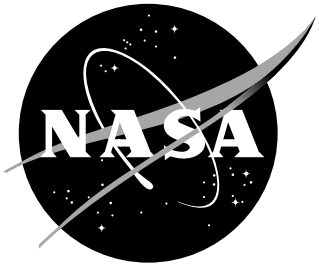
NAS Technical Report: NAS-2014-04



Toward Automatic Verification of Goal-Oriented Flow Simulations

Marian Nemec
Science & Technology Corp.

Michael J. Aftosmis
NASA Ames Research Center

Applied Modeling and Simulation Branch
Advanced Supercomputing Division

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, CA 94035

August 2014

Abstract

We demonstrate the power of adaptive mesh refinement with adjoint-based error estimates in verification of simulations governed by the steady Euler equations. The flow equations are discretized using a finite volume scheme on a Cartesian mesh with cut cells at the wall boundaries. The discretization error in selected simulation outputs is estimated using the method of adjoint-weighted residuals. Practical aspects of the implementation are emphasized, particularly in the formulation of the refinement criterion and the mesh adaptation strategy. Following a thorough code verification example, we demonstrate simulation verification of two- and three-dimensional problems. These involve an airfoil performance database, a pressure signature of a body in supersonic flow and a launch abort with strong jet interactions. The results show reliable estimates and automatic control of discretization error in all simulations at an affordable computational cost. Moreover, the approach remains effective even when theoretical assumptions, e.g., steady-state and solution smoothness, are relaxed.

Nomenclature

A	Face area
C	Constant used in estimating discretization error, $C = 2$ or $\frac{4}{3}$
$\mathbf{d}$	Distance vector from cell centroid to face centroid
$\mathcal{E}$	Discretization error
E	Total energy per unit mass
e	Discretization error with respect to a uniformly refined mesh
$\mathbf{F}$	Inviscid flux tensor
$\mathbf{G}$	Numerical flux function
J	Scalar functional or output, e.g., lift coefficient
$\hat{\mathbf{n}}$	Outward pointing unit normal
N	Number of cells
$\mathbf{P}$	Prolongation operator or matrix
p	Pressure
$\mathbf{Q}$	Flow solution vector of conservative variables $[\rho, \rho u, \rho v, \rho w, \rho E]^{\mathrm{T}}$
$\mathbf{R}$	Vector of residuals
t	Time
$\mathbf{U}$	Flow solution vector of primitive variables $[\rho, u, v, w, p]^{\mathrm{T}}$
u, v, w	Cartesian components of velocity
V	Volume

Greek letters

η	Error indicator
ρ	Density
ψ	Adjoint vector

Superscripts

a	Adjoint
H	Data reconstruction from mesh with characteristic cell-size H

Subscripts

H	Discretization on mesh with characteristic cell-size H
h	Discretization on mesh with characteristic cell-size $h = \frac{1}{2}H$
c	Adjoint correction
L	Linear interpolant
TL	Trilinear interpolant
TQ	Triquadratic interpolant
w	Wall
∞	Freestream

There is no doubt that mesh adaptation can lead to significant improvement in solution accuracy. ... What remains in doubt is whether the current methods of mesh adaptation can be brought to a sufficient level of reliability and robustness for routine use as a predictive tool.

T. Baker, 1997 [1]

1 Introduction

Despite significant progress in solution-adaptive mesh refinement [1, 2, 3, 4, 5], verification of flow simulations remains largely a manual procedure that requires expert guidance [6, 7, 8]. Most of the time is consumed by crafting a computational mesh, checking the results and refining the mesh to assess and control discretization error[a]. As the number and complexity of simulations increase (for example consider flight vehicle performance databases involving 10^3–10^5 cases), manual simulation verification becomes impractical. Instead, simulations are verified for only a subset of cases and involve "best-practice" guidelines, which do not guarantee that the results comply with the expected standards of accuracy [9, 10].

The numerical accuracy of flow simulations depends inextricably on discretization error. In other words, numerical inaccuracy is a consequence of the mesh. Establishing credibility for complex simulations, therefore, requires that mesh generation and error estimation be integral parts of each simulation. As the mesh and flow solution evolve, a systematic reduction in the discretization error is achieved through use of error estimates derived from the flow solution on the current mesh. The benefit is a straightforward quantitative assessment of convergence because a mesh refinement study is intrinsic to every case.

The goal of most engineering simulations is to predict a handful of outputs, for example aerodynamic forces and moments. In such goal-oriented simulations, it is most efficient to focus on discretization error directly affecting the outputs of interest. For example, even the simple problem of predicting the span efficiency factor of an isolated wing in subsonic inviscid flow becomes prohibitively expensive if the mesh is refined to follow the tip vortex far downstream. As the influence of the vortex on span efficiency decreases with downstream distance, so should the cell refinement. Experience with error estimates that do not target outputs, in particular direct residual or truncation-error estimates, indeed shows that the adapted meshes are frequently inferior to those crafted by experts who understand the goals of the simulation.

Remarkably, a relatively straightforward modification to residual error estimates allows the prediction of error in outputs: the cell-wise

[a]The subject of validation (modeling error) is not considered and in our experience, discretization errors dominate other aspects of simulation verification such as iterative convergence.

residuals are weighted by their influence on the output. This is the idea behind the method of adjoint-weighted residuals, where the weights are obtained from the solution of an adjoint equation. The result is not only an estimate of error in the outputs due to discretization (for example the error in lift), but also a cell-wise error indicator to guide mesh refinement. The adjoint-weighted approach was first developed within the framework of the finite element method [11, 12] and extended to finite volume methods by Giles and Pierce [13], Barth [14], and Venditti and Darmofal [15]. The approach has been steadily refined to improve its accuracy and efficiency [16, 17, 18, 19, 20, 21, 22], and has been used successfully to establish the credibility of goal-oriented simulations [23, 24, 25, 26, 27].

The routine use of adjoint-based error estimates for automatic simulation verification, however, is predicated on robust mesh generation. This is because failures in mesh generation often require expert intervention to resolve. If a typical simulation requires five to ten adaptation cycles to attain sufficient output accuracy, then the construction of an aerodynamic performance database for a moderate range of operating conditions may invoke the mesh generator ten thousand times. Therefore, the mesh generator must be fast and failsafe for automatic verification to be viable in an engineering environment. One such robust approach is the embedded-boundary (cut-cell) method, where the mesh is constructed by embedding the geometry in a regular lattice of hexahedral (Cartesian) or tetrahedral elements [28, 29, 30, 31, 23, 32, 33].

The purpose of this paper is to demonstrate that the combination of adjoint-based error estimates with a Cartesian cut-cell method is a practical approach for automatic verification of steady, goal-oriented simulations. The paper covers the development and implementation of a simulation verification framework previously described in [34, 35, 36] with additional details and improvements. The framework uses the approach of Venditti and Darmofal [15] to formulate reliable error estimates and the approach of Aftosmis and Berger [37] for incremental refinement of nested Cartesian cut-cell meshes. The framework emphasizes robustness and efficiency, in terms of both execution speed and memory requirements, because both are central considerations in engineering and decision-making.

We begin with a brief review of discretization error in Sec. 2. An estimate of the error in user-selected outputs is expressed in terms of adjoint-weighted residuals using the algebraic formulation of [15]. Section 3 presents the salient features of the flow solver, the formulation of the discrete adjoint equation and the implementation of the adjoint solver. Section 4 explains the details of the error estimation procedure and the formulation of a robust refinement criterion. The simulation verification framework is presented in Sec. 5, including a discussion of the adaptation mechanics, error control and practical aspects of the implementation. The results are organized in two parts. Section 6 presents a code verification example that establishes the accuracy of the error estimate for guiding mesh refinement. Section 7 demonstrates examples of automatic

simulation verification on a sequence of three problems of increasing difficulty. Additional examples of simulation verification through use of this framework can be found in [38, 39, 40, 41, 42, 43, 44, 45].

2 Error Estimates

2.1 Discretization Error

Our goal is to compute a reliable approximation of a scalar output functional $J(\mathbf{Q})$, for example lift or drag, derived from a flow solution $\mathbf{Q}$ that satisfies the flow equations

$$\mathbf{R}(\mathbf{Q}) = \mathbf{0} \tag{1}$$

such as the Euler or Navier–Stokes equations. To compute a discrete approximation of the functional $J_H(\mathbf{Q}_H)$, the domain is tessellated into N control volumes with characteristic cell-size H, which we call the "working" mesh. The flow equations are discretized and solved to satisfy a system of modified partial differential equations

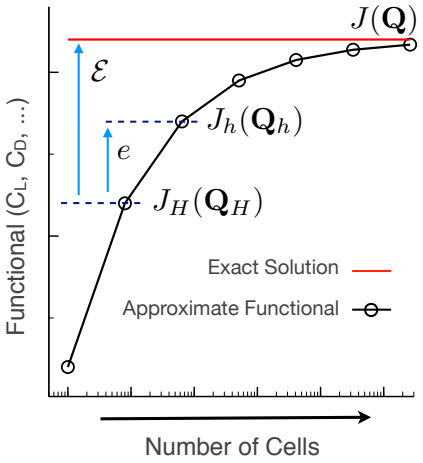


Figure 1. Example of a uniform mesh refinement study showing the definition of discretization error, $\mathcal{E}$, and the error relative to the next mesh, e.

$$\mathbf{R}_H(\mathbf{Q}_H) = \mathbf{0} \tag{2}$$

where $\mathbf{Q}_H = [\mathbf{Q}_1, \mathbf{Q}_2, \ldots, \mathbf{Q}_N]^{\mathrm{T}}$ is the discrete flow solution vector, e.g., an algebraic vector of cell-average values, and the discrete operator $\mathbf{R}_H$ represents the residual vector. Similarly, J_H represents the discrete operator used to evaluate scalar functionals, e.g., the integration of pressure to obtain lift given the flow solution on the working mesh $\mathbf{Q}_H$.

The error in the functional due to discretization is

$$\mathcal{E} = |J(\mathbf{Q}) - J_H(\mathbf{Q}_H)| \tag{3}$$

This is illustrated in Fig. 1, which shows a notional mesh refinement study on a sequence of nested, uniformly refined meshes. We assume that errors not related to H, such as distance to farfield, are negligible. If the discretization is consistent, then the approximation $J_H(\mathbf{Q}_H)$ converges to the exact solution $J(\mathbf{Q})$ as the number of cells in the computational domain is increased and the characteristic cell-size H shrinks. In practice, direct computation of the discretization error is difficult because it involves the analytic partial differential equations, Eq. 1.

An alternate approach is to compute the error relative to the functional on the next nested mesh

$$e = |J_h(\mathbf{Q}_h) - J_H(\mathbf{Q}_H)| \tag{4}$$

as shown in Fig. 1. We refer to the next uniformly refined mesh as the "embedded" mesh with characteristic cell-size h. Assuming that the problem is smooth, the discretization error $\mathcal{E}$ can be expressed as a geometric series in e. For example, the error expression for a functional with second-order convergence is

$$\mathcal{E} = \sum_{i=0}^{\infty} \frac{1}{4^i} e = \frac{4}{3} e \tag{5}$$

and $\mathcal{E} = 2e$ for first-order functionals[b]. This trades the need for the exact solution $J(\mathbf{Q})$ for the requirement that the functional be in the asymptotic range with a known convergence rate. Put another way, the starting mesh should be sufficiently fine. The key step becomes approximating $J_h(\mathbf{Q}_h)$ without solving on the embedded mesh.

2.2 Method of Adjoint-Weighted Residuals

To derive a reliable approximation of the functional $J_h(\mathbf{Q}_h)$, consider its truncated Taylor series expansion about the working-mesh solution

$$J_h(\mathbf{Q}_h) \approx J_h(\mathbf{Q}_h^H) + \frac{\partial J_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h} \left(\mathbf{Q}_h - \mathbf{Q}_h^H\right) \tag{6}$$

The algebraic vector $\mathbf{Q}_h^H$ denotes a reconstruction of the flow solution from the working mesh to the embedded mesh via a prolongation operator, $\mathbf{Q}_h^H = \mathbf{P}\mathbf{Q}_H$. The term $J_h(\mathbf{Q}_h^H)$ is the evaluation of the functional using the reconstructed flow solution on the embedded mesh, e.g., lift computation using the reconstructed state and finer boundary resolution. This is usually straightforward. The challenge is the explicit dependence on $\mathbf{Q}_h$ in the inner-product term of Eq. 6.

To eliminate $\mathbf{Q}_h$, expand the residual equation to obtain

$$\mathbf{R}_h(\mathbf{Q}_h) = \mathbf{0} \approx \mathbf{R}_h(\mathbf{Q}_h^H) + \frac{\partial \mathbf{R}_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h} \left(\mathbf{Q}_h - \mathbf{Q}_h^H\right) \tag{7}$$

Note that Eqs. 6 and 7 are approximate. The derivation can be made exact through use of the mean-value linearization; however, this only defers the use of similar approximations to obtain computable error estimates. Combining Eqs. 6 and 7 gives

$$J_h(\mathbf{Q}_h) \approx J_h(\mathbf{Q}_h^H) - \frac{\partial J_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h} \left[\frac{\partial \mathbf{R}_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h}\right]^{-1} \mathbf{R}_h(\mathbf{Q}_h^H) \tag{8}$$

[b]The sum of a geometric series $1 + r + r^2 + \ldots$ is $\frac{1}{1-r}$ if $|r| < 1$. The common ratio r is 1/4 for second-order functionals and 1/2 for first-order functionals.

which is independent of $\mathbf{Q}_h$. The adjoint equation is obtained from Eq. 8 by defining the following intermediate product

$$\left[\frac{\partial \mathbf{R}_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h}\right]^{\mathrm{T}} \psi_h = \frac{\partial J_h(\mathbf{Q}_h^H)}{\partial \mathbf{Q}_h}^{\mathrm{T}} \tag{9}$$

where the vector ψ denotes the adjoint variables. Rewriting Eq. 8 with the adjoint variables

$$J_h(\mathbf{Q}_h) \approx J_h(\mathbf{Q}_h^H) - \psi_h^{\mathrm{T}}\ \mathbf{R}_h(\mathbf{Q}_h^H) \tag{10}$$

reveals that the adjoints weight the residual errors to form a correction term to approximate the functional on the embedded mesh. Substituting Eq. 10 into Eq. 4, the error expression (Eq. 5) becomes

$$\mathcal{E} \approx C|J_h(\mathbf{Q}_h^H) - \psi_h^{\mathrm{T}}\ \mathbf{R}_h(\mathbf{Q}_h^H) - J_H(\mathbf{Q}_H)| \tag{11}$$

where the constant $C = \frac{4}{3}$ for second-order functionals and $C = 2$ for first-order functionals.

While Eq. 11 is independent of $\mathbf{Q}_h$, it does require the solution of the adjoint equation on the embedded mesh (ψ_h of Eq. 9). This is impractical because a solution of the large, linear adjoint system can be nearly as expensive as a nonlinear flow solution. Various strategies exist to circumvent this difficulty. The initial step involves solving the adjoint system on the working mesh

$$\left[\frac{\partial \mathbf{R}_H(\mathbf{Q}_H)}{\partial \mathbf{Q}_H}\right]^{\mathrm{T}} \psi_H = \frac{\partial J_H(\mathbf{Q}_H)}{\partial \mathbf{Q}_H}^{\mathrm{T}} \tag{12}$$

This solution is then prolonged to the embedded mesh to estimate ψ_h. The estimate can be sharpened by additional (implementation specific) procedures, such as relaxation. To explain the salient features of our approach, we first introduce the governing equations and the numerical method, and then return to evaluation of Eq. 11 in Sec. 4.

3 Governing Equations and Numerical Method

3.1 Flow Equations

We solve the three-dimensional Euler equations governing compressible flow of a perfect gas. For a finite region of space with volume V and surface area A, the integral form of the Euler equations is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \mathbf{Q}\ \mathrm{d}V + \oint_A \mathbf{F}\cdot\hat{\mathbf{n}}\ \mathrm{d}A = \mathbf{0} \tag{13}$$

where $\mathbf{Q} = [\rho, \rho u, \rho v, \rho w, \rho E]^{\mathrm{T}}$, $\mathbf{F}$ is the inviscid flux tensor and $\hat{\mathbf{n}}$ is the outward facing unit normal vector.

The Euler equations are solved with a finite-volume method on a regular Cartesian mesh with embedded boundaries. The body geometry is specified by a watertight surface triangulation. The volume mesh consists of hexahedral cells, except for a layer of body-intersecting cells, or cut cells, which are arbitrary polyhedra adjacent to the boundaries, as illustrated in Fig. 2. The mesh is viewed as an unstructured collection of control volumes to facilitate solution-adaptive refinement.

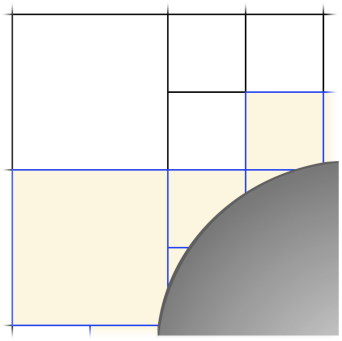


Figure 2. A multilevel Cartesian mesh with a cut-cell boundary. Adjacent cells cannot exceed 2:1 ratio.

Spatial discretization uses a cell-centered approach, where the control volumes V correspond to the mesh cells and the cell-averaged value of $\mathbf{Q}$, denoted by $\mathbf{Q}_H$, is located at the centroid of each cell. The control volumes are fixed in time. The semi-discrete form is given by

$$\mathbf{V}_H \frac{\mathrm{d}\mathbf{Q}_H}{\mathrm{d}t} + \mathbf{R}_H(\mathbf{Q}_H) = \mathbf{0} \tag{14}$$

where $\mathbf{V}_H$ is a diagonal matrix containing the cell volumes. The residual in each cell i is expressed as

$$\mathbf{R}_i = \sum_{\mathrm{j}\in\mathrm{V_i}} \mathbf{G}_j \cdot \hat{\mathbf{n}}_j A_j \tag{15}$$

where j denotes the jth face of volume V_i with area A, and $\mathbf{G}$ represents the numerical flux function.

Residual evaluation for a second-order accurate discretization proceeds by linearly reconstructing the solution to the face centroid. This is illustrated in Fig. 3, for two neighboring Cartesian cells l, r sharing a common face. Primitive variables, $\mathbf{U} = [\rho, u, v, w, p]^{\mathrm{T}}$, are used for the reconstruction, and the left and right states are given by

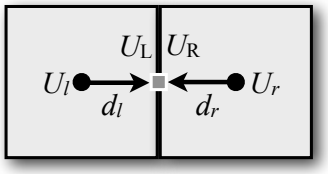


Figure 3. Reconstruction of face midpoint value from cell centroids

$$\mathbf{U}_{\mathrm{L}} = \mathbf{U}_l + \mathbf{d}_{\mathrm{l}}\,\phi_l\,\nabla\mathbf{U}_l \quad \mathbf{U}_{\mathrm{R}} = \mathbf{U}_r - \mathbf{d}_{\mathrm{r}}\,\phi_r\,\nabla\mathbf{U}_r \tag{16}$$

Here $\mathbf{d}_l$ and $\mathbf{d}_r$ are the distance vectors from the cell centroids to the face centroid, $\nabla\mathbf{U}$ is the solution gradient determined via a linear least-squares procedure and ϕ is a vector of slope limiter values used to directionally enforce monotonic solutions [46]. The flux value at the face centroid is obtained via the flux-vector splitting approach of van Leer [47]

$$\mathbf{G}(\mathbf{U}_{\mathrm{L}}, \mathbf{U}_{\mathrm{R}}) = \mathbf{f}^{+}(\mathbf{U}_{\mathrm{L}}) + \mathbf{f}^{-}(\mathbf{U}_{\mathrm{R}}) \tag{17}$$

At the implementation level, the assembly of the residual vector is accomplished by a loop over the faces of the mesh. The flux contributions are scattered from the face and accumulated in the cells

$$\mathbf{R}_l = \mathbf{R}_l + \mathbf{G}A \quad \mathbf{R}_r = \mathbf{R}_r - \mathbf{G}A \tag{18}$$

where the sign reflects the change in the direction of the outward-pointing normal.

All boundary conditions are enforced weakly. At the wall, zero normal velocity is enforced by specifying a wall flux. This flux is non-zero only for the momentum components and uses the pressure at the wall centroid of each cut cell (p_w)

$$\mathbf{G}_{\mathrm{w}} = (0, p_{\mathrm{w}}, p_{\mathrm{w}}, p_{\mathrm{w}}, 0)^{\mathrm{T}} \tag{19}$$

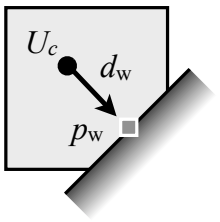


Figure 4. Linear reconstruction of pressure to face centroid at the wall

as shown in Fig. 4. Linear reconstruction, Eq. 16, is used to compute p_w. In the farfield, the flux function, Eq. 17, is used to compute the flux across faces on the boundary. The boundary state (either $\mathbf{U}_{\mathrm{L}}$ or $\mathbf{U}_{\mathrm{R}}$) is set via Riemann invariants and linear reconstruction, Eq. 16, is used for the interior state.

Steady-state solutions are obtained with a five-stage Runge–Kutta scheme accelerated by local time stepping and full approximation storage multigrid. The multigrid residual restriction operator is the sum of the residuals of the fine mesh cells enclosed by the coarse cell, while the prolongation operator is direct injection. Time-to-solution is further reduced by parallel computing using a highly-scalable domain decomposition scheme. For details on mesh generation and the flow solution algorithm, see Aftosmis *et al.* [48, 28, 49, 50] and Berger *et al.* [51].

We consider two classes of functionals as primary outputs of interest. The first are aerodynamic performance coefficients, such as coefficients of lift and drag, given by

$$J = \frac{1}{q_\infty \; A_{\mathrm{ref}}} \int_{\mathrm{w}} (\hat{\mathbf{n}} \cdot \xi)(p_w - p_\infty) \; dA \tag{20}$$

where q_∞ is the freestream dynamic pressure, A_{ref} is the reference area, p_∞ is the freestream pressure and ξ is the appropriate projection for the coefficient of interest, e.g., $\xi \perp \mathbf{V}_\infty$ for lift. The discretization of Eq. 20 uses midpoint quadrature and is summed over all cut cells. The second class are field functionals that can be specified anywhere in the computational domain. An example is a line sensor for pressure

$$J_l = \int_0^L \left(\frac{p - p_\infty}{p_\infty} \right)^n \; d\ell \tag{21}$$

where L is the length of the line and n is a user specified exponent (usually 1 or 2). The evaluation of line sensors involves finding the set of cells intersected by the line and integrating with midpoint quadrature that uses linear reconstruction of pressure to the line-segment midpoint inside each intersected cell.

3.2 Discrete Adjoint Equation

The adjoint equation as derived in Eqs. 9 and 12 is in discrete form, i.e. the discretized residual and functional operators are linearized. This is a consequence of computing the error relative to the embedded mesh, Eq. 4, instead of the exact solution, Eq. 3. For a reliable estimate of error via Eq. 11, the discrete adjoint solution must converge as the mesh is refined

$$\lim_{h \to 0} \ \psi_h \to \psi \tag{22}$$

where ψ is the solution of the analytic adjoint equation obtained by linearizing the flow equations, Eq. 13, and functional before discretization. In other words, the discretization of the flow equations and functional must yield an asymptotically consistent adjoint discretization[c].

Adjoint consistency for functionals that involve wall-boundary integrals, such as Eq. 20, is prescribed by the form of the wall flux. Referring to Eq. 19, the transpose of the wall-flux Jacobian is column-rank deficient because the flux is zero for both the continuity and energy equations. For example, the adjoint system at the wall is given by

$$\begin{bmatrix} 0 & \hat{n}_x \partial p_\rho & \hat{n}_y \partial p_\rho & 0 \\ 0 & \hat{n}_x \partial p_u & \hat{n}_y \partial p_u & 0 \\ 0 & \hat{n}_x \partial p_v & \hat{n}_y \partial p_v & 0 \\ 0 & \hat{n}_x \partial p_p & \hat{n}_y \partial p_p & 0 \end{bmatrix} \begin{bmatrix} \psi^1 \\ \psi^2 \\ \psi^3 \\ \psi^4 \end{bmatrix} = \begin{bmatrix} \partial J_\rho \\ \partial J_u \\ \partial J_v \\ \partial J_p \end{bmatrix} \tag{23}$$

where for clarity we assumed two dimensions, first-order discretization ($p_w = p$) and linearized with respect to primitive variables. This reduces to

$$\hat{n}_x \psi^2 + \hat{n}_y \psi^3 = \partial J_p \tag{24}$$

which is a well-known analytic adjoint boundary condition derived by recognizing that any variation in wall-normal velocity is zero. Moreover, Eq. 23 shows that the boundary functional should be a function of only pressure, $J = J(p)$. There are no similar restrictions on field functionals; these may be a function of any flow variable.

At the farfield boundary, linearization of the flux function, Eq. 17, in conjunction with the Riemann invariants yields an inconsistent adjoint discretization for subsonic freestream conditions [52]. Since this should have little impact on functional accuracy if the distance to the farfield is

[c] For a definition of adjoint consistency see [52, 53]; here we use this term only in the sense of Eq. 22.

sufficiently large, we leave the residuals unchanged. For an example of a duality preserving formulation see [53]. To avoid pollution of the error estimates, we omit values from cells adjacent to the farfield boundary and their face neighbors (two layers of cells).

3.3 Adjoint Solver

The solution of the adjoint equation proceeds by introducing an unsteady term in Eq. 12 to obtain the following semi-discrete form

$$\mathbf{V}_H \frac{\mathrm{d}\psi_H}{\mathrm{d}t} + \mathbf{R}^{\mathrm{a}} = \mathbf{0} \tag{25}$$

where the adjoint residual vector is given by

$$\mathbf{R}^{\mathrm{a}} = \frac{\partial \mathbf{R}_H}{\partial \mathbf{Q}_H}^{\mathrm{T}} \psi_H - \frac{\partial J_H}{\partial \mathbf{Q}_H}^{\mathrm{T}} \tag{26}$$

An important consideration for adjoint solvers is memory usage relative to the flow solver. In general, since the adjoint solver is required to run on the working mesh H, the large matrix-vector product $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^{\mathrm{T}} \psi$ introduces a complication. The flow Jacobian $\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}$ is a sparse matrix that is constant during the adjoint solution procedure. Its non-zero entries can be precomputed and stored, thereby minimizing the time to compute the matrix-vector product at each iteration. This is an effective strategy when dealing with implicit flow solvers that already store the flow Jacobian [54]. Alternatively, when dealing with explicit solvers some or all entries of the flow Jacobian can be recomputed when forming the matrix-vector product, see for example Barth [55], Giles *et al.* [56], Nielsen *et al.* [57] and Mavriplis [58].

We adopt the explicit flow-solution method outlined in Sec. 3.1 for the solution of the adjoint system. The matrix-vector product is recomputed on-the-fly at each evaluation of the residual by reusing the face-based approach of the flow solver. To demonstrate, consider first-order discretization and examine the update of the flow-solver residual vector from an arbitrary face of the mesh as shown in Fig. 3 and Eq. 18

$$\mathbf{R}_H = \begin{bmatrix} \vdots \\ +\mathbf{G}A \\ -\mathbf{G}A \\ \vdots \end{bmatrix} \begin{matrix} \\ \leftarrow \text{ cell } l \\ \leftarrow \text{ cell } r \\ \\ \end{matrix} \tag{27}$$

Linearize and apply the transpose operator to obtain

$$\frac{\partial \mathbf{R}}{\partial \mathbf{Q}}^{\mathrm{T}} \psi = \begin{bmatrix} \ddots & & & \\ & A\frac{\partial \mathbf{G}}{\partial \mathbf{Q}_l}^{\mathrm{T}} & -A\frac{\partial \mathbf{G}}{\partial \mathbf{Q}_l}^{\mathrm{T}} & \\ & A\frac{\partial \mathbf{G}}{\partial \mathbf{Q}_r}^{\mathrm{T}} & -A\frac{\partial \mathbf{G}}{\partial \mathbf{Q}_r}^{\mathrm{T}} & \\ & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ \psi_l \\ \psi_r \\ \vdots \end{bmatrix} \tag{28}$$

Therefore, the adjoint residual update when sweeping over the faces of the mesh in a fashion analogous to the flow solver is

$$\mathbf{R}_l^{\mathrm{a}} = \mathbf{R}_l^{\mathrm{a}} + \frac{\partial \mathbf{f}^+}{\partial \mathbf{Q}_l}^{\mathrm{T}} A \left(\psi_l - \psi_r\right) \tag{29}$$

$$\mathbf{R}_r^{\mathrm{a}} = \mathbf{R}_r^{\mathrm{a}} + \frac{\partial \mathbf{f}^-}{\partial \mathbf{Q}_r}^{\mathrm{T}} A \left(\psi_l - \psi_r\right) \tag{30}$$

where we include the linearization of the split fluxes of Eq. 17. Note that the transpose reverses the operator order of the flow solution procedure. For example, linearizing one of the split fluxes

$$\frac{\partial \mathbf{f}^+}{\partial \mathbf{Q}_l}^{\mathrm{T}} = \left(\frac{\partial \mathbf{U}_l}{\partial \mathbf{Q}_l}\right)^{\mathrm{T}} \left(\frac{\partial \mathbf{f}^+}{\partial \mathbf{U}_l}\right)^{\mathrm{T}} \tag{31}$$

shows that the flux-function linearization is evaluated before the adjoint of the transformation from conservative to primitive variables.

For second-order spatial discretization, the adjoint of the reconstruction procedure, Eq. 16, requires an additional pass over the faces of the mesh. Linearization of the flow gradient involves only the geometry-dependent least-squares weights, which are already computed and stored by the flow solver. Gradient linearization is omitted in cut cells with volume fractions less than single-precision machine epsilon to avoid spurious adjoint values similar to those observed in [59]. Furthermore, the linearization assumes that the limiter values, ϕ in Eq. 16, are independent of the flow solution. In other words, the limiter is treated as a constant. Although this is mostly a pragmatic choice, in [34] we showed that the impact of this simplification on the accuracy of the linearization is relatively small. The right-hand-side of Eq. 12 is the linearization of the functional, i.e. Eq. 20 or 21. This linearization involves the pressure reconstruction procedure of Eq. 16 and is exact except for the treatment of the limiter.

Since the eigenvalues of the flow Jacobian matrix are not changed by the transpose operator, we expect Eq. 25 to have similar stability properties as the flow equations, Eq. 14. Convergence to steady-state is accomplished using the same five-stage Runge–Kutta time marching and multigrid schemes of the flow solver. Giles [60, 61] derived conditions for Runge–Kutta time marching schemes and multigrid that ensure the same asymptotic convergence rate of the flow and adjoint solvers. This duality-preserving algorithm is implemented almost automatically, since the flow solver's residual prolongation operator is a transpose of the restriction operator.

Overall, the CPU time per iteration of the adjoint solver is roughly equivalent to the flow solver. This is because the additional cost of re-evaluating the matrix-vector product at each adjoint iteration is offset by reusing the local time step and limiter values directly from the flow solver. The implementation results in only a slight increase in memory usage

over the flow solver due to the storage of the converged flow solution and its gradient. Moreover, the face-based data structures and the domain decomposition scheme of the flow solver are reused with only minor modifications, see [34] for details.

4 Error Estimation

With flow and adjoint solutions in hand on the working mesh H, we return to Eq. 10 to estimate the functional on the embedded mesh h. The embedded mesh is constructed explicitly and contains about $8N$ cells[d]. Computation of the residual $\mathbf{R}_h(\mathbf{Q}_h^H)$ involves the reconstruction of the flow solution on the embedded mesh from the working mesh data $\mathbf{Q}_h^H = \mathbf{P}\mathbf{Q}_H$ (recall Eq. 6). The value in each embedded cell is obtained from its parent cell by linear reconstruction, Eq. 16. This is denoted by $\mathbf{Q}_\mathrm{L} = \mathbf{P}_\mathrm{L}\mathbf{Q}_H$, where $\mathbf{P}_\mathrm{L}$ represents the linear prolongation operator. No special treatment is performed at mesh refinement boundaries and cut cells, where irregular stencils pollute the residual on the embedded mesh. Instead, we rely on the adjoint weights to attenuate the residuals together with filters that compensate for these numerical artifacts when tagging cells for refinement (described in Sec. 5.3).

Several studies [62, 15] show that higher-order reconstruction is more accurate for evaluating the residual on the embedded mesh. Nevertheless, we find that it is difficult to match the robustness of linear reconstruction in practical applications with shocks and other strong non-linearities. Moreover, the implementation is straightforward since we can reuse functions from the existing flow-solver code.

An estimate for ψ_h in Eq. 10 is obtained through use of the adjoint solution from the working-mesh. Similar to $\mathbf{Q}_h^H$, let ψ_h^H represent a reconstructed adjoint solution of Eq. 12 and rewrite Eq. 10 to obtain

$$J_h(\mathbf{Q}_h) \approx J_h(\mathbf{Q}_h^H) - \underbrace{(\psi_h^H)^\mathrm{T}\,\mathbf{R}_h(\mathbf{Q}_\mathrm{L})}_{\text{Adjoint Correction}} - \underbrace{(\psi_h - \psi_h^H)^\mathrm{T}\,\mathbf{R}_h(\mathbf{Q}_\mathrm{L})}_{\text{Remaining Error}} \tag{32}$$

This manipulation yields a computable *adjoint-correction* term that is generally non-zero in finite-volume methods. It can be used directly to obtain a better estimate of the functional or to obtain an error estimate via Eq. 11, as long as the last term, the *remaining error*, is small. This is likely, because the remaining error is a higher-order term. Moreover, its magnitude can be controlled with adaptive mesh refinement. Note that we tacitly assume that all higher derivatives of the functional and residual equations are also small. The crux becomes finding a reliable estimate of $\psi_h - \psi_h^H$ to formulate a robust adaptation criterion.

We use a trilinear interpolant for constructing ψ_h^H and a triquadratic interpolant for approximating ψ_h. These interpolants are based on shape

[d] Nested subdivision of a hexahedron creates eight embedded hexahedra, but at cut cells some children may be inside the geometry.

functions for "brick" elements commonly used in finite-element methods. The use of interpolation for ψ_h is a compromise between accuracy, cost, and factors related to implementation and maintenance. On the one hand, interpolation reduces the quality of the remaining-error estimate because it models the solution error $\psi_h - \psi_H$ with an interpolation error that primarily detects solution non-linearity. On the other hand, this approach maximizes speed since the number of arithmetic operations on the embedded mesh is relatively small. Furthermore, in contrast to the flow reconstruction, the adjoint reconstruction is not followed by residual evaluation, which relaxes robustness constraints in the implementation.

Before constructing the interpolants, the adjoint solution is linearly reconstructed from the centroid to the vertices of each cell on the working mesh (including cut cells). We adopt Eq. 16 and each vertex receives contributions from all its coincident cells. The average of all contributions determines the vertex solution value. Put another way, this is a data smoothing step. Special logic is implemented at mesh refinement boundaries, where the hanging vertices of small cells need additional updates from their big-cell neighbors. The solution at the eight vertices of each working-mesh hexahedron is used to form a unique trilinear polynomial

$$\psi_{\mathrm{TL}} = c_0 + c_1 x + c_2 y + c_3 z + c_4 xy + c_5 xz + c_6 yz + c_7 xyz \tag{33}$$

The triquadratic reconstruction operator is given by

$$\begin{aligned}\psi_{\mathrm{TQ}} = c_0 + c_1 x + c_2 y + c_3 z + c_4 xy + c_5 xz + c_6 yz + c_7 xyz + c_8 x^2 \\ + c_9 y^2 + c_{10} z^2 + c_{11} x^2 y + c_{12} x^2 z + c_{13} xy^2 + c_{14} xz^2 + c_{15} y^2 z \\ + c_{16} yz^2 + c_{17} x^2 yz + c_{18} xy^2 z + c_{19} xyz^2\end{aligned} \tag{34}$$

To determine the 20 unknown coefficients, we use the eight solution values (from the trilinear case) in conjunction with the solution gradient at the vertices. The gradient value at a vertex is determined by the arithmetic average of all gradients from cells common to the vertex. The Barth-Jespersen limiter [63] is used to prevent oscillatory reconstruction. The resulting over-determined system of 32 equations is solved in a least-squares sense. A well-behaved triquadratic interpolant is ensured by the addition of safeguards. These involve monitoring solution differences between the triquadratic, trilinear and cell-centroid values, and using the lower-order values when large differences are detected.

We split Eq. 32 into an estimate for the corrected functional

$$J_c = J_h(\mathbf{Q}_{\mathrm{L}}) - \psi_{\mathrm{TQ}}^{\mathrm{T}} \mathbf{R}_h(\mathbf{Q}_{\mathrm{L}}) \tag{35}$$

and a cell-wise estimate of the remaining error in each cell of the working mesh

$$\boldsymbol{\eta}_H = \sum_{j \in \mathrm{V}_i} \left(\psi_{\mathrm{TQ}} - \psi_{\mathrm{TL}}\right)^{\mathrm{T}} \mathbf{R}_h(\mathbf{Q}_{\mathrm{L}})_j \tag{36}$$

where j denotes the j^{th} child of parent cell V_i and $\boldsymbol{\eta}_H = [\eta_1, \eta_2, \ldots, \eta_N]^{\mathrm{T}}$. Note that triquadratic reconstruction is used both in the functional correction and remaining error, Eqs. 35 and 36, which is a slight departure from Eq. 32. This is based on the assumption that the triquadratic interpolant is our best estimate of the embedded-mesh adjoint. Hence it is used not only to compute the remaining error but also to get an improved functional estimate.

Substituting Eq. 35 into Eq. 11 gives a computable estimate of discretization error on the working mesh

$$\mathcal{E} \approx C|J_c - J_H(\mathbf{Q}_H)| \tag{37}$$

To define a local quantity suitable for driving adaptive mesh refinement, the remaining error, Eq. 36, is localized to form an error indicator $|\boldsymbol{\eta}|_H = [|\eta_1|, |\eta_2|, \ldots, |\eta_N|]^{\mathrm{T}}$. The sum of the error-indicator values over the cells of the working mesh gives a bound on the estimate of the remaining error

$$\eta = \sum_{i=1}^{N} |\eta_i| \tag{38}$$

Since the absolute value operator prevents cell-wise error cancellation, this bound is quite conservative. In fact, when dealing with difficult simulations containing non-smooth flow and arbitrary geometry, η is typically more conservative than the value given by Eq. 37.[e] We return to this topic in the examples of Sec. 7.

An alternate approach involves the use of the adjoint correction term, $|(\psi_h^H)^{\mathrm{T}}\, \mathbf{R}_h(\mathbf{Q}_h^H)|$, as an error indicator [64, 18]. However, the remaining error term converges at about double the rate of the correction and is more conservative at sonic and stagnation lines, where adjoint variables vanish but their derivatives do not. Moreover, since the flow and adjoint equations are solved on the same mesh, there is an open question regarding the error indicator maintaining consistency of the adjoint solution as the mesh is refined. Although $|\boldsymbol{\eta}|_H$ is sensitive to non-linearities in the adjoint solution, other implementations [15, 5] use a complementary remaining-error term that involves an inner product of the flow solution with the adjoint residual. This term makes the mesh more suitable for the adjoint solution, but our numerical experiments did not show significant benefits.

There are several extensions of the present approach for handling multiple outputs. In principle, each output requires its own adjoint, which significantly increases simulation cost. One way to reduce the cost is to form a discrete error equation that is solved in conjunction with a modified adjoint system [16]. In this work, a simpler approach is

[e] Since η estimates the bound on the remaining error with respect to the embedded mesh, $C\eta$ can be used to extrapolate toward the exact value, similar to Eq. 37.

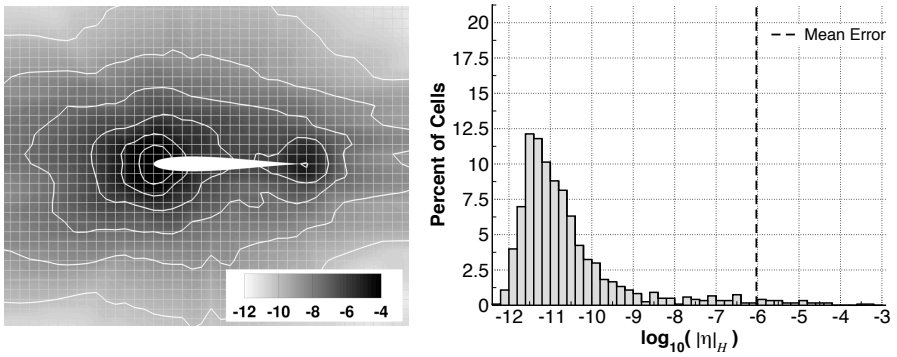


Figure 5. Near-body cell-wise error indicator for C_d ($\log_{10} |\boldsymbol{\eta}|_H$) of airfoil in subsonic flow (left) and corresponding error histogram (right)

used, where the solution of only one adjoint system is required. Multiple outputs are combined using a weighted-sum formula

$$J = \sum_{i=1}^{K} w_i J_i \tag{39}$$

where K is the number of outputs and w is an array of user-specified constants. In practice, outputs are frequently projections of wall pressure, for example aerodynamic forces and moments, for which the weighted-sum of axial, normal and side forces works well.

5 Mesh Adaptation

5.1 Equidistribution of Remaining Error

While error estimates are critical for assessing solution quality, automatic error control requires additional procedures that identify regions of high error and modify the mesh to drive the error below desired tolerances. Unlike traditional error indicators, such as feature detection and estimates of local truncation error, $|\boldsymbol{\eta}|_H$ is a direct (point-wise) estimate so there is no ambiguity regarding the selection of an indicator variable, its relation to the functional error and its convergence rate.

To introduce the concept of error equidistribution, consider an example error map shown in Figure 5. The values of $|\boldsymbol{\eta}|_H$ for C_d in the near-body region of a Joukowski airfoil in subsonic flow ($M_\infty = 0.4$ and $\alpha = 1°$) are shown in the left frame. A logarithmic scale is used to emphasize the rapid variation of the error indicator near the airfoil, with highest errors at the leading and trailing edges. The corresponding error histogram for the entire domain is shown on the right. The horizontal axis contains bins of the error-indicator values. High error cells lie to the right. The vertical axis is the percentage of cells in each bin.

The histogram provides insight into how well the mesh fits the simulation. In this case, the histogram is skewed, with most of the cells

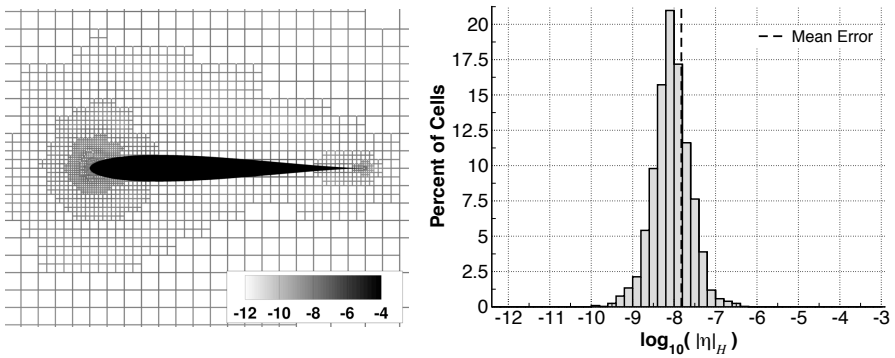


Figure 6. Near-body adapted mesh colored by $\log_{10} |\boldsymbol{\eta}|_H$ in C_d for airfoil in subsonic flow (left) and corresponding error histogram (right)

contributing little error. This indicates that the mesh is inefficient, which is expected because the near-body mesh is uniform. The high-error cells are close to the airfoil and, in particular, near regions of high curvature. The highest errors (above 10^{-5}) dominate η. This is reflected by the location of the mean-error value, η/N, shown as a dashed line in Fig. 5. The majority of cells have error of several orders below the mean, thus the mean is well to the right of the peak (the mode of the histogram). Hence, most of the computational work associated with this mesh is unnecessary.

The basic strategy for controlling cell-size to minimize error is to equidistribute the error as the adaptation advances. The principle of error equidistribution has been demonstrated in [1, 3], among others. The goal is for each cell to contribute equally toward improving the accuracy of the simulation. Intuitively, any departure from a uniform error distribution implies that the mesh points could be redistributed to obtain a lower average error. Historically, error equidistribution has been sought in global error estimates, such as truncation errors or energy norms. These error estimates, however, tend to significantly increase computational cost because they may trigger refinement of all flow features everywhere in the domain. In contrast, adjoint error estimates seek equidistribution of the functional error and consequently focus only on regions (cells) important for predicting the functional. In other words, the adaptation seeks to refine cells that make $|\boldsymbol{\eta}|_H$ uniform and η small.

Figure 6 illustrates the main idea, where an adapted mesh is generated for the airfoil example of Fig. 5. The adapted mesh is shown on the left and its histogram on the right. The histogram is nearly symmetric with little spread. The value of the error indicator varies less than an order of magnitude in over 80% of the cells. Comparing to Fig. 5, the adapted mesh reduces the error variation by seven orders of magnitude. All high error cells have migrated to the left and are clustered close to the mode of the histogram. Moreover, the mean is also close to the mode, suggesting that little computational work is wasted on low-error cells. The grading of the mesh is achieved through a series of adaptations and is a direct

reflection of the error map in each cycle, such as the one shown in Fig. 5. In the limit, the ideal histogram is a delta function. In practice, the discrete subdivision of Cartesian cells and cell-wise non-uniform error convergence, as well as overall computational cost, limit the tightness of the histogram.

5.2 Error Control

Efficiency of the adaptation procedure is driven by the amount of computational work needed to transform the error histogram of the initial mesh to a delta function and position it sufficiently far left in the region of low error. This can be controlled by carefully selecting refinement and coarsening thresholds above and below which cells are marked for refinement and coarsening, respectively. Choosing refinement thresholds where only the highest-error cells are refined yields tight histograms but requires many adaptation cycles to shift the mode into a region of low error. Therefore, the procedure should also minimize the number of cycles and, in particular, avoid solving on similar meshes when close to the final mesh.

For simulation verification, perhaps the most common approach is to prescribe a tolerance `TOL` on the remaining error or directly on $\mathcal{E}$. For example, the goal may be to construct a mesh with $\eta < 0.0001$ in C_D. This results in thresholds proportional to $\texttt{TOL}/N$ (or $\texttt{TOL}/N_{max}$) that drive the mesh toward error equidistribution. In earlier work [35, 36], we evaluated this approach and modified it to accommodate a"worst-errors-first" strategy. This reduces computational cost by avoiding the problem of generating too many cells early in the adaptive process, before the error map is accurate, and then paying for these cells on every intermediate mesh until the highest-error cells are finally addressed in the closing cycles. The threshold is set to $\lambda \cdot \texttt{TOL}/N$, where $\lambda \geq 1$ is a user-specified array of constants that typically decrease as the adaptation advances.

Use of the tolerance-driven approach in practice reveals several problems. Specifying a meaningful `TOL` is awkward for functionals such as line sensors, where there is little intuition guiding reasonable choices of desirable error level. In addition, λ is problem dependent and the resulting mesh growth is hard to control in difficult simulations, where occasional poor convergence of the flow or adjoint solver may occur and cause spurious error estimates in some region of the domain. An alternate approach is to minimize the error for a given cell budget, e.g., the goal is to construct a mesh with one million cells that predicts C_D most accurately. Cells are sorted on their level of error and a threshold is determined such that a fraction of the highest error cells is refined. The threshold in each cycle can be determined from statistics of the error distribution [65, 66, 37], such as the mean and standard deviation, or set to meet a user-specified mesh growth.

5.3 Implementation

The main steps of the simulation are given by Procedure 1.

Procedure 1: Adaptive Mesh Refinement

Input: Surface triangulation T and refinement parameter array τ
Termination Criteria: `TOL`, or maximum number of cells N_{max}, or number of cycles M, or maximum level of refinement R_{max}

$H_0 \longleftarrow InitialMesh(T)$ `// Generate initial mesh`
for $i \leftarrow 1$ **to** M **do**
  $\mathbf{Q}_H \longleftarrow FlowSolve(H_{i-1})$
  $\psi_H \longleftarrow AdjointSolve(H_{i-1}, \mathbf{Q}_H)$
  $h \longleftarrow EmbedMesh(H_{i-1})$ `// Uniform refinement`
  $\boldsymbol{\eta}_H \longleftarrow CellwiseErrorEstimate(h, H_{i-1}, \mathbf{Q}_H, \psi_H)$
  $\eta \longleftarrow \sum_{H_{i-1}} |\eta|_H$
  $H_i \longleftarrow AdaptMesh(\tau_i, |\eta|_H, H_{i-1}, \texttt{TOL})$
  break if $(\eta < \texttt{TOL} \;||\; N > N_{max} || \; R > R_{max})$
end
$\mathbf{Q}_H \longleftarrow FlowSolve(H_M)$ `// Optional solve`
Result: $\{J_H, J_c, \eta\}_i, \;\; i = 0, \ldots, M$

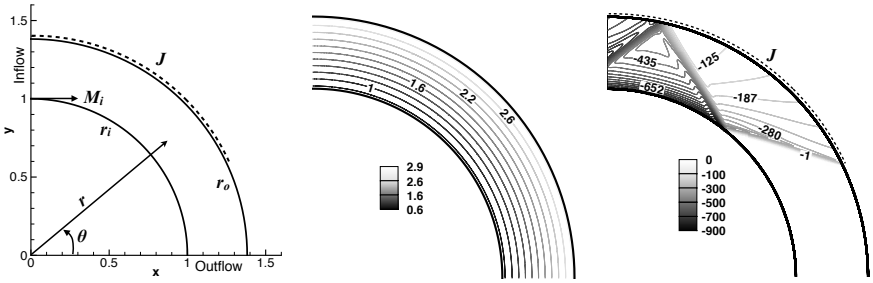
While the steps are standard, we note several places where we specialized the procedure:

1. The surface triangulation T is held fixed. Consequently, it is important that the surface triangulation is sufficiently fine to support the final volume mesh, particularly in regions of high surface curvature.

2. The exit criteria of the adaptation loop are positioned such that only the flow solution is computed on the final mesh. This is optional but effective in practice, because the primary outputs of the simulation come from the flow solver and it is sensible to use the already computed error map. In terms of simulation verification, the procedure is conservative because we expect η to decrease on the final mesh.

3. Except for the final adaptation, only one level of refinement is added per adaptation cycle. We allow more levels on the final cycle because the final error map should be close to asymptotic. Coarsening is not considered, which improves robustness without significantly impacting the efficiency of steady simulations.

4. Once the cells are marked for refinement, the tags are processed to enhance mesh smoothness. This includes filtering out refinement islands and voids, buffering tags (usually by one layer of cells) and enforcing $2:1$ cell ratios.



(a) Setup (b) Pressure (c) Density adjoint

Figure 7. Supersonic-vortex model problem ($M_i = 2.25$, $r_i = 1$, and $r_o = 1.382$)

The main results of Procedure 1 are the functional value, its correction and remaining error for the sequence of meshes generated during the simulation. Convergence analysis of these quantities is used to verify the simulation.

6 Code Verification

An analytic supersonic vortex provides a model problem with a known solution [67] to verify the performance of the error estimation framework. We also verify the accuracy of the adaptive mesh refinement procedure to establish a benchmark for automatic simulation verification. The problem involves isentropic flow between concentric circular arcs at supersonic conditions, as shown in Fig. 7(a). The exact solution is given by

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

$$u = a_i M_i \left(\frac{r_i}{r} \right) \sin\theta, \quad v = -a_i M_i \left(\frac{r_i}{r} \right) \cos\theta, \quad p = \frac{p_i}{\rho_i^\gamma} \rho^\gamma \tag{40}$$

where $a_i = \rho_i = 1$, $p_i = 1/\gamma$, $M_i = 2.25$, $r_i = 1$, and $r_o = 1.382$. The output of interest is the integral of pressure over a portion of the outer arc

$$J = \int_0^{\frac{5\pi r_o}{14}} p_{r_o} \, \mathrm{d}l = \frac{5\pi r_o}{14} p_{r_o} \approx 4.39262683 \tag{41}$$

as sketched in Fig. 7(a). This choice permits validation of the quadratic adjoint interpolant described in Sec. 2, which models the adjoint solution on the embedded mesh. Figure 7 shows that while the flow is smooth, the adjoint is not.

All simulations are initialized with the analytic solution and solved to steady state without limiter. Dirichlet boundary conditions at the inlet are prescribed from the analytic solution; the wall boundary conditions are specified via Eq. 19.

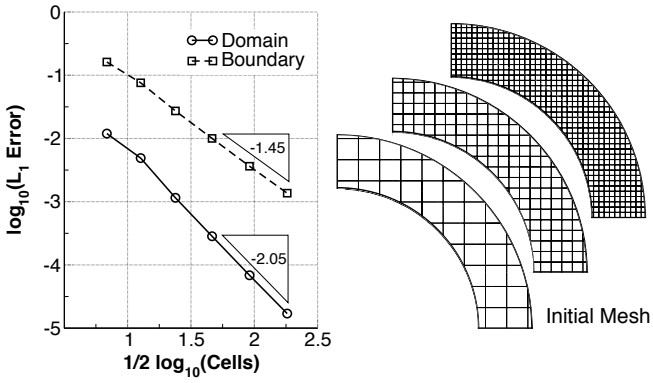


Figure 8. Error in density versus number of cells (left), and initial mesh and meshes after first two uniform refinements (right)

6.1 Uniform Mesh Refinement

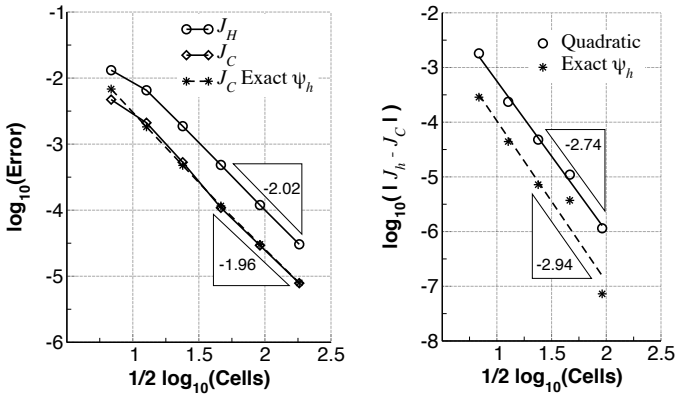
A uniform mesh refinement study is performed that involves a sequence of six nested meshes. Figure 8 shows the initial mesh, which contains 47 cells. The final mesh contains 33,069 cells. Note that the boundary discretization changes non-smoothly with refinement, e.g., on each mesh the cut-cell area fractions varied by at least five orders of magnitude. Consequently, convergence rates are obtained via linear regression of the finest four solutions.

Figure 8 shows that the solution error (point-wise L_1 norm of $\rho - \rho_H$) is $\mathcal{O}(h^2)$ when measured over the entire domain. This is consistent with the second-order accurate discretization and smooth analytic solution. The second line in Fig. 8 shows that the convergence rate is reduced to slightly below $\mathcal{O}(h^{3/2})$ when confined to include only the boundary due to the irregular discretization stencils in the cut cells.

Despite the slower convergence of the solution along the boundary, the first line in Fig. 9(a) shows that error in J_H is $\mathcal{O}(h^2)$. The second line shows the convergence rate of error in the corrected functional J_c. As expected, its convergence parallels J_H, but the error improves by about a half-order of magnitude. The third line in Fig. 9(a), labeled "J_c Exact ψ_h", is obtained by solving the adjoint equation on the embedded mesh and using these values in Eq. 35 when computing the corrected functional. This is referred to as an exact correction, which we use to validate the triquadratic interpolant[f]. Except on the initial mesh, the quadratic correction performs as well as the exact correction.

Figure 9(b) examines the accuracy of the corrected functional J_c in detail. The correction should capture the dominant part of the functional's value on the embedded mesh J_h relative to the working mesh value J_H, as sketched in Fig. 1 and expressed by Eq. 4. We examine

[f]Recall that the cost of an embedded-mesh adjoint solution is prohibitive in practice. It is roughly equal to that of a flow solution on the embedded mesh.



(a) Error (b) Relative error

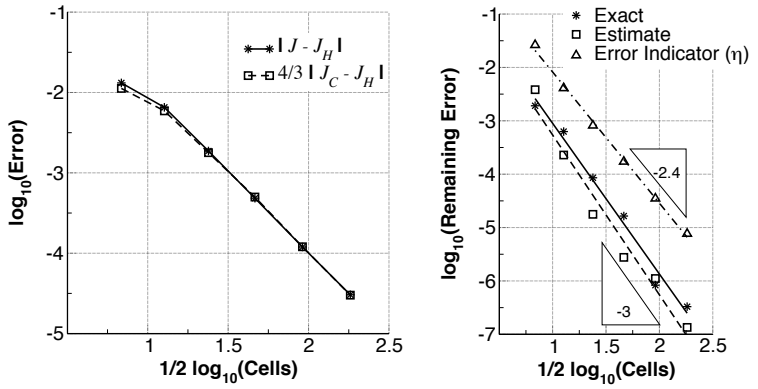
Figure 9. Convergence of functional (J_H) and its correction (J_c) for the uniform mesh refinement study of the supersonic vortex. Error is measured with respect to the exact value (left) and the value on the embedded mesh (right).

the accuracy of the corrected functional using both the quadratic adjoint interpolant and the embedded-mesh adjoint, labeled as "Exact ψ_h" in Fig. 9(b). Both corrections are superconvergent; the exact correction attains almost $\mathcal{O}(h^3)$ when measured in this relative-error metric. The quadratic mimics the exact correction at a slightly slower rate and with about a half-order offset starting from the coarsest mesh. These results show that the adjoint field on the embedded mesh is accurately predicted with the limited tri-quadratic interpolant (described in Sec. 4).

The ability to compute relative error accurately suggests that the true error should also be predicted reliably as H tends to zero. Since the convergence rate of the functional is second-order, Eq. 37 with $C = \frac{4}{3}$ estimates the true error. The first line in Fig. 10(a) shows convergence of the true error $|J - J_H|$, which is copied from Fig. 9(a) for reference. The second line (squares) is the error estimate $\mathcal{E}$ using Eq. 37. The estimate is sharp; the ratio of the estimate to the true error (effectivity) is close to one on all but the coarsest meshes, i.e., the lines essentially over-plot.

The vanishing gap between the estimated and true error is quantified by the first line in Fig. 10(b), labeled "Exact". We use this error gap to test the accuracy of the remaining error term of Eq. 32. This is done by evaluating Eqs. 36 and 38 without localization, i.e. by omitting the absolute value operator in Eq. 38. The second line in Fig. 10(b), labeled "Estimate", shows the value of $4/3\eta$ without localization[g]. The estimate is within half-order of magnitude of the measured gap and has a similar convergence rate. The agreement is excellent considering that the quadratic interpolant is used in both the correction and remaining error

[g]The factor of 4/3 is applied to extrapolate the remaining error to its analytic value based on the observed second-order functional convergence.



(a) Error estimate (b) Remaining error

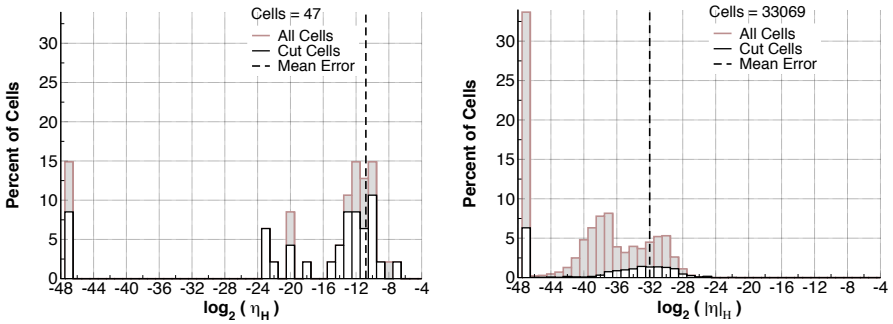
Figure 10. Accuracy of the error estimate using Eq. 11 (left). Remaining error and bound estimate (right) for the uniform mesh refinement study of the supersonic-vortex problem.

terms, and only linear prolongation of the flow solution is used when evaluating the embedded-mesh residuals. The third line in Fig. 10(b), labeled "Error Indicator", shows the effect of localization when forming the error-indicator bound η using Eq. 38. Recall that the absolute value operator prevents cell-wise error cancellation. As a result, the bound is roughly one-order of magnitude larger than the remaining error on the coarsest mesh and its convergence rate is slower, indicating that the bound is a conservative estimate of the remaining error.

Having established the accuracy of the error indicator, Fig. 11 shows histograms of the error indicator $|\boldsymbol{\eta}|_H$ on the initial and final meshes. The histograms are similar to the ones presented in Figs. 5 and 6 except we use a $\log_2$ scaling of the cell-wise error. This scaling is intuitive for predicting how far a bin moves to the left once its cells are refined because the subdivision of Cartesian cells is discrete. For example, we show in Fig. 10(b) that the convergence rate of the error-indicator bound η is roughly $\mathcal{O}(h^2)$, which implies that the mean error should shift four units to the left after each refinement[h]. Figure 11 shows that the mean error is close to 2^{-12} on the initial mesh and shifts to 2^{-32} after five refinements.

Figure 11(b) shows that the uniformly refined mesh is not particularly well suited for the simulation at hand. The error distribution is far from a delta function and more than a third of the 33,069 cells contribute no error. The adjoint field shown in Fig. 7(c) provides some insight. The large variations reveal the influence of point-source mass perturbations, which include interactions with the inner arc, on J (Eq. 41). Recall that J is defined over about 2/3 of the outer arc, at which point the adjoint variable vanishes because any perturbation downstream of this location cannot influence J in this hyperbolic problem. Hence, cell refinement

[h]The mean shifts $p + d$ units to the left, where p is the order of the error indicator and d is the spatial dimension.



(a) Initial mesh

(b) Final mesh (cut-cell percentage magnified 6× for clarity)

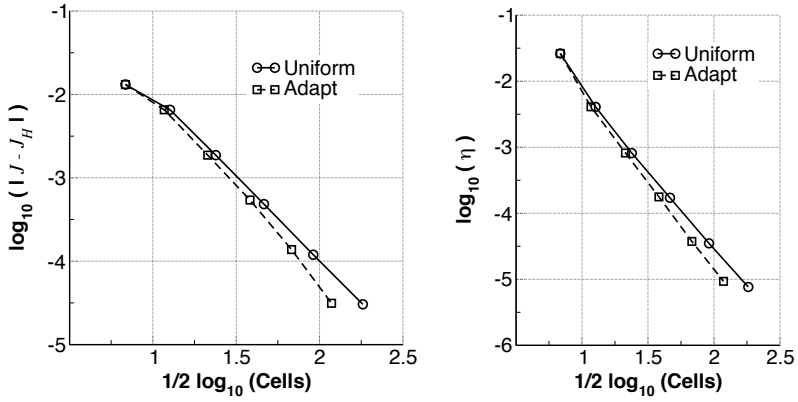
Figure 11. Histograms of the cell-wise error indicator $|\boldsymbol{\eta}|_H$ for the uniform mesh refinement study of the supersonic vortex. Heights are normalized by the total number of cells. Error indicator values in cut cells are shown as rectangles with white fill.

outside the functional's zone of dependence yields many cells with zero-error contributions. These cells increase cost without improving accuracy of the simulation.

6.2 Adaptive Mesh Refinement

Accuracy of the adaptive procedure is demonstrated by performing five adaptive refinements starting from the initial mesh of the uniform study. The refinement threshold is set to the mean error value, except in the first two adaptations where it is shifted two $\log_2$ bins to the left of the mean due to the coarse initial mesh. Figure 12 shows the error convergence of the functional and η, and compares them with those of the uniform meshes. We observe that after each refinement the functional error and η nearly match the values obtained from the uniform meshes but use fewer cells. The final mesh contains 13,929 cells compared to the 33,069 cells of the finest uniformly refined mesh.

Figure 13 shows the final mesh and its error histogram. The refinement pattern is clearly driven by features of the adjoint solution, recall Fig. 7(c). Essentially no mesh refinement occurs past the functional's zone of dependence. An error indicator based on local-truncation errors would unnecessarily adapt cells in this region. The largest sensitivity to residual errors occurs along the inner arc, which may seem counter-intuitive. This is due to the nonuniform inlet Mach number and the number of local Mach wave reflections that can reach the functional from the inner arc. More importantly, the final mesh appears to strike a good compromise for accommodating both the flow and adjoint solutions. The inner product within the error indicator $|\boldsymbol{\eta}|_H$ is dominated by the adjoint interpolation error $(\psi_{\mathrm{TQ}} - \psi_{\mathrm{TL}})$, which emphasizes regions of high adjoint curvature that are captured in the final refinement pattern.



(a) Functional error (b) Error-indicator bound

Figure 12. Convergence of the functional error (left) and error indicator η (right) on adapted meshes of the supersonic vortex problem.

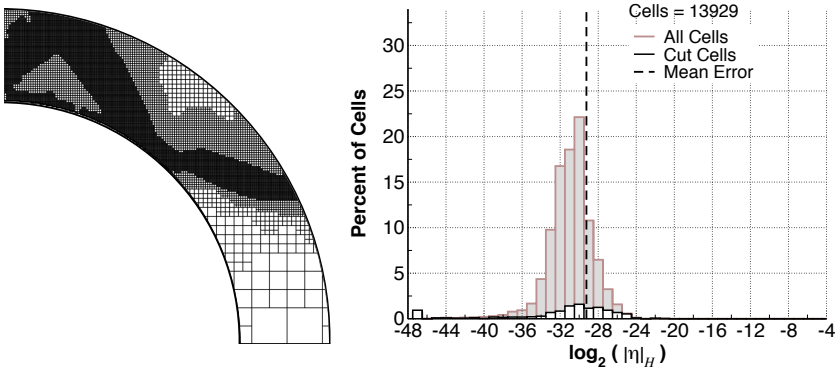


Figure 13. The final adapted mesh (left) and the corresponding error-indicator histogram (right) for the adaptive refinement of the supersonic vortex problem. The cut-cell percentage is inflated 3× for clarity.

The histogram in Fig. 13 shows that the final adapted mesh fits the simulation well and contrasts markedly from that in Fig. 11(b). The histogram is symmetric with little spread, the mean error is close to the mode and the adapted mesh has almost no cells with zero error. Moreover, the histograms of Figs. 11 and 13 show not only the error indicator values in every cell, but also isolate the errors in the cut cells. Recall that Fig. 8 shows slower convergence rate of the solution in the cut cells. To compensate, the error indicators in cut cells are inflated by a factor of 1.5 before selecting cells for refinement. This value is based on experience with many problems. Figure 13 shows that mode of the error distribution in cut cells roughly corresponds to the mode for all cells, indicating that the adaptation mechanics are not adversely affected by the non-uniform error convergence.

7 Examples of Simulation Verification

Three examples are presented to demonstrate the effectiveness of Procedure 1 in simulation verification. The examples progressively increase in complexity, from airfoil simulations to three-dimensional simulations of a launch abort vehicle. To varying degrees, the examples deliberately violate the assumptions made in the derivation of the error estimates, in particular assumptions of smoothness, steady-state and fineness of the initial mesh. These assumptions rarely apply in practical engineering simulations and our goal is to characterize the performance of Procedure 1 in such situations.

7.1 Airfoil Performance Database

The first example demonstrates verification of a model aerodynamic performance database. The goal is to predict the drag coefficient, $J = C_d$, of the familiar NACA 0012 airfoil over a range of freestream conditions. We consider a total of 60 cases involving six subsonic Mach numbers, $M_\infty = \{0.1, 0.2, 0.3, 0.5, 0.7, 0.9\}$ and four supersonic Mach numbers, $M_\infty = \{1.1, 1.3, 2, 3\}$, at six angles of attack, $\alpha = \{0, 0.5, 1, 2, 4, 8^\circ\}$.

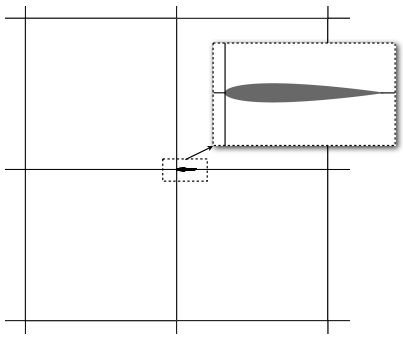


Figure 14. Near-body view of initial mesh with inset showing NACA 0012 airfoil

Figure 14 shows the near-body region of the initial mesh, including a closeup of the NACA 0012 airfoil[i]

[i]The trailing edge of the airfoil is modified to be sharp, which is accomplished by modifying the last coefficient of the equation defining its thickness distribution [68].

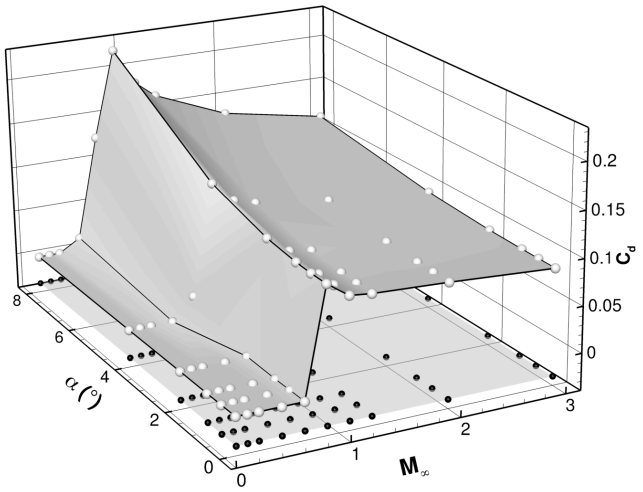
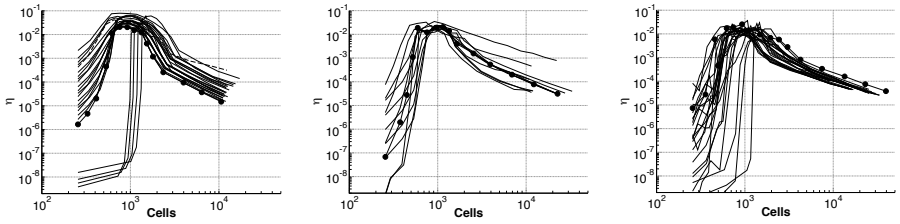


Figure 15. C_d for all freestream conditions of the airfoil performance database example

with a unit chord. The distance to the outer boundary is 64 chords. The initial mesh contains 16×16 cells with characteristic length $H = 8$ chords. The airfoil is intersected by just four cells. The spatial discretization does not involve a limiter when $M_\infty \leq 0.5$; cases with $M_\infty > 0.5$ use the Van Leer limiter.

To construct the database, we set the maximum level of refinement, R_{max}, to 18 and set the adaptation threshold to the mean error. This is a compromise of the various database strategies we examined in [39], which contrasted uniform-error and fixed-mesh databases. In general, specifying a uniform error tolerance, e.g., $\mathcal{E} < 0.0001$ in C_d, for all cases is not cost-effective. This is because the cost of a constant-error database is frequently dominated by a few corner cases, for which a less accurate computation would suffice. Alternatively, specifying the number of adaptation cycles (M in Procedure 1) and the mesh growth for each cycle fixes the cost of the database. This strategy, however, restricts the degree of control over the variation of error across the database. The current approach achieves a balance by requiring that the final meshes for all cases contain the same smallest cell-size ($H \approx 0.0002$ chords), but allows the total number of cells to vary.

Figure 15 shows the final value of C_d for all simulations in the database. Although there are no analytic solutions for the NACA 0012 airfoil in a finite computational domain, we expect C_d to approach zero in a shock-free two-dimensional inviscid flow. Reassuringly, Fig. 15 shows that C_d is essentially zero when $M_\infty \leq 0.5$. Moreover, while the variation in C_d with angle of attack is generally mild at fixed M_∞, there is a rapid rise in C_d with respect to M_∞ in the transonic regime. The largest values of C_d are observed when $M_\infty = 0.9$, which is due to a strong expansion over the aft portion of the airfoil caused by shocks at the trailing edge. As



(a) Subsonic cases (b) Transonic cases (c) Supersonic cases

Figure 16. Convergence of error indicator η for subsonic $M_\infty = \{0.1,\, 0.2, 0.3,\, 0.5\}$, transonic $M_\infty = \{0.7,\, 0.9\}$ and supersonic $M_\infty = \{1.1,\, 1.3, 2,\, 3\}$ cases of the database example

the Mach number increases into the supersonic regime, there is a gradual decrease in C_d as the bow shock becomes more oblique.

While Fig. 15 demonstrates that the simulation data follows expected trends, it offers little quantitative evidence for simulation verification. Since this example involves 60 simulations, we focus the current discussion only on the error estimate, Eq. 37, and the error indicator, Eq. 38, which are central to verification and adaptation. The single-point examples in Subsections 7.2 and 7.3 study these quantities in greater detail.

Error-Indicator Convergence

Figure 16 shows convergence of the error-indicator bound η (Eq. 38). The cases are divided according to the freestream Mach number into groups of subsonic, transonic and supersonic flow. Overall, η converges well. This implies mesh convergence for all cases and demonstrates the ability of the adaptation to control the magnitude of the remaining error in Eq. 32. The circle symbols, plotted for an arbitrary case in each plot, mark adaptation cycles. The error indicator initially increases until the meshes reach around 1000 cells. This is typical when starting from such a coarse mesh (recall Fig. 14). Increasing η indicates that new features of the solution are emerging, which are not yet captured by the error analysis. Nevertheless, the early error maps are sufficiently accurate to identify critical regions of the evolving flowfield. When combined with the incremental h-refinement strategy of Procedure 1, the error maps reliably handle coarse initial meshes. As the simulations approach R_{max}, in particular over the last four meshes of each case, η is decreasing linearly indicating that the output is asymptotic.

Figure 16(a) shows that η is reduced by three orders of magnitude for all but two of the 24 subsonic cases. The convergence pattern is very similar among the cases, especially once the meshes reach 1000 cells. The four cases with the sharp error rise at 1000 cells are all at $\alpha = 0°$. The case with the slowest convergence is when $M_\infty = 0.5$ and $\alpha = 8°$, indicated by the dashed line, which corresponds to the onset of transonic flow. To reach the desired R_{max} of 18, the meshes for all subsonic cases

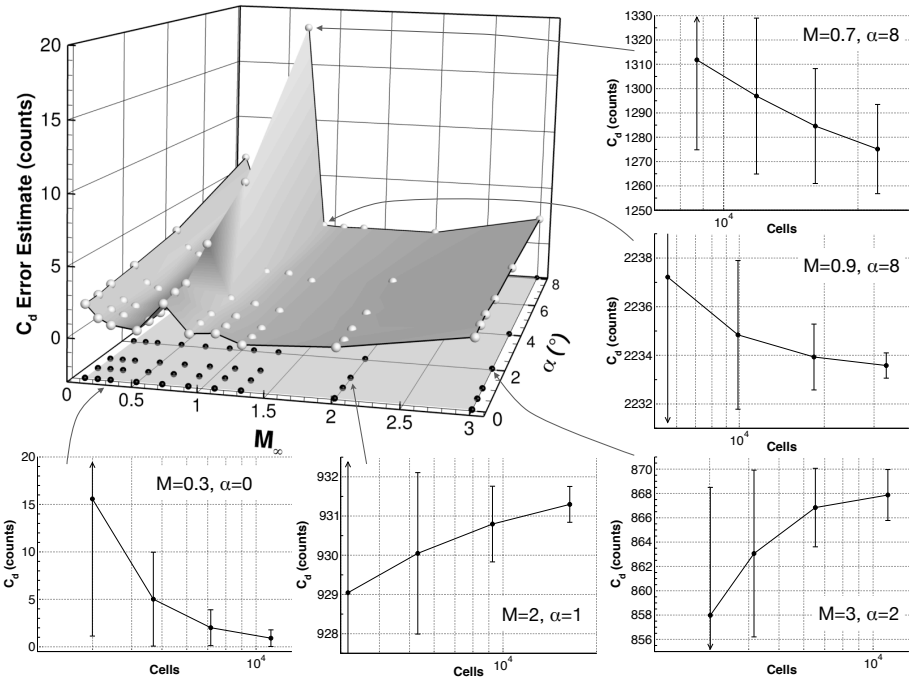


Figure 17. Final error estimate in C_d for all freestream conditions (upper left) and samples of C_d convergence on the last four meshes

require 14 adaptations[j].

Figure 16(b) shows convergence of η for the 12 transonic cases. The meshes, on average, are larger than those for the subsonic cases. The cases with the smallest values of η and also the smallest number of cells correspond to the shock-free cases, i.e., $M_\infty = 0.7$ and $\alpha \leq 1°$. The slowest convergence (largest η) occurs when $M_\infty = 0.7$ and $\alpha = 8°$; the next slowest is $M_\infty = 0.7$ and $\alpha = 4°$. These two cases are actually the slowest to converge in the entire database. We obtained similar results in [39], where the largest sensitivity of aerodynamic performance to discretization error was also near $M_\infty = 0.7$. The primary culprit is the dependence of C_d on the location of the upper surface shock. Once M_∞ reaches 0.9, the shocks migrate to the trailing edge. Figure 16(c) shows convergence of the 24 supersonic cases. These cases converge well and are tightly clustered. These meshes require 15 adaptations to reach R_{max} of 18, which is one additional cycle when compared to the subsonic and transonic cases.

Level of Discretization Error

Having established convergence of the error indicator, we now examine the level of discretization error across the database. Figure 17 shows the value of the final error estimate $\mathcal{E}$ in the carpet plot at upper-left. The

[j]The initial mesh contains four levels of refinement.

error estimates are computed with $C = \frac{4}{3}$ in Eq. 37 for shock-free cases that do not require a limiter ($M_\infty \leq 0.5$), otherwise $C = 2$ is used. The carpet plot shows that $\mathcal{E}$ is small and essentially uniform ($\mathcal{E} \leq 2$ counts) over most of the subsonic and supersonic cases. Not surprisingly, the largest error estimates are obtained in the transonic regime, specifically when $M_\infty = 0.7$, which corroborates with the slow convergence of η in Fig. 16(b). There is also an increase in $\mathcal{E}$ as $M_\infty \to 0$ that indicates some loss of accuracy in the incompressible regime due to the lack of low Mach-number preconditioning. This is consistent with results presented in [39].

Along the perimeter of the carpet plot in Fig. 17, we show convergence of C_d for the last three adaptations of representative cases. The error bars denote the value of $\mathcal{E}$. In general, the insets show that C_d changes less than one count over the final adaptation and $\mathcal{E}$ is decreasing by about a factor of two per cycle and brackets the final solution over at least the last two cycles. For example, the lower-left inset shows the classic subsonic case $M_\infty = 0.3$ and $\alpha = 0°$, where C_d correctly approaches zero as $H \to 0$. The error estimate brackets the expected final solution ($C_d = 0$ to plotting accuracy) on the last three meshes. The case at $M_\infty = 0.7$ and $\alpha = 8°$, shown at top-right, is the main exception. The value of C_d is still changing significantly and the error estimate is just starting to tighten. More adaptation cycles (smaller cells) would be required to reduce the error estimate further. Taken together, Figs. 16 and 17 provide the quantitative evidence required for verification of every case in the database.

Meshing Requirements

Figure 18 shows the number of cells required to achieve the error levels presented in Fig. 17, as well as snapshots of the final near-body meshes for selected cases. The carpet plot shows that the number of cells varies by about a factor of four over the database and peaks when $0.9 \leq M_\infty \leq 1.3$. To give an indication of the wall-clock time needed to construct this database, each case requires, on average, about four minutes on one core of a laptop computer[k].

The largest mesh is obtained when $M_\infty = 1.1$ and $\alpha = 8°$ and contains 39,500 cells. The top-left inset shows that the high cell count is primarily due to a detached bow shock far upstream from the airfoil, which is indicated by the cluster of small cells along the inset's left edge. In contrast, Fig. 18 shows that cases with $M_\infty \geq 2$ require many fewer cells. Recall that the refinement pattern is driven by the inner product of the adjoint interpolation error $(\psi_{\mathrm{TQ}} - \psi_{\mathrm{TL}})$ with the flow residual error, Eq. 36. Upstream of the bow shock, the flow residuals are zero because the spatial discretization preserves uniform freestream. Concurrently,

[k]MacBook Air (2013) with a 1.7 GHz Intel Core i7 processor and 8GB of memory.

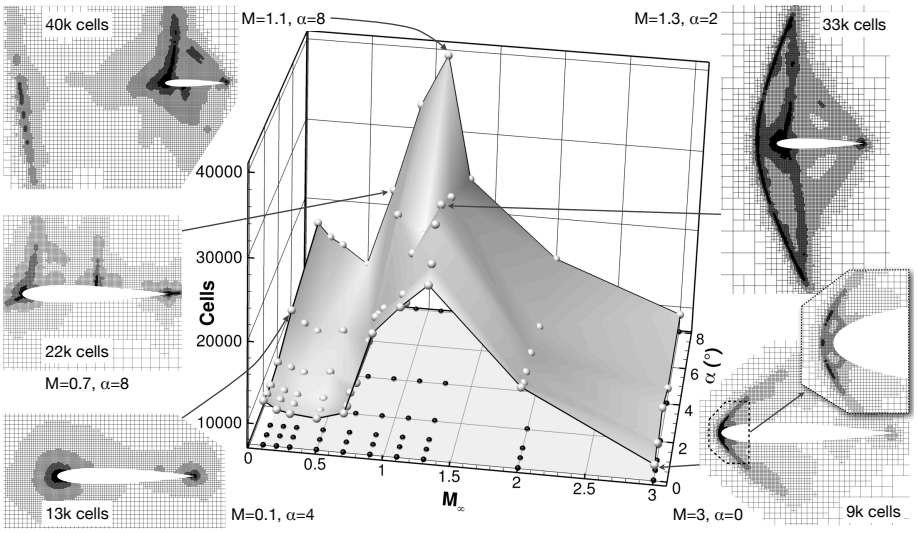


Figure 18. Number of cells for all freestream conditions with samples of final adapted meshes (near-body views)

the adjoint solution vanishes downstream of the limiting characteristic intersecting the airfoil's trailing edge. This is similar to Fig. 7(c) of the supersonic vortex problem, where the adjoint solution is zero when outside the functional's zone of dependence. Consequently, $|\eta|_H$ vanishes upstream of the bow shock and outside the limiting characteristic, as indicated in the insets on the right side of Fig. 18 for cases at $M_\infty = 1.3$ and $M_\infty = 3$. This confines the refinement pattern to a "diamond" that contracts in the cross-flow direction with increasing Mach number. As a result, the smallest mesh, containing only 8860 cells, is obtained at the highest Mach number, specifically at $M_\infty = 3$ and $\alpha = 4°$.

As $M_\infty \to 0$, the carpet plot of Fig. 18 shows a moderate increase in the number of cells that is consistent with the increase in $\mathcal{E}$ shown in Fig. 17. The inset at lower-left shows a typical mesh in the subsonic regime. At a given angle of attack, the meshes are self-similar in the sense that the refinement regions simply enlarge with decreasing Mach number to compensate for the weakening pressure variations.

Overall, Figs. 17 and 18 offer a convincing demonstration of the benefits of adaptive mesh refinement and automatic error control for simulation verification. Even in this simple 2D example, constructing a single mesh that accommodates all the flow regimes, from smooth subsonic flow to the disparate shock systems of transonic and supersonic flow, is a daunting task that requires orders of magnitude more cells. Moreover, uniform refinement of such a mesh to demonstrate verification is not practical.

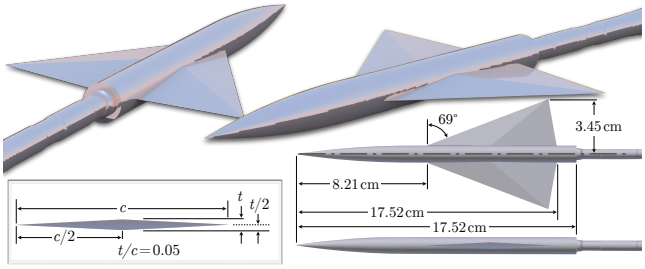


Figure 19. Delta-wing-body model with a partial view of the wind-tunnel sting

7.2 Pressure Signature of a Body in Supersonic Flow

This example demonstrates verification of a simulation that predicts the near-field pressure signature of a lifting configuration in supersonic flow. Such simulations are used to determine loudness of sonic booms [69, 70]. The key to credible noise analysis is the accurate prediction of the weak shocks and expansions, and their interactions, that are generated by low sonic-boom vehicles. These pressure disturbances are highly susceptible to attenuation by discretization error and consequently, these simulations hinge on the mesh [71, 38, 23, 72, 43, 73].

The simulation involves a popular wing-body model, originally identified as "Model 4" in the experiments of Hunton *et al.* [74] and used subsequently in many studies, for example [70, 38, 72, 75, 76]. Figure 19 shows the details of the geometry, including an axisymmetric fuselage, a delta wing with diamond-airfoil cross sections and an approximate wind-tunnel sting from [38] that forms a step junction at the base and extends downstream. The surface tessellation contains 1.3 million triangles.

Our goal is to predict the pressure signature on a line located 3.6 model lengths below the model's centerline and parallel to freestream. The output functional is given by Eq. 21, where we set $n = 2$ to emphasize the importance of accurately capturing the peaks of the signature. The freestream Mach number is 1.68 and the angle of attack is set to match a C_L of 0.15. This corresponds to one of the experiments performed in [74]. Monotonicity of the solution is maintained through use of the Barth-Jespersen limiter [63].

The left frame of Fig. 20 shows the setup for a half-body simulation. The model and the sensor (solid horizontal line) are shown on the symmetry plane of the near-field region of the initial mesh. The mesh is rotated to approximately align the cells with the freestream Mach-wave angle and the cells are stretched, about $2:1:2$, in the direction of the wave angle and spanwise to improve computational efficiency. The initial mesh contains $12 \times 12 \times 8$ cells, and the geometry and sensor are intersected by just five cells. In other words, except for the rotation and cell stretching, the mesh is not biased to anticipate features of the solution and should be far from the asymptotic region of the functional. The refinement

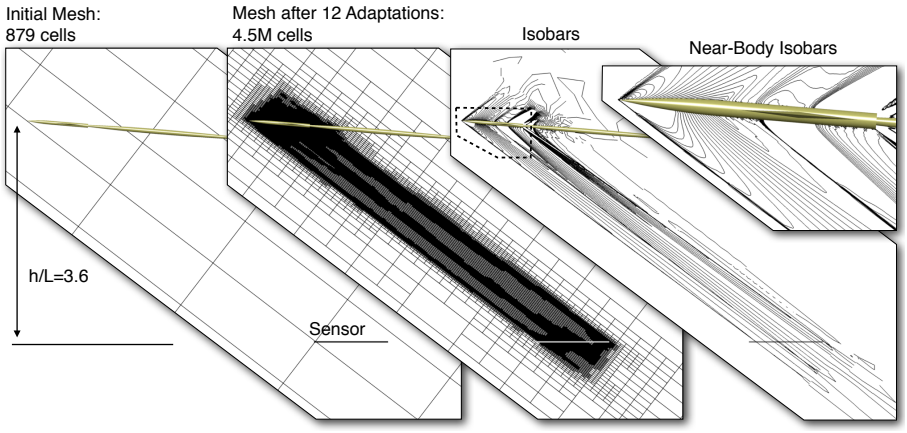


Figure 20. Near-field view of the initial mesh (left), mesh after 12 adaptations (middle) and isobars (right) with inset frame showing isobars close to the model. Pressure signature example (symmetry plane, isobar range 0.65–0.78 in 0.005 increments, $M_\infty = 1.68$)

threshold for each adaptation cycle is set to the mean error value if the error indicator is decreasing, otherwise it is set to twice the mean value to limit over-refinement. The maximum number of cells, N_{max}, is set to 20 million cells.

The middle and right frames of Fig. 20 show the mesh and flow solution after 12 adaptations. Before addressing verification, we briefly discuss the salient features of the refinement pattern. As expected, the refinement follows the shocks and expansions between the model and the sensor; however, note the refinement above the model and below the sensor. Similar to the vortex problem of Sec. 6, we visualize the functional's zone of dependence through the adjoint solution. Figure 21 shows the absolute value of the first adjoint variable. The grayscale map is tuned to indicate where point-source perturbations of the mass-conservation equation influence the pressure signature. In addition to the region between the body and the sensor, the plot confirms the dependence on regions above the model and below the sensor. This is consistent with the expected propagation of characteristics in this three-dimensional flowfield. Moreover, no refinement occurs upstream of the leading shock because the adjoint solution and the flow residual are zero[1], which makes $|\boldsymbol{\eta}|_H$ vanish. There is also little refinement in the wake. This happens because the adjoint vanishes downstream of the sensor, since residual perturbations past the sensor in supersonic flow cannot affect the signature. Lastly, the sensitivity of the error indicator to changes in adjoint curvature triggers the refinement near the sensor (middle frame of Fig. 20 and Fig. 21) to accommodate both the flow and adjoint solutions on the same mesh. This is similar to the discussion around Fig. 13 in Sec. 6.2.

[1]The adjoint is zero due to the quadratic form of J.

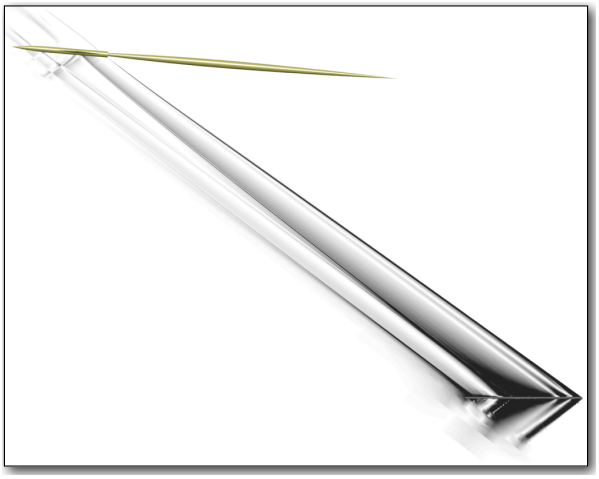
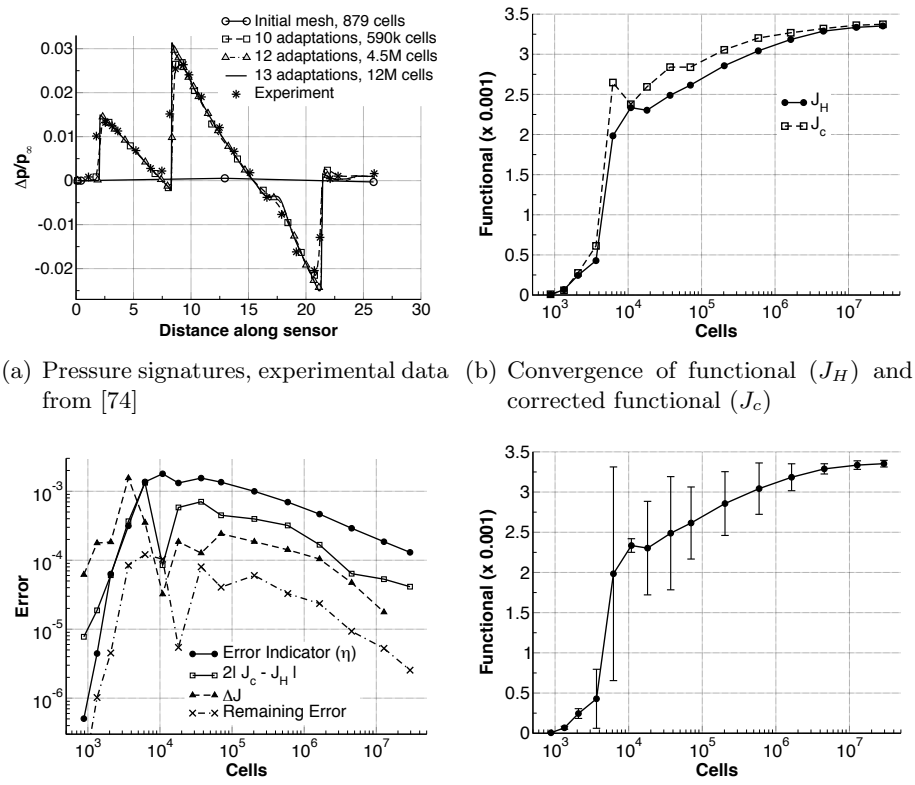


Figure 21. Contours of density-adjoint ($|\psi_0|$, white 0, black 0.012) on symmetry plane

To demonstrate verification, Fig. 22 presents convergence of the signature, the functional and its error estimates. Figure 22(a) shows the classical, qualitative evidence of mesh convergence by examining changes in the signature as the mesh is refined. The main features of the signature are captured within ten adaptations and the signal is converged to plotting accuracy in twelve adaptations. We also validate the signature using experimental data ($*$ symbols) from [74]. The agreement between the experiment and the simulation is excellent.

Figures 22(b)–22(d) show quantitative evidence of convergence as the mesh is refined. Figure 22(b) shows convergence of the functional J_H (solid circles) and the corrected functional J_c (open squares). Changes in these quantities begin to taper once the mesh reaches 200,000 cells, where the corrected functional begins to predict the value of the functional on the next mesh with increasing accuracy. Figure 22(c) shows convergence of the various error estimates. In general, the error increases over the first five cycles and thereafter decreases. As in the airfoil database example (recall Fig. 16), the increasing error indicates new features emerging in the flowfield due to the coarse initial mesh.

The top line (solid circles) in Fig. 22(c) shows convergence of the error-indicator bound η. The second line (squares) shows convergence of the error estimate $\mathcal{E}$ of Eq. 37 with $C = 2$. This assumes J_H is $\mathcal{O}(h)$, which is conservative and expected because both the flow and geometry are dominated by non-smooth features. The error-indicator bound is larger than the error estimate, but both show similar convergence behavior. The third line (solid triangles) monitors the magnitude of the functional update ($\Delta J = |J_H^i - J_H^{i-1}|$) for each cycle of the adaptation. Once the mesh reaches 70,000 cells, the update begins to converge and becomes smaller than the error estimate. The fourth line ($\times$ symbols) shows the magnitude of the remaining error term, Eqs. 36 and 38 without



(a) Pressure signatures, experimental data from [74]

(b) Convergence of functional (J_H) and corrected functional (J_c)

(c) Convergence of error estimates

(d) Functional convergence with error bars showing estimate of discretization error

Figure 22. Mesh convergence for the pressure signature example

localization. Localization increases the magnitude of the error indicator η by about two orders. This is only slightly larger than what is shown in Fig. 10(b) for the vortex problem despite the significantly more complex flow. Furthermore, this confirms that the remaining error term in Eq. 32 is small relative to the adjoint correction.

Figure 22(d) combines information presented in Figs. 22(b) and 22(c) to concisely demonstrate simulation verification. The convergence of the functional is shown with error bars that represent the estimated discretization error $\mathcal{E}$ (taken from the second line of Fig. 22(c)). The largest error occurs on the fourth adaptation and the error estimate sharpens considerably by the tenth adaptation (600,000 cells). Starting from this mesh, the error estimates reliably bracket the solution obtained on the finest mesh (29.4 million cells) indicating asymptotic convergence.

Figure 23 shows error histograms and the mean error value for the initial, fifth and final meshes. The left frame of Fig. 23 shows that most of the cells of the initial mesh contribute no error, which is due to the uniform starting mesh. The middle frame of Fig. 23 shows the histogram after five adaptations (sixth mesh), where the error indicator η reaches

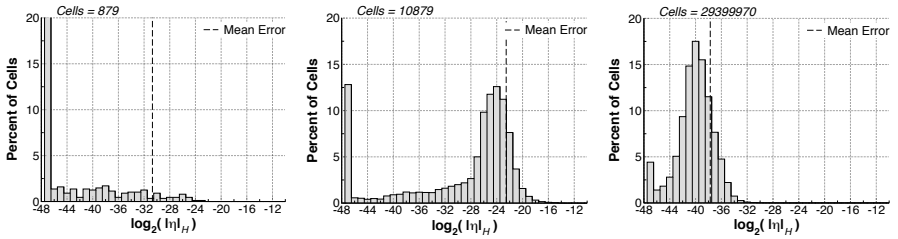


Figure 23. Histograms showing the distribution of error-indicator values for the initial mesh (left), mesh after five adaptations (middle) and final mesh (right) for the pressure signature example.

its maximum value (as shown in Fig. 22(c)). This mesh contains about 11,000 cells and the histogram shows the beginnings of error equidistribution, with the mode of the distribution already close to the mean. The right frame of Fig. 23 presents the histogram of the final mesh. The mode moved to the left, the mean stayed close to the mode and the error distribution sharpened considerably.

In terms of performance, the total simulation (wall-clock) time was 1 hour 42 minutes on 96 Intel Xeon E5-460L cores [77]. The flow and adjoint solutions on the final mesh (29.4 million cells) required about 20 minutes each. In an engineering setting, Fig. 22(d) indicates that it would be sensible to terminate the simulation after twelve adaptations (omit the last two solutions). Convergence of the functional and error becomes predictable by the tenth adaptation and the level of error is quite small after twelve adaptations (4.5 million cells). Moreover, Fig. 22(a) shows that after twelve adaptations there are essentially no changes in the signature. If the simulation is stopped after twelve adaptations, the wall-clock time is only 18 minutes, where the flow and adjoint solutions require about 4 minutes on the final mesh. Additional speedup is possible by omitting the error analysis on the final mesh, which would reduce the net wall-clock time to about ten minutes on these CPU's.

7.3 Launch-Abort Vehicle

The final example involves the prediction of aerodynamic forces for a launch-abort-vehicle prototype. The example explores the limits of Procedure 1 when applied to problems that contain regions of separated, recirculating flow. Referring to Figs. 24 and 25, the vehicle consists of a crew module with a tower-mounted abort system. During a launch emergency, the four Abort Motors (AMs) ignite to pull the crew module safely away from the rocket stack. Throughout the abort, stability and control of the vehicle is maintained by differential thrust from eight Attitude Control Motors (ACMs) near the nose. For this example, we consider a high-altitude abort case studied previously in [78] at $M_\infty = 4$ and $\alpha = 20^\circ$ that involves strong ACM jets.

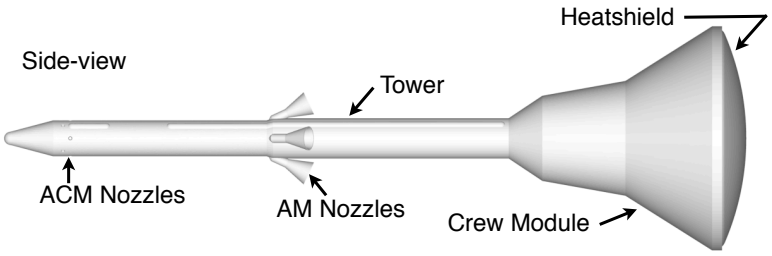


Figure 24. Launch-abort vehicle surface geometry

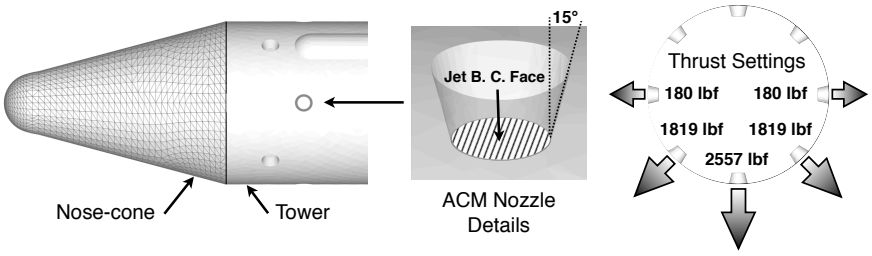


Figure 25. Close-up view of the nose-cone (shown with triangulation) and the ACM nozzles. The thrust settings are indicated in the cutaway view of the right frame.

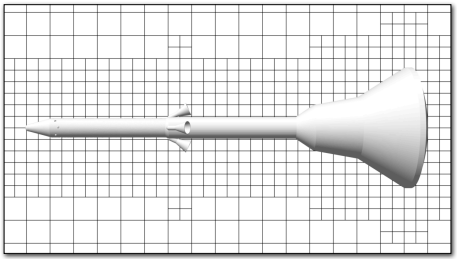


Figure 26. Near-body view of initial mesh on symmetry plane

A close-up view of the nose with the ACM nozzles is shown on the left side of Fig. 25. The jet boundary conditions are applied at the throat faces of the nozzles, which are recessed inside the tower, as shown in the middle frame of Fig. 25. The jet boundary conditions match the exit momentum, pressure and thrust of the ACM performance data [78]. The right side of Fig. 25 shows the thrust setting for each nozzle. The largest thrust is generated by the bottom (south facing) nozzle and five of the eight nozzles are active.

The surface discretization of the vehicle contains roughly 380,000 triangles and is obtained directly from a CAD model. Figure 25 shows an example of the triangulation for the nose-cone component. Figure 26 shows the initial mesh containing 12,880 cells, with only 580 cut cells. As in the previous examples, there is no bias to anticipate features of the flow solution and multiple ACM nozzles are contained within a single cut cell. The refinement threshold for each adaptation cycle is set to the mean error value if the error indicator is decreasing, otherwise it is set to twice the mean value (same as the pressure signature example 7.2). The output of interest is a weighted sum of normal and axial force coefficients

$$J = C_N + 0.4C_A \tag{42}$$

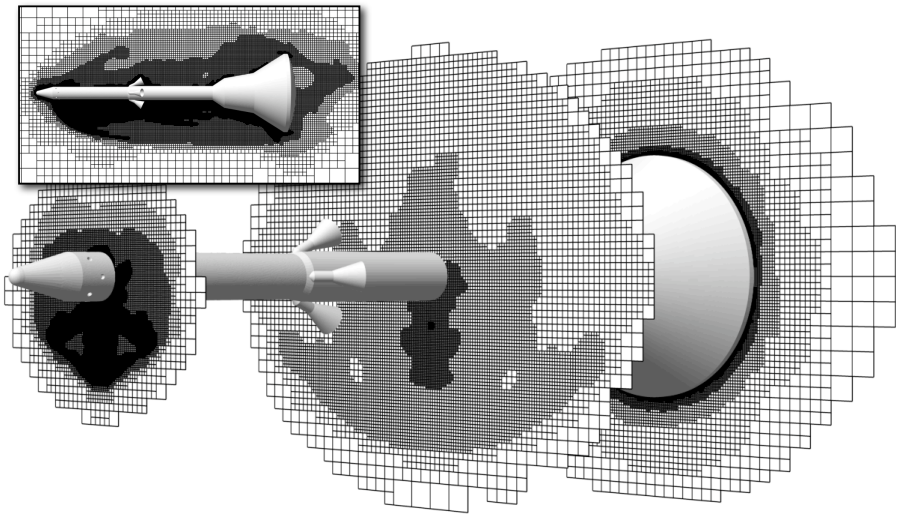


Figure 27. Near-body views of the mesh after 12 adaptations ($M_\infty = 4$ and $\alpha = 20^\circ$)

and N_{max} is set to 50 million cells.

The simulation uses the *minmod* limiter and runs for 13 adaptations. For the last two adaptations the mesh growth is fixed at 2.5 and 3. Figure 27 shows the near-body mesh after 12 adaptations. The mesh contains 16.9 million cells with the finest cells located near the nose, in the windward region near the centerline and at the heatshield shoulder. The refinement of the wake and above the vehicle is moderate, as shown in the inset of Fig. 27. Figures 28 and 29 show the solution after 12 adaptations. The Mach number contours reveal the interaction of several shocks and expansions. The left frame of Fig. 29 shows a close-up of the nose bow-shock and the south (downward pointing) ACM jet on symmetry plane. The bow-shock impinges on the ACM jet-shock and creates a channel of supersonic flow that decelerates through a series of shocks to subsonic flow just upstream of the ACM nozzle, as shown in the left inset of Fig. 29. This is a well-known shock-shock interaction pattern that Edney [79] classified as Type IV interference. The right frame of Fig. 29 shows the front view of the solution on a plane just behind the ACM nozzles. Note the distortion of the bow shock due to the ACM jets.

The salient feature of the flow is the interaction of the ACM jets with the abort motor and crew module surfaces. The jet paths cannot be determined in advance of the computation, yet they have a significant influence on forces, moments and heat transfer. In particular, the interaction of the jets with the surrounding flow can adversely affect the ACM's authority over pitching moment [78]. Figures 27 and 28 show that the finest cells track the bow and jet shocks away from the body and the first transverse cut-plane shows a highly refined mesh for the three main jets near the nose. Such fine cells are necessary because discretization errors introduced in this upstream region influence the functional over

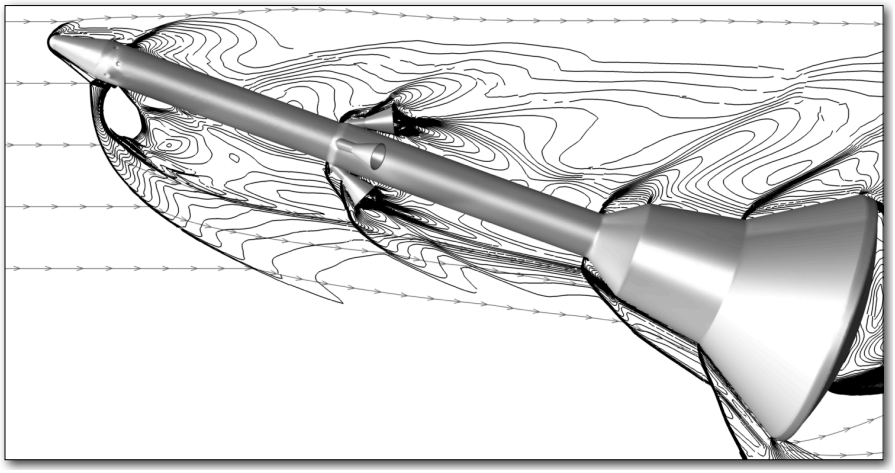


Figure 28. Near-field view of Mach isocontours after 12 adaptations on the symmetry plane. Contours are compressed using $\sqrt{M}$ to improve jet visualization. The surface of the vehicle is shaded by pressure coefficient ($M_\infty = 4$ and $\alpha = 20°$).

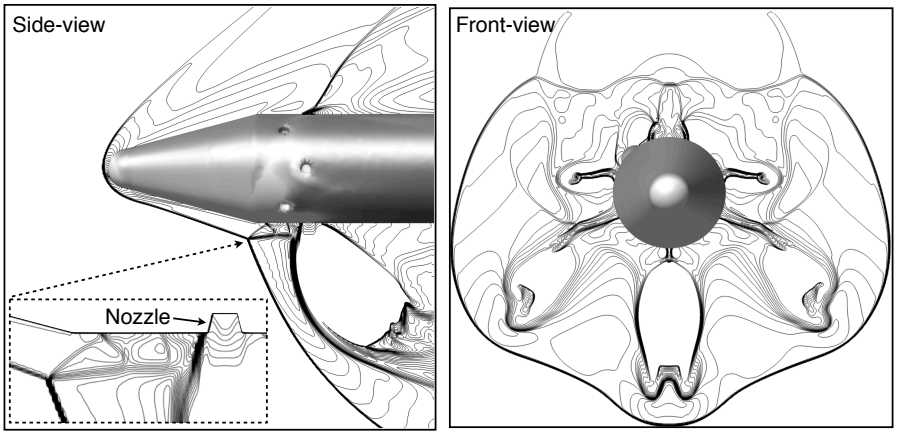


Figure 29. Close-up views near the nose showing Mach isocontours: side-view on symmetry plane (left) and front-view with a cutting plane just behind ACM nozzles (right). Inset on left shows details of the ACM shock interaction ($M_\infty = 4$ and $\alpha = 20°$).

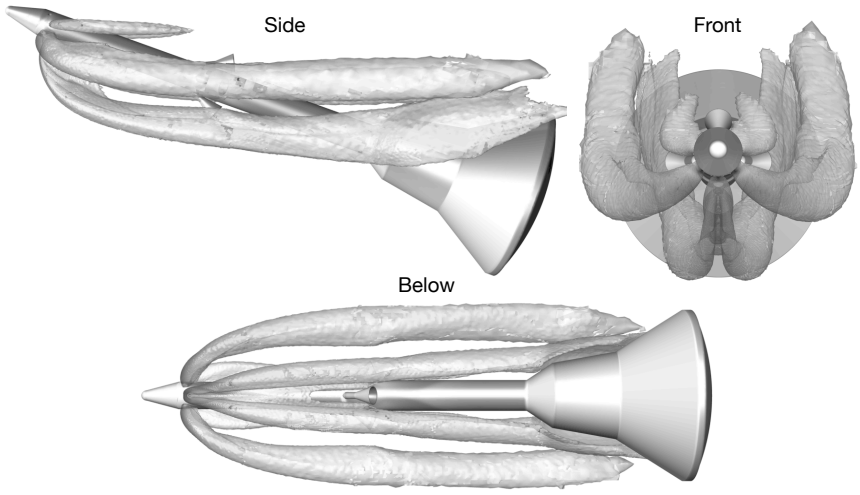
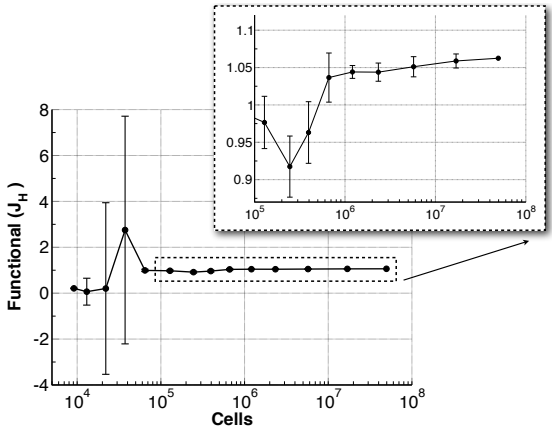


Figure 30. Jet paths around the vehicle ($M_\infty = 4$ and $\alpha = 20°$)

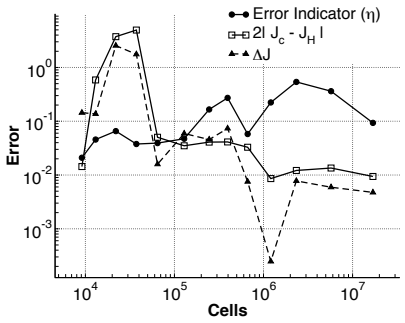
the rest of the vehicle. Further along the vehicle, the second cut-plane of Fig. 27 shows the adaptation pattern still tracking the jet paths, but with less refinement. The final cut-plane through the heatshield shoulder is nearly symmetric, indicating that at this downstream station the jets have negligible influence on the functional.

To further investigate the refinement of the jets, we use iso-surfaces of stagnation enthalpy to visualize the jet paths around the vehicle in Fig. 30. The jet surfaces are shaded by Mach number. The bifurcation of the main jet is striking. This is a consequence of the over-pressure at the ACM nozzle exit caused by the Type IV interference, followed by downstream interactions of the jet with the supersonic flow. The bifurcated jet contacts the sides of the crew module as it is swept upward by the high angle-of-attack flow. The "Below" view of Fig. 30 shows that the shocks from the abort motor nozzles alter the jet paths away from the crew module and thereby alter the heating experienced by the crew module. Lastly, as indicated previously in Fig. 27 and shown in Fig. 30, the refinement of the jets wanes as the flow approaches the heatshield shoulder. While in reality the jets persist well into the wake of the vehicle, the error analysis truncates their refinement once they pass beyond the heatshield, where they can no longer impact the functional. This is the same behavior as we observed in the supersonic regime of the airfoil database example, but here in a much more complex flow.

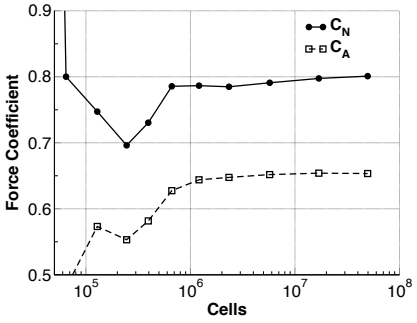
Figure 31 summarizes the convergence of the functional, the error estimates and the aerodynamic forces. Figure 31(a) shows the value of the functional on each mesh along with error bars representing the level of discretization error $\mathcal{E}$ via Eq. 37 with $C = 2$. As in the pressure signature example, we assume that J_H is $\mathcal{O}(h)$ because the flow is dominated by shocks and contact discontinuities. The large error estimates in the second and third adaptation are primarily associated with refinement of



(a) Functional convergence with error bars showing estimate of discretization error



(b) Convergence of error estimates



(c) Convergence of normal and axial force coefficients

Figure 31. Mesh convergence for the launch-abort vehicle example

the ACM jets. The inset in Fig. 31(a) shows that over the last three adaptations the error bars bracket the functional and the changes in the functional are less than 1%. Figure 31(b) shows convergence of the error-indicator bound η, the error estimate $\mathcal{E}$ and magnitude of the functional update ($\Delta J = |J_H^i - J_H^{i-1}|$). While convergence is not as convincing as in the previous examples, all three error measures are decreasing once the mesh reaches about 2 million cells.

Despite the complicated flowfield, especially the recirculations behind the heatshield and the inactive abort motors, Procedure 1 remains effective in addressing critical regions of the flow that influence the outputs. Figure 31(c) shows that there are virtually no changes in the axial and normal force coefficients over the last two adaptations. More broadly, this example characterizes the performance of Procedure 1 in verification of difficult engineering simulations, and demonstrates the benefits of affordable and automatic error control in providing insight into complex flowfields.

8 Conclusions

A procedure for automatic control of discretization error in steady simulations of inviscid flow has been presented. The procedure involves error estimates based on the method of adjoint-weighted residuals in conjunction with incremental mesh enrichment based on a Cartesian cut-cell method. We draw the following conclusions from our results:

- The code verification example demonstrates that both the functional J_H and corrected functional J_c are $\mathcal{O}(h^2)$, which is the expected order of accuracy. The error-indicator bound η converges at about the same rate due to the discontinuous adjoint field and localization. This is sufficient for the discretization error estimate $\mathcal{E}$ to be $\mathcal{O}(h^2)$ with effectivity close to one.

- The simulation verification examples show that once the mesh is sufficiently fine, $\mathcal{E}$ reliably brackets J_H and decreases by about a factor of two per adaptation cycle.

- The error map $|\boldsymbol{\eta}|_H$ reliably identifies critical regions of the mesh. Consequently, the procedure reliably handles extremely coarse initial meshes and generates a converging sequence of affordable meshes that accurately predict the output.

- The procedure offers a practical alternative to the manual generation of simulation-specific meshes that encompass the knowledge and experience of specialists and instead automatically delivers meshes and error estimates that provide significant insight into the simulation.

- Since a mesh refinement study is intrinsic to every simulation, the procedure automatically provides convergence histories of outputs of interest and error estimates. This makes the task of obtaining verified simulations straightforward and essentially automatic.

- Returning to the opening quote of the Introduction; the results show that the procedure attains a sufficient level of reliability and robustness for routine use in simulation verification.

There are many areas for future work. For example, improving the accuracy of the embedded adjoint representation through use of affordable approximate solutions could improve the sharpness of the error estimate. Incorporating the procedure within simulation-based design could significantly reduce cost of optimization problems. More broadly, since simulations in practice frequently involve some degree of unsteady flow (as in the launch abort example of Sec. 7.3), the procedure should be extended to transition smoothly from steady to unsteady flow, especially for applications where the time variation in the outputs is relatively small. Furthermore, the extension of the procedure to handle flows involving

multiple components, or species, and turbulence is an important area of future work.

Acknowledgments

This work was supported by the NASA Ames Research Center contract NNA10DF26C and by NASA mission directorates for Aeronautics Research, and Human Exploration and Operations. The authors gratefully thank Marsha Berger (NYU), Mathias Wintzer (Analytical Mechanics Associates, Inc.) and George Anderson (Stanford University) for helpful discussions and suggestions.

References

[1] Baker, T. J., "Mesh Adaptation Strategies for Problems in Fluid Dynamics," *Finite Elements in Analysis and Design*, Vol. 25, No. 3/4, 1997, pp. 243–273.

[2] Bussoletti, J. E., Johnson, F. T., Bieterman, M. B., Hilmes, C. L., Melvin, R. G., Young, D. P., and Drela, M., "TRANAIR: Solution adaptive CFD modeling for complex 3D configurations," *Recent developments and applications in aeronautical CFD*, Proceedings of the 1993 European Forum, London, United Kingdom, 1993, pp. 10.1–10.14.

[3] Löhner, R., "Mesh Adaptation in Fluid Mechanics," *Engineering Fracture Mechanics*, Vol. 50, No. 5/6, 1995, pp. 819–847.

[4] Habashi, W. G., Dompierre, J., Bourgault, Y., Fortin, M., and Vallet, M., "Certifiable Computational Fluid Dynamics Through Mesh Optimization," *AIAA Journal*, Vol. 36, No. 5, 1998, pp. 703–711.

[5] Fidkowski, K. J. and Darmofal, D. L., "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics," *AIAA Journal*, Vol. 49, No. 4, April 2011, pp. 673–694.

[6] Roache, P. J., "Verification of Codes and Calculations," *AIAA Journal*, Vol. 36, No. 5, May 1998, pp. 696–702.

[7] Oberkampf, W. L. and Trucano, T. G., "Verification and Validation in Computational Fluid Dynamics," *Progress in Aerospace Sciences*, Vol. 38, 2002, pp. 209–272.

[8] Roy, C. J., "Review of Code and Solution Verification Procedures for Computational Simulation," *Journal of Computational Physics*, Vol. 205, 2005, pp. 131–156.

[9] National Aeronautics and Space Administration (NASA), "Standard for Models and Simulations," Technical Standard NASA-STD-7009, 2008, http://standards.nasa.gov, visited March 2014.

[10] Slotnick, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD Vision 2030 Study: A Path to Revolutionary Computational Aerosciences," NASA CR 2014-218178, March 2014.

[11] Becker, R. and Rannacher, R., "An optimal control approach to a posteriori error estimation in finite element methods," *Acta Numerica 2000*, 2001, pp. 1–102.

[12] Süli, E. and Houston, P., "Adaptive Finite Element Approximation of Hyperbolic Problems," *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, edited by T. Barth and H. Deconinck, Vol. 25 of *Lecture Notes in Computational Science and Engineering*, Springer-Verlag, 2002.

[13] Giles, M. B. and Pierce, N. A., "Adjoint error correction for integral outputs," *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, edited by T. Barth and H. Deconinck, Vol. 25 of *Lecture Notes in Computational Science and Engineering*, Springer-Verlag, 2002.

[14] Barth, T., "Numerical Methods and Error Estimation for Conservation Laws on Structured and Unstructured Meshes," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 2003-04, Brussels, Belgium, March 2003.

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

[16] Hartmann, R., "Multitarget error estimation and adaptivity in aerodynamic flow simulations," *SIAM J. Sci. Comput.*, Vol. 31, No. 1, 2008, pp. 708–731.

[17] Dwight, R. P., "Heuristic *a posteriori* estimation of error due to dissipation in finite volume schemes and application to mesh adaptation," *Journal of Computational Physics*, Vol. 227, No. 5, 2008, pp. 2845–2863.

[18] Wang, L. and Mavriplis, D. J., "Adjoint-based *h-p* adaptive discontinuous Galerkin methods for the 2D compressible Euler equations," *Journal of Computational Physics*, Vol. 228, 2009, pp. 7643–7661.

[19] Krakos, J. A. and Darmofal, D. L., "Effect of Small-Scale Output Unsteadiness of Adjoint-Based Sensitivity," *AIAA Journal*, Vol. 48, No. 11, Nov. 2010, pp. 2611–2623.

[20] Yi, L. Y., Allaneau, Y., and Jameson, A., "Continuous Adjoint Approach for Adaptive Mesh Refinement," AIAA Paper 2011–3982, June 2011.

[21] Yano, M. and Darmofal, D. L., "An optimization-based framework for anisotropic mesh adaptation," *Journal of Computational Physics*, Vol. 231, No. 22, 2012, pp. 7626–7649.

[22] Copeland, S. R., Lonkar, A. K., Palacios, F., and Alonso, J. J., "Adjoint-Based Goal-Oriented Mesh Adaptation for Nonequilibrium Hypersonic Flows," AIAA Paper 2013–0552, Grapevine, TX, Jan. 2013.

[23] Park, M. A. and Darmofal, D. L., "Validation of an Output-Adaptive, Tetrahedral Cut-Cell Method for Sonic Boom Prediction," *AIAA Journal*, Vol. 48, No. 9, Sept. 2010, pp. 1928–1945.

[24] Park, M. A., Lee-Rausch, E. M., and Rumsey, C. L., "FUN3D and CFL3D Computations for the First High Lift Prediction Workshop," AIAA Paper 2011–0936, Jan. 2011.

[25] Hartmann, R., Held, J., and Leicht, T., "Adjoint-based error estimation and adaptive mesh refinement for the RANS and *k-w* turbulence model equations," *Journal of Computational Physics*, Vol. 230, No. 11, May 2011, pp. 4268–4284.

[26] Richter, T., "Goal-oriented error estimation for fluid-structure interaction problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 223–224, 2012, pp. 28–42.

[27] Ceze, M. and Fidkowski, K. J., "Drag Prediction Using Adaptive Discontinuous Finite Elements," *Journal of Aircraft*, 2014.

[28] Aftosmis, M. J., Berger, M. J., and Melton, J. E., "Robust and Efficient Cartesian Mesh Generation for Component-Based Geometry," *AIAA Journal*, Vol. 36, No. 6, 1998, pp. 952–960.

[29] Berger, M. J. and Aftosmis, M. J., "Aspects (and Aspect Ratios) of Cartesian Mesh Methods," Proceedings of the 16th international conference on numerical methods in fluid dynamics, Arcachon, France, July 1998.

[30] Colella, P., Graves, D. T., Keen, B. J., and Modiano, D., "A Cartesian grid embedded boundary method for hyperbolic conservation laws," *Journal of Computational Physics*, Vol. 211, 2006, pp. 347–366.

[31] Fidkowski, K. J. and Darmofal, D. L., "A triangular cut-cell adaptive method for high-order discretizations of the compressible Navier-Stokes equations," *Journal of Computational Physics*, Vol. 225, No. 6, 2007, pp. 1653–1672.

[32] Meinke, M., Schneiders, L., Günther, C., and Schröder, W., "A cut-cell method for sharp moving boundaries in Cartesian grids," *Computers & Fluids*, Vol. 85, 2013, pp. 135–142.

[33] Demargne, A. A. J., Evans, R. O., Tiller, P. J., and Dawes, W. N., "Practical and Reliable Mesh Generation for Complex Real-world Geometries," AIAA Paper 2014–0119, Jan. 2014.

[34] Nemec, M., Aftosmis, M. J., Murman, S. M., and Pulliam, T. H., "Adjoint Formulation for an Embedded-Boundary Cartesian Method," AIAA Paper 2005–0877, Reno, NV, Jan. 2005.

[35] Nemec, M. and Aftosmis, M. J., "Adjoint Error Estimation and Adaptive Refinement for Embedded-Boundary Cartesian Meshes," AIAA Paper 2007–4187, Miami, FL, June 2007.

[36] Nemec, M., Aftosmis, M. J., and Wintzer, M., "Adjoint-Based Adaptive Mesh Refinement for Complex Geometries," AIAA Paper 2008–0725, Reno, NV, Jan. 2008.

[37] Aftosmis, M. J. and Berger, M. J., "Multilevel Error Estimation and Adaptive h-Refinement for Cartesian Meshes with Embedded Boundaries," AIAA Paper 2002–0863, Reno, NV, Jan. 2002.

[38] Wintzer, M., Nemec, M., and Aftosmis, M. J., "Adjoint-Based Adaptive Mesh Refinement for Sonic Boom Prediction," AIAA Paper 2008-6593, Honolulu, HI, Aug. 2008, (26th AIAA Applied Aerodynamics Conference).

[39] Aftosmis, M. J. and Nemec, M., "Exploring Discretization Error in Simulation-Based Aerodynamic Databases," *Proceedings of the* 21^{st} *International Conference on Parallel Computational Fluid Dynamics*, ParCFD 2009, May 2009.

[40] Wintzer, M., "Span Efficiency Prediction Using Adjoint-Driven Mesh Refinement," *Journal of Aircraft*, Vol. 47, No. 4, July-August 2010, pp. 1468–1471.

[41] Bakhtian, N. and Aftosmis, M. J., "Analysis of Inviscid Simulations for the Study of Supersonic Retropropulsion," AIAA Paper 2011-3194, Honolulu, HI, June 2011, (29th AIAA Applied Aerodynamics Conference).

[42] Kless, J. E. and Aftosmis, M. J., "Analysis of Grid Fins for Launch Abort Vehicle Using a Cartesian Euler Solver," AIAA Paper 2011-3666, Honolulu, HI, June 2011, (29th AIAA Applied Aerodynamics Conference).

[43] Aftosmis, M. J., Nemec, M., and Cliff, S. E., "Adjoint-Based Low-Boom Design with Cart3D," AIAA Paper 2011-3500, Honolulu, HI, June 2011, (29th AIAA Applied Aerodynamics Conference).

[44] Kless, J., Aftosmis, M. J., Ning, S. A., and Nemec, M., "Inviscid Analysis of Extended-Formation Flight," *AIAA Journal*, Vol. 51, No. 7, July 2013, pp. 1703–1715.

[45] Ning, S. A., Kroo, I., Aftosmis, M. J., Nemec, M., and Kless, J. E., "Extended Formation Flight at Transonic Speeds," *Journal of Aircraft*, January 2014, pp. 0–0, http://dx.doi.org/10.2514/1.C032385.

[46] Berger, M. J., Aftosmis, M. J., and Murman, S. M., "Analysis of Slope Limiters on Irregular Grids," AIAA Paper 2005–0490, Reno, NV, Jan. 2005.

[47] van Leer, B., "Flux-Vector Splitting for the Euler Equations," ICASE Report 82-30, Sept. 1982.

[48] Aftosmis, M. J., "Solution Adaptive Cartesian Grid Methods for Aerodynamic Flows with Complex Geometries," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 1997-02, Brussels, Belgium, March 1997.

[49] Aftosmis, M. J., Berger, M. J., and Adomavicius, G., "A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries," AIAA Paper 2000–0808, Reno, NV, Jan. 2000.

[50] Aftosmis, M. J., Berger, M. J., and Murman, S. M., "Applications of Space-Filling-Curves to Cartesian Methods for CFD," AIAA Paper 2004–1232, Reno, NV, Jan. 2004.

[51] Berger, M. J., Aftosmis, M. J., , Marshall, D. D., and Murman, S. M., "Performance of a New CFD Flow Solver Using a Hybrid Programming Paradigm," *J. of Parallel and Distributed Computing*, Vol. 65, 2005, pp. 414–423.

[52] Lu, J., *An* a posteriori *Error Control Framework for Adaptive Precision Optimization using Discontinuous Galerkin Finite Element Method*, Ph.D. thesis, Massachusetts Institute of Technology, 2005.

[53] Hicken, J. E. and Zingg, D. W., "Dual consistency and functional accuracy: a finite-difference perspective," *Journal of Computational Physics*, Vol. 256, 2014, pp. 161–182.

[54] Nemec, M., *Optimal Shape Design of Aerodynamic Configurations: A Newton–Krylov Approach*, Ph.D. thesis, University of Toronto, 2003.

[55] Barth, T. J., "Parallel CFD Algorithms on Unstructured Meshes," *Special Course on Parallel Computing in CFD*, AGARD-R-807, Oct. 1995, pp. 7–1–7–41.

[56] Giles, M. B., Duta, M. C., Müller, J.-D., and Pierce, N. A., "Algorithm Developments for Discrete Adjoint Methods," *AIAA Journal*, Vol. 41, No. 2, 2003, pp. 198–204.

[57] Nielsen, E. J., Lu, J., Park, M. A., and Darmofal, D. L., "An Implicit, Exact Dual Adjoint Solution Method for Turbulent Flows on Unstructured Grids," *Computers & Fluids*, Vol. 33, 2004, pp. 1131–1155.

[58] Mavriplis, D. J., "Multigrid Solution of the Discrete Adjoint for Optimization Problems on Unstructured Meshes," *AIAA Journal*, Vol. 44, No. 1, 2006, pp. 42–50.

[59] Park, M. A., *Anisotropic Output-Based Adaptation with Tetrahedral Cut Cells for Compressible Flows*, Ph.D. thesis, Massachusetts Institute of Technology, 2008.

[60] Giles, M. B., "On the iterative solution of adjoint equations," *Automatic Differentiation of Algorithms: From Simulation to Optimization*, edited by G. Corliss, C. Faure, A. Griewank, L. Hascoët, and U. Naumann, Springer-Verlag, 2002.

[61] Giles, M. B., "On the Use of Runge–Kutta Time-Marching and Multigrid for the Solution of Steady Adjoint equations," Oxford University Computing Laboratory 00/10, June 2000.

[62] Jones, W. T., Nielsen, E. J., and Park, M. A., "Validation of 3D Adjoint Based Error Estimation and Mesh Adaptation for Sonic Boom Prediction," AIAA Paper 2006–1150, Jan 2006.

[63] Barth, T. J. and Jespersen, D., "The Design and Application of Upwind Schemes on Unstructured Meshes," AIAA Paper 1989–0366, Jan 1989.

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

[65] Dannenhoffer III, J. F. and Baron, J. R., "Adaptive Procedure for Steady State Solution of Hyperbolic Equations," AIAA Paper 84–0005, Jan. 1984.

[66] Kallinderis, Y. G. and Baron, J. R., "Adaptation Methods for a New Navier–Stokes Algorithm," *AIAA Journal*, Vol. 27, No. 1, Jan. 1989, pp. 37–43.

[67] Aftosmis, M. J., Gaitonde, D., and Tavares, T. S., "Behavior of Linear Reconstruction Techniques on Unstructured Meshes," *AIAA Journal*, Vol. 33, No. 11, Nov. 1995, pp. 2038–2049.

[68] Abbott, I. H. and Doenhoff, A. E. V., *Theory of Wing Sections*, Dover Publications, Inc., 1959.

[69] Page, J. A. and Plotkin, K. J., "An Efficient Method for Incorporating Computational Fluid Dynamics Into Sonic Boom Prediction," AIAA Paper 91–3275, Sept. 1991.

[70] Cliff, S. E. and Thomas, S. D., "Euler/Experiment Correlations of Sonic Boom Pressure Signatures," *Journal of Aircraft*, Vol. 30, No. 5, Sept. 1993, pp. 669–790.

[71] Meredith, K. B., Dahlin, J. H., Graham, D. H., Malone, M. B., Haering Jr., E. A., Page, J. A., and Plotkin, K. J., "Computational Fluid Dynamics Comparison and Flight Test Measurement of F-5E Off-Body Pressures," AIAA Paper 2005–0006, Jan. 2005.

[72] Loseille, A., Dervieux, A., and Alauzet, F., "Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations," *Journal of Computational Physics*, Vol. 229, 2010, pp. 2866–2897.

[73] Cliff, S. E., Elmiligui, A. A., Campbell, R. L., and Thomas, S. D., "Refined Tetrahedral Meshes with Mach Cone Aligned Prisms for Sonic Boom Analysis," *Journal of Aircraft*, Vol. 50, No. 3, May 2013, pp. 778–790.

[74] Hunton, L. W., Hicks, R. M., and Mendoza, J. P., "Some Effects of Wing Planform on Sonic Boom," NASA TN D–7160, Jan. 1973.

[75] Park, M. A., Aftosmis, M. J., Campbell, R. L., Carter, M. B., Cliff, S. E., and Bangert, L. S., "Summary of the 2008 NASA Fundamental Aeronautics Program Sonic Boom Prediction Workshop," AIAA Paper 2013–0649, Jan. 2013.

[76] Aftosmis, M. J. and Nemec, M., "Cart3D Simulations for the First AIAA Sonic Boom Prediction Workshop," AIAA Paper 2014–0558, Jan. 2014.

[77] NASA, "Endeavour Supercomputer, NAS Division," April 2014, http://www.nas.nasa.gov/hecc/resources/endeavour.html.

[78] Aftosmis, M. J. and Rogers, S. E., "Effects of Jet-Interaction on Pitch Control of a Launch Abort Vehicle," AIAA Paper 2008–1281, Jan. 2008.

[79] Edney, B. E., "Anomalous Heat Transfer and Pressure Distributions on Blunt Bodies at Hypersonic Speeds in the Presence of an Impinging Shock," FFA Report 115, The Aeronautical Research Institute of Sweden, June 1968.