on the border of the adapted regions. A high-degree node is defined as a node with a large number of incident elements, as compared to an average node in the mesh. The smoothing algorithm is currently unable to improve elements that are adjacent to high-degree nodes. The edge and face swapping techniques effectively improve shape measures and reduce the number of high-degree nodes, facilitating projection and node smoothing.

These three adaptation modules were developed independently to facilitate a quick initial implementation and to investigate the strengths and weaknesses of each technique. Merging the abilities of these three separate modules will allow for more flexible modifications of grids, such as point insertion, point removal, and anisotropic elements.[4,6]

Results

Adaptation results are shown for three-dimensional wing, wing–body, and high-lift configurations. The wing is simulated with incompressible and transonic flow conditions. The wing–body and extruded high-lift configurations are simulated with subsonic flow.

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

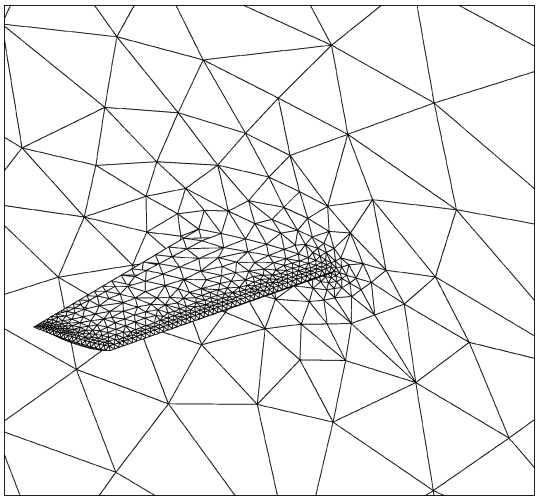
Initial Mesh Generator

The initial meshes for these error prediction and adaptation studies are generated with the FELISA mesher[4] connected to CAD geometry by CAPRI through the GridEx[32] framework. FELISA is a Delaunay mesh generator with an advancing front method for inserting nodes. The GridEx framework is currently being developed at NASA Langley Research Center to link various grid generation and adaptation strategies to geometry through CAPRI. This framework is also utilized in a batch mode to perform uniform grid refinement studies.

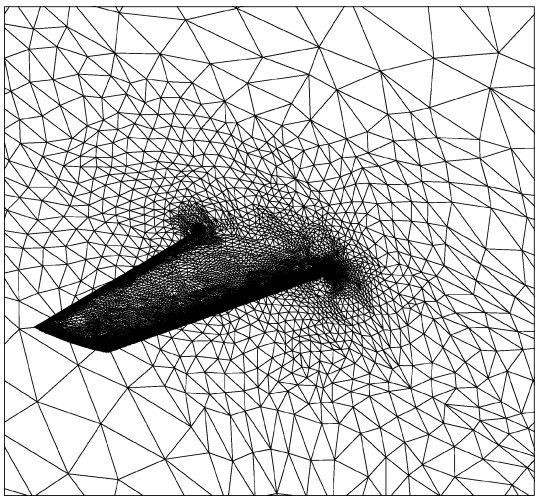
Drag Adaptation: Incompressible ONERA M6

The initial mesh for an ONERA M6 wing with 5227 nodes is shown in Fig. 3a. The mesh has extremely coarse spacing, especially at the trailing edge, and is intended to resolve the surface curvature of the leading edge and the wing tip and not any particular flow features. The spacing function for this mesh is specified manually and is intended to be representative of an automated curvature or maximum chord-height specification. The CAD geometry is represented with the CAPRI native kernel FELISA with a part generated from a surface IGES definition.

The initial ONERA M6 mesh was used in the grid adaptation process with incompressible flow at an angle of attack of 0 deg. Directly computed drag and estimates of drag for an ONERA M6 wing as a function of number of nodes is shown in the Fig. 4 log–log plot. The adaptation and error correction results are shown for a drag error tolerance of 0.001. The directly computed drag on the adapted meshes is represented by the solid lines with circular symbols. The error-corrected drag calculated with the linear interpolated solution f_{est}^{L} is represented by a dashed line and square symbols. The estimated functional calculated with the higher-order interpolated solution f_{est}^{H} is represented by a dashed line and diamond symbols. The correct drag is zero because of nonlifting, subcritical, inviscid flow. Therefore, the y-axis denoted coefficient of drag is also the error in drag. The adaptive procedure correctly self-terminated when the drag error of the adapted flow solution reached the user-specified error tolerance (dot–dash line).



a) Initial ONERA M6 mesh



b) ONERA M6 mesh adapted to incompressible drag

Fig. 3 Initial and adapted ONERA M6 meshes.

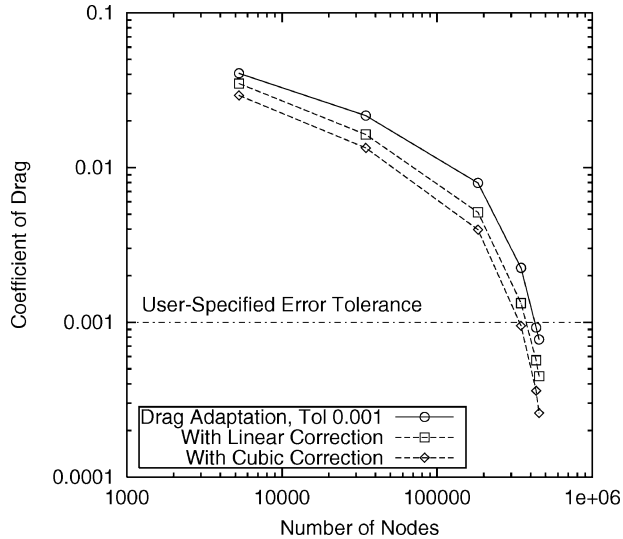


Fig. 4 Coefficient of drag for the ONERA M6 adapted to incompressible drag.

The final grid (454 thousand nodes) after five cycles of grid adaptation to incompressible drag is shown in Fig. 3b. The adaptation process clustered grid points at the leading and trailing edges of the wing. Points are also clustered in the neighborhood of the stagnation stream line.

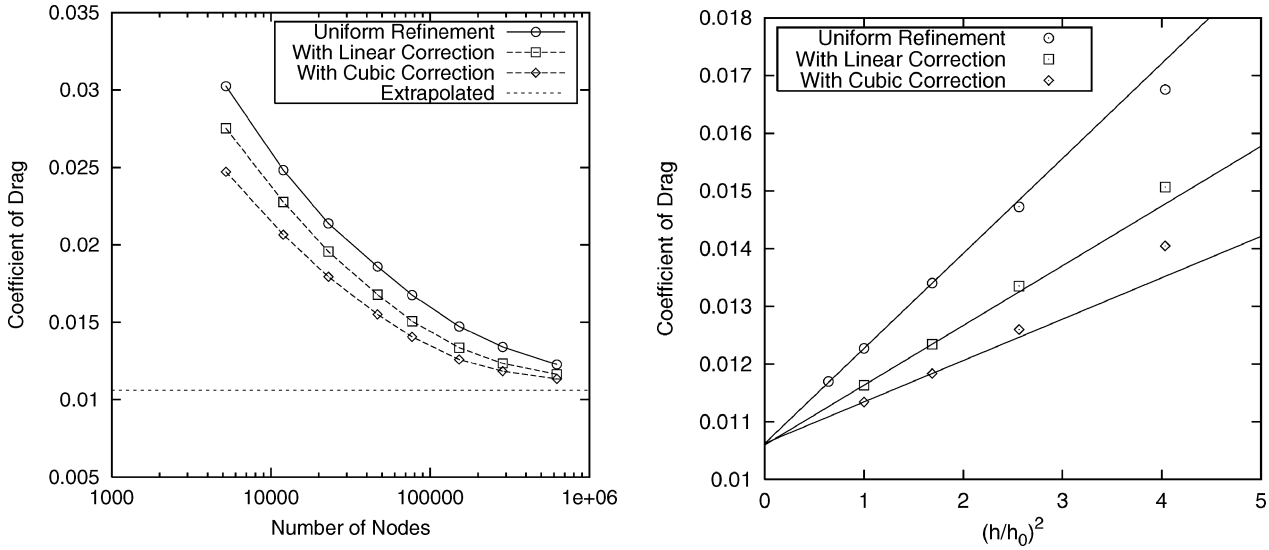
The tetrahedra shape measure AR [Eq. (23)] is minimized by the mesh improvement algorithm. The boundary node smoothing algorithm is intended to optimize the shape measures of the tetrahedral elements. Therefore, the shape measures of the boundary faces depicted in this surface plot may not be optimal.

Drag Adaptation: Transonic ONERA M6

A uniform refinement of the ONERA M6 wing mesh is computed at 0.84 Mach and an angle of attack of 3 deg. The drag directly computed by the compressible flow solver is shown with linear and higher-order interpolated error corrections as a function of the number of nodes in Fig. 5a. The extrapolated (grid-converged) drag value for Fig. 5a was estimated with Richardson extrapolation from Fig. 5b. These meshes have the same spacing function as Fig. 3a, globally modified with a scalar to reduce the element spacing uniformly. These grids were generated with the batch version of GridEx via the FELISA mesher and CAPRI for CAD geometry access.

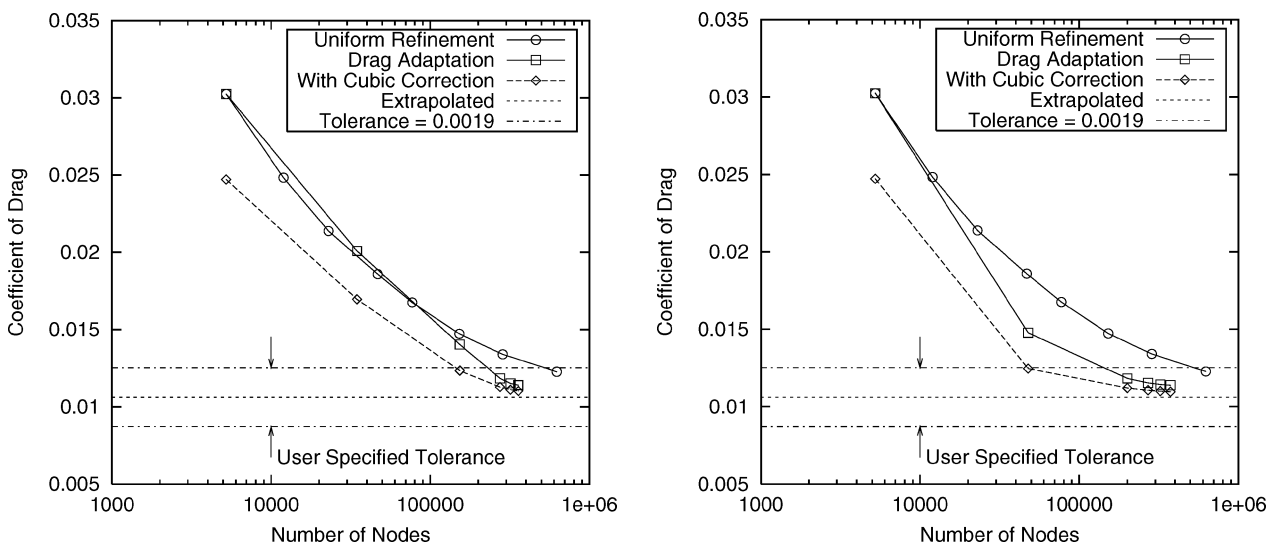
Figure 5b shows drag and estimates of drag as a function of element size for the uniform grid refinement of the ONERA M6 wing. A representative element length h was estimated by one divided by the cube root of the number of nodes. This length was normalized by the estimated length of the 624,000 node mesh h_0. The symbols are drag computed by the flow solver and error corrected values. A linear fit of the data at $(h/h_0)^2 = 1.0$ and $(h/h_0)^2 = 1.7$ is used to estimate the grid-converged answer for all three schemes. All three schemes indicate a similar grid-resolved value. An additional flow solution [1.2 million nodes, $(h/h_0)^2 = 0.6$] is shown to verify the accuracy of the linear fit; it was not used to construct the linear fit of the computed drag.

The uniform refinement study of drag and error correct drag estimates is presented to verify the implementation of the error



a) Uniformly refined drag covergence **b) Richardson extrapolation**

Fig. 5 Coefficient of drag for the uniformly refined ONERA M6 at 0.84 Mach.



a) One level of *h* refinement **b) Two levels of *h* refinement**

Fig. 6 Coefficient of drag for the drag adapted ONERA M6 at 0.84 Mach.

correction scheme and illustrate the dependency of the error correction on the embedded mesh interpolation scheme. The error correction scheme may not be an improvement in efficiency over existing techniques such as Richardson extrapolation, but the greatest advantage of this scheme is the extension to a combined error correction and adaptation methodology.

The initial coarse mesh shown in Fig. 3a is employed in the error prediction and grid adaptation procedure with two different adaptation methods. The coefficient of drag is plotted as a function of mesh size in Fig. 6. The user-specified error tolerance in drag is 0.0019. The uniformly refined flow solution from Fig. 5a is shown with the adapted grid flow solution and higher-order error prediction of the adapted grid in Fig. 6. Figure 6a demonstrates a single level of *h* refinement for all nodes, with a scaled adaptation intensity $\boldsymbol{I}_s^0$ greater than one. Figure 6b shows *h* refinement for $\boldsymbol{I}_s^0$ greater than one and a recursive call to the adaptation mechanics for $\boldsymbol{I}_s^0$ greater than 75, yielding two levels of *h* refinement at each adaptation cycle. The initial convergence of the function is better in Fig. 6b than Fig. 6a. This improvement in function convergence shows the limitations of the use of a single level of selective *h* refinement as the adaptive node-insertion procedure. The use of two levels of *h* refinement better approximates a continuous variation in element size. The two adaptation methods converged to similar meshes and drag values.

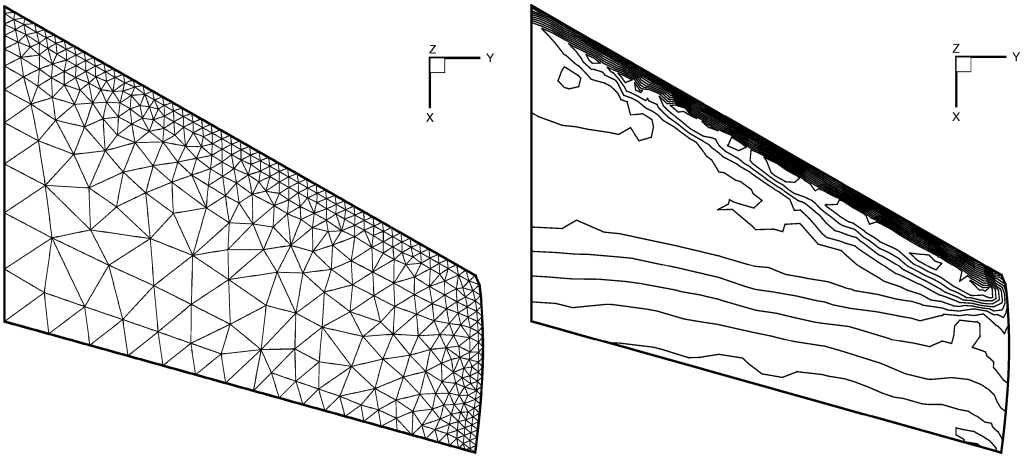
The upper wing surface grid and Mach contours of the initial flowfield computed on the mesh from Fig. 3a is shown in Fig. 7. The shocks in this initial grid are poorly resolved. The mesh (357,000 nodes) and Mach contours of the ONERA M6 adapted to drag with one level of selective *h* refinement is shown in Fig. 8. The mesh (374,000 nodes) and Mach contours of the ONERA M6 adapted to drag with two levels of selective *h* refinement is shown in Fig. 9. The final meshes and solutions are similar for both of the adaptation methods. The adaptive procedure strongly clustered nodes at the leading and trailing edges of the wing and lightly clustered nodes at the shock location on the upper surface. Feature-based adaptations of this configuration in Refs. 7 and 8 focused on the leading edge and shock locations but not the trailing edge.

Lift Adaptation: EET

The energy efficient transport (EET) initial coarse mesh is shown in Fig. 10. The initial grid spacing distribution is specified manually to resolve the surface details of the fuselage, wing leading edge, blunt trailing edge, and wing tip. The geometry is represented with a Parasolid CAD kernel accessed though the CAPRI application program interface (API).

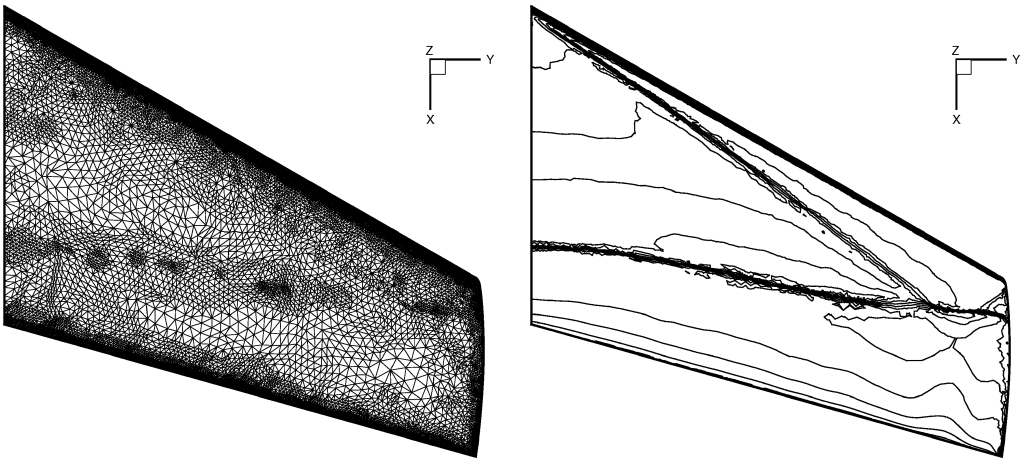
The initial coarse mesh shown in Fig. 10 is employed in the lift error prediction and grid adaptation procedure with compressible flow

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051



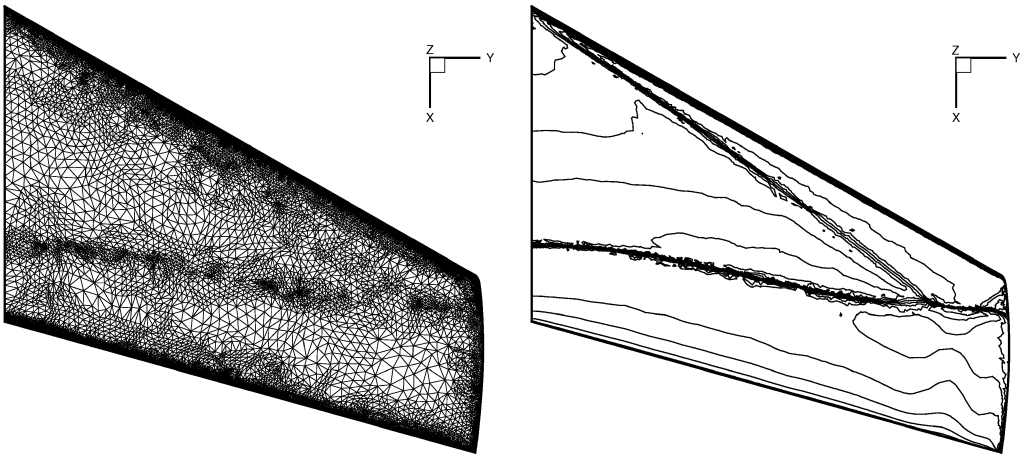
a) Upper wing surface mesh b) Upper wing surface Mach contours

Fig. 7 Initial ONERA M6 upper wing surface.



a) Upper wing surface mesh b) Upper wing surface Mach contours

Fig. 8 Final ONERA M6 upper wing surface, adapted with one level of *h* refinement.



a) Upper wing surface mesh b) Upper wing surface Mach contours

Fig. 9 Final ONERA M6 upper wing surface, adapted with two levels of *h* refinement.

at 0.40 Mach and an angle of attack of 2 deg. The lift coefficient is plotted as a function of mesh size in Fig. 11. The adaptive procedure has a user-specified error tolerance of 0.1 for the lift coefficient error. The uniformly refined lift calculation is shown with the adapted grid lift calculation and error predictions on the adapted grid in Fig. 11. The uniformly refined mesh has not reached the asymptotic convergence range for this complex geometry. The Richardson extrapolation value is not shown because a reasonable linear fit of the last three points was not possible. The lift coefficient is calculated on an adapted mesh one-tenth the size of the uniformly refined grid. The estimated adapted lift error is well within the user-specified tolerance.

Lift Adaptation: Three-Dimensional Extruded 30P-30N Airfoil

The McDonnell Douglas 30P-30N airfoil initial coarse mesh is shown in Fig. 12. The 30P-30N airfoil is extruded between two symmetry planes to form a three-dimensional geometry. The near plane has been removed to improve visualization. The geometry

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051

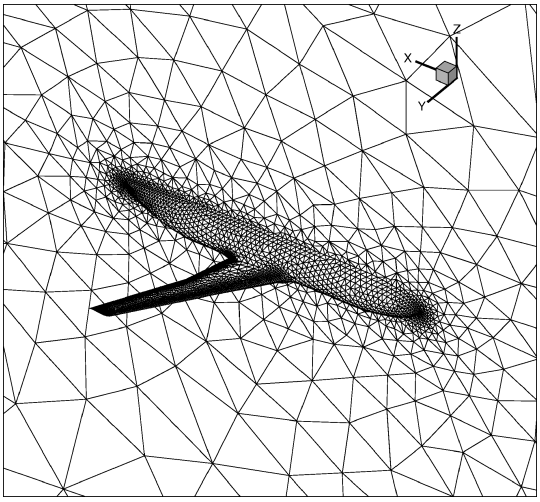


Fig. 10 Initial EET grid.

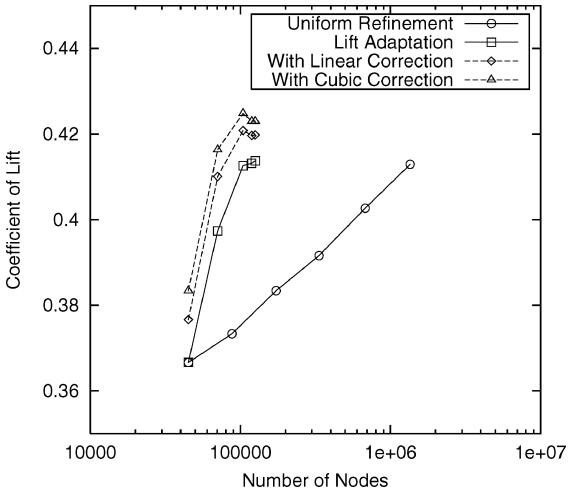


Fig. 11 Coefficient of lift for the lift-adapted EET at 0.40 Mach.

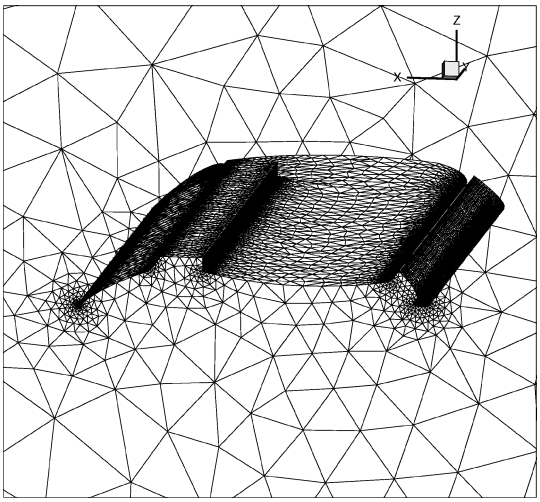


Fig. 12 Initial three-dimensional extruded 30P-30N grid.

is represented with a Parasolid CAD kernel accessed though the CAPRI API. This configuration is the subject of a recent three-dimensional CFD study.[33]

The geometry and initial coarse mesh (113,000 nodes), shown in Fig. 12, is employed in the error prediction and grid adaptation procedure with compressible flow at 0.20 Mach and an angle of attack of 16.3 deg. The lift adaptive procedure has a user-specified error in lift of 0.25. The uniformly refined grid flow solution, adapted grid flow solution, linear error prediction, and higher-order error prediction are shown in Fig. 13. The extrapolated coefficient of lift value was computed with a Richardson extrapolation of the finest two uniformly refined solutions. The original symmetry plane grid and the symmetry plane grid (832 thousand nodes) after two cycles of adaptation is shown in Fig. 14.

Table 1 shows the aspect ratio AR [Eq. (23)] for the initial and the adapted grids. The AR is the cost function for the grid-improvement

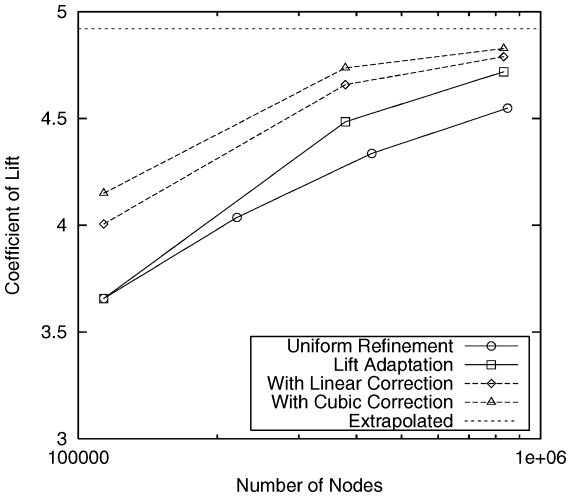
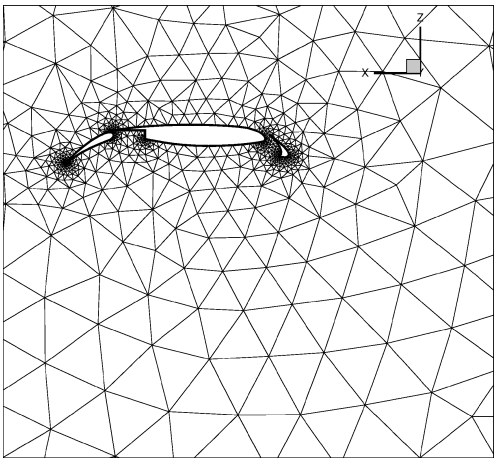
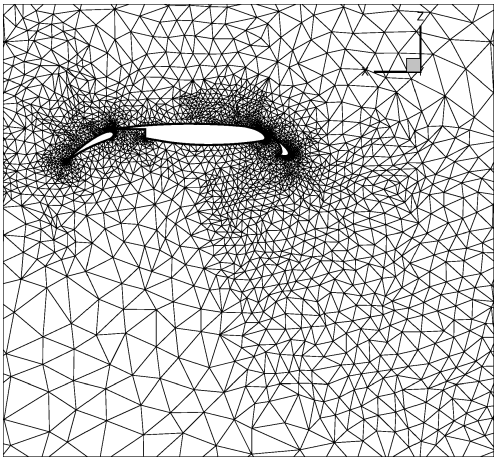


Fig. 13 Coefficient of lift for the lift-adapted three-dimensional extruded 30P-30N at 0.20 Mach.



a) Initial



b) Lift adapted

Fig. 14 Original and adapted three-dimensional extruded 30P-30N symmetry plane grids.

Table 1 Shape measure for the 30P-30N adaptation

Cycle	Aspect ratio AR, max	Face angle, max
1	8.1	154.3
2	5.0	158.1
3	7.5	167.8

optimizer. The face (dihedral) angle is not directly controlled but could be added as a constraint.

Conclusions

The initial implementation of an adjoint-based error correction and adaptation method has been demonstrated in three dimensions. With a given flow and adjoint solution, the error correction for a functional and adaptation intensity term have been described. The adaptation intensity was formulated to reduce the uncertainty in the error correction of a global functional and not a local error estimate as in a feature-based adaptation scheme. The adaptive procedure automatically terminates the simulation when a user-specified tolerance is satisfied. The error remaining in the simulation at termination was always within the user-specified tolerance, although sometimes the simulation was overly accurate.

A wing configuration was adapted to reduce drag error in incompressible and transonic flow. The drag computed by this adaptation and error correction method was shown to be as accurate as direct flow calculations that used larger uniformly refined grids. The initial convergence of adaptation procedure improved with two levels of h refinement at each adaptation cycle. Lift adaptations of Parasolid CAD models of wing–body and high-lift configurations demonstrate the utility of this adaptive methodology on complex geometries. An equivalent value for lift of the wing–body configuration was computed on an adapted grid that is one-tenth the size of an uniformly refined grid.

Acknowledgments

The author thanks Eric Nielsen for his help in employing the FUN3D flow and adjoint solvers, David Venditti and David Darmofal for their suggestions and guidance in error prediction and adaptation, Michael Brasher for his work researching and implementing mesh smoothing, William Jones for help with the GridEx framework, James Thomas for this support of this work, and the Fast Adaptive AeroSpace Tools development team.

References

[1]Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations," *Journal of Computational Physics*, Vol. 72, No. 2, 1987, pp. 449–466.

[2]Baker, T. J., "Mesh Adaptation Strategies for Problems in Fluid Dynamics," *Finite Elements in Analysis and Design*, Vol. 25, 1997, pp. 243–273.

[3]Mavriplis, D., "Adaptive Meshing Techniques for Viscous Flow Calculations on Mixed Element Unstructured Meshes," *International Journal for Numerical Methods in Fluids*, Vol. 34, No. 2, 2000, pp. 93–111.

[4]Peraire, J., and Morgan, K., "Unstructured Mesh Generation Including Directional Refinement for Aerodynamic Flow Simulation," *Finite Elements in Analysis and Design*, Vol. 25, 1997, pp. 343–356.

[5]Braack, M., and Rannacher, R., "Adaptive Finite Element Methods for Low-Mach Flows with Chemical Reactions," *30th Computational Fluid Dynamics*, Vol. 3, von Kármán Inst., Rhode Saint Genese, Belgium, 1999, pp. 1–93.

[6]Löhner, R., and Cebral, J., "Generation of Non-Isotropic Unstructured Grids Via Directional Refinement," *International Journal for Numerical Methods in Engineering*, Vol. 49, No. 1, 2000, pp. 219–232.

[7]Pirzadeh, S. Z., "A Solution-Adaptive Unstructured Grid Method by Grid Subdivision and Local Remeshing," *Journal of Aircraft*, Vol. 37, No. 5, 2000, pp. 818–824.

[8]Park, Y. M., and Kwon, O. J., "Unsteady Flow Computations Using a 3-D Parallel Unstructured Dynamic Mesh Adaptation Algorithm," AIAA Paper 2001-0865, 2001.

[9]Monk, P., and Suli, E., "The Adaptive Computation of Far-Field Patterns by A Posteriori Error Estimation of Linear Functionals," *SIAM Journal on Numerical Analysis*, Vol. 8, 1988, pp. 251–274.

[10]Paraschivoiu, M., Peraire, J., and Patera, A., "A Posteriori Finite Element Bounds for Linear-Functional Outputs of Elliptic Partial Differential Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 150, 1997, pp. 289–312.

[11]Becker, R., and Rannacher, R., "An Optimal Control Approach to A Posteriori Error Estimation in Finite Element Methods," *Acta Numerica 2001*, Vol. 10, Cambridge Univ. Press, 2001, pp. 1–102.

[12]Barth, T., and Deconinck, H. (eds.), *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, Lecture Notes in Computational Science and Engineering, Vol. 25, Springer-Verlag, Berlin, 2002.

[13]Pierce, N. A., and Giles, M. B., "Adjoint Recovery of Superconvergent Functionals from PDE Approximations," *SIAM Review*, Vol. 42, No. 2, 2000, pp. 247–264.

[14]Venditti, D. A., and Darmofal, D. L., "Adjoint Error Estimation and Grid Adaptation for Functional Outputs: Application to Quasi-One-Dimensional Flow," *Journal Computational Physics*, Vol. 164, 2000, pp. 204–227.

[15]Venditti, D. A., and Darmofal, D. L., "Grid Adaptation for Functional Outputs: Application to Two-Dimensional Inviscid Flows," *Journal Computational Physics*, Vol. 176, 2002, pp. 40–69.

[16]Venditti, D. A., "Grid Adaptation for Functional Outputs of Compressible Flow Simulations," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Massachusetts Inst. of Technology, Cambridge, MA, June 2002.

[17]Venditti, D. A., and Darmofal, D. L., "Anisotropic Grid Adaptation for Functional Outputs: Application to Two-Dimensional Viscous Flows," *Journal Computational Physics*, Vol. 187, 2003, pp. 22–46.

[18]Müller, J. D., and Giles, M. B., "Solution Adaptive Mesh Refinement Using Adjoint Error Analysis," AIAA Paper 2001-2550, 2001.

[19]Nielsen, E. J., and Anderson, W. K., "Recent Improvements in Aerodynamic Design Optimization on Unstructured Meshes," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1155–1163.

[20]Nielsen, E. J., and Anderson, W. K., "Aerodynamic Design Optimization on Unstructured Meshes Using the Navier–Stokes Equations," *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[21]Anderson, W. K., and Rausch, R. D., and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers and Fluids*, Vol. 23, No. 1, 1994, pp. 1–22.

[22]Anderson, W. K., Rausch, R. D., and Bonhaus, D. L., "Implicit/Multigrid Algorithm for Incompressible Turbulent Flows on Unstructured Grids," *Journal of Computational Fluids*, Vol. 128, 1996, pp. 391–408.

[23]Nielsen, E. J., "Aerodynamic Design Sensitivities on an Unstructured Mesh Using the Navier–Stokes Equations and a Discrete Adjoint Formulation," Ph.D. Dissertation, Dept. of Aerospace and Ocean Engineering, Virginia Polytechnic Inst. and State Univ., Blacksburg, VA, Dec. 1998.

[24]Taylor, A. C., Green, L. L., Newman, P. A., and Putko, M. M., "Some Advanced Concepts in Discrete Aerodynamic Sensitivity Analysis," *AIAA Journal*, Vol. 41, No. 7, 2003, pp. 1224–1229.

[25]Saad, Y., and Schultz, M. H., "GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal of Scientific and Statistical Computing*, Vol. 7, 1986 pp. 856–869.

[26]Giles, M., Hihai, D., Müller, J., and Pierce, N., "Algorithm Developments for Discrete Adjoint Methods," *AIAA Journal*, Vol. 41, No. 2, 2003, pp. 198–205.

[27]Nielsen, E. J., Lu, J., Park, M. A., and Darmofal, D. L., "An Implicit, Exact Dual Adjoint Solution Method for Turbulent Flows on Unstructured Grids," *Computers and Fluids*, Vol. 33, No. 9, 2004, pp. 1131–1155.

[28]Freitag, L. A., and Ollivier-Gooch, C., "Tetrahedral Mesh Improvement Using Swapping and Smoothing," *International Journal for Numerical Methods in Engineering*, Vol. 40, 1997, pp. 3979–4002.

[29]Freitag, L., "On Combining Laplacian and Optimization-Based Mesh Smoothing Techniques," *Trends in Unstructured Mesh Generation*, Vol. 220, Applied Mechanics Div., American Society of Mechanical Engineers, 1997, pp. 37–43.

[30]Haimes, R., "Automatic Generation of CFD-Ready Surface Triangulations from CAD Geometry," AIAA Paper 99-0776, 1999.

[31]Mavriplis, D., "Turbulent Flow Calculations Using Unstructured and Adaptive Meshes," *International Journal for Numerical Methods in Fluids*, Vol. 13, 1991, pp. 1131–1152.

[32]Jones, W. T., "An Open Framework for Unstructured Grid Generation," AIAA Paper 2002-3192, 2002.

[33]Rumsey, C. L., Lee-Rausch, E. M., and Watson, R. D., "Three-Dimensional Effects on Multi-Element High Lift Computations," *Computers and Fluids*, Vol. 32, No. 5, 2003, pp. 631–657.

K. Ghia
Associate Editor

Downloaded by Politecnico de Milano on March 11, 2024 | http://arc.aiaa.org | DOI: 10.2514/1.10051