Anisotropic Output-Based Adaptation with Tetrahedral Cut Cells for Compressible Flows

by

Michael Andrew Park

B.S., University of Southern California (1998)
M.S., The George Washington University (2000)
Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Aerospace Engineering
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 2008

Author . . . .
Department of Aeronautics and Astronautics
August 8, 2008

Certified by . . . .
David L. Darmofal
Professor of Aeronautics and Astronautics
Thesis Supervisor

Certified by . . . .
Robert Haimes
Principle Research Engineer

Certified by . . . .
Jaume Peraire
Professor of Aeronautics and Astronautics

Accepted by . . . .
Professor David L. Darmofal
Associate Department Head
Chair, Committee on Graduate Students

Anisotropic Output-Based Adaptation with Tetrahedral Cut Cells for Compressible Flows

by

Michael Andrew Park

Submitted to the Department of Aeronautics and Astronautics
on August 8, 2008, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Aerospace Engineering

Abstract

Anisotropic, adaptive meshing for flows around complex, three-dimensional bodies remains a barrier to increased automation in computational fluid dynamics. Two specific advances are introduced in this thesis.

First, a finite-volume discretization for tetrahedral cut-cells is developed that makes possible robust, anisotropic adaptation on complex bodies. Through grid refinement studies on inviscid flows, this cut-cell discretization is shown to produce similar accuracy as boundary-conforming meshes with a small increase in the degrees of freedom. The cut-cell discretization is then combined with output-based error estimation and anisotropic adaptation such that the mesh size and shape are controlled by the output error estimate and the Hessian (i.e. second derivatives) of the Mach number, respectively. Using a parallel implementation, this output-based adaptive method is applied to a series of sonic boom test cases and the automated ability to correctly estimate pressure signatures at several body lengths is demonstrated starting with initial meshes of a few thousand control volumes.

Second, a new framework for adaptation is introduced in which error estimates are directly controlled by removing the common intermediate step of specifying a desired mesh size and shape. As a result, output error control can be achieved without the adhoc selection of a specific field (such as Mach number) to control anisotropy, rather anisotropy in the mesh naturally results from both the primal and dual solutions. Furthermore, the direct error control extends naturally to higher-order discretizations for which the use of a Hessian is no longer appropriate to determine mesh shape. The direct error control adaptive method is demonstrated on a series of simple test cases to control interpolation error and discontinuous Galerkin finite element output error. This new direct method produces grids with less elements but the same accuracy as existing metric-based approaches.

Thesis Supervisor: David L. Darmofal
Title: Professor of Aeronautics and Astronautics

Acknowledgments

My deepest gratitude goes to Professor David Darmofal for providing the critical vision and guidance that has enabled this work. He showed great patience in working with me through countless telephone calls while I have been away from the Institute completing this thesis work. He has invested great effort in helping me to improve the presentation of this work. Robert Haimes provided insight into geometry modeling, grid mechanics and was an active co-adviser who brought a great deal of perspective to the project. I thank Professor Jaume Peraire for his input into the thesis and his insightful class lectures on the finite-volume method. Dr. James Thomas shared the depth of this knowledge of finite-volume techniques and provided detailed thesis improvements. Michael Aftosmis is a thesis reader and the author of excellent papers on the computational geometry required for cut cells and the implications of finite-volume higher-order reconstruction that provided the benchmark to strive toward.

I have enjoyed the members of the MIT Aerospace Computational Design Laboratory as we came together studying for the qualifying exam and attacking research problems or class projects, specifically the Project X team. Todd Oliver is a great friend and officemate. Krzysztof Fidkowski's work on output adaptive simplex cut cells for discontinuous Galerkin and output-based adaptation was a great motivation for this work. Garrett Barter, Laslo Diosady, and JM Modisette answered all of my Project X questions and were lots of fun to hang out with. The ideas and methods of previous ACDL members enabled this effort. This includes David Venditti's work on output-based adaptation for finite volume methods and James Lu's work on dual consistent implicit iterative schemes and output based adaptation for discontinuous Galerkin methods.

I wish to thank the whole FUN3D team for shielding me from the goings on at Langley and letting me be a student first. Eric Nielsen and Kyle Anderson laid a firm flow and adjoint solver foundation on which the cut-cell work is built. Jeff White's effort in improving finite-volume shock capturing and iterative convergence was pivotal in enabling the sonic boom applications. Bill Jones pioneered early sonic boom applications and Ray Mineck accepted my work duties in my absence. Karen Bibb and Beth Lee-Rausch applied the early versions of these adaptation techniques to changing problems. Dana Hammond provided insight into creating efficient sequential and parallel algorithms. Mark Carpenter contributed to stabilizing and accelerating the adjoint solver with Krylov methods. Bil Kleb and Bill Wood shared their interest on how software is tested, how to work in teams, and how to make the actual source code expressive without compromise.

Brooke supported me with love and compassion while letting me move to Cambridge for classes, leaving her with the dog for company. All the telephone conversations and visits every few weeks kept me motivated for class work when I did not believe in myself as strongly as she believes in me. She found the reserve to support me while I was buried in thesis preparation at the same time her family went through the worst year imaginable. My Sister, Mother, and Father supported and encouraged me in countless ways, from sending cookies to sitting through technical presentations. Michael Khandelwal provided technical editing expertise as well as a large care package and a road trip to Cambridge. I thank them all for coming to visit so that I had an excuse to actually see Boston.

NASA Fundamental Aeronautics Program Supersonics Project supports this work. It was previously supported by the EASI Project of Vehicle Systems. A deep gratitude is owed to the managers that have supported and continue to support this work: Peter Coen, Linda Bangert, James Pittman, James Thomas, Mujeeb Malik, and Ajay Kumar.

Contents

List of Figures

Nomenclature

$\bar{u}$	Interpolated Solution
ϵ_κ	Control Volume Error Ratio
ϵ_ω	Global Error Ratio
$\hat{\imath}, \hat{\jmath}, \hat{k}$	Cartesian Unit Vectors
$\hat{n}$	Outward Pointing Normal
$\hat{u}$	Reconstructed Solution
$\mathcal{H}$	van Leer[144] approximate Riemann solver
$\mathcal{L}$	Lagrangian
$\breve{\mathcal{T}}_n$	Grid Tetrahedra Changed by G
tol_Ω	Output Error Tolerance
$\tilde{u}$	Approximate Solution
$\mathcal{T}_n$	Grid Elements
$\mathcal{X}_n$	Grid Nodes
A	Area
C	Collapse Grid Operator
c	Cost Function
E	Total Energy per Volume
e	Interpolation Error
F	Euler Flux
f	Output Function
G	Generic Grid Operator

H	Mach Hessian
h	Length Specification
i	Node Index
J	Linear Mapping
j	Element Index
L	Vector Length Along an Edge
l	Edge Index
M	Grid Metric
m	Norm Power
Mo	Node Movement Grid Operator
N	Number of Control Volumes or Elements
n	Grid Index
p	Pressure
p_∞	Free Stream Pressure
Q	Conserved States
q	Primitive State
R	Residual
S	Source Term
s	Integration Surface
Sp	Split Grid Operator
Sw	Connectivity Swapping Grid Operator
t	Time
u	Exact Solution
u, v, w	Components of Velocity
V	Volume
w	Weight
X	Grid Metric Eigenvectors

η	Inverse of Tetrahedron Mean Ratio[92]
Γ	Control Volume Boundary
γ	Specific Heat Ratio
κ	Element
Λ	Grid Metric Eigenvalues
∇	Gradient Operator
Ω	Domain
Φ	Diagonal Matrix Limiter Function
ϕ	Scalar Limiter Function
ρ	Density
$'$	Transformed Space

Chapter 1

Introduction

Computational Fluid Dynamics (CFD) has become a powerful tool for the analysis and design of aerospace vehicles, but obtaining a suitable grid remains a difficult part of the CFD simulation process. Grids must provide adequate control of discretization error and remain small enough to permit reasonable computation times. The AIAA Drag Prediction Workshops[104] are well-documented examples of how difficult it is to generate grid converged results even by experts utilizing large grids.

Mavriplis[102] continued the study of a Drag Prediction Workshop transport configuration and showed that the discretization errors, estimated through grid convergence studies, were larger than certain modeling errors (e.g., turbulence model distance function calculation, thin-layer viscous approximation, level of artificial dissipation) even for the finest grids, with 65 and 72 million nodes (for a node-based scheme). He also indicated that the grid construction (spanwise stretching) could have a large impact on computed outputs, even for the finest grids.

Chaffin and Pirzadeh[35] detail the effort required to manually specify grid resolution for accurate three dimensional (3D) high-lift computations. The manual specification of the on- and off-body resolution for the grid generation system required a large effort to create. Multiple grids were generated by examining the solution on the current grid and devising better grid resolution specification. The high degree of manual intervention required in these cases prohibits the use of automated design tools. Their work demonstrates the need for better off-body grid control in the grid generation process.[35] The desire to obtain grid-converged results and reduce the manual intervention required by the grid generation process has motivated the development of grid adaptation techniques.

There are two main barriers to the use of grid adaptation in CFD simulation. The first is the ability of the adaptive indicator to produce an improved grid that will eventually yield a solution suitable for engineering application. The second is the robustness of grid adaptation mechanics. Output-based adaptation indicators have significant promise to address the first barrier, see Section 1.1. However, relying on an adaptive methodology to obtain highly resolved grids places greater emphasis on the robustness of adaptive grid mechanics, see Section 1.2.

1.1 Solution Adaptation Indicators

Solution adaptive grid methodologies have been developed and applied in an attempt to reduce the dependence on the manual iteration of CFD processes and the required expertise of the practitioner to obtain accurate results. Baker[14] provides a review of adaptive methods applied to fluid dynamics. The aim is to increase grid resolution where local discretization error estimates are large, with the goal of diminishing these estimated errors. This grid resolution adjustment produces a grid where the error estimates have been equidistributed to some extent. Some examples of local scalar adaptation indicators are interpolation error estimates,[33,65,127] solution derivatives,[40,151] entropy generation,[129] and truncation error.[2]

The local error estimate adaptive process tends to focus on the strong features that are present in the flow solution (e.g., shocks, boundary layers). A potential weakness is under resolution of important smooth regions of the flow, as these smooth regions are generally not the strongest producers of local error indicators. These local adaptation methods neglect the transportation of local errors through the solution and their impact on global output quantities (i.e., lift and drag). Local error adaptation schemes appear to converge to incorrect answers in a number of documented cases.[50,145,151]

One method to account for the transportation of errors is to estimate the error in the calculation of a specified engineering output functional. This output-based approach has been applied to finite element discretizations, including error bounds on outputs and adaptation.[21,63,67,106,134] Pierce and Giles[128] have developed an error correction technique that applies these principles to discretizations other than finite elements.

Output error indicators utilize the dual, or adjoint, solution to account for the transportation of local errors throughout the problem domain, improving the calculation of an output functional. Adaptive techniques have been applied to reduce the output error correction term[50,83,108] for finite volume solutions. Venditti and Darmofal[146,147] demonstrated an adaptive technique for compressible two-dimensional (2D) inviscid and viscous flow solutions that improves the error correction of an output function. The 2D methods of Venditti and Darmofal have been applied to 3D inviscid, laminar, and turbulent problems.[17,113,122,123] This output-based approach provides a natural termination criteria that is based on a user-specified functional error tolerance. Even when starting from coarse initial grids, output-based adaptation has reliably produced accurate simulations.[54,75,147]

These initial grids are often unable to resolve important features that are present in the final, adapted grid. Lower initial grid resolution requirements directly increase the robustness and reduce the person-hours of the initial grid generation process, especially if automated methods that query boundary complexity[132] can specify a suitable initial grid.

Aerospace flow features include discontinuities, shear layers, and boundary layers that require a strongly anisotropic grid to resolve efficiently. The related tasks of specifying a multidimensional grid resolution request and modifying a grid have been decoupled via a mapping metric.[126,127] The use of an anisotropic grid metric has become a standard way to specify a resolution request, but the metric is formed from an *a priori* estimate of interpolation error. Concessions must be made to fit output-based[147] and higher-order[54] methods into the metric framework. The metric output-based adaptation method developed by Venditti and Darmofal[147] utilized primal and dual information to determine the sizing request, but only primal information (Mach or pressure Hessian) to determine the anisotropy. This focus on primal anisotropy ignores the anisotropic nature of the dual solution, which can have a very different character than the primal solution. Fidkowski and Darmofal[54] form a high-order solution adaptation metric by deducing the anisotropic directions of the primal solution with a search methodology. Both the output-based and higher-order metric limitations can be alleviated by optimizing the grid to directly control computed error.

In this thesis, an alternative approach to adapting a grid to a metric is introduced. Specifically, the grid is adapted to directly control output error, which is expressed as a discontinuous Galerkin residual or modeled as interpolation error. Thus, direct control of actual computed error is achieved without the intervening metric specification, which is based on *a priori* error estimates. Bank and Smith[18] have investigated a similar approach for controlling estimated gradient errors of linear elements with 2D node movement.

1.2 Grid Adaptation Mechanics

Baker[14] and Mavriplis[99] have surveyed a variety of grid adaptation methods. A straight forward approach is grid regeneration, where the grid is generated from scratch using a new grid resolution specification.[126,127] Unfortunately, this can be an inefficient method for small, localized changes to the grid resolution request and may require the same level of manual intervention that initial grid generation often requires. A method that is closely associated to full grid regeneration is local regridding.[129] This method opens pockets or voids in the grid by deleting nodes and elements only in the regions that require modification. The voids are completed with a grid generation process that results in the desired change to the grid, but the robustness of this procedure can be very dependent on the shape of these voids.

Element sub-division[100] is the replacement of an element with a set of smaller elements that fill the same region. This method is suitable for tetrahedral as well as mixed element grids. The nested nature of the resultant grids makes this an extremely appealing method for time-accurate[135] and parallel applications[34,118,124,150] because the refined grids are readily coarsened to restore the original grid and the sub-division can be readily expressed in a data structure communicated to other processors.

A popular method of adaptation is Delaunay point insertion.[139] This method begins with an existing Delaunay grid, which is defined by having no triangle circumcircles (tetrahedra circumspheres) that contain a node of the grid. This grid is locally refined by adding nodes in locations in need of finer grid spacing. The element that encloses the new node is split to accept the new node, and the connectivity of nearby elements is modified to regain a Delaunay grid. The new nodes are often added in the location of the circumcenters of poorly shaped elements in order to replace these poorly shaped elements with improved shape elements. The element quality of a Delaunay grid is dependent on the node locations (the connectivity is unique). Nodes that are inserted very near existing nodes can be very detrimental to element quality. Delaunay grids have many advantageous provable properties in 2D that do not hold true in 3D. One disadvantage in 3D is the production of sliver tetrahedra that require post-processing to remove.[89] Unfortunately, maintaining the Delaunay property during coarsening is a slightly more difficult task than Delaunay insertion and Delaunay grids are isotropic.

Iterative node movement is a method that is suitable for structured and unstructured grids. This method iteratively moves nodes to cluster and align elements for satisfaction of the resolution request or to improve element quality.[10,58] The connectivity of the grid is constant, which allows grid modification without the complex data structures required of methods that modify element connectivity. The largest disadvantage of this method is that the element quality can often become very poor without the aid of connectivity changes. Also, locally increasing resolution without adding additional nodes requires the reduction of grid resolution in other portions of the grid.

Edge and face swapping[58] is a technique employed to iteratively improve the connectivity of triangles and tetrahedra without modifying the grid nodes. Local operations replace all the elements surrounding an edge of a face with elements filling the same volume. Grids are improved by selecting their worst elements for swapping, which are then replaced with more desirable elements.

The elemental operations of split, collapse, swap, and node movement can be combined to iteratively modify a grid to satisfy a metric.[65,70,90,141] This approach examines each edge in the mesh and splits or collapses the edge if it is too long or too short as compared with the mapping metric. Edge swaps and node movement are also used to modify the grid to make it more compliant with the requested anisotropic metric or to improve element quality.

Isotropic adaptation techniques have been extended to anisotropic adaptation techniques by adapting the grid in transformed space.[25,26,33,126] This includes the physically-based modeling approach of ellipsoid bubble interaction to iteratively move nodes to locations that result in high aspect ratio elements.[27,155] In 2D, the mapped isotropic methods have been successfully used to create grids with highly anisotropic stretching near curved boundaries for high Reynolds number CFD simulation.[70,86,98]

Adapting 3D grids in the neighborhood of curved boundaries to an anisotropic metric is still a difficult problem. Most of the difficulty lies in simultaneously maintaining the boundary nodes on the geometry description and element validity. The projection of nodes to curved geometry has been applied as a post-processing step after surface refinement to a linear surface approximation. Directly moving the surface nodes to the projected location without modification of the volume grid may result in invalid, inverted elements. The displacements required to project surface nodes can be distributed throughout the grid to prevent the invalidation of elements near surfaces. The volume displacement has been computed via Laplacian smoothing, linear springs, torsional springs, trusses, and linear elasticity. While these methods reduce the incidence of invalid element creation, they are not guaranteed to prevent invalid elements while projecting boundary nodes. Iterative methods have been developed to project the surface node to the "first problem plane"[90,91] (past the first element that becomes inverted during projection) with various combinations of edge swapping and node movement. However, grid regeneration[81] is reported to be necessary when this iterative method fails.

1.3 Cut-Cell Methods

The constraint of generating a body-fitted grid for domains with curved boundaries can be eliminated by introducing these boundaries after grid generation on a simpler domain that does not respect the boundaries. This cut-cell approach allows a background grid to arbitrarily intersect the boundary, and the CFD code is modified to account for the arbitrarily shaped elements resulting from this intersection. Cut-cell methods with Cartesian background grids have become very powerful tools for transonic small disturbance,[24] potential,[156] and Euler[1,37,42,45,48] flows. Extremely high productivity and automation have been demonstrated by Cartesian cut-cell methods,[103,109] because volume grid generation[5] and adaptation[2] are highly automated, even for exceptionally complex geometries. Cartesian background grids permit anisotropy only in the Cartesian directions. This has limited their use to laminar flows,[42] or turbulent flows with the aid of hybrid grids,[46,82] integral boundary methods,[3] or wall functions.[77]

The cut-cell method has also been applied to simplex meshes to permit arbitrary anisotropy.[54,143] When the constraint of providing a body-fitted grid is removed, the grid generation and adaptation task is performed in a domain with much simpler boundaries. Local incremental modification of an existing grid is then able to produce grids with high anisotropy without curved boundary related robustness issues. The extension of cut-cell methods to unstructured tetrahedral grids provides the flexibility to anisotropically resolve arbitrarily oriented flow features.

1.4 Sonic Boom Prediction

Sonic boom simulation can be dramatically impacted by an automated and robust grid adaptation scheme. Strongly anisotropic grids are essential to efficiently provide the required off-body resolution and alignment to improve the accuracy of the one dimensional (1D) approximate Riemann solvers at the heart of finite volume schemes. The acceptance of an aircraft's sonic boom to the general population is a requirement for supersonic flights over land and therefore the commercial viability of a supersonic transport. Predicting how sonic boom signatures are perceived is a challenging task that requires the prediction of the signature on the ground. This is a task complicated by the long propagation distances, atmosphere variations, and the Earth's turbulent boundary layer. A detailed review of the history and state-of-the-art of sonic boom modeling is provided by Plotkin.[130]

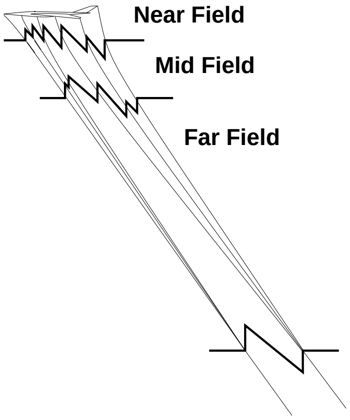


Figure 1-1: Sonic boom signature propagation zones.[30]

The propagation of a sonic boom is often separated into three logical stages or regions, depicted in Fig. 1-1, to facilitate analysis.[30] The near-field is a region near the aircraft, where shocks are formed and strongly influenced by nonlinear phenomena such as shock-shock interaction, shock curvature, and cross flow. Higher pressure portions of the signature travel faster than lower pressure portions of the signature because of variations in the local speed of sound. This slight speed difference causes the shocks to deform by elongating and coalescing in the mid-field. The signature has become primarily an N-wave in the far-field as a result of this distortion. The boundaries of these regions are case specific.

Whitham[153,154] provides analytic solutions for the signal distortion of slender axisymmetric projectiles. The signature is also refracted by variations in the atmospheric speed of sound. These principles have been implemented in a number of boom propagation computer programs.[69,142] Unfortunately, these boom propagation methods are not directly applicable to complex aircraft geometries. Page and Plotkin[121,131] applied the multipoles of George[60] to combine CFD near-field calculation with midand far-field boom propagation. This CFD matching multipole propagation technique has been revisited by Rallabhandi and Mavris.[133]

The methodology of propagating near-field signatures to far-field distances has been used by many researchers. Mendoza and Hicks[71,105] propagated wind tunnel measurements close to various bodies to longer distances to verify their technique. As CFD methods matured, they where used as a source of near-field signatures. Cheung, Edwards, and Lawrence[38] used a hybrid approach with multiple methods to propagate near-field parabolized Navier-Stokes signatures. Cliff and Thomas[41] used structured and unstructured CFD and Madson[97] used an adaptive Cartesian full-potential method to generate near-field signatures for propagation. Djomehri and Erickson[47] evaluated adaptive unstructured grid techniques with propagation. Fouladi[56] manually specified grid resolution for an unstructured grid method to analyze sonic boom. Carter and Deere[32] evaluate various manual grid resolution specification and adaptation techniques. Alonso and collaborators[9,36,39,40] used structured grid and adaptive unstructured grid signatures for propagation and design. Kandil, Ozcer, and collaborators[78–80,119,120] developed and applied various methods to avoid the use of traditional propagation methods.

CFD codes have difficulty propagating the relatively weak pressure signatures of a sonic boom to distances beyond the near-field region, where these boom propagation methods are valid. This problem is more acute for unstructured grid methods that are often employed to capture the geometrical complexity of the model, especially if the grids are not aligned with the shocks. To improve alignment, isotropic unstructured grids are stretched to align the tetrahedra with the free stream Mach angle to improve signal propagation for initial grids.[28] This alignment issue has also given rise to hybrid methods[79,85,149] where near-body unstructured grid solutions are interpolated to shock-aligned structured grid methods to increase accuracy. The hybrid methods are hindered by the interpolation process, so adaptive grid methods[40,94,120] are employed to improve the accuracy of unstructured grid methods for long propagation distances. These previous adaptive methods have used only primal solution information (Mach number, density, and pressure) to drive the adaptive process. Output-adaptive approaches have also been applied to sonic boom prediction in 2D.[19,53,113]

In this work, anisotropic output-based adaptation to improve an off-body pressure integral is applied to 3D sonic boom prediction. Anisotropically adapted tetrahedral background grids with cut-cells provide extremely robust adaptation mechanics, enabling the automated application of anisotropic output-based adaptation to nontrivial 3D sonic boom problems for the first time. This allows for the entire signature to be calculated or the pressure integral can be restricted to a specific region of interest. The adjoint solution can also provide engineering intuition with a rigorous foundation for design sensitivities and discretization error estimates.

1.5 Objectives

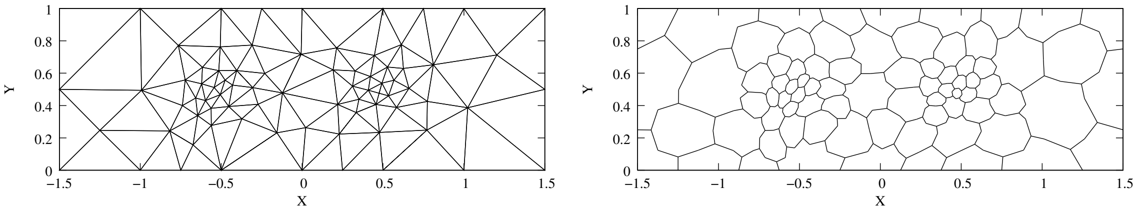
- Develop a robust anisotropic output-adaptive scheme for complex 3D geometries
- Demonstrate capability by performing 3D sonic boom prediction
- Develop methods to directly adapt a grid to improve error estimation without the approximations required for metric-based adaptation

1.6 Overview

This thesis begins with a description of tetrahedral cut cells and how their geometry is determined. The modifications to an existing finite-volume solver required to enable flow and adjoint solutions with these cut cells are detailed. The flow and adjoint problems are combined to form the output-based error estimation and correction technique. A metric-based adaptation method is described and sonic boom predictions, computed with a parallel execution, are provided to illustrate this method's utility. The assumptions required to formulate metric-based adaptation for outputs and higher-order solutions are eliminated with an approach to directly control the output error. This direct approach is demonstrated on a series of test problems. Finally, a summary and conclusions are provided with the contributions of this thesis.

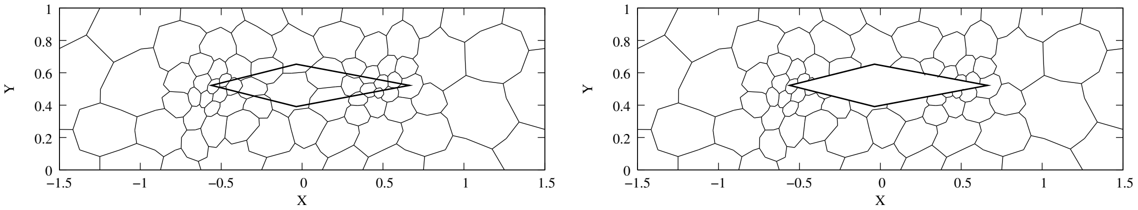
Chapter 2

Cut Cell Determination



(a) Primal grid.

(b) Median dual grid.



(c) Median dual grid with geometry.

(d) Resulting cut-cell grid.

Figure 2-1: Cut cell illustration of an diamond airfoil in 2D.

To introduce the 3D cut-cell method a simple 2D example is presented. The primal triangular grid is shown in Fig. 2-1(a). The control volumes used by the flow solver are the median duals of this triangular mesh. The median duals of this triangular grid are shown in Fig. 2-1(b). These median duals are constructed by gathering the three dual faces that are inside each primal triangle. Figure 2-2 shows the three dual faces (dashed lines) for a triangle, which each connect the triangle center to one of the triangle side midpoints. The airfoil geometry is a diamond airfoil, shown with the uncut median dual background grid in Fig. 2-1(c). The airfoil geometry is Boolean subtracted from this background grid removing the portion of the background grid that is external to the flow domain, Fig. 2-1(d).

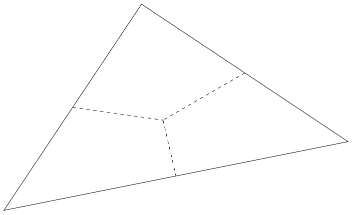
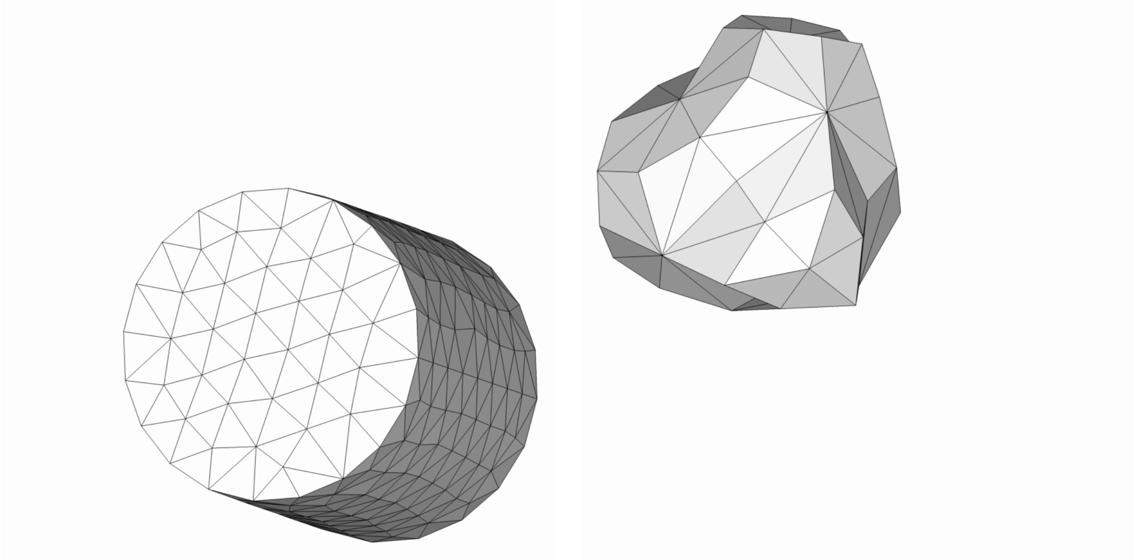


Figure 2-2: Three dual faces (dashed lines) associated with a single triangle (solid lines).

In the 3D case, the domain of the simulation is constructed by Boolean subtraction of a manifold triangular boundary representation from a background grid. This triangulation can come from many sources. Two examples are CAD geometry[6, 66, 74] and component-based geometry.[5] Figure 2-3(a) is a triangular surface grid of a cylinder constructed on a CAD solid.[74]



(a) Cylindrical cutting surface.

(b) Median dual.

Figure 2-3: Cylindrical cutting surface and median dual.

The background grid contains closed simplicial polytope control volumes. In 3D, these polyhedra are the median duals of a tetrahedral grid. The 3D median dual about a single primal node is shown in Fig. 2-3(b). Just as in the 2D case, this dual control volume may not be convex. Each dual polyhedra of a tetrahedral grid contains $\mathcal{O}(100)$ triangles. Figure 2-4 illustrates the two dual triangles associated with an edge of a primal tetrahedron. There are six edges in a tetrahedron, which contains a total of 12 triangular dual faces shared by the duals at each of its 4 nodes. For robustness and a decrease in execution time and memory usage, a triangular dual face is only represented once in the intersection procedure and shared by the two adjacent control volumes.

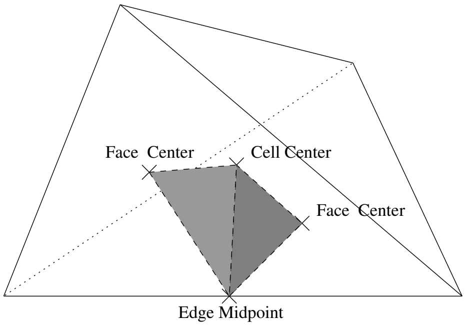
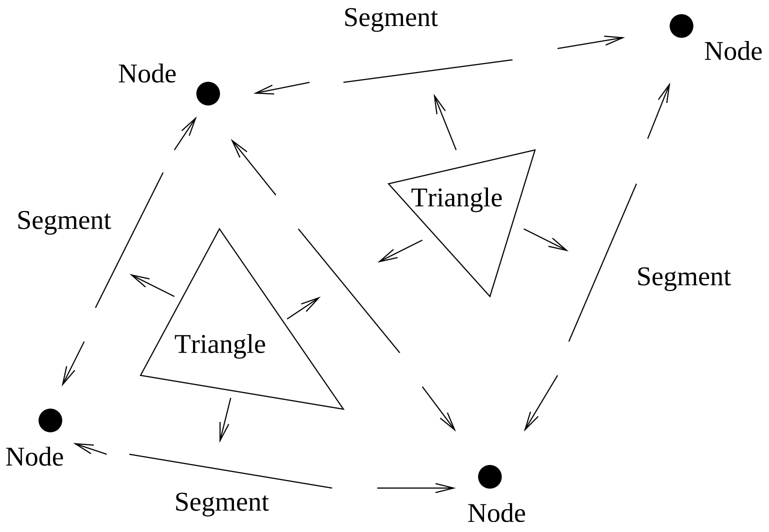


Figure 2-4: Two dual triangles associated with a single tetrahedral edge.

The Boolean subtraction of two manifold triangular polyhedra (surface grid and each of the background grid control volumes) reduces to a series of triangle-triangle intersections.[5] For computational efficiency, a near tree[13,76] is employed to only perform the intersection test for triangle pairs that have a possibility of intersecting. Only the duals that have a potential of intersecting the cut surface are created for the intersection test. This potential is determined by an approximate, but conservative, intersection of the primal tetrahedra and the cut surface triangulation. These steps reduce the complexity of the intersection problem that can be $\mathcal{O}$(number of surface triangles × number of volume triangles) for a naive implementation.

Aftosmis, Berger, and Melton[5] describe a procedure to characterize the intersection of two triangles with a sequence of signed volume computations. An *a posteriori* estimate of signed volume floating-point round-off error is also provided, and the adaptive precision arithmetic procedure of Shewchuk[138] is invoked when the round-off error exceeds the absolute value of the signed volume calculation. Cases that are degenerate in exact arithmetic are perturbed with a unique virtual displacement to resolve these degeneracies.[5] This exact intersection algorithm produces the same intersection determination with any underlying math model (e.g., processor architecture, compiler). If it is acceptable to allow the intersection determination to be dependent on the math model, floating-point arithmetic can be used as long as the geometry predicates are evaluated in a consistent manner.



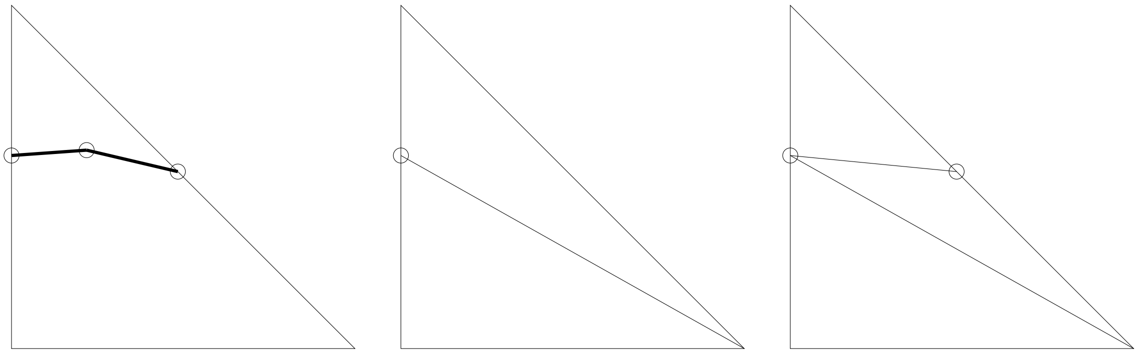
```
Triangle { Segment s0, s1, s2 }
Segment { Node n0, n1 }
Node { Real x, y, z }
```

Figure 2-5: Hierarchy of node, segment, and triangle data structures.

In the present work, the intersection test of two triangles is evaluated with only floating-point arithmetic. To ensure that each intersection test is only performed in a single orientation, a hierarchical data structure is employed, see Fig. 2-5. A triangle consists of 3 segments; a segment consist of two nodes. Two adjacent triangles in the surface or background grids share these segments. The triangle-triangle intersection determination decomposes into triangle-segment intersection determination via volume computations. The triangles and their segments have a unique orientation because they are only represented once in the data structure. When they are compared in the unique configuration, they always return the same intersection determination. This consistency is pivotal to the robustness of the cutting scheme and permits the use of floating-point arithmetic, which is faster and much simpler than utilizing adaptive precision arithmetic. Performing the same intersection test in different orientations would make an inconsistency extremely likely because of differing floating-point round-off errors.

In the rare case when floating-point arithmetic results in an exact degeneracy (zero volume), the cut surface is perturbed slightly (by a factor of machine epsilon) and the entire cutting procedure is restarted to maintain consistency. These exact degeneracies have only been observed on initial grids when the user exactly aligns the background grid and surface triangulation. The actual perturbation can be replaced with a virtual perturbation,[5,51] where a unique tie-break to the degenerate volume computing determinate is applied. This process has not been adopted in the current work, because the degeneracies are detected early in the cutting procedure (within seconds) and a single perturbation vector applied to the entire surface triangulation has been sufficient to eliminate the degeneracies. The virtual perturbation technique may be implemented as a topic for future work if it becomes necessary.

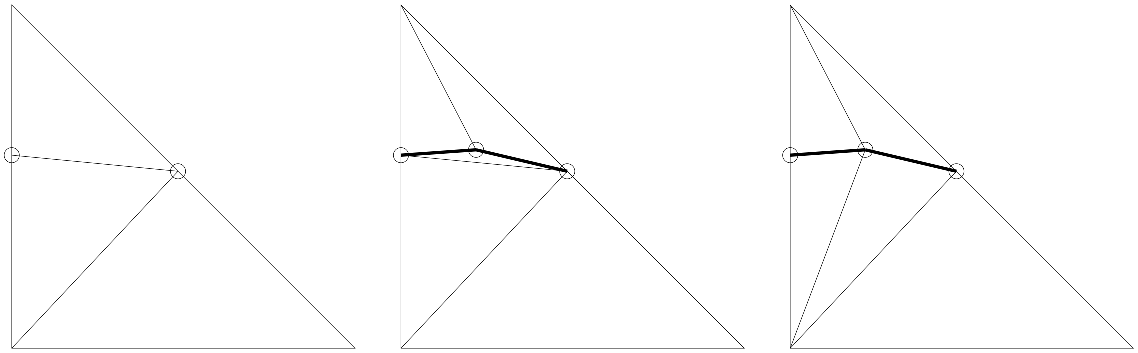
Each of the intersected triangles is constrained Delaunay triangulated into a set of subtriangles to include the intersection points and segments.[5] Local Barycentric coordinates are employed in a reference triangle, Fig. 2-6(a). The goal is to produce a Delaunay triangulation in the reference triangle, not the physical triangle. In this example, two cut segments (thick lines) and three intersection points (circles) are introduced into this reference triangle. The triangle-triangle intersection points are inserted, one at a time, into the triangles with an iterative method.[64] A point insertion involves splitting the subtriangle that surrounds the new point into two or three, Fig. 2-6(b). The target subtriangle that will be split is selected by examining the subtriangles that result after splitting. This target subtriangle is selected so that the smallest of the resultant subtriangles has the greatest signed area in floating-point arithmetic. This area calculation is always performed in a unique configuration so the resulting triangulation will have a non-negative area in finite-precision arithmetic. The subtriangles are provided to the flow solver in this same orientation preventing the introduction of negative area triangles in the flow solver. Shewchuk[137] describes



(a) Reference triangle, intersection points, and cut segments.

(b) Insert first intersection.

(c) Insert second intersection.



(d) Swap for Delaunay.

(e) Insert third intersection.

(f) Swap for Delaunay.

Figure 2-6: Subtriangle construction by Delaunay point insertion into reference triangle.

a Delaunay triangulation scheme that uses exact arithmetic for area calculations, but can provide negative area triangles (in floating-point arithmetic).

After insertion, the subtriangle sides that violate the in-circle test are swapped to regain a Delaunay grid, Fig. 2-6(d) and Fig. 2-6(f). This swapping propagates until a triangle side that satisfies the in-circle test is reached. Cut segments that are not present after the points are inserted are recovered[61] producing a constrained Delaunay grid of the reference triangle.

Once all of the intersection points and segments have been inserted into their parent triangles, the next step is to determine which subtriangles are inside or outside of the domain. Each cut has four adjacent subtriangles, see Fig. 2-7. The subtriangle pair, S1 and S2, lie on a triangle from the boundary surface triangulation, and the subtriangle pair, V1 and V2, lie on a triangle from the face of a background dual volume. Thus, each subtriangle in a pair are in the same plane since they have the same parent triangle.

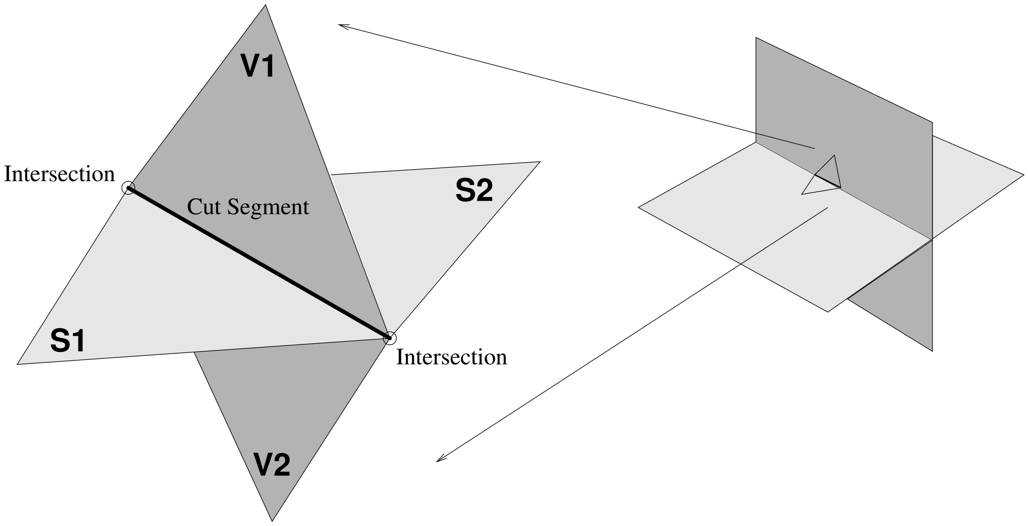


Figure 2-7: Inside/outside determination of subtriangles at cut segment.

The normals of the cut surface triangles (S1 and S2) point into the domain. The signed volume of a tetrahedron formed from the nodes of an S subtriangle and the third node of a V subtriangle are positive if the V subtriangle is inside the domain. The V subtriangle that creates a positive volume tetrahedra is given an inside status. The V subtriangle that creates a negative volume tetrahedra is given an outside status.

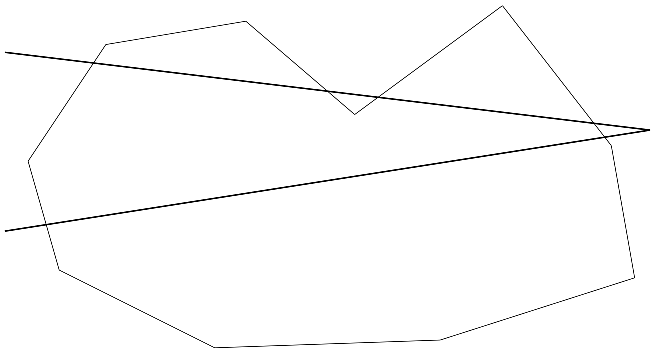
The inside/outside status of the S subtriangles is determined in the same fashion, each dual knows the orientation of its V subtriangle normals by construction.

The inside/outside determination procedure uses only local subtriangles. It does not use a global search over all triangles, which is required by ray-casting.[5] The localization of the inside/outside determination allows the current implementation to utilize computer memory cache more efficiently than ray tracing. The inside subtriangle status is propagated to adjacent subtriangle and uncut triangle neighbors with a flood-fill scheme, which propagates status until a cut segment is reached. The validity of the cut cell topology is verified during the flood-fill operation. If a subtriangle pair is set to the same status (i.e. V1 and V2 in Fig. 2-7 are both inside) the flood-fill is terminated and the user is alerted to the location, because this indicates that the cutting surface is not manifold.

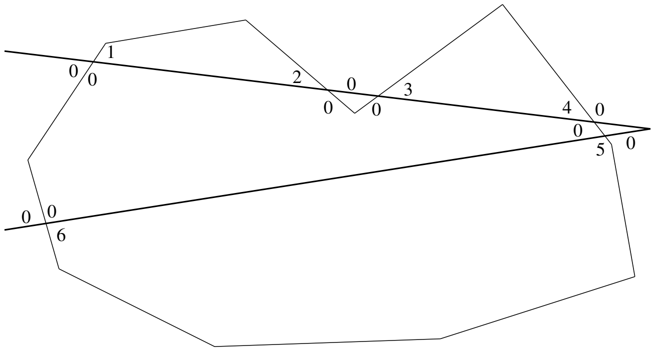
An example is provided to illustrate the inside/outside determination of a multiple region cut cell in 2D. Figure 2-8(a) contains a non-convex background grid control volume and a wing trailing edge (thicker line). The cutting and inside/outside determination is applied and segments adjacent to cuts are given an integer, Fig. 2-8(b). This integer is 0 for segments outside of the domain and positive for segments inside of a domain. The positive integers used to mark inside segments are unique. A relaxation is performed so that adjoining segments not separated by a cut are both set to the larger of their two integers. This provides inside/outside determination for segments that are not directly cut and categorizes the segments into distinct regions, Fig. 2-8(c), where each region has a distinct integer. These distinct regions are each represented as a separate control volume in the flow solver.

The 3D cut surface and dual volume surface grids from Fig. 2-3 are shown in Fig. 2-9(a). The cut surface intersecting the median dual is shown as a wire frame so that the median dual is visible. Figure 2-9(b) shows the result of the Boolean subtraction. The surface resulting from the subtraction contains the inside triangles from both surfaces and the inside subtriangles of the intersection.

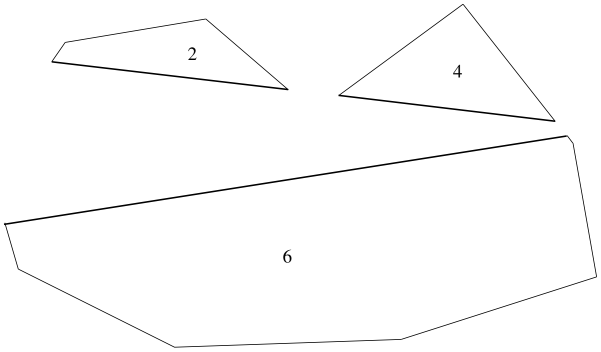
The time required for cut-cell preprocessing of the Section 6.5 adaptive cases is provided in Fig. 2-10. The background grids are adapted and the same body trian-



(a) Background grid control volume and wing trailing edge.

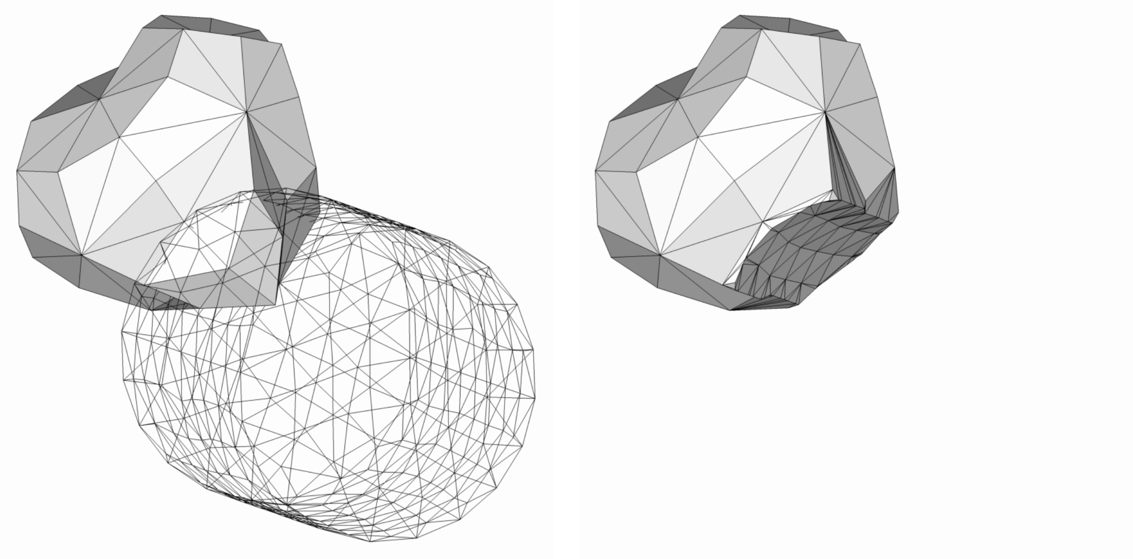


(b) Inside/outside determination at intersections.



(c) Completion of flood-fill resulting in 3 regions.

Figure 2-8: Inside/outside and multiple region determination in 2D.



(a) Median Dual And Cutting Surface.

(b) Resulting Polyhedra.

Figure 2-9: Median dual, cylindrical cutting surface, and resulting polyhedra.

gulation is Boolean subtracted from each background grid. The cutting procedure utilizes a parallel execution scheme. The number of control volumes that are in the domain after cutting is the x-axis. The time required includes disk access time to read in the background and surface grids over the network from a single file server.

The longest running delta wing body case is investigated to determine why it required more time than the other cases with more surface triangles. All of the steps before the flood-fill scheme required 50 seconds. The flood-fill scheme is load imbalanced for this case. All but 4 of the 24 processors completed the flood-fill before 1000 seconds. One of the processors completed flood-fill in 80 seconds. The grid is partitioned with a heuristic that equalizes the number of cut and uncut control volumes, Section 5.2.6. A better heuristic may account for the relative cost of computing different cut cells. The flood-fill algorithm could also be optimized to reduce total cutting time.

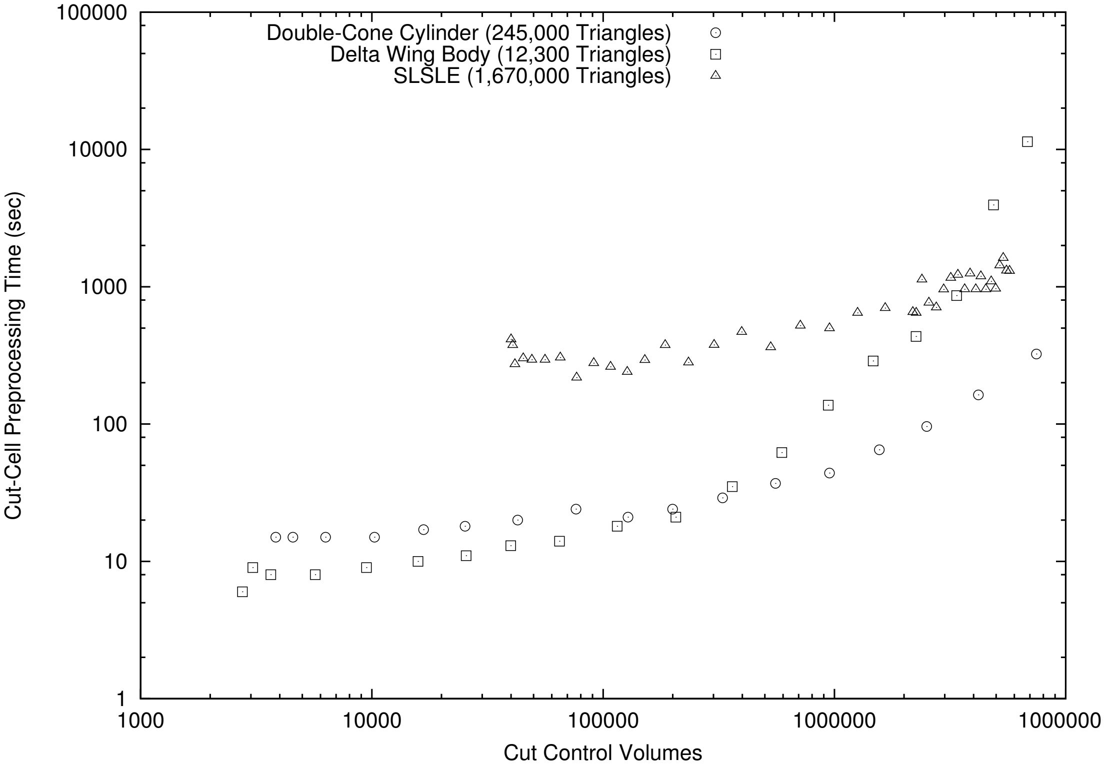


Figure 2-10: Time required for cut-cell preprocessing as a function of number of control volumes in resultant grid.

Chapter 3

Flow and Adjoint Solvers

Fully Unstructured Navier-Stokes Three-Dimensional (FUN3D) is a suite of codes for finite-volume CFD.[11,12] The FUN3D website, `http://fun3d.larc.nasa.gov`, contains the user manual and an extensive list of references. FUN3D is able to solve incompressible, Euler, and Reynolds-averaged Navier-Stokes (RANS) flow equations, either tightly or loosely coupled to a turbulence model. The Euler equations are used in this study. Domain decomposition is employed to fully exploit the distributed memory of a cluster of computers to increase problem size and reduce the execution time of the simulation process.

3.1 Governing Equations

The Euler equations are

$$\frac{\partial \mathcal{Q}}{\partial t} + \nabla \cdot F = 0, \tag{3.1}$$

$$\mathcal{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}, \tag{3.2}$$

$$F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ u(p+E) \end{bmatrix} \hat{i} + \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v w \\ v(p+E) \end{bmatrix} \hat{j} + \begin{bmatrix} \rho w \\ \rho w u \\ \rho w v \\ \rho w^2 + p \\ w(p+E) \end{bmatrix} \hat{k}, \tag{3.3}$$

where ρ is density, u, v, and w are velocity, E is total energy per unit volume, and p is pressure. These quantities are related by the ideal gas relation,

$$p = (\gamma - 1)\left(E - \rho\frac{u^2 + v^2 + w^2}{2}\right), \tag{3.4}$$

with the specific heat ratio $\gamma = 1.4$ for air.

The divergence theorem is applied over a set of control volumes to produce a finite-volume scheme,

$$\int_{V_i} \left(\frac{\partial \mathcal{Q}}{\partial t} + \nabla \cdot F\right) \, \mathrm{d}V = V_i \frac{\mathrm{d}\mathcal{Q}_i}{\mathrm{d}t} + \int_{\Gamma_i} F \cdot \vec{n} \, \mathrm{d}\Gamma = 0, \tag{3.5}$$

where Γ_i are the boundaries of the control volumes with volume V_i and $\vec{n}$ is an outward pointing normal. The average of $\mathcal{Q}$ in each control volume is $\mathcal{Q}_i$. The flux integration is approximated as,

$$\int_{\Gamma_i} F \cdot \vec{n} \, \mathrm{d}\Gamma \approx \sum_{f \in \Gamma_i} \mathcal{H}(q_{lf}, q_{rf}, \vec{n}_f) A_f = R_i(Q), \tag{3.6}$$

where R_i is the discrete residual for control volume i, the summation is over the faces of the control volume.[11] The van Leer[144] approximate Riemann solver $\mathcal{H}$ is utilized to compute the flux from the primitive states,

$$q = \begin{bmatrix} \rho \\ u \\ v \\ w \\ p \end{bmatrix}, \tag{3.7}$$

at the borders of the neighboring control volumes, $q_{r\,f}$ and $q_{l\,f}$. These face values are reconstructed from cell averages (the reconstruction method is described below). The discrete equations are established simultaneously for each control volume,

$$V\frac{\mathrm{d}Q}{\mathrm{d}t} + R(Q) = 0, \tag{3.8}$$

which makes the discrete solution vector $Q \in \mathcal{R}^{5N}$, discrete residual vector $R \in \mathcal{R}^{5N}$, and $V = \operatorname{diag}(V_i)$, where N is the number of control volumes. The flux integration scheme (including face state reconstruction from cell averages) is detailed in the following sections.

A backward Euler solution update scheme is employed with a variable pseudo-time step.[11] An approximate nearest neighbor linearization is utilized to reduce the memory required for the implicit point-iterative method. The solution update is limited to 15% of the current ρ and p to increase robustness during initial transients when starting from free stream.

3.2 Adjoint Equations

After the flow solution is known, the discrete adjoint equations[114,115] are solved to complete the dual problem. To derive these equations, the residual R is augmented with a source term S and the sensitivity of the output function with respect to this source term is sought. A potential application of this sensitivity is to model truncation error as a source term and determine the effect of this error on the output function f (e.g., lift, drag). A Lagrangian is formed from f with R included as a constraint,

$$\mathcal{L}(Q, S, \lambda) = f(Q, S) + \lambda^T R(Q, S). \tag{3.9}$$

Each adjoint variable $\lambda \in \mathcal{R}^{5N}$ corresponds to a flow equation. The Lagrangian is differentiated with respect to S,

$$\frac{\partial \mathcal{L}}{\partial S} = \frac{\partial f}{\partial S} + \left(\frac{\partial Q}{\partial S}\right)^T \left(\frac{\partial f}{\partial Q} + \lambda^T \frac{\partial R}{\partial Q}\right) + \lambda^T \frac{\partial R}{\partial S}. \tag{3.10}$$

The dependency on $\frac{\partial Q}{\partial S}$ can be eliminated by solving the adjoint equation,

$$\left(\frac{\partial R}{\partial Q}\right)^T \lambda = \left(\frac{\partial f}{\partial Q}\right)^T. \tag{3.11}$$

With the adjoint equation satisfied, the sensitivity of the output function can be expressed as,

$$\frac{\partial \mathcal{L}}{\partial S} = \frac{\partial f}{\partial S} + \lambda^T \frac{\partial R}{\partial S}. \tag{3.12}$$

The linear adjoint equations in Eq. (3.11) are solved with a dual-consistent time-marching method.[62,117] The dual-consistent solution method guarantees that the adjoint equations will have the same asymptotic convergence rate as the flow equations.

3.3 Inviscid Flux Integration

The existing FUN3D body-fitted approach lumps the median dual pieces to a single effective area and normal direction for each edge they surround.[22] After lumping, all of the inviscid terms are calculated with a loop over edges, which is computationally efficient. Conserved states Q, used in the time advancement scheme, are converted to primitive states q for face state reconstruction. The primitive state is extrapolated from the nodes to establish the primitive state at these lumped faces q_f using the gradients $\nabla q = [q_x, q_y, q_z]$ reconstructed from the cell averaged state q_0 (see Section 3.4), face center x_f, and node x_0,

$$q_f = q_0 + \nabla q \, (x_f - x_0), \tag{3.13}$$

for the unlimited scheme. For the case of supersonic flow, a limiting function is used to reduce the gradient contribution to the reconstruction (see Section 3.5).

At the completion of cut-cell preprocessing, the dual polyhedra can be classified into three groups: uncut active duals interior to the computational domain, cut duals, and inactive uncut duals exterior to the computational domain. The state is stored

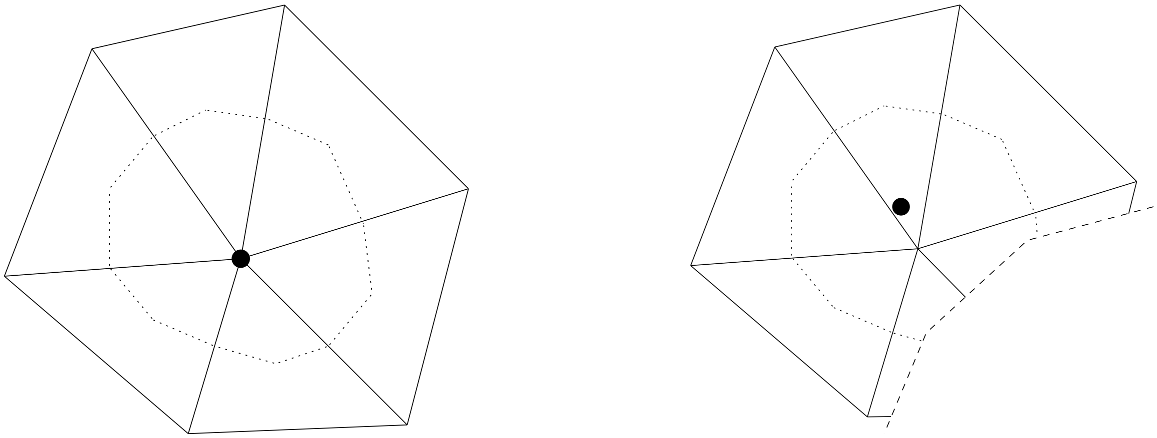

(a) Uncut control volume.
(b) Cut-cell control volume.

Figure 3-1: Dual control volumes, in 2D.

at each node in the primal grid, Fig. 3-1(a) filled circle. All nodes that correspond to dual polyhedra that have been cut or are inactive are removed. A new degree of freedom is inserted at each cut dual polyhedra centroid, Fig. 3-1(b) filled circle. Multiple degrees of freedom are added when a polyhedra is split into multiple distinct regions by the cut surface.

Once a dual control volume is cut, the approximation that the state is centered at the primal node is removed and the state becomes centered at the control volume centroid. This results in a discontinuous change in location once the control volume is infinitesimally cut. This discontinuous behavior may cause difficulties for shape sensitivities and design. Removing this issue remains a topic for future work, but may be addressed by computing the uncut cell centroids.

The median dual triangles that surround any edges that involve a cut cell are explicitly represented and employed in flux integration. Edges that involve uncut active duals utilize the lumped effective areas and normals of the body-fitted scheme. Cut-cell flux integration requires more work than the body-fitted scheme because there are multiple triangles separating the two control volumes that would be approximated as a single flux evaluation in the body-fitted scheme. It also requires more memory to store the extra triangles that would be approximated as a single effective area. The cut cells are a minority of the control volumes for a typical case, so the additional expense of utilizing cut cells does not dominate the execution time or storage.

The body-fitted, node-based scheme stores the state on the boundary of the domain. The state is interpolated between adjacent boundary nodes to integrate the boundary flux. The boundary flux is described in Section 3.6. For cut-cell boundary flux integration, the state is extrapolated with the reconstructed gradients from the cell centroids to the boundary face,

$$q_{bf} = q_0 + \nabla q\ (x_{bf} - x_0). \tag{3.14}$$

The boundary of the cut cell from Fig. 3-1(a) is shown in Fig. 3-2 to illustrate the reconstruction of the boundary state q_{bf} for cut-cell boundary flux integration. The q_{bf} is extrapolated to each cut surface piece for integration.

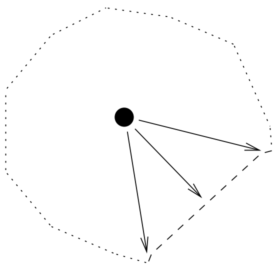


Figure 3-2: Cut-cell boundary integration, in 2D.

3.4 Gradient Reconstruction

Finite volume schemes store cell-averaged data as solution unknowns. Gradients are reconstructed from neighboring cell-averaged data to create a more accurate scheme. Barth[20] introduced a fitting procedure to reconstruct gradients,

$$\begin{bmatrix} w_1(x_1-x_0) & w_1(y_1-y_0) & w_1(z_1-z_0) \\ \vdots & \vdots & \vdots \\ w_n(x_n-x_0) & w_n(y_n-y_0) & w_n(z_n-z_0) \end{bmatrix} \begin{pmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{pmatrix} = \begin{pmatrix} w_1(\alpha_1-\alpha_0) \\ \vdots \\ w_n(\alpha_n-\alpha_0) \end{pmatrix}, \tag{3.15}$$

for each primitive state $\alpha \in q = [\rho, u, v, w, p]^T$. For uncut cells, the cell neighbors $1, \ldots, n$ that surround the central cell 0 are often more numerous than the 3 unknowns, so the overdetermined system is solved with the method of least squares. For cut cells, however, the number of neighbors can be low, resulting in poor conditioning. To improve conditioning for cut cells, the gradient reconstruction system is extended to include the neighbors of cut-cell neighbors. To reduce numerical instabilities, a Gram-Schmidt QR factorization[11] is used to invert Eq. (3.15) by precomputing and storing R, which is only a function of the problem geometry. Mavriplis[101] discusses the properties of unweighted and various weighted reconstruction schemes.

The discrete adjoint solution exhibits extreme values in small cells that are adjacent to much larger control volumes when these small cells are included in the unweighted reconstruction scheme of the larger cells. This behavior is problematic for error estimation because of the use of a high-order recovery (as described in Chapter 4). This issue is examined for scalar convection in Appendix A. The source of these large adjoint values in small cells was tracked to the gradient reconstruction system. For the unweighted reconstruction, the reconstructed gradient is highly sensitive to the solution in small cells, causing large contributions to the adjoint residual in these small cells.

To relieve this problem, the gradient reconstruction system includes a square root of volume weighting $w_i = \sqrt{V_i/V_0}$. The reconstructed gradients are still exact for linear functions with this weight. This weighting has the added benefit that a cell is smoothly included or removed from the reconstruction stencil when a control volume is infinitesimally cut. This smooth transition should aid the computation of design sensitivities.

3.5 Reconstruction Limiting

Barth and Jespersen[22] introduced limits on an unstructured grid reconstruction scheme to maintain monotonicity. Face reconstruction using a limiter of this form is

$$q_f = q_0 + \Phi \, \nabla q \, (x_f - x_0), \tag{3.16}$$

where the diagonal matrix limiting function Φ is computed in each control volume. The same Φ is employed in all face reconstructions for a given control volume. This type of limiter can compromise the convergence of the flow and therefore a dual-consistent adjoint solver.[52,116] Venkatakrishnan[148] studied this limiter in its original form as well as with the limiter function held constant after iterative convergence stalls. He proposed a new limiter to improve convergence, but both the frozen scheme and new limiter can result in stalled convergence. The Venkatakrishnan limiter is not monotone, it permits under- and over-shoots. Frozen limiters are derivative approximations that impede error estimation, output-based adaptation, adjoint iterative convergence, and design sensitivities.[16,87,113]

Balasubramanian and Newman[16] propose applying the Barth-Jespersen and Venkatakrishnan limiters on an edge-by-edge basis instead of having a single value of Φ for each control volume. They reported an improvement in the iterative convergence of the modified limiters for both the flow and adjoint systems of a wing in transonic inviscid flow. Berger, Aftosmis, and Murman[23] examine edge-based limiting in an appendix. They show that edge-based limiting can introduce new extrema into the solution.

In this study, the limiter will be used in the context of an output adaptive scheme that requires the adjoint solution. An exact linearization and steady iterative convergence of the flow and adjoint solvers is paramount to the robustness of the adaptive scheme. This iterative convergence is so critical that the accuracy of the limited scheme will be sacrificed; accuracy will be regained with adaptive grid refinement and alignment. A heuristic edge-based limiter[152] is utilized to improve the convergence of the flow solver while providing the exact linearization required for adjoint convergence. Concessions are made to improve iterative convergence; it is not total variation diminishing (TVD) or linearity preserving.

The heuristic limiter was developed[152] by examining its effect on shock capturing for regular and irregular grids and empirically adjusting its formulation to increase the width of shocks. It is a scalar limiting function ϕ that considers only the cell-averaged values of pressure and their reconstructed gradients in the cells adjacent to the face being reconstructed. Face reconstruction using a limiter of this form is

$$q_f = q + \phi \nabla q (x_f - x_0), \tag{3.17}$$

where the scalar limiting function ϕ is computed for each face f. The same ϕ is used for the left $q_{l\,f}$ and right $q_{r\,f}$ face reconstructions.

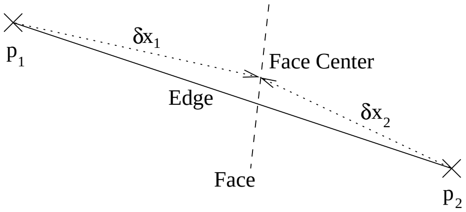


Figure 3-3: Edge and face geometry.

The basic concept employed in this heuristic limiter is to reduce the reconstruction gradient in locations where the pressure gradients are large relative to pressure. This clearly could result in limiting in regions for which the solution varies linearly (though with large magnitude), however, in combination with adaptation the proposed limiter has been found robust and accurate (see Section 6.5). The specific form of the limiter relies on a measure of the change in the pressure, δp. To form δp, the reconstructed gradient of pressure for the control volumes on the right and left of the face (Fig. 3-3), ∇p_1 and ∇p_2, are used with the right and left extrapolation vectors to the face, δx_1 and δx_2,

$$\delta p = \left\| \begin{array}{c} \delta x_{1_x} \nabla p_{1_x} - \delta x_{2_x} \nabla p_{2_x} \\ \delta x_{1_y} \nabla p_{1_y} - \delta x_{2_y} \nabla p_{2_y} \\ \delta x_{1_z} \nabla p_{1_z} - \delta x_{2_z} \nabla p_{2_z} \end{array} \right\| . \tag{3.18}$$

This sensor is active for linear functions and does not specifically penalize extrema.

The gradient reconstruction is reduced where the the δp sensor is large with the intention of spreading the detected jump over a number of control volumes. Adaptation will be employed to narrow the width of the discontinuity. The tanh function is employed to smooth the combined nondimensional pressure jump ratio,

$$\phi_{\text{heuristic}} = 1 - \tanh\left(\frac{\delta p}{\min(p_1, p_2)}\right), \tag{3.19}$$

and restricts the limiter to the range $(0, 1]$. A tanh function is employed to provide a smoothly varying and differentiable function that enables residual convergence that can be impeded by a non-smooth limiting function. This limiter is active (to some degree) in all regions with pressure variations, so it will not switch on and off intermittently during iterative convergence. The design accuracy of the limited scheme is therefore below second-order. The limiter is more active when the pressure variation is significant as compared to the local pressure.

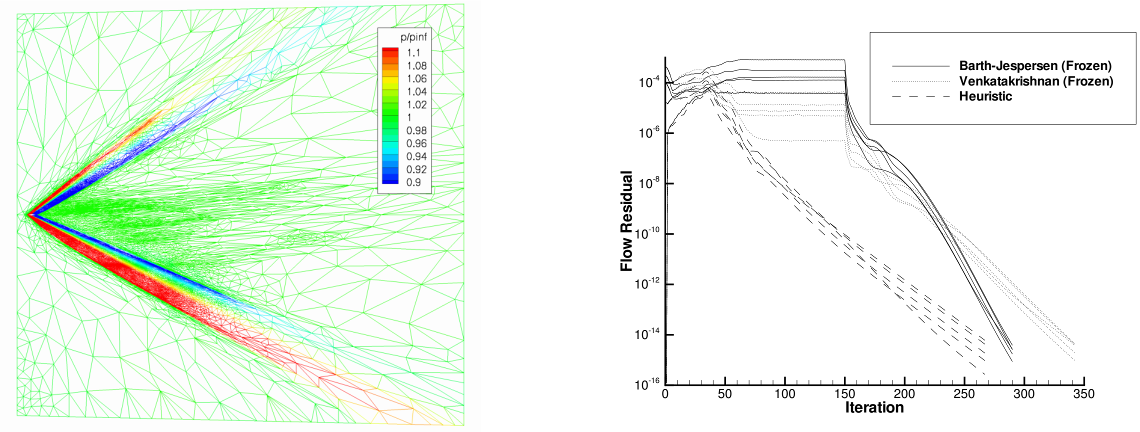
The cut cells require pressure, extrapolated to the boundaries, to compute boundary fluxes. This reconstruction requires limits to prevent unrealizable face states and must be smoothly differentiable to facilitate iterative convergence,

$$\delta p_d = \left\| \begin{array}{c} \delta x_x \nabla p_x \\ \delta x_y \nabla p_y \\ \delta x_z \nabla p_z \end{array} \right\|, \tag{3.20}$$

$$\phi_{\text{extrapolation}} = 1 - \tanh\left(\frac{\delta p}{p}\right). \tag{3.21}$$

The extrapolation limiter is formulated to mimic the interior face limiter using only the data from the cell adjacent to the boundary. These limiters reduce, but do not eliminate the incidence of unrealizable face reconstructions, as discussed in Sections 3.7 and 3.8.

A diamond airfoil in Mach 2.0 flow at 5 deg angle of attack is provided to illustrate the convergence issues of reconstruction limiters, see Fig. 3-4. The grid is anisotropically adapted to resolve the shocks. A symmetry plane grid of the 3D extruded airfoil domain is colored with pressure in Fig. 3-4(a). Figure 3-4(b) shows the convergence history of the 2-norms of the 5 conservation equations for each of the three limiters. The convergence of the Barth-Jespersen and Venkatakrishnan limiters initially stall. They both converge after Φ is frozen at iteration 150. The heuristic limiter converges without modification.

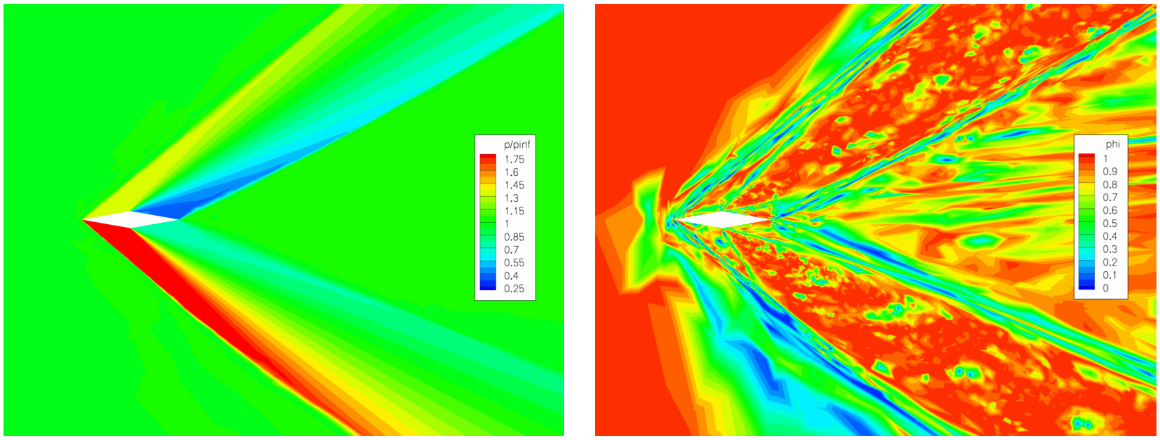


(a) Pressure colored symmetry plane grid.

(b) Convergence history (B-J and V limiters frozen at iteration 150).

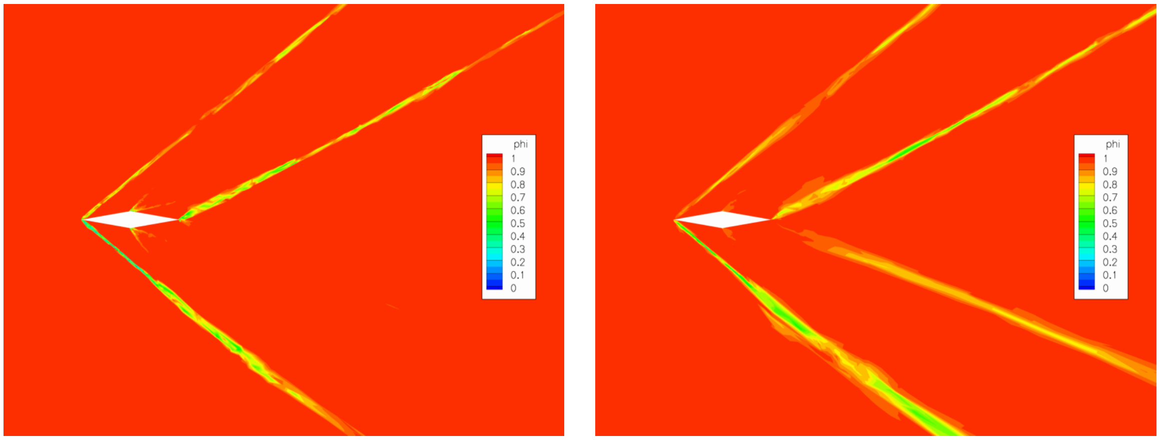
Figure 3-4: Diamond airfoil pressure colored symmetry plane grid and convergence history.

A plane in the center of the 3D domain of the extruded airfoil is shown in Fig. 3-5. A close-up of pressure around the diamond airfoil is shown in Fig. 3-5(a). The minimum Φ or ϕ involved in the face reconstruction for a control volume is shown in Fig. 3-5(b), Fig. 3-5(c), and Fig. 3-5(d). The Barth-Jespersen limiter, Fig. 3-5(b), is active over large portions of the domain, including regions with small variations. This Φ function has a large amount of high frequency variation. The Venkatakrishnan limiter, Fig. 3-5(b), is the least active and its activity is narrowly restricted to the shock and strongest expansion regions. The heuristic limiter, Fig. 3-5(d), is more active than the Venkatakrishnan limiter and includes more of the expansion region. The ϕ scalar is active in regions with moderate to large pressure variation. It has a wider active region than the Venkatakrishnan limiter and a smoother variation than the Barth-Jespersen limiter. The effect of limiter function on propagated pressure signatures for sonic boom prediction is examined in Section 6.5.



(a) Pressure.

(b) Barth-Jespersen min(Φ).



(c) Venkatakrishnan min(Φ).

(d) Heuristic min(ϕ).

Figure 3-5: Diamond airfoil pressure and limiter function.

3.6 Boundary Conditions

The boundary conditions are imposed weakly through the fluxes. The tangential flow boundary condition is implemented with zero velocity normal to the boundary, resulting in the flux,

$$F_{tangential} = \begin{bmatrix} 0 \\ pn_x \\ pn_y \\ pn_z \\ 0 \end{bmatrix}, \tag{3.22}$$

where p is interpolated along or extrapolated to the boundary. The supersonic outflow boundary condition uses the interior state to form the boundary flux. The supersonic inflow boundary condition uses the free stream state ρ_∞, u_∞, v_∞, w_∞, and p_∞ to form the boundary flux.

The inviscid flow model breaks down at a sharp corner where separation would occur in a physical flow. In the supersonic flow simulations performed in this work, this problem presents itself at blunt trailing edges. To avoid this problem in these regions, a transpiration boundary condition is specified manually. This boundary condition applies free stream velocity state, u_∞, v_∞, and w_∞ with a density and pressure of $\rho = 0.3\rho_\infty$ and $p = 0.3p_\infty$. This level of density and pressure is empirically established by examining the solution of a backward facing step with the tangential boundary condition.

3.7 Realizability

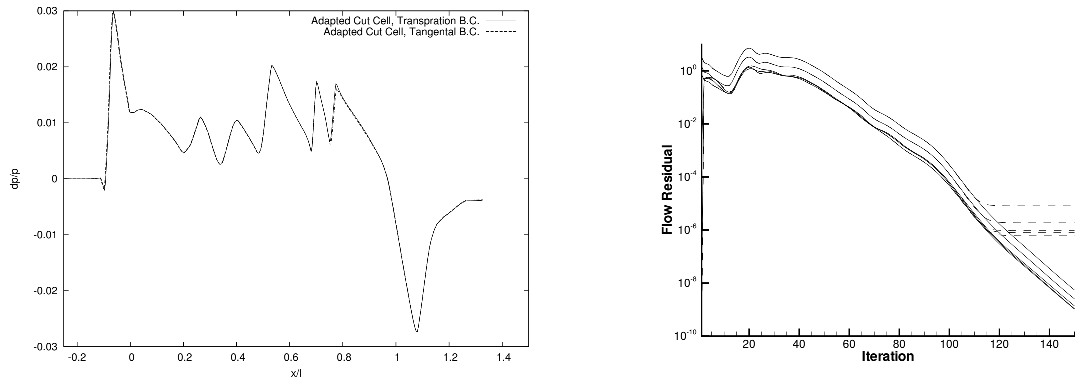
Even with the use of reconstruction limiters, it is still possible to reconstruct states with negative ρ or p. These unrealizable states cause catastrophic problems for flux calculations. To allow the calculation to proceed, unrealizable reconstructed face states are set to the cell-averaged value, locally reducing the scheme to first order. These unrealizable states are most common during start up from free stream conditions. As the simulation continues, the incidence of this clipping is reduced and often eliminated.

The cell-averaged state must also be prevented from reaching unrealizable values. Inviscid supersonic flow expands around corners to reach extremely low ρ and p. The updated values of ρ and p are artificially floored at 1% of free stream values. Flooring the ρ and p effectively changes the iterative time advancement scheme. This modification to the flow solver time advancement scheme can disrupt the iterative convergence of the flow and adjoint solver.

3.8 Tangential and Transpiration Boundary Condition Comparison

A low-boom wing body with a finite thickness trailing edge is simulated with transpiration and tangential boundary conditions to study their effect on iterative convergence and propagated pressure signature. The full description of this case is provided in Section 6.5.3. The case is adapted to improve the calculation pressure one body length below the configuration using the transpiration boundary condition. Both boundary conditions are then applied to the final adapted grid. The pressure signature, one body length below the configuration, is shown in Fig. 3-6(a) for the transpiration and tangential boundary conditions applied to the finite thickness trailing edge. The difference in pressure signature, due to the different trailing edge boundary conditions, is insignificant for this wing body configuration.

The flow expands around the backward facing step that is a result of finite thickness trailing edge. The iterative convergence stalls for the tangential boundary condition, Fig. 3-6(b), because the pressure and density floors are active to prevent the cell-averaged states from becoming negative. The ρ and p reach low values and are floored at 1% of free stream values. The stalled flow solver convergence for the tangential boundary condition may cause an instability for the dual consistent adjoint solution scheme.



(a) Pressure signature at 1 body length.

(b) Convergence history.

Figure 3-6: Results of a low-boom wing body at one body length with the transpiration and tangency boundary condition on finite thickness trailing edge.

3.9 Body-Fitted and Cut-Cell Supersonic Vortex Uniform Refinement

A supersonic vortex has been used by a number of researchers to verify the accuracy of schemes and error estimation techniques.[2,4,7,68,73,112] The 2D geometry of the problem is given in Fig. 3-7. The domain is a section of an annulus. This domain is extruded to for the 3D domain utilized for this study. The solution only varies in the radial direction and is given by,

$$\rho = \rho_i \left\{ 1 + \frac{\gamma+1}{2} M_i^2 \left[1 - \left(\frac{r_i}{r} \right)^2 \right] \right\}^{\frac{1}{\gamma-1}}, \tag{3.23}$$

$$p = p_i \left\{ 1 + \frac{\gamma+1}{2} M_i^2 \left[1 - \left(\frac{r_i}{r} \right)^2 \right] \right\}^{\frac{\gamma}{\gamma-1}}, \tag{3.24}$$

$$a = \sqrt{\gamma p / \rho}, \tag{3.25}$$

$$M = \frac{a_i M_i r_i}{a r}, \tag{3.26}$$

where M is Mach number and a is the speed of sound. There is an error in the equations provided in Ref. 73. The flow conditions at the inside radius of the vortex domain $r_i = 1$ are $\rho_i = 1$, $p_i = 1/\gamma$, $M_i = 2.25$. The outer radius for the vortex domain is $r_o = 1.384$. The vortex is modeled for a 90 deg turning angle.

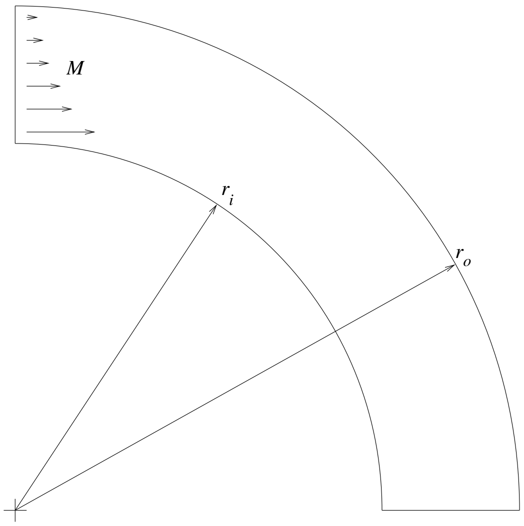


Figure 3-7: Supersonic vortex geometry.

A series of uniformly refined grids are employed to verify the design order accuracy of the existing body-fitted and current cut-cell approaches. The coarsest body-fitted grid is shown in Fig. 3-8(a). It is constructed of nearly-right tetrahedra. The coarsest background grid used for the cut-cell approach is shown in Fig. 3-8(b). It is constructed of right tetrahedra. The cut surface has 400 triangulated linear segments. This fine resolution may be excessive, but it eliminates the geometry error of the cut surface as a source of error for this uniform refinement study.

The coefficient of lift is computed on the inner and outer curved surfaces of the vortex domain. The exact pressure is constant along the curved surfaces,

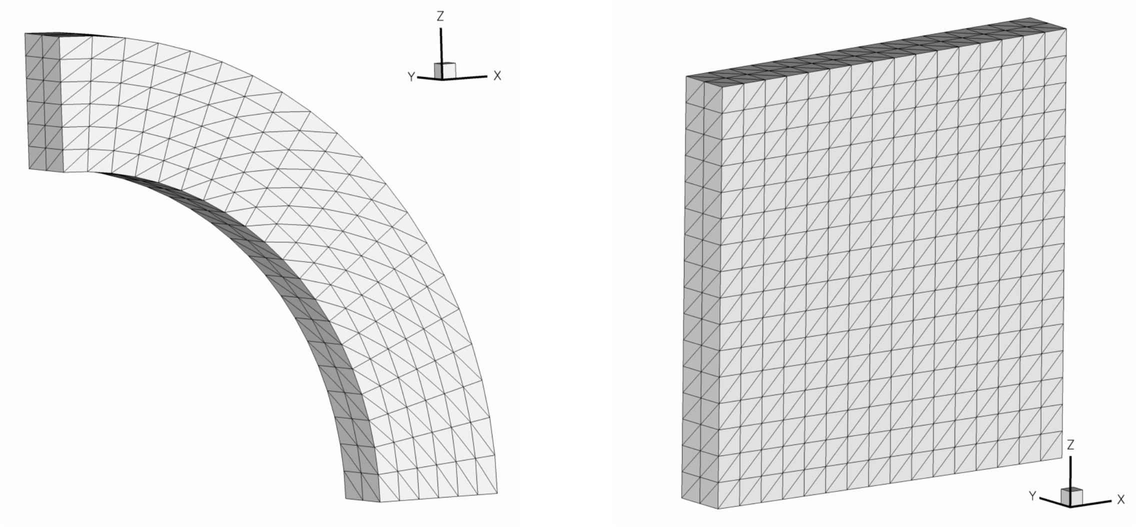
$$p_i/p_i = 1, \tag{3.27}$$

$$p_o/p_i = \left\{1 + \frac{\gamma+1}{2} M_i^2 \left[1 - \left(\frac{r_i}{r_o}\right)^2\right]\right\}^{\frac{\gamma}{\gamma-1}} = 3.98035302289919. \tag{3.28}$$

The coefficient of pressure is,

$$C_{p_i} = 0, \tag{3.29}$$

$$C_{p_o} = 2\frac{p_o/p_i - 1}{\gamma M_i^2} = 0.841016726038573. \tag{3.30}$$



(a) Uniformly refined body-fitted grid (441 control volumes).

(b) Uniformly refined cut-cell background grid (867 control volumes uncut and 409 control volumes after cut).

Figure 3-8: Coarsest uniformly refined body-fitted and cut-cell grids.

The coefficient of lift for the vortex domain per unit span is

$$C_l = r_o C_{p_o} = 1.16396714883738. \tag{3.31}$$

The error in computed lift divided by the exact lift is shown in Fig. 3-9 for a series of uniformly refined body-fitted grids and cut-cell background grids with both the heuristically limited and unlimited reconstruction schemes. The inflow conditions are specified as the analytic solution. The outflow conditions are taken from the interior of the domain. The tangency boundary conditions are applied to the inner and outer boundaries as well as the two sides. The body-fitted and cut-cell methods employing the heuristic limiter have a similar lift error level. This error level is higher than the unlimited reconstruction schemes. The error introduced by the heuristic limiter does not appear to be exacerbated by the irregular shape of the cut cells. Both unlimited reconstruction methods asymptote to second order as indicated by the triangle with a slope of two. The unlimited cut-cell method has more error than the unlimited body-fitted method for the same characteristic length (degrees of freedom). The higher error level of the cut-cell method is offset by an increase in the adaptive mechanics robustness, which can produce more efficient grids.

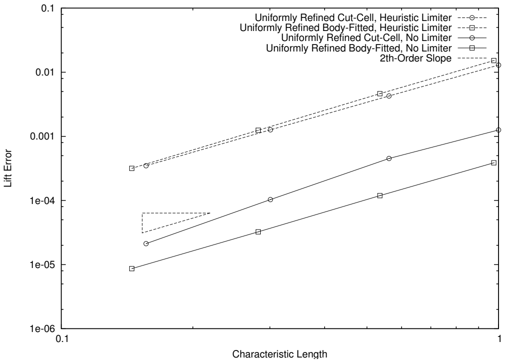


Figure 3-9: Lift error convergence for uniformly refined body-fitted and cut-cell grids.

3.10 Body-Fitted and Cut-Cell Wing Uniform Refinement

An ONERA M6 wing is simulated at 0.84 Mach and 3 deg angle of attack. A series of body-fitted and cut-cell grids are used to examine the the grid convergence properties of both schemes. For these transonic cases, the face reconstructions are not limited. Figure 3-10 shows Mach number contours on a 1,160,000 control volume cut-cell grid. The λ-shock structure, expected at these flow conditions, is clearly visible on the upper surface of the wing.

A series of cut-cell background grids with 10,000 to 1,160,000 control volumes are shown with triangular symbols in Fig. 3-11. The surface geometry is the same for all of the cut cell calculations. There are four body fitted grids sized 24,600 to 564,00 control volumes that employ the same background spacing as the cut cell grids shown as squares in Fig. 3-11. A series of body fitted grid calculations from Park[123] are also shown as circles in Fig. 3-11.

The coefficient of drag computed with these grids are plotted as a function of characteristic length squared h^2 in Fig. 3-11(a). The characteristic length h is estimated as the inverse of the cube root of the number of control volumes. The coefficient

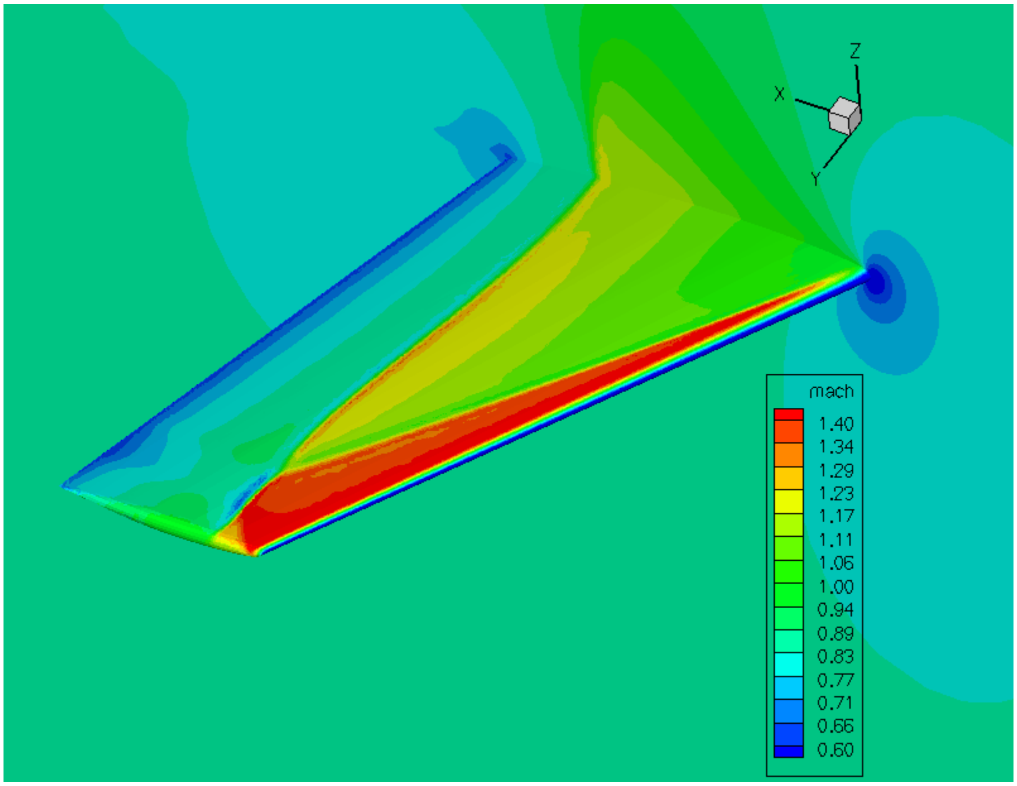
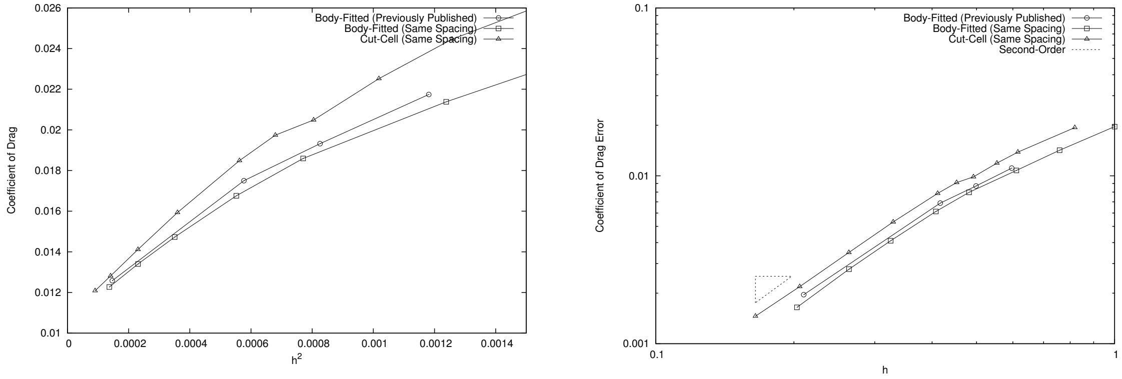


Figure 3-10: Mach number contours for 1.16 million node background mesh cut-cell ONERA M6.

of drag values are all converging with uniform grid refinement to the same value, Fig. 3-11(a). The two body-fitted grid sequences have similar values for coefficient of drag. For the same mesh size, the cut-cell results have a higher (less accurate) drag. Asymptotically, as h approaches zero, the various results are agreeing on the coefficient of drag. The exact value for coefficient of drag is estimated with Richardson extrapolation of the finest two body fitted grids assuming 2nd-order spatial accuracy, coefficient of drag $= 0.010624$. Figure 3-11(b) shows the coefficient of drag error as a function of the characteristic length. The coefficient of drag error shows smooth second-order convergence, as indicated by the triangle with a second-order slope. The cut-cell method requires more control volumes (a smaller characteristic length) for the same accuracy. This loss in efficiency is offset by an increase in the robustness of grid generation and adaptation mechanics of cut-cell background grids.



(a) Coefficient of drag.

(b) Coefficient of drag error.

Figure 3-11: ONERA M6 coefficient of drag and coefficient of drag error for cut-cell and body-fitted grids as a function of characteristic element size.

Chapter 4

Output-Based Adaptation

Venditti and Darmofal[146] describe an output-based error estimation and adaptation scheme. To formulate the error estimate, an embedded grid is required. Constructing the entire embedded grid can be infeasible for large 3D grids and has prevented the use of adjoint error estimation techniques for industrial-sized problems even with a parallel implementation.[87] While the embedded grid can be formed in sections, this increases the error estimation scheme complexity. Forming a portion or the entire embedded grid is also complicated by the need to respect curved boundaries and recompute the intersection tests of cut cells. These difficulties have motivated the desire to only employ the current grid in the error estimation procedure. A procedure is described that obtains a less expensive indicator for output adaptation with the current grid, but does not provide a functional error correction available with the embedded grid-approach.

4.1 Embedded-Grid Error Estimate

The single-grid adaptive indicator is derived following presentation of the embedded-grid error estimate provided by Venditti.[145] Let δQ_h represent a perturbation of a fine grid primal solution Q_h. This allows the introduction of an approximate fine grid solution,

$$\tilde{Q}_h \equiv Q_h + \delta Q_h. \tag{4.1}$$

In practice Q_h and δQ_h are unknown, but $\tilde{Q}_h$ is available. This perturbation can be expressed in terms of a functional,

$$\delta f_h \equiv f_h(\tilde{Q}_h) - f_h(Q_h). \tag{4.2}$$

The goal of this procedure is to compute an estimate of $f_h(Q_h)$ without computing Q_h. The perturbation in the residual is

$$\delta R_h \equiv R_h(\tilde{Q}_h) - R_h(Q_h), \tag{4.3}$$

$$\delta R_h = R_h(\tilde{Q}_h), \tag{4.4}$$

because the fine grid solution satisfies the residual, $R_h(Q_h) = 0$. A Taylor series expansion about the solution yields,

$$\delta f_h \approx \frac{\partial f_h}{\partial Q_h} \delta Q_h + \cdots , \tag{4.5}$$

$$\delta R_h \approx \frac{\partial R_h}{\partial Q_h} \delta Q_h + \cdots . \tag{4.6}$$

The discrete adjoint system on the fine grid is

$$\frac{\partial R_h}{\partial Q_h}^T \lambda_h = \frac{\partial f_h}{\partial Q_h}^T . \tag{4.7}$$

The Taylor expansion is modified by substitution,

$$\begin{aligned} \delta f_h &\approx \frac{\partial f_h}{\partial Q_h} \delta Q_h && (4.8) \\ &= \lambda_h^T \frac{\partial R_h}{\partial Q_h} \delta Q_h && (4.9) \\ &\approx \lambda_h^T R_h(\tilde{Q}_h). && (4.10) \end{aligned}$$

To avoid the expense of computing the adjoint on the fine grid, an adjoint perturbation is introduced,

$$\tilde{\lambda}_h \equiv \lambda_h + \delta \lambda_h. \tag{4.11}$$

The adjoint residual operator is

$$R_h^\lambda(\cdot) \equiv \frac{\partial R_h}{\partial Q_h}^T(\cdot) - \frac{\partial f_h}{\partial Q_h}^T. \tag{4.12}$$

Substituting Eq. (4.11) into the adjoint residual yields,

$$\begin{aligned} R_h^\lambda(\tilde{\lambda}_h) &= \frac{\partial R_h}{\partial Q_h}^T(\lambda_h + \delta\lambda_h) - \frac{\partial f_h}{\partial Q_h}^T & (4.13) \\ &= \frac{\partial R_h}{\partial Q_h}^T(\delta\lambda_h), & (4.14) \end{aligned}$$

because $R_h^\lambda(\lambda_h) = 0$. Now the fine grid adjoint solution dependency can be eliminated,

$$\begin{aligned} \delta f_h &\approx \lambda_h^T R_h(\tilde{Q}_h) & (4.15) \\ &= \tilde{\lambda}_h^T R_h(\tilde{Q}_h) - \delta\lambda_h^T R_h(\tilde{Q}_h). & (4.16) \end{aligned}$$

This term can also be expressed with the solution perturbation,

$$\begin{aligned} \delta f_h &\approx \tilde{\lambda}_h^T R_h(\tilde{Q}_h) - \delta\lambda_h^T \frac{\partial R_h}{\partial Q_h}\delta Q_h & (4.17) \\ &= \tilde{\lambda}_h^T R_h(\tilde{Q}_h) - R_h^\lambda(\tilde{\lambda}_h)^T \delta Q_h. & (4.18) \end{aligned}$$

The leading term in these equations is used as a computable error correction,

$$f_h(Q_h) \approx f_h(\tilde{Q}_h) - \tilde{\lambda}_h^T R_h(\tilde{Q}_h). \tag{4.19}$$

At this point a functional error correction for an approximate fine-grid solution and adjoint is computable, Eq. (4.19). To form the approximate fine-grid solutions, the solutions on an affordable grid, Q_H and λ_H, are combined with a quadratic prolongation operator P_h^H to compute the fine-grid correction with a coarse-grid solution,

$$f_h(Q_h) \approx f_h(P_h^H Q_H) - P_h^H \lambda_H^T R_h(P_h^H Q_H). \tag{4.20}$$

The remaining error terms $-\delta\lambda_h^T R_h(\tilde{Q}_h)$ and $-R_h^\lambda(\tilde{\lambda}_h)^T \delta Q_h$ are used to form the adaptation indicator. The terms $\delta\lambda_h$ and δQ_h are unknown, but they are estimated as the difference between the linear prolongated solution $L_h^H Q_H$ and the quadratic prolongated solution $P_h^H Q_H$ on the fine grid,

$$[I_{\text{embed}}]_{\kappa_H} = \frac{1}{2} \sum_{\kappa_h \in \kappa_H} \sum_{i=1}^{5} \left\{ \begin{array}{l} \left|[R_h^\lambda(L_h^H \lambda_H)]_{i,\kappa_h} [P_h^H Q_H - L_h^H Q_H]_{i,\kappa_h}\right| + \\ \left|[P_h^H \lambda_H - L_h^H \lambda_H]_{i,\kappa_h} [R_h(L_h^H Q_H)]_{i,\kappa_h}\right| \end{array} \right\}, \tag{4.21}$$

where the summation is over the embedded fine grid control volumes that each coarse grid control volume contains. The five conservation equations are contracted by the index i. The vector $I_{\text{embed}} \in \mathcal{R}^N$ is the fine grid error restricted to a single value for each coarse grid control volume.

4.2 Single-Grid Error Estimate

The single-grid error estimation and adaptive indicator is

$$[I_{\text{single}}]_\kappa = \frac{1}{2} \sum_{i=1}^{5} \left\{ \left|[R^\lambda(\hat{\lambda})]_{i,\kappa} [\hat{Q} - \bar{Q}]_{i,\kappa}\right| + \left|[\hat{\lambda} - \bar{\lambda}]_{i,\kappa} [R(\hat{Q})]_{i,\kappa}\right| \right\}. \tag{4.22}$$

It has the same pieces as Eq. (4.21), where the five conservation equations are contracted by the summation over i. The vector $I_{\text{single}} \in \mathcal{R}^N$ has a single value for each grid control volume κ. The $\hat{\lambda}$ and $\hat{Q}$ higher-order reconstructions and the $\bar{\lambda}$, and $\bar{Q}$ lower-order reconstructions on the current grid are described in the next section. The original residual operators are utilized and $\hat{\lambda}$ and $\hat{Q}$ are constructed to make $R^\lambda(\hat{\lambda})$ and $R(\hat{Q})$ reliable adaptive indicators. The $\hat{\lambda}$ and $\hat{Q}$ reconstructions are formed with a fit of quadratic functions to cell averaged states and their gradients. The difference between the $\hat{()}$ and $\bar{()}$ reconstructions is intended to provide adequate guidance for the relative distribution of error, not a sharp bound on error.

The corresponding term of the form,

$$f(\hat{Q}) - \hat{\lambda}^T R(\hat{Q}), \tag{4.23}$$

is a correction to the single-grid residual error of the reconstructed solution. It does not provide an improved functional estimate, which is shown in the supersonic vortex example later in this chapter.

4.3 Grid Resolution Request

Venditti[145] provides a procedure to calculate a new grid spacing request h from the adaptive indicator I_κ and an error tolerance tol_Ω. The adaptation indicator is summed to find the global indicator $I_\Omega = \sum I_\kappa$, which is also referred to as the remaining error after correction. The ratio of the remaining error to a user-specified error tolerance tol_Ω is

$$\epsilon_\Omega = \frac{I_\Omega}{\mathrm{tol}_\Omega}. \tag{4.24}$$

The ratio of the control volume indicator to an equal share of tol_Ω is

$$\epsilon_\kappa = \frac{N}{\mathrm{tol}_\Omega} I_\kappa, \tag{4.25}$$

where N is the number of control volumes. When a cost function does not have an intuitive error tolerance, i.e., sonic boom surface pressure integrals, the tol_Ω is set to half I_Ω at each adaptive iteration.

Now that the ratios of estimated to desired error levels have been established, the sizing request for an improved grid is computed. For cut-cell simulations, the background grid is modified using the error estimates from the cut grid. The requested isotropic element length h is computed with an estimate of the spacing on the original mesh h^0 and the global and local error ratios,

$$h_\kappa = h_\kappa^0 \left(\frac{1}{\epsilon_\Omega \epsilon_\kappa} \right)^\omega . \tag{4.26}$$

The original spacing h_κ^0 is computed with an implied metric[84] on the uncut background grid. A portion of the nodes of the background grid are outside of the computational domain after the cut-cell procedure. For the nodes in the background grid that are external to the problem domain, the implied h_κ^0 is doubled, $h_\kappa = 2h_\kappa^0$, to reduce the total size of background grid where it does not contribute to resolving the problem of interest. The exponent value of $\omega = 1/4$ is based on the assumed convergence rates of $\epsilon_\Omega \epsilon_\kappa$ for smooth problems.[145] Fidkowski and Darmofal[54] utilize a locally varying ω based on *a priori* estimates of the spatial error convergence rate. Nemec, Aftosmis, and Wintzer[113] also suggest different strategies for converting the ϵ_κ into an adaptation request in the context of Cartesian grids.

4.4 Error Estimation Reconstruction Operators

The cut-cell grid does not have underlying tetrahedral elements to support an element based reconstruction, currently used for body-fitted grids. It is possible to create local bounding elements and use these to reconstruct a smooth solution. However, the approach taken here is edge-based reconstruction. The $\hat{\lambda}$ and $\hat{Q}$ reconstructions utilize a quadratic reconstruction on each edge. The $\bar{\lambda}$ and $\bar{Q}$ reconstructions utilize a constant reconstruction on each edge.

The first step of the edge-based quadratic reconstruction is to calculate least-squares gradients at each control volume, Section 3.4. Each edge now has four pieces of information, the value of the cell-averaged solution at each of its control volumes and the reconstructed gradients projected in the edge direction. A quadratic representation of the solution is created along each edge of the mesh. This quadratic function has three unknowns per edge, which will be fit at edge endpoints to the solution values and their gradients. The reconstructed solution at a control volume is formed by averaging the incident quadratic edge functions.

A quadratic function,

$$y = ax^2 + bx + c, \tag{4.27}$$

defined along a parametrized edge, is evaluated and differentiated at edge endpoints

0 and 1, yielding:

$$y(0) = c, \tag{4.28}$$

$$y(1) = a + b + c, \tag{4.29}$$

$$y'(0) = b, \text{ and} \tag{4.30}$$

$$y'(1) = 2a + b. \tag{4.31}$$

These expressions are formed into a matrix,

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \\ 2 & 1 & 0 \end{bmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} Q_0 \\ Q_1 \\ Q'_0 \\ Q'_1 \end{pmatrix}, \tag{4.32}$$

where Q_0 and Q_1 are cell averaged values and Q'_0 and Q'_1 are the reconstructed gradients projected in the edge direction to make this a 1D reconstruction along an edge. The overdetermined system is inverted with QR factorization,

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{bmatrix} 0.0 & 0.0 & -0.5 & 0.5 \\ -0.2 & 0.2 & 0.9 & -0.1 \\ 0.6 & 0.4 & -0.2 & -0.2 \end{bmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q'_0 \\ Q'_1 \end{pmatrix}. \tag{4.33}$$

The quadratic fit $\hat{y}$ is evaluated at the end points,

$$\begin{pmatrix} \hat{y}(0) \\ \hat{y}(1) \end{pmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{bmatrix} 0.6 & 0.4 & -0.2 & -0.2 \\ 0.4 & 0.6 & 0.2 & 0.2 \end{bmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q'_0 \\ Q'_1 \end{pmatrix}. \tag{4.34}$$

The reconstructed value at the nodes $\hat{q}$ is the average of all incident edge i reconstructions $\hat{y}_i$,

$$\hat{Q} = \frac{1}{n}\sum_{i=1}^{n} \hat{y}_i \tag{4.35}$$

For $\bar{Q}$ and $\bar{\lambda}$ reconstruction, the solution is assumed to be constant over each edge (the average of the edge midpoints).

$$\begin{pmatrix} \bar{y}(0) \\ \bar{y}(1) \end{pmatrix} = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \begin{pmatrix} Q_0 \\ Q_1 \end{pmatrix}. \tag{4.36}$$

The reconstructed value at the nodes $\hat{Q}$ is the average of all incident edge i reconstructions $\hat{y}_i$,

$$\bar{Q} = \frac{1}{n}\sum_{i=1}^{n} \bar{y}_i. \tag{4.37}$$

4.4.1 Accuracy of Reconstruction Operators for Supersonic Vortex

The supersonic vortex problem from Section 3.9 is utilized to show the accuracy of the reconstruction operator used in the single-grid error estimation method. An analytic function for ρ_{exact} is available in Eq. (3.23). This analytic function is used to compute the 2-norm of the error in density of a series of uniformly refined background grids, Fig. 3-8(b), using the cut-cell method. The error in computed ρ is compared to the error in the $\hat{Q}$ higher-order reconstruction for $\hat{\rho}$. This error improvement is given by,

$$\frac{||\hat{\rho} - \rho_{\text{exact}}||_2}{||\rho - \rho_{\text{exact}}||_2}, \tag{4.38}$$

and plotted for a series of uniformly refined background grids in Fig. 4-1. The relative error is below unity for all grids indicating that the reconstruction yields a slight reduction in density error.

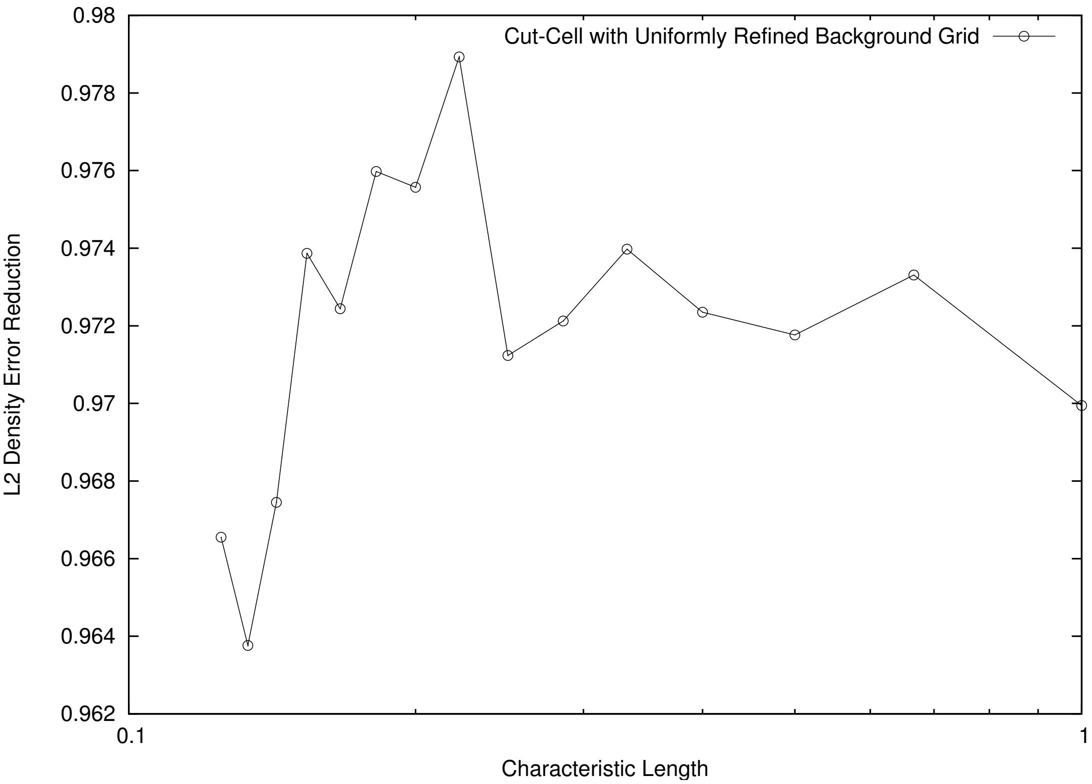
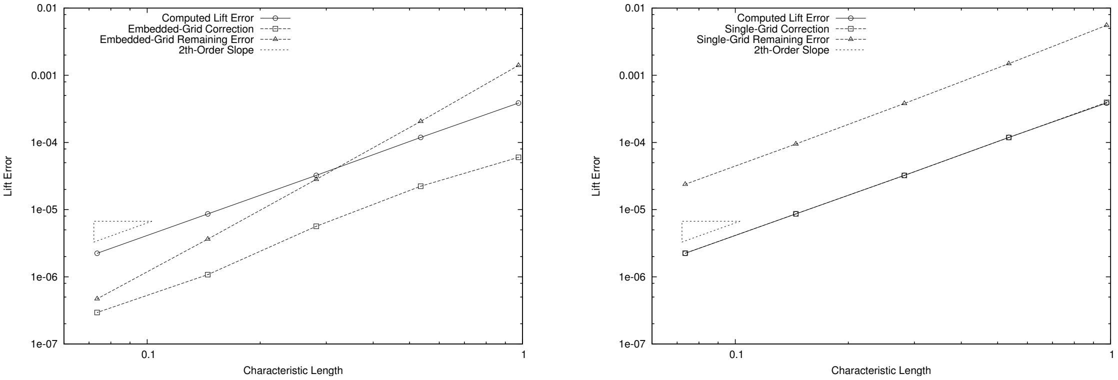


Figure 4-1: Improvement in reconstructed density error.

4.5 Embedded- and Single-Grid Output Error Estimation for Supersonic Vortex

Output (lift) error estimation is applied to the supersonic vortex problem described in Section 3.9. Nemec and Aftosmis[112] examined the same case for a similar output. The exact computed lift error is compared to error estimates in Fig. 4-2. The embedded-grid approach is applied in Fig. 4-2(a) and the single-grid approach is applied in Fig. 4-2(b). A series of uniformly-refined body-fitted grids are used with the coarsest body-fitted grid shown in Fig. 3-8(a). The circles are the error in lift computed with the standard body-fitted FUN3D scheme. Figure 4-2(a) squares are the error in embedded-grid[123] correction Eq. (4.20) and triangles are the remaining error estimate $\sum[I_{\text{embed}}]_\kappa$. Figure 4-2(b) squares are the error in single-grid correction Eq. (4.23) and triangles are the remaining error estimate $\sum[I_{\text{single}}]_\kappa$.

The embedded-grid functional correction is converging at a rate slightly greater than second order for this smooth problem. The remaining error estimate is also converging at a higher rate. Venditti[145] has shown this superconvergent behavior for smooth problems. The single-grid error correction term over-plots the computed functional. This is expected; without the embedded grid this term is simply a residual error correction of the reconstruction on the current grid. The single-grid error estimate is larger (more conservative) than the embedded-grid error estimate. The single-grid remaining error estimate reduces at second-order.

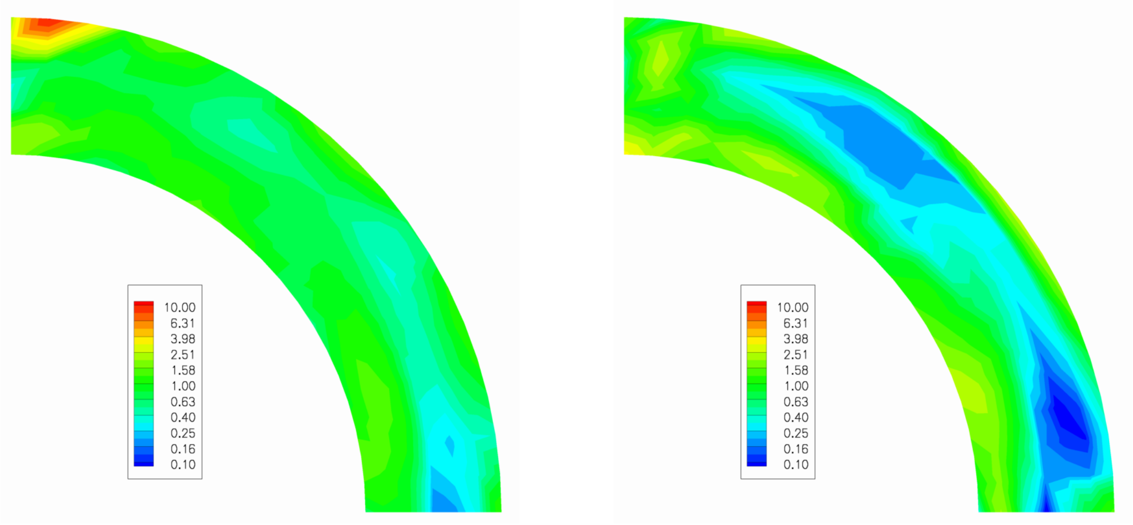


(a) Embedded-grid.

(b) Single-grid.

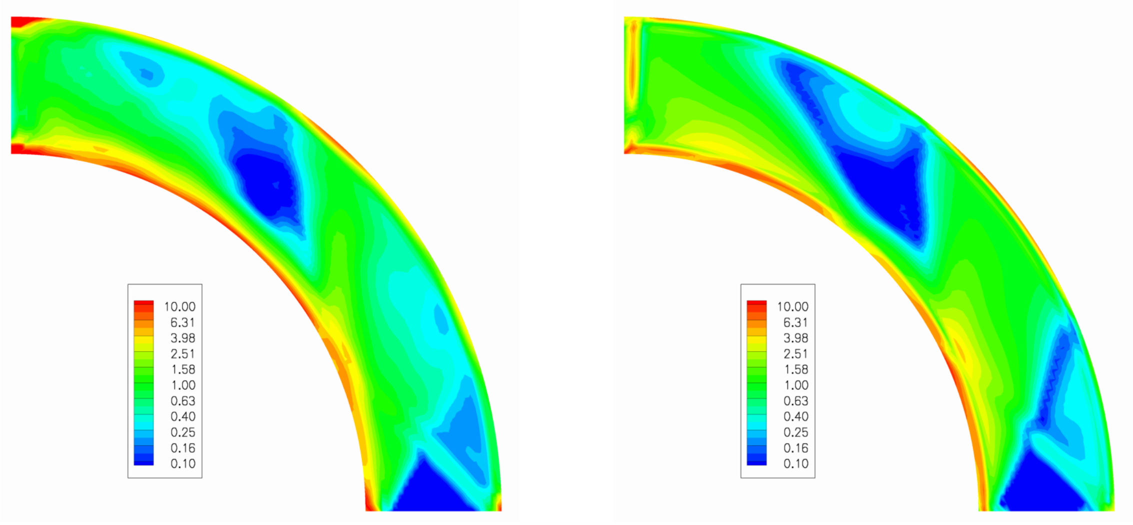
Figure 4-2: Lift, error estimation, and remaining error for uniformly refined body-fitted supersonic vortex.

It is desirable for the single-grid remaining error estimate I_{single} to have a similar distribution to the embedded-grid remaining error estimate I_{embed}. This allows the less expensive single-grid estimate to drive an adaptation scheme in a similar manner to the more expensive embedded-grid estimate. The remaining error estimates of the embedded-grid and single-grid approaches are shown in Fig. 4-3 for an initial and uniformly refined supersonic vortex grid. The remaining error estimates have been normalized by the average remaining error estimate on each grid to facilitate comparisons between grids and error estimation methods. The color scale is logarithmic and varies between one tenth to ten times the average value. The comparison between the single-grid and embedded-grid estimates is improved on the uniformly refined grid where the two methods have a similar estimate distribution. The error estimates are greatest on the curved boundaries and lowest near the outflow boundary.



(a) Initial grid embedded-grid remaining error estimate.

(b) Initial grid single-grid remaining error estimate.



(c) Uniformly refined grid embedded-grid remaining error estimate.

(d) Uniformly refined grid single-grid remaining error estimate.

Figure 4-3: Remaining error estimate distribution normalized with the average remaining error estimate for uniformly refined body-fitted supersonic vortex (logarithmic color scale).

4.6 Embedded- and Single-Grid Output Error Estimation for Wing Configuration

Output (drag) error estimation is applied to the ONERA M6 wing at Mach 0.84 and 3 deg angle of attack. This 3D embedded grid approach was previously reported by Park.[123] The sequence of 9 uniformly refined isotropic grids is shown in Fig. 4-4. The truth value for drag is 0.010624, which is obtained from Section 3.10. The circles are drag computed with the standard body-fitted FUN3D scheme. The squares are the embedded-grid[123] remaining error estimate $\sum [I_{\text{embed}}]_\kappa$. The triangles are the single-grid remaining error estimate $\sum [I_{\text{single}}]_\kappa$. The embedded-grid remaining error estimate on the finest grid required a parallel scheme[87] on 32 clustered computers, each with 4.0 Gbyte memory; it exceeds the memory on 32 typical 1.5 Gbyte clustered computers. The remaining error estimates (used to drive adaption) reach similar values on the finer grids. In Section 6.4, the single-grid adaptive indicators perform similarly to the embedded-grid approach when employed in an adaptive setting.

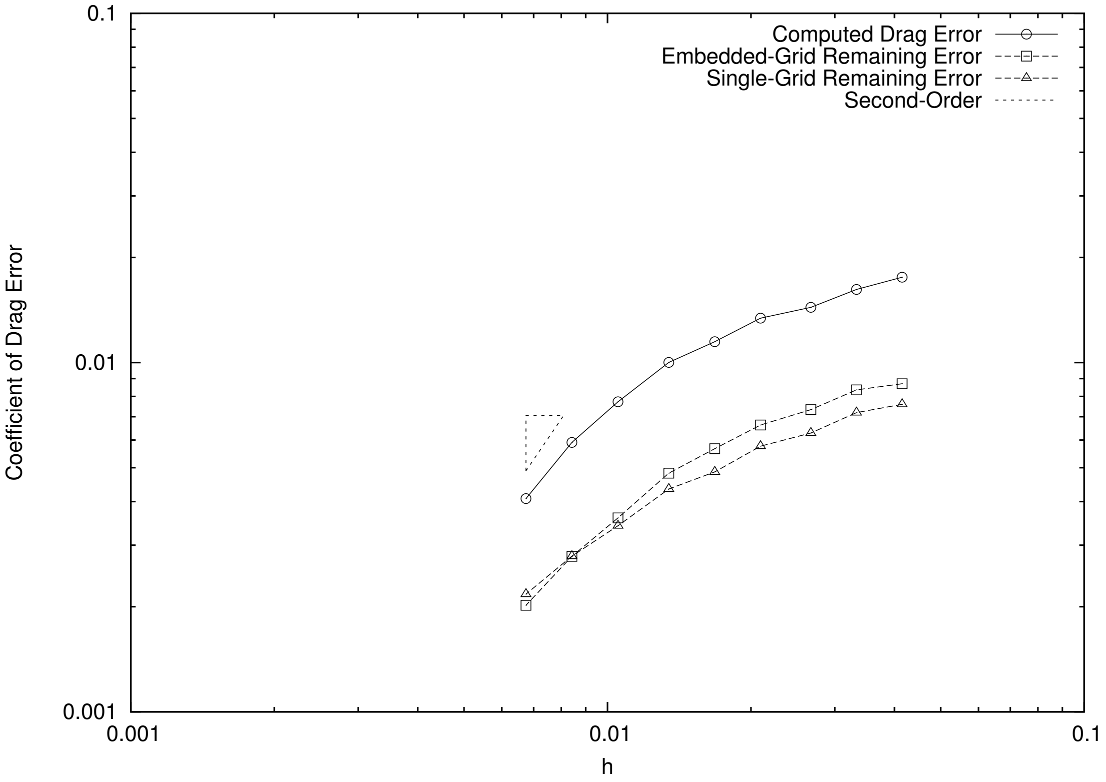


Figure 4-4: Drag, error estimation, and remaining error for uniformly refined body-fitted Mach 0.3 ONERA M6 wing.

Chapter 5

Adaptation Mechanics

The grid adaptation algorithm description is separated into high-level logic and elemental grid operators. The high-level logic is described in the following chapters for metric-based and direct grid optimization. All of these algorithms employ the same elemental grid operators, which are described in this chapter. The infrastructure to allow multiple elemental operators to execute in a parallel is also described.

5.1 Grid Operators

The grid adaptation scheme is constructed from a sequence of 3D nearest-neighbor grid modifications: node movement, tetrahedra swap, tetrahedra split, and tetrahedra collapse. Each grid in this sequence is represented as the current set of nodes $x_i \in \mathcal{X}_n$ and tetrahedra $\kappa_j \in \mathcal{T}_n$,

$$(\mathcal{X}_n, \mathcal{T}_n), \tag{5.1}$$

where x_i is the position of node i and κ_j are the nodes that element j connects. The edge segments, segments connecting element nodes, are indexed by l. Each of the grid operators G modify the current grid to create a new grid,

$$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = G(\mathcal{X}_n, \mathcal{T}_n, i, j, l, c(), \bar{c}), \tag{5.2}$$

acting on the grid constitutes by index and improving an objective function $c()$ to a tolerance $\bar{c}$. Each specific grid operator may use a subset of these arguments. The tetrahedra removed and replaced in an operation are the sets,

$$\breve{\mathcal{T}}_n = \mathcal{T}_n \setminus \mathcal{T}_{n+1}, \tag{5.3}$$

$$\breve{\mathcal{T}}_{n+1} = \mathcal{T}_{n+1} \setminus \mathcal{T}_n. \tag{5.4}$$

5.1.1 Node Movement

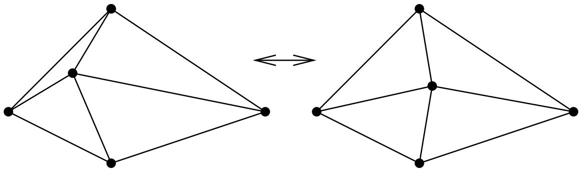


Figure 5-1: 2D node movement grid operator.

Node movement changes the locations of the nodes while keeping the tetrahedra connectivity fixed,

$$(\mathcal{X}_{n+1}, \mathcal{T}_n) = Mo(\mathcal{X}_n, \mathcal{T}_n, i, c()), \tag{5.5}$$

see Fig. 5-1 for a 2D example with triangles. This is also a popular technique for structured grid methods^{14} because of the fixed connectivity. A node location is adjusted with the neighboring node locations fixed. This method allows explicit constraints on the positivity of tetrahedral volume, i.e.,

$$V(\mathcal{X}_{n+1}, \mathcal{T}_n) > V_{\mathrm{tol}}, \tag{5.6}$$

where V_{tol} prevents the ambiguity of small element validity in floating-point arithmetic. Global relaxation is not used because explicit constraints cannot be enforced as readily. An objective function $c(\kappa)$ is defined for each tetrahedron incident to a node, $\kappa \ni x_i$. An optimization procedure is invoked to minimize a norm of the incident tetrahedron objective functions,

$$c(x_i) = \left(\sum_{\kappa \ni x_i} c(\kappa)^m \right)^{1/m}, \tag{5.7}$$

$$x_i|_{n+1} = \operatorname{argmin} c(x_i). \tag{5.8}$$

If the ∞-norm is selected, smart-Laplacian and quadratic programming optimization schemes[58] are employed. A gradient-free simplex search is used for other norms. Nodes on boundaries are moved in their respective parametrized spaces.

5.1.2 Tetrahedra Swapping

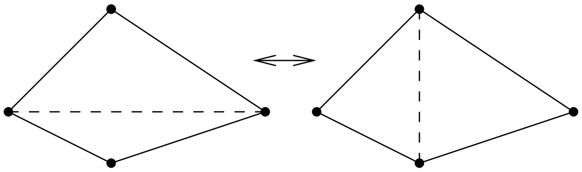


Figure 5-2: 2D element swap grid operator.

The 3D operations of face and edge swapping[58] are performed,

$$(\mathcal{X}_n, \mathcal{T}_{n+1}) = Sw(\mathcal{X}_n, \mathcal{T}_n, j, c()). \tag{5.9}$$

A 2D example with triangles is shown in Fig. 5-2, where this is a simple replacement of two triangles for two triangles. In 3D, this operator is much more complicated. The element κ_j and its neighbors are examined, and the configuration that reduces a norm of the involved tetrahedron objective functions to the greatest degree is chosen,

$$\breve{\mathcal{T}}_{n+1} = \operatorname{argmin} \left(\sum_{\kappa \in \breve{\mathcal{T}}_{n+1}} c(\kappa)^m \right)^{1/m}, \tag{5.10}$$

if it has positive volume tetrahedra,

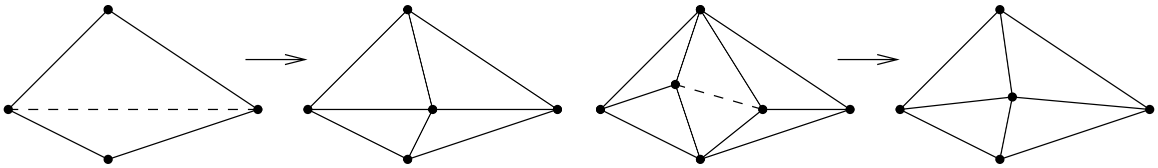
$$V(\mathcal{X}_n, \mathcal{T}_{n+1}) > V_{\text{tol}}. \tag{5.11}$$

If there is no swapped configuration with a lower norm, the current configuration is retained, which ensures

$$\left(\sum_{\kappa\in\check{\mathcal{T}}_{n+1}} c(\kappa)^m\right)^{1/m} \leq \left(\sum_{\kappa\in\check{\mathcal{T}}_{n}} c(\kappa)^m\right)^{1/m} . \tag{5.12}$$

The 3D configurations that are evaluated by the argmin in Eq. (5.10) are face swapping[58] and edge swapping.[58] Face swapping is replacing two tetrahedra that share three nodes with three tetrahedra. Edge swapping is replacing all of the tetrahedra that share two nodes with a new set of tetrahedra that fill the same volume. Edge swaps are implemented for three to seven tetrahedra surrounding an edge. The number of possible configurations that can result from an edge swap grows rapidly with the number of tetrahedra surrounding an edge. Fortunately, there is a smaller set of canonical configurations for each of these swap operations. Freitag and Ollivier-Gooch[58] describe these canonical configurations and how to reduce the cost of evaluating the norm of the new configurations. The boundary triangles are also swapped as a result of tetrahedra swapping.

5.1.3 Tetrahedra Split and Collapse



(a) Triangle split.

(b) Triangle collapse.

Figure 5-3: 2D split and collapse operators.

The split and collapse operations modify the density of the grid tetrahedra as well as contribute to obtaining the desired anisotropy. There are many possible splitting stencils.[44,100] In this work, the split operator inserts a single new node on the segment l connecting two nodes. All the tetrahedra that share l are split into two tetrahedra to include this new node. Segments that are on a boundary also result in split boundary triangles. The tetrahedra split operator is used in two forms,

$$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = Sp(\mathcal{X}_n, \mathcal{T}_n, l, L()), \tag{5.13}$$

$$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = Sp(\mathcal{X}_n, \mathcal{T}_n, j, c()). \tag{5.14}$$

Equation (5.13) splits an edge if its length function $L(l) > 1$. Equation (5.14) splits the edge $l \in \kappa_j$ of element j that produces the largest decrease in $c()$,

$$l = \underset{l \in \kappa_j}{\operatorname{argmin}} \frac{\left(\sum_{\kappa \in \breve{\mathcal{T}}_n} c(\kappa)^m\right)^{1/m}}{\left(\sum_{\kappa \in \breve{\mathcal{T}}_{n+1}} c(\kappa)^m\right)^{1/m}}. \tag{5.15}$$

In absolute precision arithmetic, these split tetrahedra have positive volume if the original tetrahedra have positive volume and the new node is placed on the segment connecting two nodes. In floating-point arithmetic, this may not be true due to floating-point errors. An additional check is placed on the computed volume after the split and the split is not permitted unless,

$$V(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) > V_{\text{tol}}. \tag{5.16}$$

For a collapse, all the tetrahedra that share l are removed. One of these two nodes is removed and the remaining tetrahedra incident to the removed node are reconnected to the remaining node. The collapse operator also has two forms,

$$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = C(\mathcal{X}_n, \mathcal{T}_n, l, L()), \tag{5.17}$$

$$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = C(\mathcal{X}_n, \mathcal{T}_n, j, c(), \bar{c}). \tag{5.18}$$

The collapse operation is only permitted if the new configuration results in positive volume tetrahedra,

$$V(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) > V_{\text{tol}}. \tag{5.19}$$

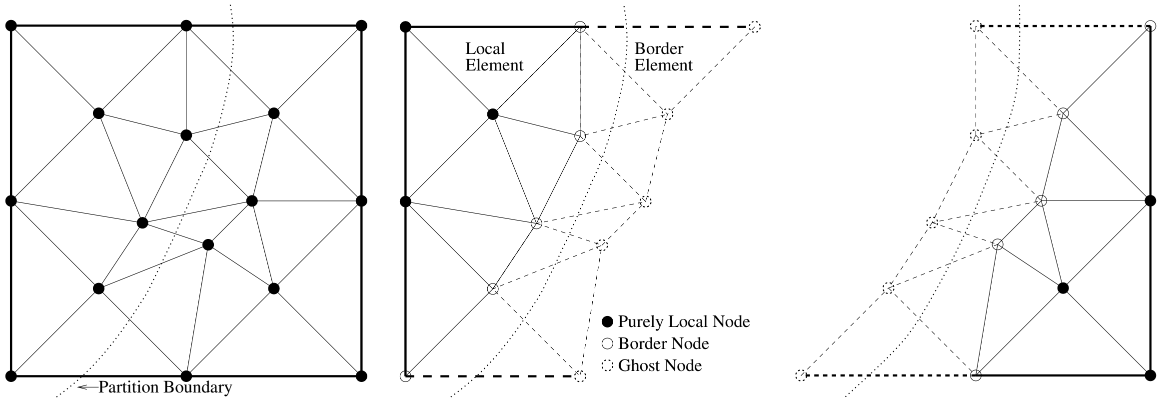
Equation (5.17) collapses an edge if $L(l) < 1.0 \; \forall \; l \in \breve{\mathcal{T}}_{n+1}$. Equation (5.18) collapses an edge of a tetrahedra if $c(\kappa_j) < \bar{c} \; \forall \; \kappa_j \in \breve{\mathcal{T}}_{n+1}$.

5.2 Parallelization

The grid operators are executed in parallel with a domain-decomposed scheme. The resulting partitions allow the independent application of operators to the partition interiors. Most of the complexity of this parallel approach arises when these operators are applied in the partition-border regions.

5.2.1 Domain Decomposition

To fit large problems on a cluster of distributed-memory machines, the domain or grid is decomposed into partitions. A node-based, domain-decomposition scheme is utilized, where each node is uniquely assigned to a single partition. Tetrahedra that span the partitions are duplicated to complete the border regions of the partitions. This is the Single Program Multiple Data (SPMD) paradigm; each partition data structure is processed by the same algorithm. The domain decomposition scheme is identical to the method utilized by FUN3D.[115]



(a) Global grid. (b) Partition 1. (c) Partition 2.

Figure 5-4: Two partition domain decomposition 2D example.

A two partition example is illustrated in Fig. 5-4 for 2D. A node is denoted local if it has been assigned to that partition (denoted filled circles and open circles). The geometry edges, boundary triangles, and tetrahedra that contain one or more local nodes are also added to the partition. These include tetrahedra that are entirely local to a partition (solid lines) or border tetrahedra that span partition boundaries (dashed lines). Local nodes that are only used to construct local tetrahedra are denoted as purely-local nodes (filled circles). Local nodes that are used to construct border tetrahedra are denoted as border nodes (open circles).

The local nodes of other partitions are added as ghost nodes (dashed circles) to the current partition as required to complete the border tetrahedra. The partition to which these ghost nodes are assigned is also stored to reduce the parallel communication required to update partition-border regions. Purely-local nodes are only connected by tetrahedra to other local nodes. A partition's border nodes are local nodes that are connected to a ghost node by a border tetrahedron. Therefore, nodes are either purely-local, border, or ghost. Tetrahedra are either local or border. Nodes have a unique global index and a local index on each partition.

5.2.2 Partition Coloring

To increase parallel efficiency, groups of unconnected partitions are allowed to simultaneously modify border element connectivities and node positions. Only allowing execution on one partition at a time results in a sequential algorithm, because the other partitions are idle until they have an opportunity to process their border region. The amount of time that processors remain idle during adaptation can be reduced by allowing concurrent execution through partition coloring.

The sets of unconnected partition groups are calculated with a greedy coloring scheme[43] where no two adjoining partitions share a color. A 17 partition example with four colors is shown in Fig. 5-5. All the partitions in a single color can then modify border regions concurrently with current ghost information. In this example, the six red partitions in Fig. 5-5 can execute at once. Without this coloring scheme, this color would require six separate grid modification steps with parallel communication between each step. With coloring, parallel communication is only required between the processing of different colors, which are less than or equal to the number of partitions.

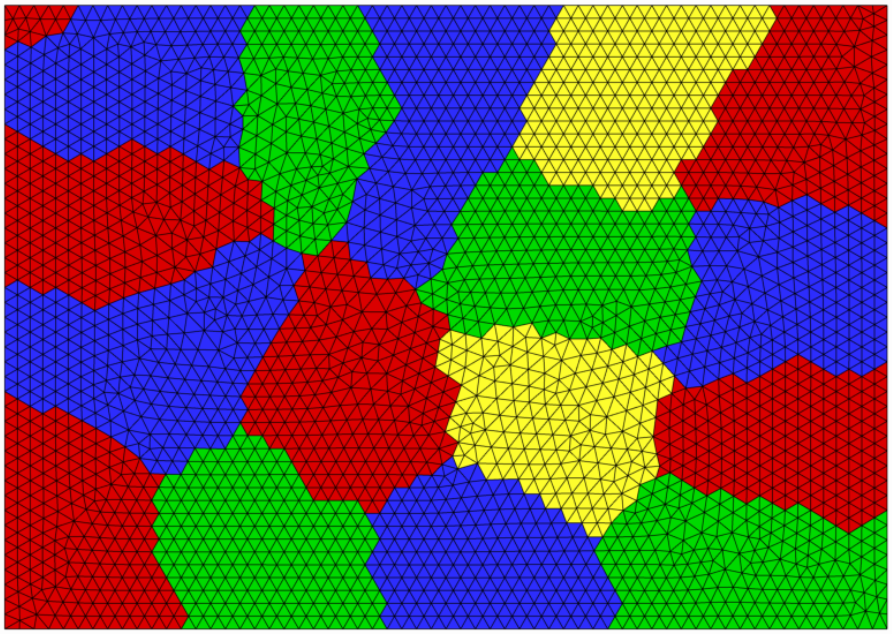


Figure 5-5: Partition coloring scheme.

5.2.3 Connectivity Changes

The connectivity of the tetrahedra change as a result of swapping Sw, splitting Sp, and collapsing C. In other parallel adaptation implementations,[8,34,44,88] connectivity changes are performed in the interior of the partition and migration is used to make border regions interior. In this work, the tetrahedron connectivity changes are performed on purely-local and border tetrahedra in separate operations without migration. These connectivity changes may be due to edge swapping, face swapping, or the insertion of new nodes. The collapse operator is currently only allowed in purely-local partition regions, because the geometry topology information required to maintain a topologically valid discrete surface triangulation is not stored with ghost nodes. This limitation may be removed with a small amount of additional storage and an increase in code complexity.

Initially, connectivity changes are only performed on local tetrahedra (solid lines, Fig. 5-4(b) and (c)). This is an efficient parallel operation because all partitions perform connectivity changes simultaneously. Updates of the border tetrahedra region (dashed lines, Fig. 5-4(b) and (c)) are performed one partition color at a time. To correctly mirror connectivity changes across partitions, transcripts are recorded of the nodes and tetrahedra that are added or removed as a result of this border tetrahedra adaptation. These transcripts are serialized and broadcast to other partitions. The receiving partitions apply the transcripts to maintain consistency between partitions.

The transcript is composed of all the data required to update connectivities (tetrahedra, geometry triangles, and geometry edge segments) and create new ghost nodes (position, face parameters, and edge parameters). The size of these transcripts is reduced by utilizing the stored ghost node partition assignments to restrict the transcript to the minimum information required to maintain consistency. Minimizing the contents of the transcripts reduces parallel communication costs and the time required to perform the specified transcript operations on the other partitions. Parallel efficiency is maintained while applying the transcripts to other partitions, because transcripts are processed concurrently. Ghost nodes that are no longer connected to a border tetrahedron after connectivity changes are removed.

5.2.4 Node Movement

The nearest-neighbor information must be current when smoothing nodes Mo to prevent the creation of invalid tetrahedra. Freitag[57] and Löhner[93] address the issues for maintaining current nearest-neighbors while node smoothing on a shared-memory architecture. In other distributed memory implementations,[8,34,88] border nodes are frozen during adaptation. Then the domain is repartitioned and migration is used to change these frozen border nodes into local nodes. A number of repartitioning steps ultimately allow the processing of all nodes as local nodes.

In this implementation, the nearest-neighbor node movement operators are applied separately to local and border regions without repartitioning. An initial pass is made through all the purely-local nodes on all partitions. These nodes do not require any information from other partitions. Therefore, the partitions can modify purely-local nodes concurrently without parallel communication. After the purely-local regions of the partition have been smoothed, the border nodes are smoothed one partition color at a time. The ghost nodes are updated after each color to ensure that nearest-neighbor information is current.

Ghost node parameters (position, face parameters, and edge parameters) must be updated when the corresponding border node on another partition is modified. This operation is complicated by tetrahedron connectivity changes during adaptation. These connectivity changes in partition borders alter the inter-partition communication pattern. To recompute the communication pattern, every partition sends a list of ghost node global indexes to the partition that is assigned those nodes. On the assigned partitions, these requests for updated nodal information are translated into local node indexes. The translation is performed in $\mathcal{O}(\log_2 n)$ time with a binary search of the global indexes of local nodes stored in a sorted list. The translated indexes allow access to the stored local values of the requested nodal position, face parameters, and edge parameter. The requested information is sent back to the partitions to update their ghost nodes.

5.2.5 Global Indexes

The global node indexes are required to reconstruct the parallel communication pattern after connectivity changes. Global cell indexes are not required for parallel adaptation, but are maintained as a convenience to the application requesting adaptation. Unique global cell indexes are maintained with exactly the same algorithm as the global node indexes, so only the global node index scheme is described.

The algorithm to assign and maintain global node indexes is simple. A partition takes ownership of a global node index during the initialization of its local nodes. Once a partition is assigned a global index, it never relinquishes it until the completion of adaptation. If a local node is deleted during adaptation, the partition stores this unused global index in a list. If the partition needs a unique global index to insert a node into the discretization, it will extract an unused global index from the list.

When this list of unused global indexes is exhausted, the partition creates a new global index by independently incrementing its copy of the total number of global indexes. This allows two or more partitions to obtain the same global index. These repeated global indexes are made unique by shifting them. After shifting the newly created global node indexes, the true count of global indexes is shared amongst partitions.

	Partition	Total	Indexes of New Nodes
Before Shift	A	103	101 102 103
	B	102	101 102
	C	100	none
	D	104	101 102 103 104
After Shift	A	109	101 102 103
	B	109	104 105
	C	109	
	D	109	106 107 108 109

Figure 5-6: An example of creating new global indexes on four partitions with 100 original global indexes, before and after shift.

Figure 5-6 provides a four processor example that begins with 100 unique global nodes. Three of four partitions need new global nodes to insert nodes while concurrently performing edge split operations. Partition C does not insert any nodes. After the edge split operation, these newly created nodes are shifted to make them unique. The correct total number of nodes is also computed by partition D and communicated to all partitions. The parallel communication required to maintain unique global indexes is reduced by allowing the temporary creation of repeated global indexes that are later corrected.

The unused global indexes are removed at the completion of the adaptation by a reverse, global-index shifting procedure. This ensures that the calling application will be returned a grid with continuous and unique global indexes.

5.2.6 Load Balancing

Load balancing is employed to maintain parallel efficiency. The parallel, graph-partitioning tool ParMETIS[136] is utilized to create well-balanced partitions with a minimum number of connections. During adaptation, the number of nodes in each partition can change significantly due to the addition and removal of nodes. Also, the communication cost can increase due to connectivity changes. The multi-constraint formulation of ParMETIS is invoked to simultaneously balance the number of cut and total control volumes. This improves the load balance of the cutting procedure as well as the flow and adjoint solvers because a cut cell residual evaluation is more expensive than an uncut cell residual evaluation. At the completion of the adaptation process, ParMETIS is called and portions of the grid are migrated to regain an optimal partitioning.

Chapter 6

Metric-Based Adaptation

A metric tensor M is commonly employed to define the desired multidimensional grid resolution because it is a natural way to express local interpolation error estimates of linear functions.[33, 59, 65, 126] The symmetric positive definite matrix M in 3D has the diagonal decomposition

$$M = X \begin{bmatrix} \Lambda_1 & & \\ & \Lambda_2 & \\ & & \Lambda_3 \end{bmatrix} X^T = X \Lambda X^T. \tag{6.1}$$

The eigenvectors X define an orthonormal basis with length specifications h_i in this basis,

$$\Lambda = \begin{bmatrix} \left(\frac{1}{h_1}\right)^2 & & \\ & \left(\frac{1}{h_2}\right)^2 & \\ & & \left(\frac{1}{h_3}\right)^2 \end{bmatrix}. \tag{6.2}$$

This metric can be interpreted as an ellipsoid with major and minor axes of direction X_i and length h_i.[126] The linear mapping J is employed to map a vector in physical space $\vec{x} = [x \; y \; z]^T$ to a vector in transformed space $\vec{x}' = [x' \; y' \; z']^T$ where a triangle or tetrahedron becomes equilateral with unit length edges; that is,

$$\vec{x}' = J\vec{x}. \tag{6.3}$$

The metric tensor M is related to J by

$$M = J^T J, \tag{6.4}$$

$$J = \Lambda^{\frac{1}{2}} X^T. \tag{6.5}$$

If $\vec{x}$ describes an edge in physical space, the length L in mapped space is

$$L = \sqrt{\vec{x}'^T \vec{x}'}. \tag{6.6}$$

Employing Eq. (6.3), this expression of length becomes

$$L = \sqrt{(J\vec{x})^T (J\vec{x})}, \tag{6.7}$$

$$L = \sqrt{\vec{x}^T M \vec{x}}. \tag{6.8}$$

Equation (6.8) is employed to directly compute edge lengths in the specified metric. Equation (6.3) is applied to the physical coordinates of tetrahedron nodes to map them to the transformed space before evaluating a quality of the tetrahedron's shape.

The resolution request is stored as M during adaptation instead of storing J directly due to these benefits:

- It is compact; only 6 entries of the 3D symmetric M are stored.
- Multiple M can be readily interpolated and intersected.[33, 59]
- Lengths in the specified metric can be computed efficiently with Eq. (6.8).

A slight computational cost is incurred computing J from M when J is needed to obtain a metric transformed tetrahedron. This cost is a 3×3 symmetric positive definite matrix diagonalization and Eq. (6.5). This cost is minimized by tridiagonalizing M with a single rotation and applying an iterative method to find the eigendecomposition of the tridiagonal matrix.

Venditti and Darmofal[147] incorporated the adjoint adaptation parameter into an anisotropic Hessian-based framework. This combined approach sets the anisotropy of mesh elements by using the Mach Hessian, and scales the element size so that the tightest spacing is dictated by the adjoint adaptation parameter.

The Mach Hessian H is decomposed into eigenvalues Λ and eigenvectors X. The matrix M has the same eigenvectors as matrix H and absolute value of the corresponding eigenvalues, making it symmetric positive definite. The adjoint adaptation parameter is incorporated into the Hessian framework by scaling the three eigenvalues so that the largest eigenvalue corresponds to the adjoint adaptation spacing requirement. The eigenvalues and corresponding eigenvectors are sorted so that $|\Lambda_1| > |\Lambda_2| > |\Lambda_3|$. Thus, after scaling, the largest eigenvalue is

$$\Lambda_1 = \left(\frac{1}{h}\right)^2, \tag{6.9}$$

where the specified element length h is computed by the output error estimation process, Eq. (4.26).

6.1 Shape Measure

In 3D, considering only edge length as a shape measure can result in degenerate tetrahedra, because a nearly-zero-volume 'sliver' tetrahedra can be constructed without a short edge. Shape measures[49,92] are employed to penalize near-degeneracies and provide a smooth function to optimize. These measures are based on interpolation error estimates, departure from a right or isotropic tetrahedron, or transformation matrix conditioning.[140] A norm of the mesh conformity to a specified multidimensional grid resolution[84] can also be stated directly.

Shape measures are typically defined in the range $[0, 1]$, with 0 denoting a degenerate tetrahedron and 1 denoting an ideal tetrahedron. In this work, the inverse of a shape measure is used so that grid optimization becomes a minimization problem. The inverse of the mean ratio[92] is

$$\eta(\kappa) = \frac{L_{01}^2 + L_{02}^2 + L_{03}^2 + L_{12}^2 + L_{13}^2 + L_{23}^2}{12(3V)^{2/3}}, \tag{6.10}$$

where V is the volume of the tetrahedron and L are the six lengths of vectors defined between nodes 0–3 of the tetrahedron. It is in the range of $[1, \infty]$, where an isotropic tetrahedra is 1 and a degenerate tetrahedra is ∞. The mean ratio is efficient to evaluate and a well-behaved continuously differentiable function,[140] which makes it very suitable for gradient-based optimization. It also has a connection to metric conformity without size specification.[84] Minimizing η in the specified metric produces tetrahedra that are isotropic in the metric space. This tends to produce tetrahedra with large dihedral angles in physical space. Shape measures that penalize large angles can have discontinuous derivatives, making them difficult to optimize with gradient-based methods, see Shewchuk[140] for examples.

6.2 Metric Adaptation Iteration

The goal of the ad-hoc adaptive processes is to produce a grid with the following properties in decreasing priority:

- All edges have a length less than or equal to unity in the specified metric field.
- The number of nodes and tetrahedra in the mesh should be reduced by collapsing edges shorter than unity in the metric field.
- The tetrahedra shape quality in the metric should be improved.

These are potentially contradictory goals that do not unify to a single minimization statement. However, grid validity with elements equal or smaller than the metric is possible with tetrahedra splits alone for a domain with planar boundaries, avoiding being trapped in the local minima of an objective function. The last two goals are for efficiency (minimizing degrees of freedom) and regularity of the resulting grid.

An iterative procedure to reach these goals is:

- Sort the tetrahedra by decreasing η in the specified metric. Consider the elements one-by-one, swap them with their neighbors to minimize the maximum

η of the involved tetrahedra ($m = \infty$),

$$(\mathcal{X}_n, \mathcal{T}_{n+1}) = Sw(\mathcal{X}_n, \mathcal{T}_n, j, \eta()) \; \forall \; \kappa_j \in \mathcal{T}_n \text{ from largest } \eta(\kappa_j) \text{ to smallest } \eta(\kappa_j). \tag{6.11}$$

If a element is changed remove it from the queue.

- Perform the node movement operation on all nodes from largest $\eta(x_i)$ over incident tetrahedra to smallest,

  $$(\mathcal{X}_{n+1}, \mathcal{T}_n) = Mo(\mathcal{X}_n, \mathcal{T}_n, i, \eta()) \; \forall \; x_i \in \mathcal{T}_n \text{ from largest } \eta(x_i) \text{ to smallest } \eta(x_i). \tag{6.12}$$

  Smart-Laplacian and quadratic programming optimization schemes[58] are performed on the prioritized node list to minimize η.

- The orbit of tetrahedra that surround segments connecting two nodes are examined for potential collapse if the length of the edge in the metric is $L(l) < 0.3$.

  $$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = C(\mathcal{X}_n, \mathcal{T}_n, l, L()) \; \forall \; l \in \mathcal{T}_n : l < 0.3 \text{ from smallest } L \text{ to largest } L. \tag{6.13}$$

- Sort the grid edges from largest l to smallest. The edges with $l > 1.0$ are split,

  $$(\mathcal{X}_{n+1}, \mathcal{T}_{n+1}) = Sp(\mathcal{X}_n, \mathcal{T}_n, l, L()) \; \forall \; l \in \mathcal{T}_n : l > 1.0 \text{ from largest } L \text{ to smallest } L. \tag{6.14}$$

  A element newly created from a Sp operation is not added into the split queue.

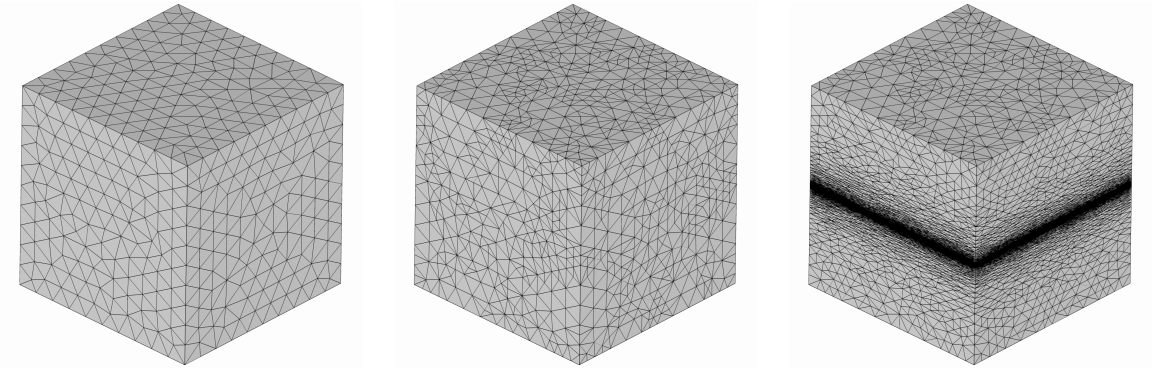
- The adaptation process repeats until $L(l) < 1.0 \; \forall \; l \in \mathcal{T}_n$.

6.3 Analytic Metric Adaptation

The volume grid of a unit cube domain $[0,1] \times [0,1] \times [0,1]$ is adapted to an analytic metric field. The original grid, Fig. 6-1(a), is created by the FELISA grid generator[125] through the GridEx framework[74] with an isotropic spacing of 0.1. This original grid is adapted to ensure that all edges are less than or equal to 0.1, Fig. 6-1(b). An anisotropic metric

$$[0.1\hat{\imath}, h + (0.1 - h)\frac{|y - 0.5|}{0.5}\hat{\jmath}, 0.1\hat{k}] \tag{6.15}$$

is used for adaptation with h decreasing from 0.1 to 0.0001. This clustering is evident on the $x = 0$ and $z = 0$ visible surfaces of the cube, Fig. 6-1(c). The volume grid is clustered near the $y = 0.5$ plane. To reach this high anisotropy, the metric in the $\hat{\jmath}$-direction is reduced in a number of steps. The grid is adapted to this intermediate metric at each step. The adaptation history of edge length is shown as a histogram at the beginning of each adaptation iteration for $h = 0.0001$, Fig. 6-2. The x-axis of each histogram is $\log_{10}(L)$. At the completion of each adaptation step, all edges are shorter than one in the metric and very few are shorter than $\log_{10}(0.3) \approx -0.52$.



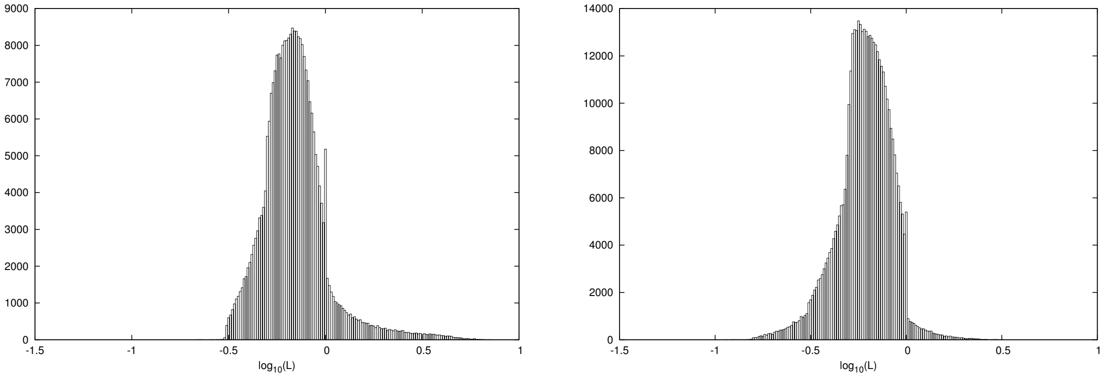
(a) FELISA grid.

(b) All edges smaller than metric 0.1–0.1–0.1.

(c) All edges smaller than Eq. (6.15) metric, $h = 0.0001$.

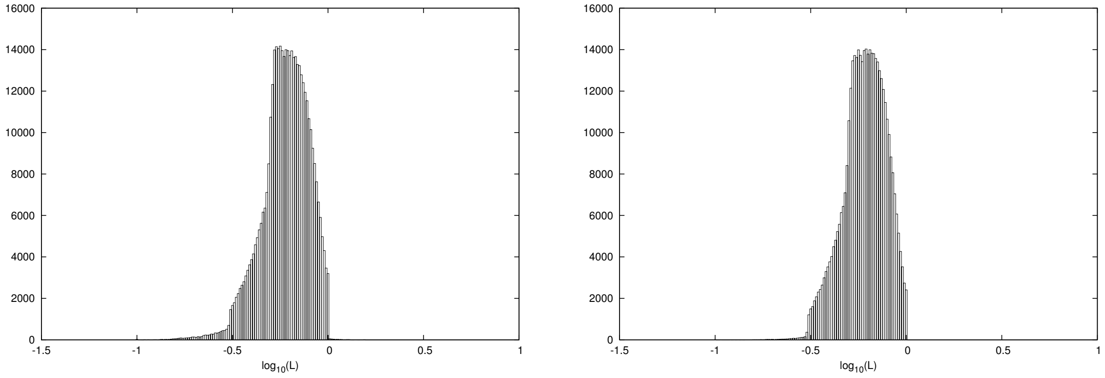
Figure 6-1: Isometric views of a cube.

Anisotropic scaling of the figures aids illustration of the final grid corresponding to this simple metric field. The $z = 0$ face of the Fig. 6-1(c) cubic grid is shown in Fig. 6-3. The y-axis is progressively scaled in the series of subfigures, but the grid itself remains fixed. Figure 6-3(a) shows the anisotropically adapted grid with 1–1 scaling. The y-axis scales by a factor of ten in each subsequent subfigure while the x axis is held constant. Figure 6-3(d) shows the anisotropically adapted grid with 1–1000 scaling, which results in a figure width of 1 and a figure height of 0.001 in



(a) Beginning of first cycle.

(b) Beginning of second cycle.



(c) Beginning of third cycle.

(d) Beginning of fourth cycle.

Figure 6-2: Histogram of the log of edge length in metric $\log_{10}(L)$ for adapted cube for $h = 0.0001$.

physical space. The center of the strongly anisotropic grid appears isotropic in this anisotropically scaled view as a result of this mapped isotropic method.

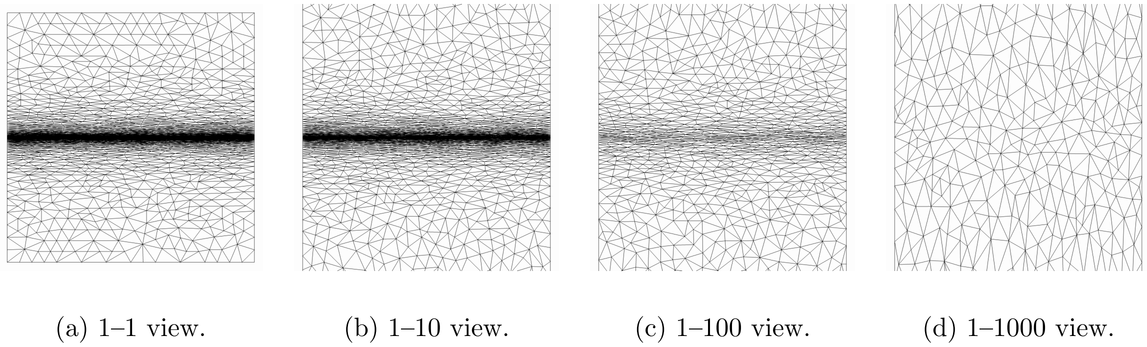

(a) 1–1 view. (b) 1–10 view. (c) 1–100 view. (d) 1–1000 view.

Figure 6-3: Face of a cube ($z = 0$) adapted to Eq. (6.15) metric, $h = 0.0001$.

6.4 Embedded- and Single-Grid Output Adaptation for Wing Configuration

Output (drag) adaptation is applied to the ONERA M6 wing at Mach 0.84 and 3 deg angle of attack. A sequence of 6 adapted body-fitted grids, circles Fig. 6-4, is produced by an embedded-grid error estimation process. This ONERA M6 wing application was previously reported by Park.[122] The 0.0019 drag coefficient output error tolerance is satisfied by the last grid. The output error estimate and grid resolution request is described in Chapter 4.

The single-grid output-based adaptation is also applied to the transonic ONERA M6 wing. This case utilizes an initial background grid with fewer control volumes for the cut-cell drag adaptation. The square symbols in Fig. 6-4 show the computed drag coefficient and remaining error estimate for the single-grid error estimation adaptive approach with cut cells. The final adapted grid satisfies the drag coefficient output error tolerance of 0.0019. The final adapted cut-cell grid has more control volumes than the body-fitted example for the same accuracy. While the cut-cell approach is less efficient in terms of control volumes for the same accuracy, it utilizes more robust adaptation mechanics that enable application to more complex geometries by avoiding the requirement of a boundary conforming adapted grid.

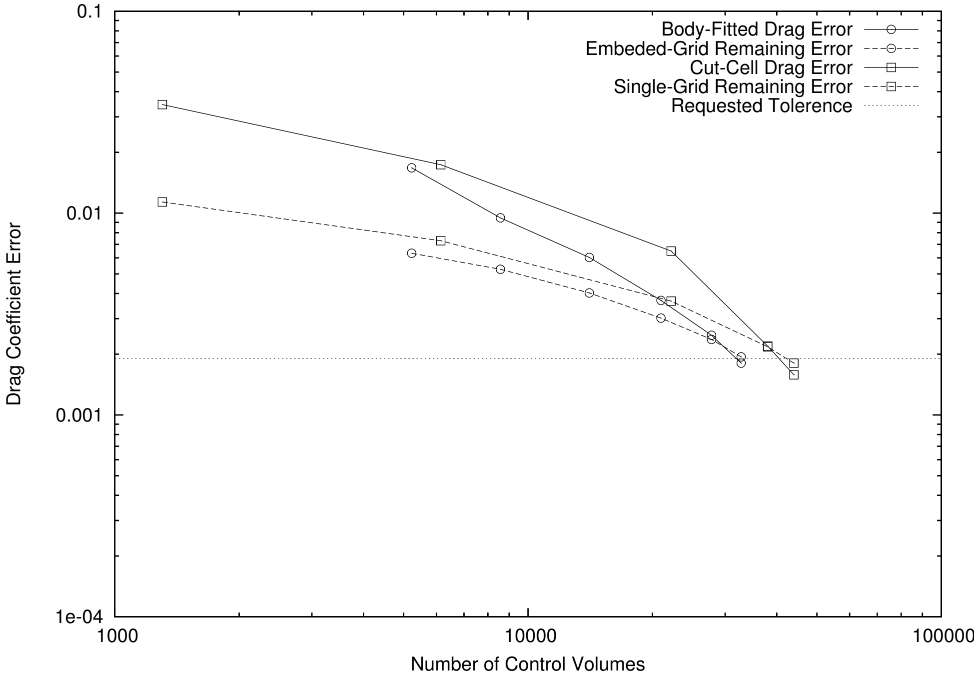


Figure 6-4: Drag coefficient, error estimation, and remaining error for embedded-grid body-fitted and single-grid cut-cell drag adaptation on a Mach 0.84 ONERA M6 wing.

6.5 Application to Sonic Boom Prediction

The parallel metric-based adaptation algorithm is applied to sonic boom prediction. Tetrahedral background grids are adapted and the model geometries are simulated with cut cells. The metric is obtained from an output error estimate of the integral of quadratic pressure deviation over a surface s in the domain,

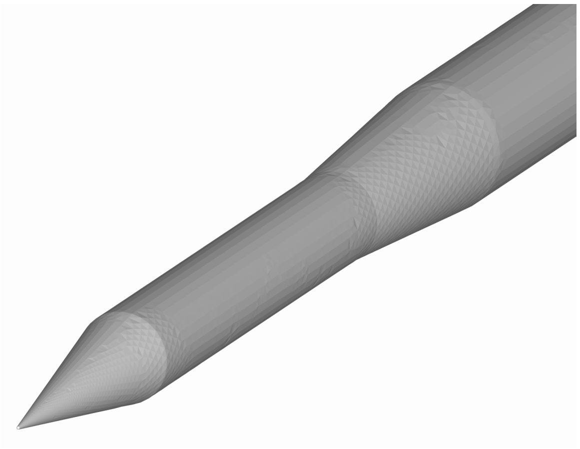
$$f = \frac{1}{A_s} \iint_s \left(\frac{p - p_\infty}{p_\infty} \right)^2 \, \mathrm{d}s, \tag{6.16}$$

where A_s is the area of the integration surface. This focuses the adaptation on improving the calculation of pressure near this surface. Previous applications have been performed with the integral of pressure deviation.[75,87] However, the square of this deviation has been shown to produce more accurate signatures with less control volumes.[113] A cylindrical integration surface, aligned to the x-axis, is employed. The extent of the integration surface can be optionally restricted to the interior of a box to focus on a subsection of the cylinder.

The current output-based adaptation approach is validated with wind tunnel measurements. Wind tunnel testing of sonic boom configurations is a challenging task. Wind tunnel models are typically small to obtain pressure signatures a relatively large distance from the model within the finite size of tunnel test sections. Carlson and Morris[31] present some of difficulties inherent in wind tunnel testing of these small models including extraneous variations in pressure larger than the signals measured. A test apparatus and procedure that mitigates the extraneous spatial and temporal distortions is described. Morgenstern[107] also documents variations in ambient static pressure wind tunnel measurements that are of the same magnitude as the desired signature measurement.

6.5.1 Double-Cone Cylinder

(a) Drawing.[72]



(b) Shaded Surface.

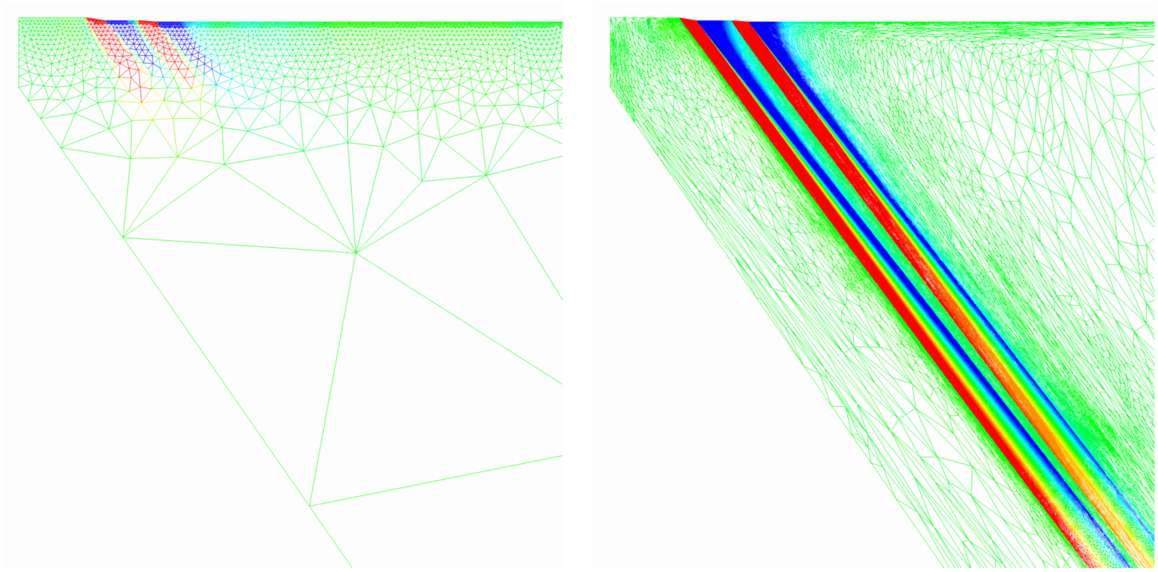
Figure 6-5: Double cone-cylinder geometry.

A double cone geometry, denoted “Model 8” in a 1965 wind tunnel report,[29] is shown in Fig. 6-5 with a shaded triangular surface grid. This same case was employed to evaluate[87] and then validate[75] a parallel adaptive body-fitted grid approach. This configuration has also been used by other researchers to evaluate their signature prediction techniques.[79, 120] The pressure integral output function was defined as a cylinder, six body lengths in radius, centered about the geometry axis. The cylinder is clipped forward of 3 body lengths behind the nose, aft of 9 body lengths behind the nose, and outside of 0.1 body lengths off the centerline to focus only on the region where wind tunnel data is available. The surface grid is Boolean subtracted from a 9-degree wedge-shaped background tetrahedral volume grid. A symmetry plane of the volume grid is shown in Figure 6-6. The initial grid (4,000 control volumes) was created with no prior knowledge of where the shocks would propagate through the domain, Fig. 6-6(a). The free stream Mach number is 1.26 and the heuristic limiter is employed during adaptation. The parallel execution scheme used 32 partitions, and the 17th adapted grid (7,500,000 control volumes) is shown in Fig. 6-6(b). The shocks have been implicitly targeted and refined to propagate the signal to the pressure integral surface. The anisotropy of the grid, based on the Mach Hessian, is clearly evident. This anisotropy reduces the number of required control volumes.

The adaptation history of the pressure integral and its remaining error estimation is shown in Fig. 6-7. The remaining error estimate is given by Eq. (4.22). Error is underestimated on the initial few adapted grids before the shocks are propagated to the integration surface. Once this connection is established, the error is reduced.

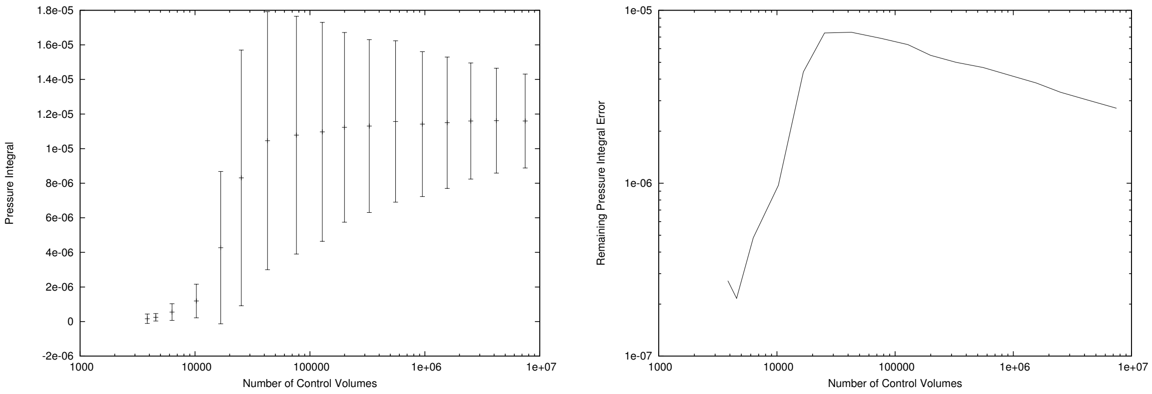
The adaptation history of the pressure signature extracted at 6 body lengths is shown in Fig. 6-8. The circular symbols are digitized from a wind tunnel report.[29] The solid line is the final, adapted signature. The signal is absent on the original coarse grid. The extrema of the pressure signal start to form and grow. The inflection points at $x/l = 2.3$ is the last part of the signal to form. The over- and under-shoots of the signal intensify on the final few adapted grids as the grid-shock alignment improves.

The final adapted grid is simulated with the Venkatakrishnan, heuristic, and Barth-Jespersen limiters in Fig. 6-9. The Venkatakrishnan limiter has similar over- and under-shoots to the heuristic limiter. The Barth-Jespersen limiter produces a



(a) Initial grid. (b) Output adapted grid.

Figure 6-6: Symmetry planes of initial and output adapted double cone 3D volume grids.



(a) Pressure integral. (b) Pressure integral remaining error.

Figure 6-7: Model 8 pressure integral and uncertainty convergence at 6 body lengths.

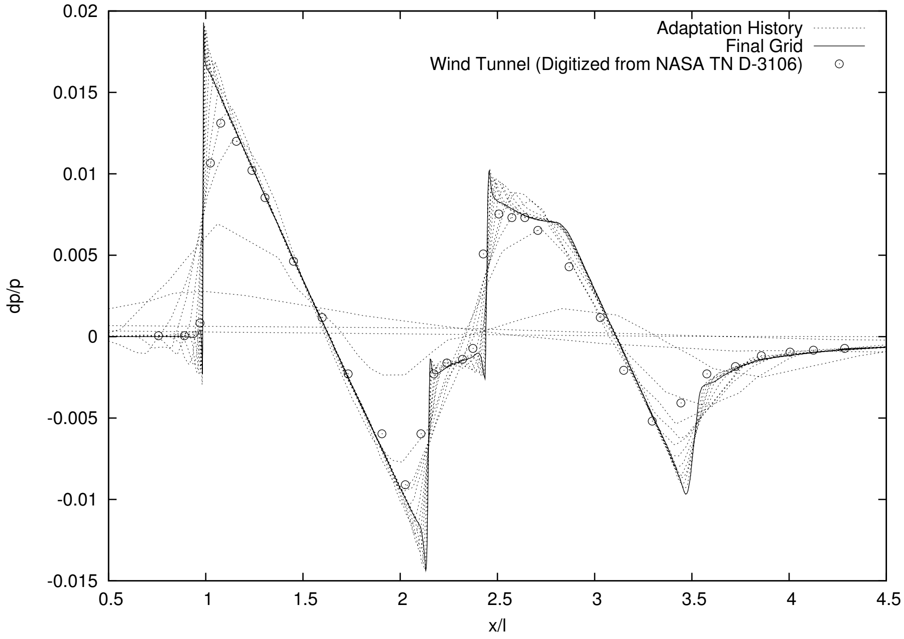


Figure 6-8: Model 8 pressure signature adaptation history at 6 body lengths.

signature without over- and under-shoots. All of these limiters have very similar signatures except at the discontinuities.

The number of control volumes is shown as a function of wall clock time for the 32 processor parallel adaptation in Fig. 6-10. The time required for the cut-cell preprocessing is between the circle and square symbols. The time required for the flow solution is between the square and triangle symbols. The time required for the flow solution is between the triangle and × symbols. The time required for error estimation and adaptation is between the × and circle symbols. The grid grows in size during the adaptation step and is fixed during the other phases. The time required for the error estimation and adaptation step is approximately equal to the combined time of cutting, flow solve, and adjoint solve at each iteration.

Lee-Rausch et al.[87] applied output-based adaptation in 9-degree wedge-shaped domain with an unstructured body-fitted method. Twelve adaptation cycles produced a grid of 2.2 million control volumes. A reconstruction limiter was not used and a similar signal to the current results was calculated. Jones, Nielsen, and Park[75] used the same output-adaptation method to compute signals at 18 body lengths and presented data at 6, 10, and 18 body lengths. Their 16th grid contained 400 thousand

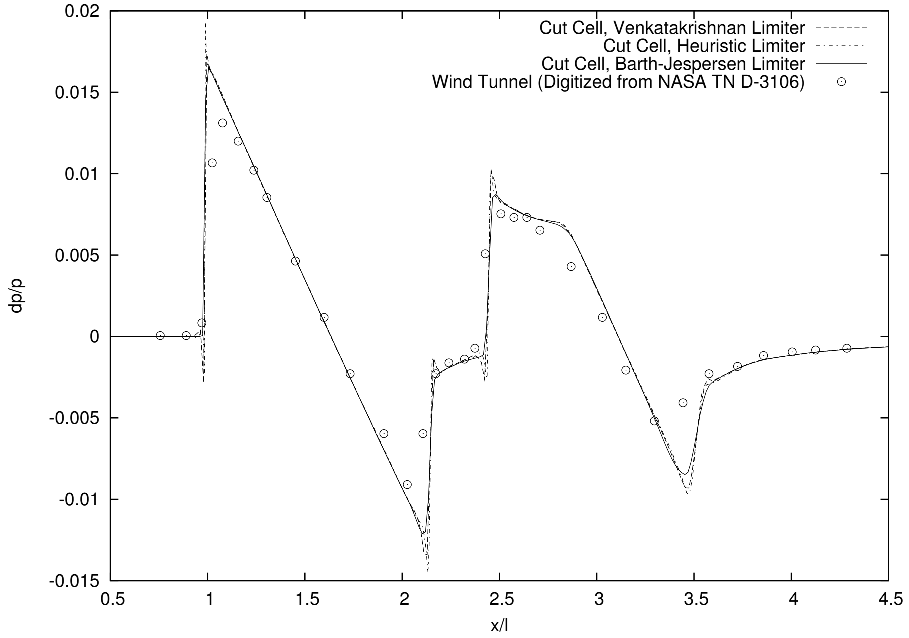


Figure 6-9: Model 8 final adapted pressure signature at 6 body lengths for various limiters.

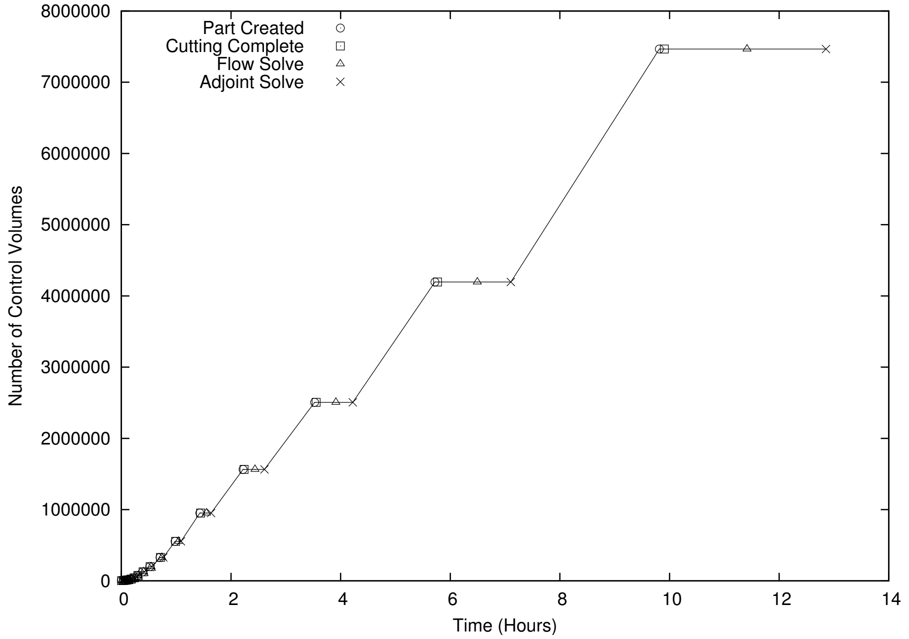
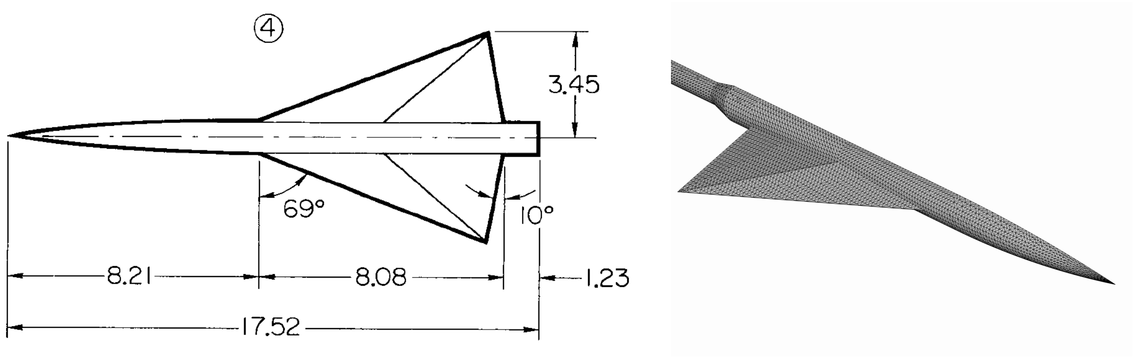


Figure 6-10: Wall time required for Model 8 pressure signature at 6 body lengths.

nodes, but the signal at 6 body lengths appears less resolved than the current results. Kandil and Ozcer[79] use an adaptive structured grid Euler method that includes shock fitting to 6 body lengths. They also apply adapted structured and unstructured Euler methods to 2 body lengths, which are then propagated to 6, 10, and 18 body lengths with a full potential method. Grid sizes are not reported. Ozcer and Kandil[120] apply an adaptive-grid unstructured method which uses a combination of normal Mach number and pressure. Results are presented at 6, 10, and 18 body lengths for a 4-degree wedge-shaped domain with a final grid size of 468 thousand nodes. The signal at 6 body lengths appears less resolved than the current results.

6.5.2 Delta Wing Body



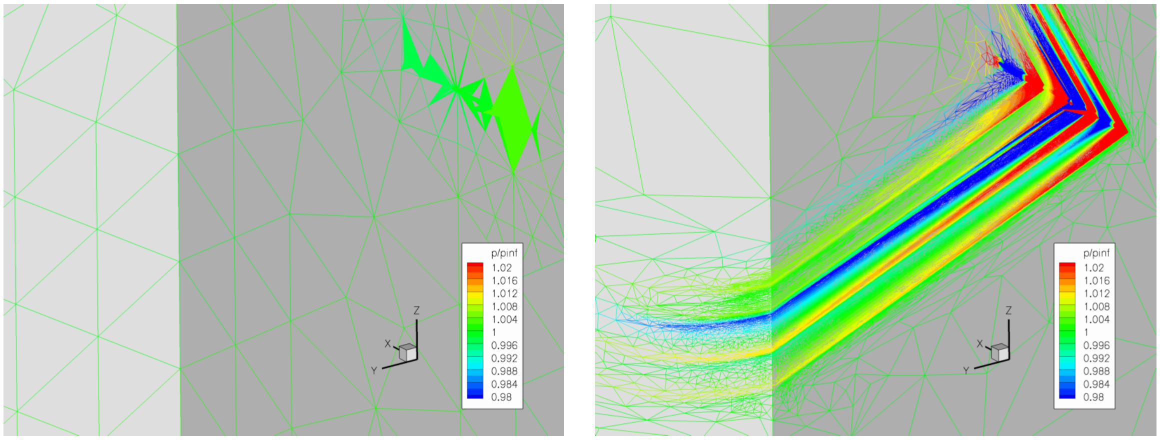
(a) Drawing (in cm.).[72]

(b) Shaded Surface.

Figure 6-11: Delta wing geometry.

A delta wing body, denoted “Model 4” in a 1973 wind tunnel report,[72] is shown in Fig. 6-11 with the triangular surface grid. This case has also been used by other researchers to evaluate their techniques.[38, 41, 47, 78, 97] The delta wing body was simulated at Mach 1.68 and 0.0 deg angle of attack. The cylindrical integration surface has a radius of 3.6 body lengths and is clipped fore ($x/l = 4.3$) and aft ($x/l = 6.6$) to restrict its extent to the location of the wind tunnel data. These clipping locations are 0.6 body lengths ahead of and 1.7 body lengths behind of the free stream Mach cone emanating from the nose. The width of the integration surface is clipped to half a wingspan. Figure 6-12 is the initial (2,800 control volumes) and final adapted

(4,900,000 control volumes) grids colored with pressure. The background grid exit plane and symmetry plane intersected with the model surface grid is shown. The initial grid is extremely coarse. The model is contained in a small number of initial background grid control volumes. The final adapted grid is well aligned with the propagated signal. The refinement region coarsens very rapidly away from the symmetry plane (the integration surface is only a half wingspan wide).



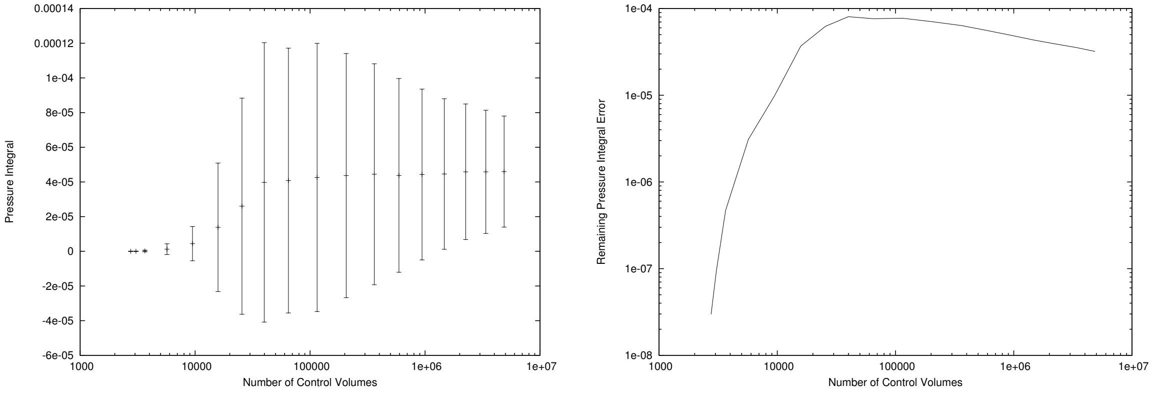
(a) initial grid.

(b) output adapted grid.

Figure 6-12: Exit and symmetry planes of initial and output adapted delta wing body background grids.

The adaptation history of the pressure integral and remaining error estimation is shown in Fig. 6-13. Error is initially under predicted for this extremely coarse initial grid. The robustness of the adaptive procedure is illustrated by the use of this coarse initial grid. The remaining error estimate improves as the shocks are resolved and propagated to the integration surface on grids smaller than 100,000 control volumes. This remaining error estimate steadily decreases on grids larger than 100,000 control volumes as the signature is refined. The grid has grown in size by three orders of magnitude.

Figure 6-14 is the pressure signatures for the series of grids employed during adaptation. The pressure signature is not apparent on the initial grid. The signals grow as a result of adaptation and the over- and under-shoots increase on the final few grids. The heuristic limiter is employed during adaptation. Figure 6-15 is the



(a) Pressure integral.

(b) Pressure integral remaining error.

Figure 6-13: Delta wing body pressure integral and uncertainty convergence at 3.6 body lengths.

final adapted grid pressure signatures for the Venkatakrishnan, heuristic, and Barth-Jespersen limiters. The signatures of all three limiters are very similar, except near discontinuities. The Barth-Jespersen limiter eliminates the over- and under-shoots of the bow and tail shocks. The difference between the Venkatakrishnan and heuristic limiter signatures is very small.

The number of control volumes is shown as a function of wall clock time for 24 processors in Fig. 6-16. The time require for the cutting procedure was larger portion of the total iteration time for the last two iterations. This is due to a load imbalance in the parallel inside/outside determination flood-fill (see Chapter 2 for a description).

Two groups of researchers computed the signature at at 0.3 body lengths and propagated it to 3.6 body lengths. Cliff and Thomas[41] used a structured grid of 1.5 million nodes and an unstructured grid of 177 thousand nodes. Djomehri and Erickson[47] used an adapted grid with 193 thousand nodes. Madson[97] used an adaptive Cartesian full-potential method to 0.1, 0.2, and 0.325 body lengths with a grid of 330 thousand boxes, which was then propagated to 3.6 body lengths. Kandil et al.[78] apply a structured Euler code to 0.4 body lengths, which is further propagated to 3.6 body lengths with a full potential code for the delta wing case at $C_L = 0.08$. Cheung, Edwards, and Lawrence[38] applied various methods to examine the same geometry at 2.7 Mach with signals propagated to 3.1 body lengths. None of the previously

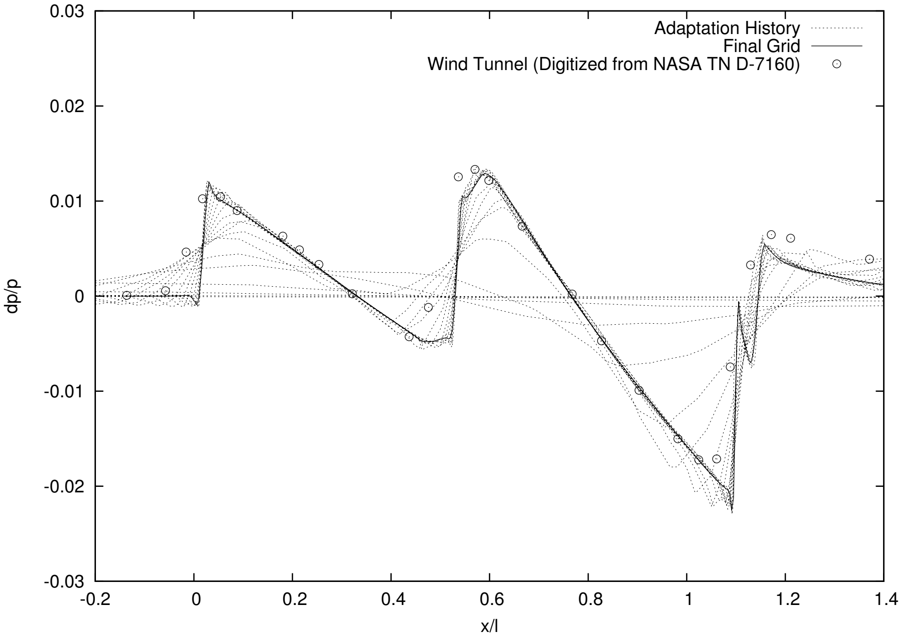


Figure 6-14: Delta wing body pressure signature adaptation history at 3.6 body lengths.

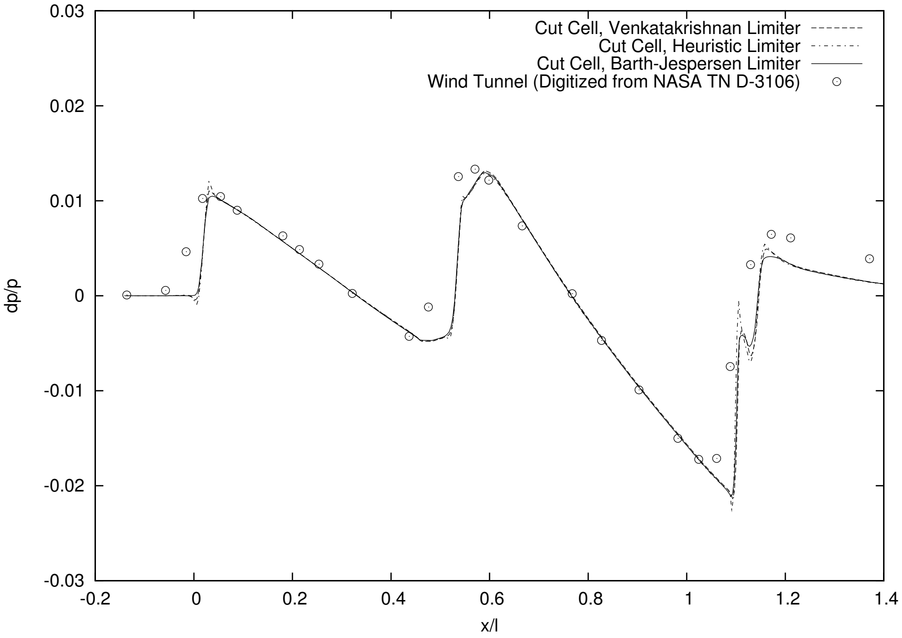


Figure 6-15: Delta wing body final adapted pressure signature at 3.6 body lengths for various limiters.

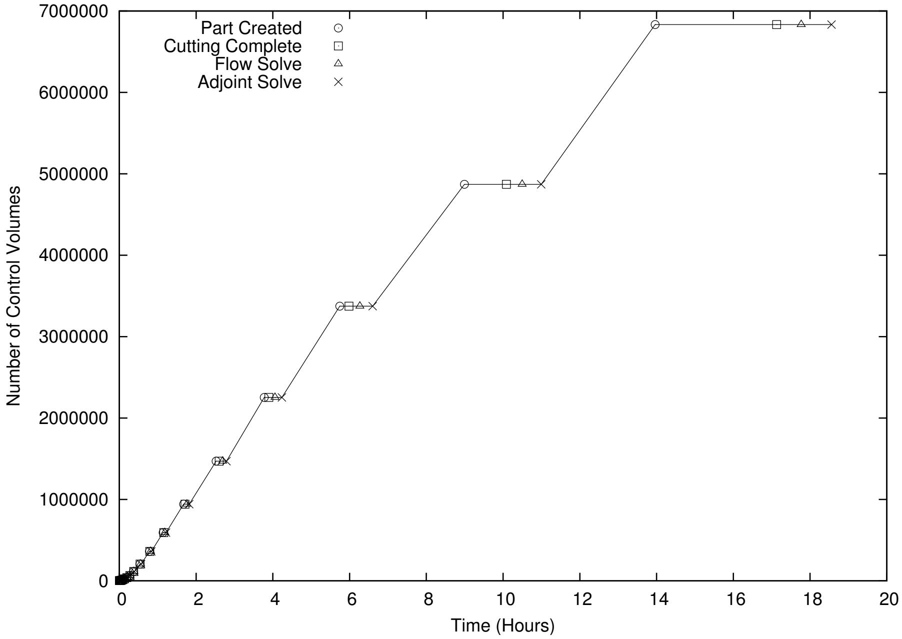
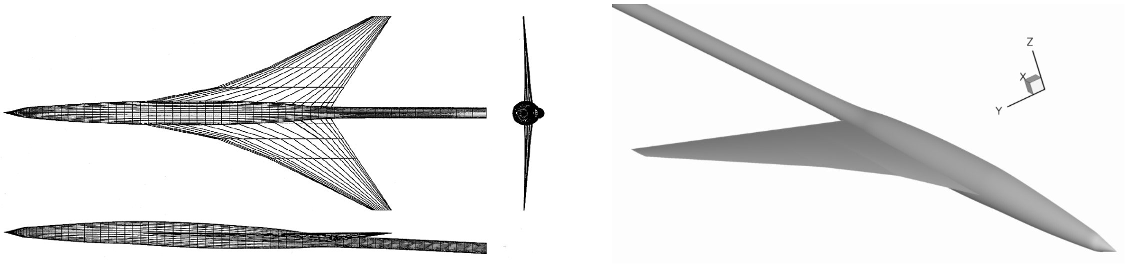


Figure 6-16: Wall time required for delta wing body pressure signature.

reported methods used a single Euler method to a distance of 3.6 body lengths.

6.5.3 Low-Boom Wing Body



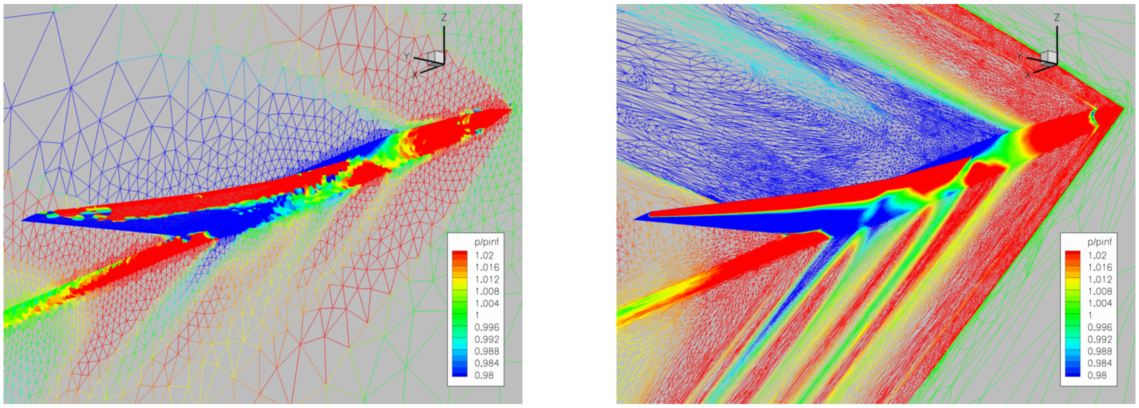
(a) Three-view drawing.[95]

(b) Shaded Surface.

Figure 6-17: Low-boom wing body geometry.

The Straight-Line Segmented Leading Edge (SLSLE) low-boom configuration, Fig. 6-17, is described by Mack and Kuhn.[95,96] These reports provide wind tunnel data from two tests, performed at the Langley Research Center Unitary Plan Wind Tunnel Facility[95] and the John Glenn Research Center 10×10 ft Wind Tunnel Facility.[96] The test condition is Mach 2.0. The model surface geometry for the aircraft includes the model and sting incidence that provides the wind tunnel lift coefficient $C_L = 0.08309$ so the angle of attack is zero.[85] The configuration has a finite thickness trailing edge, which was modeled with a transpiration boundary condition to prevent a strong inviscid supersonic corner flow expansion. Preliminary body-fitted results for this configuration[87] extended the blunt trailing edge to sharp trailing edge to avoid the strong supersonic expansion.

The original symmetry plane and cut surface grid colored with pressure is shown in Fig. 6-18(a). A linear distribution of pressure is shown in each control volume, resulting in a discontinuous pressure distribution on the surface. The initial background grid is isotropic. The final adapted symmetry plane and cut surface grid colored with pressure is shown in Fig. 6-18(b). The initial background grid contains 40,000 control volumes, and the final adapted background grid contains 5,700,000 control volumes. The anisotropy of the adapted grid has been established with the Mach Hessian, aligning the grid with the shocks.

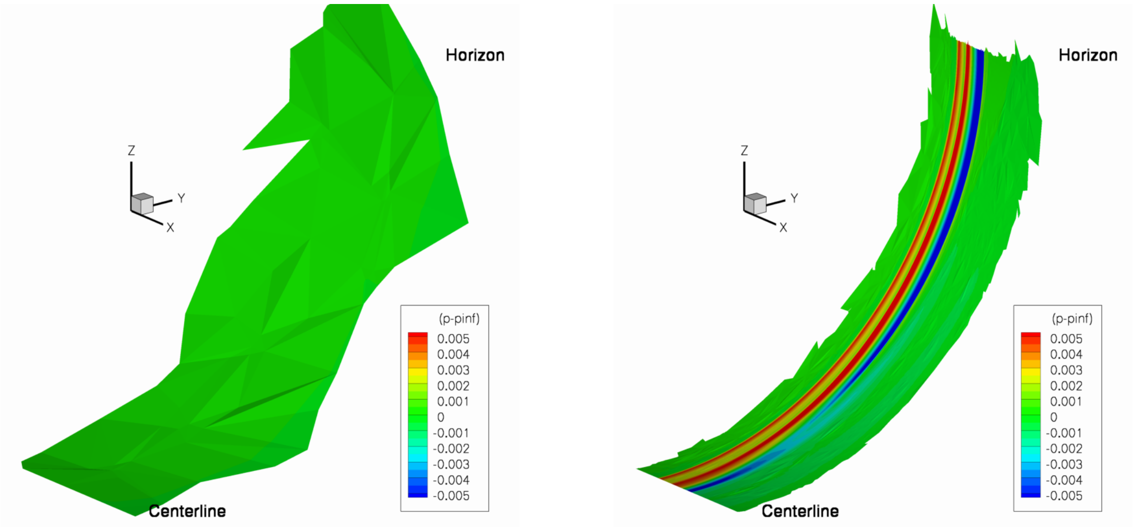


(a) Initial grid.

(b) Final grid.

Figure 6-18: SLSLE surface grid colored with pressure.

The initial, Fig. 6-19(a), and final, Fig. 6-19(b), grid integration surfaces are colored with pressure deviation from free stream. These cylindrical integration surfaces are used to compute the output, which drives the adaptation. The cylinder has a radius of 10 body lengths, which is the location of the most distant wind tunnel data that is available. The cylinder is clipped ahead of 32.6 body lengths aft of the model and behind 41.0 body lengths aft of the model. The cylinder is restricted to its lower quadrant. The initial integration surface is poorly resolved due to the initial coarse grid. The pressure signature is not visibly propagated to this initial integration surface. The final adapted grid integration surface is a much better approximation of a cylinder due to the background grid refinement. The peak signature pressure is larger at the horizon than the centerline because the model is designed to have a reduced centerline pressure signature.

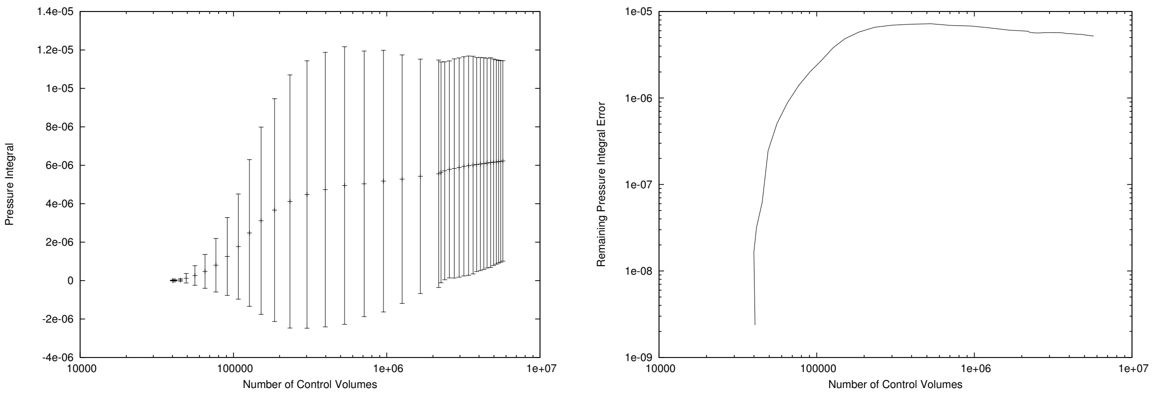


(a) Initial grid.

(b) Final grid.

Figure 6-19: SLSLE pressure on quarter cylinder integration surface 10 body lengths below the model.

The adaptation history of the pressure integral and its error estimation is shown in Fig. 6-20. The requested error tolerance tol_Ω is set to half of I_Ω at each adaptive iteration for grids sized less than 2,000,000. Above 2,000,000 control volumes, the tol_Ω is set to I_Ω, reducing the rate of grid growth. The goal of increasing tol_Ω is to obtain more resolved results with a more efficiently distributed and aligned grid at the expense of wall clock time and more adaptation cycles.[54] The change in requested error tolerance is observed as a reduction in grid growth per adaptive iteration in Fig. 6-20(a). This case shows a less dramatic reduction in the remaining error estimate over the final few grids than the previous cases, which may be due to the less aggressive $\text{tol}_\Omega = I_\Omega$ on the final grids.



(a) Pressure integral.

(b) Pressure integral remaining error.

Figure 6-20: SLSLE pressure integral and uncertainty convergence at 10 body lengths.

The pressure signatures at one body length for the series of grids employed during adaptation is shown in Fig. 6-21. One body length is much closer than the integration surface, but the signal must be resolved at this location for it to propagate to the integration surface. The pressure signature is broadly smeared on the initial grid. The signals grow in amplitude with adaptation and the pressure peaks sharpen on the final few grids. The heuristic limiter is employed during adaptation.

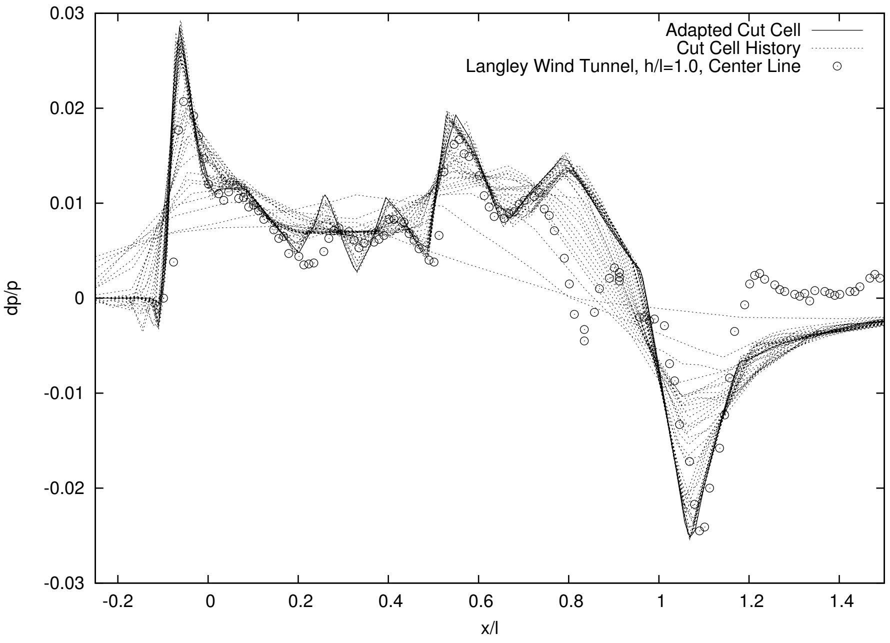


Figure 6-21: SLSLE pressure signature adaptation history at 1.0 body lengths.

The Venkatakrishnan, heuristic, and Barth-Jespersen limiters are applied on the final grid. The resulting pressure signatures at one body length are shown in Fig. 6-22. The signatures of all three limiters are very similar, except near discontinuities. The Barth-Jespersen limiter reduces the over- and under-shoots of the bow and tail shocks. The difference between the different limiter signatures is greater for this case than the cone cylinder and delta wing body cases.

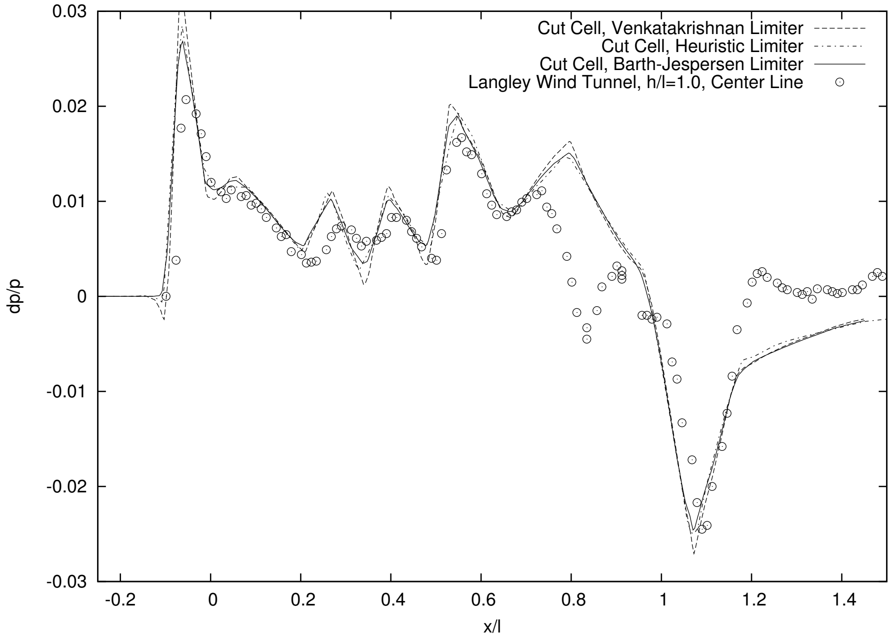


Figure 6-22: SLSLE final adapted pressure signature at 1.0 body lengths for various limiters.

Centerline pressure signatures are presented in Fig. 6-23 for 1.0, 1.5, 2.0, and 2.5 body lengths below the model. Langley wind tunnel data is available for all four locations, but the closest Glenn wind tunnel data is available at 2.5 body lengths below the model. The Langley and Glenn wind tunnel measurements are generally in good agreement at 2.5 body lengths in Fig. 6-23, but the small differences of the two measurements gives an indication of the level of uncertainty in the measurements. The agreement between the wind tunnel and computed signatures is good, except in the region near $x/l = 0.8$. Other investigators[85,87] also showed a difference between wind tunnel and computed pressure signatures at $x/l = 0.8$. Both wind tunnel measurements agree favorably with each other near $x/l = 0.8$. The next section investigates model geometry sensitivity in this mismatch region.

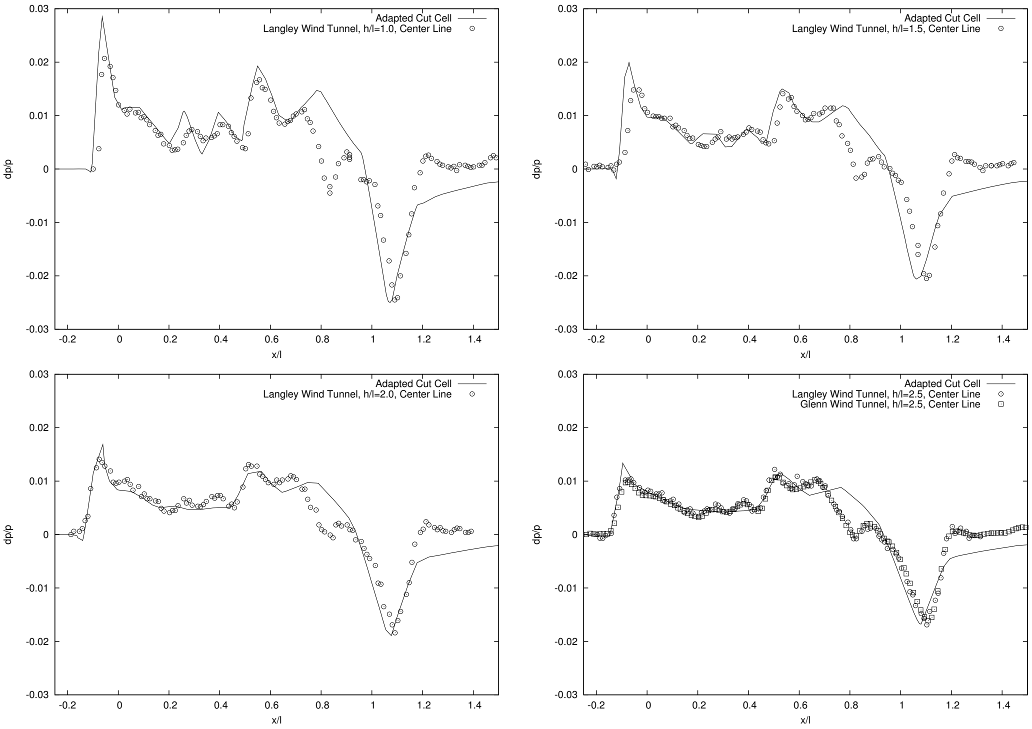


Figure 6-23: SLSLE centerline pressure signatures for various locations below the model.

Figure 6-24 compares the adapted cut-cell method with the Glenn wind tunnel measurement at 10 body lengths. This is the same distance as the integration surface. The shock strength increases away from the configuration centerline. The front portion of the signature is well predicted in both Fig. 6-23 and 6-24. The aft portion of the computed signal shows the largest difference from the wind tunnel data at all propagation distances. The discrepancy between the wind tunnel and computed signatures near $x/l = 0.8$ decreases for the signatures away from the model centerline. This is the first published CFD prediction of the off centerline signatures. The integration surface is an entire quarter cylinder. The grid size may be reduced for the same or better signature resolution if the output pressure integration surface is restricted to the wind tunnel data locations.

None of the previously reported simulations of this model provided off centerline signature comparisons at 10 body lengths. Preliminary body-fitted grid output-based adaptation results by Lee-Rausch et al.[87] targeted the centerline pressure signature

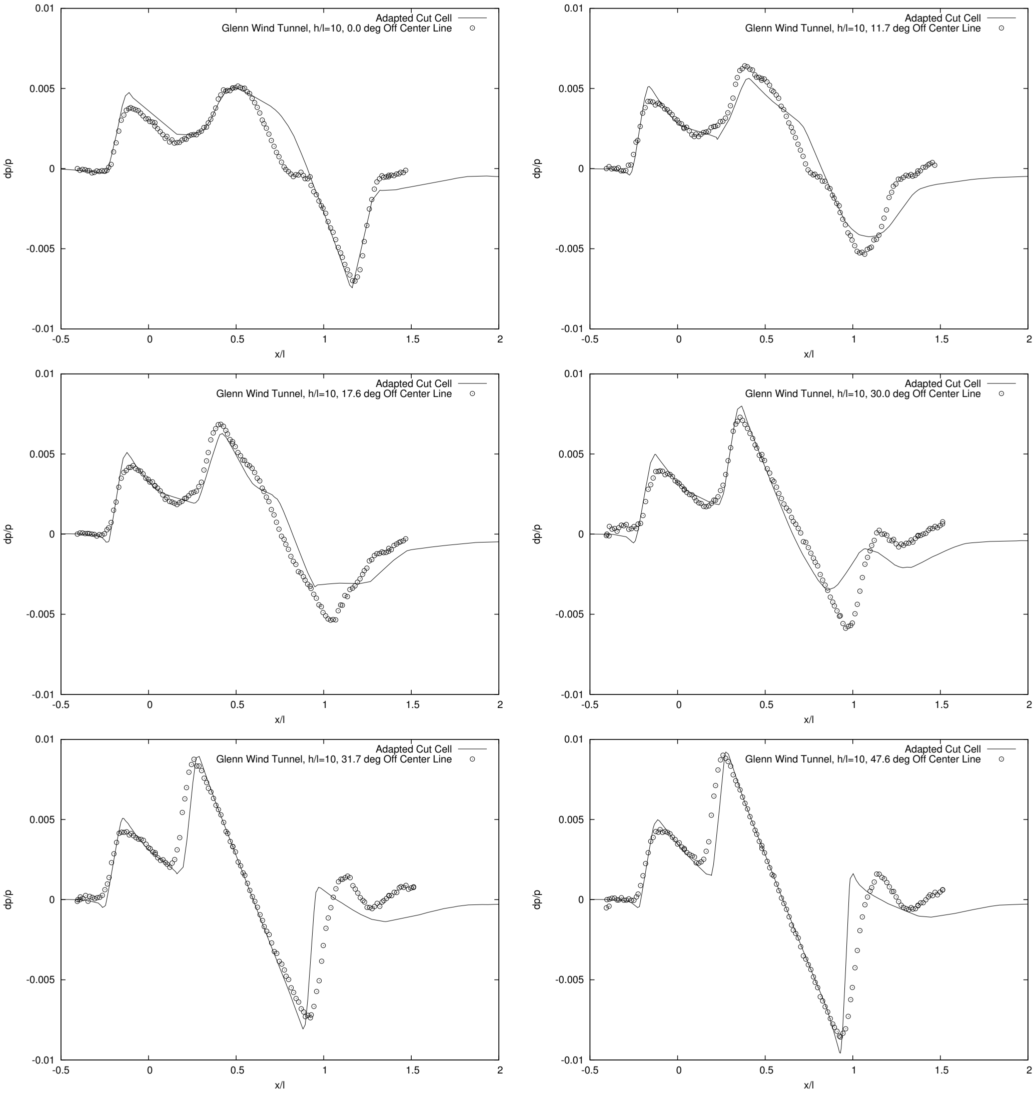


Figure 6-24: SLSLE pressure signatures for various locations 10 body lengths below the model.

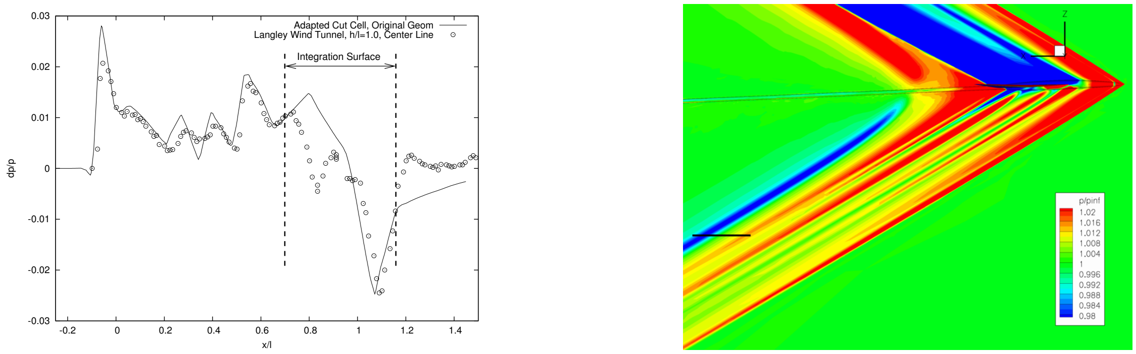
at 2.5 body lengths with a final grid of 2 million control volumes. Laflin, Klausmeyer, and Chaffin[85] used a hybrid method with an unstructured extreme-near-field grid of 1.25 million nodes adapted to propagate the signal less than 0.25 body lengths. A structured grid was employed to propagate the signal at 0.25 body length to the wing tunnel data at 2.5 body lengths. Carter and Deere[32] showed centerline comparisons at 10 body length, but the grid size was not reported.

6.5.4 Low-Boom Wing Body Signature Sensitivity and Scanned Geometry

The adjoint solution is used to drive the output-based adaptation process. It can also be employed to gain intuition for the output sensitivity of the problem. In this section, the adjoint solution will provide an indication of the sensitivity of the computed signal to the model geometry. After this sensitivity is presented, an alternative geometry is evaluated for its effect on the pressure signal.

To investigate the discrepancy in the aft portion of the predicted and measured signals, the output pressure integration surface is restricted to the region of mismatch at 1 body length. The final adapted grid from the previous section is reused with a different integration surface. This restricted region is shown in Fig. 6-25. The adapted pressure signature with the area of mismatch bracketed is shown in Fig. 6-25(a). Pressure on the symmetry plane is shown with a black line to illustrate the integration surface in Fig. 6-25(b).

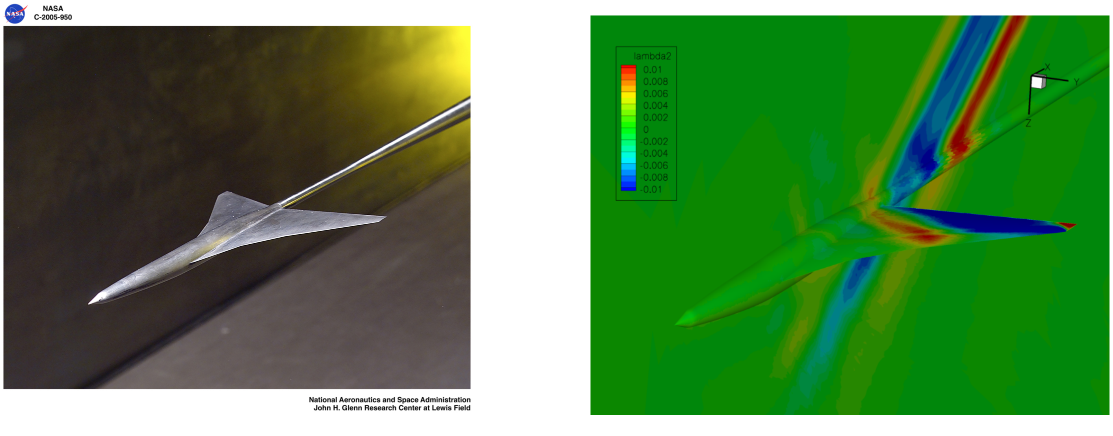
Figure 6-26(a) is a photograph of the model installed upside down in the Glenn wind tunnel. The x-momentum adjoint for the restricted integration surface is shown in Fig. 6-26(b). The x-momentum equation is affected by the model surface slope through the tangent flow boundary condition. Therefore, this adjoint solution approximates the linearized sensitivity of the pressure signature to surface slope. The adjoint solution follows the characteristics of the flow equations backward from the integration surface to the model surface. The pressure signal is most sensitive to the model geometry in the red and blue regions, and insensitive in the green region.



(a) Signal focus region.

(b) Symmetry plane pressure and integration surface (black line).

Figure 6-25: Integration surface focus region at 1 body length.



(a) Glenn Wind Tunnel model installation.

(b) x-momentum adjoint.

Figure 6-26: SLSLE tunnel installation and adjoint solution (upside down).

The original geometry used in this study was derived from the "as designed" CAD model. After the model was built, it was scanned to determine the actual "as built" geometry of the wind tunnel model. The scanned points were obtained at a sparse set of locations, and the "as designed" geometry was modified to interpolate the scanned points, creating the scanned geometry. Figure 6-27 shows the shaded underside of the scanned geometry. Its inset shows the shaded original geometry and a wire frame of the scanned surface geometry slightly aft of sting-body juncture. The sting diameter is larger for the "as built" geometry in the same region that is blue in Fig. 6-26(b). This indicates that the mismatch region of the signature is sensitive to the geometry slope in the location of greatest geometry difference.

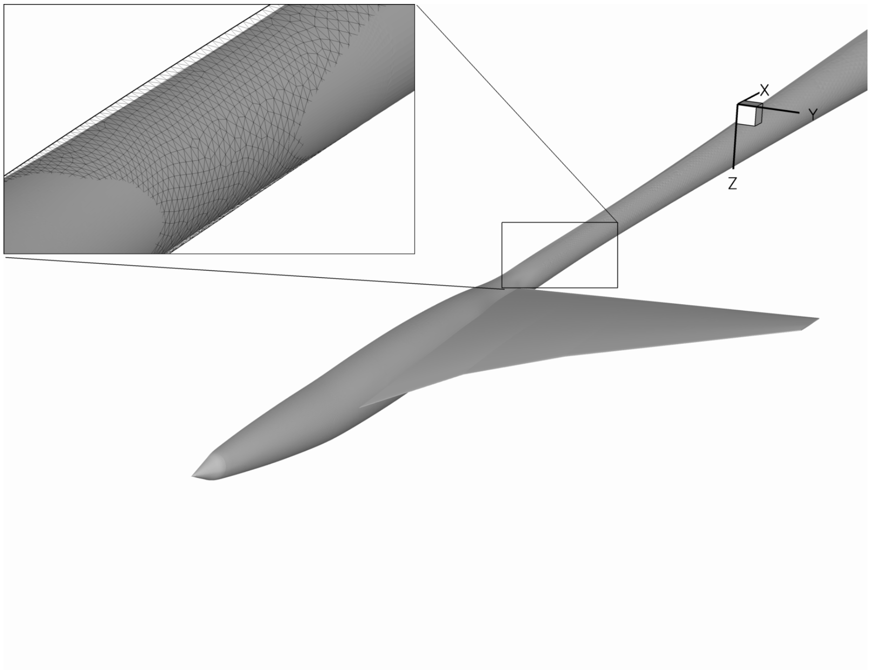


Figure 6-27: Comparison of the underside of two surface geometries (upside down).

The background grid, adapted to the original "as designed" geometry, is re-cut with the new "as built" geometry. This highlights the flexibility of the cut-cell method, the background grid and the surface discretizations are independent. The reuse of adapted background grids facilitates design by allowing for quick evaluation of perturbed surface geometries. A comparison of the pressure signature of the original and scanned geometry at one body length with the same background grid is shown in Fig. 6-28. The "as built" geometry signal shows the greatest change between $x/l = 0.7$ and $x/l = 0.9$, with an improved prediction of the inflection of the wind tunnel measured data at $x/l = 0.9$. The pressure recovery in the aft portion of the signal is also improved, which may be due to the "as built" wind tunnel sting having a conical shape, where the "as designed" has a cylinder. The symmetry planes of the background grid are shown colored with pressure for the two geometries in Fig. 6-29. The forward portion of the pressure field is very similar between the two geometries. The aft pressure field is changed by the differently shaped sting of the the new geometry.

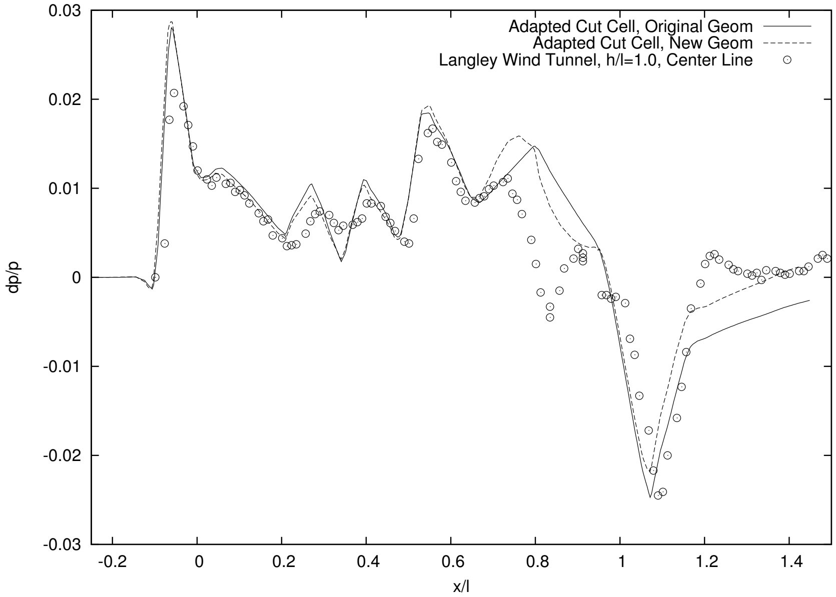
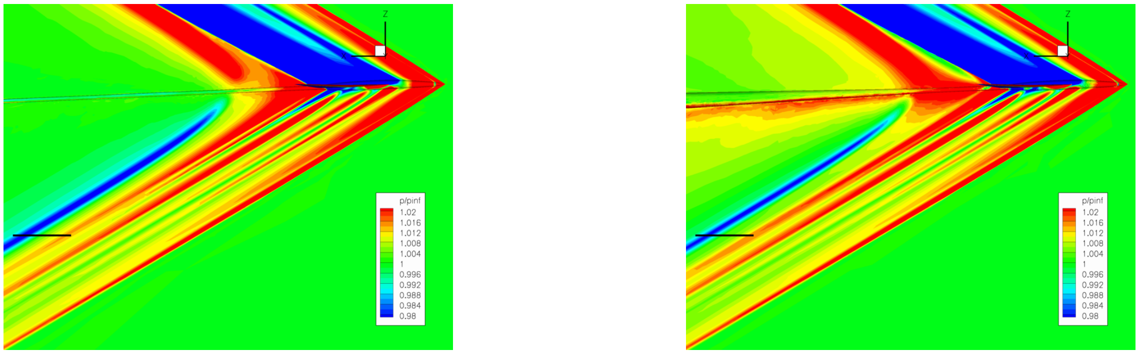


Figure 6-28: SLSLE pressure signatures for two geometries 1 body length below the model.



(a) Original “as designed” geometry.

(b) Scanned “as built” geometry.

Figure 6-29: Symmetry plane pressure and integration surface (black line).

Chapter 7

Direct Adaptation for Output Error

In Chapter 6, the adaptation algorithm employed an anisotropic element size request (metric) formed by combining a scalar output error estimate with the Hessian of a scalar field (Mach number). This approach has at least two limitations:

- Although the choice of Mach number for the Hessian has produced satisfactory results in the past, its selection as the Hessian scalar is arbitrary. For example, the intersection of the metrics obtained from all flow variables[33] may be superior.
- The adaptation metric is based on local interpolation error estimates, specifically, the departure of a linear and quadratic representation of the solution via a Hessian.[127] The Hessian method determines the dominate direction of the largest next higher order derivative for linear functions. For functions represented with higher order polynomials than linear, a search for this direction must be employed.[54] Creating a size request in the next higher order derivative direction requires an interpolation error estimate[127] or an *a priori* estimate of spatial error convergence.[54, 147]

In this Chapter, an adaptive method that directly controls the output error without an intermediate element size request is introduced. In particular, this proposed approach removes the limitations of the previous metric-based methods. This new method is first introduced through application to the control of interpolation error of a known analytic function in 1D, 2D, and 3D. Then this method is applied to output error control in the remainder of the chapter.

7.1 1D Analytic Function Adaptation Demonstration

The interpolation error is defined as,

$$e(\Omega) = \left(\int_{\Omega} |(\bar{u} - u)|^2 \, \mathrm{d}\Omega \right)^{1/2}, \tag{7.1}$$

where the exact analytical function u and interpolant $\bar{u}$ are defined in the domain Ω. The $\bar{u}$ is defined as a p-order polynomial in each element κ, which completely discretizes Ω. The u is sampled at each node of $\bar{u}$ to form $\bar{u}$ in each element. The elemental interpolation error is

$$e(\kappa) = \left(\int_{\kappa} |(\bar{u} - u)|^2 \, \mathrm{d}\kappa \right)^{1/2}. \tag{7.2}$$

The total interpolation error for the domain is the norm of the elemental errors,

$$e(\Omega) = \left(\sum_{\kappa \in \Omega} e(\kappa)^2 \right)^{1/2}. \tag{7.3}$$

Following Zienkiewicz and Zhu,[157] the goal is to equidistribute this error estimate. An equal amount of error for each element is

$$\mathrm{tol}_{\kappa} = \left(\frac{(\mathrm{tol}_{\Omega})^2}{N} \right)^{1/2}, \tag{7.4}$$

where tol_{Ω} is the requested total interpolation error and N is the number of elements. An iteration of an adaptive method that directly controls interpolation error is defined as:

- Repeat until all $e(\kappa) \leq \text{tol}_\kappa$:
  - Evaluate Eq. (7.2) for all elements in the grid; split the element in half with the largest value of $e(\kappa)$ if $e(\kappa) > \text{tol}_\kappa$,

    $$Sp(\mathcal{X}_n, \mathcal{T}_n, j, e()) \ \forall \ j \in \mathcal{T}_n : e(\kappa_j) > \text{tol}_\kappa \text{ from largest } e(\kappa_j) \text{ to smallest } e(\kappa_j). \tag{7.5}$$

  - Evaluate Eq. (7.2) for all elements in the grid; merge the element with the smallest value of e_κ with a neighbor,

    $$C(\mathcal{X}_n, \mathcal{T}_n, j, e(), \text{tol}_\kappa) \ \forall \ l \in \mathcal{T}_n : e(\kappa_j) < \text{tol}_\kappa \text{ from smallest } e(\kappa_j) \text{ to largest } e(\kappa_j). \tag{7.6}$$

- Perform two sweeps of node position optimization; each sweep progresses from the node of the element with the largest $e(x_i)$ to the nodes of element with the smallest $e(x_i)$,

  $$Mo(\mathcal{X}_n, \mathcal{T}_n, i, e()) \ \forall \ i \in \mathcal{X}_n \text{ from largest } e(x_i) \text{ to smallest } e(x_i). \tag{7.7}$$

- Exit adaptive procedure if $e(\Omega) \leq \text{tol}_\Omega$.

A tanh exact function u is defined in the 1D region $[0, 1]$,

$$u = \tanh(50(x - 0.4)). \tag{7.8}$$

For a metric-based approach, a requested mesh size is established from a scaled interpolation estimate[127] commonly used for anisotropic adaptation,

$$h = \min\left(0.5, \frac{1}{\sqrt{100|u_{xx}|}}\right). \tag{7.9}$$

The min function is used to set a maximum element size of 0.5 when u_{xx} approaches zero. The scaling factor 100 is chosen arbitrarily to produce a reasonably-sized mesh of 32 elements. A metric-based grid is generated that satisfies this h-request with a curve discretization method.[126] This metric-based grid is shown in Fig. 7-1(a). Element boundaries are marked with a +-symbol. The metric-based grid has a total interpolation error of $e(\Omega) = 1.12 \times 10^{-3}$ with 32 $p = 1$ elements.

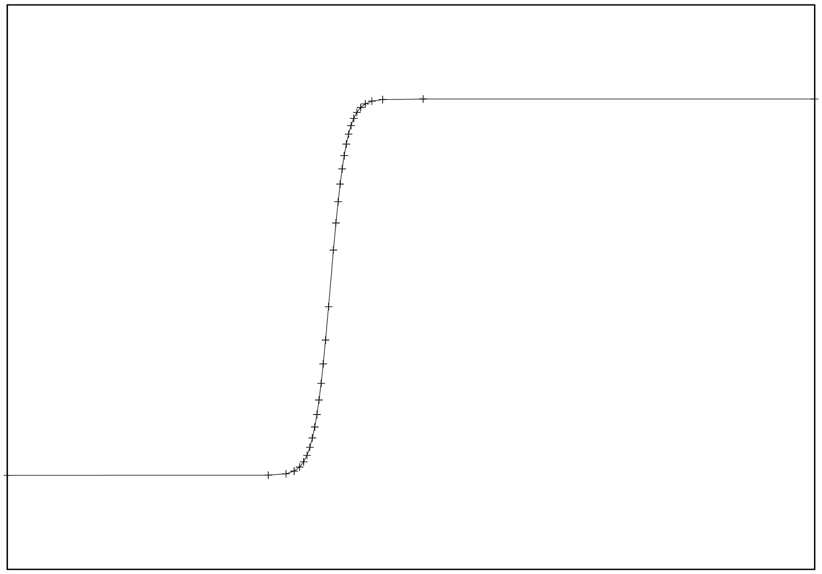
The direct adaptive method is invoked with a error tolerance equal to the computed interpolation error of the metric-based grid. The final directly adapted grid is shown in Fig. 7-1(b). It has 27 $p = 1$ elements and a total error of $e(\Omega) = 1.06 \times 10^{-3}$, which is slightly lower than the metric-based grid, 1.12×10^{-3}, which has more elements. The metric-based, Fig. 7-1(a), and direct error control, Fig. 7-1(b), grids are very similar for this 1D example with $p = 1$ elements. The error estimate[127] is sharp for this simple 1D case. The clustering of elements in the high-curvature regions of the tanh function is evident for both methods. The direct adaption method produced a grid with slightly lower interpolation error and a reduction of the number of elements.

The metric-based approach is repeated for a sequence of α with the specified

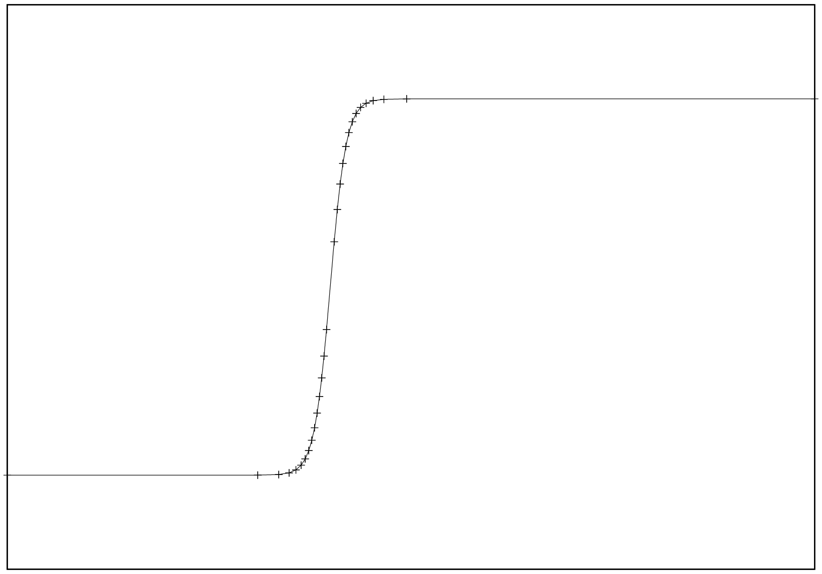
$$h = \min\left(0.5, \frac{1}{\sqrt{\alpha |u_{xx}|}}\right). \tag{7.10}$$

The 2-norm of domain interpolation error for this sequence grids is shown as the squares in Fig. 7-2 as a function of effective $h = 1/N$. The direct interpolation adaption method is invoked multiple times with a tol_Ω equal to the $e(\Omega)$ of the metric-based approach. The $e(\Omega)$ for these directly adapted grids is shown as circles in Fig. 7-2. Both methods show second-order convergence of the interpolation norm. The direct approach is more efficient in terms of error for an effective h over the entire sequence of grids. The direct method has 0.7 the error of the metric approach for the same number of elements.

To investigate the optimization scheme of direct adaptation, the node movement operator Mo is omitted from the adaptation cycle. The direct approach with and without the Mo operator is shown in Fig. 7-3. The direct approach with the Mo operator is shown with the circle symbols. The direct approach without the Mo operator is shown with the triangle symbols. The direct approach without the Mo is less efficient in terms of interpolation error for a given number of elements. Without



(a) Metric-based, 32 $p = 1$ elements, $e(\Omega) = 1.12 \times 10^{-3}$.



(b) Direct error control, 27 $p = 1$ elements, $\text{tol}_\Omega = 1.12 \times 10^{-3}$, $e(\Omega) = 1.06 \times 10^{-3}$.

Figure 7-1: Grids to control the interpolation error of tanh with $p = 1$ basis.

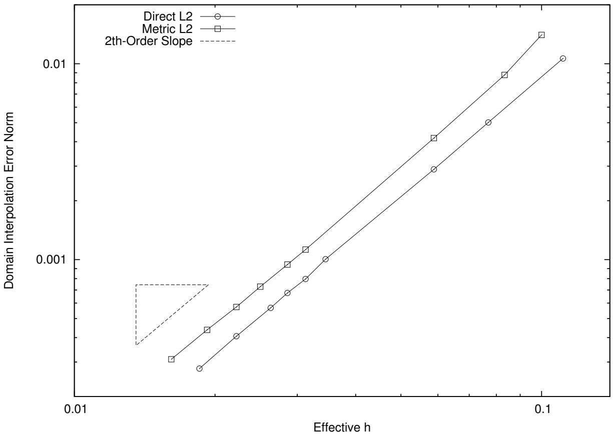


Figure 7-2: Convergence of the interpolation error in the 2-norm for the metric and direct adaptation method on the tanh function with $p = 1$ elements.

the Mo operator, the error convergence is less smooth, because the optimization can only use discrete node locations instead of continuously optimizing these locations.

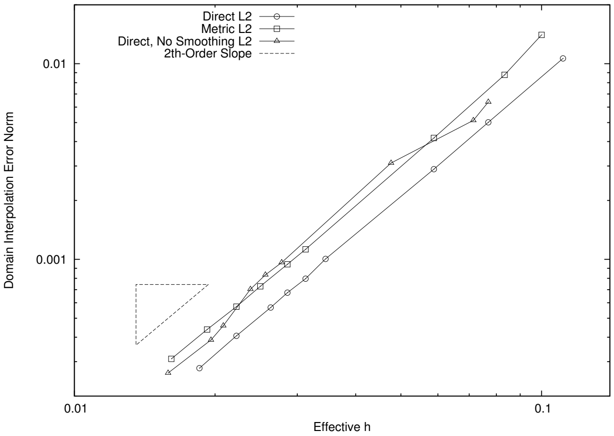


Figure 7-3: Convergence of the interpolation error in the 2-norm for the direct adaptation method on the tanh function with and without the Mo operator with $p = 1$ elements.

An advantage of the direct adaption method is that the order of the polynomial representation can be increased without modifying the adaptation algorithm. The h specification based on interpolation estimates has been formulated for $p = 1$ elements. In 1D, an h can be specified for higher degree polynomials, but this task becomes more complicated for multiple dimensions.[54] The 2-norm of domain interpolation error for a sequence grids is shown in Fig. 7-4 as a function of effective $h = 1/N$ for various p. Triangles are provided with slopes listed on the left of each triangle. These indicate that the error is converging at an optimal rate of $p + 1$.

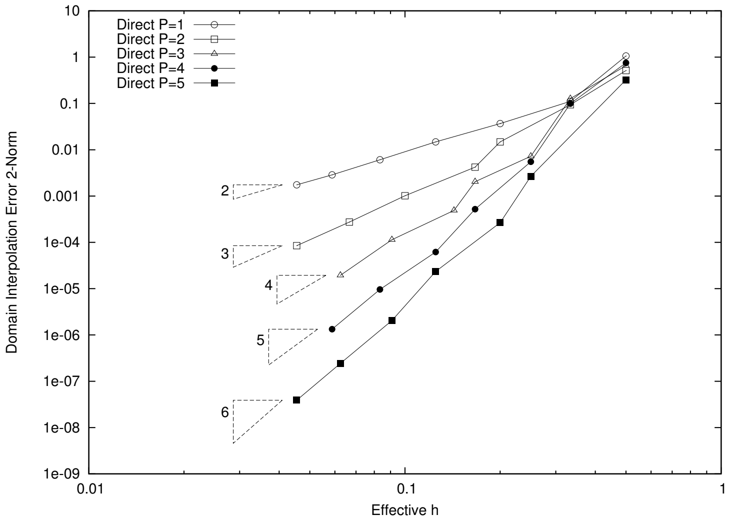


Figure 7-4: Convergence of the interpolation error in the 2-norm for the direct adaptation method on the tanh function with $p = 1, 2, 3, 4, 5$ elements.

The 1D error adaptation example is repeated for $p = 5$ elements, Fig. 7-5. Typically, u is an unknown solution (i.e., of a partial differential equation (PDE)). In this example, an approximate solution $\tilde{u}$ is assumed to be available on a set of discrete elements κ_0, where $\tilde{u}$ is a set of p-th order polynomials. Although the exact solution is available for this case, a $\tilde{u}$ will be constructed by fitting the p-order Lagrangian basis that minimizes

$$\left(\int_{\kappa_0} |\tilde{u} - u|^2 \, \mathrm{d}\kappa_0 \right)^{1/2} \tag{7.11}$$

independently in each element. The left column of Fig. 7-5 shows this discontinuous fit for a series of adapted 1D grids.

To estimate the leading order terms of the exact interpolation error, a $p+1$-order solution $\hat{u}$ is reconstructed from $\tilde{u}$ on κ_0. A $\mathcal{C}^0$ $p+1$-order function $\hat{u}$ is formed from the discontinuous $\tilde{u}$ by minimizing

$$\left(\int_{\Omega} |\hat{u} - \tilde{u}|^2 \, \mathrm{d}\Omega \right)^{1/2} \tag{7.12}$$

over the entire domain. The formation of $\hat{u}$ is not local; an inexpensive mass matrix

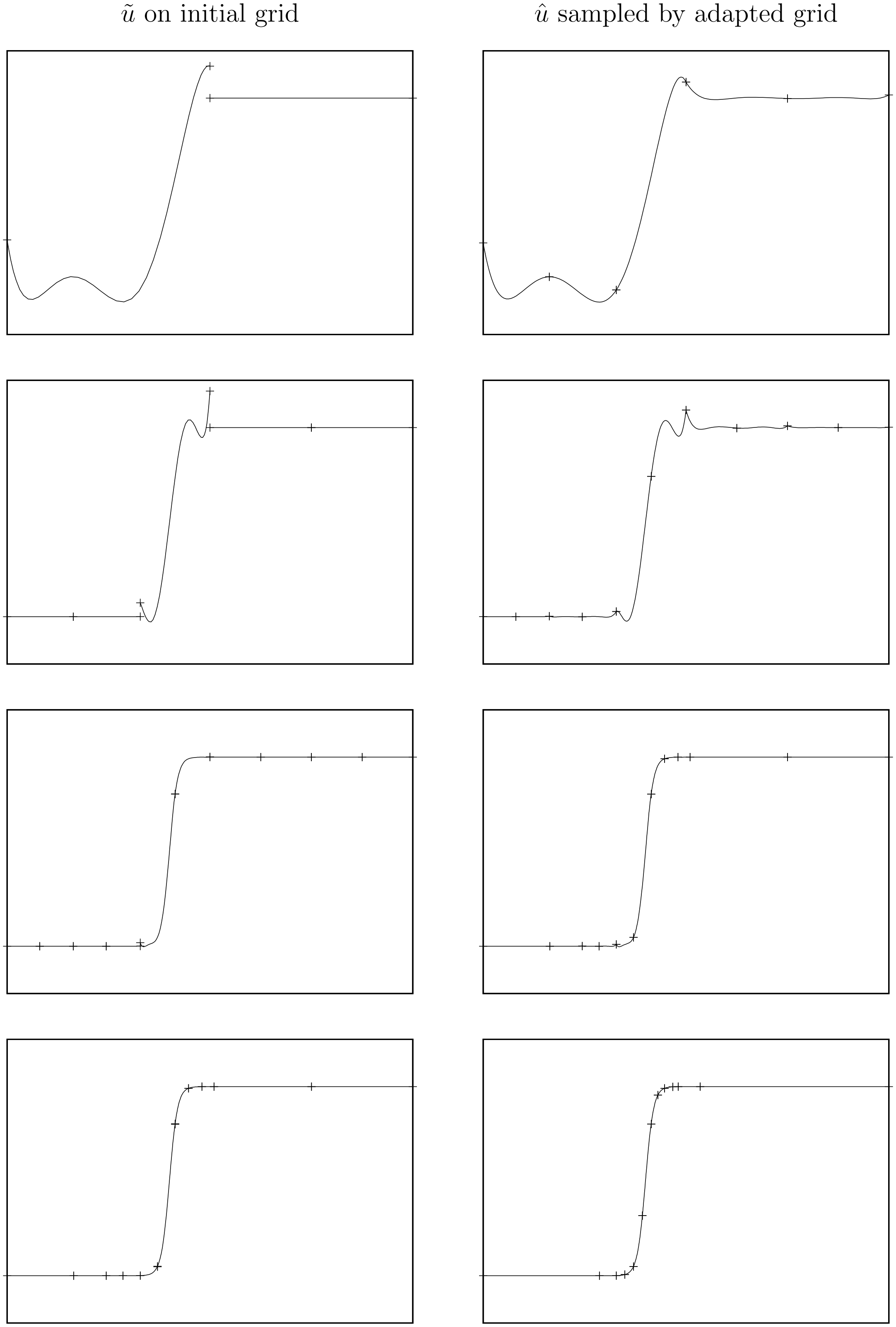


Figure 7-5: Four 1D adaptation cycles to the interpolation error of tanh with $p = 5$ basis (one cycle per row, approximate solution in left column, reconstructed solution in right column)

inversion is computed. A $\mathcal{C}^0$ $\hat{u}$ is required to prevent the adaptation process from introducing zero width elements at discontinuous function jumps. This $\hat{u}$ is shown in the right column of Fig. 7-5. Over- and under-shoots are present in $\hat{u}$. The domain discretization is adapted into elements κ. An elemental interpolation error of the form,

$$e(\kappa) = \left(\int_\kappa |(\bar{u} - \hat{u})|^2 \, \mathrm{d}\kappa \right)^{1/2}, \tag{7.13}$$

is utilized, where $\bar{u}$ is a p-th order polynomial representation of the function $\hat{u}$ in each element. The function $\bar{u}$ is directly interpolated from $\hat{u}$. A step of the adaptive procedure follows;

- Given the current grid, construct a discontinuous p basis $\tilde{u}$ with Eq. (7.11) to simulate the solution to a PDE.
- Reconstruct a continuous $p+1$ basis $\hat{u}$ with a 2-norm fit of $\tilde{u}$ over the domain, Eq. (7.12).
- Exit adaptive procedure if $e(\Omega) \leq \mathrm{tol}_\Omega$.
- Repeat until all $e(\kappa) \leq \mathrm{tol}_\kappa$:
  - Evaluate Eq. (7.13) for all elements in the grid; split the element in half with the largest value of $e(\kappa)$ if $e(\kappa) > \mathrm{tol}_\kappa$,

    $$Sp(\mathcal{X}_n, \mathcal{T}_n, j, e()) \;\forall\; j \in \mathcal{T}_n : e(\kappa_j) > \mathrm{tol}_\kappa \text{ from largest } e(\kappa_j) \text{ to smallest } e(\kappa_j). \tag{7.14}$$

  - Evaluate Eq. (7.13) for all elements in the grid; merge the element with the smallest value of e_κ with a neighbor,

    $$C(\mathcal{X}_n, \mathcal{T}_n, j, e(), \mathrm{tol}_\kappa) \;\forall\; l \in \mathcal{T}_n : e(\kappa_j) < \mathrm{tol}_\kappa \text{ from smallest } e(\kappa_j) \text{ to largest } e(\kappa_j). \tag{7.15}$$

- Perform two sweeps of node position optimization; each sweep progresses from the node of the element with the largest $e(x_i)$ to the nodes of element with the smallest $e(x_i)$,

$$Mo(\mathcal{X}_n, \mathcal{T}_n, i, e()) \; \forall \; i \in \mathcal{X}_n \text{ from largest } e(x_i) \text{ to smallest } e(x_i). \tag{7.16}$$

These steps are repeated to form the adaptive procedure.

Each step of the adaptive procedure is shown as a row in Fig. 7-5. The upper left subfigure shows $\tilde{u}$ on the initial two element grid. Element boundaries are marked with a + symbol. The inter-element jump is evident for this discontinuous $p = 5$ Lagrangian basis on the initial grid. The upper right subfigure shows the first adapted grid interpolating the $p = 6$ continuous Lagrangian basis $\hat{u}$ on κ_0. The second row of subfigures exhibits the Gibbs phenomenon near the rise in the tanh function. This phenomenon is damped in the third and fourth rows as adaptation progresses. The final grid is coarsened away from $x = 0.4$ after the Gibbs phenomenon is reduced.

7.2 Direct Control of 2D Scalar Interpolation Error

In this section, the direct error control approach is extended to 2D by considering adaptation to minimize the 2-norm of interpolation error of a known function. The direct adaptation consists of four modes. These are element swapping, node movement, element collapse, and element split.

- Sort the elements from largest e_κ to smallest. Perform the swap operation on all element with $e(\kappa) > \text{tol}_\kappa$,

$$Sw(\mathcal{X}_n, \mathcal{T}_n, j, e()) \; \forall \; j \in \mathcal{T}_n : e(\kappa_j) > \text{tol}_\kappa \text{ from largest } e(\kappa_j) \text{ to smallest } e(\kappa_j). \tag{7.17}$$

- Perform the node movement operation on all nodes from largest,

$$e(x_i) = \left(\sum_{\kappa \ni x_i} \frac{e(\kappa)^2}{V(\kappa)} \right)^{1/2}, \tag{7.18}$$

over incident elements to smallest,

$$Mo(\mathcal{X}_n, \mathcal{T}_n, i, e) \; \forall \; i \in \mathcal{X}_n \text{ from largest } e(x_i) \text{ to smallest } e(x_i). \tag{7.19}$$

The volume (area in 2D) of the elements $V(\kappa)$ is included in the denominator to penalize nearly degenerate elements. A simplex search is performed on the prioritized node list to optimize the inverse volume weighted norm.

- The elements with low error are examined for potential collapse,

$$C(\mathcal{X}_n, \mathcal{T}_n, j, e(), \text{tol}_\kappa) \; \forall \; j \in \mathcal{T}_n : e(\kappa_j) < \text{tol}_\kappa \text{ from smallest } e(\kappa_j) \text{ to largest } e(\kappa_j). \tag{7.20}$$

- Sort the elements from largest $e(\kappa)$ to smallest,

$$Sp(\mathcal{X}_n, \mathcal{T}_n, j, e()) \; \forall \; j \in \mathcal{T}_n : e(\kappa_j) > \text{tol}_\kappa \text{ from largest } e(\kappa_j) \text{ to smallest } e(\kappa_j). \tag{7.21}$$

- The adaptation process repeats until $e(\Omega) < \text{tol}_\Omega$.

An analytic function is defined,

$$u = (2.0 + \sin(10.0x)) \exp(-10.0(y - 0.5)^2), \tag{7.22}$$

that mimics the scalar convection diffusion solution in Section 7.4. The function is shown in Fig. 7-6 for a domain sized $[-1.5, 1.5] \times [0, 1]$. Direct interpolation error control and metric-based algorithms are employed with an initial grid shown in Fig. 7-7.

The interpolation error is between $p = 2$ and $p = 1$ elements. The target metric M is set to $10|H|$ for the entire adaptation process, where the Hessian,

$$H = \begin{bmatrix} u_{xx} & u_{xy} \\ u_{xy} & u_{yy} \end{bmatrix}, \tag{7.23}$$

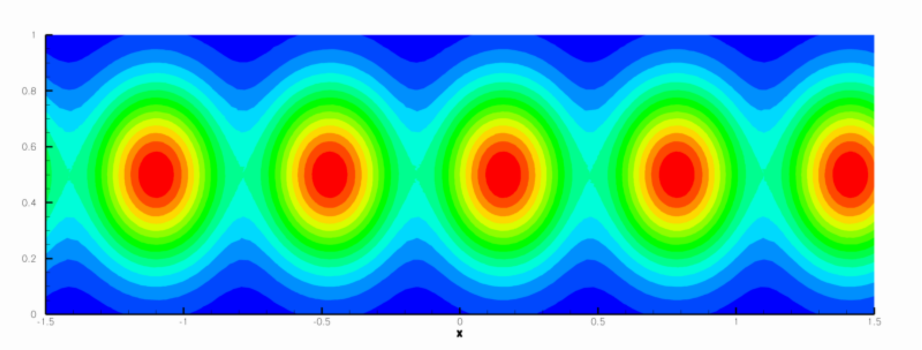


Figure 7-6: Analytic function, Eq. (7.22).

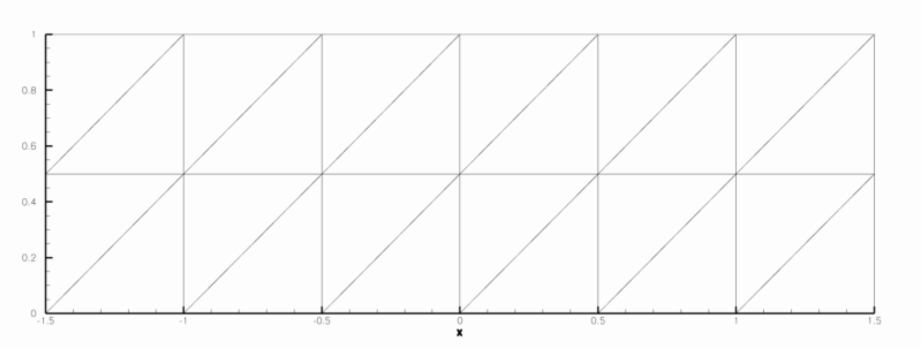


Figure 7-7: Initial grid of the domain $[-1.5, 1.5] \times [0, 1]$.

is computed analytically. The convergence of the interpolation error in the 2-norm for the metric and direct adaptation methods is shown in Fig. 7-8. The metric-based approach employed the algorithm from Section 6.2. The metric M is interpolated from a grid with quadratic elements that is frozen at the start of each iteration. The direct approach error tolerance is set to half the initial error norm at each adaptation cycle. The direct approach is terminated when it reached the error level of the metric-adapted grid. The direct approach utilizes the reconstructed solution $\hat{u}$ described in Section 7.1 to define the interpolation error $e(\kappa)$.

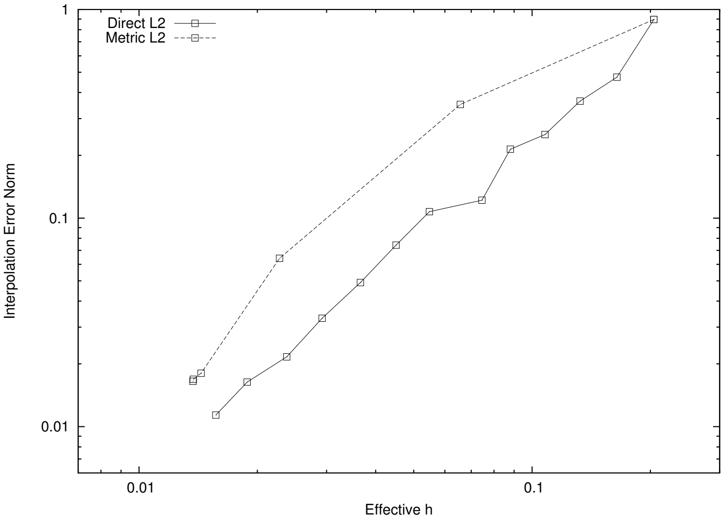


Figure 7-8: Convergence of the interpolation error in the 2-norm for the metric and direct adaptation methods.

The final metric-adapted grid is shown in Fig. 7-9 with the 2-norm interpolation error for each element. The direct-adapted grid is shown in Fig. 7-10 with the 2-norm interpolation error for each element. The direct-adapted element error is scattered with the largest values on the border of the domain and radially around each peak in the function. The metric-adapted element error is clustered in the saddle-shaped troughs of the function. The metric-adapted grid is much more regular than the direct-adapted grid. Both methods produce similar grids at the peaks of the functions, where the grid is isotropic. The direct-adapted grid has greater anisotropy radiating from the peaks and along the troughs of the function. The direct approach uses less triangles than the metric-based approach for the same error norm for the final and

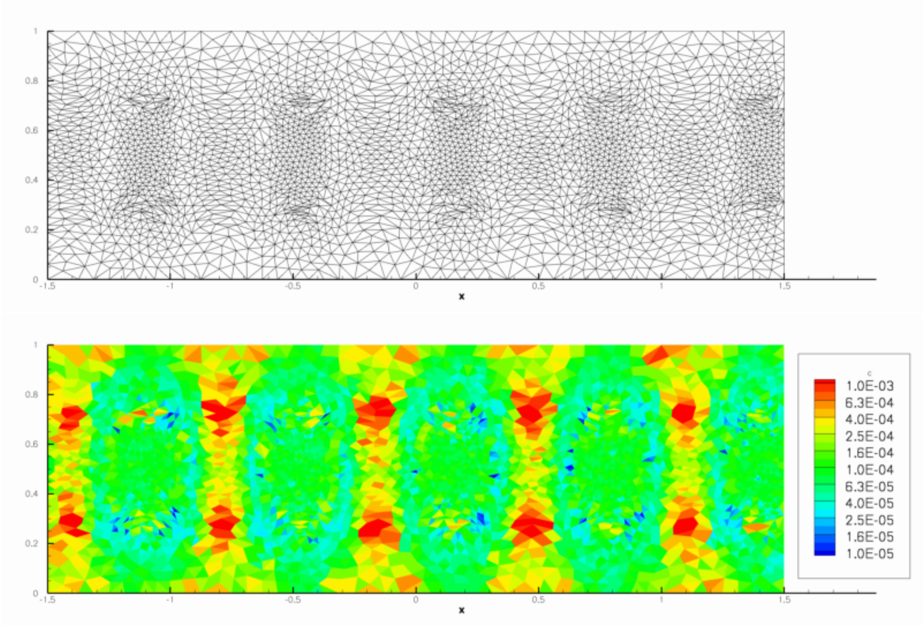


Figure 7-9: Metric-adapted grid and element interpolation error in 2-norm (color scale is logarithmic).

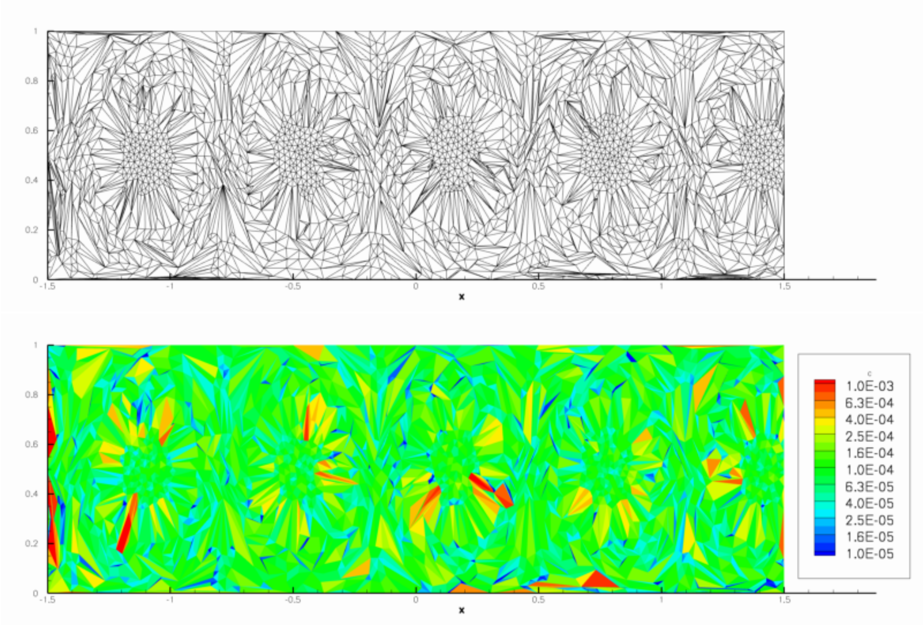


Figure 7-10: Direct interpolation error adapted grid and element interpolation error in 2-norm (color scale is logarithmic).

intermediate grids.

The metric-based approach is intended to equidistribute an *a priori* estimate of the interpolation error in the 2-norm.[126] The direct approach may be more efficient in terms of number of elements because it is employing an actual calculation of the error. The direct-adapted grids may be less regular because computed error is a more difficult function to optimize than metric conformity.

The metric-based approach is repeated for a sequence of α with the specified

$$M = \alpha |H|. \tag{7.24}$$

The 2-norm of domain interpolation error for this sequence grids is shown as the squares in Fig. 7-11 as a function of the effective $h = 1/\sqrt{N}$. The $e(\Omega)$ for the directly adapted grids is shown as circles in Fig. 7-11. This is the same sequence of grids shown in Fig. 7-8, which was produced by halving the initial computed error to set tol_Ω for each cycle. Both methods show second-order convergence of the interpolation norm in terms of an effective h. The direct approach is more efficient in terms of error for a given number of elements for the entire sequence of grids. The improvement in efficiency is a constant factor for this entire sequence.

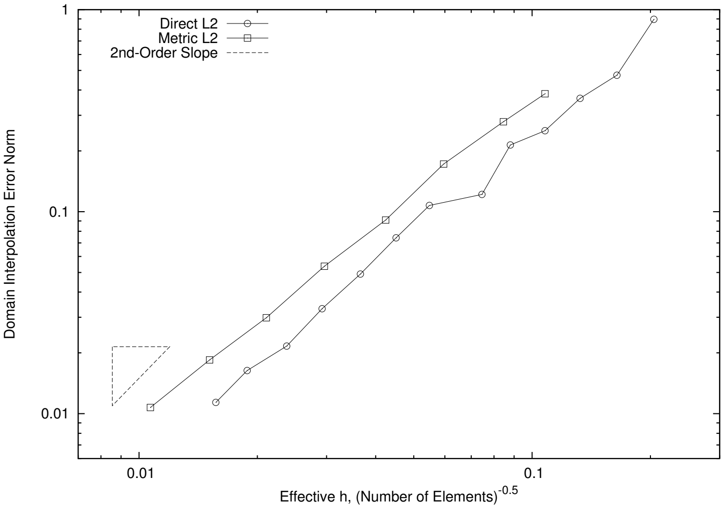


Figure 7-11: Convergence of the interpolation error in the 2-norm for the 2D metric and direct adaptation method.

The optimization methodology of the direct adaptation method is investigated by omitting the Mo operator and repeating the adaptive sequence. The interpolation error as a function of the number of elements and an effective length scale are shown in Fig. 7-12. The finest grids have a very similar efficiency, but the optimization with the Mo operator is more efficient for the majority of the coarser grids. There may be an opportunity to improve the Mo operator because it does not provide an improvement for the finer grids.

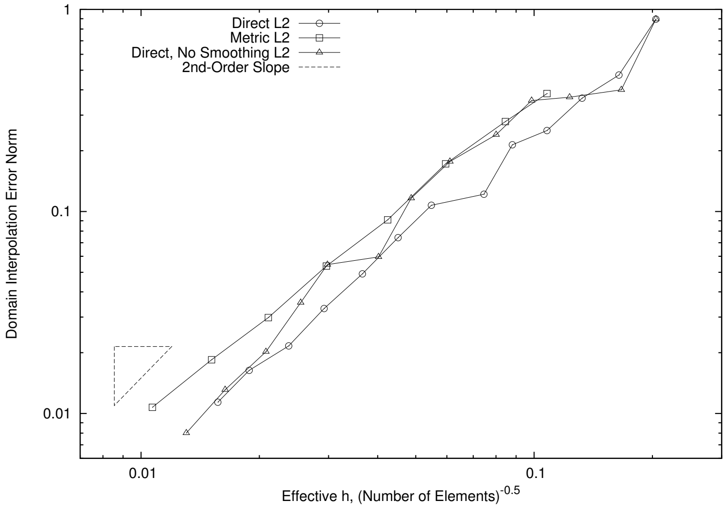


Figure 7-12: Convergence of the interpolation error in the 2-norm for the direct adaptation method with and without the Mo operator.

7.3 3D Analytic Function Adaptation Example

A cubic domain is employed in an 3D adaptation example to directly control interpolation error. The exact function for this example is

$$u = x^2 + 1000y^2 + 10z^3, \tag{7.25}$$

Fig. 7-13, used to define the interpolation error $e(\kappa)$. Approximate and reconstructed solutions are not used for this 3D case; the exact function is evaluated. The function is dominated by the $1000y^2$ term, as seen in the nearly planar contours. The direct

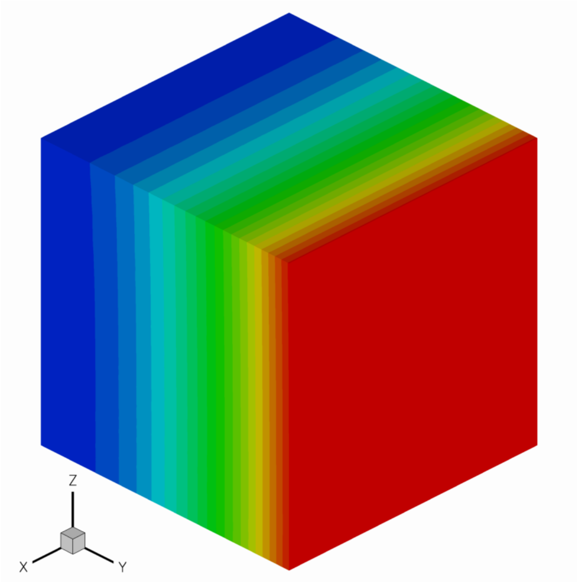
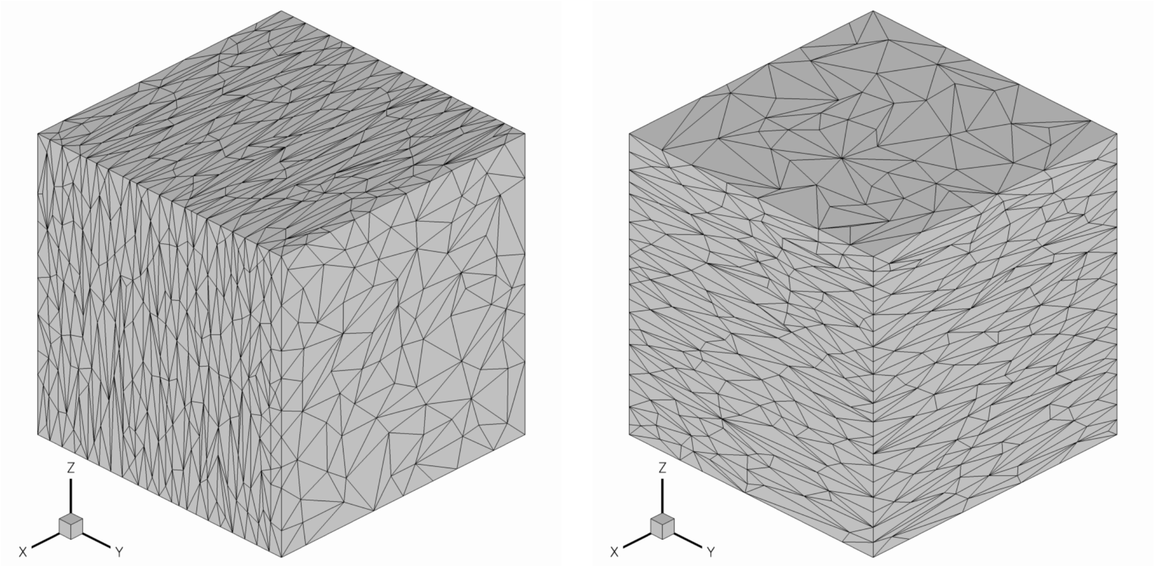


Figure 7-13: Exact $x^2 + 1000y^2 + 10z^3$ function in a cube.

adaptation algorithm described in Section 7.2 is utilized. The initial grid is shown in Fig. 6-1(a). The grid adapted to $\text{tol}_\Omega = 5.0 \times 10^{-1}$ for $p = 1$ tetrahedra is shown in Fig. 7-14(a). The grid adapted to $\text{tol}_\Omega = 5.0 \times 10^{-5}$ for $p = 2$ tetrahedra is shown in Fig. 7-14(b). For both cases the final interpolation error norm is approximately half the requested error norm. An anisotropic metric is not employed for this case; interpolation error is directly computed and controlled. The anisotropy seen in Fig. 7-14 is a result of efficiently resolving the anisotropic function. The $p = 1$ basis adapts to anisotropically resolve the $1000y^2$ term. The $p = 2$ basis resolves the quadratic term, $1000y^2$, exactly, so it anisotropically resolves the $10z^3$ term. The shift in the primary anisotropic direction is clearly evident as the interpolation error is directly controlled.

7.4 Direct Control of 2D Scalar Convection-Diffusion Output Error

The same direct interpolation error control procedure illustrated in this chapter is used to directly control the output error of a scalar convection-diffusion example.



(a) Adapted $p = 1$ to $\text{tol}_\Omega = 5.0 \times 10^{-1}$, actual $e_\Omega = 2.5 \times 10^{-1}$.

(b) Adapted $p = 2$ to $\text{tol}_\Omega = 5.0 \times 10^{-5}$, actual $e_\Omega = 2.0 \times 10^{-5}$.

Figure 7-14: Isometric views of an interpolation error adapted cube.

The steady, 2D convection-diffusion equation,

$$\nabla(Uu) = \frac{1}{\text{Pe}}\nabla^2 u + S, \tag{7.26}$$

is solved for the scalar u, where U is a fixed velocity, Pe is the Péclet number, and S is a source term. The Project X discontinuous Galerkin code[55] is employed to drive this equation to steady state providing an approximate solution $\tilde{u}$ and its discrete adjoint $\tilde{\lambda}$. The Project X website, `http://acdl.mit.edu/projectx.html`, contains a current list of references to its formulation. The diffusion operator stabilization parameter is set to $\eta_f = 3$, for all faces in the grid.[55]

The goal of output adaptation for this discontinuous Galerkin case is to control the error indicator utilized by Fidkowski and Darmofal,[54]

$$e_\kappa = \frac{1}{2}\left(|\mathcal{R}(\bar{u}, (\hat{\lambda} - \bar{\lambda}))| + |\mathcal{R}^\lambda(\bar{u}; (\hat{u} - \bar{u}), \bar{\lambda})|\right), \tag{7.27}$$

Where $\bar{u}$ and $\bar{\lambda}$ are the p-order interpolants and $\hat{u}$ and $\hat{\lambda}$ are the $p+1$-order reconstructions. This error term is evaluated on the adapted grid as grid modification decisions are made.

The reconstructed solution $\hat{u}$ employed in output error adaptation is formed by inverting the mass matrix of an order-$p+1$ continuous Galerkin basis to fit the order-p $\tilde{u}$ computed by the discontinuous Galerkin method. The mass matrix inversion is computed with a conjugate gradient iterative scheme (the mass matrix is symmetric positive definite).

The domain is sized $[-1.5, 1.5] \times [0, 1]$. The left, upper, and lower boundaries of the domain have Dirichlet boundary conditions with $u = e^{-10.0(y-0.5)^2}$. The source term is $S = sin(10.0x)$. The right boundary state is obtained from the interior. A uniform velocity field is established in the positive x-direction, $U = [1, 0]^T$ and Pe $= 1000.0$. The output is heat flux integrated over the lower boundary of the domain. The primal and dual solutions are shown in Fig. 7-15 for a fine grid using a $p = 3$ basis. The effect of the source term is manifested as x-oscillations of the primal solution. This source term does not affect the dual solution. The heat flux cost function along the lower boundary creates a dual solution with a boundary layer behavior. The source term creates a primal solution with a very different character than the dual solution. The convergence of heat flux for a series of uniformly refined grids is shown in Fig. 7-16 for first-, second-, and third-order polynomials. The truth value of heat flux, 0.0080361705, is computed using Richardson extrapolation from the finest two $p = 3$ outputs, assuming 4th-order spatial convergence.

The grid is adapted to improve the calculation of heat flux. The adaptive grid heat flux convergence is shown in Fig. 7-17 with the uniformly refined grid heat flux for $p = 1$ and an existing metric-based method[54] that utilizes BAMG.[70] The direct approach provided equivalent accuracy on the final grid with a sixth of the degrees of freedom.

The final adapted grids for the direct and metric-based method are shown in Fig. 7-18. A detail of the lower boundary of the domain is provided in Fig. 7-19 to illustrate the differences in the metric-based and direct approaches. The metric-based approach only utilizes the anisotropy of the primal problem, which can have a very different character than the dual solution. The direct approach implicitly mixes the anisotropy of the primal and dual solution depending on their impact on output error. By only considering the primal anisotropy, the metric-based approach is forced to reduce the size of the inappropriately stretched elements to resolve the dual solution errors. The metric-based approach produces a more regular grid. The direct approach places more refinement in the upper half of the domain. The direct approach is applied to the primal and dual portions of the output error separately in Appendix B.

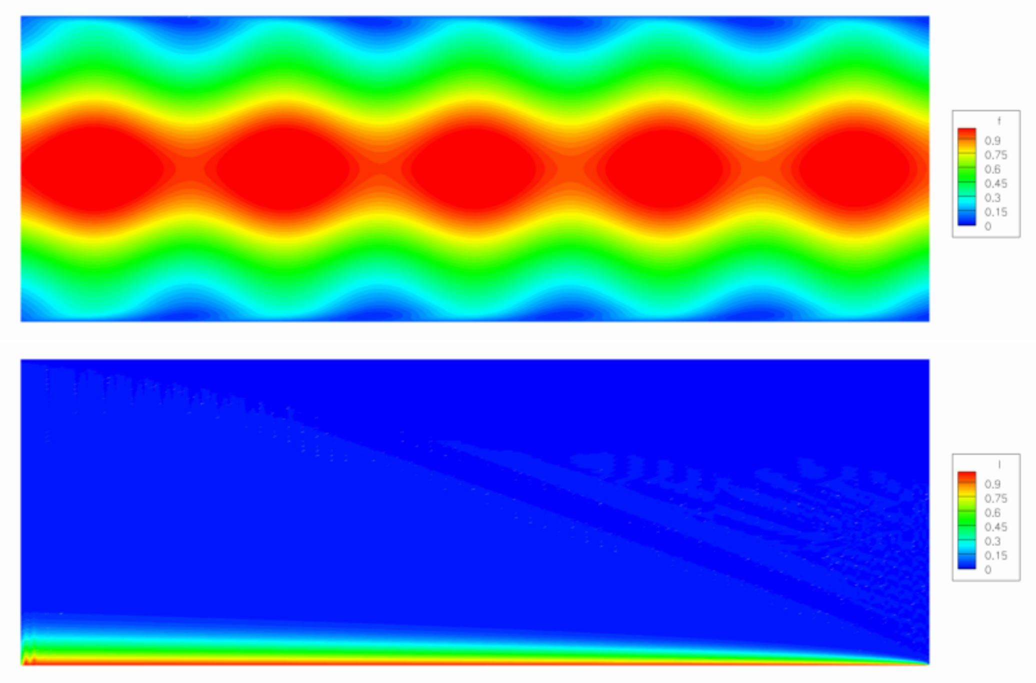


Figure 7-15: Scalar convection-diffusion solution and adjoint.

The final metric-adapted grid is shown in Fig. 7-21(a) with the error indicator Eq. (7.27) for each element. The final direct-adapted grid is shown in Fig. 7-21(b). An intermediate metric-adapted grid with a comparable number of elements to the final direct-adapted grid is shown in Fig. 7-21(c). The color scale is logarithmic. The metric-adapted element error is clustered in the upper and right portions of the domain. The direct-adapted grid has the largest error levels in irregular, high aspect ratio regions in the interior of the domain. The metric-based approach is excessively refining the lower region of the grid, possibly due to inappropriate element stretching. These results have similar characteristics to the 2D interpolation error control

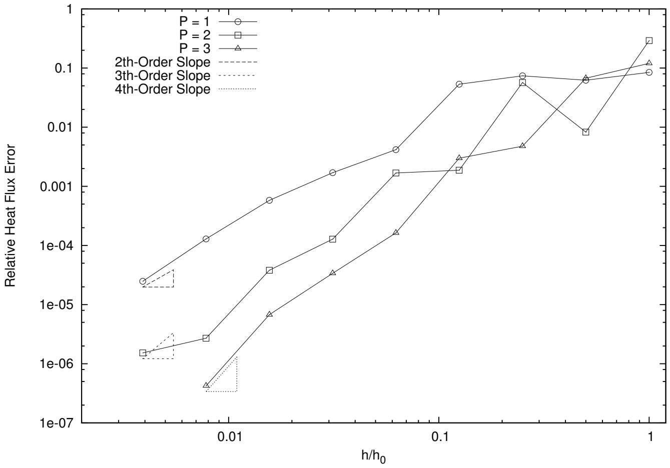


Figure 7-16: Scalar convection-diffusion uniformly refined grid heat flux convergence.

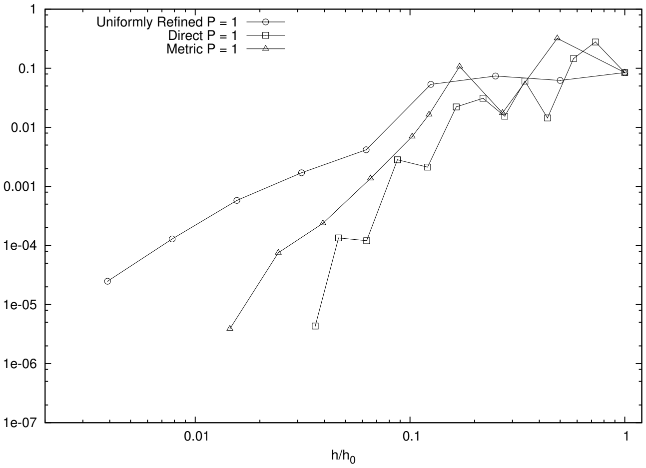


Figure 7-17: Scalar convection-diffusion adapted grid heat flux convergence.

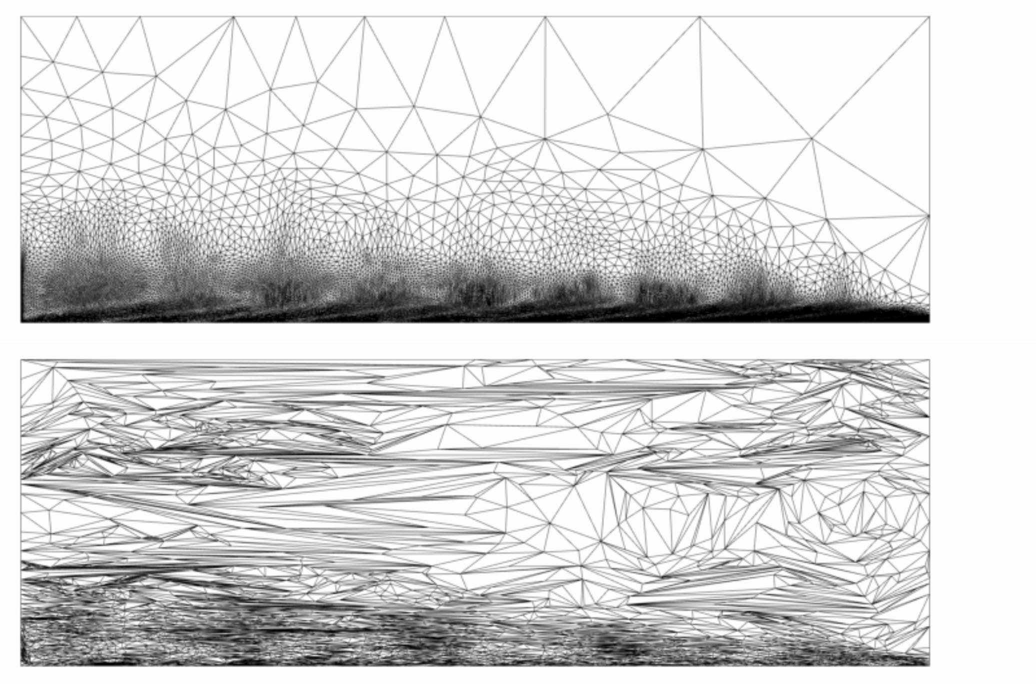


Figure 7-18: Final grids adapted by metric-based BAMG and direct method.

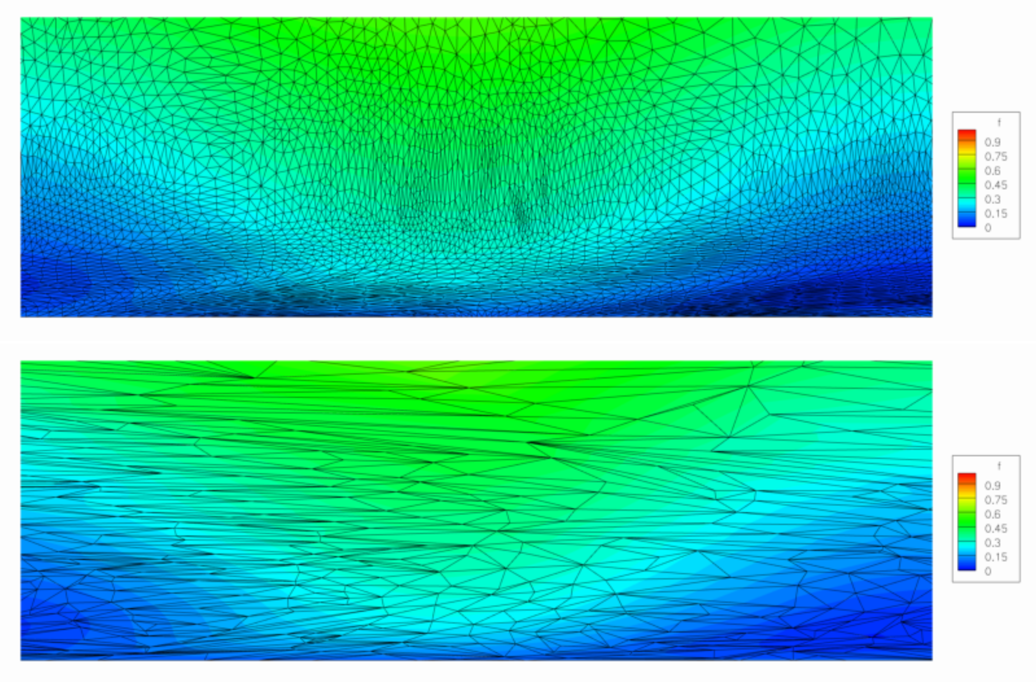


Figure 7-19: Detail of final grids adapted by metric-based BAMG and direct method with scalar solution.

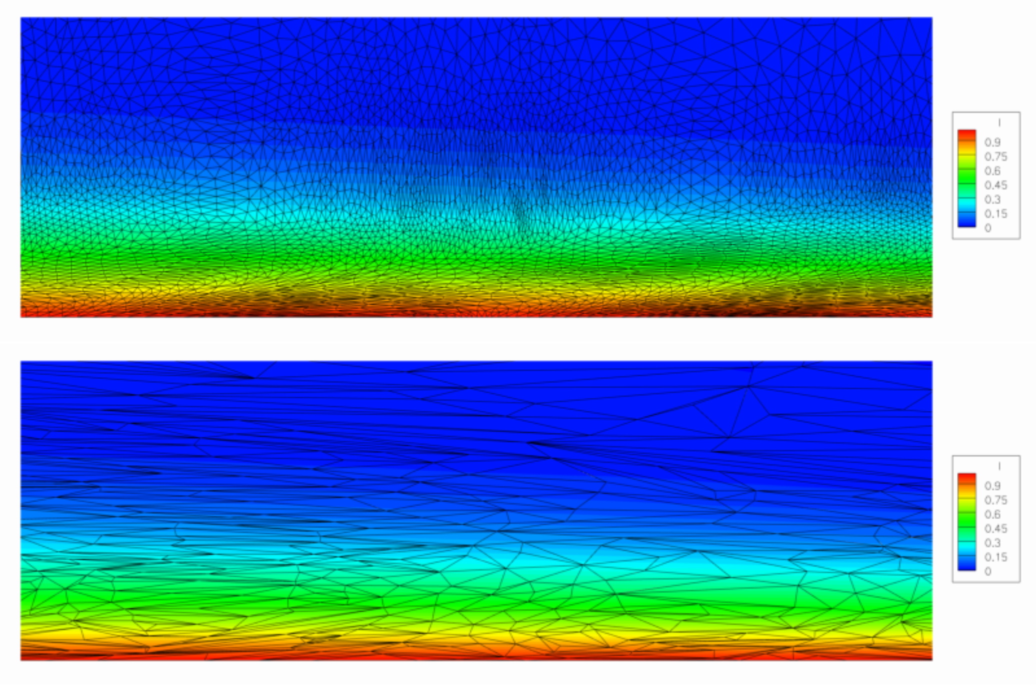
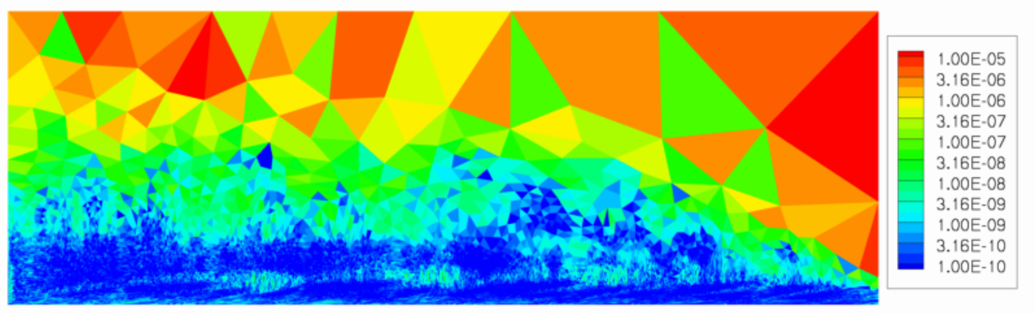
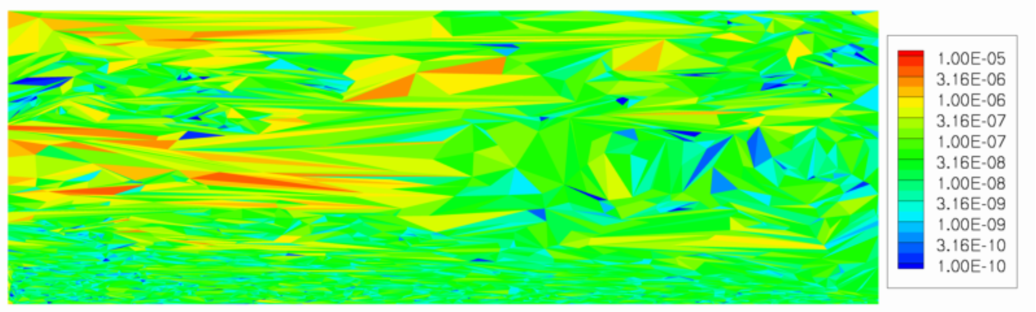


Figure 7-20: Detail of final grids adapted by metric-based BAMG and direct method with adjoint solution.

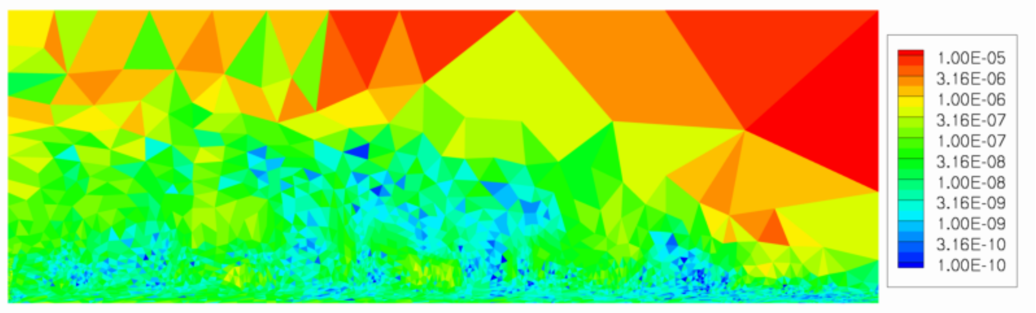
example, Section 7.2. In both examples, the metric-based errors are located in well defined regions and the direct approach errors are spread in a more erratic pattern. The direct approach may be hindered by the difficulty of the combined discrete and continuous optimization problem. An improved optimization methodology may increase the regularity and efficiency of the resultant grid, remaining a topic for future work.



(a) Final metric-based BAMG grid (38088 triangles).



(b) Final direct method grid (6101 triangles).



(c) Intermediate metric-based BAMG grid (5177 triangles).

Figure 7-21: Output error estimates.

Chapter 8

Conclusion

A robust cut-cell determination methodology is presented, which utilizes double-precision floating point error predicates applied systematically to prevent topology inconsistencies. The volume defined by a triangular cut surface is subtracted from the median dual of an anisotropically adapted tetrahedral background grid. An existing node-based finite-volume flow and adjoint solver are modified to utilize these cut cells. A differentiable heuristic limiter is utilized to provide reliable iterative convergence in supersonic flow with shocks. The continuous limiter's exact linearization enables stable flow and adjoint convergence.

An output-based error estimation scheme is provided to combine the flow and adjoint problems into a computable adaptation indicator for use with cut cells. This single grid remaining error estimation procedure provides reliable guidance for adaptation while eliminating the memory barrier of the existing embedded grid approach. The tetrahedral grid adaptation operators are placed in functional form to enable precise descriptions of higher-level adaptation algorithms. These higher-level algorithms include an established metric-based approach and a novel direct approach. The utility of the anisotropic metric-based approach is demonstrated with sonic boom prediction because of the need to propagate the relatively weak signals long distances. The propagated pressure signatures are validated with wind tunnel measurements. The automation and robustness of the output-adaptive technique is illustrated by the use of extremely coarse initial grids that were created without knowledge of the final grid resolution requirements. A domain decomposed parallel adaptation scheme is described to permit the creation of large grids.

The direct adaptation approach allows for a concise expression for the error-estimate, which is directly evaluated and controlled via the grid operators. The new direct method is able to implicitly conform to the anisotropy of the dual solution as well as the primal solution. The anisotropies of the primal and dual solutions are implicitly blended depending on their relative importance to computing a specified output function. The direct approach provides a natural extension to higher-order solution schemes and systems of equations. A variety of results show that the direct approach controls the error in fewer degrees of freedom than metric-based adaptation, but produces a less regular grid.

8.1 Contributions

- The first 3D finite-volume cut-cell method that utilizes the median dual of an anisotropically adapted tetrahedral background grid
- A reliable remaining output error estimation procedure for adaptation that does not use an expensive embedded grid, which requires a prohibitive amount of memory for large 3D problems
- Dramatic increase in robustness for anisotropic 3D output-based adaptation is demonstrated for sonic boom applications
- General anisotropic adaptation scheme to directly control interpolation error and output-based error estimates without an intervening metric specification that is directly applicable to higher-order solutions

8.2 Future Work

Output-adaptive cut-cell simulation of 3D viscous flows. Utilize the increase in robustness and automation of anisotropically adaptive 3D cut-cell simulations for turbulent calculations. Cut cells with Cartesian background grids have been applied to laminar flows,[42] or turbulent flows with the aid of hybrid grids,[46,82] integral boundary methods,[3] or wall functions.[77] Utilizing the general anisotropy of tetrahedral background grids may allow for turbulent cut-cell calculations without the approximations of wall functions, hybrid grids, or integral boundary methods. Fidkowski and Darmofal[54] have demonstrated output-adaptive cut-cell simulations for 2D laminar and scalar convection-diffusion equations with a discontinuous Galerkin method. Venditti and Darmofal[147] have demonstrated output-adaptive 2D turbulent simulations with a body-fitted finite-volume method. Lee-Rausch et al.,[87] Balasubramanian,[15] and Kim and Nakahashi[83] performed output-adaptive 3D turbulent simulations, but limitations in grid mechanics have prevented its routine use for typical problems. The use of floating-point grid validity checks for adaptive tetrahedral grids may need to be reconsidered as the required element aspect ratios increase and cell volumes decrease.

Design sensitivities and optimization with output-based adaptation. The adjoint solution is computed during the output-based adaptation process to form the adaptation indicator. The adjoint solution can be utilized to form adjoint-based sensitivities for gradient-based design optimization.[115] Nemec and Aftosmis[110,111] have demonstrated design sensitivities and optimization for bodies in inviscid flows with cut cells and Cartesian background grids. The general anisotropy of tetrahedral background grids allows this technique to be efficiently applied to strongly anisotropic problems, such as sonic boom shaping. In the current formulation, there is discontinuous change once the control volume is infinitesimally cut, see Section 3.3. This discontinuous behavior may cause difficulties that must be addressed for shape sensitivities and design.

Improve metric-based implementation to reduce execution time. The implementation of the output-based adaptation framework has potential for reduced execution time. The current cut cell determination algorithm suffers from a load imbalance during parallel execution. As mentioned in Chapter 2, an improved partitioning heuristic may improve parallel efficiency and reduce execution time. An other potential improvement could come from using the primal elements (tetrahedra) as the control volumes, because they are composed fewer triangular faces than the median duals. The adaptation segments of the timing presented in Section 6.5 are a large portion of the entire execution time. Any improvements to reduce adaptation time will significantly reduce the total execution time. Reducing the analysis time remains a topic for future work and would be critical to affordable design optimization.

Improving grid optimization to directly control output error. The direct control of output error grid adaptation approach has shown a benefit in reduced degrees of freedom for a comparable accuracy to existing metric-based approaches. The grids produced by this new technique lack the regularity of the metric-based approach. This may be due to output error estimates being a more difficult objective function to optimize than metric conformity. Improving the grid optimization algorithm for direct control of output error may produce more regular and efficient grids. The technique must also be extended to 3D cut-cells to utilize the same increase in robustness shown by the metric-based cut-cell simulations of complex geometries.

Appendix A

Finite-Volume Scalar Convection

The 2D steady-state scalar r convection equation is

$$\frac{\partial f}{\partial x} + \frac{\partial g}{\partial y} = 0 \tag{A.1}$$

where the fluxes are defined as a function of the two components of velocity u and v

$$f = ru \tag{A.2}$$

and

$$g = rv. \tag{A.3}$$

The velocity field (u, v) is constant with $u = 1$ and $v = 0$ for this analysis. The zero v velocity makes the problem effectively 1D. The geometry of the domain is shown in Fig. A-1. There is no flux in the y-direction, because $v = 0$. The state r_0 is specified

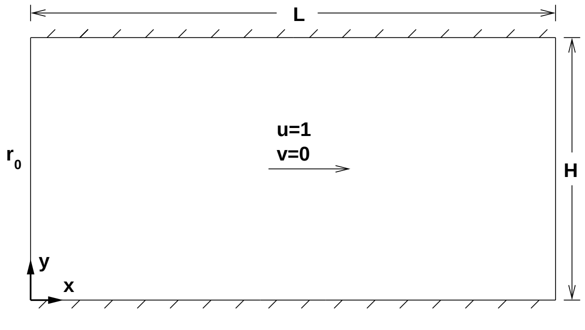


Figure A-1: Scalar convection problem geometry.

at the left inflow boundary. The scalar output functional J is defined as the integral of r over the domain

$$J = \int_{\Omega} r \, d\Omega. \tag{A.4}$$

The steady-state equation and solution is

$$\frac{\partial f}{\partial x} = \frac{\partial (ru)}{\partial x} = u\frac{\partial r}{\partial x} = 0 \Rightarrow r(x) = r_0, \tag{A.5}$$

which is determined by the inflow boundary state r_0. The adjoint equation for the cost function in Equation (A.4) is

$$-u\frac{\partial \lambda}{\partial x} = 1. \tag{A.6}$$

The characteristic of the adjoint problem travels in the opposite direction as the primal problem. The adjoint solution is determined by the downstream boundary condition where $\lambda(L) = 0$,

$$\lambda(x) = \frac{L - x}{u}. \tag{A.7}$$

A.1 Discrete Solution

The domain in Fig. A-1 is subdivided into $n-2$ cells of uniform width $\Delta x = L/(n-1)$ and two boundary cells of width $\Delta x/2$. The discretized domain is shown in Fig. A-2. This domain has been divided into 8 cells. The solution storage location of the cells are marked with + symbols. The cell averaged value of each cell i solution is r_i. The state s_i is used to construct upwind flux f_i at the right face of cell i.

The residual in each cell is

$$R_i = \begin{cases} f_1 H - r_o u H : & i = 1; \\ f_i H - f_{i-1} H : & i > 1 \end{cases} \tag{A.8}$$

where r_0 is the specified value of r at the inflow boundary. The fluxes f_i are con-

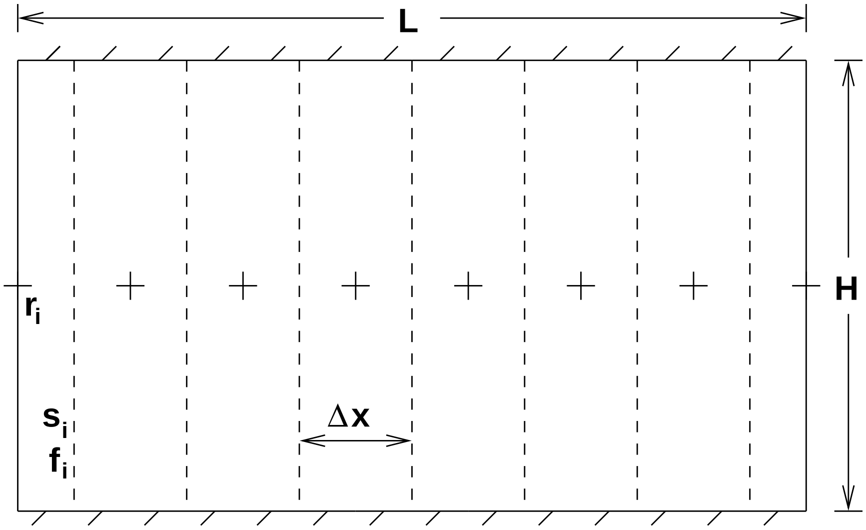


Figure A-2: Convection problem uniformly discretized into 8 cells.

structed from the reconstructed state s_i on the upwind side of the cell interface

$$f_i = s_i u. \tag{A.9}$$

The upwind cell centered value of r_i is combined with a reconstructed gradient $\nabla\tilde{r}_i$ to construct the extrapolated state s_i

$$s_i = r_i + \frac{\Delta x}{2}\nabla\tilde{r}_i. \tag{A.10}$$

The gradients $\nabla\tilde{r}_i$, defined at at the cell centroids, are reconstructed by a fitting procedure of cell averaged data. These gradients are chosen to be the best point-wise fit of the reconstructed solution with the cell averaged solution at the cell centroids of neighboring cells. The point-wise error in this fit at cell i with neighbor j is

$$e_{ji} = (r_j - r_i) - (x_j - x_i)\nabla\tilde{r}_i. \tag{A.11}$$

The unknowns $\nabla\tilde{r}_i$ are chosen to minimize the squared error norm for the m neighbors of cell i

$$\left[\begin{array}{c}(x_1-x_i)\cdots(x_m-x_i)\end{array}\right]\left[\begin{array}{c}(x_1-x_i)\\ \vdots\\ (x_m-x_i)\end{array}\right]\nabla\tilde{r}_i=\left[\begin{array}{c}(x_1-x_i)\cdots(x_m-x_i)\end{array}\right]\left[\begin{array}{c}(r_1-r_i)\\ \vdots\\ (r_m-r_i)\end{array}\right] \tag{A.12}$$

$$\nabla\tilde{r}_i=\frac{\sum_{j=1}^{m}(x_j-x_i)(r_j-r_i)}{\sum_{j=1}^{m}(x_j-x_i)^2}. \tag{A.13}$$

The special case of two adjacent neighbors of a uniform mesh of spacing Δx becomes

$$\nabla\tilde{r}_i=\frac{r_{i+1}-r_{i-1}}{2\Delta x}, \tag{A.14}$$

the uniform grid central difference gradient approximation. In the cells adjacent to the boundaries, this reconstruction degrades to a one-sided finite difference approximation.

$$\nabla\tilde{r}_1=\frac{r_2-r_1}{\Delta x} \tag{A.15}$$

$$\nabla\tilde{r}_n=\frac{r_n-r_{n-1}}{\Delta x}. \tag{A.16}$$

Figure A-3 shows the primal and dual solutions for an eight cell grid with the first cell shrunk to 0.1 and 1.0×10^{-6} the width of the original. The first cell is reduced in size to model a small boundary cell. The discrete primal solution is exact, but a small, bounded error is created in interior of the dual solution.

To investigate large discrete adjoint values in small cut cells, a small cell is introduced in between two uniform cells, see Fig. A-4. Cell centroids are shown with + symbols. The cell numbering convention is shown in Fig. A-5. The upwind flux at each face is denoted as f#. The fluxes are

$$\begin{aligned}f_1 &= H(r_0),\\ f_2 &= h(r_1+(H-0.5h)\nabla r_1),\\ f_3 &= (H-h)(r_1+H\nabla r_1),\\ f_4 &= h(r_3+0.5h\nabla r_3),\text{ and}\\ f_5 &= Hr_2.\end{aligned} \tag{A.17}$$

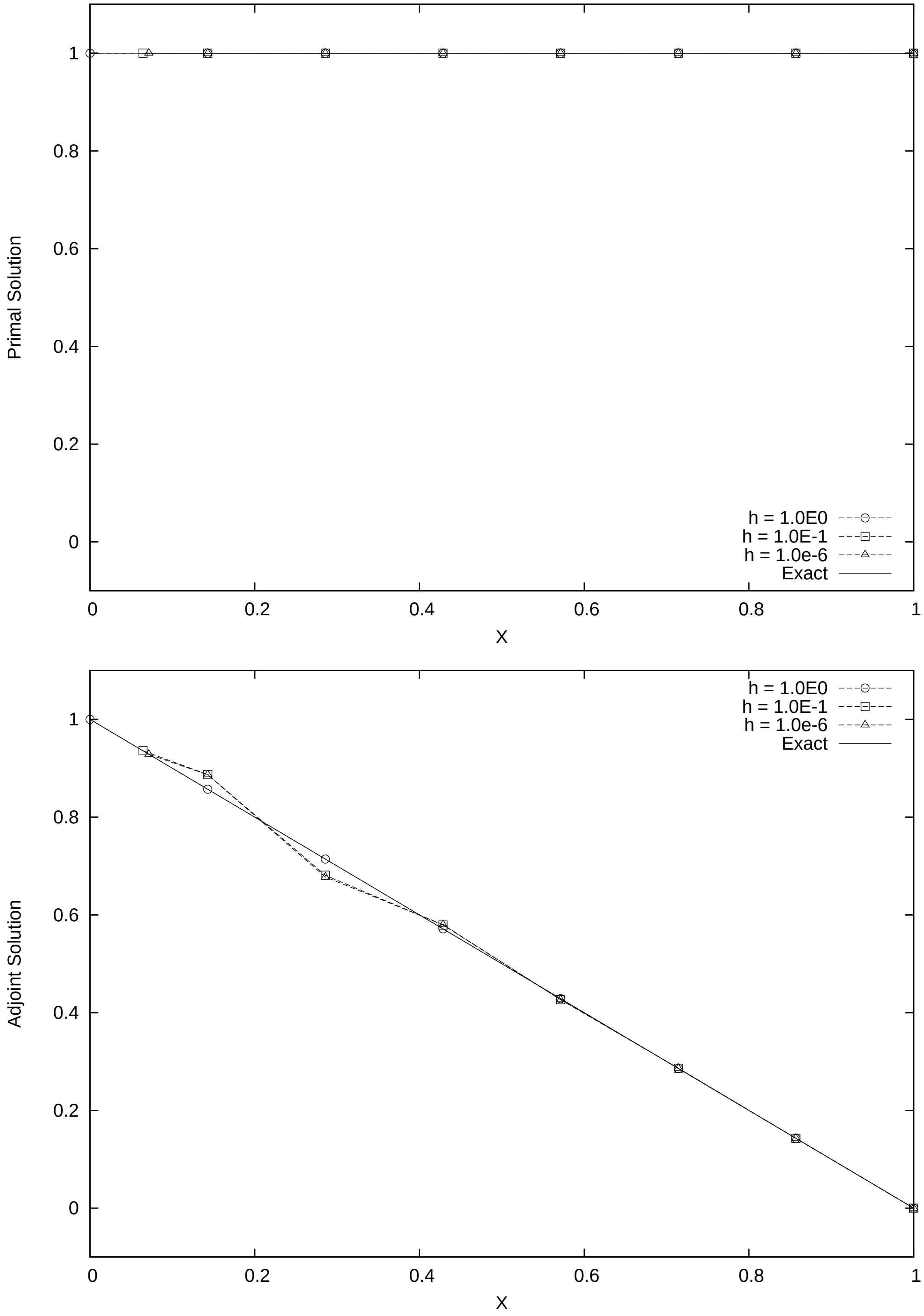


Figure A-3: Discrete adjoint solution for 8 cells.

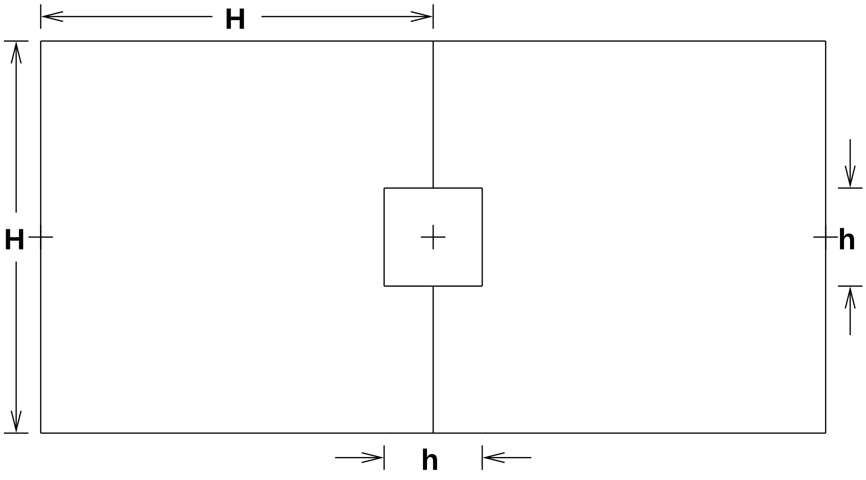


Figure A-4: Geometry of a small cell near two large neighbors.

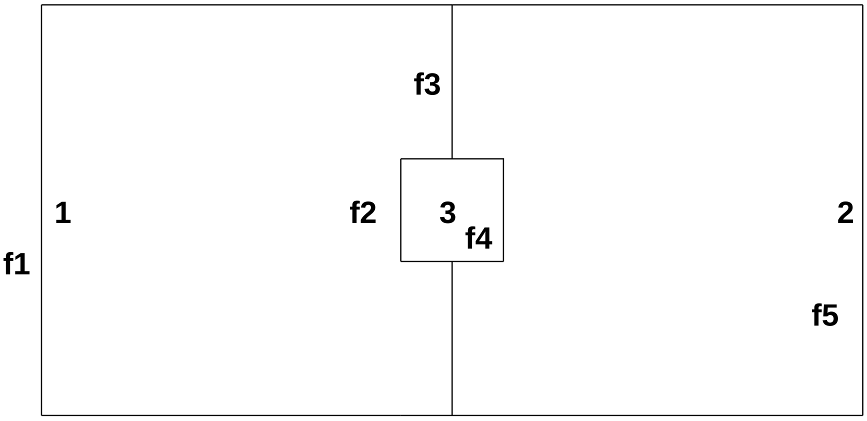


Figure A-5: Cell and upwind flux numbering convention.

The reconstructed gradients are

$$\begin{aligned}\nabla r_1 &= \tfrac{2(r_2-r_1)+(r_3-r_1)}{5H},\\ \nabla r_2 &= \tfrac{-2(r_1-r_2)-(r_3-r_2)}{5H}, \text{ and}\\ \nabla r_3 &= \tfrac{-(r_1-r_3)+(r_2-r_3)}{2H}.\end{aligned} \tag{A.18}$$

The modified reconstructed gradients (small cell is excluded from larger cell) are

$$\begin{aligned}\nabla r_1 &= \tfrac{(r_2-r_1)}{2H},\\ \nabla r_2 &= \tfrac{-(r_1-r_2)}{2H}, \text{ and}\\ \nabla r_3 &= \tfrac{-(r_1-r_3)+(r_2-r_3)}{2H}.\end{aligned} \tag{A.19}$$

The cost function for the adjoint is the integral of r in the domain.

$$\begin{aligned}i_1 &= H^2(r_1+0.5\nabla r_1),\\ i_2 &= H^2(r_2-0.5\nabla r_2), \text{ and}\\ i_3 &= h^2 r_3.\end{aligned} \tag{A.20}$$

The error in the discrete adjoint solution, as compared with the analytic solution, is plotted in Fig. A-6. The error in the small central cell with the original gradient reconstruction stencil grows without bound. The error is reduced and stays constant with the modified gradient reconstruction system.

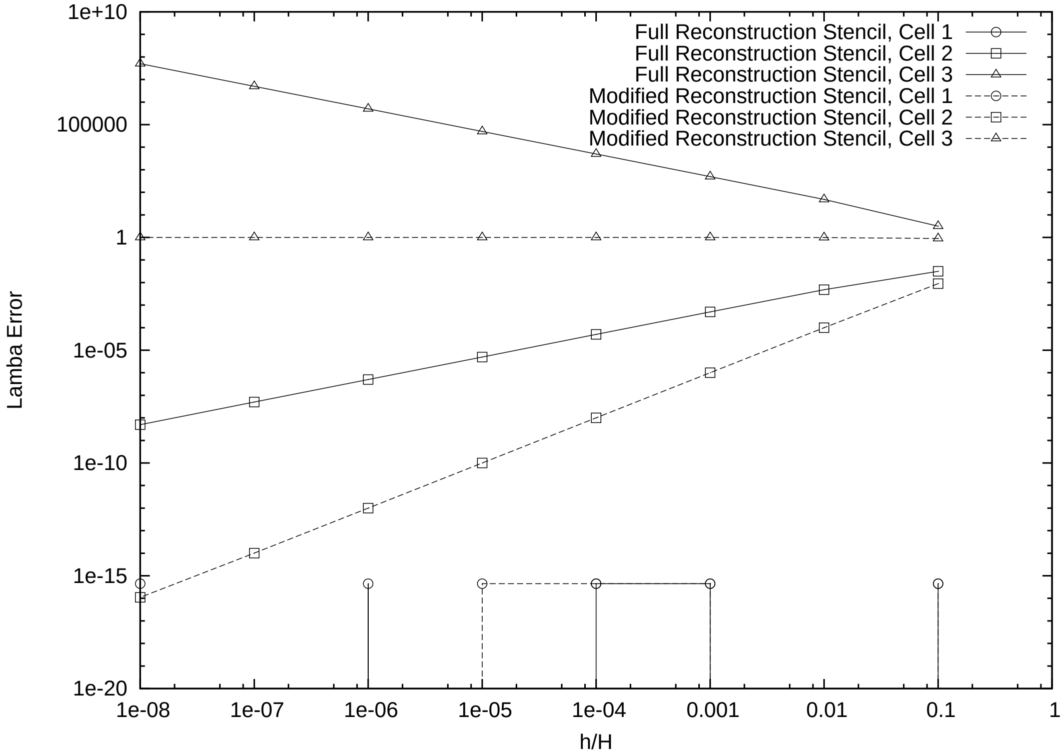


Figure A-6: Small cell case adjoint error.

Appendix B

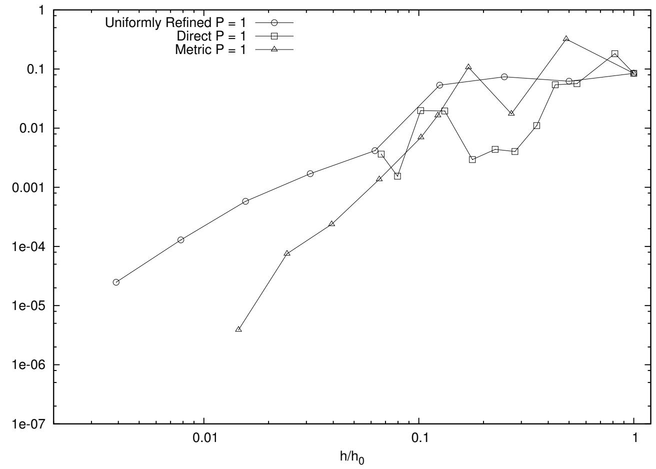
Direct Control of 2D Scalar Convection-Diffusion Output Error Components

In Section 7.4, a direct output error control and metric-based adaption schemes are applied to 2D convection-diffusion. The grid resulting from the direct output error control approach was less regular than the metric-based approach. To investigate the connection between the output adaptation error function, Eq. (7.27), and the resulting grid, this equation is split into two components,

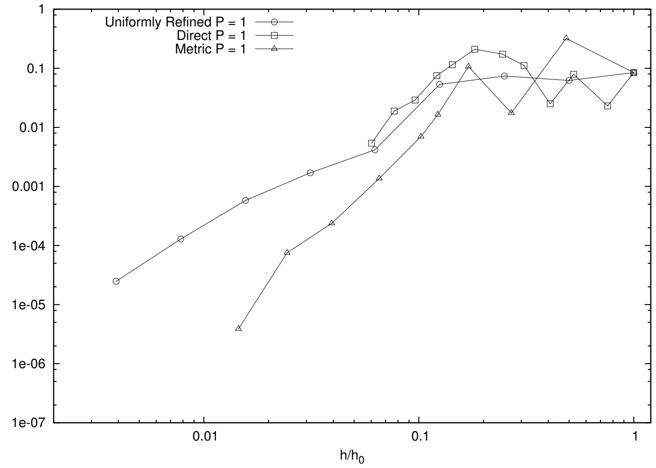
$$e_{\text{primal}} = |\mathcal{R}(\bar{u}, (\hat{\lambda} - \bar{\lambda}))|, \tag{B.1}$$

$$e_{\text{dual}} = |\mathcal{R}^{\lambda}(\bar{u}; (\hat{u} - \bar{u}), \bar{\lambda})|. \tag{B.2}$$

The separated error functional only includes a primal or a dual residual term instead of the combined error functional of Eq. (7.27). The direct output error control grid adaptation process is run individually for e_{primal} and e_{dual}. The convergence of the heat flux error for the primal, Fig. B-1(a), and the dual, Fig. B-1(b), terms are inferior to the combined output error, Fig. 7-17. Neither separated error functional out performs the metric-based approach and they are very close to the uniformly refined grid error convergence.



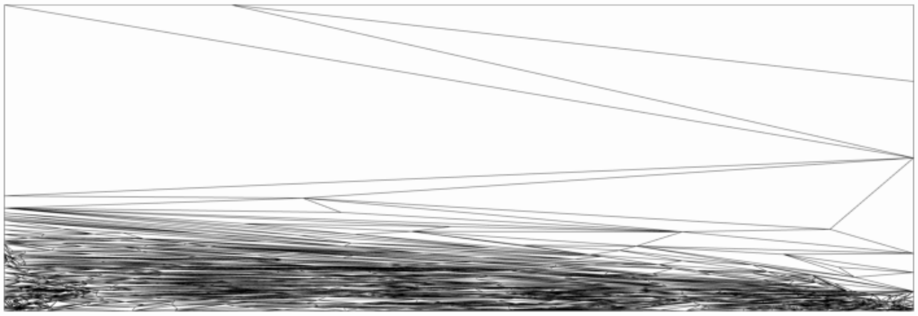
(a) e_{primal} optimization function.



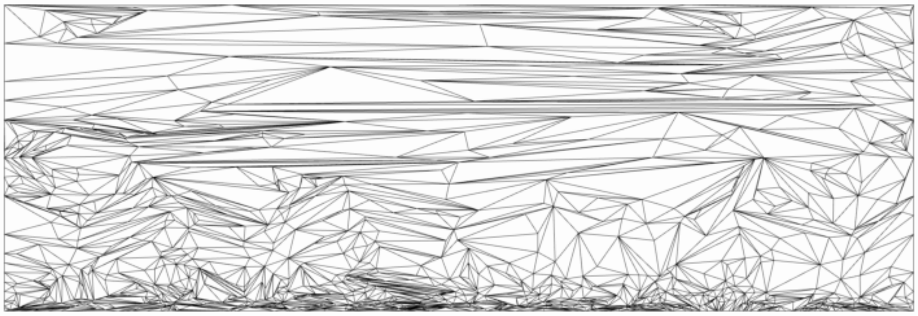
(b) e_{dual} optimization function.

Figure B-1: Scalar convection-diffusion adapted grid heat flux convergence.

The final adapted grids using e_{primal} and e_{dual} optimization functions are shown in Fig. B-2. The e_{primal} adapted grid, Fig. B-2(a), is strongly anisotropic and clustered to resolve the adjoint solution boundary layer seen in Fig. 7-15. The e_{dual} adapted grid, Fig. B-2(b), is resolving the primal solution near the lower boundary.

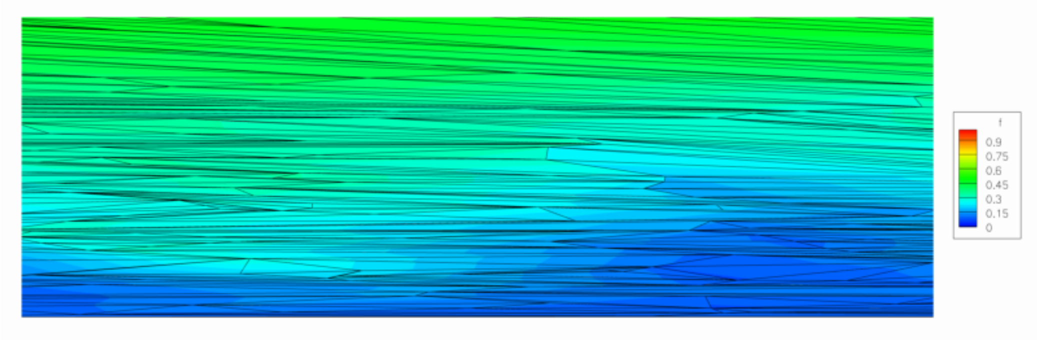


(a) e_{primal} optimization function

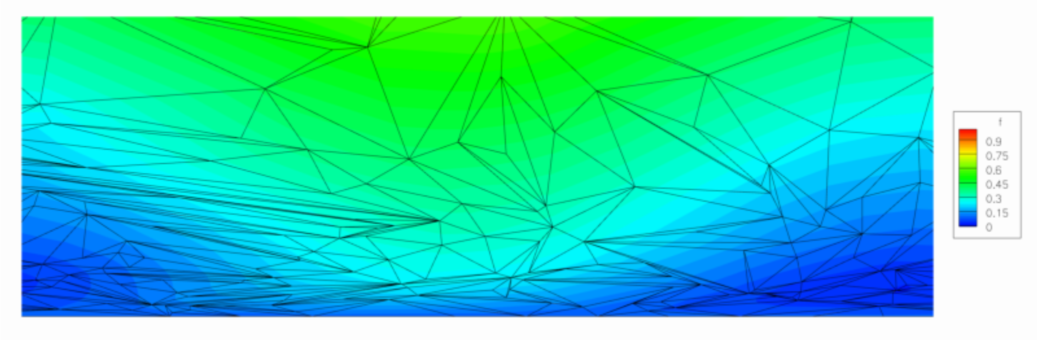


(b) e_{dual} optimization function

Figure B-2: Final grids adapted by the direct method.

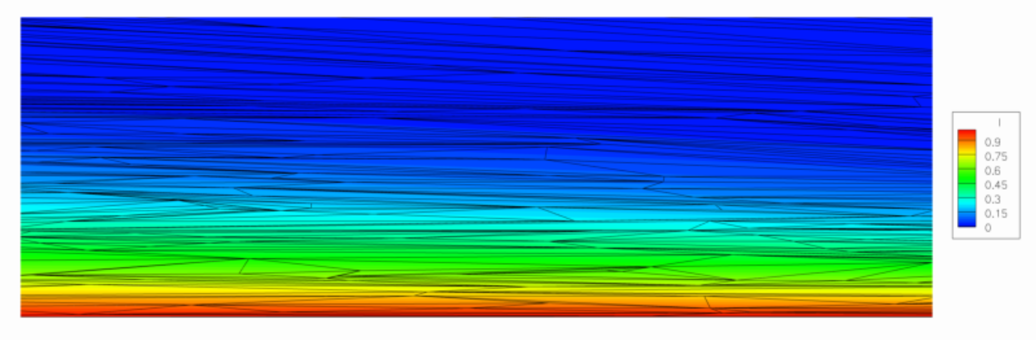


(a) e_{primal} optimization function

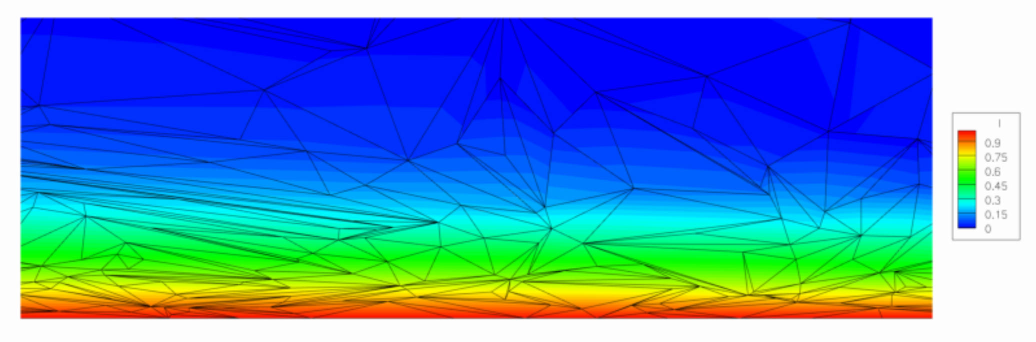


(b) e_{dual} optimization function

Figure B-3: Detail of final grids adapted by the direct method with primal solution.



(a) e_{primal} optimization function



(b) e_{dual} optimization function

Figure B-4: Detail of final grids adapted by the direct method with adjoint solution.

Bibliography

[1] M. J. Aftosmis. Solution adaptive cartesian grid methods for aerodynamic flows with complex geometries. 28th Computational Fluid Dynamics Lecture Series. VKI Lecture Series 1997-2, 1997.

[2] M. J. Aftosmis and M. J. Berger. Multilevel error estimation and adaptive h-refinement for cartesian meshes with embedded boundaries. AIAA Paper 2002-0863, 2002.

[3] M. J. Aftosmis, M. J. Berger, and J. J. Alonso. Applications of a cartesian mesh boundary-layer approach for complex configurations. AIAA Paper 2006-0652, 2006.

[4] M. J. Aftosmis, M. J. Berger, and J. E. Melton. Adaptation and surface modeling for cartesian mesh methods. AIAA Paper 95-1725, 1995.

[5] M. J. Aftosmis, M. J. Berger, and J. E. Melton. Robust and effcient cartesian mesh generation for component-based geometry. *AIAA Journal*, 36(6):952–960, 1998.

[6] M. J. Aftosmis, M. Delanaye, and R. Haimes. Automatic generation of cfd-ready surface triangulations from cad geometry. AIAA Paper 99-0776, 1999.

[7] M. J. Aftosmis, D. Gaitonde, and T. S. Tavares. Behavior of linear reconstruction techniques on unstructured meshes. *AIAA Journal*, 33(11):2038–2049, Nov. 1995.

[8] F. Alauzet, X. Li, E. S. Seol, and M. S. Shephard. Parallel anisotropic 3d mesh adaptation by mesh modification. *Engineering with Computers*, 21(3):247–258, 2006.

[9] J. J. Alonso, I. M. Kroo, and A. Jameson. Advanced algorithms for design and optimization of quiet supersonic platforms. AIAA Paper 2002–144, 2002.

[10] N. Amenta, M. Bern, and D. Eppstein. Optimal point placement for mesh smoothing. *Journal of Algorithms*, 30(2):302–322, 1999.

[11] W. K. Anderson and D. L. Bonhaus. An implicit upwind algorithm for computing turbulent flows on unstructured grids. *Computers and Fluids*, 23(1):1–22, 1994.

[12] W. K. Anderson, R. D. Rausch, and D. L. Bonhaus. Implicit/multigrid algorithm for incompressible turbulent flows on unstructured grids. *Journal of Computational Physics*, 128(2):391–408, 1996.

[13] L. Andrews. A template for the nearest neighbor problem. *C/C++ Users Journal*, 19(11):40–49, Nov. 2001.

[14] T. J. Baker. Mesh adaptation strategies for problems in fluid dynamics. *Finite Elements in Analysis and Design*, 25(3–4):243–273, 1997.

[15] R. Balasubramanian. *Adjoint-Based Error Estimation and Grid Adaptation for Functional Outputs from CFD Simulations.* PhD thesis, Mississippi State University, 2005.

[16] R. Balasubramanian and J. C. Newman III. Discrete direct and discrete adjoint sensitivity analysis for variable mach flows. *International Journal for Numerical Methods in Engineering*, 66(2):297–318, 2006.

[17] R. Balasubramanian and J. C. Newman III. Solution adaptive mesh refinement using adjoint error analysis. AIAA Paper 2006–3314, 2006.

[18] R. E. Bank and R. K. Smith. Mesh smoothing using a posteriori error estimates. *SIAM Journal on Numerical Analysis*, 34(3):979–997, 1997.

[19] G. E. Barter. *Shock Capturing with PDE-Based Artificial Viscosity for an Adaptive, Higher-Order Discontinuous Galerkin Finite Element Method.* PhD thesis, Massachusetts Institute of Technology, 2008.

[20] T. J. Barth. A 3-d upwind euler solver for unstructured meshes. AIAA Paper 91–1548, 1991.

[21] T. J. Barth and H. Deconinck, editors. *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics.* Number 25 in Lecture Notes in Computational Science and Engineering, Springer-Verlag, 2002.

[22] T. J. Barth and D. C. Jespersen. The design and application of upwind schemes on unstructured meshes. AIAA Paper 89–366, 1989.

[23] M. J. Berger, M. J. Aftosmis, and S. M. Murman. Analysis of slope limiters on irregular grids. AIAA Paper 2005-0490, 2005.

[24] C. W. Boope. Towards complete configurations using an embedded grid approach. NASA CR-3030, NASA Langley Research Center, July 1978.

[25] H. Borouchaki, P. L. George, F. Hecht, P. Laug, and E. Saltel. Delaunay mesh generation governed by metric specifications. part i. algorithms. *Finite Elements in Analysis and Design*, 25(1–2):61–83, 1997.

[26] H. Borouchaki, P. L. George, F. Hecht, P. Laug, and E. Saltel. Delaunay mesh generation governed by metric specifications. part ii. applications. *Finite Elements in Analysis and Design*, 25(1–2):85–109, 1997.

[27] F. J. Bossen and P. S. Heckbert. A pliant method for anisotropic mesh generation. In *5th International Meshing Roundtable*, pages 63–74, Oct. 1996. http://www.cs.cmu.edu/ ph.

[28] R. Campbell, M. Carter, K. Deere, and K. A. Waithe. Efficient unstructured grid adaptation methods for sonic boom prediction. AIAA Paper 2008–7327, 2008.

[29] H. Carlson, R. Mack, and O. Morris. *A Wind-Tunnel Investigation of the Effect of Body Shape on Sonic-Boom Pressure Distributions.* NASA/TN-1965-3106, 1965.

[30] H. W. Carlson. Experimental and analytic research on sonic boom generation at nasa. In A. R. Seebass, editor, *Sonic Boom Research*, pages 9–23. NASA, NASA-SP-147, Apr. 1967.

[31] H. W. Carlson and O. A. Morris. Wind-tunnel sonic-boom testing techniques. *AIAA Journal of Aircraft*, 4(3):245–249, 1967.

[32] M. Carter and K. Deere. Grid sourcing and adaptation study using unstructured grids for supersonic boom prediction. AIAA Paper 2008–6595, 2008.

[33] M. J. Castro-Diáz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic unstructured mesh adaptation for flow simulations. *International Journal for Numerical Methods in Fluids*, 25(4):475–491, 1997.

[34] P. A. Cavallo, N. Sinha, and G. M. Feldman. Parallel unstructured mesh adaptation method for moving body applications. *AIAA Journal*, 43(9):1937–1845, Sept. 2005.

[35] M. S. Chaffin and S. Pirzadeh. Unstructured navier-stokes high-lift computations on a trapezoidal wing. AIAA Paper 2005–5084, 2005.

[36] M. K. Chan. *Supersonic Aircraft Optimization for Minimizing Drag and Sonic Boom.* PhD thesis, Stanford University, Aug. 2003.

[37] E. F. Charlton and K. G. Powell. An octree solution to conservation laws over arbitrary regions (oscar). AIAA Paper 97-198, 1997.

[38] S. H. Cheung, T. A. Edwards, and S. L. Lawrence. Application of computational fluid dynamics to sonic boom near- and mid-field prediction. *AIAA Journal of Aircraft*, 29(5):920–926, September–October 1992.

[39] S. Choi, J. J. Alonso, and H. S. Chung. Design of a low-boom supersonic business jet using evolutionary algorithms and an adaptive unstructured mesh method. AIAA Paper 2004–1758, 2004.

[40] S. Choi, J. J. Alonso, and E. van der Weide. Numerical and mesh resolution requirements for accurate sonic boom prediction of complete aircraft configurations. AIAA Paper 2004–1060, 2004.

[41] S. E. Cliff and S. D. Thomas. Euler/experiment correlations of sonic boom pressure signatures. *AIAA Journal of Aircraft*, 30(5):669–675, Sep.–Oct. 1993. See also AIAA Paper 91–3276.

[42] W. J. Coirier. *An Adaptively-Refined, Cartesian, Cell-Based Scheme for the Euler and Navier-Stokes Equations.* PhD thesis, The University of Michigan, 1994.

[43] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (Second Edition).* MIT Press, Cambridge, Massachusetts, 2003.

[44] H. L. De Cougny and M. S. Shephard. Parallel refinement and coarsening of tetrahedral meshes. *International Journal for Numerical Methods in Engineering*, 46(7):1101–1125, 1999.

[45] D. L. De Zeeuw. *A Quadtree-Based Adaptively-Refined Cartesian-Grid Algorithm For Solution of the Euler Equations.* PhD thesis, The University of Michigan, 1993.

[46] M. Delanaye, M. J. Aftosmis, M. J. Berger, Y. Liu, and T. H. Pulliam. Automatic hybrid-cartesian grid generation for high-reynolds number flows around complex geometries. AIAA Paper 99-0777, 1999.

[47] M. J. Djomehri and L. L. Erickson. An assessment of the adaptive unstructured tetrahedral grid, euler flow solver code felisa. TP 3526, NASA Ames Research Center, Dec. 1994.

[48] N. D. Domel and S. L. Karman, Jr. Splitflow: Progress in 3d cfd with cartesian omni-tree grids for complex geometries. AIAA Paper 2000–1006, 2000.

[49] J. Dompierre, M.-G. Vallet, and P. L. F. Guibault. An analysis of simplex shape measures for anisotropic meshes. *Computer Methods in Applied Mechanics and Engineering*, 194:4895–4914, 2005.

[50] R. P. Dwight. Goal-oriented mesh adaptation using a dissipation-based error indicator. AIAA Paper 2007–4093, 2007.

[51] H. Edelsbrunner and E. P. Mücke. Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics*, 9(1):66–104, Jan. 1990.

[52] J. K. Elliott. *Aerodynamic optimization based on the Euler and Navier-Stokes equations using unstructured grids.* PhD thesis, Massachusetts Institute of Technology, 1998.

[53] K. J. Fidkowski and D. L. Darmofal. Output-based adaptive meshing using triangular cut cells. Technical Report ACDL TR-06-2, Aerospace Computational Design Laboratory, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 2006.

[54] K. J. Fidkowski and D. L. Darmofal. A triangular cut-cell adaptive method for high-order discretizations of the compressible navier–stokes equations. *Journal Computational Physics*, 225(2):1653–1672, Aug. 2007.

[55] K. J. Fidkowski, T. A. Oliver, J. Lu, and D. L. Darmofal. p-multigrid solution of high-order discontinuous galerkin discretizations of the compressible navierstokes equations. *Journal Computational Physics*, 207:92–113, 2005.

[56] K. Fouladi. Unstructured grids for sonic-boom analysis. AIAA Paper 99–2929, 1993.

[57] L. A. Freitag, M. T. Jones, and P. E. Plassmann. A parallel algorithm for mesh smoothing. *SIAM Journal on Scientific Computing*, 20(6):2023–2040, 1999.

[58] L. A. Freitag and C. Ollivier-Gooch. Tetrahedral mesh improvement using swapping and smoothing. *International Journal for Numerical Methods in Engineering*, 40:3979–4002, 1997.

[59] P. J. Frey and F. Alauzet. Anisotropic mesh adaptation for cfd computations. *Computer Methods in Applied Mechanics and Engineering*, (194):5068–5082, Nov. 2005.

[60] A. R. George. Reduction of sonic boom by azimuthal redistribution of overpressure. AIAA Paper 68–159, 1968.

[61] P.-L. George, F. Hecht, and E. Saltel. Automatic mesh generator with specified boundary. *Computer Methods in Applied Mechanical Engineering*, 92(3):269–288, 1991.

[62] M. Giles, M. Duta, J.-D. Müller, and N. Pierce. Algorithm developments for discrete adjoint methods. *AIAA Journal*, 41(2):198–205, 2003. See also AIAA Paper 2001–2596.

[63] M. B. Giles and E. Süli. Adjoint methods for pdes: a posteriori error analysis and postprocessing by duality. *Acta Numerica*, 11:145–236, 2002.

[64] P. J. Green and R. Sibson. Computing dirichlet tessellations in the plane. *Computer Journal*, 21(2):168–173, 1978.

[65] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent cfd. part i: general principles. *International Journal for Numerical Methods in Fluids*, 32(6):725–744, 2000.

[66] R. Haimes and C. Crawford. Unified geometry access for analysis and design. 12th International Meshing Roundtable, pages 21–31. Sandia National Lab, Sept. 2003.

[67] R. Hartmann and P. Houston. Adaptive discontinuous galerkin finite element methods for the compressible euler equations. *Journal Computational Physics*, 183(2):508–532, 2002.

[68] A. Haselbacher. A weno reconstruction algorithim for unstructured grids based on explicit stencil construction. AIAA Paper 2005–879, 2005.

[69] W. D. Hayes, R. C. Haefeli, and H. E. Kulsrud. *Sonic boom propagation in a stratified atmosphere, with computer program.* NASA-CR-1299, Apr. 1969.

[70] F. Hecht. The mesh adapting software: Bamg. INRIA Report, 1998. `http://www-rocq1.inria.fr/gamma/cdrom/www/bamg/eng.htm`.

[71] R. M. Hicks and J. P. Mendoza. *Prediction of aircraft sonic boom characteristics from experimental near field results.* NASA-TM-X-1477, Nov. 1967.

[72] L. W. Hunton, R. M. Hicks, and J. P. Mendoza. *Some effects of wing planform on sonic boom.* NASA/TN-D 7160, Jan. 1973.

[73] C. Ilinca, X. D. Zhang, J.-Y. Trépanier, and R. Camarero. A comparison of three error estimation techniques for finite-volume solutions of compressible flows. *Computer Methods in Applied Mechanics and Engineering*, 189(4):1277–1294, 2000.

[74] W. T. Jones. Gridex – an integrated grid generation package for cfd. AIAA Paper 2003–4129, 2003.

[75] W. T. Jones, E. J. Nielsen, and M. A. Park. Validation of 3d adjoint based error estimation and mesh adaptation for sonic boom prediction. AIAA Paper 2006–1150, 2006.

[76] I. Kalantari and G. McDonald. A data structure and an algorithm for the nearest point problem. *IEEE Transactions on Software Engineering*, SE-9(5):631–634, Sept. 1983.

[77] T. Kamatsuchi. Turbulent flow simulation around complex geometries with cartesian grid method. AIAA Paper 2007–1459, 2007.

[78] O. Kandil, I. Ozcer, X. Zheng, and P. Bobbitt. Comparison of full-potential propagation-code computations with the f-5e "shaped sonic boom experiment" program. AIAA Paper 2005–13, 2005.

[79] O. Kandil and I. A. Ozcer. Sonic boom computations for double-cone configuration using cfl3d, fun3d and full-potential codes. AIAA Paper 2006–414, 2006.

[80] O. A. Kandil, Z. Yang, and P. J. Bobbitt. Prediction of sonic boom signature using euler-full potential cfd with grid adaptation and shock fitting. AIAA Paper 2002–2542, 2002.

[81] B. K. Karamete, M. W. Beall, and M. S. Shephard. Triangulation of arbitrary polyhedra to support automatic mesh generators. *International Journal for Numerical Methods in Engineering*, 49(12):167–191, 2000.

[82] S. L. Karman, Jr. Splitflow: A 3d unstructured cartesian/prismatic grid cfd code for complex geometries. AIAA Paper 1995–343, 1995.

[83] H.-J. Kim and K. Nakahashi. Output-based error estimation and adaptive mesh refinement using viscous adjoint method. AIAA Paper 2006–1395, 2006.

[84] P. Labbé, J. Dompierre, , M.-G. Vallet, F. Guilault, and J.-Y. Trépanier. A universal measure of the conformity of a mesh with respect to an anisotropic metric field. *International Journal for Numerical Methods in Engineering*, 61(15):2675–2695, 2004.

[85] K. R. Laflin, S. M. Klausmeyer, and M. Chaffin. A hybrid computational fluid dynamics procedure for sonic boom prediction. AIAA Paper 2006–3168, 2006.

[86] P. Laug and B. H. The bl2d mesh generator: Beginer's guide, user's and programmer's manual. INRIA Report 0194, 1996.

[87] E. M. Lee-Rausch, M. A. Park, W. T. Jones, D. P. Hammond, and E. J. Nielsen. Application of a parallel adjoint-based error estimation and anisotropic grid adaptation for three-dimensional aerospace configurations. AIAA Paper 2005–4842, 2005.

[88] C. Y. Lepage and W. G. Habashi. Parallel unstructured mesh adaptation on distributed-memory systems. AIAA Paper 2004-2532, 2004.

[89] X. Li. *Sliver-Free Three Dimensional Delaunay Mesh Generation*. PhD thesis, University of Illinios and Urbana-Champaign, 2000.

[90] X. Li. *Mesh Modification Procedures for General 3-D Non-Manifold Domains*. PhD thesis, Rensselaer Polytecnic Institute, 2003.

[91] X. Li, M. S. Shephard, and M. W. Beall. Accounting for curved domains in mesh adaptation. *International Journal for Numerical Methods in Engineering*, 58(1):247–276, 2000.

[92] A. Liu and B. Joe. Relationship between tetrahedron shape measures. *BIT Numerical Mathematics*, 34(2):268–287, 1994.

[93] R. Löhner. A parallel advancing front grid generation scheme. *International Journal for Numerical Methods in Engineering*, 51(6):647–661, 2001. See also AIAA Paper 2000-1005.

[94] A. Loseille, A. Dervieux, P. Frey, and F. Alauzet. Achievement of global second order mesh convergence for discontinuous flows with adapted unstructured meshes. AIAA Paper 2007–4186, 2007.

[95] R. J. Mack and N. Kuhn. *Determination of Extrapolation Distance With Measured Pressure Signatures From Two Low-Boom Models*. NASA/TM-2004-213264, 2004.

[96] R. J. Mack and N. Kuhn. *Determination of Extrapolation Distance With Pressure Signatures Measured at Two to Twenty Span Lengths From Two Low-Boom Models*. NASA/TM-2006-214524, 2006.

[97] M. D. Madson. Sonic boom predictions using a solution-adaptive full-potential code. *AIAA Journal of Aircraft*, 31(1):57–63, Jan.–Feb. 1994. See also AIAA Paper 91–3278.

[98] D. J. Mavriplis. Turbulent flow calculations using unstructured and adaptive meshes. *International Journal for Numerical Methods in Fluids*, 13:1131–1152, 1991.

[99] D. J. Mavriplis. Unstructured mesh generation and adaptivity. 26th Computational Fluid Dynamics Lecture Series. VKI Lecture Series 1995-2, Mar. 1995. See also ICASE Report 95-26 and NASA-CR-195069.

[100] D. J. Mavriplis. Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes. *International Journal for Numerical Methods in Fluids*, 34(2):93–111, 2000. See also AIAA Paper 97-0857.

[101] D. J. Mavriplis. Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes. AIAA Paper 2003–3986, 2003.

[102] D. J. Mavriplis. Grid resolution study of a drag prediction workshop configuration using the nsu3d unstructured mesh solver. AIAA Paper 2005–4729, 2005.

[103] D. J. Mavriplis, M. J. Aftosmis, and M. Berger. High resolution aerospace applications using the nasa columbia supercomputer. In *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*, page 61, Washington, DC, USA, 2005. IEEE Computer Society.

[104] D. J. Mavriplis, J. C. Vassberg, E. N. Tinoco, M. Mani, O. P. Brodersen, B. Eisfeld, R. A. Wahls, J. H. Morrison, T. Zickuhr, D. Levy, and M. Murayama. Grid quality and resolution issues from the drag prediction workshop series. AIAA Paper 2008–930, 2008.

[105] J. P. Mendoza and R. M. Hicks. *Further studies of the extrapolation of near field overpressure data*. NASA-TM-X-2219, Mar. 1971.

[106] L. Michiels, J. Peraire, and A. Patera. A posteriori finite-element output bounds for the incompressible navierstokes equations: Application to a natural convection problem. *Journal Computational Physics*, 172(2):401–425, 2001.

[107] J. M. Morgenstern. Wind tunnel testing of a sonic boom minimized tail-braced wing transport configuration. AIAA Paper 2004–4536, 2004.

[108] J. D. Müller and M. B. Giles. Solution adaptive mesh refinement using adjoint error analysis. AIAA Paper 2001–2550, 2001.

[109] S. M. Murman, M. J. Aftosmis, and M. Nemec. Automated parameter studies using a cartesian method. AIAA Paper 2004-5076, 2004.

[110] M. Nemec and M. J. Aftosmis. Adjoint algorithm for cad-based shape optimization using a cartesian method. AIAA Paper 2005-4987, 2005.

[111] M. Nemec and M. J. Aftosmis. Aerodynamic shape optimization using a cartesian adjoint method and cad geometry. AIAA Paper 2006-3456, 2006.

[112] M. Nemec and M. J. Aftosmis. Adjoint error estimation and adaptive refinement for embedded-boundary cartesian meshes. AIAA Paper 2007-4187, 2007.

[113] M. Nemec, M. J. Aftosmis, and M. Wintzer. Adjoint-based adaptive mesh refinement for complex geometries. AIAA Paper 2008-725, 2008.

[114] E. J. Nielsen. *Aerodynamic Design Sensitivities on an Unstructured Mesh Using the Navier-Stokes Equations and a Discrete Adjoint Formulation.* PhD thesis, Virginia Polytechnic Institute and State University, 1998.

[115] E. J. Nielsen and W. K. Anderson. Recent improvements in aerodynamic design optimization on unstructured meshes. *AIAA Journal*, 40(6):1155–1163, 2002. See also AIAA Paper 2001–596.

[116] E. J. Nielsen and W. L. Kleb. Efficient construction of discrete adjoint operators on unstructured grids using complex variables. *AIAA Journal*, 44(4):827–836, 2006. See also AIAA Paper 2005–324.

[117] E. J. Nielsen, J. Lu, M. A. Park, and D. L. Darmofal. An implicit, exact dual adjoint solution method for turbulent flows on unstructured grids. *Computers and Fluids*, 33(9):1131–1155, 2004. See also AIAA Paper 2003–272.

[118] C. D. Norton, J. Z. Lou, and T. A. Cwik. Status and directions for the pyramid parallel unstructured amr library. In *Proceedings of the 15th International Parallel and Distributed Processing Symposium*, pages 1224–1231, Washington, DC, USA, 2001. IEEE Computer Society.

[119] I. A. Ozcer. Sonic boom prediction using euler/full-potential methodology. Master's thesis, Old Dominion University, Dec. 2005.

[120] I. A. Ozcer and O. Kandil. Fun3d / optigrid coupling for unstructured grid adaptation for sonic boom problems. AIAA Paper 2006–61, 2006.

[121] J. A. Page and K. J. Plotkin. An efficient method for incorporating computational fluid dynamics into sonic boom prediction. AIAA Paper 91–3275, 1991.

[122] M. A. Park. Three–dimensional turbulent rans adjoint–based error correction. AIAA Paper 2003–3849, 2003.

[123] M. A. Park. Adjoint–based, three–dimensional error prediction and grid adaptation. *AIAA Journal*, 42(9):1854–1862, Sept. 2004. See also AIAA Paper 2002–3286.

[124] Y. M. Park and O. J. Kwon. A parallel unstructured dynamic mesh adaptation algorithm for 3-d unsteady flows. *International Journal for Numerical Methods in Fluids*, 48(6):671–690, 2005. See also AIAA Paper 2001–865.

[125] J. Peiro, J. Peraire, and K. Morgan. Felisa reference manual and user's guide, volume i. University of Wales/Swansea Report CR/821/94, 1994.

[126] J. Peraire, J. Peirò, and K. Morgan. Adaptive remeshing for three-dimensional compressible flow computations. *Journal of Computational Physics*, 103(2):269–285, 1992.

[127] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz. Adaptive remeshing for compressible flow computations. *Journal of Computational Physics*, 72(2):449–466, 1987.

[128] N. A. Pierce and M. B. Giles. Adjoint recovery of superconvergent functionals from pde approximations. *SIAM Review*, 42(2):247–264, 2000.

[129] S. Z. Pirzadeh. A solution-adaptive unstructured grid method by grid subdivision and local remeshing. *AIAA Journal of Aircraft*, 37(5):818–824, September–October 2000. See also AIAA Paper 99–3255.

[130] K. J. Plotkin. State of the art of sonic boom modeling. *The Journal of the Acoustical Society of America*, 111(1):530–536, Jan. 2002.

[131] K. J. Plotkin and J. A. Page. Extrapolation of sonic boom signatures from cfd solutions. AIAA Paper 2002–922, 2002.

[132] W. R. Quadros, K. Shimada, and S. J. Owen. Skeleton-based computational method for the generation of a 3d finite element mesh sizing function. *Engineering with Computers*, 20(3):249–264, 2004.

[133] S. K. Rallabhandi and D. N. Mavris. A new approach for incorporating computational fluid dynamics into sonic boom prediction. AIAA Paper 2006–3312, 2006.

[134] R. Rannacher. Adaptive galerkin finite element methods for partial differential equations. *Journal of Computational and Applied Mathematics*, 128:205–233, 2001.

[135] R. D. Rausch, J. T. Batina, and H. T. Y. Yang. Spatial adaption procedures on unstructured meshes for accurate unsteady aerodynamic flow computation. AIAA Paper 1991-1106, 1991.

[136] K. Schloegel, G. Karypis, and V. Kumar. Parallel static and dynamic multi-constraint graph partitioning. *Concurrency and Computation: Practice and Experience*, 14(3):219–240, 2002.

[137] J. R. Shewchuk. Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In M. C. Lin and D. Manocha, editors, *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. Springer-Verlag, May 1996. From the First ACM Workshop on Applied Computational Geometry.

[138] J. R. Shewchuk. Adaptive precision floating-point arithmetic and fast robust geometric predicates. *Discrete & Computational Geometry*, 18(3):305–363, Oct. 1997.

[139] J. R. Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, 22(1–3):21–74, 2002.

[140] J. R. Shewchuk. What is a good linear element? interpolation, conditioning, and quality measures. In *11th International Meshing Roundtable*, Sept. 2002.

[141] F. Suerich-Gulick, C. Y. Lepage, and W. G. Habashi. Anisotropic 3-d mesh adaptation on unstructured hybrid meshes. AIAA Paper 2002-0859, 2002.

[142] C. L. Thomas. *Extrapolation of sonic boom pressure signatures by the waveform parameter method*. NASA/TN-D 6832, June 1972.

[143] R. Tilch, A. Tabbal, M. Zhu, F. Decker, and R. Löhner. Combination of body-fitted and embedded grids for external vehicle aerodynamics. AIAA Paper 2007–1300, 2007.

[144] B. van Leer. Flux-vector splitting for the euler equations. ICASE Report 82-30, 1982.

[145] D. A. Venditti. *Grid Adaptation for Functional Outputs of Compressible Flow Simulations*. PhD thesis, Massachusetts Institute of Technology, 2002.

[146] D. A. Venditti and D. L. Darmofal. Grid adaptation for functional outputs: Application to two-dimensional inviscid flows. *Journal Computational Physics*, 176:40–69, 2002. See also AIAA Paper 2000–2244.

[147] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: Application to two-dimensional viscous flows. *Journal Computational Physics*, 187:22–46, 2003.

[148] V. Venkatakrishnan. Convergence to steady state solutions of the euler equations on unstructured grids with limiters. *Journal of Computational Physics*, 118(1):120–130, 1995. See also AIAA Paper 93–880.

[149] K. A. Waithe. Application of usm3d for sonic boom prediction by utilizing a hybrid procedure. AIAA Paper 2008–129, 2008.

[150] J. Waltz. Parallel adaptive refinement for 3d unstructured grids. AIAA Paper 2003-1115, 2003.

[151] G. P. Warren, W. K. Anderson, J. L. Thomas, and S. L. Krist. Grid convergence for adaptive methods. AIAA Paper 91-1592, 1991.

[152] J. A. White. private communication, 2007.

[153] G. B. Whitham. The flow pattern of a supersonic projectile. *Communications on Pure and Applied Mathematics*, 5(3):301–348, Aug. 1952.

[154] G. B. Whitham. On the propagation of weak shock waves. *Journal of Fluid Mechanics*, 1(3):290–318, Sept. 1956.

[155] S. Yamakawa and K. Shimada. Anisotropic tetrahedral meshing via bubble packing and advancing front. *International Journal for Numerical Methods in Engineering*, 57(13):1923–1942, 2003.

[156] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, S. S. Samant, and J. E. Bussoletti. A locally refined rectangular grid finite element method: Application to computational fluid dynamics and computational physics. *Journal of Computational Physics*, 92(1):1–66, 1991.

[157] O. C. Zienkiewicz and J. Z. Zhu. Adaptivity and mesh generation. *International Journal for Numerical Methods in Engineering*, 35(4):783–810, 1991.