Three-Dimensional Benchmark RANS Computations Using Discontinuous Finite Elements on Solution-Adapted Meshes

Krzysztof J. Fidkowski*

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA

This paper presents results of an output-based, high-order, adaptive fluids solver on two benchmark problems for turbulence model verification. The problems are a hemisphere cylinder and an ONERA M6 wing, and both are solved using the Reynolds-averaged compressible Navier-Stokes equations, with a negative-viscosity formulation of the Spalart-Allmaras (SA) one-equation closure. Initial hexahedral meshes for the adaptive runs are generated manually, with cubic curved elements to adequately represent the geometry. In the adaptive runs, resolution is added using fixed-fraction, hanging-node refinement driven by an output error indicator calculated from a discrete adjoint-weighted residual. Comparisons of the output convergence results to previous data verify the discretization and demonstrate the benefits of high-order and adaptation for the two cases.

I. Introduction

The Reynolds-averaged Navier-Stokes (RANS) equations are routinely used in the analysis and design of aerospace vehicles. Compared to more direct approaches for simulating turbulent flow at high Reynolds numbers, RANS offers computational efficiency, especially on anisotropic meshes. Generating valid anisotropic meshes, in particular for high-order methods that require large, curved elements, remains a challenge and is the subject of ongoing research.

The combination of high-order approximation and output-based mesh adaptation yields a robust solution strategy for RANS simulations.[1,2] Output-based adaptive methods[3–6] offer a systematic approach for identifying regions of the domain that require more resolution for the prediction of scalar outputs of interest, and they return error estimates that can be used for solution verification.

In previous works, we applied our output-based adaptive solution approach to two- and three-dimensional turbulent benchmark problems.[2,7] The present work extends the three-dimensional study to more complex geometries and to more demanding flow conditions. We consider flow over a hemisphere cylinder at various angles of attack, and transonic flow over an ONERA M6 wing at one angle of attack. We show that for the first problem, the combination of high-order approximation and mesh adaptation yields a significant reduction in the number of degrees of freedom required to reach a certain error level in a scalar output. The wing simulation also shows benefits, though not as significant, and the reason for this discrepancy is attributed to the transonic conditions and the shock capturing approach used in the wing case.

The remainder of this paper is organized as follows. Section II presents the compressible Navier-Stokes equations closed with the Spalart-Allmaras RANS model, and Section III discusses their

*Associate Professor, AIAA Senior Member, kfid@umich.edu

discretization with a high-order discontinuous finite-element method. Section IV discusses several implementation aspects specific to three-dimensional RANS simulations, including the output error estimation and adaptation techniques, and Sections V and VI present results for the two benchmark cases considered. Section VII concludes with a summary and a discussion of possible future research directions.

II. The Reynolds-Averaged Compressible Navier-Stokes Equations

In this work we use the compressible Navier-Stokes equations, Reynolds-averaged with a version of the Spalart-Allmaras (SA) turbulence model that is modified for improved stability for negative values of the turbulence working variable, $\tilde{\nu}$.[8] The resulting Reynolds-averaged Navier-Stokes (RANS) equations are, using index notation with implied summation on repeated indices,

$$\begin{array}{lcllcll} \partial_t \rho & + & \partial_j(\rho u_j) & & & = & 0 \\ \partial_t(\rho u_i) & + & \partial_j(\rho u_j u_i + p\delta_{ij}) & - & \partial_j \tau_{ij} & = & 0 \\ \partial_t(\rho E) & + & \partial_j(\rho u_j H) & - & \partial_j(u_i \tau_{ij} - q_j) & = & 0 \\ \underbrace{\partial_t(\rho\tilde{\nu})}_{\text{unsteady}} & + & \underbrace{\partial_j(\rho u_j \tilde{\nu})}_{\text{convective}} & - & \underbrace{\partial_j\left[\frac{1}{\sigma}\rho(\nu+\tilde{\nu}f_n)\partial_j\tilde{\nu}\right]}_{\text{diffusive}} + \underbrace{S_{\tilde{\nu}}}_{\text{source}} & = & 0 \end{array} \tag{1}$$

where the terms have been split according to their treatment in the discretization, and where the source term for the $\tilde{\nu}$ equation is

$$S_{\tilde{\nu}} = \frac{1}{\sigma}(\nu + \tilde{\nu} f_n)\partial_j \rho \partial_j \tilde{\nu} - \frac{c_{b2}\rho}{\sigma}\partial_j \tilde{\nu}\partial_j \tilde{\nu} - P + D.$$

In the above equations, ρ is the density, ρu_j is the momentum, E is the total energy, $H = E + p/\rho$ is the total enthalpy, $p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho u_i u_i\right)$ is the pressure, γ is the ratio of specific heats, P is the turbulence production, D is the turbulence destruction, and i, j index the spatial dimension, dim. The Reynolds stress, τ_{ij}, is

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

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

$$\mu = \mu_{\text{ref}}\left(\frac{T}{T_{\text{ref}}}\right)^{1.5}\left(\frac{T_{\text{ref}} + T_{\text{s}}}{T + T_{\text{s}}}\right) = \mu_{\text{ref}}\left(\frac{T}{T_{\text{ref}}}\right)^{1.5}\left(\frac{1 + \frac{T_{\text{s}}}{T_{\text{ref}}}}{\frac{T}{T_{\text{ref}}} + \frac{T_{\text{s}}}{T_{\text{ref}}}}\right), \tag{2}$$

where $T = p/(\rho R)$ is the temperature, R is the gas constant for air (the difference in specific heats), and the eddy viscosity, μ_t, is

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

The heat flux, q_j, is given by

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

where Pr and Pr_t are the laminar and turbulent Prandtl numbers, and C_p is the specific heat at constant pressure. 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

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

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

$$D = \begin{cases} c_{w1} f_w \dfrac{\rho\tilde{\nu}^2}{d^2} & \chi \geq 0 \\ -c_{w1}\dfrac{\rho\tilde{\nu}^2}{d^2} & \chi < 0 \end{cases}, \quad f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{1/6}, \quad g = r + c_{w2}(r^6 - r), \quad r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2}.$$

Finally, in Eqn. 1, $f_n = 1$ 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	c_{w1}	=	$\frac{c_{b1}}{\kappa^2} + \frac{1 + c_{b2}}{\sigma}$	c_{v1}	=	7.1
c_{b2}	=	0.622	c_{w2}	=	0.3	κ	=	0.41
σ	=	2/3	c_{w3}	=	2	Pr_t	=	0.9
c_{n1}	=	16	c_{v2}	=	0.7	c_{v3}	=	0.9

III. Discontinuous Galerkin Discretization

We discretize Eqn. 1 using a discontinuous Galerkin (DG) finite element method.[9,10] Defining the state vector as $\mathbf{u} = [\rho, \rho u_i, \rho E, \rho\tilde{\nu}]^T$, we write Eqn. 1 in compact conservative form,

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

where $\vec{\mathbf{F}}$ is the combined inviscid/viscous flux vector, and $\mathbf{S}$ is the source term associated with the turbulence closure equation. We approximate the state in a finite dimensional space, $\mathbf{u}_h \in \boldsymbol{\mathcal{V}}_h$, where $\boldsymbol{\mathcal{V}}_h$ is the space of element-wise discontinuous polynomials of order p. Choosing a basis for $\boldsymbol{\mathcal{V}}_h$ yields the following state approximation on element k,

$$\mathbf{u}(\vec{x})\big|_k = \sum_{j=1}^{n(p)} \mathbf{U}_{kj}\phi_j(\vec{x}),$$

where $\mathbf{U}_{kj}$ contains the six unknowns associated with basis function j on element k. We denote by $\mathbf{U} = \{\mathbf{U}_{kj}\}$ all of these unknowns rolled into one vector. By virtue of the discontinuous approximation space inherent to DG, the basis functions used to approximate the state need not correspond to the geometrical shape of the element. For example, whereas tensor product basis functions are typically used on hexahedral elements, in DG one can also use full-order "tetrahedral" basis functions. This results in a significant savings in degrees of freedom for the same nominal order of convergence: for example, a $p = 4$ tensor product basis requires 125 unknowns per element, while a $p = 4$ full-order basis only requires 35 unknowns per element. This yields storage savings factors of over 3.5 for the state and 12 for the residual Jacobian matrix.

Multiplying Eqn. 5 by test functions in $\boldsymbol{\mathcal{V}}_h$, which are the same as the basis functions for DG, integrating by parts on each element, and using the Roe[11] convective flux and the second form of Bassi and Rebay (BR2)[12] for the viscous treatment, we obtain the following system of nonlinear equations,

$$\mathbf{R}(\mathbf{U}) \quad = \quad \mathbf{0}. \tag{6}$$

III.A. Boundary conditions

III.A.1. Free-stream

In this boundary condition, the entire boundary state, $\mathbf{u}^b$, is specified, and a numerical flux function is used to compute the boundary flux,

$$\hat{\mathbf{F}}^b = \hat{\mathbf{F}}(\mathbf{u}^+, \mathbf{u}^b, \vec{n}), \tag{7}$$

where $\mathbf{u}^+$ is the interior state. The flux function performs the required upwinding to properly distinguish inflow and outflow sections of the boundary. The full state boundary condition is typically used at farfield boundaries, where the state is approximately uniform.

III.A.2. Static pressure outflow

At a subsonic outflow, the boundary static pressure, p^b, is specified. The complete exterior state, $\mathbf{u}^b$, is calculated from the Riemann invariant $J^+(\mathbf{u}^+)$, the interior entropy $S^+ = p^+/(\rho^+)^\gamma$, and the interior tangential velocity. The calculation proceeds as follows. First, the exterior density is

$$\rho^b = \left(\frac{p^b}{S^+}\right)^{1/\gamma}. \tag{8}$$

The boundary normal velocity, u_n^b, is found using J^+ and $c^b = \sqrt{\gamma p^b/\rho^b}$,

$$u_n^b = J^+ - \frac{2c^b}{\gamma - 1}. \tag{9}$$

Setting the boundary tangential velocity to the interior tangential velocity then fully defines the boundary velocity $\vec{v}^b$, according to

$$\vec{v}^b = \vec{v}^+ - (\vec{v}^+ \cdot \vec{n})\vec{n} + (\vec{v}^b \cdot \vec{n})\vec{n}.$$

$(\rho E)^b$ is calculated as $p^b/(\gamma - 1) + \frac{1}{2}\rho^b|\vec{v}^b|^2$. The boundary flux is calculated directly from the boundary state.

III.A.3. Adiabatic wall

This is a no-slip boundary condition that is used on the surface of a stationary, insulated object. The boundary state is constructed using the interior density and total energy, and a zero velocity. The boundary flux is then computed from the boundary state, with a projection of the viscous flux: the diffusive mass flux is set to zero, and the diffusive energy flux is set to zero. Note that viscous stabilization terms are included in the calculation of the boundary flux from the boundary state.

III.A.4. Symmetry

Symmetry boundary conditions are applied on planes of symmetry, to allow modeling of half of the domain. Starting with the state, we require that at a symmetry boundary, the velocity vector has its normal component zeroed out. This results in a linear transformation from the interior ($\mathbf{u}^+$) to the boundary ($\mathbf{u}^b$) state vector, which reads $\mathbf{u}^b = \mathbf{A}\mathbf{u}^+$. $\mathbf{A}$ is the identity matrix for all states except the momentum, which transforms as $(\rho\vec{v})^b = \underline{V}(\rho\vec{v})^+$, where $\underline{V} = \underline{I} - \vec{n} \otimes \vec{n} = \delta_{ij} - n_i n_j$. $\vec{n}$ is the outward-pointing normal, and $\underline{I} = \delta_{ij}$ is the dim $\times$ dim identity matrix.

As described in our previous work,[2] the state gradient transformation must account for possibly nonzero normal velocity components. We define an exterior, reflected, state by $\mathbf{u}^- = \mathbf{B}\mathbf{u}^+$, where $\mathbf{B}$ is an identity matrix for all states except the momentum, which transforms as $(\rho\vec{v})^- = \underline{W}(\rho\vec{v})^+$, where $\underline{W} = \underline{I} - 2\vec{n} \otimes \vec{n} = \delta_{ij} - 2n_i n_j$. The exterior gradient is then given by $\nabla\mathbf{u}^- = \mathbf{B}\nabla\mathbf{u}^+\underline{W}^T$, so that the gradient on the boundary is

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

III.B. Scaling of $\tilde{\nu}$

As the SA working variable, $\tilde{\nu}$, is generally orders of magnitude smaller than the other state components, we scale $\tilde{\nu}$ to make its range of numerical values similar to the other state components. This improves solver performance,[10] reducing the number of nonlinear iterations, and in some cases making convergence possible. We store the scaled quantity, $\rho\tilde{\nu}'$, given by

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

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

IV. Implementation

IV.A. Mesh Generation

For the cases in this paper, curved, hexahedral meshes are generated manually as starting meshes for the adaptive runs. High-order curved elements provide geometric fidelity that is essential for robust and accurate solutions via the discontinuous Galerkin method, even at solution approximation orders of $p = 1$.[13] In this work, we employ curved elements, near the boundary and for several layers off it, as the presence of highly-anisotropic elements near the wall precludes the use of just one layer of curved elements. The anisotropy is needed for efficiency, as RANS simulations require much higher resolution, i.e. smaller length scales, perpendicular to the wall compared to parallel to the wall, so that anisotropic elements reduce the total degrees of freedom without sacrificing accuracy.

In this work, each hexahedron in physical space is obtained by mapping a unit reference cube via tensor-product polynomials of order q. Nodal Lagrange basis functions, with equal node spacing in reference space, yield curved elements that interpolate the provided nodes, $(q+1)^3$ total. This property is useful for defining curved hexahedra via the coordinates of their $(q+1)^3$ nodes. The high-order nodes inside the elements are spaced approximately uniformly in physical space to minimize skewness and to avoid the risk of negative mapping Jacobians.

Although we use high-order curved elements, slope/normal continuity is not explicitly enforced at inter-element boundaries. However, the mismatch in the slope diminishes as the mesh is refined.

Curved elements require non-canonical mass matrices, and these are computed via quadrature and stored for the duration of the run. Upon refinement, new nodes placed on the curved boundary are snapped to the geometry. This demands sufficient resolution from the initial mesh to prevent element inversion, as only the first layer of elements adjacent to the boundary is affected by the snapping.

IV.B. Wall Distance Calculation

The distance to the closest wall, d, is required for the Spalart-Allmaras turbulence model. This distance is used at every quadrature point, during the construction of the residual and residual Jacobian matrix. Rather than separately storing the distance at every quadrature point, the wall distance is approximated by a polynomial of order p_{wd} on each element. This polynomial is constructed by evaluating the wall distance at each Lagrange node used in the polynomial approximation on an element. These distance calculations involve projection of the point in question to a faceted, but refined, representation of the surface, followed by snapping to the true curved geometry.[2] For the results in this paper, we use $p_{\text{wd}} = 2$.

IV.C. Nonlinear Solver

The DG discretization yields a system of nonlinear equations, $\mathbf{R}(\mathbf{U}) = \mathbf{0}$, that must be solved for the unknown solution approximation coefficients, $\mathbf{U}$. We use a Newton-Raphson method with pseudo-transient continuation for robustness when not close to the root. Specifically, at each Newton iteration, the following linear system is solved via the restarted generalized minimal residual method (GMRES[14]):

$$\left(\frac{\mathbf{M}}{\Delta t_a} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Big|_{\mathbf{U}_0}\right)\Delta\mathbf{U} + \mathbf{R}(\mathbf{U}_0) \quad = \quad \mathbf{0}, \tag{10}$$

where $\mathbf{U}_0$ is the solution guess, $\mathbf{M}$ is the block-diagonal mass matrix, and Δt_a is an element-specific artificial time step given by

$$\Delta t_a = \text{CFL}\frac{h}{c_{\max}},$$

with h = volume/(surface area), and $c_{\max}$ the maximum characteristic speed over the quadrature points of the element. CFL is a global Courant-Friedrichs-Lewy number that starts low and progressively increases according to an evolution strategy, described next, as the solution converges.

When not close to the root, the solution to Eqn. 10 may produce an update that makes $\mathbf{U}_0+\Delta\mathbf{U}$ non-physical. We therefore under-relax the update according to the following line-search strategy:

1. Given: $\mathbf{U}_0$ and $\Delta\mathbf{U}$, the solution to Eqn. 10.
2. Compute ω^{phys} = maximum fraction such that $\mathbf{U}_0+\omega^{\text{phys}}\Delta\mathbf{U}$ remains physical. This involves checks at quadrature points of each element.
3. Set $\omega = \min(1, \omega^{\text{phys}})$. If $\omega < 1$, set $\omega = \omega\beta^{\text{phys}}$, where $\beta^{\text{phys}} < 1$ is a buffer reduction factor that keeps the solution from being borderline non-physical.
4. While $\omega > \omega^{\min}$ and $\|\mathbf{R}(\mathbf{U}_0 + \omega\Delta\mathbf{U})\| > \beta^{\text{residual}}\|\mathbf{R}(\mathbf{U}_0)\|$: set $w = w\beta^{\text{line}}$, where $\beta^{\text{line}} < 1$.
5. If $\omega < \omega^{\min}$, do not take the update. Instead, set $\text{CFL} = \text{CFL}\beta^{\text{CFL,decrease}}$ and return to the first step.
6. If $\omega \geq \omega^{\min}$, take the update: $\mathbf{U} = \mathbf{U}_0+\omega\Delta\mathbf{U}$. Furthermore, if $\omega = 1$, raise the CFL number: $\text{CFL} = \text{CFL}\beta^{\text{CFL,increase}}$. Return to the first step.

The parameters do not need much tuning, and the ones below used in this study also work for many other problems.

$$\beta^{\text{phys}} = 0.5,\ \beta^{\text{residual}} = 2.0,\ \beta^{\text{line}} = 0.5,\ \omega^{\min} = 0.24,\ \beta^{\text{CFL,increase}} = 1.2,\ \beta^{\text{CFL,decrease}} = 0.1.$$

In all runs, the starting CFL number is 0.1.

IV.D. Shock Capturing

In the ONERA M6 wing case, the flow is transonic and requires stabilization to prevent solution oscillations at the shocks from hampering convergence. Stabilization is introduced in the form of element-constant artificial viscosity,[15] where the amount of viscosity is dictated by a resolution indicator, defined as

$$S_\kappa = \frac{\int_\kappa (\rho - \hat{\rho})^2 \, dx}{\int_\kappa \rho^2 \, dx}, \tag{11}$$

where ρ is the density, and $\hat{\rho}$ is a truncated polynomial representation of ρ of one lower order. The smoothness indicator is converted to an artificial viscosity using a nonlinear switch,

$$\epsilon_\kappa = \epsilon_0 \frac{f^2}{f+1}, \qquad f \equiv \frac{S_\kappa}{S_0}, \tag{12}$$

where S_0 is an empirically-defined constant dependent on the order, and where ϵ_0 is a viscosity measure that scales as h/p: $\epsilon_0 = \frac{\bar{\lambda}_{\max} h}{p}$, where $\bar{\lambda}_{\max}$ is a local maximum characteristic propagation speed, and h is a local mesh size. For anisotropic grids, h can be made directional by using a metric.[16] In Eqn. 12, the solution is deemed smooth as long as the smoothness indicator S_κ is well below S_0. If the indicator value is comparable to or greater than S_0, the amount of artificial viscosity added will be roughly proportional to S_κ/S_0. The form of the artificial viscosity term added to the PDE is Laplacian, which means that the same viscosity is used for all equations.

IV.E. Error Estimation

We use an adjoint-based output error estimate to drive mesh adaptation. The discrete system of nonlinear equations reads $\mathbf{R}(\mathbf{U}) = \mathbf{0}$, where the state and residual vectors are both in $\mathbb{R}^N$. For a scalar output, $J(\mathbf{U})$, the discrete adjoint vector, $\mathbf{\Psi} \in \mathbb{R}^N$, is the sensitivity of J to perturbations in $\mathbf{R}$.[6] It satisfies the following linear equation,

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^T \mathbf{\Psi} + \left(\frac{\partial J}{\partial \mathbf{U}}\right)^T = \mathbf{0}. \tag{13}$$

The adjoint vector provides an estimate of the error in the output when computing on a finite-dimensional approximation space. Consider 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 1), on which we compute the adjoint and relative to which we estimate the error. We would like to measure the output error in the coarse solution relative to the fine space,

$$\text{output error: } \delta J \equiv J_H(\mathbf{U}_H) - J_h(\mathbf{U}_h). \tag{14}$$

We assume that the fine approximation space contains the coarse approximation space, so that a lossless state injection, $\mathbf{U}_h^H \equiv \mathbf{I}_h^H \mathbf{U}_H$, exists, where $\mathbf{I}_h^H$ is the coarse-to-fine state injection (prolongation) operator. The injected state will generally not give zero fine-space residuals, and we

expect an output perturbation given by the inner product between the adjoint and the residual perturbation,

$$\delta J \approx -\mathbf{\Psi}_h^T \mathbf{R}_h(\mathbf{U}_h^H). \tag{15}$$

This estimate assumes small perturbations in the state when the output or equations are nonlinear. It does not require the fine-space primal solution, $\mathbf{U}_h$, but it does require the fine-space adjoint. In this work, we fully converge the fine-space adjoint about the injected state, $\mathbf{U}_h^H$, storing the fine-space Jacobian and using $\mathbf{\Psi}_h^H \equiv \mathbf{I}_h^H \mathbf{\Psi}_H$ as a initial guess in the GMRES iterative solver for $\mathbf{\Psi}_h$. We note that techniques such as iterative smoothing or reconstruction can be used to approximate the fine-space adjoint and reduce the cost of its calculation.[6,9,10]

In an adjoint-consistent formulation, the discrete adjoint vector $\mathbf{\Psi}$, consists of expansion coefficients that, when combined with the finite element basis (test) functions, approximate the continuous adjoint vector. We can therefore visualize the adjoint, i.e. the sensitivity of the output to residual perturbations, in the same way as the state. We note that in this work, the discretization is not strictly adjoint consistent, as the state gradients required for the turbulence source term are computed directly from the element-interior state, not accounting for the inter-element jumps. However, the impact of this inconsistency on the adaptive refinement has previously been shown to not be very large.[17]

IV.F. Error Localization

The adjoint-weighted residual error estimate in Eqn. 15 can be localized to the elements, indexed by k, according to

$$J_H(\mathbf{U}_H) - J_h(\mathbf{U}_h) \approx -\mathbf{\Psi}_h^T \mathbf{R}_h(\mathbf{U}_h^H) = -\sum_k \mathbf{\Psi}_{hk}^T \mathbf{R}_{hk}(\mathbf{U}_h^H) \quad \Rightarrow \quad \epsilon_k = \left| \mathbf{\Psi}_{hk}^T \mathbf{R}_{hk}(\mathbf{U}_h^H) \right|,$$

where the subscript k indicates degrees of freedom associated with element k, and the adaptive indicator ϵ_k is the absolute value of the elemental contribution. High values of the error indicator denote elements that contribute the most to the output error, and these should be targeted for refinement to reduce their residual and to improve their approximation of the adjoint.

IV.G. Hanging-Node Hexahedral Mesh Refinement

The mesh refinement strategy used in this work is hanging-node subdivision of elements in an initially structured hexahedral mesh.[9,10,18] In this strategy, a fixed fraction, $f^{\text{frac}} = .03$, of elements with the highest error indicator is flagged for refinement. For the present results, we consider isotropic refinement in which each hexahedron is subdivided uniformly into eight hexahedra. This 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.

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.

On curved boundaries, new nodes are snapped to the geometry, a perturbation that is usually small when high-order curved elements are used. This perturbation is propagated into the interior of the first layer of elements adjacent to the boundary by a linear weight that drops to zero on the face opposite the boundary. A caveat of snapping to geometry with three-dimensional hanging-node meshes is that it is easy to generate invalid, non water-tight, meshes. When the nodes of the small elements on the refined side of the hanging-node face are snapped to the geometry, their edges

and faces may no longer match the edges and face of the coarse-side element. We employ a simple water-tightness fix to the mesh: following the snapping of boundary nodes and propagation to element interiors, we loop over all hanging node faces in the mesh and set all nodes of the fine-side elements on the face to match the geometry approximation on the coarse-side neighbors.

V. Hemisphere Cylinder Results

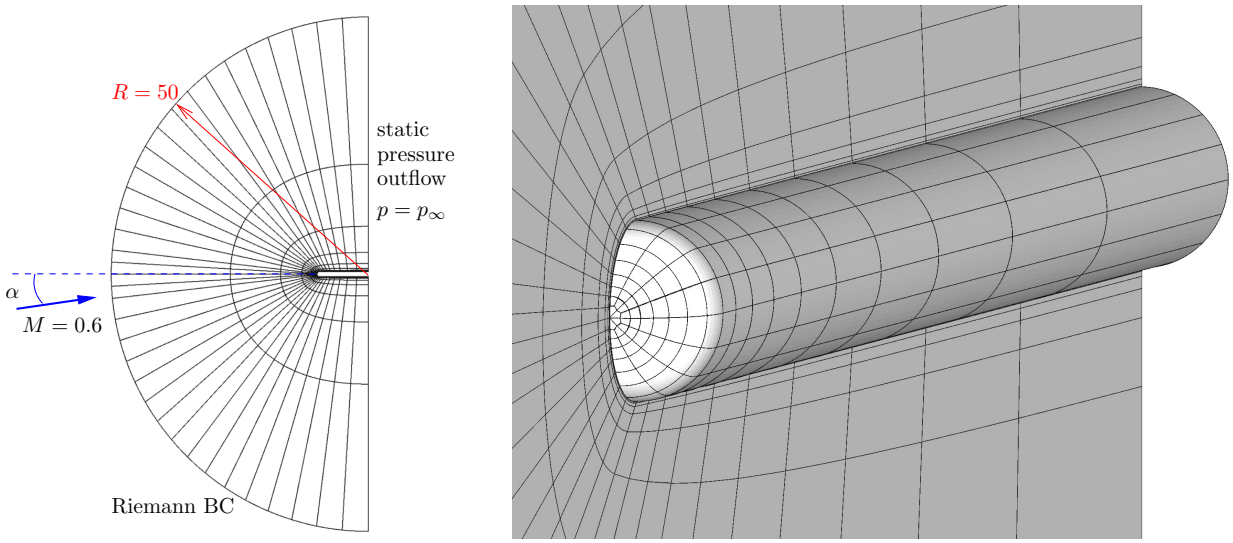
V.A. Problem Description

The geometry in this case consists of a cylinder of length 9.5 and radius $r_c = 0.5$ capped with a hemisphere of radius 0.5, for a total body length of 10. For comparison to experiment, the units of length correspond to inches. We model half the body, and the reference area for computing force coefficients is 5. The body is placed in an external flow with $M = 0.6$ and Reynolds number per unit length of $Re = 3.5 \times 10^5$.

Figure 1 illustrates the problem setup and the initial computational mesh. The farfield is placed 50 units away from the base of the body, and a full-state boundary condition is imposed with a Riemann flux. The free-stream conservative state vector, in convenient units with $\rho_\infty = 1, |\vec{v}_\infty| = 1$, is

$$\mathbf{u}_\infty = \left[1, \cos\alpha, 0, \sin\alpha, \frac{1}{\gamma(\gamma-1)M^2}, \frac{3}{Re}\right]^T,$$

where $\gamma = 1.4$. Note that the free-stream turbulence working variable is set to $3\nu_\infty$, where $\nu_\infty = 1/Re$ is the free-stream laminar viscosity, which is also the reference viscosity for Sutherland's law. The Sutherland's law ratio is $T_s/T_{\text{ref}} = 110.33/300$, and in our units $T/T_{\text{ref}} = \rho p/p_\infty$, where the free-stream static pressure is $p_\infty = 1/(\gamma M^2)$. This static pressure is also used for the outflow.



(a) Domain definitions (b) Baseline surface mesh

Figure 1. Hemisphere cylinder: problem setup and baseline mesh.

When refining boundary elements on the body during the adaptive runs, all additional nodes, including high-order ones, are snapped to the true geometry, according to

$$\vec{x}' = \begin{cases} (x, yr_c/\sqrt{y^2+z^2}, zr_c/\sqrt{y^2+z^2}) & \text{if } x \geq r_c \\ \vec{x}r_c/\sqrt{x^2+y^2+z^2} & \text{if } x < r_c \end{cases}$$

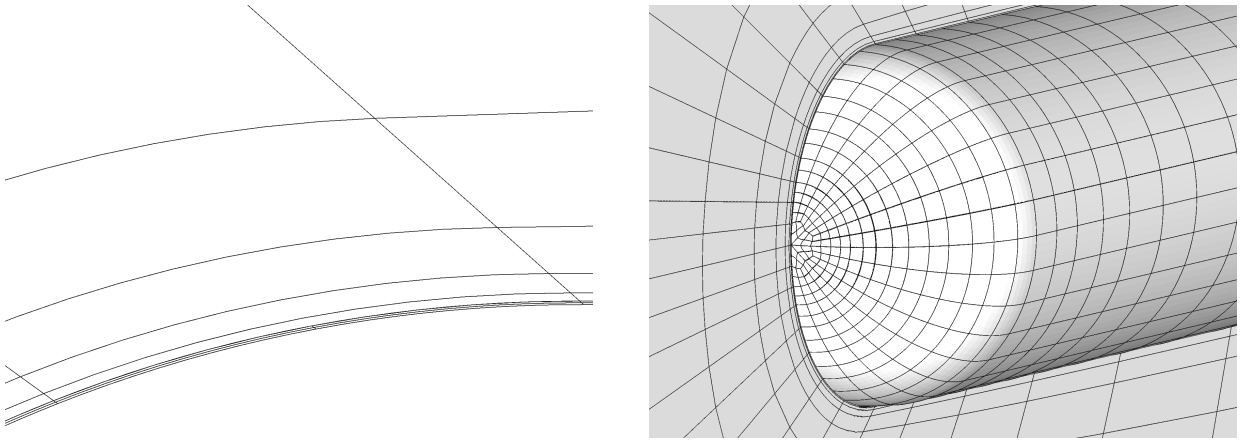
where $\vec{x} = (x, y, z)$ are the coordinates of the original point, $r_c = 0.5$ is the cylinder/sphere radius, and $\vec{x}'$ is the new point. Note that the origin is at the hemisphere apex, x is the axis of the cylinder, y is normal to the plane of symmetry, and z points upwards in the symmetry plane.

V.B. Mesh Generation

The baseline mesh, shown in Figure 1(b), consists of 1680 hexahedral elements. This mesh is constructed by paving both the hemisphere cylinder body and the farfield with the same number and pattern of quadrilaterals. On the body, the spacing of the quadrilaterals is clustered towards the nose and becomes progressively larger towards the cylinder base. Eight equal intervals are used along the 180° arc of the half body of revolution. Note the special treatment of the hemisphere apex: instead of using eight triangles, four skewed quadrilaterals are constructed by agglomerating pairs of adjacent triangles, as shown in Figure 1(b). This lets us use our hanging-node hexahedral adaptation without modification for hybrid hexahedral/prismatic meshes.

The volume mesh is constructed by connecting the body and farfield quadrilateral surface meshes with straight lines. Note that the distribution of quadrilaterals on the farfield mesh is not as clustered near the nose as on the body. The radial spacing of nodes emanating from the body to the farfield is exponential, with highly-anisotropic elements near the wall. The spacing of the first element off the wall is .0015 at the nose of the hemisphere, and grows to .002 at the outflow. The elements are first generated as all curved, using a cubic ($q = 3$) tensor product geometry mapping, and then as many elements as possible are snapped to linear ($q = 1$), with the constraint that no negative Jacobians are created. Several layers of elements near the wall remain $q = 3$.

The initial mesh for the adaptive calculation consists of the baseline mesh with all elements adjacent to the wall uniformly refined once. This yields a mesh of 2660 elements, illustrated in Figure 2.



(a) Symmetry plane zoom near the shoulder

(b) Initial surface mesh

Figure 2. Hemisphere cylinder: initial mesh for adaptive runs.

V.C. Output Convergence

For comparison to existing data, we consider angles of attack $\alpha = 0, 10, 15, 19$ degrees. For each α, we run separate adaptive simulations that target drag and lift outputs. When computing the force on the body, we account for the fact that the base of the cylinder is on the static pressure outflow by adding $-p_\infty \pi r_c^2 / 2\hat{\mathbf{x}}$ to the force. The drag force is then the component of the force

aligned with the free-stream, and the lift force is the component in the $x - z$ plane perpendicular to the free-stream.

We record the lift and drag coefficients at each iteration of the adaptive simulations. Figures 3 through 7 show the convergence of the lift and drag coefficients with adaptive refinement, with an additional curve obtained by using the error estimate as an output correction. Data from other codes, available on NASA's Turbulence Modeling Resource (TMR) website,[19] is also included for comparison. Note that the definition of $h = \text{dof}^{-1/3}$ accounts for the number of degrees of freedom per element, which is 10 for a $p = 2$ full-order (tetrahedral) basis.

For $\alpha = 0°$, in Figure 3, we see that the structured meshes yield zero lift identically, due to mesh symmetry, whereas the unstructured grids only yield asymptotic convergence to zero lift. The present results, labeled XFLOW in the figures, begin with a nominally structured baseline mesh that becomes unstructured after adaptation. Although the mesh is adapted for drag, the lift shows very quick asymptotic convergence, as is particularly evident in the zoomed-in plot. For drag, the convergence is again very quick, with the closest competitor being the stabilized continuous finite element method (SFE). At tight error tolerances, the output-based adaptive runs flatten out at/near the exact values much sooner than the runs from the other codes. Specifically, the gain in h is a factor of 5 to 10, which means that the meshes for adaptive, high-order runs may have up to 1000 fewer degrees of freedom compared to the second-order codes for this case.

In Figure 3 we also compare two different approximation orders, $p = 2$ and $p = 3$. Both show similar convergence properties, with $p = 3$ flattening out a little sooner than $p = 2$ in the zoomed-in drag plot, though showing more oscillations in the lift plot.

The results for angles of attack of $5°, 10°$, and $15°$ in Figures 4 to 6 show similar trends: the present drag- or lift-adapted meshes yield outputs that converge must faster than the traditional codes. SFE, the stabilized finite-element solver, is again a strong competitor, though it still has higher degree of freedom counts compared to the XFLOW adaptive runs.

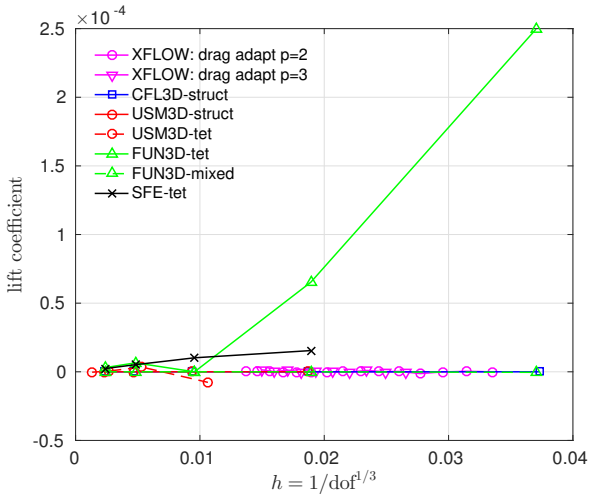
Figure 7 shows the convergence of the drag and lift under adaptive refinement for $\alpha = 19°$. Compared to the lower angles, the convergence in this case is not as rapid, particularly in the latter iterations, where variations are still visible. Nonetheless, the savings in degrees of freedom compared to other codes is still quite large, in particular for the drag output.

V.D. Adapted Meshes and Field Plots

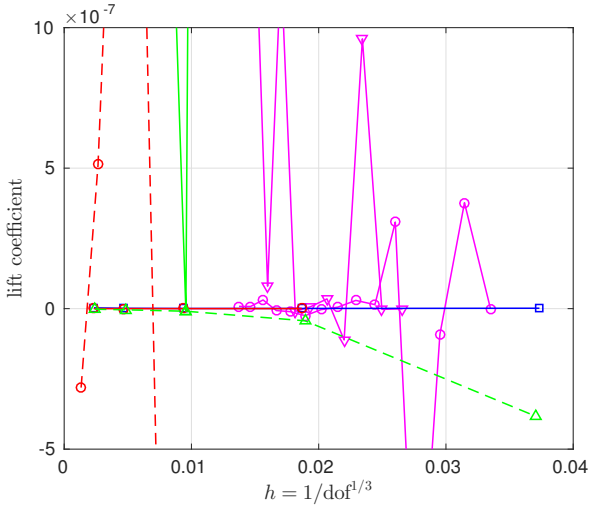
Figures 8 to 10 show the adapted meshes and field plots of the Mach number and pressure for $\alpha = 0°$ and $\alpha = 19°$. The drag-adapted mesh for $\alpha = 0°$ shows the expected refinement in the boundary layer, near the leading edge of the body, and in the area in front of the body. From the surface mesh, we see that all areas of the body are generally refined to the same extent. This is expected in the azimuthal direction due to the symmetry of the flow. Elements away from the body, especially towards the outflow, are not targeted for refinement and remain quite large.

The drag and lift adapted meshes for $\alpha = 19°$, in Figures 9 and 10, respectively, show more asymmetric refinement. In both meshes, the areas targeted for refinement include the upper surface, the leading-edge stagnation streamline, and the lower surface at 25%-30% of the body length. The refinement on the upper surface extends off-body due to separated flow in this region. The lift-adapted mesh targets the leading-edge stagnation streamline more heavily than does the drag adapted mesh, due to to a stronger singularity in the lift adjoint in this region compared to the drag adjoint. Both meshes exhibit refinement at the intersection of the body and the outflow boundary that is heavier than in the surrounding areas. This suggests that the uniform static pressure outflow boundary condition, as implemented, may not be appropriate immediately adjacent to the body.

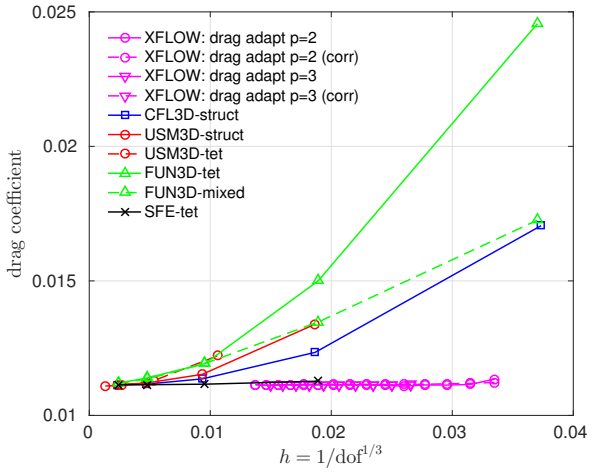
Figure 11 shows the conservation of energy component of the lift and drag adjoint fields for the $\alpha = 19°$ case. We see strong variation of the adjoint in the leading-edge stagnation streamline, in



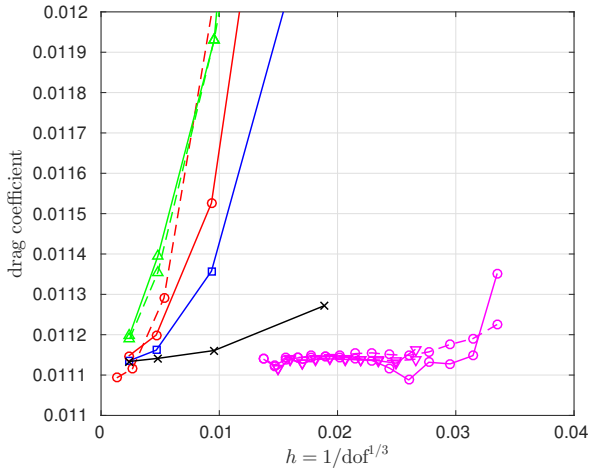
(a) lift convergence



(b) lift convergence (zoom)

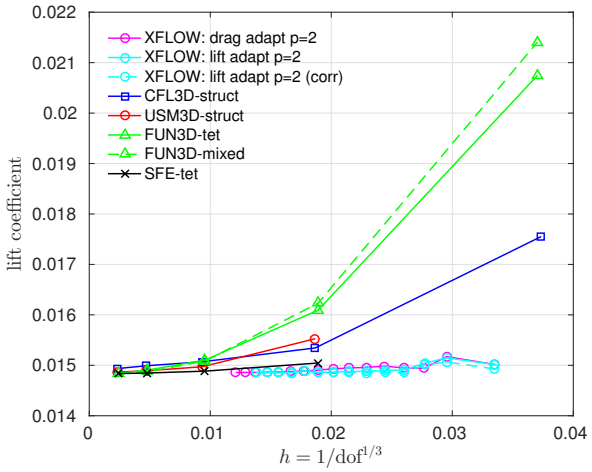


(c) drag convergence

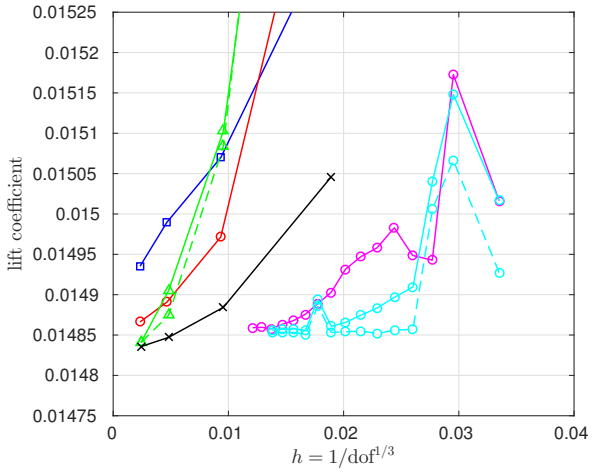


(d) drag convergence (zoom)

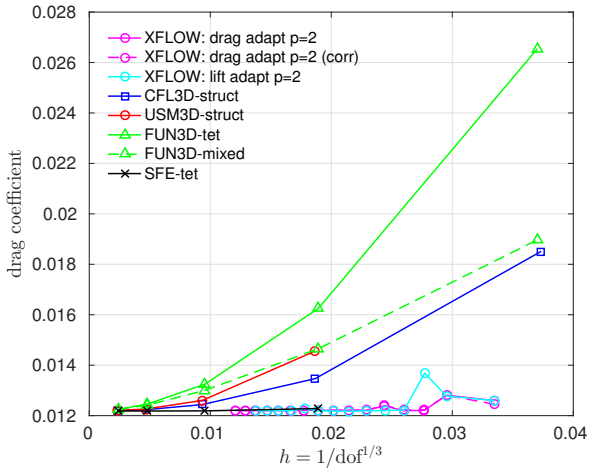
Figure 3. Hemisphere cylinder: lift and drag convergence for $\alpha = 0°$.



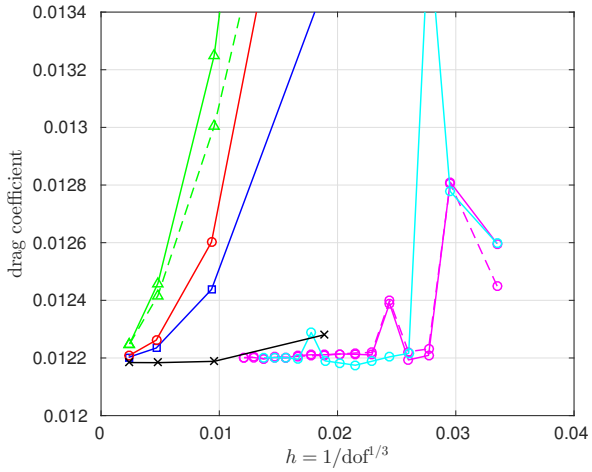
(a) lift convergence



(b) lift convergence (zoom)

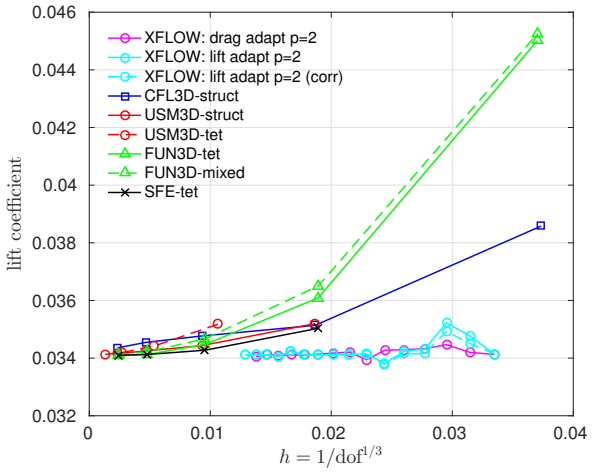


(c) drag convergence

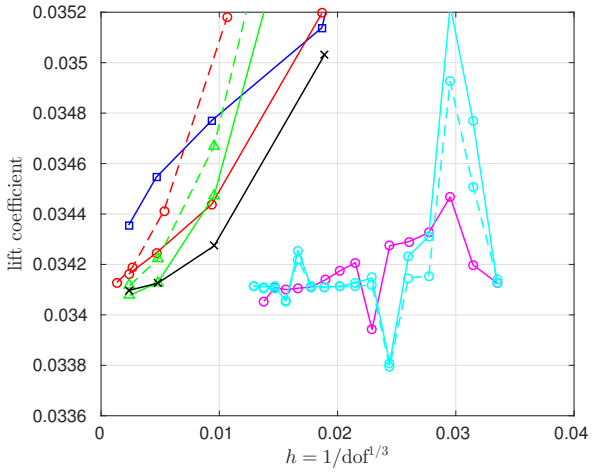


(d) drag convergence (zoom)

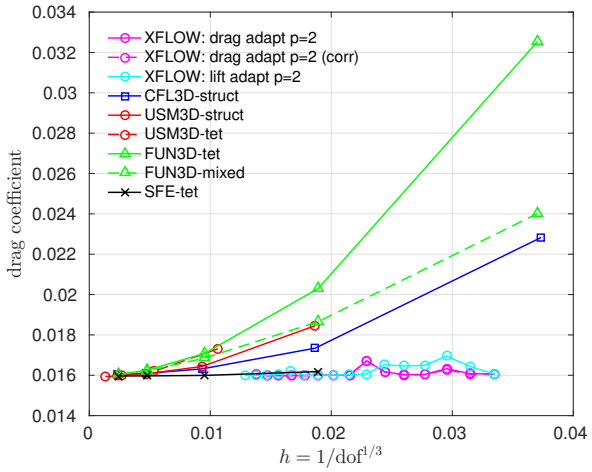
Figure 4. Hemisphere cylinder: lift and drag convergence for $\alpha = 5°$.



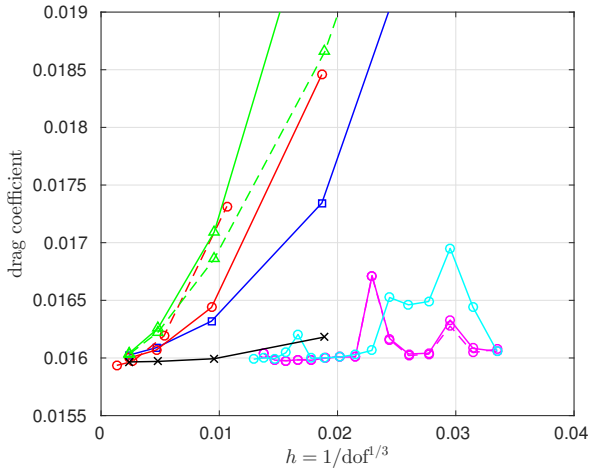
(a) lift convergence



(b) lift convergence (zoom)

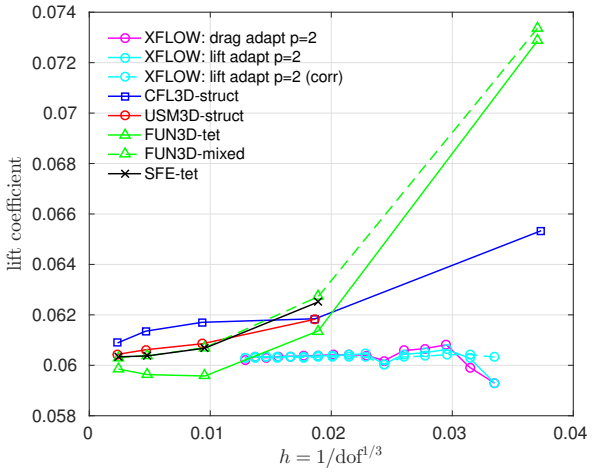


(c) drag convergence

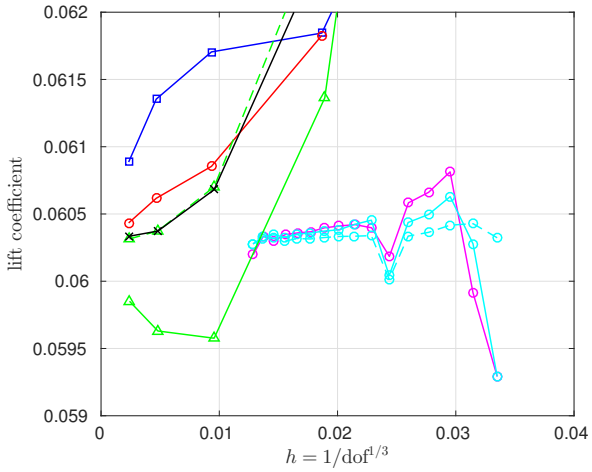


(d) drag convergence (zoom)

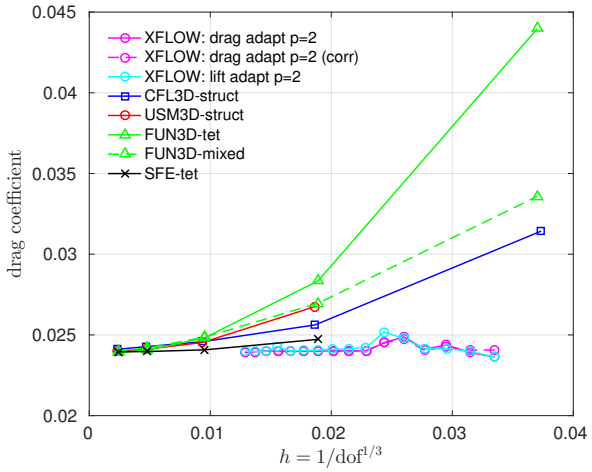
Figure 5. Hemisphere cylinder: lift and drag convergence for $\alpha = 10°$.



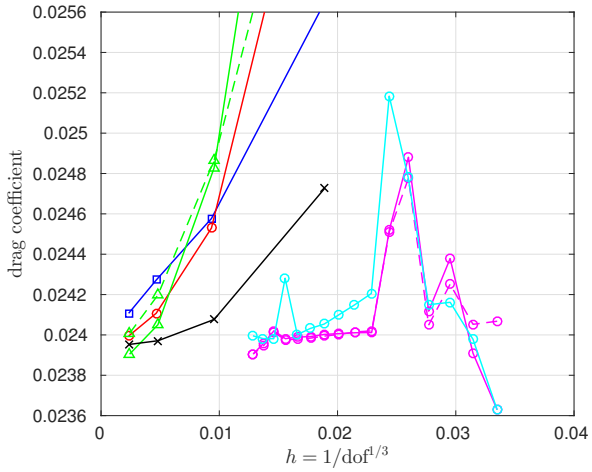
(a) lift convergence



(b) lift convergence (zoom)

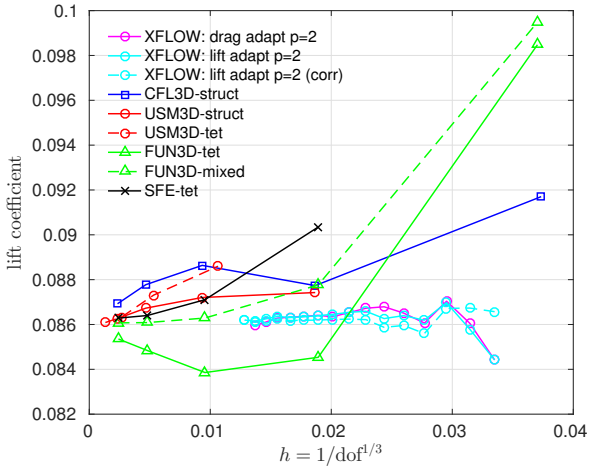


(c) drag convergence

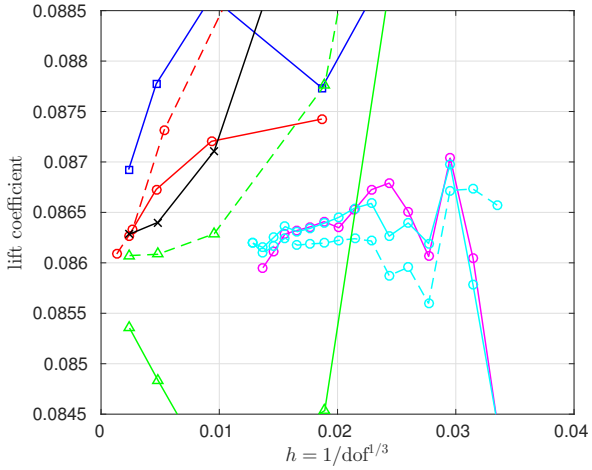


(d) drag convergence (zoom)

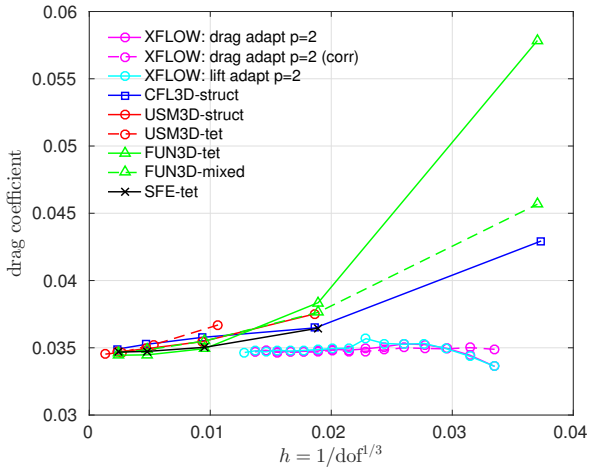
Figure 6. Hemisphere cylinder: lift and drag convergence for $\alpha = 15°$.



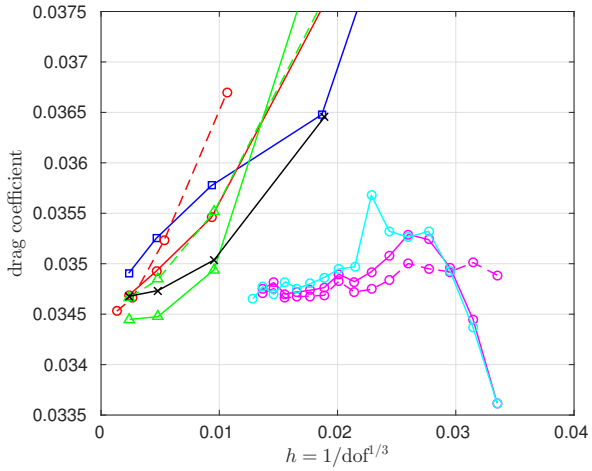
(a) lift convergence



(b) lift convergence (zoom)

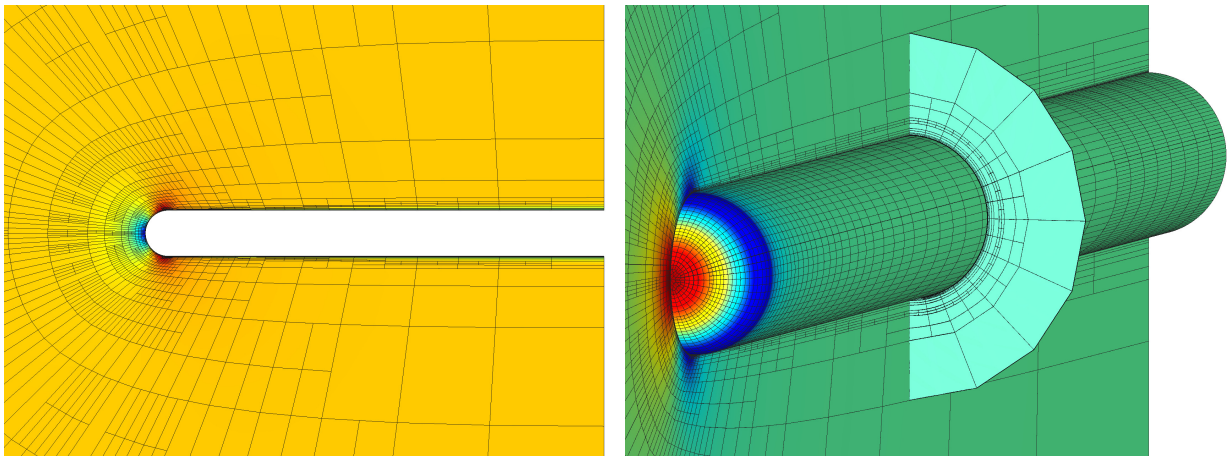


(c) drag convergence



(d) drag convergence (zoom)

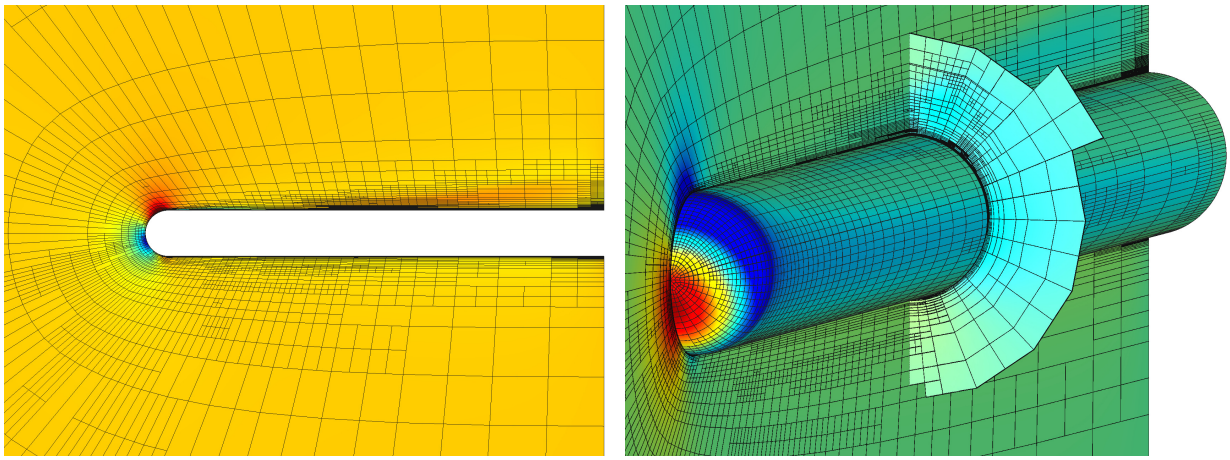
Figure 7. Hemisphere cylinder: lift and drag convergence for $\alpha = 19°$.



(a) Mach contours on symmetry plane

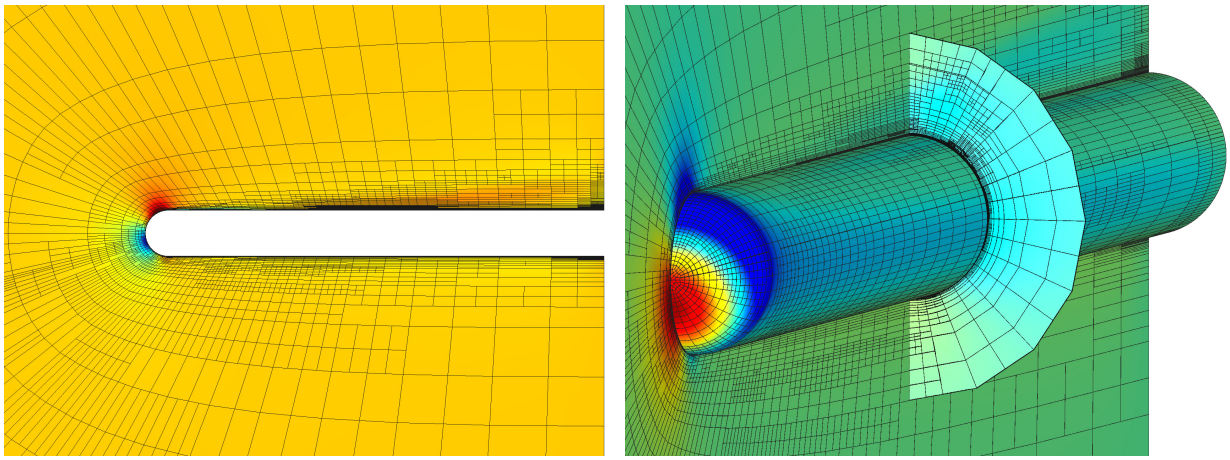
(b) Pressure contours on body and cut

Figure 8. Hemisphere cylinder: $14^{\mathbf{th}}$ drag-adapted mesh, 38578 elements, and solution for $\alpha = 0°$.



(a) Mach contours on symmetry plane (b) Pressure contours on body and cut

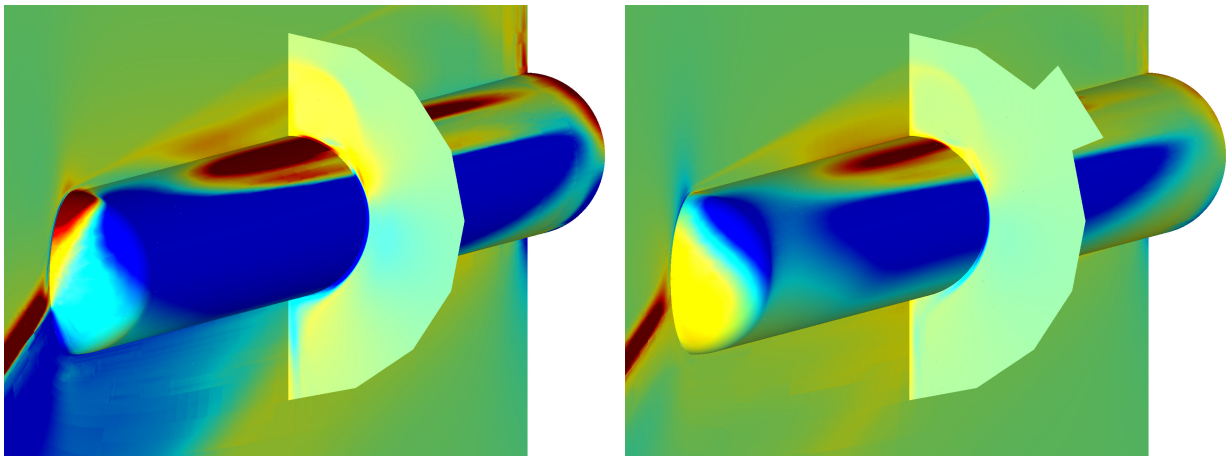
Figure 9. Hemisphere cylinder: 14^{th} drag-adapted mesh, 38974 elements, and solution for $\alpha = 19°$.



(a) Mach contours on symmetry plane (b) Pressure contours on body and cut

Figure 10. Hemisphere cylinder: 14^{th} lift-adapted mesh, 38932 elements, and solution for $\alpha = 19°$.

the vicinity of the separated flow on the upper surface, and near the juncture of the body with the outflow boundary. These variations are consistent with the refinement patterns observed in the adapted meshes. The non-smooth behavior of the adjoint at the outflow boundary appears to be grid-dependent, which again suggests that the static pressure boundary condition is not appropriate in close proximity to the body.



(a) Lift adjoint (b) Drag adjoint

Figure 11. Hemisphere cylinder: lift and drag adjoint fields, conservation of energy component, on the 14^{th} adapted meshes, for $\alpha = 19°$. The color ranges have been clipped to show the most variation.

Finally, Figure 12 shows the state along a vertical line at $x = 5$, $y = -0.21$, for the 14^{th} drag-adapted mesh at $\alpha = 19°$. The pressure has been nondimensionalized by the free-stream density times the square of the free-stream speed of sound. The velocity components have been nondimensionalized by the freestream speed of sound. For comparison, data from the finest-mesh CFL3D runs are also shown. We observe very good agreement in all of the state components, even though the drag-adapted mesh is not designed to necessarily produce accurate off-body field states, but instead an accurate integrated output along the body surface.

VI. ONERA M6 Results

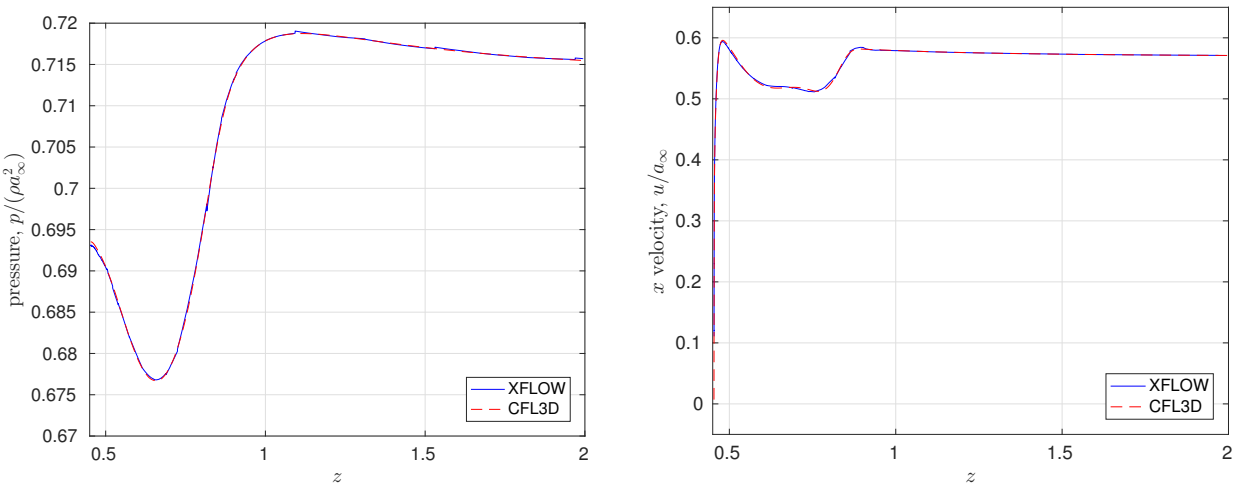
VI.A. Problem Description

This case consists of the ONERA M6 wing[20] in transonic turbulent flow. The specific conditions are a freestream Mach number of $M_\infty = 0.84$, angle of attack $\alpha = 3.06°$, and a Reynolds number based on the root chord of $Re_{c_{\text{root}}} = 14.6 \times 10^6$. The airfoil in the numerical simulation has been modified from that used in the experiment by splining a subset of the documented airfoil points, and by extending the trailing edge to make it sharp, as described in Appendix A. The Reynolds number is based on the root chord with the trailing-edge extension.

The farfield is a hemisphere that is placed approximately 50 root chord lengths away from the wing, and a full-state boundary condition is specified on the farfield. Using convenient units with $\rho_\infty = 1, |\vec{v}_\infty| = 1$, the conservative state vector takes the same form as in the previous example,

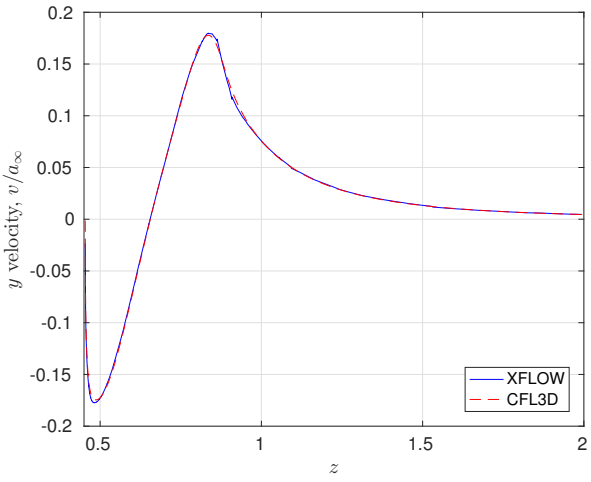
$$\mathbf{u}_\infty = \left[1, \cos\alpha, 0, \sin\alpha, \frac{1}{\gamma(\gamma-1)M^2}, \frac{3}{Re}\right]^T,$$

where $\gamma = 1.4$. The Sutherland's law ratio for computing the laminar viscosity is $T_s/T_{\text{ref}} = 110.33/300$.

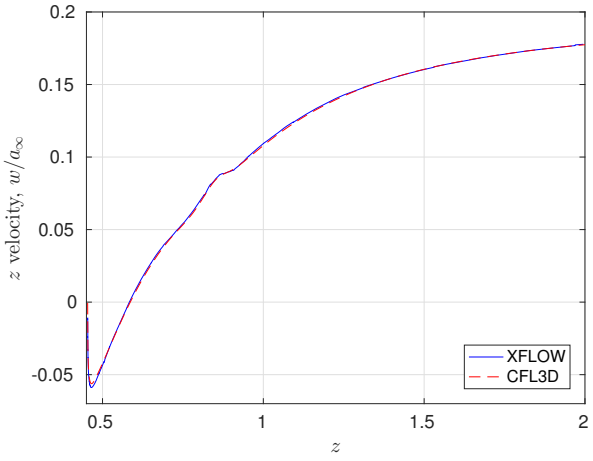


(a) pressure

(b) x velocity



(c) y velocity



(d) z velocity

Figure 12. Hemisphere cylinder: states on a vertical line at $x = 0.5$, $y = -0.21$. Comparison between the 14^{th} drag-adapted mesh and the finest-mesh CFL3D results.

The wing planform without the wingtip revolution is defined in the original reference by the parameters presented in Figure 13. The sharp trailing-edge modification extends the airfoil chord

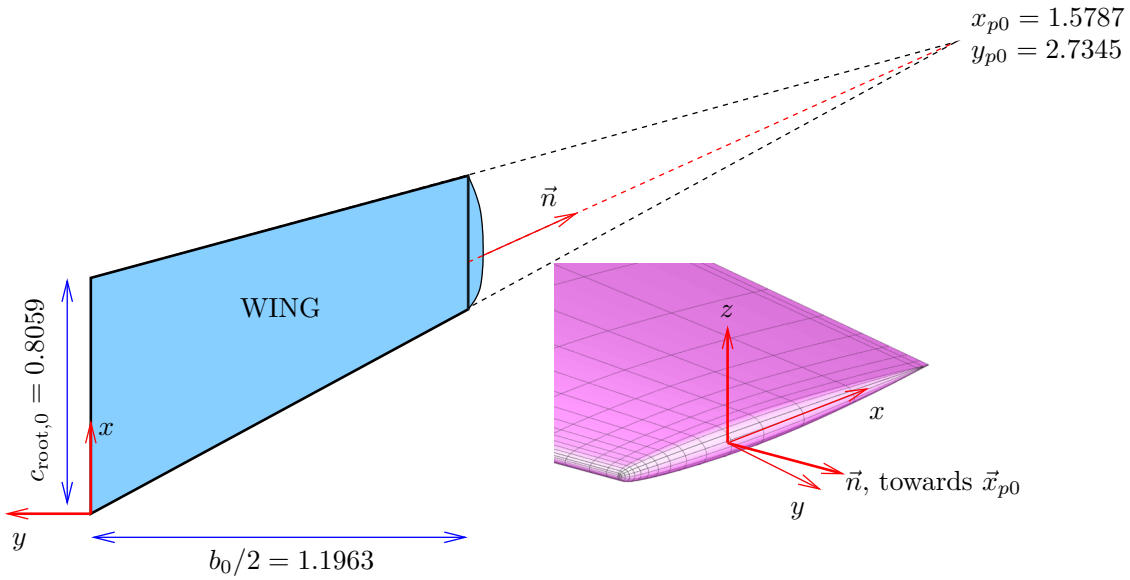


Figure 13. ONERA M6 planform definition and wingtip body of revolution.

by the factor x_{te} given in Appendix A. For the numerical simulation, the wing is re-normalized to have unit root chord by multiplying all coordinates by $f_{\text{scale}} = 1/(c_{\text{root},0} x_{\text{te}})$. We denote these scaled coordinates below by omitting the 0 subscript, e.g. $\vec{x}_p = \vec{x}_{p0} f_{\text{scale}}$.

The half-body of revolution at the wing tip is obtained by connecting the top of the airfoil to the bottom via a 180° arc. To avoid a non-smooth shoulder at the tip leading edge,[21] the rotation is not done in the $y - z$ plane, but rather in the $n - z$ plane, where the n coordinate is specific to each location on the tip and points in the direction of $\vec{x}_p$, i.e. where the sweeps meet. Figure 13 illustrates this definition and shows the rounded tip generated in this manner.

Snapping surface points, e.g. during adaptation, to the true geometry consists of the following steps. Let $\vec{x} = (x, y, z)$ be the original point. First, if $y \geq b/2$, we are on the main section – note, y is negative, as we are considering the left wing. The projection then proceeds as follows:

$x_{\text{le}} = -y x_p / y_p$	leading edge location
$c = (y_p + y)/y_p$	local chord
$x_c = (x - x_{\text{le}})/c$	position along local chord
$z_c = z/c$	scaled z coordinate
$(x_c, z_c) = \text{Project}(x_c, z_c)$	projection to spline
$x = x_{\text{le}} + c x_c, \;\; z = c z_c$	new coordinates

If instead $y < b/2$, we are on the body of revolution, and projection in this case proceeds as follows:

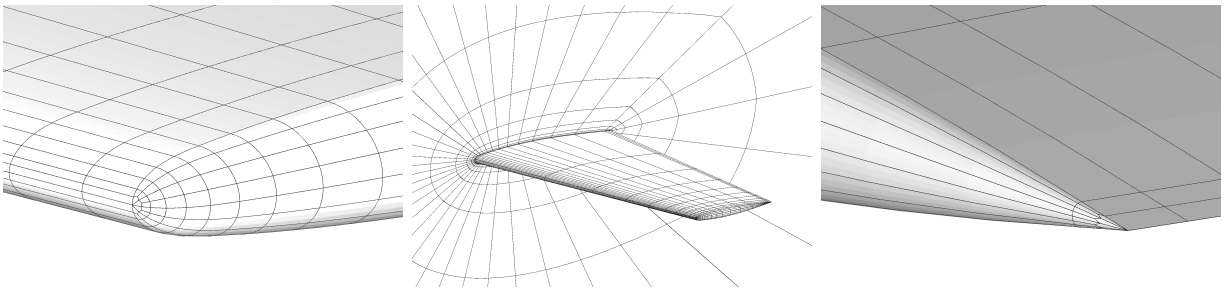
$x_{\text{le}} = -yx_p/y_p;$	leading edge location, past tip
$c = (y_p + y)/y_p$	local chord, past tip
$x_c = (x - x_{\text{le}})/c$	position along local chord
$\theta = \tan^{-1}((y_p + y)/(x_p - x))$	local sweep angle
$\Delta y = -h_b - y$	displacement in y
$\Delta x = \Delta y \tan(\theta)$	displacement in x
$g = \sqrt{\Delta x^2 + \Delta y^2}$	offset in x-y plane
$d = \sqrt{g^2 + z^2}$	distance from rotation axis
$\phi = \tan^{-1}(z/g)$	angle from vertical
$c = (y_p - b/2)/y_p$	tip chord
$x_c = x_c,\ z_c = d/c;$	local coordinates on airfoil
$(x_c, z_c) = \text{Project}(x_c, z_c)$	projection to spline
$d = z_c c$	new distance from rotation axis
$g = d\sin(\phi)$	new offset in the x-y plane
$\theta = \tan^{-1}(y_p/(x_p - x_c))$	new θ
$x_{\text{le}} = (b/2)x_p/y_p$	leading edge at the tip
$x = x_{\text{le}} + x_c c + g\sin(\theta)$	new x
$y = -b/2 - g\cos(\theta)$	new y
$z = d\cos(\phi)$	new z

Projection to the airfoil spline is performed by connecting the point in question to the spline with a segment perpendicular to the spline.

VI.B. Mesh Generation

The baseline mesh, shown in Figure 14, consists of 7936 hexahedral elements. This mesh is constructed by paving both the wing and the farfield with the same number and pattern of quadrilaterals. The spacing of the quadrilaterals is clustered towards the leading and trailing edges of the wing, and the spanwise spacing is clustered towards the wing tip. Eight equal intervals are used along the 180° arc of the wingtip body of revolution. As in the hemisphere case, we employ a special treatment of the wingtip leading and trailing edges: four skewed quadrilaterals are constructed by agglomerating pairs of adjacent triangles, as shown in Figure 14. This creates a mesh of all hexes for use in hanging-node adaptation.

The volume mesh is generated by connecting the quadrilaterals on the wing to those on the farfield, using a radial spacing that grows exponentially from the wing surface. The spacing of the first element off the wall is approximately .0001 at the leading edge (root chord is 1) and increases to twice that near the trailing edge. The mesh is first generated using all cubic elements, but as in the previous case, as many elements as possible are snapped to linear to improve efficiency. For the adaptive calculations, the baseline mesh is modified by uniformly refining all elements adjacent to the wing, and this yields an initial mesh of 11408 elements.



(a) Leading edge (b) Wing and symmetry (c) Trailing edge

Figure 14. ONERA M6 baseline mesh.

VI.C. Output Convergence

Starting from the initial mesh, adaptive runs were performed at order $p = 2$ for both the lift and the drag outputs. The lift and drag values were computed at each adaptive iteration, using a reference area of $S_{\text{ref}} = 1.15315084119231$ (in normalized units where the root chord is 1). Figure 15 shows the convergence of these outputs with adaptive refinement. Data from other codes is also included.

In the lift coefficient convergence plot, we see that the lift-adapted meshes converge more quickly than the drag-adapted meshes. Both are on a trajectory that is similar to the stabilized finite-element code. The final jump in the data is due to a lowering of the viscous jump stabilization constant on the final, more resolved mesh. This reduces the added dissipation, leading to larger lift values. The fact that the stabilization parameter still has a noticeable impact on the solution suggests that the meshes are not yet adequately resolved.

The drag convergence plot in Figure 15 shows that drag adaptation is, as expected, more effective at accurately predicting the drag compared to the lift adaptation. Both sets of results show improved efficiency compared to the other codes, with the benefits over stabilized finite elements as the smallest. The drop in the drag at the last adaptive iteration in each case is due to the lowering of the viscous stabilization parameter. We note that both adaptive runs yield drag numbers that are similar to those of the other codes.

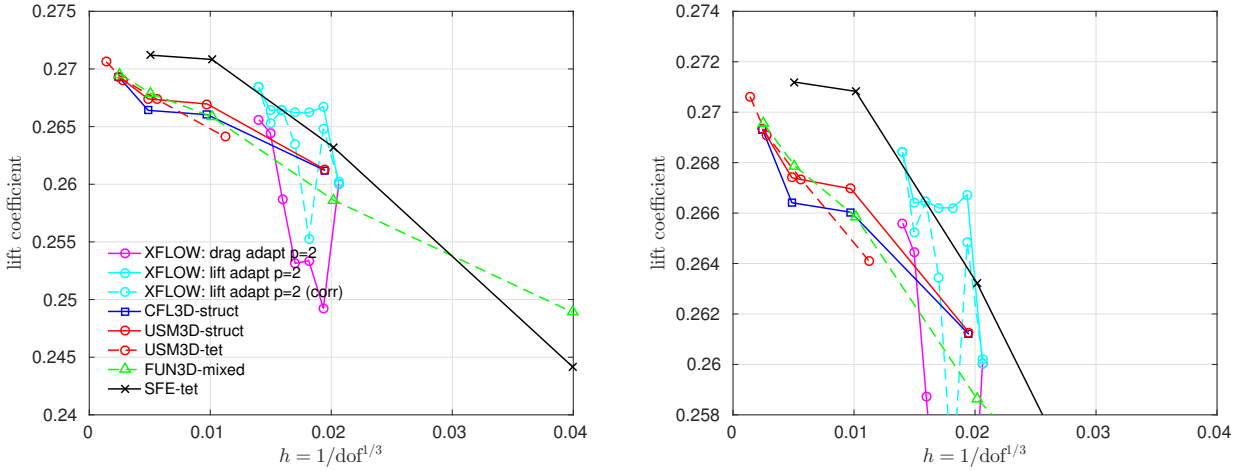
VI.D. Adapted Meshes and Field Plots

Figure 16 shows the pressure contours obtained on solutions using the output-adapted meshes. The shock structure is evident on the wing surface, and it is more resolved on the lift-adapted mesh, which is finer as it is further along in the adaptation. The resolution on and off the wing surface is not very high in the drag-adapted mesh, even though the outputs are converging reasonably well to a final value.

Figure 17 shows the corresponding adapted meshes. As expected, we observe refinement in the vicinity of the shocks, though this refinement does not extend very far off the wing surface. We also observe refinement at the leading edge of the wing, particularly along the stagnation streamline, and at the wing tip trailing edge, where the tip vortex forms. These areas, except possibly well in front of the leading edge, are consistent with expectations of where the important flow features are, and they are identified automatically.

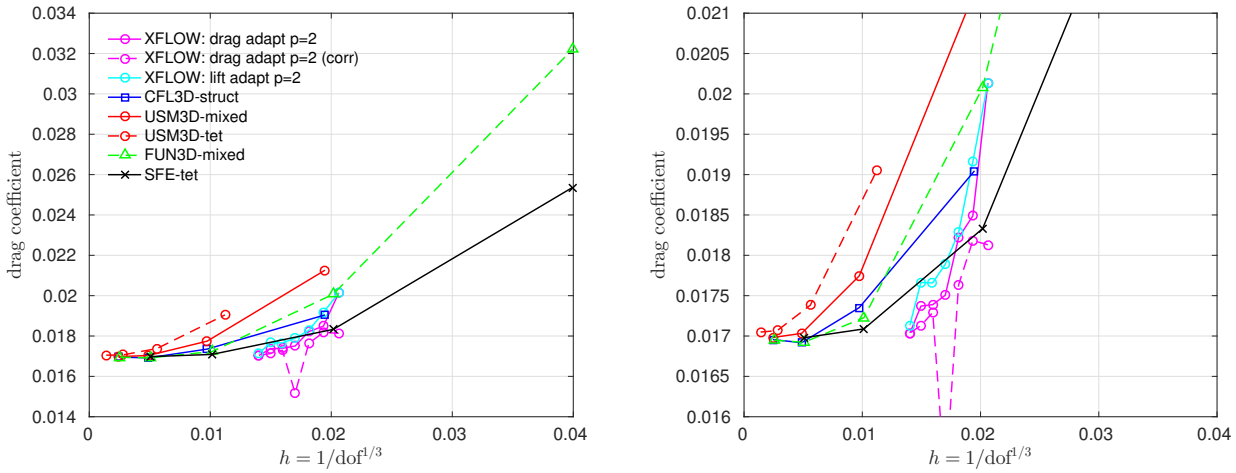
VII. Conclusions

This paper presents the results of adaptive simulations for Reynolds-averaged Navier-Stokes simulations of three-dimensional benchmark cases. The two cases consist of a hemisphere cylinder



(a) lift convergence

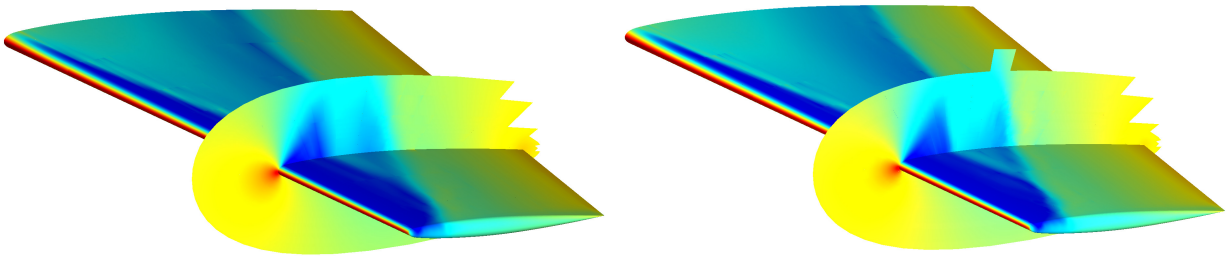
(b) lift convergence (zoom)



(c) drag convergence

(d) drag convergence (zoom)

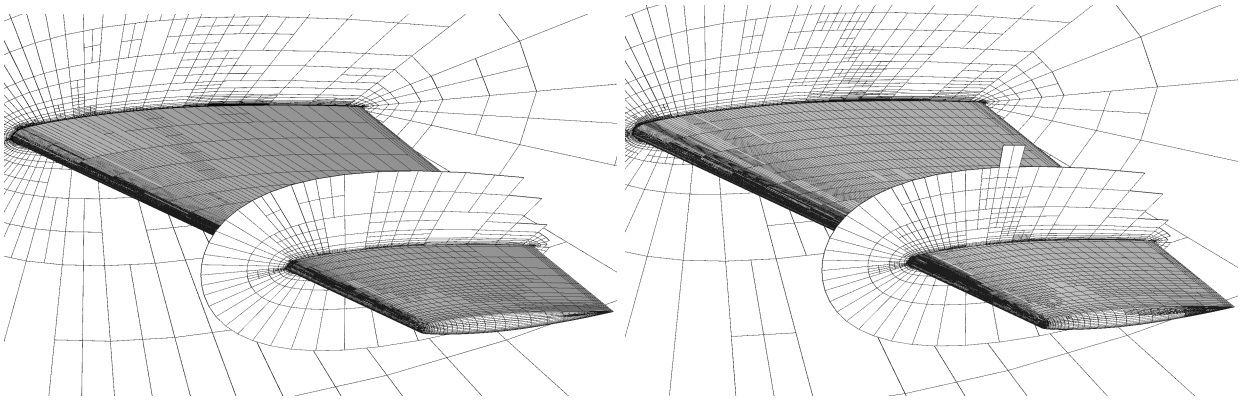
Figure 15. ONERA M6 wing: lift and drag convergence for $\alpha = 3.06°$.



(a) 5^{th} drag adaptation iteration

(b) 7^{th} lift adaptation iteration

Figure 16. ONERA M6: pressure contours on meshes adapted using the drag and lift output adjoints. The cut plane is at $y = -1$.



(a) 5^{th} drag adaptation iteration, 29869 elements (b) 7^{th} lift adaptation iteration, 43967 elements

Figure 17. ONERA M6: drag and lift adapted meshes

in subsonic flow at various angles of attack, and the ONERA M6 wing at a single angle of attack in transonic conditions. A discontinuous Galerkin finite-element discretization was applied to the Navier-Stokes equations closed with the Spalart-Allmaras turbulence model, using a negative turbulent viscosity correction. Initial unstructured hexahedral meshes for the adaptation were generated manually: these were curved using cubic approximation and of high anisotropy near the body surface to provide sufficient initial spacing to begin resolving the boundary layer. Output-based error estimation was applied to the solutions, and the resulting adjoint-weighted residual error indicator was used to drive fixed-fraction hanging-node adaptation.

The results for the hemisphere cylinder show very good agreement in the lift and drag coefficients compared to existing data from other codes. The path to visually converged values is rapid, requiring effective resolutions, h, that are 5-10 times larger than the h values from the other codes. This translates into two to three orders of magnitude fewer total degrees of freedom. The off-body state variation also shows good agreement versus data from the other codes.

The adaptive results for the ONERA M6 wing show convergence properties that are generally better or at least comparable to other codes. The adaptive lift coefficient convergence is similar to the convergence of a stabilized continuous finite element method, which is the most rapid of the other codes, whereas the drag convergence is improved by the adaptation. A distinguishing feature of this case is that shock stabilization is applied in the form of element-wise constant artificial viscosity, which introduces additional dissipation and pollutes the error estimates. More advanced shock capturing strategies could address some of these deficiencies and improve convergence.

This study demonstrates the benefits of applying adaptive high-order methods to Reynolds-averaged turbulent flow simulations in three dimensions. Comparisons to other codes show a more efficient use of degrees of freedom and more rapid convergence on coarse meshes. Directions for further research include: order refinement in an $h-p$ setting, fully-unstructured mesh adaptation, curving of meshes with highly-anisotropic elements, and solver robustness improvements.

Acknowledgements

The author acknowledges support from the Boeing Company, with technical monitor Dr. Mori Mani, and the Department of Energy under grant DE-FG02-13ER26146/DE-SC0010341.

A. ONERA M6 airfoil coordinates

Table 1 lists the original ONERA M6 coordinates,[22] upper surface only and normalized to unit chord, that were used as a starting point for the construction of the symmetric airfoil in the present study. Points 2, 39, 64, and 71 were removed to improve smoothness of the spline,[21] leaving 68 points. The trailing edge was then closed by linearly extending the last two (remaining) points, yielding a trailing edge location at

$$x_{\text{te}} = x_{72} + y_{72}(x_{72} - x_{70})/(y_{70} - y_{72}).$$

Finally, the entire set of 69 coordinates, x and y augmented with $(x_{\text{te}}, 0)$, was multiplied by $1/x_{\text{te}}$ to normalize the airfoil back to unit chord. These coordinates, with a symmetric mirroring for the lower surface, were then used as input in a cubic two-dimensional spline with zero second-derivative endpoint conditions at the trailing edge to fully define the airfoil section.

Table 1. Upper surface coordinates for the ONERA M6 airfoil. Removed points indicated via a strike-through.

	x	y		x	y		x	y
1	0.0000000	0.0000000	25	0.0793366	0.0338372	49	0.6752726	0.0349542
2	~~0.0000165~~	~~0.0006914~~	26	0.0956354	0.0357742	50	0.6993027	0.0329402
3	0.0000696	0.0014416	27	0.1149796	0.0377923	51	0.7231995	0.0308662
4	0.0001675	0.0022554	28	0.1378963	0.0398522	52	0.7469658	0.0287365
5	0.0003232	0.0031382	29	0.1649976	0.0419089	53	0.7705998	0.0265505
6	0.0005508	0.0040959	30	0.1919327	0.0436214	54	0.7941055	0.0243027
7	0.0008657	0.0051343	31	0.2187096	0.0450507	55	0.8174828	0.0219842
8	0.0012868	0.0062598	32	0.2453310	0.0462358	56	0.8407324	0.0195838
9	0.0018364	0.0074784	33	0.2717978	0.0471987	57	0.8638564	0.0170915
10	0.0025441	0.0087958	34	0.2981113	0.0479494	58	0.8868235	0.0145051
11	0.0034428	0.0102163	35	0.3242726	0.0484902	59	0.9061905	0.0122389
12	0.0045704	0.0117419	36	0.3502830	0.0488183	60	0.9225336	0.0102727
13	0.0059751	0.0133708	37	0.3761446	0.0489296	61	0.9363346	0.0085827
14	0.0077112	0.0150951	38	0.4018567	0.0488202	62	0.9479946	0.0071423
15	0.0098413	0.0168984	39	~~0.4274223~~	~~0.0484833~~	63	0.9578511	0.0059224
16	0.0124479	0.0187537	40	0.4528441	0.0479351	64	~~0.9661860~~	~~0.0048907~~
17	0.0156171	0.0206220	41	0.4781197	0.0471661	65	0.9732361	0.0040180
18	0.0194609	0.0224545	42	0.5032514	0.0461903	66	0.9792020	0.0032796
19	0.0241067	0.0242004	43	0.5282426	0.0450209	67	0.9842508	0.0026547
20	0.0297008	0.0258245	44	0.5530937	0.0436741	68	0.9885252	0.0021257
21	0.0364261	0.0273317	45	0.5778043	0.0421684	69	0.9921438	0.0016778
22	0.0444852	0.0287912	46	0.6023757	0.0405241	70	0.9952080	0.0012985
23	0.0541248	0.0303278	47	0.6268104	0.0387613	71	~~0.9978030~~	~~0.0009773~~
24	0.0656303	0.0320138	48	0.6511093	0.0368990	72	1.0000000	0.0007052

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]Fidkowski, K. J. and Ceze, M. A., "High-order output-based adaptive simulations of turbulent flow over a three dimensionsional bump," AIAA Paper 2015–0862, 2016.

[3]Becker, R. and Rannacher, R., "An optimal control approach to a posteriori error estimation in finite element methods," Acta Numerica, edited by A. Iserles, Cambridge University Press, 2001, pp. 1–102.

[4]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.

[5]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.

[6]Fidkowski, K. J. and Darmofal, D. L., "Review of output-based error estimation and mesh adaptation in computational fluid dynamics," American Institute of Aeronautics and Astronautics Journal, Vol. 49, No. 4, 2011, pp. 673–694.

[7]Ceze, M. A. and Fidkowski, K. J., "High-order output-based adaptive simulations of turbulent flow in two dimensions," AIAA Paper 2015–1532, 2015.

[8]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) 1902, 2012.

[9]Ceze, M. A. and Fidkowski, K. J., "An anisotropic hp-adaptation framework for functional prediction," American Institute of Aeronautics and Astronautics Journal, Vol. 51, 2013, pp. 492–509.

[10]Ceze, M. A. and Fidkowski, K. J., "Drag prediction using adaptive discontinuous finite elements," AIAA Journal of Aircraft, Vol. 51, No. 4, 2014, pp. 1284–1294.

[11]Roe, P., "Approximate Riemann solvers, parameter vectors, and difference schemes," Journal of Computational Physics, Vol. 43, 1981, pp. 357–372.

[12]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, 2002, pp. 197–207.

[13]Bassi, F. and Rebay, S., "High–order accurate discontinuous finite element solution of the 2-D Euler equations," Journal of Computational Physics, Vol. 138, 1997, pp. 251–285.

[14]Saad, Y. and Schultz, M. H., "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," SIAM Journal on Scientific Computing, Vol. 7, No. 3, 1986, pp. 856–869.

[15]Persson, P.-O. and Peraire., J., "Sub-cell shock capturing for discontinuous Galerkin methods," AIAA Paper 2006-112, 2006.

[16]Barter, G. and Darmofal, D., "Shock capturing with higher–order, PDE-based artificial viscosity." AIAA Paper 2007-3823, 2007.

[17]Oliver, T. A., A High–order, Adaptive, Discontinuous Galerkin, Finite Elemenet Method for the Reynolds-Averaged Navier-Stokes Equations, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2008.

[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]Rumsey, C., Smith, B., and Huang, G., "Turbulence modeling resource website," 2017, `https://turbmodels.larc.nasa.gov`.

[20]Barche, J., "Experimental data base for computer program assessment," Tech. Rep. 2, Advisory Group for Aerospace Research and Development, North Atlantic Treaty Organization, 1979, Report of the Fluid Dynamic Panel Working Group 04.

[21]Mayeur, J., Dumont, A., Desterac, D., and Gleize, V., "RANS simulations on TMR test cases and M6 wing with the Onera elsA flow solver," AIAA Paper 2015-1745, 2015.

[22]Slater, J. W., "ONERA M6 wing," 2015, `www.grc.nasa.gov/WWW/wind/valid/m6wing/m6wing.html`.