An anisotropic *hp*-adaptation framework for functional prediction

Marco Ceze[1] and Krzysztof J. Fidkowski[2]
Department of Aerospace Engineering,

University of Michigan, Ann Arbor, MI 48109

This paper presents a method for concurrent mesh and polynomial-order adaptation with the objective of direct minimization of output error using a selection process for choosing the optimal refinement option from a discrete set of choices that includes directional spatial resolution and approximation order increment. The scheme is geared towards compressible viscous aerodynamic flows, in which various solution features make certain refinement options more efficient compared to others. No attempt is made, however, to measure the solution anisotropy or smoothness directly or to incorporate it into the scheme. Rather, mesh anisotropy and approximation order distribution arise naturally from the optimization of a merit function that incorporates both an output sensitivity and a measure of the computational cost of solving on the new mesh. The method is applied to output-based adaptive simulations of the laminar and Reynolds-averaged compressible Navier-Stokes equations on body-fitted meshes in two and three dimensions. Two-dimensional results show significant reduction in the degrees of freedom and computational time to achieve output convergence when discrete choice optimization is used compared to uniform h or p refinement. Three-dimensional results show that the presented method is an affordable way of achieving output convergence on notoriously difficult cases such as the third Drag Prediction Workshop W-1 configuration.

---

[1] PhD. Candidate, Department of Aerospace Engineering, University of Michigan, 1320 Beal Ave., 48109, Ann Arbor, MI, USA, AIAA Member.
[2] Assistant Professor, Department of Aerospace Engineering, University of Michigan, 1320 Beal Ave., 48109, Ann Arbor, MI, USA, AIAA Member.

Nomenclature

$\boldsymbol{\psi}$ = Adjoint solution

α = Angle of attack

ϕ = Basis functions

b = Benefit measure

β = CFL amplification factor

ξ = CFL decrease factor

D = Computational domain

T^H = Computational mesh

$\mathcal{C},\ \mathcal{D}$ = Convective and dissipative fluxes

c_{DOF} = Cost measure based on number of degrees of freedom

c_{NZ} = Cost measure based on number of non-zero entries in the residual Jacobian

CFL = Courant-Friedrichs-Lewy number

$\mathbf{R}$ = Discrete residual vector

$\mathbf{U}$ = Discrete solution vector

η_{κ^H} = Elemental error indicator

Λ = Element aspect ratio

κ^H = Element of the mesh

$\mathbb{S}$ = Element surface area

$\mathbb{V}$ = Element volume

$\widehat{\cdot}$ = Flux-averaging operator

M_∞ = Free-stream Mach number

$\mathbf{M}$ = Mass matrix

$f_{\min}$ = Minimum under-relaxation factor

Re = Reynolds number

J = Scalar output of interest

$\mathcal{V}^{H,p}$ = Space of piecewise polynomials of degree p in the mesh T^H

f = Under-relaxation factor

$\mathbf{u}$ = Vector of conserved variables

$\mathbf{w}$ = Vector of weight functions

$\mathbb{C}$, $\mathbb{D}$ = Weak forms of the convective and dissipative fluxes

$\mathbb{R}$ = Weak form of the residual

I. Introduction

Improvements in computational power and numerical methods have solidified the foothold of Computational Fluid Dynamics (CFD) in the engineering community. CFD simulations boast rapid startup and turnaround times for new configurations, and they allow for test conditions that may be difficult to produce experimentally. Yet compared to experimental results, many CFD answers are still treated with skepticism, or at least caution. Arguably, such caution is well-founded, considering, for example, the spread of results for representative aerodynamic geometries in the AIAA drag prediction workshops [1–3]. Although the accuracy of CFD solutions is improving with increases in computational power, the grid sizes currently used to approach acceptable engineering solutions on representative geometries still prohibit CFD from being used as a high-fidelity design tool.

With the growth in complexity of CFD configurations, managing the liability of accurate solutions is no longer possible solely at the user level. A robust approach to managing this liability is through output-based error estimation, which has already been demonstrated for many complex problems, including those in aerospace applications [4–9]. The goal of these methods is to provide confidence measures in the form of error bars for scalar output quantities of engineering interest. Moreover, the theory behind output error estimation allows for the attribution of output error contributions to different elements or volumes of the computational mesh. The resulting adaptive indicator can then be used to drive mesh adaptation that specifically targets the output of interest [7, 8, 10–13]. Although output error estimation techniques require more computation, the resulting adapted meshes yield cost savings, in terms of mesh size and computational time for a given output accuracy, that generally more than offset the cost of the additional computation.

Aerodynamic flows exhibit features in a wide range of length scales and singular features whose distributions are not known *a priori*. Hence, it is key for an efficient computation that the adaptive algorithm is capable of generating stretched elements in areas where the solution exhibits anisotropy and choosing a local approximation order appropriate to the smoothness of the solution. Anisotropic

features include boundary layers, wakes, and shocks, where the disparity of length scales is such that stretching ratios in the hundreds or thousands are common. Singular or near-singular features, at least for the primal solution, include shocks, trailing edges, edges of boundary layers and wakes, and trailing vortices,

The choice between subdividing an element or locally changing the scheme's discretization order is not trivial and has been the subject of much previous research [14–19]. Bey [14] uses the error equidistribution principle to first subdivide elements and then increase the polynomial order where the solution is deemed smooth. Conversely, Heuveline and Rannacher [15] propose a process that prioritizes p-refinement and only subdivides an element when the previous step leads to an increase in the elemental error indicator. Houston and Süli [17] introduced two methods for assessing the local smoothness of the solution using Legendre series expansions and estimates of the local Sobolev index. In that same article, they also provide an overview of different strategies for the decision of the refinement type. Burgess and Mavriplis [19] use a solution-jump indicator to decide between h and p refinements. Following a different approach, Rachowicz *et al* [16] choose h or p refinement based on an estimated lowest interpolation error.

Anisotropic mesh adaptation in aerospace applications is also a prolific research topic. The dominant method for detecting anisotropy has relied on estimates of the directional interpolation error of a representative scalar, such as the Mach number [20, 21]. When used alone, this technique reduces to equidistributing the interpolation error of the chosen scalar over the computational domain, with the absolute level of interpolation error prescribed by the user [22, 23]. Alternately, this technique can be combined with output-based error estimation by using the output adaptive indicator to set the element size and the directional interpolation error to set the element stretching [7, 24]. The same idea can be extended to high-order discretizations [25, 26], although the measurement of directional interpolation error becomes more tedious. A more fundamental problem with this approach in the context of output-based adaptation is the assumption that mesh anisotropy should be governed by the directional interpolation error of one scalar quantity. This assumption is heuristic because it does not take into account the process by which interpolation errors create residuals that affect the output of interest. As a result, recent research has turned to adaptation algorithms that

directly target the output error.

Formaggia *et al* [27–30] combine Hessian-based interpolation error estimates with output-based *a posteriori* error analysis to arrive at an output-based error indicator that explicitly includes the anisotropy of each element. Schneider and Jimack [31] calculate the sensitivities of the output error estimate with respect to node positions and formulate an optimization problem to reduce the output error estimate by redistributing the nodes. They then combine this node repositioning with isotropic local mesh refinement sequentially in a hybrid optimization/adaptation algorithm. Park [32] introduces an algorithm that directly targets the output error through local mesh operators of element swapping, node movement, element collapse, and element splitting. Using the output error indicator to rank elements and nodes, these operations are performed in sequence and automatically result in mesh anisotropy.

Following a similar approach presented by Houston *et al* [33], we proposed in Ref. [34] a direct mesh optimization technique in which a particular mesh refinement is chosen from a discrete number of possible choices in a manner that directly targets reduction of the output error. That strategy is specifically suited for hanging-node meshes, in which a handful of refinement options is typically available for each element and in which the adaptation mechanics are relatively simple. Here, we extend our previous work to *hp*-adaptation of quadrilateral and hexahedral meshes. Although these meshes are more restrictive compared to general triangular and tetrahedral meshes, many body-fitted quadrilateral and hexahedral meshes already exist as these are the predominant element types for high-Reynolds number viscous flows. As such, our goal is to apply the direct optimization technique to practical aerodynamic flows to gauge the importance of anisotropy and higher-order solution approximation in output-driven mesh adaptation.

The structure of this paper is as follows. In Sections II and III we review the Navier-Stokes equations and the Spalart-Allmaras turbulence model. Section IV describe the discontinuous Galerkin discretization used in this work followed by the time integration method in Section V. Section VI presents the output-based error estimation framework that drives the mesh adaptation described in Section VII. In Section VIII, we describe the objective function used to rank the refinement options. We present results for inviscid, laminar and turbulent flows in Section IX and conclude in

Section X.

II. Compressible Navier-Stokes Equations

In this section, we present the flow equations and the assumptions used in our results. Consider the Navier-Stokes equations without source terms in conservative form,

$$\partial_t \mathrm{u}_s + \partial_i \mathcal{C}_{is}(\mathbf{u}) - \partial_i \mathcal{D}_{is}(\mathbf{u}) = 0, \tag{1}$$

where $i \in [1, .., \mathrm{dim}]$ indexes the spatial dimensions and s indexes the equations of conservation of mass, momentum, and energy. Accordingly, the state vector is denoted by $\mathbf{u} = [\rho, \rho v_i, \rho E]^T$, where ρ is the density, v_i are the spatial components of the velocity and E is the specific total energy. In Eqn. 1, the convective and diffusive fluxes are denoted by $\mathcal{C}$ and $\mathcal{D}$ respectively. These terms correspond to the following conservation statements:

- Conservation of mass, $s = 1$:

$$\mathcal{C}_{i1} = \rho v_i, \quad \mathcal{D}_{i1} = 0. \tag{2}$$

- Conservation of momentum, $s = 2 \rightarrow$dim+1:

$$\mathcal{C}_{is} = \rho v_{s-1} v_i + \delta_{i\ (s-1)} p, \quad \mathcal{D}_{is} = \tau_{i\ (s-1)}. \tag{3}$$

- Conservation of energy, $s = \mathrm{dim} + 2$:

$$\mathcal{C}_{is} = \rho v_i H, \quad \mathcal{D}_{is} = \kappa_T \partial_i T + v_j \tau_{ij}. \tag{4}$$

Note, δ_{ij} is the Kronecker delta symbol. For inviscid calculations, the physical diffusion term $\mathcal{D}$ is not included in the equation set. In viscous calculations, we consider Newtonian fluids for which the viscous stress tensor is given by

$$\tau_{ij} = \mu(\partial_i v_j + \partial_j v_i) + \lambda \delta_{ij} \partial_k v_k, \tag{5}$$

where μ and λ are the dynamic and bulk viscosities, respectively. Furthermore, we assume a calorically and thermally perfect gas to close the system in Eqn. 1. This allows us to relate the

pressure, p, temperature, T and specific total enthalpy, H, to the conserved variables as follows:

$$p = (\gamma - 1)\left(\rho E - \rho\frac{v_i v_i}{2}\right), \tag{6}$$

$$H = E + \frac{p}{\rho}, \tag{7}$$

$$T = \frac{p}{R\rho}. \tag{8}$$

For all results presented in this article, we use the physical properties of air:

$$\text{Dynamic viscosity: } \mu = \mu_{\text{ref}}\left(\frac{T}{T_{\text{ref}}}\right)^{1.5}\left(\frac{T_{\text{ref}} - T_{\text{s}}}{T + T_{\text{s}}}\right),$$

$$\text{(Sutherland's law: } T_{\text{ref}} = 288.15K,\ T_{\text{s}} = 110K)$$

$$\text{Bulk viscosity coefficient: } \lambda = -\frac{2}{3}\mu,$$

$$\text{Thermal conductivity: } \kappa_T = \frac{\lambda\mu R}{(\gamma - 1)Pr},$$

$$\text{Specific-heat ratio: } \gamma = 1.4,$$

$$\text{Prandtl number: } Pr = 0.71,$$

where R is the gas constant.

III. Spalart-Allmaras Turbulence Model

In this work we use the one-equation Spalart-Allmaras (SA) turbulence model [35]. All turbulent cases presented in this paper are at sufficiently-high Reynolds number such that it is reasonable to assume them to be fully turbulent. Therefore we do not include trip terms in the model. In addition, we adopt the modifications proposed by Oliver and Allmaras [36] in order to improve robustness when using a high-order discontinuous Galerkin discretization. The PDE for the quantity $\rho\tilde{\nu}$ is written in conservation form as

$$\partial_t(\rho\tilde{\nu}) + \partial_i\mathcal{C}_i^{(\text{SA})}(\rho\tilde{\nu}) - \partial_i\mathcal{D}_i^{(\text{SA})}(\rho\tilde{\nu}) = \mathcal{S}^{(\text{SA})}(\rho\tilde{\nu}), \tag{9}$$

where $\tilde{\nu}$ is related to the kinematic viscosity through the ratio $\chi = \tilde{\nu}/\nu$. The convective and diffusive fluxes are respectively given by

$$\mathcal{C}_i^{(\text{SA})}(\rho\tilde{\nu}) = \rho v_i\tilde{\nu}, \quad \mathcal{D}_i^{(\text{SA})}(\rho\tilde{\nu}) = \sigma^{-1}(\mu + \rho\tilde{\nu})\partial_i\tilde{\nu}, \tag{10}$$

and the source term is given by a balance of production, distribution and destruction terms,

$$\mathcal{S}^{(\text{SA})}(\rho\tilde{\nu}) = \mathcal{P}^{(\text{SA})}(\rho\tilde{\nu}) + \mathcal{B}^{(\text{SA})}(\rho\tilde{\nu}) - \mathcal{T}^{(\text{SA})}(\rho\tilde{\nu}). \tag{11}$$

The distribution term remains unaltered from the original model,

$$\mathcal{B}^{(\mathrm{SA})}(\rho\tilde{\nu}) = \sigma^{-1} c_{b2} \rho \partial_j \tilde{\nu} \partial_j \tilde{\nu}. \tag{12}$$

The production and destruction terms are modified according to Ref. [36] to ensure stability of the magnitude of $\tilde{\nu}$.

The modified production term is given by

$$\mathcal{P}^{(\mathrm{SA})}(\rho\tilde{\nu}) = \begin{cases} c_{b1}\tilde{s}\rho\tilde{\nu} & \chi \geq 0, \\ c_{b1}\tilde{s}\rho\tilde{\nu} g_n(\chi) & \chi < 0, \end{cases} \qquad \tilde{s} = \begin{cases} |\omega| + \bar{s} & \bar{s} \geq -c_{v2}|\omega|, \\ |\omega| + \dfrac{|\omega|(c_{v2}^2|\omega| + c_{v3}\bar{s})}{(c_{v3} - 2c_{v2})|\omega| - \bar{s}} & \bar{s} < -c_{v2}|\omega|, \end{cases} \tag{13}$$

where $|\omega| = \sqrt{2\Omega_{ij}\Omega_{ij}}$ is the vorticity magnitude and the function g_n provides C^1-continuity to $\mathcal{P}^{(\mathrm{SA})}$ at $\tilde{\nu} = 0$,

$$g_n(\chi) = 1 - \frac{10^3 \chi}{1 + \chi^2}. \tag{14}$$

The modified destruction term is

$$\mathcal{T}^{(\mathrm{SA})}(\rho\tilde{\nu}) = \begin{cases} c_{w1} f_w \dfrac{\rho\tilde{\nu}^2}{d_w^2} & \chi \geq 0, \\ -c_{w1} \dfrac{\rho\tilde{\nu}^2}{d_w^2} & \chi < 0, \end{cases} \tag{15}$$

where d_w is distance to the nearest wall and the wall function f_w is given by

$$f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{\frac{1}{6}}, \quad g = r + c_{w2}(r^6 - r), \quad \text{and} \quad r = \frac{\tilde{\nu}}{\tilde{s}\kappa^2 d_w^2}. \tag{16}$$

The closure functions are

$$\bar{s} = \frac{\tilde{\nu} f_{v2}}{\kappa^2 d_w^2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \tag{17}$$

and the coefficients for the SA model are given in Table 2.

Finally, the SA equation is coupled with the Navier-Stokes system through the diffusion term in the momentum equation. We use Boussinesq's assumption and augment the viscous stress tensor in Eqn. 5 with the eddy viscosity μ_t as follows:

$$\tau_{ij} = (\mu + \mu_t)(\partial_i v_j + \partial_j v_i) + \lambda \delta_{ij} \partial_k v_k, \tag{18}$$

where μ_t is also modified to

$$\mu_t = \begin{cases} \rho\tilde{\nu} f_{v1} & \tilde{\nu} > 0 \\ 0 & \tilde{\nu} \leq 0. \end{cases} \tag{19}$$

Table 2 Spalart-Allmaras turbulence model closure parameters.

Parameter	Value
c_{b1}	0.1355
c_{b2}	0.622
σ	2/3
κ	0.41
c_{w1}	$c_{b1}/\kappa^2 + (1 + c_{b2})/\sigma$
c_{w2}	0.3
c_{w3}	2.0
c_{v1}	7.1
c_{v2}	0.7
c_{v3}	0.9

IV. Discontinuous Galerkin Spatial Discretization

In this section, we describe the discontinuous Galerkin (DG) spatial discretization of the flow equations. Let $\mathcal{V}^{H,p}$ be the space of piecewise polynomials of degree p with local support on each element $\kappa^H \in T^H$, where T^H is the set of elements resulting from a non-overlapping discretization of the domain, D. Using the method of weighted residuals, the steady-state problem statement becomes: find $\mathbf{u}^{H,p} \in \mathcal{V}^{H,p}$ such that $\forall \mathbf{w}^{H,p} \in \mathcal{V}^{H,p}$,

$$\mathbb{R}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p}) = \sum_{\kappa^H \in T^H} \mathbb{C}_{\kappa^H}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p}) - \mathbb{D}_{\kappa^H}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p}) = 0, \tag{20}$$

where $\mathrm{w}_s^{H,p}$ and $\mathrm{u}_s^{H,p}$ are piecewise polynomials that reside in $\mathcal{V}^{H,p}$ and s is a local index in each element that corresponds to the conserved state components. $\mathbb{C}$ and $\mathbb{D}$ respectively correspond to the convective and diffusive terms.

The convective term is obtained using integration by parts over each element and invoking Gauss's theorem,

$$\mathbb{C}_{\kappa^H}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p}) = -\int_{\kappa^H} \partial_i \mathrm{w}_s^{H,p} \mathcal{C}_{is}(\mathbf{u}^{H,p}) d\mathbf{x} + \int_{\partial\kappa^H} \mathrm{w}_s^{(H,p)+} \widehat{\mathcal{C}}_{is}(\mathbf{u}^{(H,p)+}, \mathbf{u}^{(H,p)-}, \mathbf{n})\, ds. \tag{21}$$

The superscripts + and − respectively indicate values corresponding to the interior and exterior of element κ^H on the boundary $\partial\kappa^H$. Also, $\widehat{\mathcal{C}}_{is}$ is the Riemann flux computed for the two states in

parentheses and $\mathbf{n}$ is a vector normal to the boundary pointing out of element κ^H. Roe's [37] approximate Riemann solver, augmented with fully-coupled conserved scalar transport of the turbulent working variable, is used for $\widehat{\mathcal{C}}_{is}$.

The second form of Bassi & Rebay [38] (BR2) is used to discretize the diffusion term. In this form, the steady-state Navier-Stokes equations are treated as a system of first order equations by exploiting the linear dependence of the diffusion term with respect to the spatial gradient of the state,

$$\mathcal{D}_{is}(\mathbf{u}) = \mathcal{A}_{isjk}(\mathbf{u})\partial_j \mathrm{u}_k, \tag{22}$$

where the tensor $\mathcal{A}_{isjk}$ is a nonlinear function of the state vector. Note, i, j index the spatial dimension and s, k index the state vector. Using integration by parts and Gauss's theorem yields the weak form of the diffusion term,

$$\begin{aligned}\mathbb{D}_{\kappa^H}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p}) = & -\int_{\kappa^H} \partial_i \mathrm{w}_s^{H,p} \mathcal{A}_{isjk}(\mathbf{u}^{H,p}) \partial_j \mathrm{u}_k^{H,p} d\mathbf{x} + \int_{\partial\kappa^H} \partial_i \mathrm{w}_s^{(H,p)+} \mathcal{A}_{isjk}(\mathbf{u}^{(H,p)+}) \mathrm{u}_k^{(H,p)+} \mathrm{n}_j \; ds \\ & - \int_{\partial\kappa^H} \partial_i \mathrm{w}_s^{(H,p)+} \widehat{\mathcal{A}_{isjk} \mathrm{u}_k^{H,p}} \mathrm{n}_j \; ds + \int_{\partial\kappa^H} \mathrm{w}_s^{(H,p)+} \widehat{\mathcal{D}}_{is} \mathrm{n}_i \; ds,\end{aligned} \tag{23}$$

where $\widehat{\cdot}$ indicates flux averaging of discontinuous quantities and $\widehat{\mathcal{D}}_{is}$ includes jump stabilization terms. For the conservation of mass, momentum, and energy, the choice of flux averages is compact, primal and dual-consistent as presented in Ref. [25]. We use Oliver's [36] dual-inconsistent formulation for the SA model due to its simpler implementation and similar output-adapted results when compared to dual-consistent formulations which introduce additional terms into $\mathbb{R}(\mathbf{u}^{H,p}, \mathbf{w}^{H,p})$.

In each element, the state $\mathrm{u}_s^{H,p}$ and the weight functions $\mathrm{w}_s^{H,p}$ are expanded in terms of the basis functions $\phi_b^{H,p}$ as follows:

$$\mathrm{u}_s^{H,p} = \mathrm{U}_{sb}\phi_b^{H,p}(\mathbf{x}), \quad \mathrm{w}_s^{H,p} = \mathrm{W}_{sb}\phi_b^{H,p}(\mathbf{x}), \tag{24}$$

where U_{sb} is the unknown discrete state. Note that the number of unknowns per element is $N_b \times N_s$, where N_b is the number of basis functions in the element and N_s is the number of components in the conserved state vector. The discrete residual operator is obtained by substituting the expressions in Eqn. 24 into the semilinear form of Eqn. 20 and choosing, in each element, W_{sb} to have the value of 1 for each combination of state and basis components. Finally, the semi-discrete flow equations

are written as:

$$\mathbf{M}\partial_t \mathbf{U} = -\mathbf{R}(\mathbf{U}), \tag{25}$$

where $\mathbf{R}$ is the discrete residual operator and $\mathbf{M}$ is the block diagonal mass matrix that corresponds to the volume integral of basis function products on each element in the mesh. In the interest of notation, we will refer to the discrete residual and state as vectors that correspond to unrolling U_{sb},

$$\mathrm{U}_l \Leftarrow \mathrm{U}_{sb}\mathrm{V}_{sbl}, \tag{26}$$

where V_{sbl} is a bookkeeping tensor that encodes the unrolling. To improve robustness of the high-order DG solver for complex flows, we adopt the physical realizability constraint handling technique proposed in Ref. [39]. Additionally, in some of the results presented here, artificial dissipation[40] is used for capturing shocks and improving stability.

V. Time Integration

Since we are interested in the steady-state solution of the flow equations, high-accuracy is not required for discretizing the unsteady term of Eqn. 25. Instead, stability is the main attribute which makes backward Euler an attractive choice. The fully discrete form of Eqn. 25 is then

$$\mathbf{M}\frac{1}{\Delta t}(\mathbf{U}^{n+1} - \mathbf{U}^n) + \mathbf{R}(\mathbf{U}^{n+1}) = 0, \tag{27}$$

where n indexes the time nodes.

In time-accurate calculations, Eqn. 27 is solved for the future state using a nonlinear solver such as Newton-Raphson. For steady calculations, the residual at the future state in Eqn. 27 is linearized about the current state and the solution update $\Delta\mathbf{U}^k = \mathbf{U}^{k+1} - \mathbf{U}^k$ is obtained by solving the following linear system:

$$\left(\mathbf{M}\frac{1}{\Delta t} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Big|_{\mathbf{U}^k}\right)\Delta\mathbf{U}^k = -\mathbf{R}(\mathbf{U}^k), \tag{28}$$

where k is used for the iteration number to distinguish the method from the time-accurate backward Euler case. In principle, Newton's root-finding method can be used to solve $\mathbf{R}(\mathbf{U}) = 0$ directly. However, the time discretization term is kept in Eqn. 28 to alleviate the spectral conditioning of the linear systems in the initial stages of the calculation and to improve the global convergence property

of the solver. The Generalized Minimal Residual (GMRES) algorithm with an element-line-Jacobi [41] preconditioner is used as the linear solver. Note that for $\Delta t \to \infty$ the iterative procedure of Eqn. 28 reduces to Newton's root-finding method.

In the first stages of calculations initialized by states that do not satisfy all boundary conditions, strong transients are observed due to the propagation of boundary information into the domain. To alleviate adverse effects of these transients and to avoid robustness problems, small time steps are used in an attempt to make the solution follow a physical path. This causes a diagonal dominance in the coefficient matrix in Eqn. 28 and makes the calculation closer to time-accurate if Δt does not vary spatially. As an alternative to global time stepping, element-wise time steps are used by setting a global CFL number defined as

$$\mathrm{CFL} = \frac{\lambda_{\max} \Delta t}{L_e}, \tag{29}$$

where $\lambda_{\max}$ is the maximum wave speed and L_e is a measure of element size, *e.g.* hydraulic diameter.

At each Newton iteration, the flow state vector $\mathbf{U}^k$ is updated with $\Delta \mathbf{U}^k$. For robustness purposes, an under-relaxation parameter is used to keep the solution physically realizable,

$$\mathbf{U}^{k+1} = \mathbf{U}^k + f^k \Delta \mathbf{U}^k. \tag{30}$$

The relaxation factor f^k is limited such that the changes in pressure and density at selected limit points of the interpolated field $\mathbf{u}^H(t, \mathbf{x})$ are within a fraction, $\eta_{\max}$, of the current values. Based on the relaxation factor, an exponential progression CFL evolution strategy is used. In this strategy, the CFL increases by a factor $\beta > 1$ if a full update ($f = 1$) happened in the previous step of the solver. On the other end, if $f < f_{\min}$ the CFL is reduced by multiplying it by $\xi < 1$ and the solver step is repeated. This strategy is summarized below:

$$\mathrm{CFL}^{k+1} = \begin{cases} \beta \cdot \mathrm{CFL}^k \quad \text{for } \beta > 1 & \text{if} \quad f^k = 1 \\ \mathrm{CFL}^k & \text{if} \quad f_{\min} < f^k < 1 \\ \xi \cdot \mathrm{CFL}^k \quad \text{for } \xi < 1 & \text{if} \quad f^k < f_{\min} \end{cases} . \tag{31}$$

The following values were used in this paper: $\mathrm{CFL}^0 = 1.0$, $\beta = 1.5$, $\xi = 0.1$, $f_{\min} = 0.01$, and $\eta_{\max} = 0.1$.

VI. Output Error Estimation

Output-based error estimation techniques identify all areas of the domain that are important for the accurate prediction of an engineering output. The resulting estimates properly account for error propagation effects that are inherent to hyperbolic problems, and they can be used to ascribe confidence levels to outputs or to drive adaptation. A key component of output error estimation is the solution of an adjoint equation for the output of interest. In a continuous setting, an adjoint, $\boldsymbol{\psi} \in \mathcal{V}$, is a Green's function that relates residual source perturbations to a scalar output of interest, $J(\mathbf{u})$, where $\mathbf{u} \in \mathcal{V}$ denotes the state, and where $\mathcal{V}$ is an appropriate function space. Specifically, given a variational formulation of a partial differential equation: determine $\mathbf{u} \in \mathcal{V}$ such that

$$\mathbb{R}(\mathbf{u}, \mathbf{w}) = 0, \qquad \forall \mathbf{w} \in \mathcal{V}, \tag{32}$$

the adjoint $\boldsymbol{\psi} \in \mathcal{V}$ is the sensitivity of J to an infinitesimal source term $\delta\mathbf{r} \in \mathcal{V}$ added to the left-hand side of the original PDE. $\boldsymbol{\psi}$ satisfies a linear equation,

$$\mathbb{R}'[\mathbf{u}](\mathbf{w}, \boldsymbol{\psi}) + J'[\mathbf{u}](\mathbf{w}) = 0, \qquad \forall \mathbf{w} \in \mathcal{V}, \tag{33}$$

where the primes denote Fréchét linearization with respect to the arguments in square brackets. Details on the derivation of the adjoint equation can be found in many sources, including the review in Ref. [42]. Specifically, in the present work we employ the discrete adjoint method, in which the system is derived systematically from the primal system [43, 44].

An adjoint solution can be used to estimate the numerical error in the corresponding output of interest. The resulting adjoint-weighted residual method is based on the observation that a solution $\mathbf{u}^{H,p}$ in a finite-dimensional approximation space $\mathcal{V}^{H,p}$ will generally not satisfy the original PDE. The adjoint $\boldsymbol{\psi} \in \mathcal{V}$ translates the residual perturbation to an output perturbation via,

$$\delta J = J(\mathbf{u}^{H,p}) - J(\mathbf{u}) \approx -\mathbb{R}(\mathbf{u}^{H,p}, \boldsymbol{\psi}). \tag{34}$$

This expression is based on a linear analysis, and hence for nonlinear problems and finite-size perturbations, the result is approximate.

Although the continuous solution $\mathbf{u}$ is not required directly, the continuous adjoint $\boldsymbol{\psi}$ must be approximated to make the error estimate in Eqn. 34 computable. In practice, $\boldsymbol{\psi}^{h,p^+}$ is solved

approximately or exactly on a finer finite-dimensional space $\mathcal{V}^{h,p^+} \supset \mathcal{V}^{H,p}$ [45–47]. This finer space can be obtained either through mesh subdivision or approximation order increase [36, 48, 49] – denoted here by changes in the superscript H and p, respectively.

The adjoint-weighted residual evaluation in Eqn. 34 can be localized to yield an adaptive indicator consisting of the relative contribution of each element to the total output error. In this work, the finer space is obtained by approximation order increment, $\mathcal{V}^{H,p+1} \supset \mathcal{V}^{H,p}$, and $\boldsymbol{\psi}^{H,p+1}$ is approximated by injecting $\boldsymbol{\psi}^{H,p}$ into $\mathcal{V}^{H,p+1}$ and applying 5 element block-Jacobi smoothing iterations. The output perturbation in Eqn. 34 is approximated as

$$\delta J \approx -\sum_{\kappa^H \in T^H} \mathbb{R}_{\kappa^H}(\mathbf{I}_{H,p}^{H,p+1}(\mathbf{u}^{H,p}), \boldsymbol{\psi}^{H,p+1} - \mathbf{I}_{H,p}^{H,p+1}(\boldsymbol{\psi}^{H,p})), \tag{35}$$

where $\mathbf{I}_{H,p}^{H,p+1}(\cdot)$ is an injection operator from p to $p+1$ in the coarse mesh T^H, and $\mathbb{R}_{\kappa^H}$ corresponds to the elemental residual as defined in Eqn. 20. Note, the difference between the coarse-space and fine-space adjoints is not strictly necessary due to Galerkin orthogonality [42]. However, when the primal residual is not fully-converged to machine precision levels the use of the adjoint perturbation gives better error estimates. Equation 35 expresses the output error in terms of contributions from each coarse element. A common approach for obtaining an adaptive indicator is to take the absolute value of the elemental contribution in Eqn. 35 [4, 6, 34, 46, 50, 51],

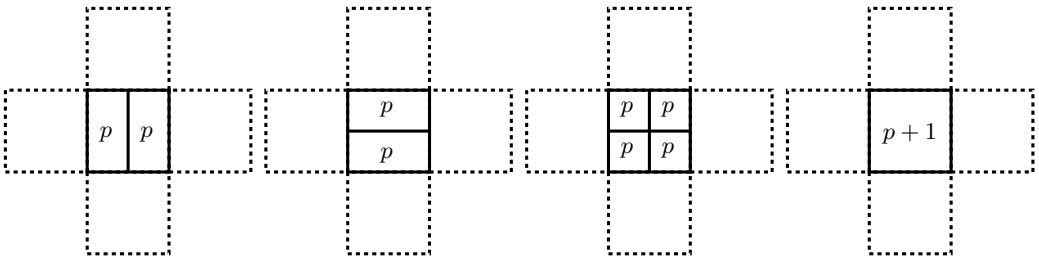
$$\eta_{\kappa^H} = \left|\mathbb{R}_{\kappa^H}(\mathbf{I}_{H,p}^{H,p+1}(\mathbf{u}^{H,p}), \boldsymbol{\psi}^{H,p+1} - \mathbf{I}_{H,p}^{H,p+1}(\boldsymbol{\psi}^{H,p}))\right|. \tag{36}$$

With systems of equations, indicators are computed separately for each equation and summed together. Due to the absolute values, the sum of the indicators, $\sum_{\kappa^H} \eta_{\kappa^H}$, is greater or equal to the original output error estimate. However, it is not a bound on the actual error because of the approximations made in the derivation.

VII. Mesh Adaptation Mechanics

The elemental adaptive indicator, η_{κ^H}, drives a fixed-fraction hanging-node adaptation strategy. In this strategy, which was chosen for simplicity and predictability of the adaptive algorithm, a certain fraction, f^{adapt}, of the elements with the largest values of η_{κ^H} is marked for refinement. Marked elements are refined according to discrete options which correspond to subdividing the element in

different directions or increasing the approximation order. For quadrilaterals, the discrete options are: x-refinement, y-refinement, xy-refinement and p-increment, as depicted in Figure 1. Note, x and y refer to reference-space coordinates of elements that can be arbitrarily oriented and curved in physical space. Also, the subelements created through refinement inherit the approximation order from the original element. In three dimensions a hexahedron can be refined in eight ways: three single-plane cuts, three double-plane cuts, isotropic refinement, and p increment.



(a) x-refinement (b) y-refinement (c) xy-refinement (d) p-refinement

Fig. 1 Quadrilateral refinement options. The dashed lines indicate the neighbors of the refined element.

h-refinement is performed in an element's reference space by employing the coarse element's reference-to-global coordinate mapping in calculating the refined element's geometry node coordinates. The refined elements inherit the same geometry approximation order and quadrature rules as the parent coarse element. As a result, there is no loss of element quality when a nonlinear mapping is used to fit the element to a curved geometry. Therefore, curved elements near a boundary can be efficiently refined to capture boundary layers in viscous flow. For simplicity of implementation, the initial mesh is assumed to capture the geometry sufficiently well, through a high enough order of geometry interpolation on curved boundaries, such that no additional geometry information is used throughout the refinements. That is, refinement of elements on the geometry boundary does not change the geometry. We note that for highly-anisotropic meshes, curved elements may be required away from the boundary, and for simplicity we use meshes with curved elements throughout the domain.

Note that elements created in a hanging-node refinement can be marked for h-refinement again in subsequent adaptation iterations. In this case, neighbors will be cut to keep one level of refinement

difference between adjacent cells. This is illustrated in Figure 2.

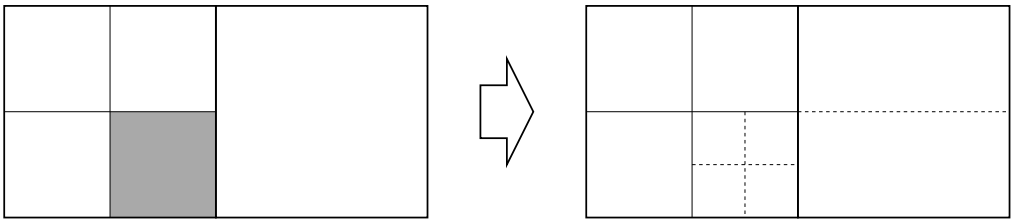


Fig. 2 Hanging-node adaptation for a quadrilateral mesh, with a maximum of one level of refinement separating two elements. The shaded element on the left is marked for refinement, and the dashed lines on the right indicate the additional new edges formed.

VIII. Merit Function

The choice of a particular refinement option is made locally in each element flagged for refinement. This choice is made by defining a merit function $m(i)$ that ranks each available refinement option i. This function is defined as

$$m(i) = \frac{b(i)}{c(i)} \,, \tag{37}$$

where b and c respectively correspond to measures of the benefit and the computational cost of the refinement option indexed by i. These measures depend on the method used for solving the flow equations and they should be tailored for each specific solver. We define them further in this section in the context of the applications presented in this paper.

During calculation of the merit function, local mesh and data structures are created that include the flagged element and its first-level neighbors along with the corresponding primal and adjoint states. In these local structures, the central element is refined in turn according to each of the discrete options. On the refined local mesh, the merit function is computed and the refinement option with the largest value of $m(i)$ is chosen.

The method for selecting a refinement option presented in this paper is similar to that presented by Houston *et al* [33] for quadrilateral meshes. These authors employ a heuristic that consists of the sum of the subelement error indicators computed for each refinement option, and a ratio, θ, of the maximum to minimum sum is used to make the decision of adapting isotropically or in one direction.

Anisotropy is only deemed important when θ is larger than a user-prescribed threshold, for which a value of 3 is found to work well. The method proposed in the present work is an extension to *hp*-adaptation of the method proposed in Ref. [34] and it differs from Houston's approach in that it employs the merit function in Eqn. 37 instead of a user-prescribed parameter.

A. Cost

We consider two measures of computational cost. The first measure is solution storage that is proportional to the number of degrees of freedom in the discrete state vector. For tractability, we consider only the degrees of freedom pertinent to the flagged element κ^H,

$$c_{\mathrm{DOF}}(i) = \sum_{\kappa^h \in \kappa^H} (p_{\kappa^h}(i) + 1)^{\mathrm{dim}}, \tag{38}$$

where $\kappa^h \in \kappa^H$ denotes the subelements embedded in the original element selected for refinement and $p_{\kappa^h}(i)$ is the element's approximation order after the refinement as depicted in Figure 1. Note that $p_{\kappa^h} = p_{\kappa^H}$ for h-refinement while the number of embedded elements changes. Conversely, $p_{\kappa^h} = p_{\kappa^H} + 1$ for p-refinement and there is only one embedded element, *i.e.* the original element. Also, we are not considering the rank of the conserved state vector N_s because it is a constant throughout the mesh. It is worth emphasizing that this measure of cost is insensitive to the type of time integration used to solve Eqn. 25, and therefore it is a generic measure of cost.

The second measure of computational cost incorporates information about the time integration method. In this work, most of the computational time is spent solving the linear system in Eqn. 28 using the GMRES algorithm. In a sparse structure such as in Eqn. 28, we approximate the number of floating point operations in applying GMRES by the number of non-zero entries in the Jacobian matrix. Based on this observation, we define the second measure of cost as:

$$c_{\mathrm{NZ}}(i) = \sum_{\kappa^h \in \kappa^H} \left\{ (p_{\kappa^h}(i) + 1)^{2 \cdot \mathrm{dim}} + \sum_{\partial\kappa^h \setminus \partial D} [(p_{\kappa^h}(i) + 1) \cdot (p^-_{\kappa^h}(i) + 1)]^{\mathrm{dim}} \right\}, \tag{39}$$

where $p^-_{\kappa^h}$ denotes the approximation order of the neighboring element across face $\partial\kappa^h$, which must not be part of the boundary of the domain, ∂D. The first term in Eqn. 39 accounts for the self-blocks of the residual Jacobian matrix corresponding to each of the subelements. The second term corresponds to the dependence of the subelements' residual on the neighboring states. The cost

function does not take into account possible sparsity within the blocks of the Jacobian matrix, as such sparsity is not taken into account by the solver. Note that c_{NZ} is more sensitive to the number of spatial dimensions than c_{DOF}.

B. Benefit

The benefit $b(i)$ is a measure of how much improvement in the prediction of an output results from refinement option i. Evidently, the definition of benefit is not unique and it may be tailored for different applications and solution methods. However, it is desirable that such a definition is tractable and computationally inexpensive.

In an output-based mesh adaptation cycle, the steady-state residual is driven to zero at each step. Therefore, mesh modification on the element level can be interpreted as a local residual perturbation. Since an adjoint solution represents the sensitivity of an output with respect to a residual perturbation, we define our benefit function as:

$$b(i) = \sum_{\kappa^h \in \kappa^H} |\mathrm{R}_{\kappa^h}(\mathrm{U}_{sb}\mathrm{T}_{bd}(i))_j| \cdot |\Psi_{sb}\mathrm{T}_{bd}(i)\mathrm{V}_{sdj}|, \tag{40}$$

where $R_{\kappa^h}(\cdot)_j$ is a discrete residual component in the embedded element, d indexes the basis functions, $\mathrm{T}_{bd}(i)$ is a matrix that transfers the discrete solution U_{sb} to the local meshes for each refinement i, Ψ_{sb} is the discrete adjoint solution and V_{sdj} is an unrolling tensor as defined in Eqn. 26. Note that the adjoint variables act as positive weights for each of the perturbations.

The definition in Eqn. 40 relies on the following observations:

- At each step of the adaptation cycle, a discrete primal solution is found so that the residual vector is machine-zero. Therefore, the benefit as defined in Eqn. 40 is also machine-zero if computed before refining the central element.

- In the limit of the discrete solution representing the exact solution to machine precision, the result of Eqn. 40 will be of the order of machine precision for any refinement option.

- The refinement option with the largest $b(i)$ is expected to be the option that produces the largest change in the output of interest.

Note that Eqn. 40 is inexpensive to compute since only a residual calculation in the local mesh and data structures is required for each refinement option. Also, this framework is different than a residual-based decision because the values of the discrete adjoint provide information on the distribution of output sensitivity.

IX. Results

In this section, we assess the performance of our *hp*-adaptation framework using the cost measures c_{DOF} and c_{NZ}. The performance is measured in terms of number of degrees of freedom and CPU time. In the output-based adaptation methods, the time stamps include the solution of both the primal and adjoint solves, while for the uniform refinements only the primal solve time is included. For generality, we express the computational time in TauBench [52] time-units computed on one processor core with a grid-size of 2.5×10^5 and 10 steps.

We limit the maximum approximation order to $p_{\max} = 3$ for the two-dimensional cases to improve the performance and robustness of the adaptive method. Specifically, c_{DOF} under-estimates the computational expense of p-increment so that the adaptive algorithm prioritizes increasing p over anisotropically h-refining. Even though c_{NZ} more accurately estimates the computational expense of the different refinement options, $p_{\max}$ is the same for both cost measures to establish a basis of comparison between the cost measures. In addition, the artificial viscosity shock capturing approach used in some of the cases is not perfect and suffers from increased dissipation at high p that pollutes the error estimates and interferes with adaptation; limiting p is a simple and effective fix to this problem. In three dimensions, the effect of spatial dimensionality on c_{DOF} reduces the impact of its under-estimation of cost on the adaptive process, resulting in fewer elements targeted for p-refinement. For this reason, the p-orders are not limited in the three-dimensional problems.

A. NACA 0012

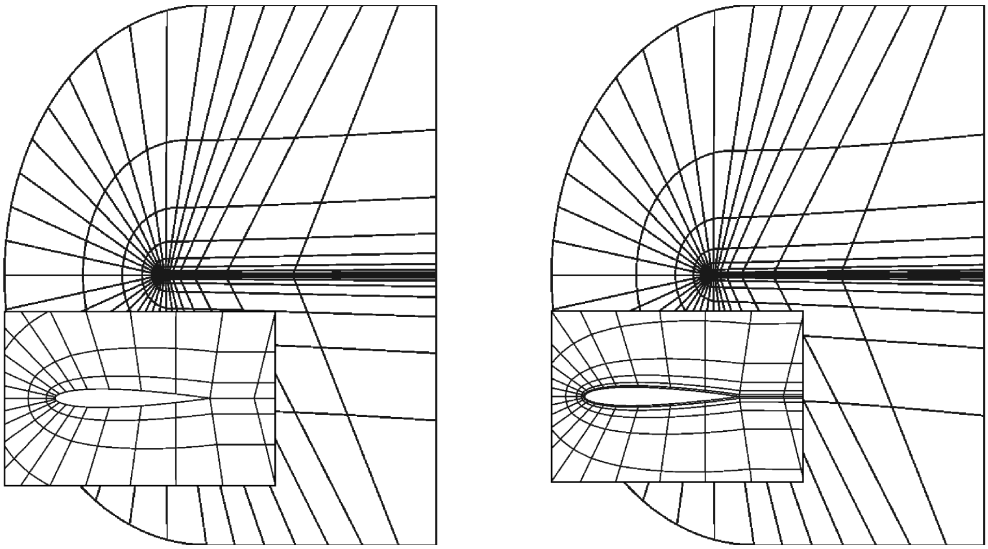
The first set of results we present consists of the NACA 0012 airfoil under three flow conditions:

1. $M_\infty = 0.5, \alpha = 2.0^o$, inviscid;

2. $M_\infty = 0.5, \alpha = 1.0^o, Re = 5 \times 10^3$;

3. $M_\infty = 0.8, \alpha = 1.25^o$, inviscid.

The airfoil geometry was modified to have zero-thickness at the trailing edge. The initial meshes (Figure 3) for these cases are composed of quartic ($q = 4$) quadrilaterals and the outer boundary is located 50 chord-lengths away from the airfoil. Our output of interest is the near-field drag. We compare the performance of the *hp*-adaptation routine using c_{DOF} and c_{NZ} against uniform h and p refinements. At each step of the adaptive procedure, $f^{\text{adapt}} = 10\%$ of elements with the largest error indicators, η_{κ^H}, is selected for refinement. All runs used 4 Nehalem 8-core nodes from the Nyx cluster at the University of Michigan.



(a) Initial mesh for inviscid flows over the NACA 0012.

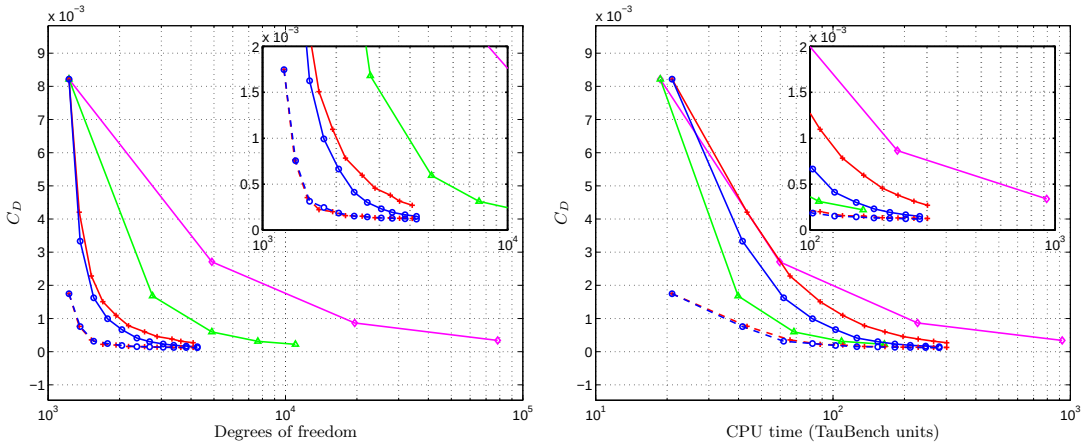
(b) Initial mesh for viscous flow over the NACA 0012.

Fig. 3 Initial quartic ($q = 4$) meshes for the NACA 0012 cases.

1. $M_\infty = 0.5, \alpha = 2.0^o$, inviscid

The first test case is inviscid flow at $M_\infty = 0.5$ and $\alpha = 2.0^o$. In the hypothetical case of the outer boundary of the computational domain being located infinitely far, the drag measured on the surface of the airfoil should converge to zero since there are neither viscous effects nor shocks in the flow. However, when the far-field is located at a finite distance the near-field drag converges to a finite value [53] as plotted in Figure 4. When corrected by the error estimates, the performance of the *hp*-adaptation routine is not very sensitive, in this case, to the different cost measures. However,

the uncorrected drag values (solid lines) converge faster when the *hp*-adaptation uses c_{DOF}, both in degrees of freedom (Figure 4(a)) and in CPU time (Figure 4(b). Note that uniform *p*-refinement performs very well in this case since the flow is smooth and the cost of uniformly high-order solutions is moderate in two dimensions. Also, the computational load balance in parallel runs adversely affects the *hp*-runs due to non-uniformity of *p*-orders.



(a) Drag coefficient evolution with respect to degrees of freedom

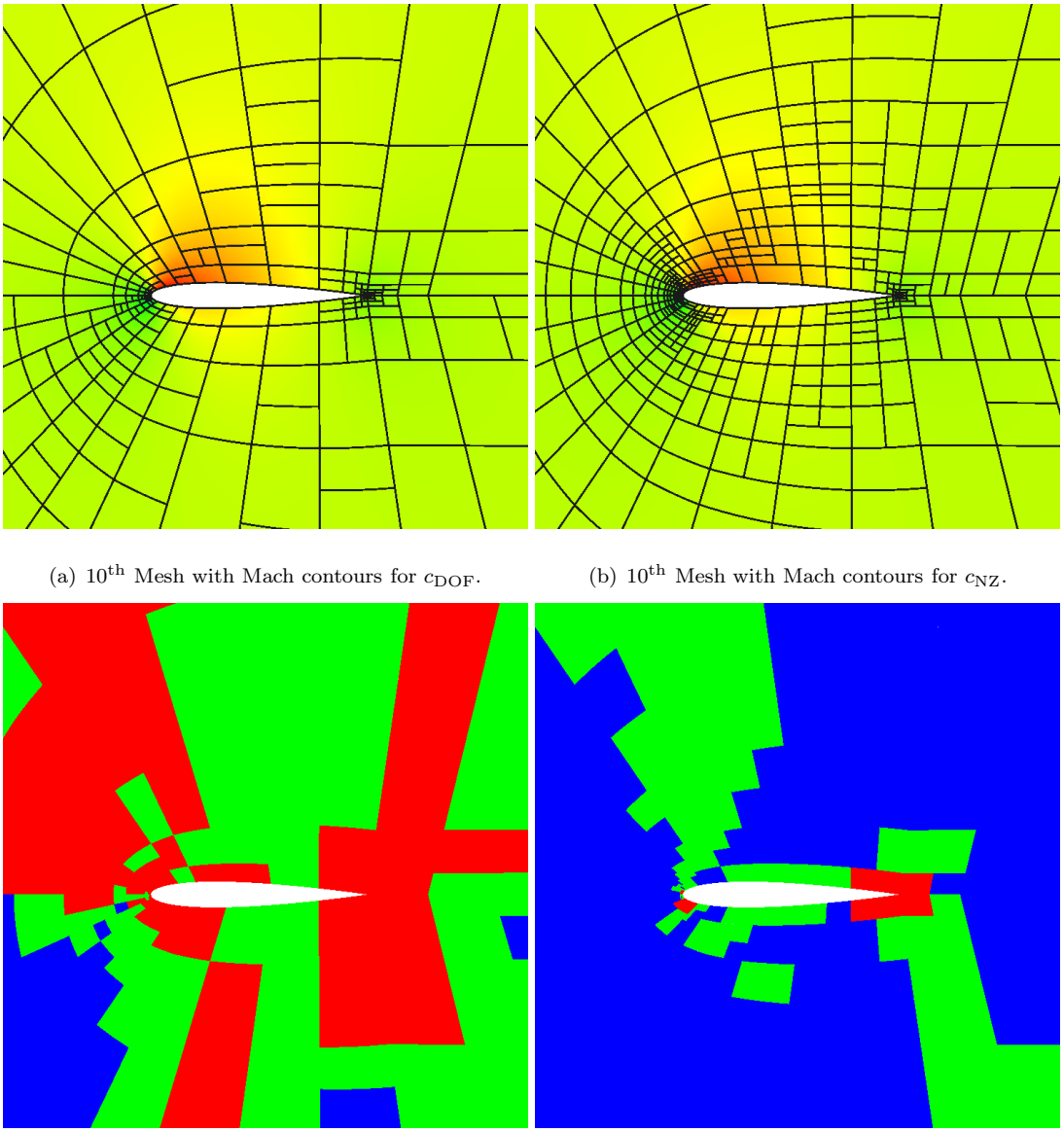
(b) Drag coefficient evolution with respect to CPU time

Fig. 4 NACA 0012, $M_\infty = 0.5, \alpha = 2.0^o$, inviscid: drag coefficient convergence; $\diamond$: uniform *h*-refinement; $\triangle$: uniform *p*-refinement; $\circ$: *hp*-adaptation with c_{DOF}; $+$: *hp*-adaptation with c_{NZ}. The dashed lines correspond to the drag values corrected with the error estimate.

Figure 5 shows the final *hp*-adapted meshes. In both c_{DOF} and c_{NZ} cases, isotropic *h*-refinement and order increment are active at the trailing edge in order to accurately represent the total-pressure recovery that is related to drag through entropy generation [53, 54]. Note that the *hp*-mesh obtained with c_{DOF} has a larger area at higher approximation order ($p > 1$), while the adaptive method with c_{NZ} shows significantly more mesh subdivisions.

2. $M_\infty = 0.5, \alpha = 1.0^o, Re = 5 \times 10^3$

The second two-dimensional test case is subsonic viscous flow at $M_\infty = 0.5, \alpha = 1.0^o$ and $Re = 5 \times 10^3$. Similarly to the previous case, we compare the *hp*-adaptation framework using both cost measures against uniform *h* and *p* refinements. Figure 6(a) shows the drag coefficient convergence in terms of number of degrees of freedom. While both *hp*-adaptation runs present



(a) 10^{th} Mesh with Mach contours for c_{DOF}.

(b) 10^{th} Mesh with Mach contours for c_{NZ}.

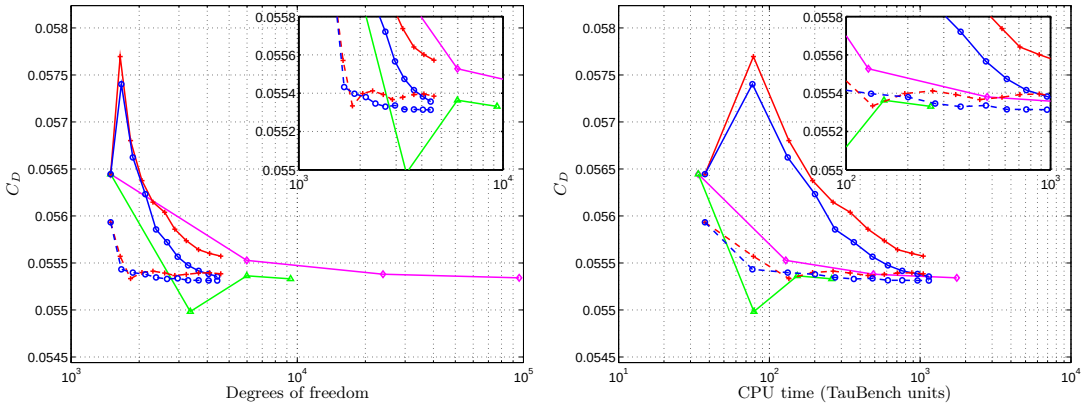
(c) 10^{th} p-order distribution for c_{DOF}; blue indicates $p = 1$; red indicates $p = 3$.

(d) 10^{th} p-order distribution for c_{NZ}; blue indicates $p = 1$; red indicates $p = 3$.

Fig. 5 NACA 0012, $M_\infty = 0.5, \alpha = 2.0^o$, inviscid: hp-adapted meshes for drag.

similar convergence histories for the corrected output (dashed lines), the uncorrected drag values (solid lines) converge faster with c_{DOF} than when c_{NZ} is employed. Additionally, the hp-adaptation runs converge the corrected output with significantly fewer degrees of freedom than the uniform refinements. This observation is also valid in terms of CPU time (Figure 6(b)), however the savings are smaller.

As observed in the previous case, the adaptive scheme produces a larger area of the domain



(a) Drag coefficient evolution with respect to degrees of freedom

(b) Drag coefficient evolution with respect to CPU time

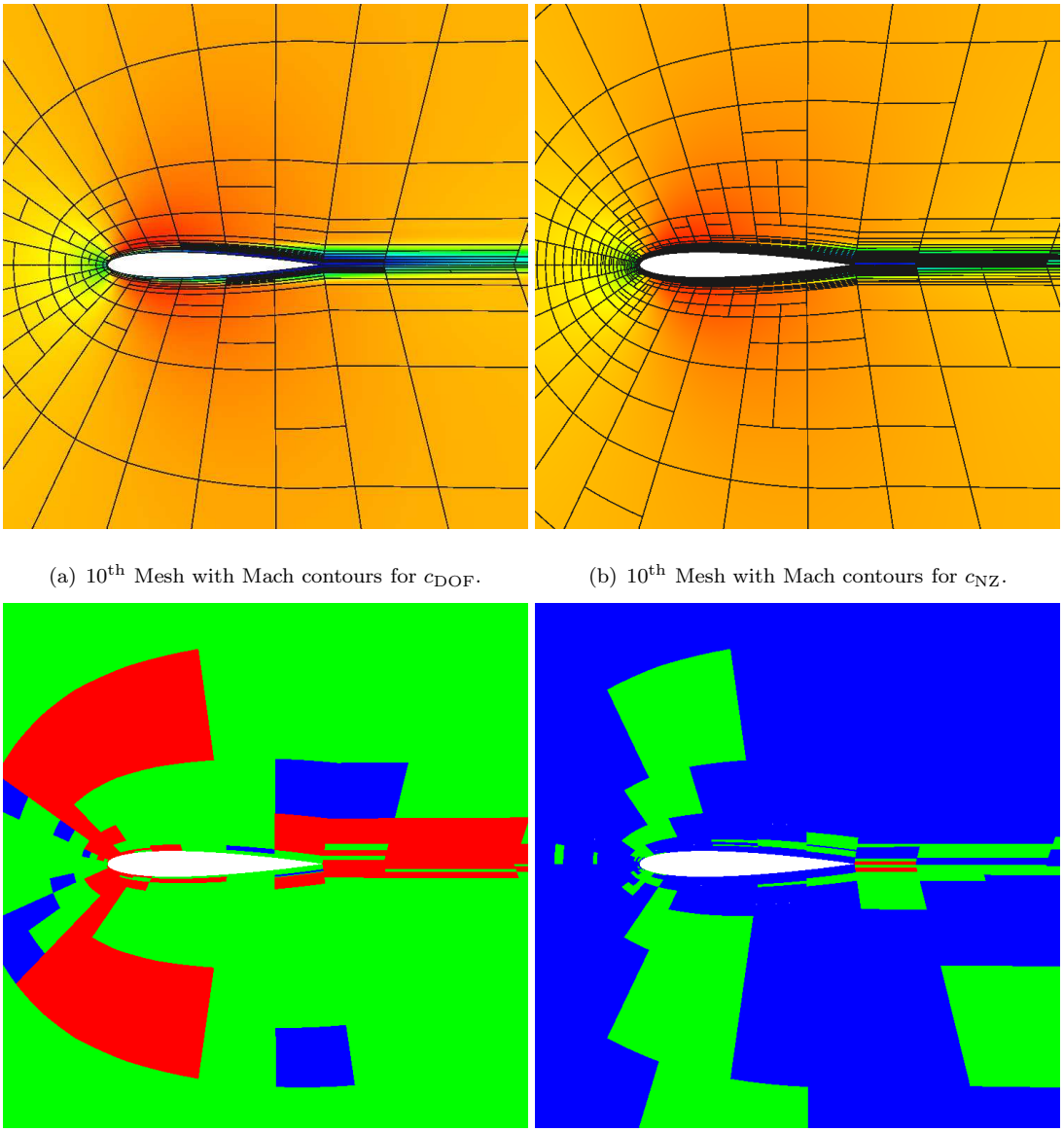
Fig. 6 NACA 0012, $M_\infty = 0.5, \alpha = 1.0^o, Re = 5 \times 10^3$: drag coefficient convergence; $\diamond$: uniform h-refinement; $\triangle$: uniform p-refinement; $\circ$: hp-adaptation with c_{DOF}; $+$: hp-adaptation with c_{NZ}. The dashed lines correspond to the drag values corrected with the error estimate.

with higher order cells when c_{DOF} is used to measure the cost of the refinement options (Figure 7(c)). In contrast, the adaptive algorithm, when using c_{NZ}, chooses p-increment mostly in the wake region combined with anisotropic h-refinement as seen in Figures 7(d) and 7(b) respectively.

3. $M_\infty = 0.8, \alpha = 1.25^o$, *inviscid*

The final two-dimensional case we present is inviscid transonic flow over the NACA 0012 geometry. For shock-capturing, we use an element-wise constant artificial viscosity approach [40]. The ability to robustly resolve discontinuities like shocks in high-order discretizations is a current challenge in CFD. Due to robustness problems, the error estimates can suffer and their convergence may not be reliable. Specifically, noise in the error estimates arises from dual-inconsistency of the shock indicator and the use of hierarchical mesh refinement on a fixed background topology. For this reason, we consider in this case only the pure drag output, that is, the output without the correction by the error estimate.

We limit the maximum approximation order to $p_{\max} = 3$ for the reason described in the beginning of this section. Also, we found that the piecewise-constant artificial viscosity combined with a resolution-based discontinuity indicator used for shock capturing adds excessive dissipation for $p \geq 4$. The excessive dissipation leads to benefit-function values that do not necessarily favor the



(a) 10^{th} Mesh with Mach contours for c_{DOF}.

(b) 10^{th} Mesh with Mach contours for c_{NZ}.

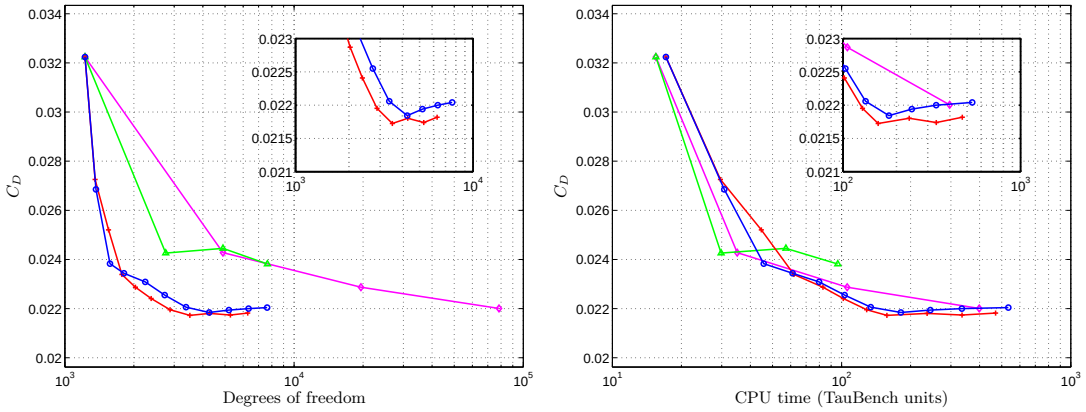
(c) 10^{th} p-order distribution for c_{DOF}; blue indicates $p = 1$; red indicates $p = 3$.

(d) 10^{th} p-order distribution for c_{NZ}; blue indicates $p = 1$; red indicates $p = 3$.

Fig. 7 NACA 0012, $M_\infty = 0.5, \alpha = 1.0^o, Re = 5 \times 10^3$: hp-adapted meshes for drag.

best refinement direction.

Figure 8(a) shows the drag convergence with respect to degrees of freedom. Note that the adaptation converges slightly faster with c_{NZ} than with c_{DOF} and in comparison with uniform h-refinement, the hp methods use approximately an order of magnitude fewer degrees of freedom. The savings in CPU time are smaller but still significant, since the hp methods took about half the time to achieve drag convergence to within 2 counts of drag (Figure 8(b)).



(a) Drag coefficient evolution with respect to degrees of freedom.

(b) Drag coefficient evolution with respect to CPU time.

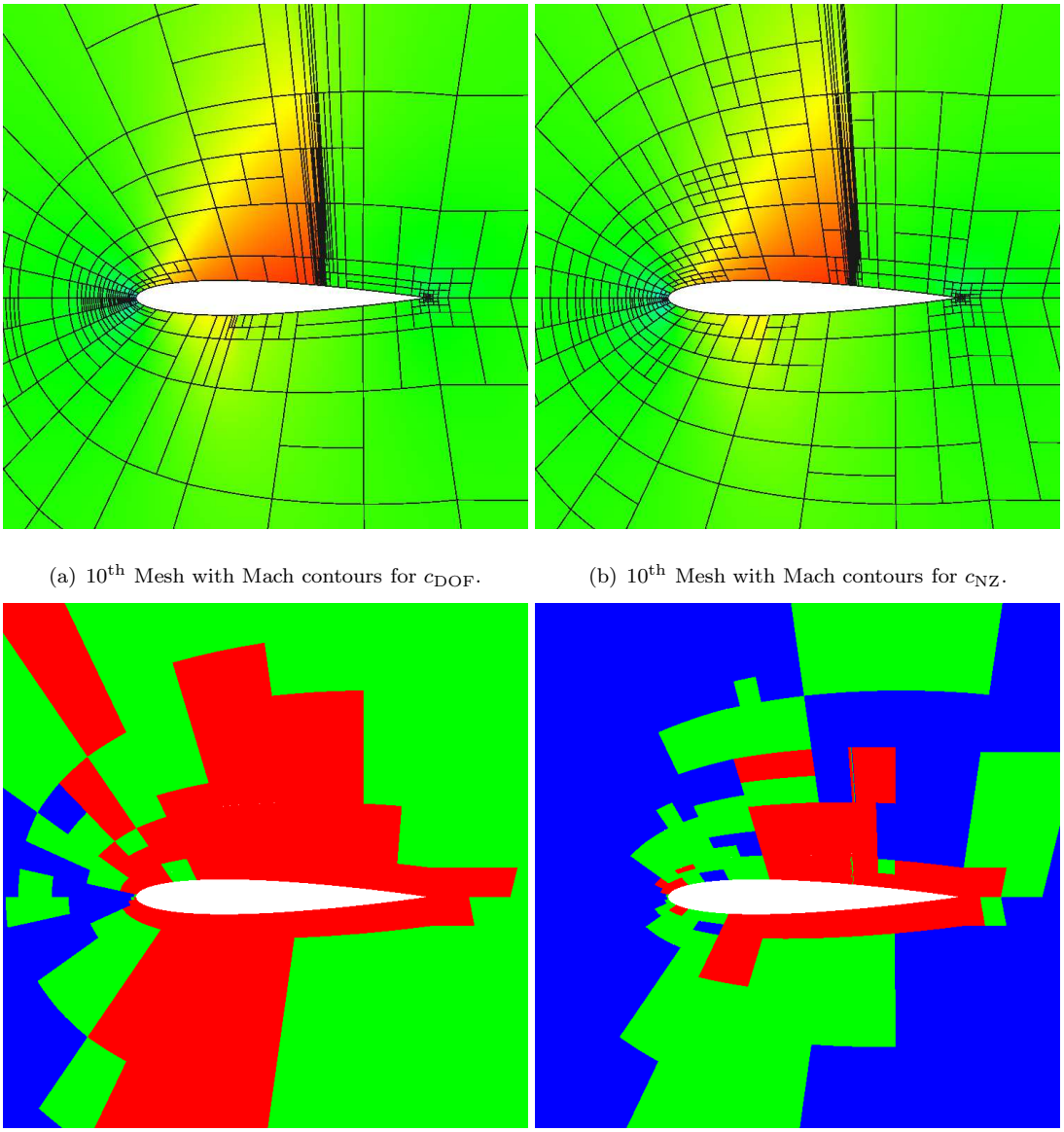
Fig. 8 NACA 0012, $M_\infty = 0.8, \alpha = 1.25^o$, inviscid: drag coefficient convergence; $\diamond$: uniform h-refinement; $\triangle$: uniform p-refinement; $\circ$: hp-adaptation with c_{DOF}; $+$: hp-adaptation with c_{NZ}.

The meshes for c_{DOF} and c_{NZ} are shown in Figure 9. Note that both methods choose anisotropic h-refinement in combination with $p-$increment in the vicinity of the strong shock on the upper surface and the weaker shock on bottom surface of the airfoil. It is also notable that both isotropic h-refinement and higher-order cells are present at the trailing edge in order to accurately represent the pressure recovery.

B. NLR Delta wing, $M_\infty = 0.3, \alpha = 12.5^o, Re = 4 \times 10^3$

The second case we present is laminar flow over the NLR delta wing at a high angle of attack. The vortical structure of this flow presents both sharp and smooth features that can benefit from hp-adaptation. We consider the hp-adaptation framework using both c_{DOF} and c_{NZ} cost measures. At each step of these strategies, $f^{\text{adapt}} = 10\%$ of the elements in the mesh was adapted starting from a $p = 1$ solution (Figure 10(a)) on the second-level linear mesh generated by NLR as part of the ADIGMA [55] project. The output of interest is drag and we present uniform h-refinement and uniform p-refinement along with the output-adapted meshes.

All calculations used 50 Harpertown 8-core nodes from NASA's Pleaides supercomputer and we fixed the maximum CPU time for each of the mesh-improvement strategies. The reason for fixing a CPU budget is to simulate a condition in which a practitioner has a certain amount of time to



(a) 10^{th} Mesh with Mach contours for c_{DOF}.

(b) 10^{th} Mesh with Mach contours for c_{NZ}.

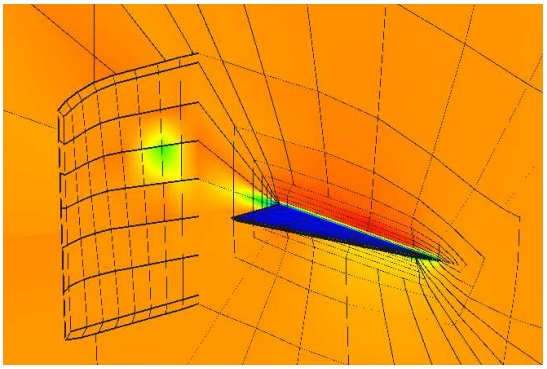
(c) 10^{th} p-order distribution for c_{DOF}; blue indicates $p = 1$; red indicates $p = 3$.

(d) 10^{th} p-order distribution for c_{NZ}; blue indicates $p = 1$; red indicates $p = 3$.

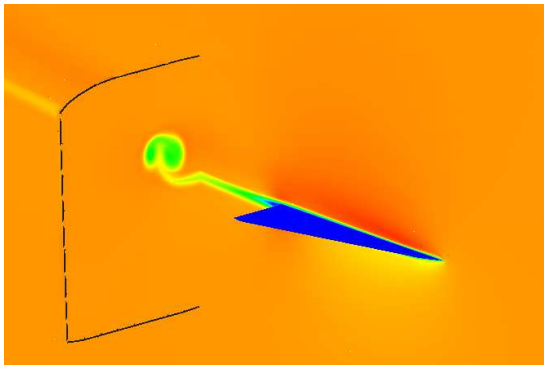
Fig. 9 NACA 0012, $M_\infty = 0.8, \alpha = 1.25^o$, inviscid: hp-adapted meshes for drag.

provide an answer to an engineering problem. We then assess the quality of the answer that each of the mesh-improvement strategies obtained within that CPU-time budget. The last converged solutions of all strategies are shown in Figure 10 along with the initial and reference solutions in Figure 10(a) and Figure 10(b) respectively.

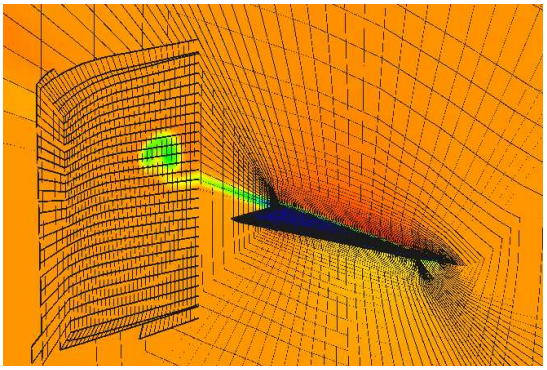
Figure 11 shows the evolution of the drag coefficient in terms of degrees of freedom and CPU-time. The dashed lines in that figure are the outputs of the adjoint-based adaptation methods



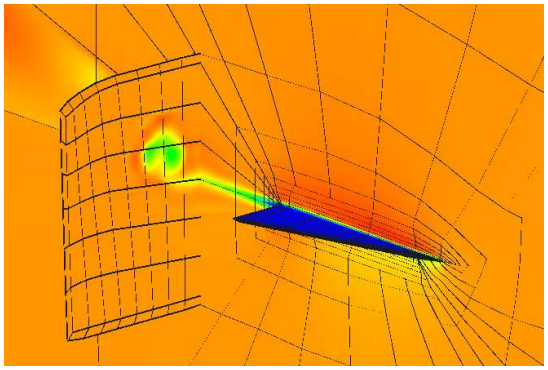
(a) Initial mesh with Mach number contours computed with $p = 1$ (3264 linear elements).



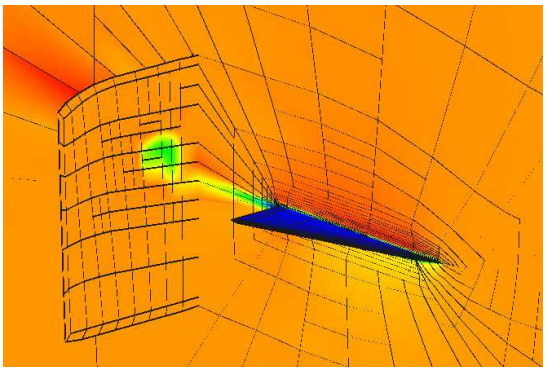
(b) Mach contours on a finer mesh with $p = 2$.



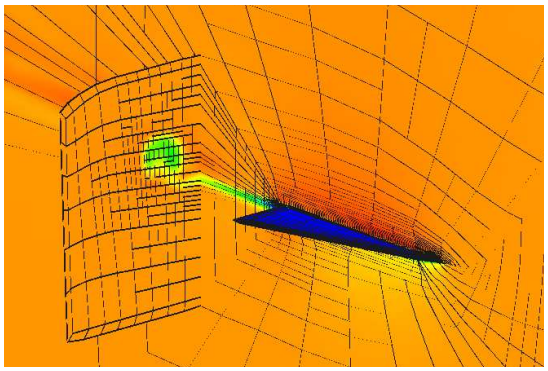
(c) Two levels of uniform h-refinement with $p = 1$ (208896 linear elements).



(d) Two levels of uniform p-refinement $p = 1 \rightarrow 3$ (3264 linear elements).



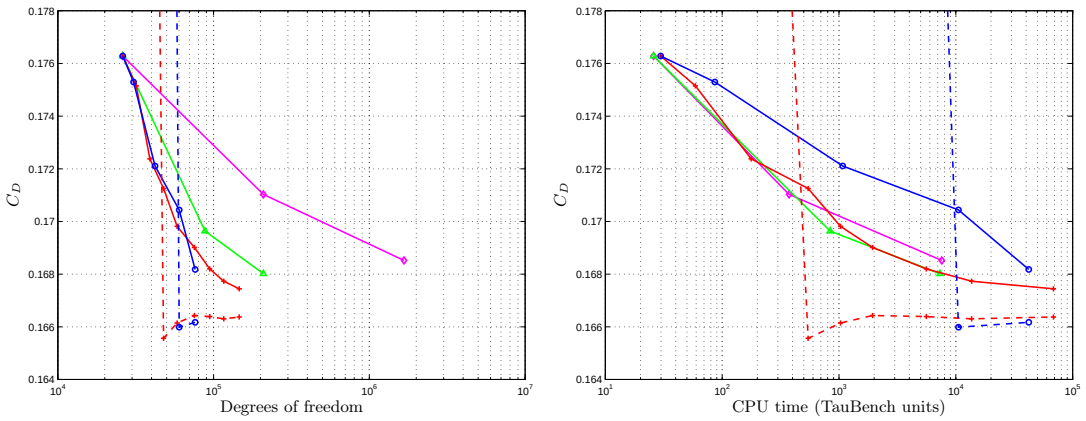
(e) 4^{th} drag-adapted mesh using c_{DOF} (5778 elements).



(f) 8^{th} drag-adapted mesh using c_{NZ} (16260 elements).

Fig. 10 NLR Delta wing, $M_\infty = 0.3, \alpha = 12.5^o, Re = 4 \times 10^3$: Initial and adapted meshes with Mach number contours.

corrected by their corresponding error estimates. Note that in the initial adaptation steps, the error estimates are very large. However, they converge rapidly after 2 to 3 adaptation steps. Even though the performance of c_{NZ} and of c_{DOF} are similar in terms of degrees of freedom (Figure 11(a)), in



(a) Drag coefficient evolution with respect to degrees of freedom

(b) Drag coefficient evolution with respect to CPU time

Fig. 11 NLR Delta, $M_\infty = 0.3, \alpha = 12.5^o, Re = 4 \times 10^3$: drag coefficient convergence; $\diamond$: uniform h-refinement; $\triangle$: uniform p-refinement; $\circ$: hp-adaptation with c_{DOF}; $+$: hp-adaptation with c_{NZ}. The drag values with the error estimates for the first three adaptive steps are out of the range of the vertical axis. The dashed lines correspond to the drag values corrected with the error estimate.

terms of computational time (Figure 11(b)), c_{NZ} shows a clear advantage over c_{DOF} since higher p-orders are more expensive than c_{DOF} estimates. Also, the uniform refinement strategies perform remarkably well in CPU-time. This is due to an adequate off-wall spacing and overall high-quality of the initial mesh which makes the error close to equally distributed and, in this case, the large increments in number of degrees of freedom provides an advantage since no time is spent solving on intermediate meshes and the homogeneity of p-order helps the balance of computational load on the processors.

Table 3 lists the frequency of choice of each of the refinement options for the different cost measures. When using c_{DOF}, the adaptation method chooses p-refinement significantly more often compared to when c_{NZ} is employed. This larger frequency of p-refinement with c_{DOF} is due to under-estimation of the computational cost of solving higher-order discretizations and it causes the slower output convergence in terms of CPU time shown in Figure 11(b). Note that both methods tend to choose p-refinement more frequently in the later adaptation steps.

The larger frequency of p-refinement when using c_{DOF} is illustrated in Figure 12. This figure

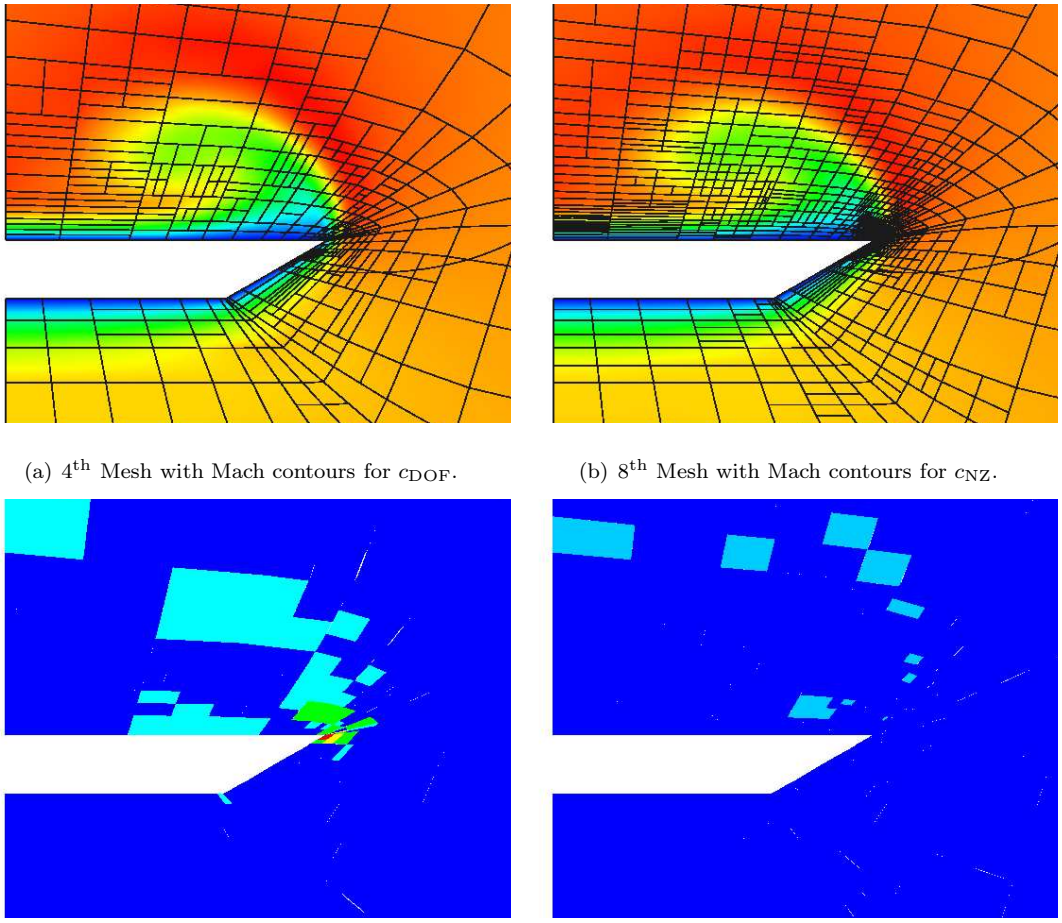
Table 3 NLR Delta wing, $M_\infty = 0.3, \alpha = 12.5^o, Re = 4 \times 10^3$, drag-driven adaptation: percentage of choice for each refinement option; iso-h: isotropic h-refinement; sc-h: single-cut h-refinements; dc-h: double-cut h-refinements; iso-p: isotropic p-refinement.

	c_{DOF}				c_{NZ}			
Adaptation step	iso-h	sc-h	dc-h	iso-p	iso-h	sc-h	dc-h	iso-p
1	0.0	84.9	0.0	15.0	0.0	100.0	0.0	0.0
2	0.0	75.1	0.8	24.0	0.0	98.0	1.5	0.5
3	0.0	66.8	1.4	31.8	0.0	95.4	2.5	2.1
4	0.0	70.4	0.0	29.6	0.0	96.2	0.4	3.4
5	–	–	–	–	0.0	95.2	1.2	3.5
6	–	–	–	–	0.0	96.0	0.9	3.0
7	–	–	–	–	0.0	95.7	0.6	3.9
8	–	–	–	–	0.0	95.9	0.3	3.8

compares the mesh and approximation order distribution for both measures of cost at a mid-chord cut of the delta wing. Note that both methods choose h-refinement on the upper sharp corner where shear effects are prominent. When using c_{NZ}, the adaptation routine reserves p-refinement mostly for regions where the flow field is smooth, while with c_{DOF} the mesh shows a combination of h and p refinements in sharp and smooth areas.

C. Third Drag Prediction Workshop W1 geometry, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$

In this case study, we consider the baseline wing geometry (DPW-W1) from the third AIAA Drag Prediction Workshop [3]. This case consists of turbulent, transonic flow over a tapered wing and the mesh adaptation routine is driven by the drag output. The initial curved mesh, shown in Figure 13(a), was obtained through agglomeration of cells from a finer structured linear C-grid generated specifically for this purpose. In the agglomeration, each curved hexahedral element was obtained by merging twenty seven linear elements using a distance-based Lagrange interpolation of the nodal coordinates, resulting in cubic ($q = 3$) geometry interpolation. Also, the spacing of the linear mesh is such that the agglomerated mesh presents $y^+ \approx 1$ for the first element off the wall



(a) 4^{th} Mesh with Mach contours for c_{DOF}.

(b) 8^{th} Mesh with Mach contours for c_{NZ}.

(c) 4^{th} p-order distribution for c_{DOF}; the range is $p = 1 \rightarrow 5$.

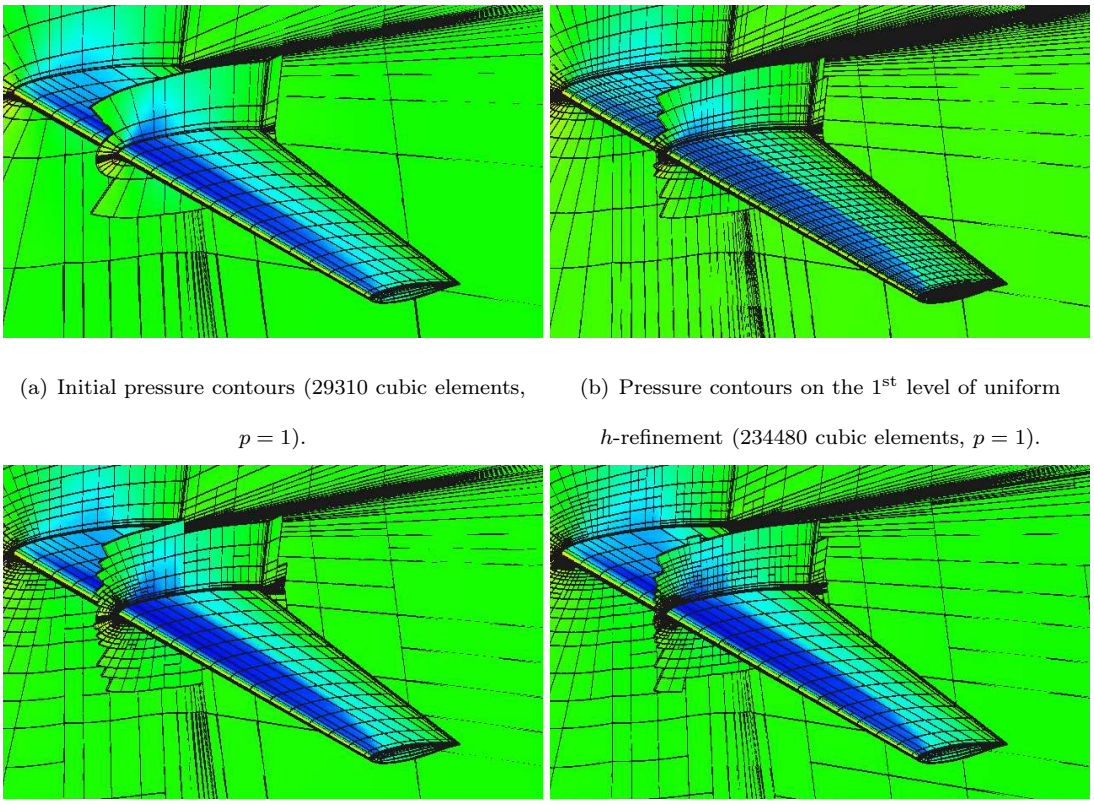
(d) 8^{th} p-order distribution for c_{NZ}; the range is $p = 1 \rightarrow 6$.

Fig. 12 NLR Delta wing, $M_\infty = 0.3, \alpha = 12.5^o, Re = 4 \times 10^3$: half-chord cut of the drag-adapted meshes.

as recommended in the workshop [56] and the outer boundary is located at 100 mean-aerodynamic-chord-lengths away from the wing.

As described in Section III, we used the Spalart-Allmaras turbulence model without trip terms and we assumed the flow as fully turbulent. Also, Persson and Peraire's [40] shock-capturing method is used to improve stability. The baseline flow solution was obtained with linear ($p = 1$) approximation order. As a basis of comparison, all the adaptive schemes started from the same initial solution so that all methods are compared against the same initial time-stamp. For the adjoint-based adaptation methods, the CPU time taken for the initial adjoint solve is also included in the initial starting time.

All of the calculations for this case were executed on 180 Harpertown 8-core nodes from NASA's Pleiades supercomputer. Due to the computational expense of these runs, we did not perform a statistical study to account for machine performance variability in the CPU-time measurements.



(a) Initial pressure contours (29310 cubic elements, $p = 1$).

(b) Pressure contours on the 1^{st} level of uniform h-refinement (234480 cubic elements, $p = 1$).

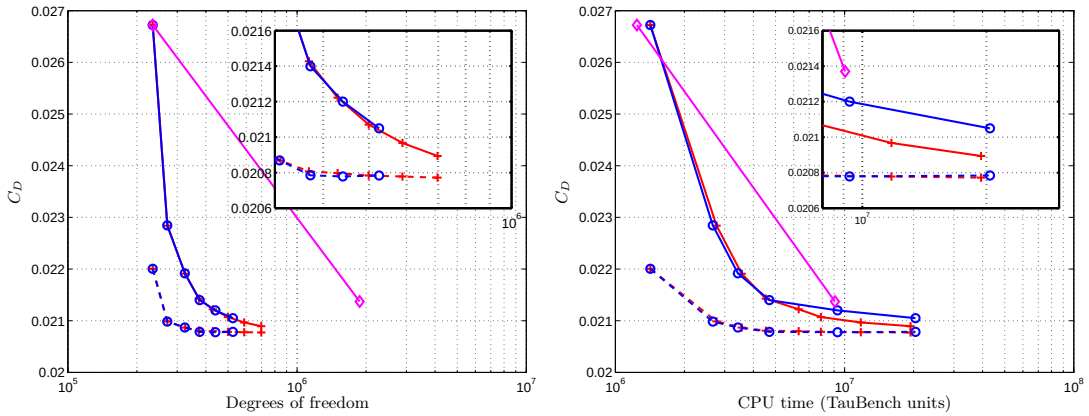
(c) Pressure contours on the 5^{th} drag-adapted mesh using c_{DOF} (59503 cubic elements).

(d) Pressure contours on the 7^{th} drag-adapted mesh using c_{NZ} (85377 cubic elements).

Fig. 13 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: Initial and drag-adapted meshes with pressure contours.

We compare three mesh improvement strategies starting from the initial $p = 1$ solution shown in Figure 13(a). As reference, one of the strategies is uniform h-refinement (Figure 13(b)) in which all hexahedra are divided into 8 elements. The two cost measures described earlier are compared for hp-adaptation in which $f^{\text{adapt}} = 10\%$ of the elements is selected for refinement at each adaptation step. Additionally, we fixed the overall budget of CPU wall-time for each of the three runs and the last converged solution obtained within that budget are shown in Figure 13.

Figure 14 shows the drag coefficient convergence for the mesh refinement strategies. The solid lines in Figures 14(a) and 14(b) are the computed drag values and the dashed lines correspond to



(a) Drag coefficient evolution with respect to number of degrees of freedom

(b) Drag coefficient evolution with respect to CPU time

Fig. 14 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: drag coefficient convergence; $\diamond$: uniform h-refinement; $\circ$: hp-adaptation with $c_{\mathbf{DOF}}$; $+$: hp-adaptation with $c_{\mathbf{NZ}}$. The dashed lines correspond to the drag values corrected with the error estimate.

the output corrected with the error estimate. The difference between these corrected values for the last two adaptation steps of the output-based strategies is within 0.15 counts of drag. Note that the performance in terms of degrees of freedom of the output-based strategies is very similar. However, in terms of CPU time, the use of c_{NZ} leads to faster output convergence. This difference is due to the more representative measure of solution cost by c_{NZ}. This effect is illustrated in Table 4 where we show the frequency at which the refinement options were chosen for each cost measure at each adaptation step. Note that for c_{NZ}, p-refinement is chosen significantly less often than for c_{DOF} and both methods have a propensity to choose p-refinement more often in the later stages of adaptation. Moreover, the large increase in CPU time between the third and fourth adaptation steps for c_{DOF} (Figure 14(b)) is an effect of p-increment being chosen more often for c_{DOF} (Table 4) which makes the primal and adjoint solves more expensive.

In flows with high Reynolds number, highly stretched cells in regions such as boundary layers and wakes are key to an efficient calculation. To assess the levels of anisotropy in our meshes, we define an aspect ratio measure for one element as follows:

$$\Lambda = \frac{\left(\frac{\mathbb{S}}{2 \cdot \mathrm{dim}}\right)^{\frac{\mathrm{dim}}{(\mathrm{dim}-1)}}}{\mathbb{V}}, \tag{41}$$

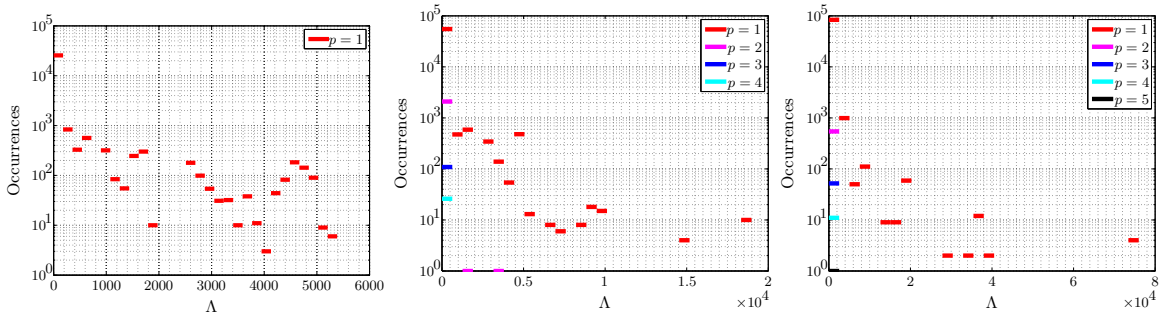
where $\mathbb{S}$ and $\mathbb{V}$ are the cell surface area and volume respectively. Note that $\Lambda = 1$ for a square in

Table 4 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: percentage of choice for each refinement option; iso-h: isotropic h-refinement; sc-h: single-cut h-refinements; dc-h: double-cut h-refinements; iso-p: isotropic p-refinement.

	c_{DOF}				c_{NZ}			
Adaptation step	iso-h	sc-h	dc-h	iso-p	iso-h	sc-h	dc-h	iso-p
1	0.0	99.3	0.0	0.7	0.0	100.0	0.0	0.0
2	0.0	97.3	0.0	2.7	0.0	99.9	0.0	0.1
3	0.0	94.9	0.0	5.1	0.0	99.8	0.0	0.2
4	0.0	91.8	0.4	7.8	0.0	99.1	0.3	0.6
5	0.0	90.6	0.3	9.1	0.0	98.7	0.5	0.8
6	–	–	–	–	0.0	98.6	0.5	0.9
7	–	–	–	–	0.0	98.6	0.4	1.0

two dimensions and a cube in three dimensions. Since the refinement is performed in the elements' reference space, isotropic refinement does not necessarily preserve Λ on curved elements.

Figure 15 shows histograms of the aspect ratios of the cells in the initial and adapted meshes. Note that the aspect ratios in the adapted meshes are in the range of tens of thousands and the higher-order cells generally have lower aspect ratios (in the hundreds range).

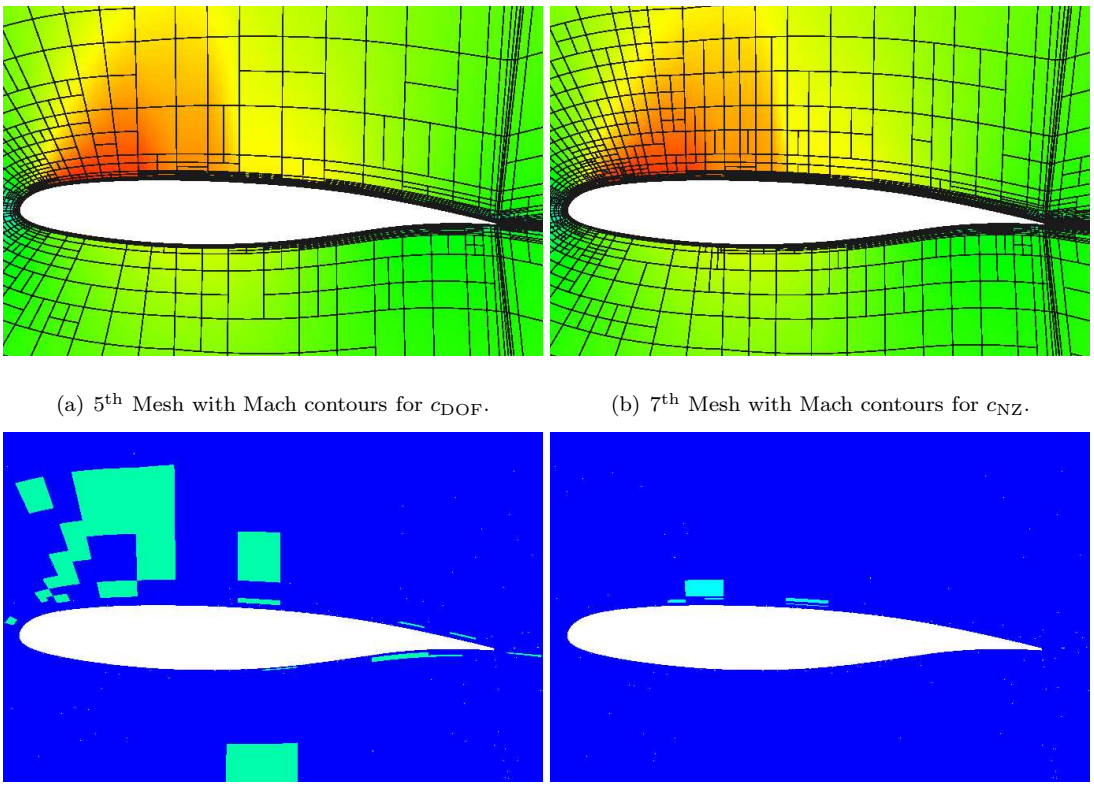


(a) Initial solution. (b) Final (5^{th}) adapted mesh with c_{DOF}. (c) Final (7^{th}) adapted mesh with c_{NZ}.

Fig. 15 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: aspect ratio histograms for the initial and adapted meshes.

Figures 16 and 17 show two cuts at representative span-wise positions. For comparison purposes,

the contours are scaled to the same range for both c_{DOF} and c_{NZ}. The Mach-number contours are similar for both strategies, however c_{NZ} presents a larger number of anisotropic cells along the shock and on the boundary layer. The larger percentage of p-refinement observed for c_{DOF} is illustrated in the order distribution figures. Note that both methods have mostly $p = 1$ cells at the shock and higher-order cells on each side of the shock.



(a) 5^{th} Mesh with Mach contours for c_{DOF}.

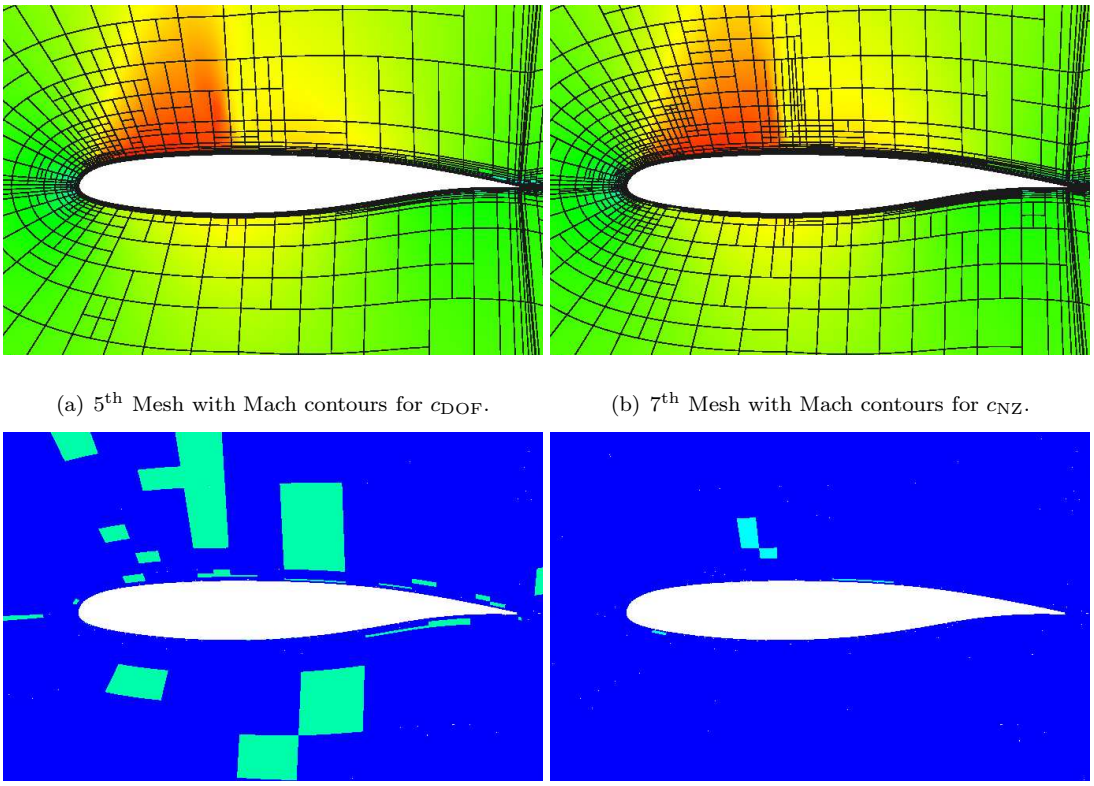
(b) 7^{th} Mesh with Mach contours for c_{NZ}.

(c) 5^{th} p-order distribution for c_{DOF}; the range is $p = 1 \rightarrow 5$.

(d) 7^{th} p-order distribution for c_{NZ}; the range is $p = 1 \rightarrow 5$.

Fig. 16 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$**: cut at** $y = 220$**mm of the drag-adapted meshes.**

Design optimization methods offer insight on improving vehicle configurations. Similarly, an optimization-based mesh adaptation algorithm offers insight on improving gridding guidelines. We noticed that several regions of the flow were frequently targeted for refinement. One of these regions is near the leading edge where the flow accelerates through the sonic condition. This transition causes strong variations in the adjoint solution which are responsible for large error indicators. In fact, the adjoint solution is C^1-discontinuous through the sonic condition in inviscid quasi-1D flows



(a) 5^{th} Mesh with Mach contours for c_{DOF}.

(b) 7^{th} Mesh with Mach contours for c_{NZ}.

(c) 5^{th} p-order distribution for c_{DOF}; the range is $p = 1 \rightarrow 4$.

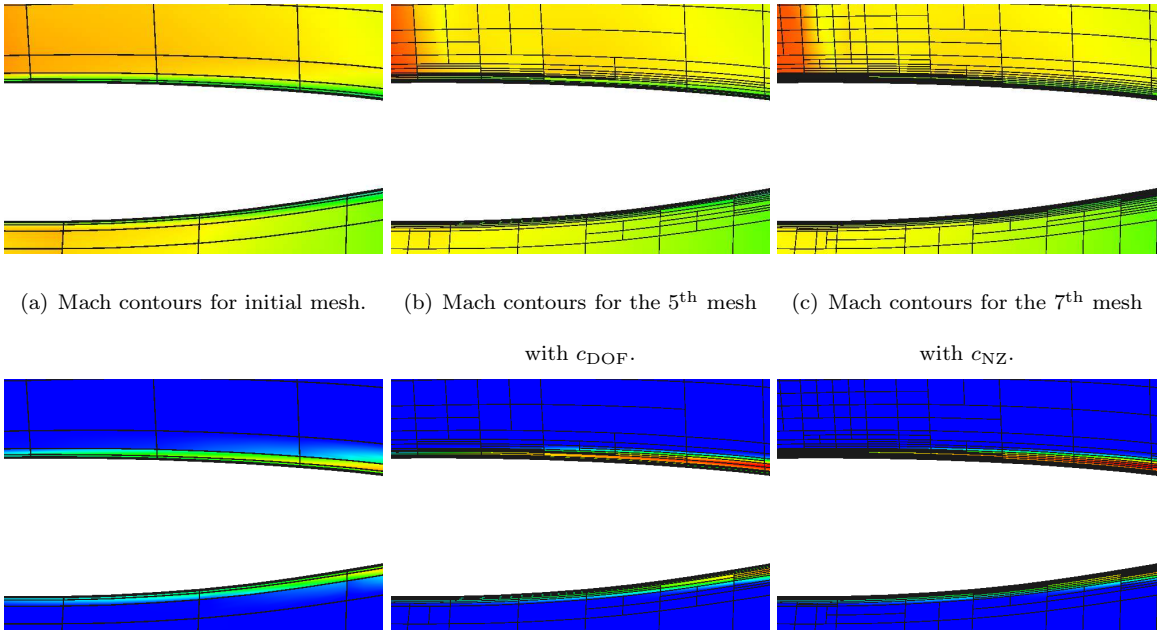
(d) 7^{th} p-order distribution for c_{NZ}; the range is $p = 1 \rightarrow 5$.

Fig. 17 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: cut at $y = 620$mm of the drag-adapted meshes.

[57]. Another region is the edge of the boundary layer, where the turbulent working variable, $\tilde{\nu}$, transitions to zero rapidly. The other two regions are the shock-boundary-layer interaction and the trailing edge. These regions exhibit strong gradients of $\tilde{\nu}$ that contribute to the drag output. Figure 18 shows the interaction between the shock and the boundary layer. Note the concentration of cells in the boundary layer and the sharp variation of $\tilde{\nu}$. Further downstream, in the trailing edge region (Figure 19), the beginning of turbulent wake is also adapted.

X. Conclusion

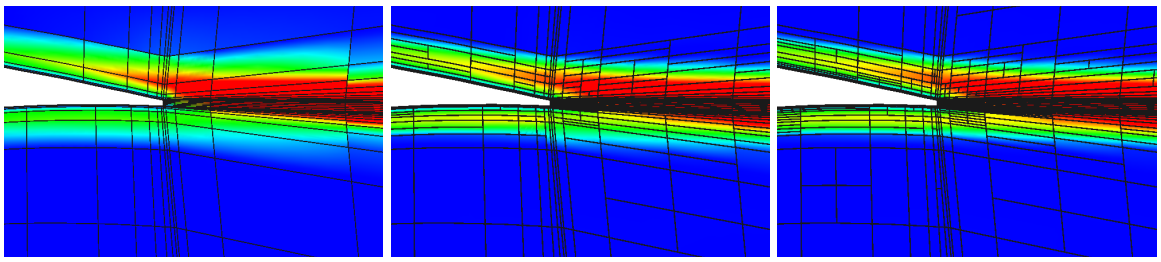
An optimization-based anisotropic *hp*-adaptation method is proposed in which p-refinement is considered a refinement option amongst directional cuts in a cell. The refinement options are ranked based on a cost-benefit analysis in which the benefit is an output sensitivity with respect to the different ways of refining the solution space. The cost is estimated considering two measures of



(a) Mach contours for initial mesh. (b) Mach contours for the 5^{th} mesh with c_{DOF}. (c) Mach contours for the 7^{th} mesh with c_{NZ}.

(d) $\tilde{\nu}$ contours for initial mesh. (e) $\tilde{\nu}$ contours for the 5^{th} mesh with c_{DOF}. (f) $\tilde{\nu}$ contours for the 7^{th} mesh with c_{NZ}.

Fig. 18 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: interaction between shock and boundary-layer at $y = 620$mm.



(a) $\tilde{\nu}$ contours for initial mesh. (b) $\tilde{\nu}$ contours for the 5^{th} mesh with c_{DOF}. (c) $\tilde{\nu}$ contours for the 7^{th} mesh with c_{NZ}.

Fig. 19 DPW Wing 1, $M_\infty = 0.76, \alpha = 0.5^o, Re = 5 \times 10^6$: trailing edge at $y = 620$mm.

CPU work: number of degrees of freedom and number of floating point operations. The latter is correlated to the number of non-zero entries in the residual Jacobian.

The cost measures presented here are not perfect as they do not account for stiffness effects of different refinements in the iterative solution method. These effects have a direct impact in the CPU time, but they are difficult and computationally intensive to estimate because they are global measures and their effect depends on the type of solver and preconditioner used. Since the merit

function is computed multiple times in each refinement cycle, the local property of cost and benefit measures is attractive.

The two-dimensional results show savings of up to an order of magnitude in terms of degrees of freedom and up to a factor of 2 in terms of CPU time when compared to uniform refinement. In three dimensions, the *hp*-adaptation using c_{NZ} saves degrees of freedom and CPU time to achieve output convergence, especially when the output is corrected by its error estimate. Properly accounting for the effect of dimensionality in the computational expense is the reason for c_{NZ} out-performing c_{DOF} in three dimensions. Lastly, tuning of the DG solver can likely lead to savings in computational time.

At higher p-orders, the time taken to solve the primal and dual problems increases and the non-homogeneity of p affects the distribution of computational work amongst the processors. Therefore, dynamic load-balancing for *hp*-adaptive methods is important for efficient use of computational resources. However, such balance is not trivial and, in fact, is a topic of research rarely explored. The difficulty is that the computational effort for the residual operator is not constant between elements in the mesh and the performance of the preconditioner deteriorates when cells with strong coupling reside in different processors. The results presented here do not use dynamic load balancing as it is subject of ongoing research.

Acknowledgments

The authors acknowledge the support given by the University of Michigan to the development of this work and appreciate the computational resources provided by NASA which were essential to obtaining the results presented here. Funding for this research is provided by the Air Force Office of Scientific Research under contract number FA9550-10-C-0040.

[1] Levy, D. W., Zickuhr, T., Vassberg, J., Agrawal, S., Wahls, R. A., Pirzadeh, S., and Hemsch, M. J., "Data summary from the first AIAA computational fluid dynamics drag prediction workshop," *Journal of Aircraft*, Vol. 40, No. 5, 2003, pp. 875–882.

[2] Laflin, K. R., Vassberg, J. C., Wahls, R. A., Morrison, J. H., Brodersen, O., Rakowitz, M., Tinoco, E. N., and Godard, J.-L., "Summary of data from the second AIAA CFD drag prediction workshop," AIAA Paper 2004-0555, 2004.

[3] Frink, N. T., "Test case results from the 3rd AIAA drag prediction workshop," NASA Langley, 2007, `http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/Workshop3/final_results_jm.tar.gz`.

[4] 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.

[5] Giles, M. and Pierce, N., "Adjoint error correction for integral outputs," *Lecture Notes in Computational Science and Engineering: Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, Vol. 25, Springer, Berlin, 2002.

[6] 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.

[7] 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.

[8] Nemec, M., Aftosmis, M. J., and Wintzer, M., "Adjoint-based adaptive mesh refinement for complex geometries," AIAA Paper 2008-0725, 2008.

[9] Dwight, R. P., "Heuristic a posteriori estimation of error due to dissipation in finite volume schemes and application to mesh adaptation," *Journal of Computational Physics*, Vol. 227, 2008, pp. 2845–2863.

[10] Park, M. A., "Adjoint-based, three-dimensional error prediction and grid adaptation," AIAA Paper 2002-3286, 2002.

[11] Hartmann, R. and Houston, P., "Goal-oriented a posteriori error estimation for multiple target functionals," *Hyperbolic Problems: Theory, Numerics, Applications*, edited by T. Hou and E. Tadmor, Springer-Verlag, 2003, pp. 579–588.

[12] Fidkowski, K. J. and Darmofal, D. L., "An adaptive simplex cut-cell method for discontinuous Galerkin discretizations of the Navier-Stokes equations," AIAA Paper 2007-3941, 2007.

[13] Park, M. A. and Darmofal, D. L., "Validation of an output-adaptive, tetrahedral cut-cell method for sonic boom prediction," *AIAA Journal*, Vol. 48, No. 9, 2010, pp. 1928–1945.

[14] Bey, K. S., *An hp-adaptive discontinuous Galerkin method for hyperbolic conservation laws*, Ph.D. thesis, University of Texas at Austin, 1994.

[15] Heuveline, V. and Rannacher, R., "Duality-based adaptivity in the *hp*-finite element method," *Journal of Numerical Mathematics*, Vol. 11, No. 2, 2003, pp. 95–113.

[16] Rachowicz, W., Pardo, D., and Demkowicz, L., "Fully automatic *hp*-adaptivity in three dimensions," Tech. Rep. 04-22, ICES, 2004.

[17] Houston, P. and Süli, E., "A note on the design of *hp*-adaptive finite element methods for elliptic partial differential equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 194,

2005, pp. 229–243.

[18] Giani, S. and Houston, P., "High–order *hp*–adaptive discontinuous Galerkin finite element methods for compressible fluid flows," *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*, edited by N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, and K. Sørensen, Vol. 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Springer Berlin / Heidelberg, 2010, pp. 399–411.

[19] Burgess, N. K. and Mavriplis, D. J., "An *hp*-adaptive discontinuous Galerkin solver for aerodynamic flows on mixed-element meshes," *49th AIAA Aerospace Sciences Meeting and Exhibit*, 2011.

[20] Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive remeshing for compressible flow computations," *Journal of Computational Physics*, Vol. 72, 1987, pp. 449–466.

[21] Mavriplis, D. J., "Adaptive mesh generation for viscous flows using Delaunay triangulation," *Journal of Computational Physics*, Vol. 90, 1990, pp. 271–291.

[22] 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, 1997, pp. 475–491.

[23] Habashi, W. G., Dompierre, J., Bourgault, Y., Ait-Ali-Yahia, D., Fortin, M., and Vallet, M.-G., "Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles," *International Journal for Numerical Methods in Fluids*, Vol. 32, 2000, pp. 725–744.

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

[25] Fidkowski, K. J., *A Simplex Cut-Cell Adaptive Method for High–order Discretizations of the Compressible Navier-Stokes Equations*, PhD dissertation, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, June 2007.

[26] 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, 2007, pp. 1653–1672.

[27] Formaggia, L., Perotto, S., and Zunino, P., "An anisotropic a posteriori error estimate for a convection-diffusion problem," *Computing and Visualization in Science*, Vol. 4, 2001, pp. 99–104.

[28] Formaggia, L. and Perotto, S., "New anisotropic a priori error estimates," *Numerische Mathematik*, Vol. 89, 2001, pp. 641–667.

[29] Formaggia, L., Micheletti, S., and Perotto, S., "Anisotropic mesh adaptation with applications to CFD

problems," *Fifth World Congress on Computational Mechanics*, edited by H. A. Mang, F. G. Rammerstorfer, and J. Eberhardsteiner, Vienna, Austria, July 7-12 2002.

[30] Mavriplis, D. J. and Jameson, A., "Hermite-based mesh adaptation for functional outputs improvement in fluid flow simulation," *AIAA Journal*, Vol. 47, No. 8, 2009, pp. 1965–1976.

[31] Schneider, R. and Jimack, P. K., "Toward anisotropic mesh adaptation based upon sensitivity of a posteriori estimates," Tech. Rep. 2005.03, University of Leeds, School of Computing, 2005.

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

[33] Houston, P., Georgoulis, E. H., and Hall, E., "Adaptivity and a posteriori error estimation for DG methods on anisotropic meshes," *International Conference on Boundary and Interior Layers*, 2006.

[34] Ceze, M. and Fidkowski, K. J., "Output-driven anisotropic mesh adaptation for viscous flows using discrete choice optimization," *48th AIAA Aerospace Sciences Meeting and Exhibit*, 2010.

[35] Spalart, P. R. and Allmaras, S. R., "A one-equation turbulence model for aerodynamic flows," *30th Aerospace Sciences Meeting and Exhibit*, No. AIAA-92-0439, AIAA, 1992.

[36] 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.

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

[38] Bassi, F. and Rebay, S., "GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations," *Discontinuous Galerkin Methods: Theory, Computation and Applications*, edited by K. Cockburn and Shu, Springer, Berlin, 2000, pp. 197–208.

[39] Ceze, M. and Fidkowski, K. J., "A robust adaptive solution strategy for high-order implicit CFD solvers," *20th AIAA Computaional Fluid Dynamics Conference*, AIAA, 2011.

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

[41] Fidkowski, K. J., *A High–order Discontinuous Galerkin Multigrid Solver for Aerodynamic Applications*, MS thesis, M.I.T., Department of Aeronautics and Astronautics, June 2004.

[42] 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.

[43] Giles, M. B., Duta, M. C., Müller, J.-D., and Pierce, N. A., "Algorithm developments for discrete adjoint methods," *AIAA Journal*, Vol. 41, No. 2, 2003, pp. 198–205.

[44] Mader, C. A., Martins, J. R., Alonso, J. J., and van der Weide, E., "ADjoint: An approach for the rapid development of discrete adjoint solvers," *AIAA Journal*, Vol. 46, No. 4, 2008, pp. 863–873.

[45] Rannacher, R., "Adaptive Galerkin finite element methods for partial differential equations," *Journal of Computational and Applied Mathematics*, Vol. 128, 2001, pp. 205–233.

[46] Barth, T. and Larson, M., "A posteriori error estimates for higher order Godunov finite volume methods on unstructured meshes," *Finite Volumes for Complex Applications III*, edited by R. Herban and D. Kröner, Hermes Penton, London, 2002, pp. 41–63.

[47] Solín, P. and Demkowicz, L., "Goal-oriented *hp*-adaptivity for elliptic problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 193, 2004, pp. 449–468.

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

[49] Hartmann, R., "Adjoint consistency analysis of discontinuous Galerkin discretizations," *SIAM Journal on Numerical Analysis*, Vol. 45, No. 6, 2007, pp. 2671–2696.

[50] Venditti, D. A. and Darmofal, D. L., "Grid adaptation for functional outputs: application to two-dimensional inviscid flows," *Journal of Computational Physics*, Vol. 176, No. 1, 2002, pp. 40–39.

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

[52] DLR Germany, "TauBench - IPACS," `http://www.ipacs-benchmark.org`.

[53] Fidkowski, K. J., Ceze, M. A., and Roe, P. L., "Entropy-based drag error estimation and mesh adaptation in two dimensions," *AIAA Journal of Aircraft*, 2012, Accepted.

[54] Oswatitsch, K., *Gas Dynamics*, Academic Press, New York, 1956.

[55] "ADIGMA – A European project on the development of adaptive higher-order variational methods for aerospace applications," *47th AIAA Aerospace Sciences Meeting and Exhibit*, edited by AIAA, 2009.

[56] Frink, N. T., "3rd AIAA CFD drag prediction workshop gridding guidelines," NASA Langley, 2007, `http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/Workshop3/gridding_guidelines.html`.

[57] Volpe, E. and de C. Santos, L., "Boundary and internal conditions for adjoint fluid flow problems," *Journal of Engineering Mathematics*, Vol. 65, No. 1, 2009.