AIAA JOURNAL
Vol. 54, No. 9, September 2016

High-Order Output-Based Adaptive Simulations of Turbulent Flow in Two Dimensions

Marco A. Ceze*
NASA Ames Research Center, Moffett Field, California
and
Krzysztof J. Fidkowski†
University of Michigan, Ann Arbor, Michigan 48109

DOI: 10.2514/1.J054517

Output-based high-order adaptive results are presented for several benchmark two-dimensional turbulent-flow simulations. The discretization is a high-order discontinuous Galerkin finite element method, and the equations solved are compressible Navier–Stokes, Reynolds-averaged with a modified version of the Spalart–Allmaras one-equation model. Mesh refinement requirements are studied through automated output-based adaptation in which a discrete adjoint solution associated with an output (e.g., the drag coefficient) weights a fine-space residual and automatically selects the elements that need more resolution. The roles of high-order and mesh anisotropy are also investigated. Finally, differences are investigated between two mesh refinement strategies: hanging-node refinement of structured meshes versus metric-based remeshing of unstructured triangles.

I. Introduction

ALTHOUGH improvements in computing capabilities have made advanced computational fluid dynamics techniques such as large-eddy simulation (LES) possible for a range of applications, the Reynolds-averaged Navier–Stokes (RANS) equations remain an invaluable tool routinely used in analysis and design. Compared with LES, RANS simulations are much cheaper because they can take advantage of anisotropic (stretched) computational elements that reduce the degrees of freedom required to accurately resolve thin boundary and shear layers. This advantage is not always easy to realize, in particular for high-order methods that require curved elements, which are difficult to keep from tangling/inverting when stretched.

High-order methods for RANS suffer from additional debate and scrutiny: RANS solutions often possess singular features that do not lend themselves to high-order approximation, and RANS modeling errors are generally viewed as dominant compared with numerical resolution (discretization) errors that high order would address. Regarding the latter point, our position is that both modeling and numerical errors need to be estimated and controlled through methods appropriate for each error. For instance, modeling errors may be addressed through an uncertainty quantification study, and this study may require simulations with different model parameter settings but low discretization errors to isolate the effects of the parameters on the model.

Regarding the former point of RANS solutions containing singular features, these features can be resolved via small elements using a mesh adaptation technique: most flowfields will still possess large smooth regions where high order will be advantageous [1]. In particular, output-based methods [2–5] offer a systematic approach for identifying regions of the domain that require more resolution for the prediction of scalar outputs of interest. These methods also return error estimates that can improve robustness of solution verification and uncertainty quantification studies. It is for these reasons that we consider output-based methods in the present study.

In this paper, we apply a high-order adaptive solution technique to several test cases modeled with the two-dimensional RANS equations, closed with a recent modification of the Spalart–Allmaras (SA) one-equation model [6]. Many previous works have investigated the RANS-SA equations, including in a high-order adaptive setting [1,7–10]. The majority of the latter work has focused on demonstrating benefits of adaptive refinement and/or high order over uniform or heuristic refinement for such flows. These comparisons have been done in solely structured and solely unstructured settings. The present work distinguishes itself in that we compare both structured and unstructured mesh refinement techniques, and that we consider a set of well-defined benchmark test cases with available previous (typically second-order) data.

The remainder of this paper is organized as follows. Section II presents the RANS-SA equations, and Sec. III discusses their discretization. Sections IV and V describe the output error estimation and adaptation techniques, and Sec. VI presents results for the several benchmark cases considered. Section VII concludes with a summary and a discussion of possible future directions.

II. Turbulence Model

We use the Spalart–Allmaras turbulence model, modified for stability for negative values of the turbulence working variable [6]. The RANS equations closed with this turbulence model read

$$\begin{aligned}
&\partial_t\rho+\partial_j(\rho u_j)=0\\
&\partial_t(\rho u_i)+\partial_j(\rho u_j u_i+p\delta_{ij})=\partial_j\tau_{ij}\\
&\partial_t(\rho E)+\partial_j(\rho u_j H)=\partial_j(u_i\tau_{ij}-q_j)\\
&\partial_t(\rho\tilde{\nu})+\partial_j(\rho u_j\tilde{\nu})=\partial_j\left[\frac{1}{\sigma}\rho(\nu+\tilde{\nu}f_n)\partial_j\tilde{\nu}\right]\\
&\qquad -\frac{1}{\sigma}(\nu+\tilde{\nu}f_n)\partial_j\rho\partial_j\tilde{\nu}+\frac{c_{b2}\rho}{\sigma}\partial_j\tilde{\nu}\partial_j\tilde{\nu}+P-D\\
&\qquad \frac{1}{\sigma}\rho\partial_j[(\nu+\tilde{\nu}f_n)\partial_j\tilde{\nu}]+\frac{c_{b2}\rho}{\sigma}\partial_j\tilde{\nu}\partial_j\tilde{\nu}+P-D
\end{aligned}\tag{1}$$

where ρ is the density, ρu_j is the momentum, E is the total energy, $H=E+p/\rho$ is the total enthalpy, $p=(\gamma-1)(\rho E-(1/2)\rho u_k u_k)$ is the pressure, γ is the ratio of specific heats, and i, j index the spatial dimension (dim). The Reynolds stress τ_{ij} is

Received 2 June 2015; revision received 15 October 2015; accepted for publication 24 November 2015; published online 17 February 2016.

*Postdoctoral Fellow, Oak Ridge Associated Universities; marco.a.ceze@nasa.gov.

†Associate Professor, Department of Aerospace Engineering. Senior Member AIAA.

$$\tau_{ij} = 2(\mu + \mu_t)\bar{\epsilon}_{ij}, \qquad \bar{\epsilon}_{ij} = \frac{1}{2}(\partial_i u_j + \partial_j u_i) - \frac{1}{3}\partial_k u_k \delta_{ij}$$

μ is the laminar dynamic viscosity, obtained using Sutherland's law,

$$\mu = \mu_{\text{ref}} \left(\frac{T}{T_{\text{ref}}}\right)^{1.5} \left(\frac{T_{\text{ref}} + T_s}{T + T_s}\right) \tag{2}$$

where T is the temperature, and the eddy viscosity μ_t is

$$\mu_t = \begin{cases} \rho\tilde{\nu} f_{v1} & \tilde{\nu} \geq 0 \\ 0 & \tilde{\nu} < 0 \end{cases} \qquad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \qquad \chi = \frac{\tilde{\nu}}{\nu}$$

The heat flux q_j is given by

$$q_j = (k + k_t)\partial_i T, \qquad k = C_p \mu / Pr, \qquad k_t = C_p \mu_t / Pr_t$$

The production term P is

$$P = \begin{cases} c_{b1} \tilde{S} \rho \tilde{\nu} & \chi \geq 0 \\ c_{b1} S \rho \tilde{\nu} & \chi < 0 \end{cases}$$

where the modified vorticity $\tilde{S}$ is written as

$$\tilde{S} = \begin{cases} S + \bar{S} & \bar{S} \geq -c_{v2} S \\ S + \frac{S(c_{v2}^2 S + c_{v3}\bar{S})}{(c_{v3} - 2c_{v2})S - \bar{S}} & \bar{S} < -c_{v2} S \end{cases}, \quad \bar{S} = \frac{\tilde{\nu} f_{v2}}{\kappa^2 d^2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \tag{3}$$

In Eq. (3), $S = \sqrt{2\Omega_{ij}\Omega_{ij}}$ is the vorticity magnitude (summation implied on i, j), and $\Omega_{ij} = (1/2)(\partial_i v_j - \partial_j v_i)$ is the vorticity tensor; d is the distance to the closest wall. The destruction term D is given by

$$D = \begin{cases} c_{w1} f_w \frac{\rho\tilde{\nu}^2}{d^2} & \chi \geq 0 \\ -c_{w1} \frac{\rho\tilde{\nu}^2}{d^2} & \chi < 0 \end{cases}, \qquad f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{1/6},$$

$$g = r + c_{w2}(r^6 - r), \qquad r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}$$

Finally, the coefficient f_n in Eq. (1) is one for positive $\tilde{\nu}$ and

$$f_n = \frac{c_{n1} + \chi^3}{c_{n1} - \chi^3}, \quad \text{when } \chi < 0 \tag{4}$$

Relevant closure coefficients are

$$c_{b1} = 0.1355 \qquad c_{w1} = \frac{c_{b1}}{\kappa^2} + \frac{1 + c_{b2}}{\sigma} \qquad c_{v1} = 7.1$$
$$c_{b2} = 0.622 \qquad c_{w2} = 0.3 \qquad \kappa = 0.41$$
$$\sigma = 2/3 \qquad c_{w3} = 2 \qquad Pr_t = 0.9$$
$$c_{n1} = 16 \qquad c_{v2} = 0.7 \qquad c_{v3} = 0.9$$

III. Discretization

We discretize Eq. (1) using a discontinuous Galerkin (DG) finite element method [9,11]. Defining the state vector as $\boldsymbol{u} = [\rho, \rho u_i, \rho E, \rho\tilde{\nu}]^T$, we write Eq. (1) in compact conservative form,

$$\partial_t \boldsymbol{u} + \nabla \cdot \boldsymbol{F}(\boldsymbol{u}, \nabla\boldsymbol{u}) + \boldsymbol{S}(\boldsymbol{u}, \nabla\boldsymbol{u}) = 0 \tag{5}$$

where $\boldsymbol{F}$ is the combined inviscid/viscous flux vector, and $\boldsymbol{S}$ is the source term associated with the turbulence closure equation. We approximate the state as $\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h$, where $\boldsymbol{\mathcal{V}}_h$ is the space of elementwise discontinuous polynomials of order p.[‡] Multiplying Eq. (5) by test functions $\boldsymbol{v}_h \in \boldsymbol{\mathcal{V}}_h$, integrating by parts on each element, and using the Roe [12] convective flux and the second form of Bassi and Rebay [13] for the viscous treatment, we obtain the following semilinear form:

$$\mathcal{R}_h(\boldsymbol{u}_h, \boldsymbol{v}_h) = 0 \tag{6}$$

Note that, in Eqs. (1) and (5), the RANS source term depends on the gradient of the state. For the present work, we use an adjoint-inconsistent treatment in which the gradient is taken pointwise directly from the polynomial solution approximation, without consideration of interface jump contributions [14]. Much of the DG discretization is standard; the following sections outline a few practical details.

A. Wall Distance Calculation

The wall distance $d(\boldsymbol{x})$, required in the SA model, is approximated on each element by a polynomial of the same order p as the solution. The procedure for calculating $d(\boldsymbol{x})$ is brute force: Lagrange interpolating polynomials are used, and at each node associated with a Lagrange polynomial in every element, the wall distance is calculated. This calculation considers all of the boundary faces associated with the walls in the domain. The distance to each boundary node is calculated to preselect the closest boundary faces. For each of these boundary faces, which are high-order/curved, the wall distance is estimated by calculating the minimum distances to a set of facets obtained by subdividing the high-order face into $2(Q + 1)$ linear segments. Once the minimum-distance facet is found, reference-space coordinates on the face corresponding to a projection onto this facet are used to compute a point on the true high-order geometry. The wall distance is then the distance to this point on the true geometry. Although this distance function calculation could be made more efficient, its cost is negligible compared with the flow solution.

B. Symmetry Boundary Conditions

Several test cases in the results call for symmetry boundary conditions (BCs). In the continuous limit, symmetry requires vanishing normal state derivatives. A finite-dimensional solution will generally violate these requirements pointwise, so that we enforce the BCs weakly. This enforcement involves transforming the state and gradient, similar to methods in previous works [15], though we construct a state/gradient on the boundary instead of employing a ghost cell. Starting with the state, we require that, at a symmetry boundary, all vectors in the state (e.g., a velocity) have their normal components zeroed out. This results in a linear transformation from the interior $\boldsymbol{u}^+$ to the boundary $\boldsymbol{u}^b$ state vector, which reads $\boldsymbol{u}^b = \boldsymbol{A}\boldsymbol{u}^+$. $\boldsymbol{A}$ is the identity matrix for all states except the momentum, which transforms as $(\rho\boldsymbol{v})^b = \underline{V}(\rho\boldsymbol{v})^+$, where $\underline{V} = \underline{I} - \boldsymbol{n} \otimes \boldsymbol{n} = \delta_{ij} - n_i n_j$; $\boldsymbol{n}$ is the outward-pointing normal, and $\underline{I} = \delta_{ij}$ is the $\dim \times \dim$ identity matrix.

The state gradient transformation must account for possibly nonzero normal velocity components. We first consider a hypothetical ghost state $\boldsymbol{u}^-$ and gradient $\nabla\boldsymbol{u}^-$, obtained by reflecting the velocity about the symmetry line. Specifically, $\boldsymbol{u}^- = \boldsymbol{B}\boldsymbol{u}^+$, where $\boldsymbol{B}$ is an identity matrix for all states except the momentum, which transforms as $(\rho\boldsymbol{v})^- = \underline{W}(\rho\boldsymbol{v})^+$, where $\underline{W} = \underline{I} - 2\boldsymbol{n} \otimes \boldsymbol{n} = \delta_{ij} - 2n_i n_j$. Note that $\boldsymbol{B} = 2\boldsymbol{A} - \boldsymbol{I}$ and that $\underline{W} = 2\underline{V} - \underline{I}$. Differentiating the expression for $\boldsymbol{u}^-$ in space gives the gradient, which we must reflect by applying $\underline{W}$, so that $\nabla\boldsymbol{u}^- = \boldsymbol{B}\nabla\boldsymbol{u}^+\underline{W}^T$. Finally, we obtain the gradient at the boundary $\nabla\boldsymbol{u}^b$ by averaging the interior and exterior gradients; this is consistent with what would happen in the viscous flux calculation if there were actually a symmetrical mesh on the other side of the symmetry line. Therefore, we have

$$\begin{aligned} \nabla\boldsymbol{u}^b &= \frac{1}{2}(\nabla\boldsymbol{u}^+ + \nabla\boldsymbol{u}^-) = \frac{1}{2}(\nabla\boldsymbol{u}^+ + \boldsymbol{B}\nabla\boldsymbol{u}^+\underline{W}^T) \\ &= \frac{1}{2}(\nabla\boldsymbol{u}^+ + (2\boldsymbol{A} - \boldsymbol{I})\nabla\boldsymbol{u}^+(2\underline{V}^T - \underline{I})) \\ &= \nabla\boldsymbol{u}^+ + \boldsymbol{A}\nabla\boldsymbol{u}^+(2\underline{V}^T - \underline{I}) - \nabla\boldsymbol{u}^+\underline{V} \\ &= \nabla\boldsymbol{u}^+\boldsymbol{n} \otimes \boldsymbol{n} + \boldsymbol{A}\nabla\boldsymbol{u}^+(\underline{I} - 2\boldsymbol{n} \otimes \boldsymbol{n}) \end{aligned}$$

[‡]This order may change from element to element in p refinement.

C. Scaling of $\tilde{\nu}$

The SA working variable $\tilde{\nu}$ will generally be orders of magnitude smaller than the other state components. We use scaling or "nondimensionalization" of $\tilde{\nu}$ to make its range of numerical values similar to the other state components. This proved to be effective in improving the performance of the linear and nonlinear solvers [11]. We store the scaled quantity $\rho\tilde{\nu}'$ given by

$$\rho\tilde{\nu}' = \frac{\rho\tilde{\nu}}{\kappa_{\mathrm{SA}}\mu_\infty}$$

where κ_{SA} is a scaling factor, typically $\mathcal{O}(\sqrt{Re})$, and μ_∞ is the freestream laminar dynamic viscosity. In addition, the SA $\tilde{\nu}$ equation is divided by $\kappa_{\mathrm{SA}}\mu_\infty$.

D. Implicit Solver

The system of nonlinear equations that forms the primal problem is solved using Newton's method with pseudotransient continuation [16] for improved robustness. Two versions of the solver are considered: 1) a relatively aggressive Courant–Friedrichs–Lewy (CFL) number evolution strategy in which the CFL grows by a factor of 2 after each full Newton update, in combination with incomplete Newton updates when certain physical quantities (e.g., density and pressure) change too drastically; and 2) a more moderate CFL evolution strategy in which the increase factor is 1.2, but in which the physical quantity changes are not limited, but rather the line search prevents the residual from growing too quickly and no update is taken if physical constraints are violated. Both strategies are found to perform similarly for the present test cases. The linear systems at each Newton iteration are solved with an element-line preconditioned [17] GMRES solver.

IV. Output Error Estimation

Choosing a basis for the test space in Eq. (6) gives a discrete system of nonlinear equations

$$\boldsymbol{R}(\boldsymbol{U}) = 0 \tag{7}$$

where $\boldsymbol{U}$ and $\boldsymbol{R}$, both in $\mathbb{R}^N$ are, respectively, the state and residual vectors. For a scalar output $J(\boldsymbol{U})$, we define the discrete adjoint vector $\boldsymbol{\Psi} \in \mathbb{R}^N$ as the sensitivity of J to perturbations in $\boldsymbol{R}$ [5]. The adjoint satisfies the following linear equation:

$$\left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\right)^T \boldsymbol{\Psi} + \left(\frac{\partial J}{\partial \boldsymbol{U}}\right)^T = 0 \tag{8}$$

We use the adjoint to estimate the error in an output when computing on a finite-dimensional approximation space. Without access to infinite resolution, estimating the true numerical error in an output is practically out of reach for general nonlinear problems. We thus restrict ourselves to estimating the output error between two finite-dimensional spaces: a coarse approximation space $\boldsymbol{\mathcal{V}}_H$ on which we calculate the state and output, and a fine space $\boldsymbol{\mathcal{V}}_h$ (obtained by incrementing the approximation order by one on the same mesh) relative to which we estimate the error. We would like to measure the output error in the coarse solution relative to the fine space:

Output error:

$$\delta J \equiv J_H(\boldsymbol{U}_H) - J_h(\boldsymbol{U}_h) \tag{9}$$

We assume that the fine approximation space contains the coarse approximation space, so that the following lossless state injection is possible, $\boldsymbol{U}_h^H \equiv \boldsymbol{I}_h^H \boldsymbol{U}_H$, where $\boldsymbol{I}_h^H$ is the coarse-to-fine state injection (prolongation) operator. On the fine space, the exact solution $\boldsymbol{U}_h \in \mathbb{R}^{N_h}$ would give us zero fine-space residuals $\boldsymbol{R}_h(\boldsymbol{U}_h) = 0$. However, the state injected from the coarse space will generally not be a fine-space solution and hence will not give us zero fine-space residuals $\boldsymbol{R}_h(\boldsymbol{U}_h^H) \neq 0$. Instead, the injected coarse state solves a perturbed fine-space problem $\boldsymbol{R}_h(\boldsymbol{U}_h') - \boldsymbol{R}_h(\boldsymbol{U}_h^H) = 0$. Because this is just the fine-space problem with a residual perturbation, the fine-space adjoint Y_h tells us to expect an output perturbation given by the inner product between the adjoint and the residual perturbation:

$$\delta J \approx -\boldsymbol{\Psi}_h^T \boldsymbol{R}_h(\boldsymbol{U}_h^H) \tag{10}$$

This derivation assumes small perturbations in the state when the output or equations are nonlinear. Note that this error estimate does not require the fine-space primal solution $\boldsymbol{U}_h$. However, it requires the solution of Eq. (8) with residual and output linearizations about $\boldsymbol{U}_h^H$. In this work, we fully converge the fine-space adjoint, storing the fine-space Jacobian and using $\boldsymbol{\Psi}_h^H \equiv \boldsymbol{I}_h^H \boldsymbol{\Psi}_H$, as an initial guess in the GMRES iterative solver for $\boldsymbol{\Psi}_h$. This does not add a dominant contribution to the total cost, which for these cases is dominated by dozens of Newton–Raphson iterations of the primal problem. The fine-space adjoint system, though larger, is linear and for these runs remains less expensive than the coarse primal. For larger simulations, techniques such as iterative smoothing or reconstruction can be used to approximate the fine-space adjoint [5,9,11].

V. Mesh Adaptation

The adjoint-weighted residual error estimate in Eq. (10) can be localized to the elements by keeping track of the contributions from each fine-space element, indexed by k:

$$J_H(\boldsymbol{U}_H) - J_h(\boldsymbol{U}_h) \approx -\boldsymbol{\Psi}_h^T \boldsymbol{R}_h(\boldsymbol{U}_h^H) = -\sum_k \boldsymbol{\Psi}_{hk}^T \boldsymbol{R}_{hk}(\boldsymbol{U}_h^H)$$

$$\Rightarrow \epsilon_k \equiv |\boldsymbol{\Psi}_{hk}^T \boldsymbol{R}_{hk}(\boldsymbol{U}_h^H)|$$

where the subscript k indicates restriction to element k, and the adaptive indicator ϵ_k is obtained by taking the absolute value of the elemental contributions. This indicator then drives mesh adaptation, the goal of which is to reduce the output error. We consider two adaptation strategies, as outlined next.

A. Hanging-Node Quadrilateral Refinement

The first adaptation strategy used in this work is hanging-node refinement of an initially structured quadrilateral mesh [9,11,18]. In this strategy, a fixed fraction f^{frac} of elements with the highest error indicators is flagged for refinement. For the present results, we only consider isotropic refinement in which each quadrilateral is subdivided uniformly into four quadrilaterals, as illustrated in Fig. 1. This refinement is done in each element's reference space by employing the reference-to-global coordinate mapping in calculating the refined elements' geometry node coordinates. The refined elements inherit the same geometry approximation order and quadrature rules as the parent coarse element. When curved boundary representations are employed, new nodes introduced on the boundary are snapped to the geometry; this perturbation is usually very small for high-order curved elements on the boundary, because these already approximate the geometry with high fidelity. For the cases involving curved geometry in this paper, we use quartic polynomials as element edges and the element's interior mapping is generated via a tensor product between those polynomials. At the boundaries, the quartic polynomials interpolate the true mathematical description of the geometry.

Elements created in a hanging-node refinement can be marked for h refinement again in subsequent adaptation iterations. In this case, neighbors are divided in the minimal possible fashion, generally anisotropically, to keep one level of refinement difference between adjacent cells, as illustrated in Fig. 1.

B. Metric-Based Unstructured Remeshing

The second adaptation strategy used in this work is an unstructured metric-based remeshing approach on triangles. The idea in this strategy is to create, at every adaptation iteration, a new mesh using the current mesh and the error indicator. During remeshing, the current mesh serves as a background mesh on which the desired metric field is prescribed. The program used for the remeshing is the bidimensional anisotropic mesh generator [19]. This program

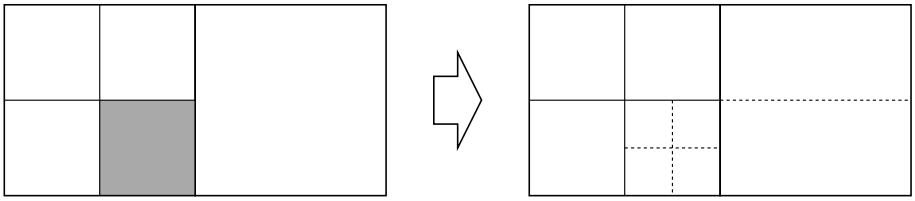


Fig. 1 Rule for hanging-node adaptation for quadrilaterals.

generates straight-edged meshes, which are then curved via a linear elasticity analogy to represent the curved geometry [20].

The first step is a calculation of the current metric field, computed as the grid-implied metric on each element k:

$$\underline{M}_k^c = (\underline{J}_k \underline{J}_k^T)^{-1} \tag{11}$$

where $\underline{J}_k \in \mathbb{R}^{\dim \times \dim}$ is the Jacobian of the geometric mapping of the reference element to element k. The eigenvalues of the metric are the inverse squares of the principal stretching magnitudes, and the eigenvectors are the principal stretching directions. For curved elements, $\underline{J}_k$ is evaluated at the centroid of the element. We now describe two methods used for determining the requested metric field based on the error estimates: one based on a priori estimates and one based on sampling.

1. Fixed Growth Using a Priori Estimates

This method for constructing the requested metric field is similar to that presented in previous work [21]. We assume a fixed-growth refinement strategy in which the number of elements desired on the refined mesh is $N^f = f^{\text{growth}} N^c$, where $f^{\text{growth}} > 1$ is the growth fraction and N^c is the current number of elements. We relate the growth in elements to an error reduction factor through an a priori estimate. In particular, we sum the error indicators on the current mesh to obtain a conservative estimate of the current global error:

$$e^c = \sum_k \epsilon_k^c \tag{12}$$

Assuming that, with adaptive refinement, the global error decreases at a rate of r (with $h \propto N^{-1/\dim}$), we calculate the global error estimate on the refined mesh as

$$e^f = e^c \left(\frac{N^c}{N^f}\right)^{r/\dim} \tag{13}$$

We would like the error to be equidistributed on the fine mesh, which means that every fine-space element should have an error of e^f/N^f. We now apply the a priori estimate to each element, and we assume that for anisotropic elements the error depends on the shortest principal length h_1. The resulting a priori relationship is

$$n_k \frac{e^f}{N^f} = \epsilon_k^c \left(\frac{h_1^f}{h_1^c}\right)^{r_k} \tag{14}$$

where n_k is the number (not necessarily an integer) of refined elements per current element k, h_1^f/h_1^c are the shortest principal lengths on the refined/current meshes, and r_k is a possibly element-specific error convergence rate. We estimate the number of fine elements for coarse element k as

$$n_k = \frac{h_1^c}{h_1^f}\frac{h_2^c}{h_2^f} = \left(\frac{h_1^c}{h_1^f}\right)^2 \frac{\mathcal{AR}_k^f}{\mathcal{AR}_k^c}, \tag{15}$$

where $\mathcal{AR}^f/\mathcal{AR}^c$ are the desired/current aspect ratios on element k. Substituting Eq. (15) into Eq. (14), we obtain an expression for the scaling of the shortest principal stretching length:

$$\frac{h_1^f}{h_1^c} = \left[\frac{e^f}{N_f}\frac{1}{\epsilon_k^c}\frac{\mathcal{AR}^c}{\mathcal{AR}^f}\right]^{1/(r_k+\dim)}$$

The desired aspect ratio on each element is calculated heuristically from the Hessian (second derivative matrix) of the Mach number scalar field [22,23]. Although only strictly applicable to linear approximations, we have found that the directions obtained from the Hessian often correlate reasonably well with directions obtained from approaches that use higher-order derivatives [21].

For the a priori convergence rates in the present work, we use $r = r_k = p + 1$, where p is the solution approximation order. An exception is "outlier" elements: those whose error indicator ϵ_k is larger than five standard deviations from the mean error. On these elements, we assume a first-order rate $r_k = 1$.

2. Target-Cost Optimization Through Sampling

An alternate method that does not rely on a priori error estimates is the sampling approach introduced by Yano [10]. Briefly, this method constructs models for the error indicator and cost function (degrees of freedom) on each element as a function of the metric step change tensor. This construction proceeds via a regression over errors and cost functions computed by sampling canonical subdivisions of the element. An iterative optimization approach is then employed to equidistribute the ratio of the marginal error to marginal cost over the elements, at a fixed target cost. Multiple adaptive iterations at a single cost target then ensure the construction of the optimal mesh at that cost.

In the present work, we slightly modify the error sampling approach presented by Yano [10] to use projections of the fine-space adjoint to the sampled subdivisions, to avoid having to solve local problems that can be somewhat cumbersome. We test the modified optimization approach for several cases and compare it to the hanging-node refinement and a priori driven remeshing.

VI. Results

A. Flat Plate, $Re_L = 5 \times 10^6$, ($L = 1$), $M = 0.2$

The first case we consider is turbulent flow over a flat plate. The geometry and boundary conditions for this case are set according to NASA's turbulence modeling resource website (Fig. 2a). The inflow and far-field values for the eddy viscosity are set to 3 times the value of the laminar viscosity, because this value corresponds to a fully turbulent simulation. The laminar viscosity is calculated using Sutherland's law (2) with $T_s = 110{,}000$ and $T_{\text{ref}} = 300{,}000$. Figure 2b shows the initial mesh used for this case.

We initialize the flow with uniform conditions at $M = 0.2$ and solve the discretized equations with $p = 2$ to a residual tolerance of 10^{-8}. The output of interest is the total drag on the plate, and we consider both hanging-node adaptation and unstructured remeshing using metric optimization. At each step of the hanging-node adaptation, we select $f^{\text{frac}} = 10\%$ of the elements with the largest error indicators and refine them isotropically. For the metric optimization, we consider four degree-of-freedom (DOF) target values: 2000, 4000, 8000, and 16,000. We present a code-to-code comparison in Fig. 3a of our simulations to the results from CFL3D and FUN3D, provided by NASA's turbulence modeling group. We see that our results converge to both CFL3D and FUN3D (roughly; the FUN3D results do not yet appear completely converged) results, and that the unstructured metric-based optimization yields faster convergence

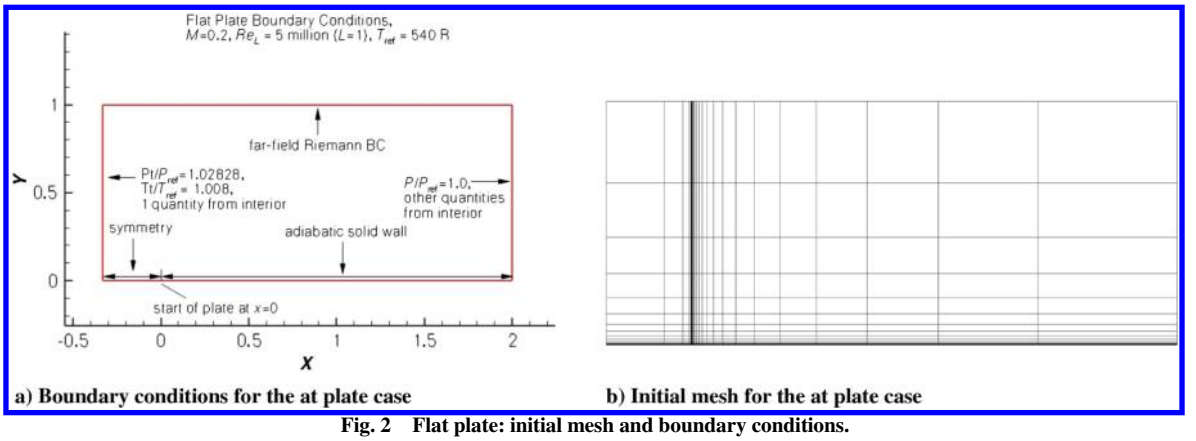


Fig. 2 Flat plate: initial mesh and boundary conditions.

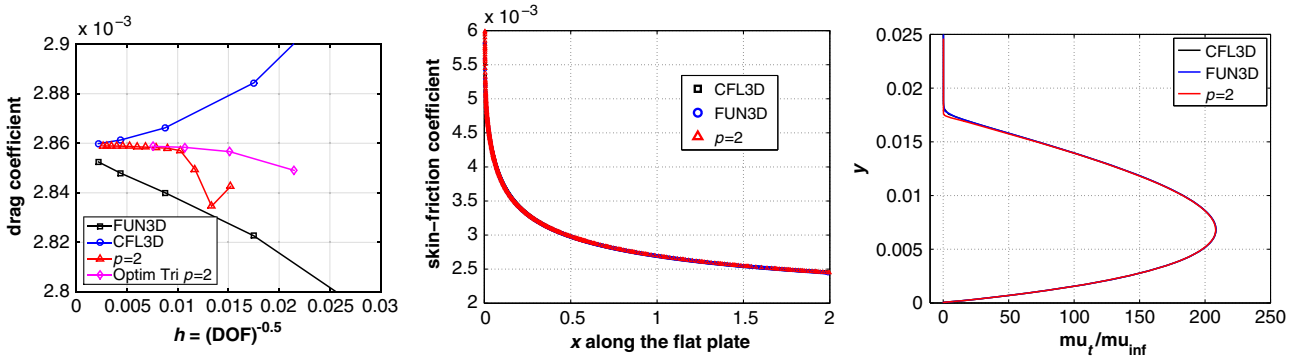


Fig. 3 Flat plate: drag convergence and comparison of skin-friction coefficient and turbulent viscosity distributions (at $x = 0.97$ on the flat plate).

compared with hanging-node refinement. This is expected because the hanging-node refinement is constrained by the optimality of the starting mesh. The skin-friction results in Fig. 3b show very good agreement between all results. We also observe very good agreement in the turbulent viscosity distribution (Fig. 3c), with just a small discrepancy at the outer edge of the boundary layer.

Figure 4 shows the final hanging-node adapted mesh for this case. Note that the adaptive procedure targets the outer edge of the turbulent boundary layer where there is a rapid variation of eddy viscosity. The component of the drag adjoint correspondent to the SA variable shows large negative values at the leading edge of the turbulent boundary layer (Fig. 4b).

Finally, Fig. 5 shows the final metric-optimized meshes. Compared with the hanging-node meshes, the refinement pattern is similar: the near-wall region and the edge of the boundary layer, as indicated by the eddy viscosity variable, are targeted for refinement. However, the resulting meshes are more efficient because they are not limited to the suboptimality of the structured starting mesh in the hanging-node refinement case.

B. Smooth Bump, $Re = 3 \times 10^6$, $M = 0.2$

This is another verification case from the NASA turbulence modeling resource group. Reynolds number $Re_L = 3 \times 10^6$ ($L = 1$) flow is simulated in a channel with a bump on the bottom wall. Symmetry boundary conditions are used for the top and bottom of the channel, with the exception of $x \in [0, 1.5]$ on the bottom boundary, where an adiabatic wall boundary condition is applied. A static

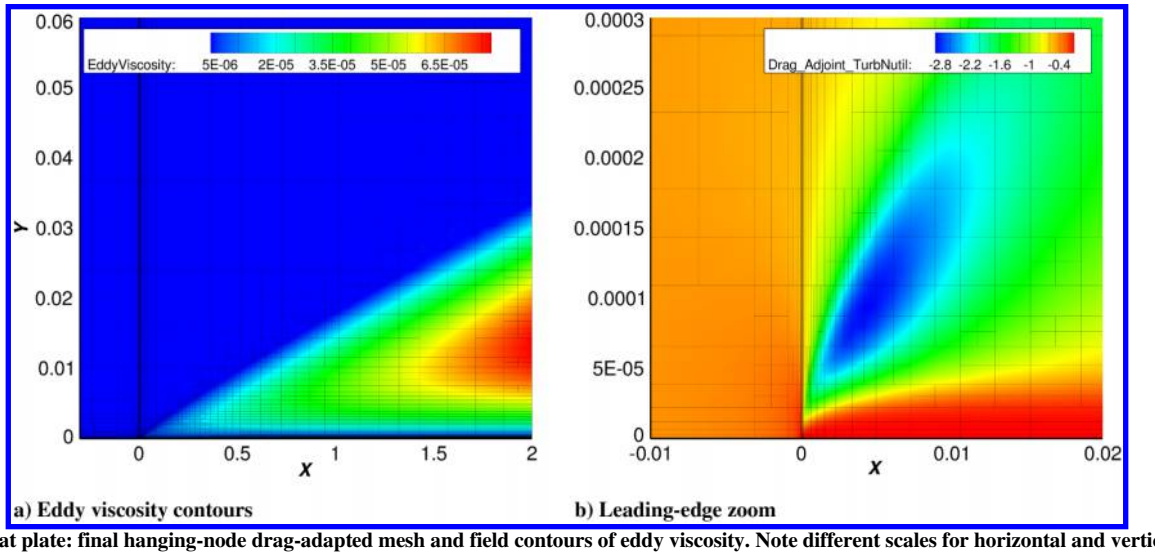
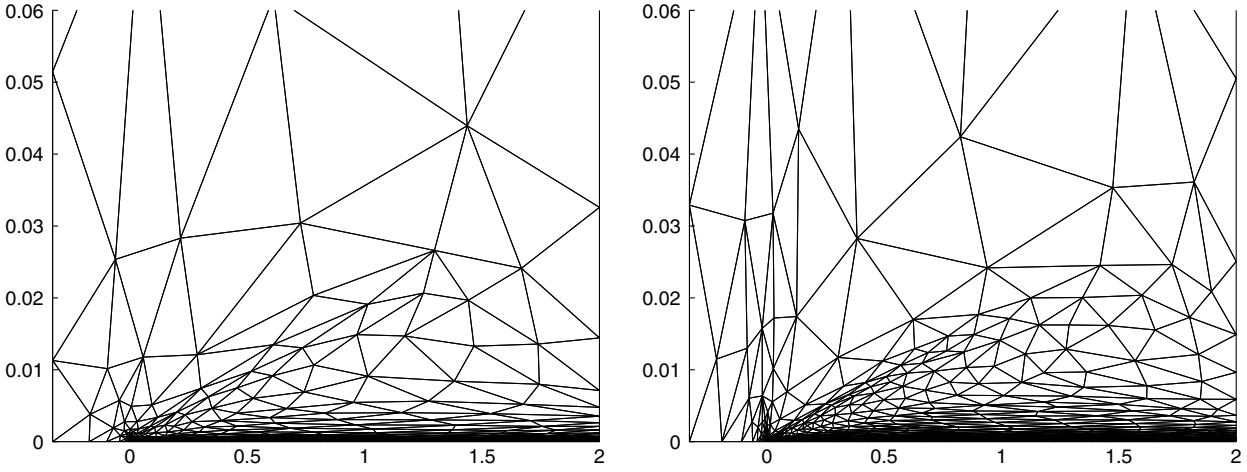


Fig. 4 Flat plate: final hanging-node drag-adapted mesh and field contours of eddy viscosity. Note different scales for horizontal and vertical axes.



a) DOF=8000 b) DOF=16,000

Fig. 5 Flat plate: final metric-optimized drag-adapted meshes at two target DOF values. Note different scales for horizontal and vertical axes.

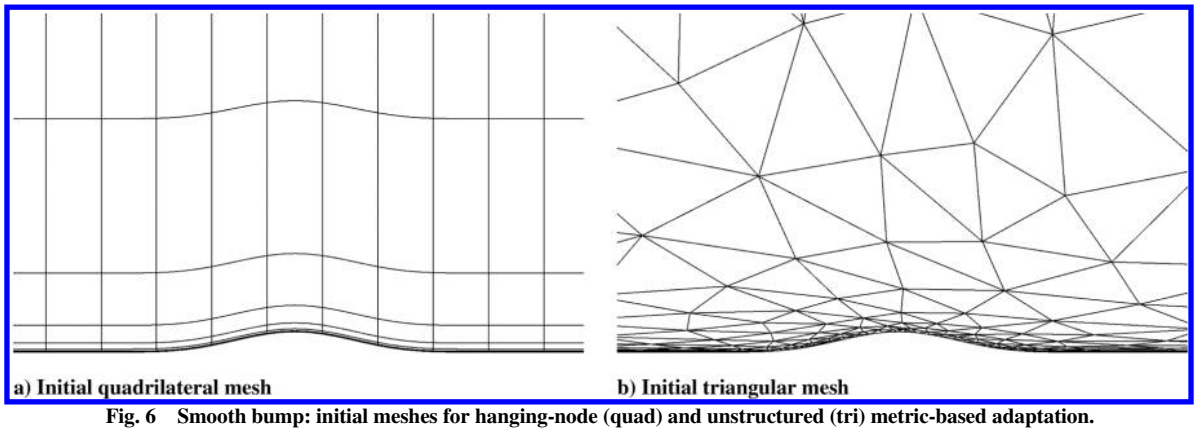


Fig. 6 Smooth bump: initial meshes for hanging-node (quad) and unstructured (tri) metric-based adaptation.

pressure is imposed at the right (outflow) boundary, and total temperature, total pressure, and angle of attack (zero) are prescribed at the inflow. Total/stagnation quantities are computed using a Mach number of $M = 0.2$. The dynamic viscosity is computed using Sutherland's law (2), with $T_s = 110{,}000$ and $T_{\text{ref}} = 300{,}000$. The inflow turbulence eddy viscosity μ_t is set to 3 times the laminar viscosity.

Initial structured and unstructured meshes for adaptation are shown in Fig. 6. Adjoint-based adaptive runs are performed from these meshes using the drag force as the target output and adaptive factors of $f^{\text{frac}} = 0.07$ and $f^{\text{growth}} = 1.3$. Figure 7 shows field plots of the wall distance function and Mach number on one of the unstructured adapted meshes. Note the heavy refinement in the boundary layer, an area to which the drag output is highly sensitive.

As a code-to-code verification of the turbulence model, Fig. 8 shows the pressure and skin-friction distributions compared with those of two other codes, CFL3D and FUN3D. Data for these codes were obtained from the NASA turbulence modeling resource group. The agreement in both of these quantities is very good: note that flow singularities at the leading ($x = 0$) and trailing ($x = 1.5$) edges of the bump cause oscillations there.

Figure 9 shows the convergence of the drag and lift coefficients with adaptive mesh refinement at $p = 2$ solution approximation. In these plots, the degrees of freedom are measured as DOF $= N_e n(p)$, where N_e is the number of elements and $n(p)$ is the number of unknowns per element: $n(p) = (p+1)^2$ for tensor-product approximation and $n(p) = (p+1)(p+2)/2$ for full-order approximation.

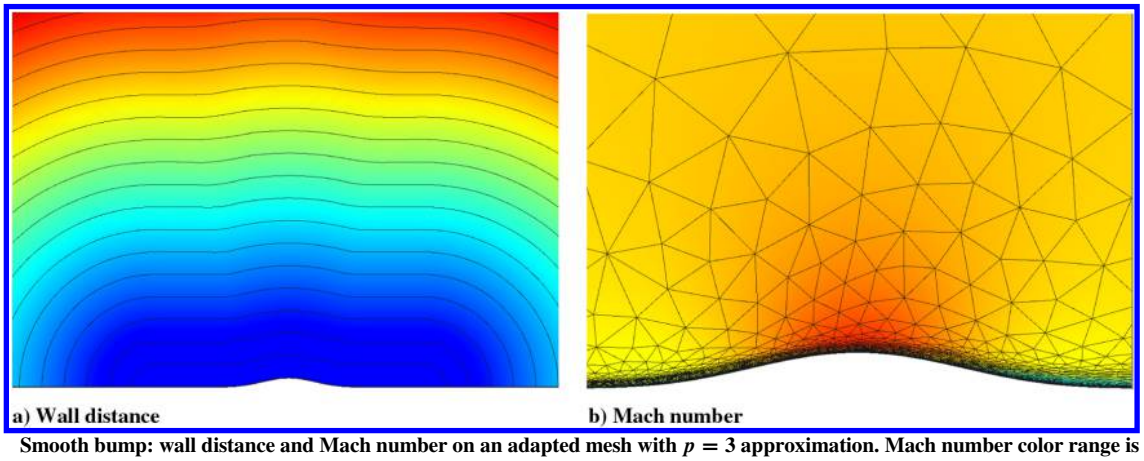
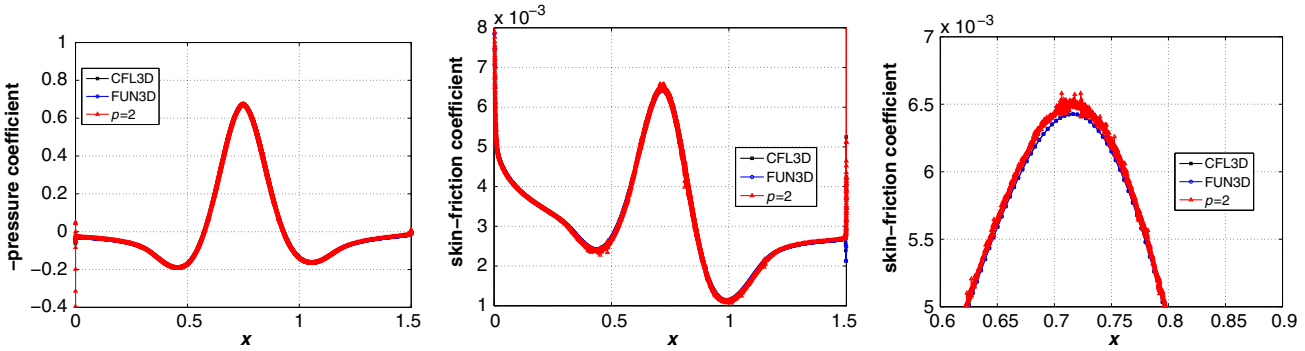
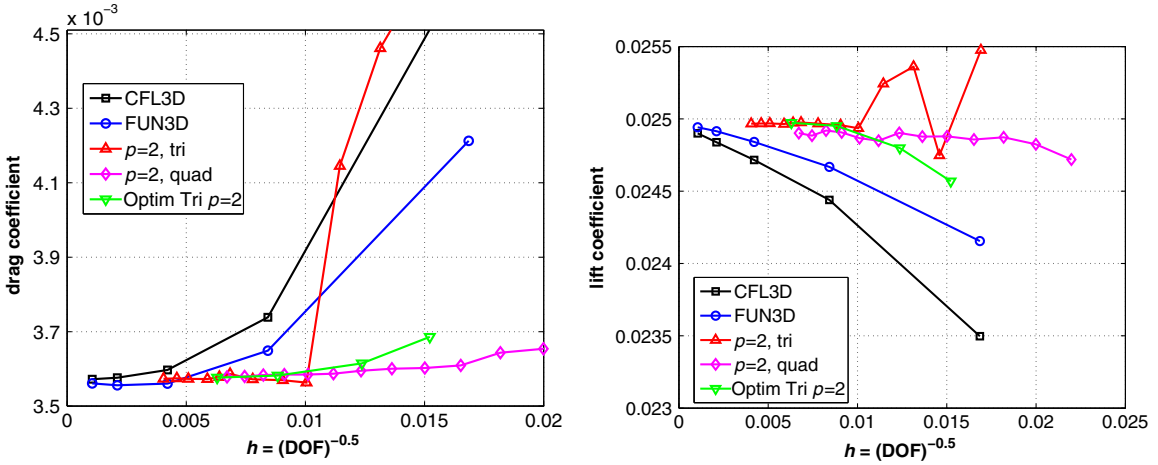


Fig. 7 Smooth bump: wall distance and Mach number on an adapted mesh with $p = 3$ approximation. Mach number color range is 0–0.3.



a) Pressure coefficient b) Skin-friction coefficient c) Skin-friction zoom

Fig. 8 Smooth bump: pressure and skin-friction coefficients for the $p = 2$ drag-adapted triangular mesh.



a) Drag coefficient b) Lift coefficient

Fig. 9 Smooth bump: drag and lift coefficient convergence comparisons for drag adaptation with $p = 2$, using unstructured (tri) and hanging-node (quad) meshes.

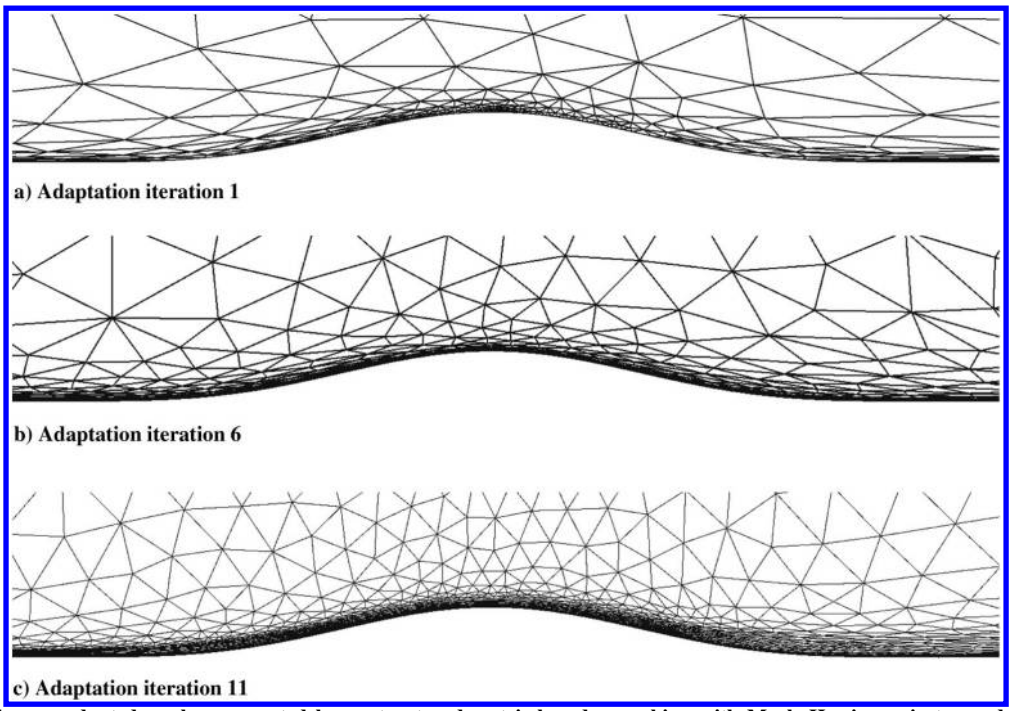


Fig. 10 Smooth bump: adapted meshes generated by unstructured metric-based remeshing with Mach–Hessian anisotropy detection and $p = 2$ approximation.

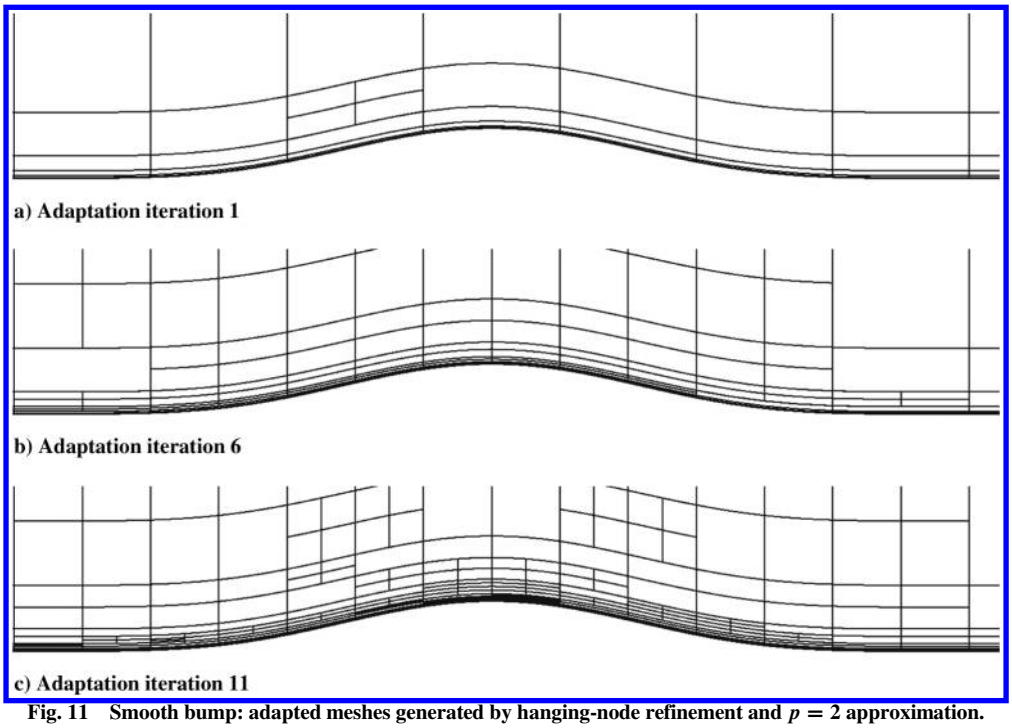


Fig. 11 Smooth bump: adapted meshes generated by hanging-node refinement and $p = 2$ approximation.

Note the rapid convergence of the drag and lift coefficients to their nearly asymptotic values, relative to the second-order codes. The unstructured Mach–Hessian adaptive results show larger errors on the initial meshes because these are relatively underresolved in the critical boundary-layer region. However, after a few adaptive iterations, the drag and lift "snap" to their asymptotic values. In addition, the optimized triangular meshes yield smoother convergence and lower errors compared with the Mach–Hessian results.

Figures 10–12 show selected meshes in the adaptive refinement sequences. As expected, the adaptive refinement targets the boundary-layer region, where anisotropic elements are possible, whereas most of the remainder of the flow is approximated with isotropic elements.

C. NACA 0012, $Re = 6 \times 10^6$, $M = 0.15$

In this case, we consider a NACA 0012 airfoil in $Re = 6 \times 10^6$, $M = 0.15$ flow. The dynamic viscosity is computed using Sutherland's law (2), with $T_s = 110{,}000$ and $T_{\text{ref}} = 300{,}000$. The inflow turbulence eddy viscosity μ_t is set to 3 times the laminar viscosity. Freestream boundary conditions are imposed at a far field that is over 15,500 chords away from the airfoil in each direction. We do not use a point vortex to correct the boundary condition.

This case was run adaptively at $p = 2$ using hanging-node refinement of a structured initial mesh, with drag as the target output and a fixed refinement fraction of $f^{\text{frac}} = 0.07$. Figure 13 shows the initial mesh and adapted results for $\alpha = 10$ deg. The regions targeted for

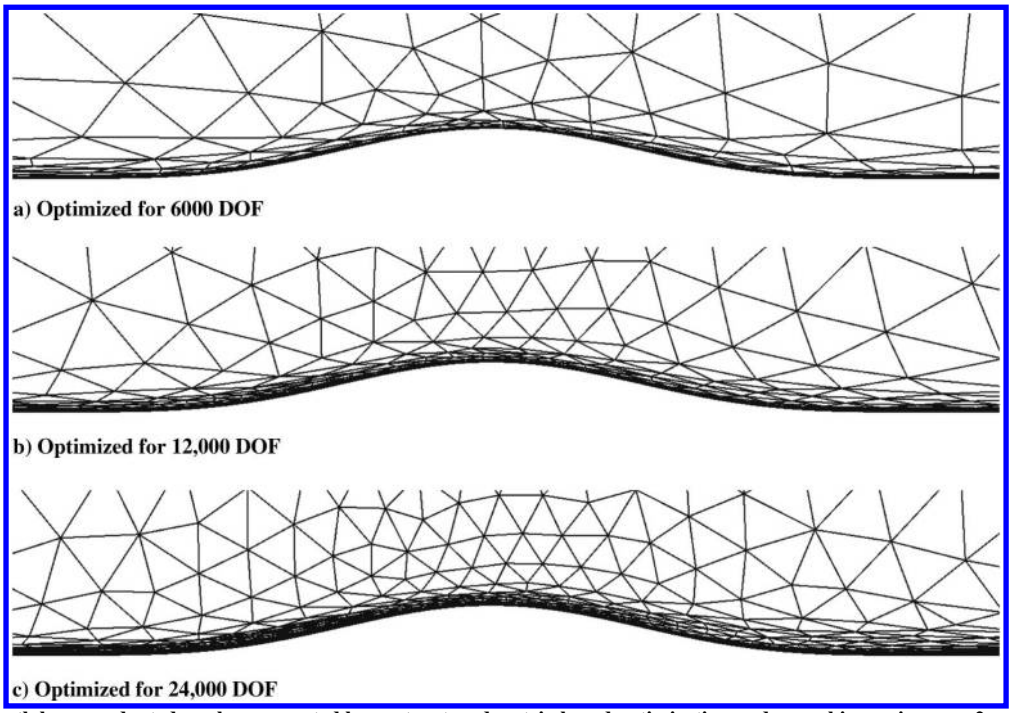


Fig. 12 Smooth bump: adapted meshes generated by unstructured metric-based optimization and remeshing, using $p = 2$ approximation.

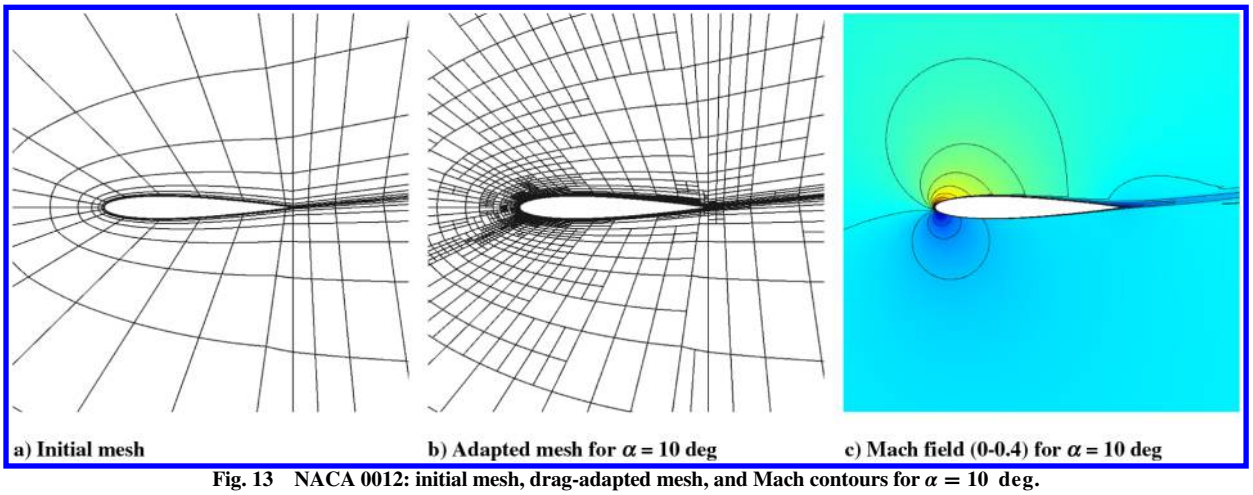


Fig. 13 NACA 0012: initial mesh, drag-adapted mesh, and Mach contours for $\alpha = 10$ deg.

refinement include the boundary layer, wake, and leading-edge stagnation streamline, where errors can have a large effect on the drag output.

Figure 14 shows a comparison of pressure coefficient and skin-friction distributions for $\alpha = 0$ and 10 deg. The comparison is made against data from CFL3D with a far field at $500c$ and a lift-based point vortex correction, and the results are in excellent agreement: the curves are virtually on top of each other. Figure 15 shows the lift coefficient versus angle of attack and drag polar for the adapted results. Again, excellent agreement with CFL3D data is observed.

Finally, Fig. 16 shows the convergence of the lift and drag coefficients with adaptive refinement for two runs: $\alpha = 0$ and 10 deg. We see that drag converges faster than lift, which makes sense because we are adapting on the drag outputs. In addition, convergence slows with increasing angle of attack, likely because the flowfield becomes more complex (e.g., the boundary layer on the upper surface becomes thicker and requires more resolution). In all cases, for the last several adaptive iterations, the outputs show little variation.

D. NACA 0012, $Re = 6 \times 10^6$, $M = 0.15$, $\alpha = 10$ deg

In this case, we consider a slight variant of the previous test case for the purpose of verification with detailed data made available by the NASA turbulence modeling resource group. The airfoil is still a

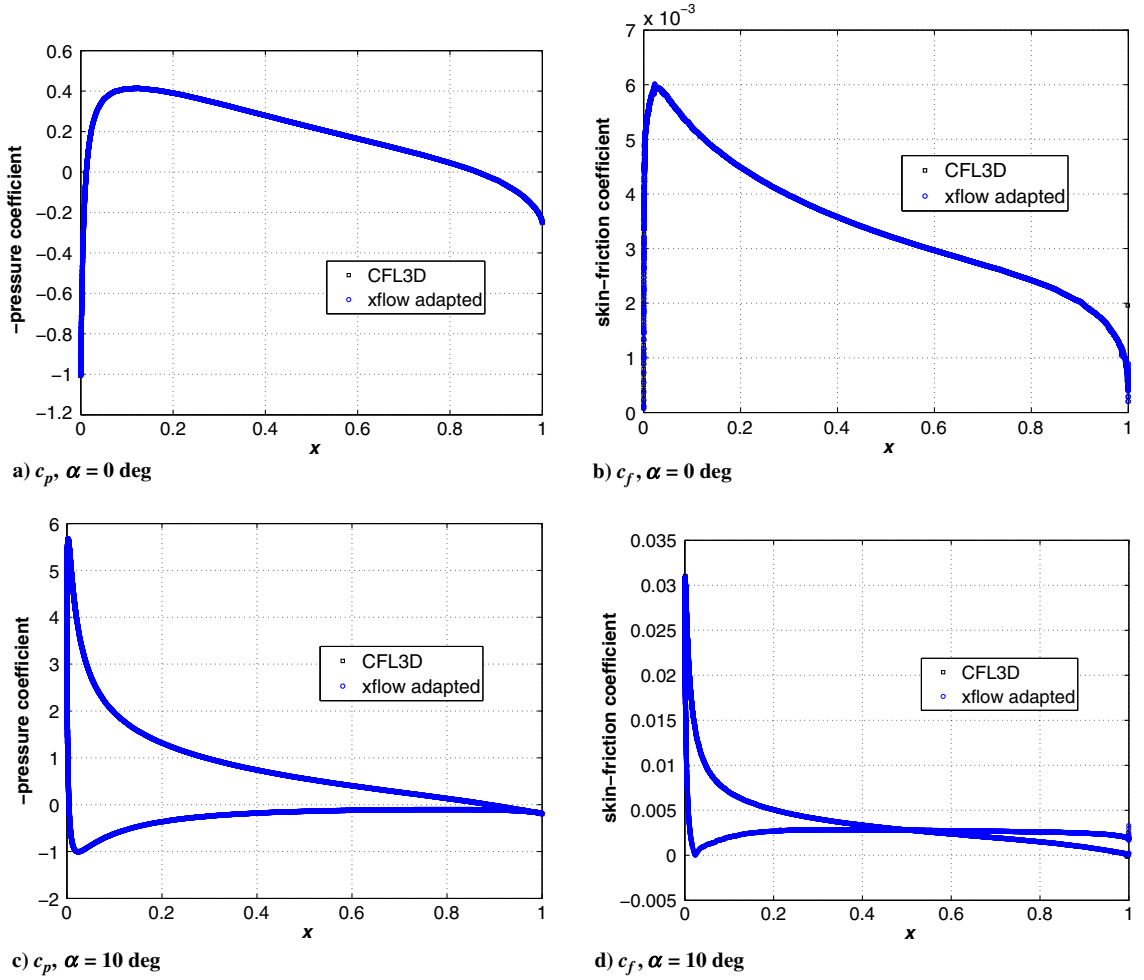
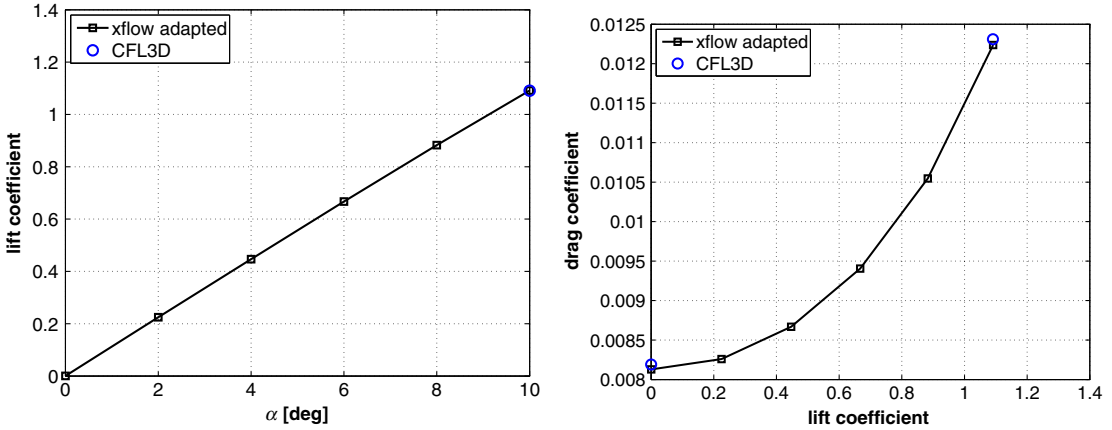
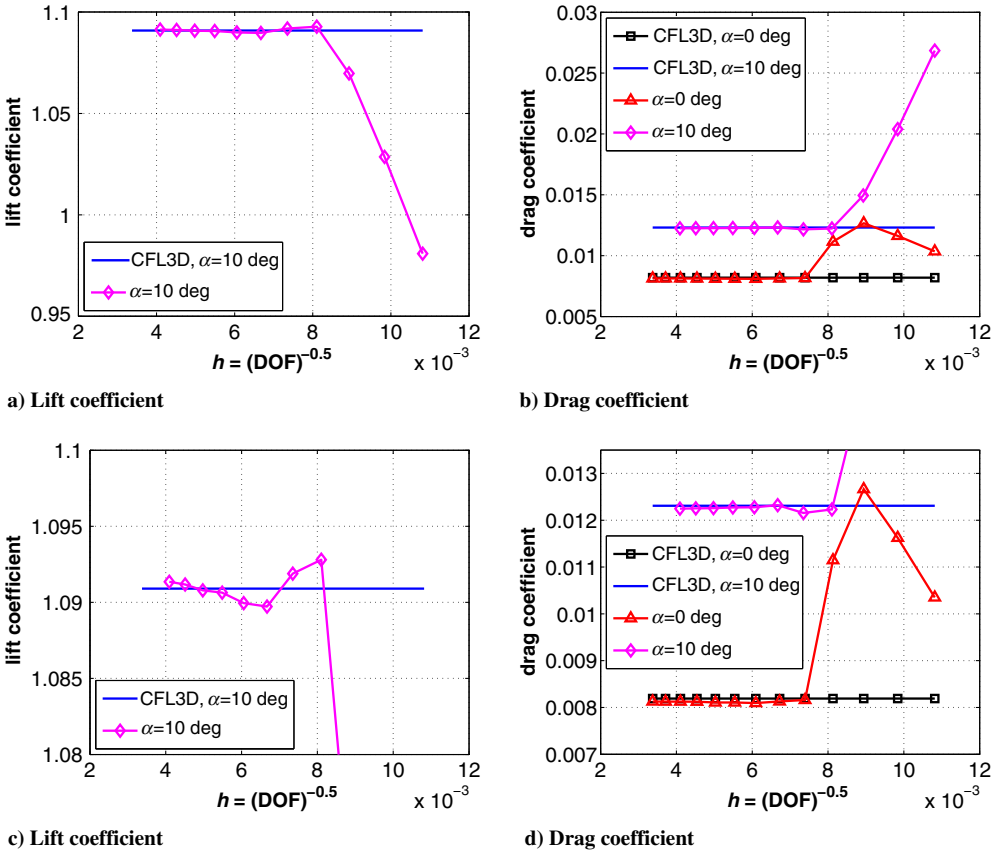


Fig. 14 NACA 0012: pressure and skin-friction coefficient distributions for $\alpha = 0$, 10 deg, comparing the final adapted mesh result with data from CFL3D.



a) c_l versus α b) c_d versus c_l

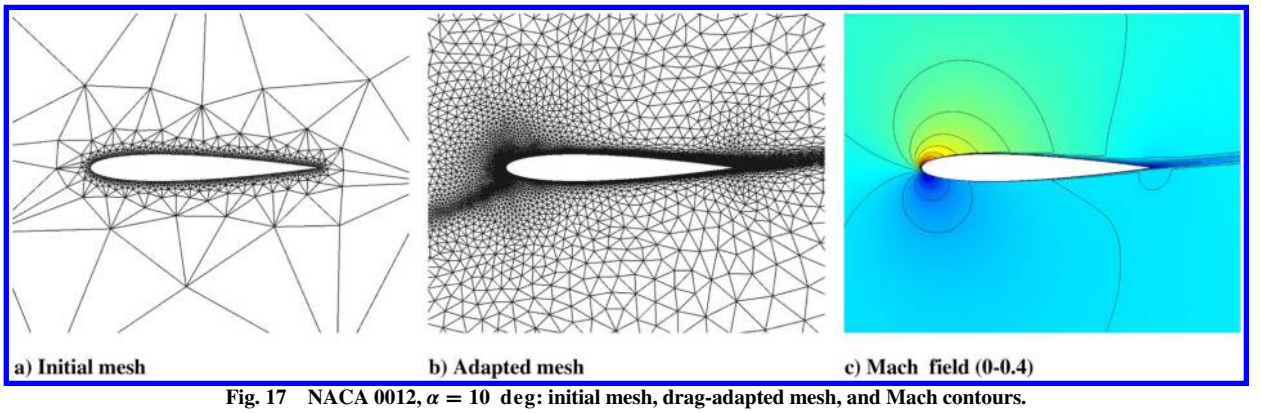
Fig. 15 NACA 0012: lift coefficient versus angle of attack and drag polar, with comparison to CFL3D data.



a) Lift coefficient b) Drag coefficient

c) Lift coefficient d) Drag coefficient

Fig. 16 NACA 0012: convergence of lift and drag coefficients with hanging-node adaptation and $p = 2$.



a) Initial mesh b) Adapted mesh c) Mach field (0-0.4)

Fig. 17 NACA 0012, $\alpha = 10$ deg: initial mesh, drag-adapted mesh, and Mach contours.

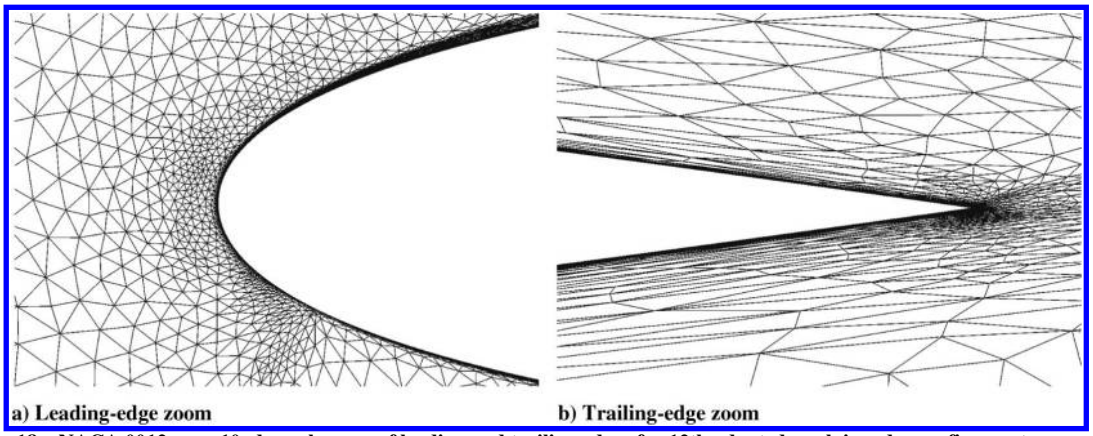


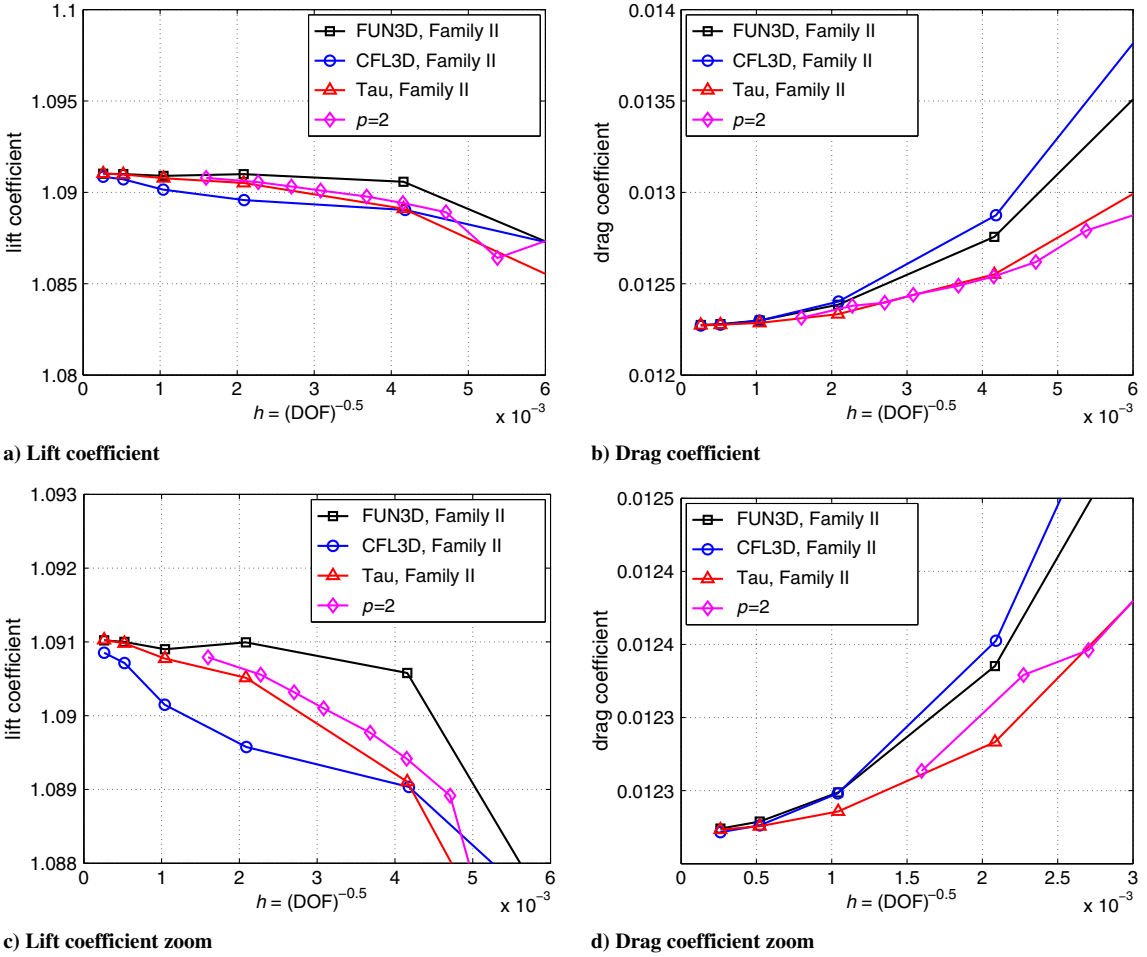
Fig. 18 NACA 0012, $\alpha = 10$ deg: close-up of leading and trailing edges for 12th adapted mesh in a drag refinement sequence.

NACA 0012, with a closed trailing edge as prescribed on the NASA website. The far field is approximately 500 chords away from the airfoil, but the far-field geometry is constructed to be consistent with the far-field geometry of the grids provided on the NASA website. No vortex correction is employed on the far field. The website also provides detailed conditions and setup information on the case.

This case was run adaptively at $p = 2$ using metric-based triangular refinement of a relatively coarse initial mesh. Drag is chosen as the target output and a fixed growth fraction of $f^{\text{growth}} = 1.3$ is used. Figure 17 shows the initial mesh and adapted results. The regions targeted for refinement include the boundary layer, wake, and leading-edge stagnation streamline. Figure 18 shows a closeup of the leading- and trailing-edge regions for the 12th adapted mesh.

Figure 19 shows the convergence of the lift and drag coefficients with adaptive refinement. We see that both coefficients agree well with the provided data obtained from the CFL3D, FUN3D, and TAU codes. The adapted values still show variation on the finest grids, and this variation could be due to insufficient resolution (i.e., more adaptations needed) or to an inadequate measure of anisotropy (currently based on the Mach–Hessian) during mesh optimization. Future work will investigate the precise cause and possible mesh efficiency improvements.

Finally, Fig. 20 shows the pressure coefficient off the bottom surface of the airfoil trailing edge at $x = 0.999c$. This pressure coefficient was obtained from a $p = 3$ run on the finest adapted $p = 2$ mesh. Data from the FUN3D on a sequence of grids are overlaid. As



a) Lift coefficient **b) Drag coefficient**

c) Lift coefficient zoom **d) Drag coefficient zoom**

Fig. 19 NACA 0012, $\alpha = 10$ deg: convergence of lift and drag coefficients with adaptive mesh refinement and $p = 2$.

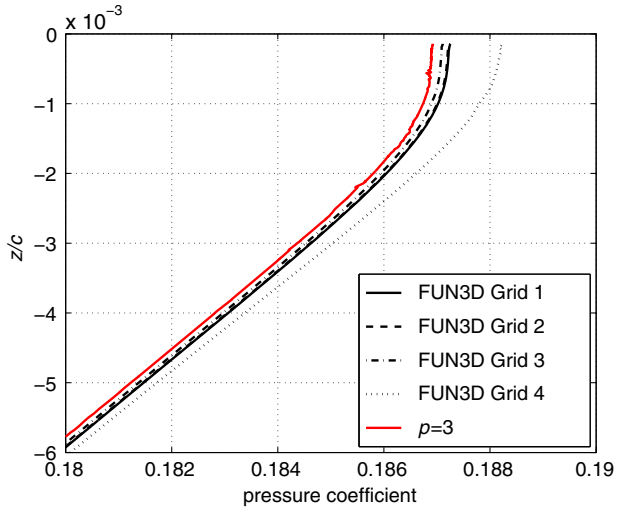


Fig. 20 NACA 0012, $\alpha = 10$ deg: pressure coefficient profiles off the bottom surface of the trailing edge.

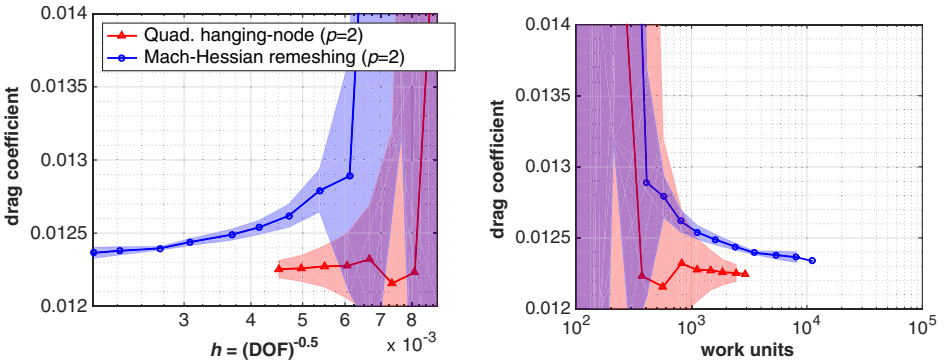
shown, the pressure coefficient is close to the fine grid data. It is possible that the drag-adapted mesh, even at an increased approximation order, is not ideally suited for predicting the off-body pressure coefficient distribution. Adapting on the lift, which would be more sensitive to such pressure errors, could give even better results.

We now compare the efficiency of two adaptive runs for this case: isotropic hanging-node adaptation on quadrilaterals, and triangular anisotropic remeshing based on the Hessian of the Mach number. The remeshing results are the same as those presented in Sec. VI.D, whereas the hanging-node results are equivalent to what is presented Sec. VI.C, with the exception that the far field is located at 500 chords. We emphasize that the CPU time is strongly dependent on solver tuning and adaptive parameters (e.g., f^{frac}, f^{growth}, and initial mesh) and the solvers have not been meticulously optimized for the present results. Instead, both runs used the same solver parameters. The quadrilateral strategy starts with a fairly resolved initial mesh (Fig. 13a), but it is restricted to the initial topology, whereas the triangular remeshing strategy starts with a poor mesh (Fig. 17a), but is not restricted to the initial mesh topology and anisotropy distribution.

Figure 21 compares the two adaptive runs. In this case, the quadrilateral hanging-node strategy is advantageous. The difference observed in number of degrees of freedom is explained by comparing the adapted meshes shown in Figs. 13b and 17b. Note that the triangular remeshing approach using a priori estimates produces a higher density of elements above and below the airfoil away from the regions of interest (stagnation streamline, boundary layer, and wake) than the quadrilateral hanging-node approach. This difference also translates into slower convergence in terms of time. Figure 22 shows the computational cost breakdown of both adaptive strategies. Note that the primal solve is most expensive in the first adaptive step as the flow is initialized with freestream conditions. Then, as the differences in the flowfield become small between consecutive adaptive steps, the cost of the primal solve settles at approximately 60% for hanging nodes and 70% for the remeshing strategy. The error estimation and adaptation procedures represent a larger portion of the cost for hanging nodes than for remeshing based on a priori estimates. Finally, we note that the success of hanging-node refinement is tied to the type of problem and the initial mesh. In the present case, the well-resolved anisotropic initial quadrilateral mesh provides a very good distribution of degrees of freedom, so that the hanging-node refinement converges in only a few iterations.

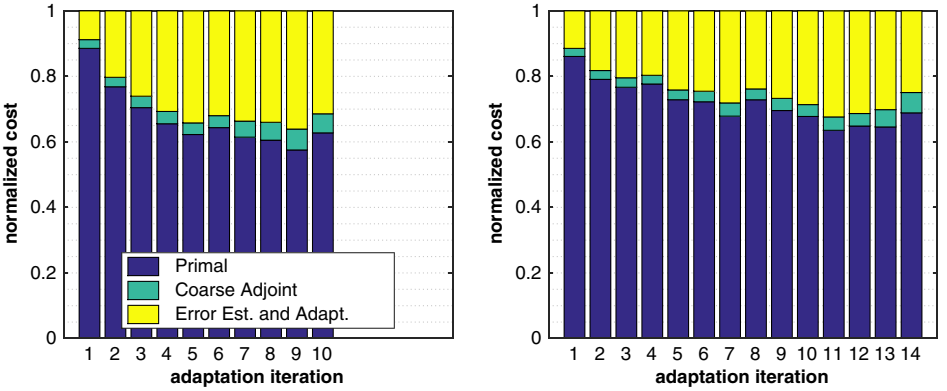
E. NACA 4412, $Re = 1.52 \times 10^6$, $M = 0.09$, $\alpha = 13.87$ deg

This test case consists of a NACA 4412 airfoil at high angle of attack $\alpha = 13.87$ deg, at $Re = 1.52 \times 10^6$, $M = 0.09$. The dynamic viscosity is computed using Sutherland's law (2), with $T_s = 110$ K



a) Drag coefficient versus degrees of freedom b) Drag coefficient versus computational cost

Fig. 21 NACA 0012, $\alpha = 10$ deg: efficiency comparison between hanging-node quadrilateral adaptation and Mach–Hessian remeshing; shaded region denotes $J_H \pm e^c$ [Eq. (12)].



a) Quadrilateral hanging node b) Mach-Hessian remeshing

Fig. 22 NACA 0012, $\alpha = 10$ deg: breakdown of computational cost for adaptive strategies.

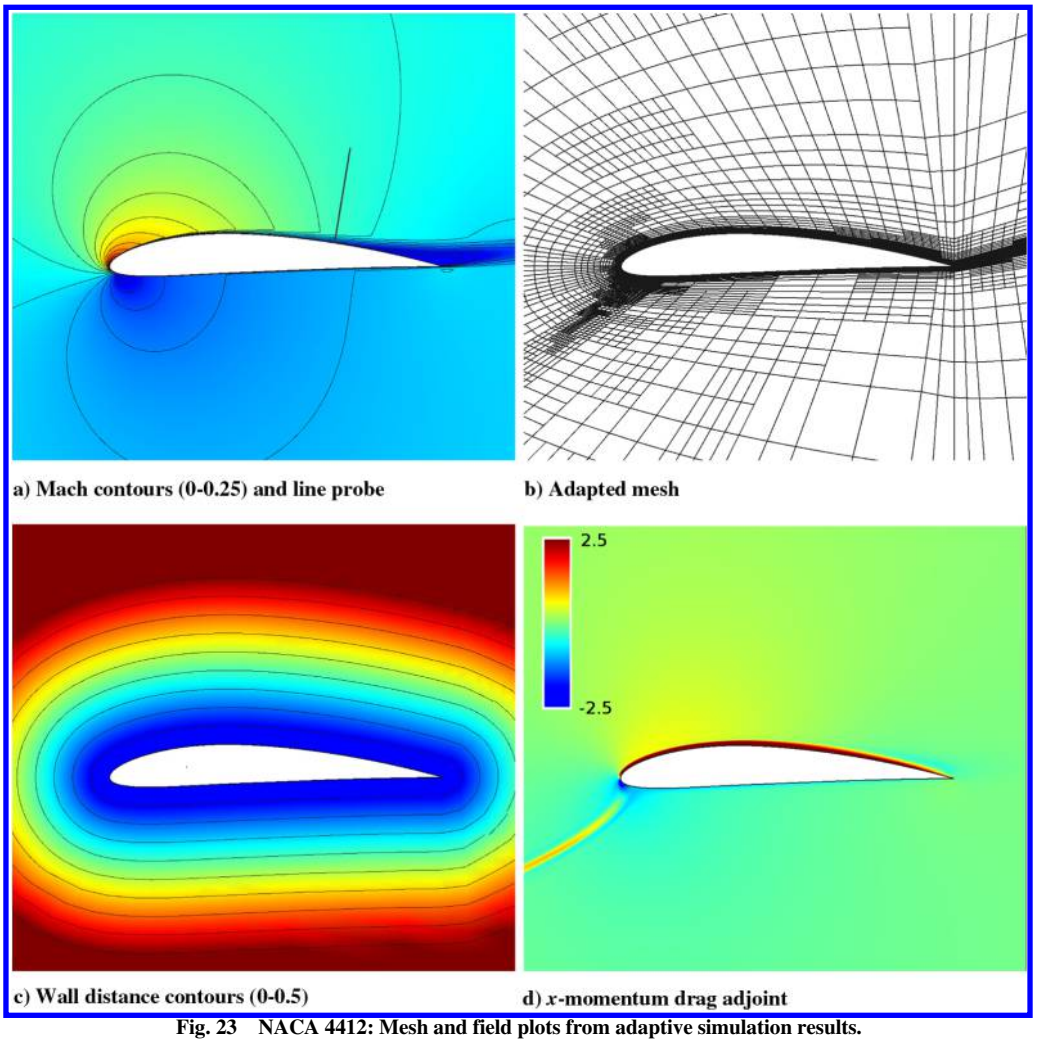
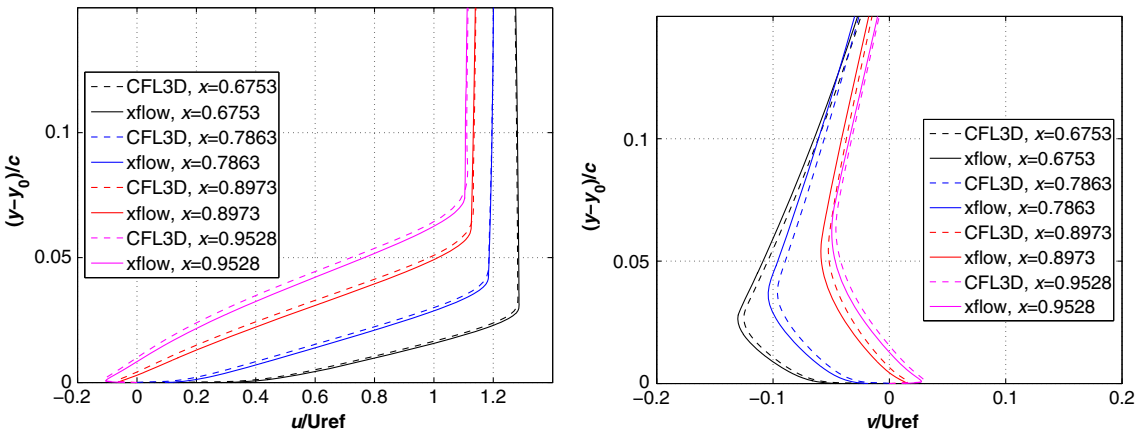


Fig. 23 NACA 4412: Mesh and field plots from adaptive simulation results.

and $T_{\text{ref}} = 297.8$ K. The inflow turbulence eddy viscosity μ_t is set to 3 times the laminar viscosity. Freestream boundary conditions are imposed at a far field that is over 1500 chords away from the airfoil in each direction.

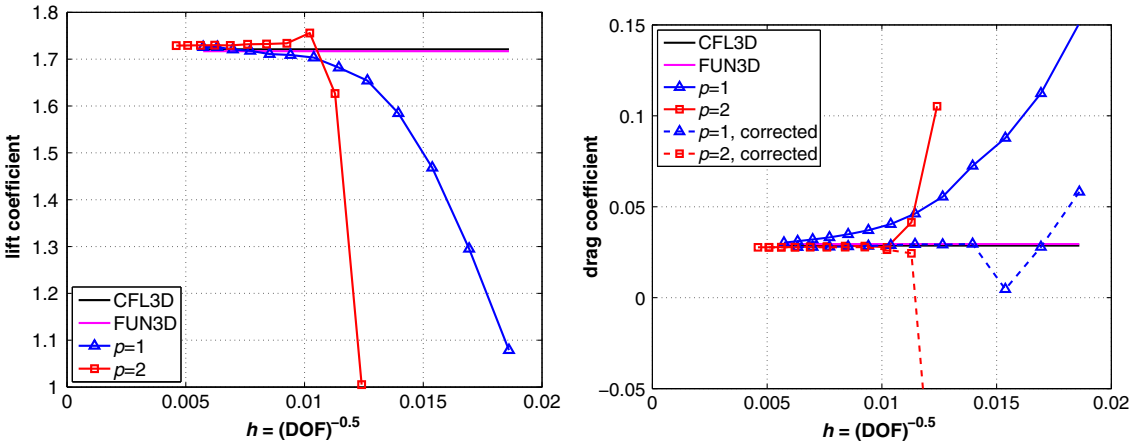
This case was run adaptively using hanging-node refinement of a structured initial mesh, with drag as the target output and a fixed refinement fraction of $f^{\text{frac}} = 0.07$. Figure 23 shows the Mach number contours, an adapted mesh, wall distance contours, and the x momentum component of the drag adjoint. The regions targeted for refinement include the boundary layer and wake, but also a large portion of the mesh in front of the airfoil, on the leading-edge stagnation streamline; note that the adjoint exhibits rapid variation in this area, indicating that error sources near the stagnation streamline can have a large effect on the drag output.

Figure 24 shows velocity profiles along line probes extending roughly normal to the wall at several locations on the aft portion of the airfoil upper surface. The mesh used for this comparison was the final $p = 2$ adapted mesh (after 10 adaptive iterations), uniformly refined and with order increased to $p = 3$ (a total of 140,880 degrees of freedom). Experimental data are available at points on these lines, as



a) Horizontal velocity profiles **b) Vertical velocity profiles**

Fig. 24 NACA 4412: velocity profile comparisons for several wall-normal line probes.



a) Lift coefficient b) Drag coefficient

Fig. 25 NACA 4412: convergence of lift and drag coefficients with adaptive mesh refinement for $p = 1, 2$.

are data from other codes, including CFL3D; in this work, we focus on code-to-code verification. Compared with CFL3D, the horizontal and vertical velocities along the lines are very close. The agreement is not exact, and the small differences could be due to variants in the SA turbulence model or to numerical errors. Note that whereas our data came from meshes adapted to reduce the numerical error, only numerical errors affecting the drag output were targeted, so that the velocity profiles could potentially not yet be converged.

Figure 25 shows the convergence of the drag and lift coefficients with adaptive refinement. Though we only adapt on drag, lift is a similar output and exhibits good convergence too. We see that the drag converges rapidly with indiscernible variations past $h = 0.01$, or 10,000 degrees of freedom for both $p = 1$ and 2 approximation. The lift takes a little longer, in part because of higher sensitivity of the lift to refinement at the trailing edge and because we do not specifically target the lift. For the drag, because we use an adjoint-based method, we have an output error estimate (the adjoint-weighted residual) at each adaptive iteration. We can use this error estimate to correct the drag; these corrected outputs are also shown in Fig. 25. As we converge the adjoint solution to high precision on the fine space, we obtain excellent error corrections: even with $p = 1$ approximation, the corrected drag varies little after about 5000 deg of freedom.

VII. Conclusions

A high-order output-based adaptive solution technique has been presented for the RANS equations closed with a recent variant of the Spalart–Allmaras model, "SA-neg." A discontinuous finite-element method is used for the discretization and key practical details relevant to the implementation are presented. The results compare two variants of the high-order adaptive solution technique to each other and to standard second-order techniques in terms of accuracy versus degrees of freedom. Isotropic quadrilateral-element adaptation using hanging nodes and an anisotropic initial mesh is found to yield similar asymptotic results compared with metric-based unstructured mesh refinement. Using the easy-to-generate isotropic unstructured initial meshes considered, the unstructured adaptation produced larger errors on the first adaptive iterations compared with hanging-node refinement with a more meticulously tailored initial structured mesh. This result is expected given the higher quality of the initial structured meshes. The ability of the unstructured method to automatically snap to the RANS mesh from an initially isotropic mesh is a desirable capability, though it requires robustness of the solver to underresolution.

Relative to uniform refinement at second order, high-order adaptation is found to yield faster convergence: the adaptive runs often quickly snap (close) to the correct solution in a few steps. This comparison does not take into account the computational cost of the error estimation and adaptation, which is primarily that of the fine-space adjoint solve. However, even though the fine space involves an order increment, because the adjoint problem is linear, whereas the RANS equations are highly nonlinear, the adjoint cost is less than (at most ~25%) that of the primal for all cases considered. A topic for future studies, however, is a reduction in the cost of the primal solve, which for the current implementation is likely larger than that of the second-order methods for a given error level.

Acknowledgment

The authors acknowledge support from the U.S. Air Force Office of Scientific Research under grant FA9550-11-1-0081.

References

[1] Yano, M., Modisette, J., and Darmofal, D., "The Importance of Mesh Adaptation for Higher-Order Discretizations of Aerodynamics Flows," AIAA Paper 2011-3852, 2011.

[2] Becker, R., and Rannacher, R., "An Optimal Control Approach to a Posteriori Error Estimation in Finite Element Methods," *Acta Numerica*, edited by Iserles, A., Cambridge Univ. Press, Cambridge, England, U.K., 2001, pp. 1–102.

[3] Venditti, D. A., and Darmofal, D. L., "Anisotropic Grid Adaptation for Functional Outputs: Application to Two-Dimensional Viscous Flows," *Journal of Computational Physics*, Vol. 187, No. 1, 2003, pp. 22–46. doi:10.1016/S0021-9991(03)00074-3

[4] Hartmann, R., and Houston, P., "Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations," *Journal of Computational Physics*, Vol. 183, No. 2, 2002, pp. 508–532. doi:10.1006/jcph.2002.7206

[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, 2011, pp. 673–694. doi:10.2514/1.J050073

[6] Allmaras, S., Johnson, F., and Spalart, P., "Modifications and Clarifications for the Implementation of the Spalart–Allmaras Turbulence Model," *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, 2012, Paper 1902.

[7] Oliver, T. A., "A High-Order, Adaptive, Discontinuous Galerkin Finite Element Method for the Reynolds-Averaged Navier–Stokes Equations," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2008.

[8] Oliver, T. A., and Darmofal, D. L., "Impact of Turbulence Model Irregularity on High-Order Discretizations," AIAA Paper 2009-953, 2009.

[9] Ceze, M. A., and Fidkowski, K. J., "Anisotropic hp-Adaptation Framework for Functional Prediction," *AIAA Journal*, Vol. 51, No. 2, 2013, pp. 492–509. doi:10.2514/1.J051845

[10] Yano, M., "An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2012.

[11] Ceze, M. A., and Fidkowski, K. J., "Drag Prediction Using Adaptive Discontinuous Finite Elements," *Journal of Aircraft*, Vol. 51, No. 4, 2014, pp. 1284–1294. doi:10.2514/1.C032622

[12] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 3, 1981, pp. 357–372.
doi:10.1016/0021-9991(81)90128-5

[13] Bassi, F., and Rebay, S., "Numerical Evaluation of Two Discontinuous Galerkin Methods for the Compressible Navier–Stokes Equations," *International Journal for Numerical Methods in Fluids*, Vol. 40, Nos. 1–2, 2002, pp. 197–207.
doi:10.1002/(ISSN)1097-0363

[14] Oliver, T., and Darmofal, D., "Analysis of Dual Consistency for Discontinuous Galerkin Discretizations of Source Terms," *SIAM Journal on Numerical Analysis*, Vol. 47, No. 5, 2009, pp. 3507–3525.
doi:10.1137/080721467

[15] Leicht, T., and Hartmann, R., "Error Estimation and Anisotropic Mesh Refinement for 3D Laminar Aerodynamic Flow Simulations," *Journal of Computational Physics*, Vol. 229, No. 19, 2010, pp. 7344–7360.
doi:10.1016/j.jcp.2010.06.019

[16] Ceze, M. A., and Fidkowski, K. J., "Constrained Pseudo-Transient Continuation," *International Journal for Numerical Methods in Engineering*, Vol. 102, No. 11, June 2015, pp. 1683–1703.
doi:10.1002/nme.v102.11

[17] Fidkowski, K. J., Oliver, T. A., Lu, J., and Darmofal, D. L., "*p*-Multigrid Solution of High–Order Discontinuous Galerkin Discretizations of the Compressible Navier–Stokes Equations," *Journal of Computational Physics*, Vol. 207, No. 1, 2005, pp. 92–113.
doi:10.1016/j.jcp.2005.01.005

[18] Ceze, M. A., and Fidkowski, K. J., "Output-Driven Anisotropic Mesh Adaptation for Viscous Flows Using Discrete Choice Optimization," AIAA Paper 2010-0170, 2010.

[19] Borouchaki, H., George, P., Hecht, F., Laug, P., and Saltel, E., "Mailleur bidimensionnel de Delaunay gouverné par une carte de métriques. Partie I: Algorithmes," Inst. National de Recherche en Informatique et en Automatique TR-2741, Rocquencourt, France, 1995.

[20] Persson, P.-O., and Peraire, J., "Curved Mesh Generation and Mesh Refinement Using Lagrangian Solid Mechanics," AIAA Paper 2009-0949, 2009.

[21] 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. 2, 2007, pp. 1653–1672.
doi:10.1016/j.jcp.2007.02.007

[22] Baker, T. J., "Mesh Adaptation Strategies for Problems in Fluid Dynamics," *Finite Elements in Analysis and Design*, Vol. 25, Nos. 3–4, 1997, pp. 243–273.
doi:10.1016/S0168-874X(96)00032-7

[23] Castro-Diaz, M. J., Hecht, F., Mohammadi, B., and Pironneau, O., "Anisotropic Unstructured Mesh Adaptation for Flow Simulations," *International Journal for Numerical Methods in Fluids*, Vol. 25, No. 4, 1997, pp. 475–491.
doi:10.1002/(ISSN)1097-0363

Z. J. Wang
Associate Editor