AIAA Journal
Vol. 55, No. 6, June 2017

Mesh Topology Preserving Boundary-Layer Adaptivity Method for Steady Viscous Flows

D. Moro,[*] N. C. Nguyen,[†] J. Peraire,[‡] and M. Drela[‡]
Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

DOI: 10.2514/1.J054961

The efficiency and accuracy of viscous flow simulations depend crucially on the quality of the boundary-layer mesh. Too coarse meshes can result in inaccurate predictions and in some cases lead to numerical instabilities, whereas too fine meshes produce accurate predictions at the expense of long simulation times. Constructing an optimal (or near-optimal) boundary-layer mesh has been recognized as an important problem in computational fluid dynamics. For few simple flows, one may be able to construct such a mesh a priori before simulation. For most viscous flow simulations, however, it is difficult to generate such mesh in advance. In this paper, a boundary-layer adaptivity method is developed for the efficient computation of steady viscous flows. This method turns the problem of determining the location of the mesh nodes into a set of equations that are solved simultaneously with the flow equations. The mesh equations are designed so that the boundary-layer mesh adapts to the viscous layers as the flow solver marches toward the converged solution. Extensive numerical experiments are presented to demonstrate the performance of the method.

Nomenclature

c	=	local speed of sound
c^*	=	critical speed of sound
c_f	=	friction coefficient
c_p	=	pressure coefficient
$\boldsymbol{E}$	=	total energy
$\boldsymbol{G}$	=	gradient of mesh mapping
$\mathcal{G}$	=	mapping from reference to physical domain
$\boldsymbol{H}$	=	total enthalpy
$\boldsymbol{h}$	=	thermodynamic enthalpy
$\{h_i\}$	=	stack distribution of the boundary-layer mesh
$h(\boldsymbol{x})$	=	element size approximation
k_δ	=	safety scale for the thickness indicator
M_∞	=	Mach number
$\hat{\boldsymbol{n}}$	=	extrusion direction for the boundary-layer mesh
n_{norm}	=	number of elements across the boundary-layer mesh
$\boldsymbol{n}_r$	=	normal vector in the reference domain
$\boldsymbol{n}_x$	=	normal vector in the physical domain
P	=	pressure
Pr, Pr_t	=	laminar and turbulent Prandtl number
p	=	polynomial order
R	=	perfect gas constant
$Re_{(\cdot)}$	=	Reynolds number, based on $(\cdot)$
T	=	temperature
$\boldsymbol{v}$	=	fluid velocity
$\tilde{\nu}$	=	Spalart–Allmaras working variable
$\boldsymbol{v}_G$	=	mesh velocity
α	=	angle of attack
δ	=	normal scaling of the boundary layer
δ_{BL}	=	estimate of the boundary-layer thickness
ε	=	artificial viscosity
κ	=	heat conductivity
μ	=	dynamic viscosity
μ_{ref}, T_{ref}, C	=	constants for Sutherland's law
μ_t	=	eddy viscosity
μ_δ	=	smoothing parameter of normal scaling equation
ρ	=	density
τ_δ	=	time constant of normal scaling equation
χ	=	mesh deformation measure
χ_e	=	integrated element deformation

Received 16 December 2015; revision received 24 June 2016; accepted for publication 1 February 2017; published online 27 April 2017. Copyright © 2017 by Massachusetts Institute of Technology. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. All requests for copying and permission to reprint should be submitted to CCC at www.copyright.com; employ the ISSN 0001-1452 (print) or 1533-385X (online) to initiate your request. See also AIAA Rights and Permissions www.aiaa.org/randp.

[*]Ph.D. Student, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue. Student Member AIAA.

[†]Research Scientist, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue. Member AIAA.

[‡]Professor, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue. Fellow AIAA.

I. Introduction

ONE of the most important tasks in computational fluid dynamics (CFD) is the generation of suitable meshes for the problem of interest. Herein, we focus on mesh generation for high Reynolds number flows. One peculiarity of these flows is the presence of very thin boundary layers in which the solution varies rapidly. The boundary-layer thickness is a function of the Reynolds number Re and scales roughly with $Re^{1/2}$ for laminar flows and $Re^{4/5}$ for turbulent flows. The latter is commonly broken into three regions (viscous sublayer, buffer layer, and log layer), each of which grows at a different rate depending on the flow conditions. The presence of boundary layers and the disparity of length scales require anisotropic meshes, where the elements are oriented along the flow direction. Indeed, the efficiency and accuracy of viscous flow simulations depend crucially on the quality of a boundary-layer mesh. Too coarse meshes can result in inaccurate predictions and in some cases lead to numerical instabilities, whereas too fine meshes produce accurate predictions at the expense of long simulation times. For few simple flows, one may be able to construct suitable meshes a priori before performing the simulations. However, for most viscous flow simulations of interest, it is difficult to generate meshes in advance because it is difficult to estimate the thickness of the boundary layers before the simulations are conducted.

It is common to solve the mesh generation problem by constructing a sequence of meshes and solutions iteratively, a procedure known as *mesh adaptation*. Mesh adaptation can be divided into three classes of techniques: h adaptivity (adapting the size of the elements), p adaptivity (adapting the degree of the local polynomials), and r adaptivity (relocating the mesh nodes.) Various other adaptive strategies can be devised by combining these techniques.

In h adaptivity, starting from an initial coarse mesh, the adaptivity module and the CFD solver are called iteratively in an alternating fashion until an accurate solution is obtained on a final refined mesh. In this procedure, the critical step is the specification of the mesh size, which is commonly based on some measure of the error. A common choice for the latter is to use derivative reconstructions to minimize the interpolation error, either alone [1–3] or combined with output error estimates [4]. Only very recently, a different approach based on output error estimates (in particular, dual weighted residual or DWR) and element splitting and sampling has been proposed by Yano [5] for unstructured meshes or Ceze and Fidkowski [6] for structured meshes. The alternating between the solver and the adaptivity module serves to simplify the implementation and save computational cost. However, it may face difficulty to converge to the final solution if intermediate meshes are too coarse to capture sharp features of the solution, such as shock waves and boundary layers. This has prompted research into increasing the robustness of the CFD solver on meshes without adequate resolution, such as modified turbulence models [7–10] and advanced nonlinear solvers [11,12].

In p adaptivity, a similar adaptation procedure is carried out, except that the approximation order inside each element can be varied depending on the smoothness of the solution. In general, however, p adaptivity is seldom used alone but rather in the form of h/p adaptivity [13]. The h/p adaptivity technique has been used with discontinuous Galerkin finite element methods, where the implementation is relatively simple. The technique requires an error indicator to drive the change of approximation order, which can be based on extrapolation using a sequence of meshes [13], a resolution indicator [14], a postprocessed solution [15], or a DWR-type error estimate [6], among others.

In r adaptivity, the adaptation procedure relocates the nodes of the mesh without changing the mesh connectivity. This process relies on the definition of a monitor function to yield the location of the nodes. The particular choice of a monitor function is discussed at length in the reviews by Budd et al. [16]. For CFD analysis, r adaptivity has been applied to unsteady inviscid compressible flow problems, as well as free shear flows [17,18]. As is the case with h and p adaptivity, the update of the mesh is produced in between iterations of the flow solution and is designed to be oblivious to the flow features. This makes the mesh adaptation reasonably general, but overly complicated for the case of wall-bounded flows. However, there are r-apativity methods that take advantage of the flow features, like the differential boundary-layer solvers of Drela [19] and Allmaras [20].

In this paper, we tackle the problem of generating a boundary-layer mesh for steady viscous flows by means of an r-apativity formulation that evolves the mesh "on the fly" and is inspired by the differential boundary-layer solvers of Drela [19] and Allmaras [20]. In our method, the locations of the mesh nodes are determined by the solution of a set of equations that are solved simultaneously with the flow equations. The mesh equations are designed so that the boundary-layer mesh adapts to the viscous layers as the flow solver marches toward convergence. The main ingredients of our method include 1) a surface partial differential equation (PDE) for the boundary-layer thickness, which allows us to adapt our structured boundary-layer mesh to the flow boundary layer, 2) the deformation of the mesh using the procedure recently introduced by Roca et al. [21] and Gargallo-Peiro et al. [22], and 3) a fully coupled high-order implicit solver based on the hybridizable discontinuous Galerkin (HDG) method [23,24] to discretize the coupled system of equations. Our method ensures the robustness of the flow solver and the quality of the final solution because the boundary-layer mesh is simultaneously adapted to the viscous layers as the flow solver marches toward the steady-state solution. We present extensive numerical experiments to demonstrate the performance of the method.

The paper is organized as follows. In Sec. II, we describe the main components of the viscous flow discretization, including the governing equations. In Sec. III, we develop the mesh adaptivity technique to provide an efficient mesh for the computation of viscous flows. In Sec. IV, we present a set of numerical examples for laminar and turbulent flows to validate the proposed method. Finally, in Sec. V, we discuss extensions and future work.

II. Implicit Flow Solver

In this section, we present the implicit high-order approach that we use to solve the compressible Navier–Stokes equations on a deformable mesh. We begin by introducing the Navier–Stokes equations for laminar flows and the Reynolds-averaged Navier–Stokes (RANS) equations with the modified Spalart–Allmaras equation [9,25] for turbulent flows. Next, we review the artificial viscosity model used to stabilize the numerical solution of viscous flows with shock waves [26]. Then, we employ the arbitrary Lagrangian–Eulerian (ALE) formulation [23,27] to treat moving domains. Finally, we give a brief overview of the HDG scheme used to discretize these equations.

A. Navier–Stokes Equations

We consider the compressible Navier–Stokes equations for laminar flows

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{F}_{\text{inv}} = \nabla \cdot \boldsymbol{F}_{\text{visc}} \tag{1}$$

where

$$\boldsymbol{u} = \begin{bmatrix} \rho \\ \rho v_i \\ \rho E \end{bmatrix}, \qquad \boldsymbol{F}_{\text{inv}} = \begin{bmatrix} \rho v_i \\ \rho v_i v_j + P\delta_{ij} \\ \rho v_i H \end{bmatrix},$$

$$\boldsymbol{F}_{\text{visc}} = \begin{bmatrix} 0 \\ \tau_{ij} \\ \sum_{j=1}^{d} \tau_{ij} u_j + q_i \end{bmatrix} \tag{2}$$

$$\tau_{ij} = \mu\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial v_k}{\partial x_k}\right), \qquad q_i = \kappa\frac{\partial T}{\partial x_i} \tag{3}$$

Here, ρ represents the density, v_i is the ith component of the velocity, E is the total specific energy, P is the pressure, and $H = E + P/\rho$ is the total specific enthalpy. We assume that the flow obeys the ideal gas law:

$$P = \rho R T \tag{4}$$

where R is the specific gas constant of air and T is the temperature. The coefficients μ and κ are the dynamic viscosity and heat conductivity, respectively. Here, $\mu = \mu(T)$ according to Sutherland's law:

$$\mu = \mu_{\text{ref}}\frac{T_{\text{ref}} + C}{T + C}\left(\frac{T}{T_{\text{ref}}}\right)^{3/2} \tag{5}$$

where $C = 120$ K and the pair (T_{ref}, μ_{ref}) denotes a reference temperature and viscosity that we take to be equal to the freestream values: (T_∞, μ_∞). The thermal conductivity κ is related to the viscosity through the Prandtl number, which we assume to be constant:

$$Pr = \frac{c_p \mu}{\kappa} = 0.72 \tag{6}$$

The Navier–Stokes equations are written in nondimensional form using the following reference states: $\rho_{\text{ref}} = \rho_\infty$, $\rho u_{\text{ref}} = \rho_\infty \|\boldsymbol{v}_\infty\|$, and $\rho E_{\text{ref}} = \rho_\infty \|\boldsymbol{v}_\infty\|^2$.

To close the problem, we need to prescribe the boundary conditions. At the viscous wall, we impose an adiabatic wall condition, namely, zero velocity and zero heat flux. At the outer boundary, we impose a far-field characteristic boundary condition for steady viscous flows (see [24] for a detailed discussion about the boundary conditions).

B. Reynolds-Averaged Navier–Stokes Modeling

The effect of turbulence on the flowfield is modeled using the RANS equations together with Boussinesq's analogy for the eddy viscosity. In this way, the only modification required to the system described by Eqs. (1–3) is the introduction of the turbulent viscosity and thermal conductivity, so that the stresses are given by

$$\tau_{ij} = (\mu + \mu_t)\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial v_k}{\partial x_k}\right), \qquad q_i = \left(\frac{\mu}{Pr} + \frac{\mu_t}{Pr_t}\right)\frac{\partial h}{\partial x_i} \tag{7}$$

Here, μ_t is the eddy viscosity, Pr_t is the turbulent Prandtl number (here we assume $Pr_t = 0.9$), and $h = H - (1/2)\boldsymbol{v}^2$ is the thermodynamic enthalpy. Note that, now, the vector of unknowns $\boldsymbol{u}$ denotes the average density, momentum, and total energy fields and we retain time-dependent terms to facilitate convergence to steady state.

The use of Boussinesq's analogy requires a model for the eddy viscosity μ_t that appears in Eq. (7). For this, we use a version of the Spalart–Allmaras (SA) turbulence model [28], which incorporates modifications [9,10] that avoid stability problems when a high-order discretization is used. The resulting model, as presented by Chaurasia [25], is based on the following PDE for the working variable $\rho\tilde{\nu}$:

$$\begin{aligned}\frac{\partial \rho\tilde{\nu}}{\partial t} + \nabla\cdot(\rho\boldsymbol{v}\tilde{\nu}) &= \rho(s_P - s_D) + \frac{1}{\sigma}\nabla\cdot[\rho(\nu + \psi\nu)\nabla\tilde{\nu}] \\ &+ \frac{c_{b2}}{\sigma}\rho(\nabla\tilde{\nu})^2 + \frac{1}{\sigma}(\nu + \psi\nu)\nabla\rho\cdot\nabla\tilde{\nu}\end{aligned} \tag{8}$$

where the different terms on the right-hand side of the equation are defined next. In particular, let

$$\begin{aligned}\chi &= \frac{\rho\tilde{\nu}}{\mu}, \qquad \psi = \chi\left(\frac{\arctan(b\chi)}{\pi} + \frac{1}{2}\right) + c_0, \\ c_0 &= \frac{1}{2} - \frac{\arctan(b)}{\pi}\end{aligned} \tag{9}$$

Here, χ denotes a nondimensional version of the working variable and ψ denotes a regularized version of it that approximates the nondifferentiable operation $\max(0,\chi)$. Such regularization is controlled by the constant b and the associated term $c_0(b)$. In addition, the production s_P and destruction s_D terms in Eq. (8) are defined as

$$s_P = c_{b1}\tilde{S}\psi\nu, \qquad s_D = c_{w1}f_w\left(\frac{\psi\nu}{d}\right)^2 \tag{10}$$

and require the following auxiliary relationships:

$$\begin{aligned}S &= \sqrt{2\Omega_{ij}\Omega_{ij}}, \qquad \bar{S} = \frac{\psi\nu}{(\kappa d)^2}f_{v2}, \qquad f_{v1} = \frac{\psi^3}{\psi^3 + c_{v1}^3}, \\ f_{v2} &= 1 - \frac{\psi}{1 + \psi f_{v1}}\end{aligned} \tag{11}$$

$$\tilde{S} = 0.1S + (\bar{S} + 0.9S)\left(\frac{\arctan[b(\bar{S}/S + 0.9)]}{\pi} + \frac{1}{2}\right) + cS \tag{12}$$

$$\begin{aligned}\bar{r} &= \frac{\psi\nu}{\tilde{S}(\kappa d)^2}, \\ r &= r_{\lim} - (r_{\lim} - \bar{r})\left(\frac{\arctan[b(r_{\lim} - \bar{r})]}{\pi} + \frac{1}{2}\right) - c\end{aligned} \tag{13}$$

$$f_w = g\left[\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right]^{1/6}, \qquad g = r + c_{w2}(r^6 - r) \tag{14}$$

where d is the distance to the closest wall. This version of the SA model uses the same constants as the original model [28]: $c_{b1} = 0.1355$, $c_{b2} = 0.622$, $c_{v1} = 7.1$, $\sigma = 2/3$, $c_{w1} = 3.2391$, $c_{w2} = 0.3$, $c_{w3} = 2$, $\kappa = 0.41$, and $r_{\lim} = 10$. Note that here κ denotes the von Kármán constant, which should not be confused with the thermal conductivity. In addition, the regularization parameter b is set to $b = 100$ as suggested by Chaurasia [25].

The relationship between the working variable and the eddy viscosity that approximates the Reynolds stresses in Eq. (7) is given by

$$\mu_t = \mu\psi f_{v1} \tag{15}$$

The system of PDEs that governs the flow is obtained by appending the modified version of the SA model [Eq. (8)] to the RANS equations [Eqs. (1–3) with the Reynolds stresses in Eq. (7)]. The system is written in nondimensional form using the same reference magnitudes used for the Navier–Stokes equations, plus a reference value for the working variable given by $\rho\tilde{\nu}_{\text{ref}} = \mu_\infty$. To close the problem, we need to specify the correct boundary conditions. In the case of the RANS–SA equations, these are similar to the Navier–Stokes case except for the addition of a condition for the SA model. In particular, we set $\rho\tilde{\nu} = \mu_\infty$ at the far field using a characteristic decomposition and $\rho\tilde{\nu} = 0$ at a viscous wall. These boundary conditions are imposed weakly by the discretization scheme.

C. Shock Capturing

The solution of compressible flows involving shocks with high-order methods requires some form of stabilization. In this paper, we use an artificial viscosity technique based on the divergence of the velocity $(\nabla\cdot\boldsymbol{v})$ as a shock indicator [26]. The stabilization enters the governing equations through an artificial dissipation term that is added to the right-hand side of the conservation laws:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla\cdot\boldsymbol{F}_{\text{inv}} = \nabla\cdot\boldsymbol{F}_{\text{visc}} + \nabla\cdot\varepsilon(\boldsymbol{u})\nabla\boldsymbol{u}_{\text{AV}}(\boldsymbol{u}) \tag{16}$$

Here, $\boldsymbol{u}_{\text{AV}}(\boldsymbol{u}) = \{\rho, \rho\boldsymbol{v}, \rho H\}$ and $\varepsilon(\boldsymbol{u})$ is the artificial viscosity given by

$$\varepsilon(\boldsymbol{u}) = \left(k_h\frac{\tilde{h}}{p}\right)\sqrt{\boldsymbol{v}\cdot\boldsymbol{v} + c^2}f(\tilde{s}^*) \tag{17}$$

where

$$\tilde{s}^*(\boldsymbol{u}) = -\frac{(k_h\tilde{h}/p)\nabla\cdot\boldsymbol{v}}{c^*}, \qquad f(x) = \frac{\log(1 + \exp[\alpha(x - \beta)])}{\alpha} \tag{18}$$

The model is closed by setting $k_h = 1.5$, $\alpha = 10^4$, and $\beta = 0.01$ [26].

The artificial viscosity model requires an approximation to the local element size distribution across the whole domain $\tilde{h}(\boldsymbol{x})$. Here, we follow the original paper [26] and compute a continuous piecewise linear reconstruction of the minimum height of the elements of the mesh.

D. Arbitrary Lagrangian–Eulerian Formulation

The governing equations are discretized on a mesh that evolves with the solution until steady state is reached. This implies that we are solving a set of PDEs on a domain that deforms in time. For this reason, we use an ALE formulation as described by Persson et al. [27].

We consider a mapping $\mathcal{G}$ from a reference domain $(\Omega_x \in \mathbb{R}^d)$ to the physical domain $(\Omega_r \in \mathbb{R}^d)$ denoted by

$$\boldsymbol{x} = \mathcal{G}(\boldsymbol{r}, t), \qquad \boldsymbol{r}\in\Omega_r, \qquad \boldsymbol{x}\in\Omega_x \tag{19}$$

which is differentiable in the arguments. In particular, let

$$\boldsymbol{G} = \frac{\partial\mathcal{G}}{\partial\boldsymbol{r}}, \qquad \boldsymbol{v}_G = \frac{\partial\mathcal{G}}{\partial t} \tag{20}$$

denote the gradients of the mapping. This allows us to transform a generic conservation law written as a first-order system in the physical space

$$\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot \boldsymbol{A}(\boldsymbol{u}, \boldsymbol{Q}) = \boldsymbol{s}, \quad \forall\, \boldsymbol{x} \in \Omega_x \tag{21}$$

$$\boldsymbol{Q} - \nabla \boldsymbol{u} = 0 \quad \forall\, \boldsymbol{x} \in \Omega_x \tag{22}$$

into an equivalent conservation statement in the reference space of the form

$$\frac{\partial g\boldsymbol{u}}{\partial t} + \nabla_r \cdot \{g\boldsymbol{G}^{-1}(\boldsymbol{A}(\boldsymbol{u}, \boldsymbol{G}^{-1}\boldsymbol{Q}_r) - \boldsymbol{u} \otimes \boldsymbol{v}_G)\} = g\boldsymbol{s}, \quad \forall\, \boldsymbol{r} \in \Omega_r \tag{23}$$

$$\boldsymbol{Q}_r - \nabla_r \boldsymbol{u} = 0, \quad \forall\, \boldsymbol{r} \in \Omega_r \tag{24}$$

where $g = \det(\boldsymbol{G})$ and the divergence and gradient operators work on the reference coordinates $\boldsymbol{r}$. Here, the flux $\boldsymbol{A} = \boldsymbol{F}_{\text{inv}}(\boldsymbol{u}) - \boldsymbol{F}_{\text{visc}}(\boldsymbol{u}, \boldsymbol{Q})$ includes both inviscid and viscous contributions.

Notice that the ALE formulation leads to a modified conservation law with some added complexity on the fluxes and sources. These extra terms, like the velocity of the mesh $\boldsymbol{v}_G$ or the cofactor of the mapping $g\boldsymbol{G}^{-1}$, need to be accounted for appropriately when computing the eigendecomposition of the inviscid portion of the ALE fluxes, which is used to impose the boundary conditions and develop stable Riemann solvers. In this case, such eigendecomposition is given by

$$\frac{\partial(\boldsymbol{A}_{\text{ALE}} \cdot \boldsymbol{n}_r)}{\partial \boldsymbol{u}} = \|g\boldsymbol{G}^{-1}\boldsymbol{n}_r\| \boldsymbol{K}(\boldsymbol{\Lambda} - v_{Gn}\boldsymbol{I})\boldsymbol{K}^{-1} \tag{25}$$

where v_{Gn} is the normal velocity of the mesh in the physical space given by

$$v_{Gn} = \boldsymbol{v}_G \cdot \frac{g\boldsymbol{G}^{-1}\boldsymbol{n}_r}{\|g\boldsymbol{G}^{-1}\boldsymbol{n}_r\|} \tag{26}$$

and $\boldsymbol{K}$ and $\boldsymbol{\Lambda}$ denote the eigenvector and eigenvalue matrix of the original inviscid flux [29].

In addition, the fact that the mesh deformation is solved simultaneously with the flow implies that the geometric entities that enter the ALE fluxes (velocity of the mapping, Jacobian of the mapping, etc.) and its gradients need to be defined as functions of the degrees of freedom that represent the mesh.

In some instances, the formulation can be augmented with the use of a Geometric Conservation Law (GCL) to correct for possible errors in the time integration of the geometry [27]. However, this only makes a small difference in unsteady flows [27,30] and is irrelevant when steady-state solutions are sought. For this reason, we did not implement the GCL in this work.

E. Numerical Discretization of the Flowfield Equations

The ALE equations in conservative form are discretized using a high-order hybridizable discontinuous Galerkin method [9,23,24,31,32]. The HDG method allows the use of a high-order approximation and the simple treatment of hybrid structured/unstructured meshes, as required for the boundary-layer flows that we are interested in. In addition to this, the HDG method was designed to reduce the number of globally coupled unknowns and produce an approximation to the gradient of the solution that converges optimally for diffusion problems, while being stable in the convective regime. In this work, we follow closely the HDG discretization of the Navier–Stokes equations presented by Peraire et al. [24] and later extended to the SA equations [9]. Finally, the problem is discretized in time using a backward Euler scheme to evolve to steady state.

III. Boundary-Layer Adaptivity

The equations that describe the flow are discretized on a moving mesh whose topology is fixed but is adapted to the boundary-layer features.

A. Measuring the Boundary-Layer Thickness

One of the advantages of adapting to the boundary layer versus other flow features, like shock waves or wakes, is the fact that the location of the boundary layer is known. In particular, in the case of attached and moderately separated flows, the boundary layer is contained in a region close to solid walls, which means that the only variable that needs to be determined is its thickness. The boundary-layer thickness is not uniform and can range from being very small near the stagnation point to the order of the airfoil thickness close to the trailing edge. Thus, the first technical challenge we need to address is how to determine the boundary-layer thickness.

The boundary layer ends where the viscous effects are negligible or when a certain percentage of the external velocity is recovered. In practice, these criteria are hard to implement because they require an exhaustive search for the point at which the condition is met.

When it comes to implementation, especially in the context of adaptivity, we can trade exactness for simplicity. In particular, we are interested in approximate indicators based on an algebraic relationship involving integral boundary-layer quantities, like the ones routinely used in integral boundary-layer (BL) solvers. In this work, we will use the one proposed by Drela [33], which is given by the following equation:

$$\delta_{\text{BL}} = \theta_k\left(A + \frac{B}{H_k - 1}\right) + C\delta_k^* \tag{27}$$

$$A = 3.15, \qquad B = 1.72, \qquad C = 1.0 \tag{28}$$

Here, δ_{BL} is an estimate for the boundary-layer thickness and δ_k^*, θ_k, and H_k represent the kinematic displacement thickness, momentum thickness, and shape parameter, respectively, which are defined as

$$\delta_k^* = \int_0^{y_e}\left(1 - \frac{u}{u_e}\right)\mathrm{d}y, \qquad \theta_k = \int_0^{y_e}\left(1 - \frac{u}{u_e}\right)\frac{u}{u_e}\,\mathrm{d}y,$$
$$H_k = \frac{\delta_k^*}{\theta_k} \tag{29}$$

The form of Eq. (27) and the calibrated constants A, B, C are set to give a good estimate for δ_{BL} for a very wide range of shear layer profiles. All three terms contribute comparably in a typical laminar or turbulent wall boundary layer. The B term dominates when H_k approaches unity, as in wakes and strongly accelerated turbulent wall boundary layers, whereas the C term dominates for massively separated boundary layers where $H_k \gg 1$.

The δ_k^*, θ_k integrals depend on u and u_e, which denote the velocity in the direction tangent to the wall and its asymptotic value at the edge of the boundary layer, respectively. The integration variable is y, which is the direction normal to the wall. Note that, because u quickly asymptotes to u_e, the upper limit of integration is truncated to a distance y_e of the order of the boundary-layer thickness. It is possible to remove this ambiguity by using a velocity recovered through an integration of vorticity [34,35], however, this requires a double integration for δ_k^* and θ_k, which complicates the implementation without adding capability. By using Eq. (27), we have transferred the complexity of measuring the boundary-layer thickness into the numerical approximation of a series of integrals across the boundary layer. This requires us to provide a way to 1) extract the boundary-layer profiles from the mesh and 2) define the state at the edge of the boundary layer in an unambiguous way. Both of these conditions are satisfied by the mesh structure that we propose next.

B. Boundary-Layer Mesh and External Mesh

We propose to solve the problem of computing δ_k^* and θ_k by providing structure to the region near solid walls. More precisely, we explicitly require a hybrid mesh composed of stacks of quadrilateral elements extruded from the wall in the boundary-layer region and triangles in the outer inviscid region. In this way, extracting the boundary-layer profiles within each stack is a trivial task using the appropriate data structures. Similarly, the "flow state" at the edge of the boundary layer can be extracted from the elements at the top of the stack. We will show later how this is not a problem if the mesh is computed as part of the solution, even if the flow separates. In all the results we present in this paper, the external mesh is unstructured and composed of triangles, which allows the use of standard mesh generation tools. The mesh is generated once and its connectivities are kept fixed.

For illustration purposes, we include a sketch of possible meshes in Fig. 1. In the case of a closed curve without angles (e.g., a cylinder or turbine blade), the boundary-layer mesh resembles an O mesh. If the geometry has a trailing edge, the boundary-layer mesh becomes a C mesh. In the case where the geometry is not closed (e.g., flat plate), the boundary layer is a simple rectangular structured mesh.

The use of a structured quadrilateral mesh on the boundary-layer domain has advantages beyond the convenience of the associated data structure. In particular, it appears that the combination of high-order methods and quadrilateral meshes in the vicinity of the wall delivers solutions of higher quality with less degrees of freedom and less dependency on grid stretching [36,37]. However, the use of a high-order discretization in space requires the use of a mesh that represents the geometry with a high level of fidelity [38].

In this work, we represent the geometry using isoparametric elements. The degrees of freedom of the mesh are the locations of the mesh nodes that are continuous (e.g., conformal) across the element interfaces. We denote by $\boldsymbol{x}^{\mathrm{bl}}$ the location of the boundary-layer mesh node and by $\boldsymbol{x}^{\mathrm{ext}}$ the location of the external mesh nodes. In what follows, we present the equations that govern the evolution of the boundary-layer mesh and the deformation of the external mesh.

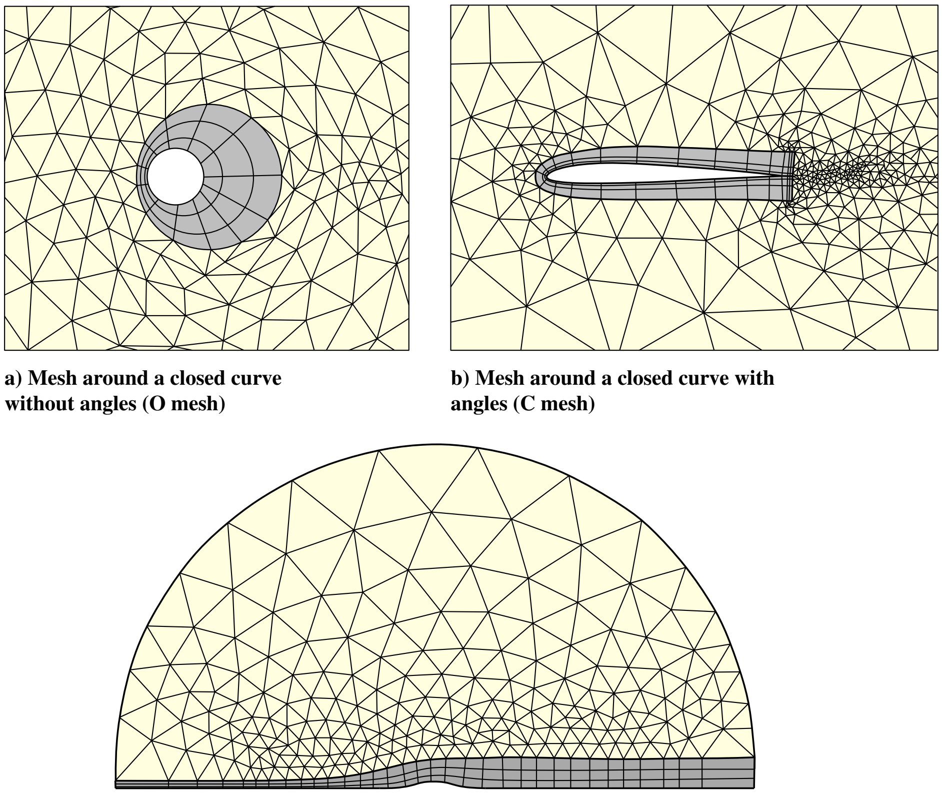
C. Normal Scaling Equation

The first governing equation is associated with the so-called normal scaling δ, which is an approximation of the thickness of the boundary-layer mesh. Ideally, δ should be a scalar field defined on the wall that targets the value of δ_{BL} in Eq. (27). Here, we will use the surface PDE developed by Allmaras [20] for an adaptive differential boundary-layer code, given by

$$\frac{\partial \delta}{\partial t} = \frac{k_\delta \delta_{\mathrm{BL}} - \delta}{\tau_\delta} + \mu_\delta \Delta_\Gamma \delta \tag{30}$$

where Δ_Γ represents the Laplace–Beltrami operator. This equation contains a reaction term that drives δ toward $k_\delta \delta_{\mathrm{BL}}$, a regularization term in space by diffusion, and a time relaxation. Here, $k_\delta > 1$ is a safety factor that ensures that the boundary-layer mesh is thicker than the fluid boundary layer itself, τ_δ is a constant that controls the response time of the mesh to changes in the flow, and μ_δ is a constant that governs the smoothness of the solution along the surface. To close the model, we need to define the constants k_δ, μ_δ, and τ_δ. For all the results in this work, we take $k_\delta = 1.5$ for laminar flows, $k_\delta = 2$ for turbulent flows, $\tau_\delta = 2\Delta t$, and $\mu_\delta = 0.1\Delta x^2/\tau_\delta$. Here, Δt represents the time step and Δx represents an approximation of the surface element length. This choice ensures that δ_h evolves fast enough to track the growth of the boundary layer toward steady state.

The surface PDE Eq. (30) relates δ to the flow quantities through the definition of the normal scaling indicator δ_{BL}. We discretize the equation for δ using the surface finite element technique by Dziuk and Elliott [39], with a minor modification to accommodate the high-order representation of the boundary [40]. The details of the discretization are as follows. Let Γ_h denote a conformal high-order approximation to the solid wall manifold. In it, each element $K^\Gamma \in \Gamma_h$



a) Mesh around a closed curve without angles (O mesh)

b) Mesh around a closed curve with angles (C mesh)

c) Mesh around an open curve (Rectangular mesh)

Fig. 1 Sketch of the mesh topology for different kinds of geometries. For visualization purposes, the boundary-layer and external meshes are colored differently.

is defined through an isoparametric mapping (or chart) from a reference domain in Euclidean space $\xi \in \mathbb{R}^n$ to the physical space $\boldsymbol{x}(\xi) \in \mathbb{R}^{n+1}$. The union of these mappings defines an atlas of the approximate manifold, which can, in principle, be nondifferentiable at element boundaries. In the two-dimensional (2-D) cases treated in this papers, this is equivalent to a piecewise polynomial representation of the boundary of the domain. On it, we can define the standard differential geometry tools, such as the covariant metric tensor

$$(\boldsymbol{G}_s)_{ij} = \frac{\partial \boldsymbol{x}}{\partial \xi_i} \cdot \frac{\partial \boldsymbol{x}}{\partial \xi_j}$$

its contravariant counterpart $(\boldsymbol{G}_s)^{ij} = (\boldsymbol{G}_s)^{-1}_{ij}$, and its determinant $g_s = \det[(\boldsymbol{G}_s)_{ij}]$.

Using this mesh as support, we construct the finite element space for the solution $\mathcal{U}_h$, consisting of piecewise linear functions on Γ_h:

$$\mathcal{U}_h = \{v \in C^0(\Gamma_h) : v|_{K^\Gamma} \in P^1(K^\Gamma), \forall K^\Gamma \in \Gamma_h\} \tag{31}$$

In addition, we need to define the following inner products:

$$(u, v)_{K^\Gamma} = \int_{K^\Gamma} uv \, \mathrm{d}\Omega, \quad \text{and} \quad \langle u, v\rangle_{\partial K^\Gamma} = \int_{\partial K^\Gamma} uv \, \mathrm{d}s \tag{32}$$

To derive the weak form of Eq. (30), we integrate by parts against the test space $\mathcal{V}_h$. Assuming that the test and trial spaces coincide, $\mathcal{U}_h = \mathcal{V}_h$, the semidiscrete weak form for the evolution of δ_h in time reads find $\delta_h \in \mathcal{U}_h$, such that

$$\begin{aligned}\sum_e \left(\frac{\partial \delta_h}{\partial t}, v\right)_{K^\Gamma} &- \sum_e \left(\frac{k_\delta \delta_{\mathrm{BL}} - \delta_h}{\tau_\delta}, v\right)_{K^\Gamma} \\ &+ \sum_e (\mu_\delta (\boldsymbol{G}_s)^{ij} \nabla_i \delta_h, \nabla_j v)_{K^\Gamma} \\ &- \sum_e \langle \mu_\delta (\boldsymbol{G}_s)^{ij} \nabla_i \delta_h \boldsymbol{n}_i, v\rangle_{\partial K^\Gamma} = 0, \quad \forall\, v \in \mathcal{V}_h \end{aligned} \tag{33}$$

Here, $\boldsymbol{n}_i$ represents the covariant components of the normal to the element boundary. Notice that Eq. (33) looks like a standard discretization of a parabolic reaction–diffusion equation, except for the term involving the integrals in the contours of the elements, which is due to the nondifferentiability of the approximate manifold (see Cantwell et al. [40]).

D. Adapting the Boundary-Layer Mesh

The geometry of the boundary-layer mesh is defined analytically as a function of δ_h, with the help of the following three geometric entities: the surface mesh Γ_h, the extrusion direction $\hat{\boldsymbol{n}}$ representing a continuous vector field on Γ_h, and the stack distribution $\{h_i\}$ that sets the relative thickness of the different layers in the boundary-layer mesh. The direction in which the extrusion will happen $\hat{\boldsymbol{n}}$ is decoupled from the thickness, which is measured by δ_h. That is, $\hat{\boldsymbol{n}}$ is a continuous piecewise linear vector field on the surface triangulation: $\hat{\boldsymbol{n}} \in (\mathcal{U}_h)^{n+1}$ precomputed by averaging the normals of the neighboring elements and setting $\|\hat{\boldsymbol{n}}\| = 1$ at the vertices.

The stack distribution $\{h_i\}$ is used to assign a fraction of the whole extrusion to each layer of the boundary-layer mesh, which controls the growth rate of the elements away from the wall. Here, h_i denotes the relative thickness of element $i = 1, 2, \ldots, n_{\mathrm{norm}}$ in the stack, with $i = 1$ being the closest to the wall and n_{norm} being the total number of elements. We construct the stack using a geometric sequence of constant ratio α_h, normalized so that

$$\sum_i h_i = 1$$

Here, we set $\alpha_h = 1.4$ for laminar flows and $\alpha_h = 1.6$ for turbulent flows, as recently proposed by Drosson et al. [36].

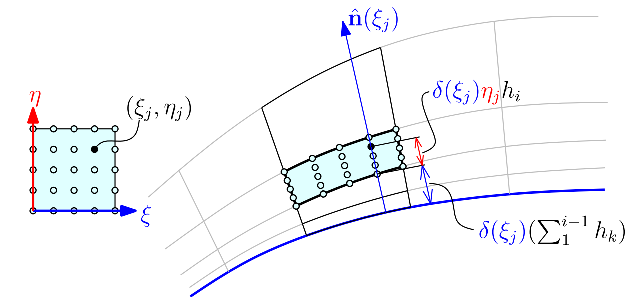

$\hat{\mathbf{n}}(\xi_j)$
η
(ξ_j, η_j)
$\delta(\xi_j)\eta_j h_i$
ξ
$\delta(\xi_j)(\sum_1^{i-1} h_k)$

Fig. 2 Reconstruction of the high-order geometry of an element in the i-th layer of the stack.

Next, we define the geometry of a given element i of the stack using the formula

$$\boldsymbol{x}^{\mathrm{bl}}(\xi, \eta) = \boldsymbol{x}^{\mathrm{bl}}(\xi, 0)|_{K^\Gamma} + \left(\sum_{j=1}^{i-1} h_j + \eta h_i\right)(\delta_h(\xi)|_{K^\Gamma})\hat{\boldsymbol{n}}(\xi)|_{K^\Gamma} \tag{34}$$

where (ξ, η) represents the coordinate in the reference element $K^{\mathrm{ref}} \equiv [-1, 1] \times [-1, 1]$. The extrusion from the wall is given by the second term on the right-hand side, which is the product of the normal scaling and the extrusion direction (both of which only depend on the coordinates at the surface) and the normalized distance away from the wall, which is a function of the element count i across the stack and the normal coordinate η. This procedure is sketched in Fig. 2.

The total extrusion, given by the term $(\delta_h(\xi)|_{K^\Gamma})\hat{\boldsymbol{n}}(\xi)|_{K^\Gamma}$, is the product of a first-order scalar field and a first-order vector field, hence, a polynomial of second degree in ξ. This implies that the collocation of the geometry is exact, provided $\boldsymbol{x}^{\mathrm{bl}}(\xi, 0)|_{K^\Gamma}$ is a polynomial of degree 2 or more. This requirement is automatically satisfied for all the cases considered in this paper.

Once the boundary-layer mesh has been constructed, we can approximate the mass and momentum defect using the formulas

$$\delta_k^* \approx \delta_h \times (\boldsymbol{n} \cdot \hat{\boldsymbol{n}}) \times \sum_{i=1}^{n_{\mathrm{norm}}} h_i \int_{\eta=0}^{\eta=1} \left(1 - \frac{u}{u_e}\right) \mathrm{d}\eta \tag{35}$$

$$\theta_k \approx \delta_h \times (\boldsymbol{n} \cdot \hat{\boldsymbol{n}}) \times \sum_{i=1}^{n_{\mathrm{norm}}} h_i \int_{\eta=0}^{\eta=1} \left(1 - \frac{u}{u_e}\right) \frac{u}{u_e} \mathrm{d}\eta \tag{36}$$

Here, $\boldsymbol{n}$ denotes the actual normal at the point in the surface where the integral is approximated and $(\boldsymbol{n} \cdot \hat{\boldsymbol{n}})$ takes into account a possible mismatch with the extrusion direction. In addition, the velocity profile u/u_e is computed by projecting the flow velocity along the tangent to the wall as follows:

$$\frac{u}{u_e} = \frac{\|\boldsymbol{v} - (\boldsymbol{v} \cdot \boldsymbol{n})\boldsymbol{n}\|}{\|\boldsymbol{v}_e - (\boldsymbol{v}_e \cdot \boldsymbol{n})\boldsymbol{n}\|} \tag{37}$$

where we assume that $\boldsymbol{v}_e$ is constant across the boundary layer and equal to the velocity at the edge of the boundary-layer domain.

E. Adapting the External Mesh

The external mesh has to conform to the evolution of the boundary-layer geometry during the solution process. To achieve this, we use the mesh deformation algorithm by Roca et al. [21] and Gargallo-Peiro et al. [22]. This algorithm is based on an optimization procedure that minimizes the distortion of the mesh and penalizes inverted elements, hence reducing the risk of producing invalid meshes.

Given a high-order mesh, we define the deformation mapping as the function $\boldsymbol{\phi}: \boldsymbol{x}_0 \mapsto \boldsymbol{x}$ that maps a reference/initial configuration to a current deformed configuration. The algorithm proceeds by defining a pointwise metric $\zeta = \zeta(\boldsymbol{D\phi})$ based on the gradient $\boldsymbol{D\phi}$, which is computed implicitly with the help of the isoparametric representation of the reference and current configuration as follows:

$$\zeta = \frac{\|\boldsymbol{D}\phi\|_{\mathrm{Fr}}^2}{2\sigma^*} \tag{38}$$

$$\sigma^* = \frac{1}{2}[\det(\boldsymbol{D}\phi) + \sqrt{\det(\boldsymbol{D}\phi)^2 + 10^{-2}\det(\boldsymbol{D}\varphi_0)^2}] \tag{39}$$

The metric is designed to be insensitive to rotation, translation, or isotropic scaling of the elements, but deviates from $\zeta = 1$ as the element is deformed. The particular details of this can be found in [22].

The distortion field can be collapsed into an elementwise indicator to measure the deformation of the mesh. This is given by

$$\zeta_e = \frac{\int_{e_0} (\zeta - 1)^2 \, \mathrm{d}V_0}{\int_{e_0} \mathrm{d}V_0} \tag{40}$$

and depends exclusively on the location of the high-order nodes of the element at the current configuration $\boldsymbol{x}^{\mathrm{ext}}_{\{\mathcal{J}(e)\}}$.

We can reduce the elementwise indicator ζ_e to a single scalar for the whole mesh configuration by adding up the contributions over all the elements. In this way, we can translate the problem of deforming the mesh into that of minimizing the overall measure of distortion, given by

$$\min_{\boldsymbol{x}_j^{\mathrm{ext}}} \sum_{e\in\mathcal{T}_h} \zeta_e(\boldsymbol{x}^{\mathrm{ext}}_{\{\mathcal{J}(e)\}}) \tag{41}$$

$$\text{subject to } \boldsymbol{x}^{\mathrm{ext}}_{\{\mathcal{J}_{bou}\}} = \boldsymbol{x}^{\mathrm{bl}}_{\{\mathcal{J}_{bou}\}}(\delta_h) \tag{42}$$

Here, $\{\mathcal{J}_{bou}\}$ is the subset of nodes that lie on the interface between the external mesh and the boundary-layer mesh and $\boldsymbol{x}^{\mathrm{bl}}_{\{\mathcal{J}_{bou}\}}(\delta_h)$ is given by Eq. (34) evaluated at the nodes in $\{\mathcal{J}_{bou}\}$. We note that $\boldsymbol{x}^{\mathrm{bl}}_{\{\mathcal{J}_{bou}\}}$ depends explicitly on δ_h, which is the driver for the whole mesh deformation.

At optimality, the solution needs to satisfy the following conditions:

$$\frac{\partial \sum_{e\in\mathcal{T}_h} \zeta_e}{\partial \boldsymbol{x}_j^{\mathrm{ext}}} = 0, \quad \forall\, j \in \{\mathcal{J}\}\setminus\{\mathcal{J}_{bou}\} \tag{43}$$

$$\boldsymbol{x}_j^{\mathrm{ext}} = \boldsymbol{x}_j^{\mathrm{bl}}(\delta_h), \quad \forall\, j \in \{\mathcal{J}_{bou}\} \tag{44}$$

The solution of the optimization problem can be undertaken using a variety of techniques. In particular, Gargallo-Peiro et al. propose to use a local Gauss–Seidel method [22]. In our case, we work directly with the optimality conditions and treat them as extra nonlinear equations that are solved together with the flow equations. This approach reduces the risk inherent to staggered iterations and fits well within the Newton–Raphson scheme that we use to solve the nonlinear system.

F. Coupling the Flow Equations to the Mesh Equations

The adaptive solver is composed of three sets of discrete nonlinear equations that describe the flow solution as well as the associated mesh: 1) the equations for δ_h that govern the evolution of the boundary-layer thickness [Eq. (33)], 2) the equations for $\boldsymbol{x}^{\mathrm{ext}}$ that govern the mesh deformation for the external domain [Eqs. (43) and (44)], and 3) the discretization of the fluid model in ALE form using the HDG method (Sec. II.D). The coupling between these three sets of equations is summarized in Fig. 3.

We are interested in steady-state solutions to this system that yield the flowfield and the supporting adapted mesh at convergence. To achieve this, we consider the time-dependent problem, which is discretized using a backward Euler formula. The result is an algebraic system of nonlinear equations at each time step that we solve using a Newton–Raphson iteration. This requires the assembly of the residual and the Jacobian at every iteration, which we compute using Gaussian quadrature of order $4p$ (where p is the polynomial order of approximation) and store using a sparse format. For all the results presented in this paper, we use uniformly spaced nodes inside each element and a high-order interpolation basis on them. The Newton–Raphson update is combined with a backtracking line-search algorithm and a check to verify that δ_h remains positive at every iteration. We end the nonlinear solver when the norm of the update vector or the residual is converged to machine precision.

This procedure is repeated until the solution reaches steady state. We accelerate this process by doubling the time step Δt when the Newton–Raphson scheme converges as fast (in terms of numbers of iterations) as the previous time step, and halve it otherwise. In some instances, the nonlinear solver may fail to converge or throw a numerical exception. If that is the case, we do not update the solution and divide Δt by 10 before attempting again. This process is carried out until Δt is greater than 20 convective times, at which point a steady-state nonlinear solve is performed.

We start the solver from uniform freestream conditions for the flow and a constant value of the normal scaling:

$$\delta_h/L \approx 5\frac{1}{\sqrt{Re_L}}\sqrt{\frac{\Delta t u_\infty}{L}}$$

which depends on Δt for the first time step. We set the latter to $\Delta t u_\infty/L = 10^{-3}$ for laminar flows and $\Delta t u_\infty/L = 10^{-5}$ for

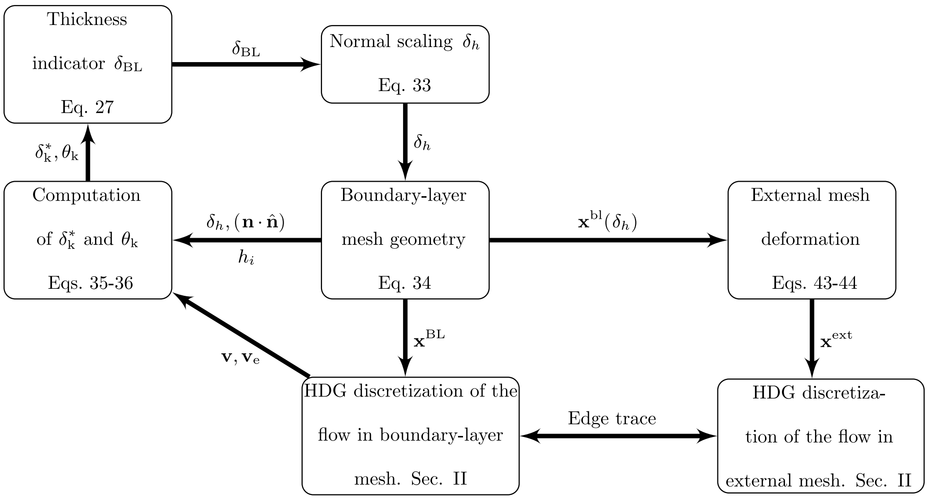

Thickness indicator δ_{BL} Eq. 27
δ_{BL}
Normal scaling δ_h Eq. 33
δ_h
δ_k^*, θ_k
Computation of δ_k^* and θ_k Eqs. 35-36
$\delta_h, (\mathbf{n}\cdot\hat{\mathbf{n}})$
h_i
Boundary-layer mesh geometry Eq. 34
$\mathbf{x}^{\mathrm{bl}}(\delta_h)$
External mesh deformation Eqs. 43-44
$\mathbf{x}^{\mathrm{BL}}$
$\mathbf{x}^{\mathrm{ext}}$
$\mathbf{v}, \mathbf{v}_e$
HDG discretization of the flow in boundary-layer mesh. Sec. II
Edge trace
HDG discretization of the flow in external mesh. Sec. II

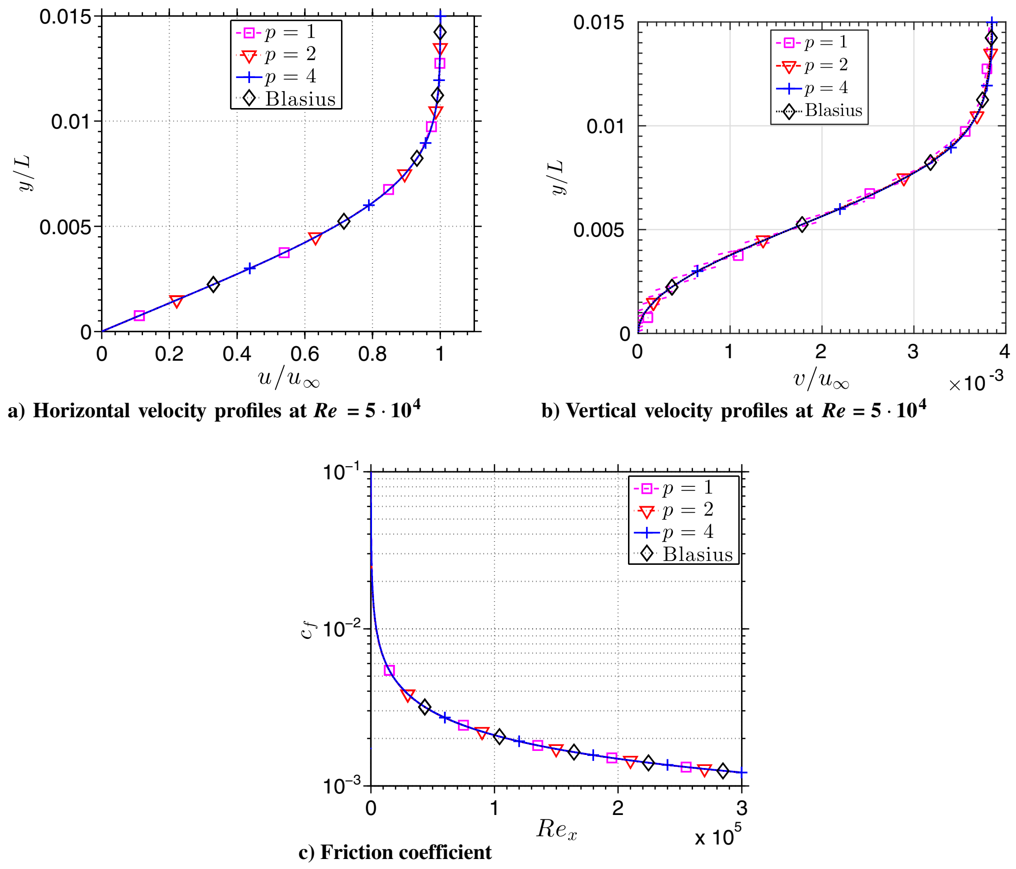
Fig. 3 Flow chart of the adaptive solver showing the interdependence of different modules (boxes) as well as the variables (arrows).

turbulent flows. This particular choice of initialization makes δ_k^* and θ_k zero at the first iteration, which causes an indefinite value of δ_{BL}. To avoid this numerical issue, we force $\partial\delta_h/\partial t = 0$ during the first time step.

The use of a monolithic approach for the solution of the flowfield and the mesh has advantages and disadvantages. On the one hand, the fact that the mesh grows as the boundary layer is forming alleviates some of the problems associated with underresolution and reduces the number of nonlinear iterations required at each time step compared with running the solver on a fixed mesh. This is especially true for turbulent cases where the nonlinearities in the model ought to be treated with enough mesh resolution. On the other hand, the addition of mesh unknowns makes the problem slightly more expensive in terms of global number of degrees of freedom (equivalent to two extra unknowns per node on the mesh in 2-D) and fill of the Jacobian. The latter is closer to a continuous Galerkin connectivity pattern due to the relationships between the nodes of the mesh. The limited number of unknowns associated with the boundary-layer indicator are negligible compared with the rest and do not affect the connectivity pattern if treated appropriately.

IV. Results

In this section, we apply our boundary-layer adaptivity technique to a variety of laminar and turbulent flows. In all test cases, we compare our results to existing results reported in the literature.

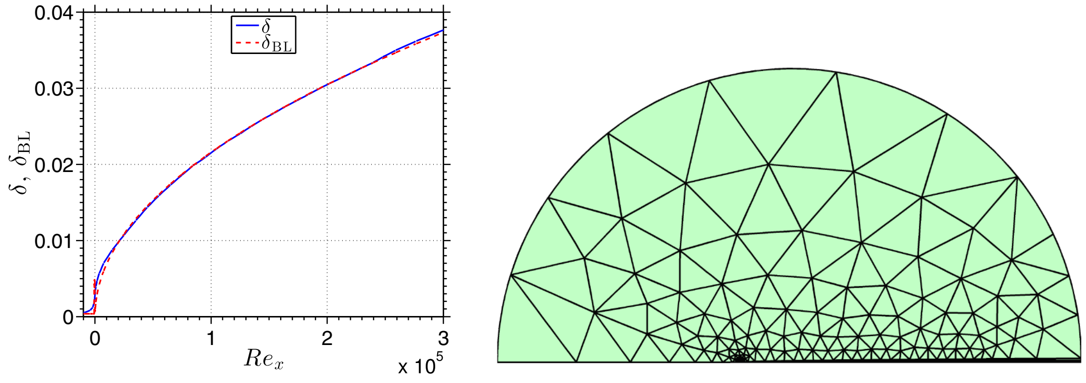


a) Horizontal velocity profiles at $Re = 5 \cdot 10^4$

b) Vertical velocity profiles at $Re = 5 \cdot 10^4$

c) Friction coefficient

Fig. 4 Comparison of solutions obtained on a sequence of meshes with equivalent resolution against the Blasius analytical solution. By properly adapting the boundary layer, results become insensitive to the polynomial order.



a) Normal scaling and thickness indicator

b) Mesh

Fig. 5 Mesh and normal scaling at convergence for the case of the laminar flat plate. Notice how the normal scaling follows the thickness indicator and can avoid the leading-edge singularity thanks to the diffusive terms in the governing PDE.

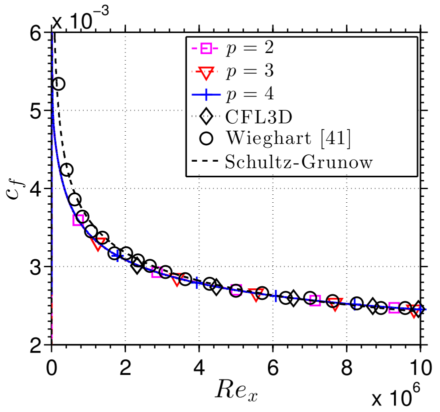


Fig. 6 Friction coefficient along the flat plate compared with experimental data and empirical correlations, as well as grid converged results for the CFL3D solver.

A. Laminar Flow over a Flat Plate

For this problem, we can compare our results to an analytical solution, hence its usefulness as a validation and verification case. In particular, we are concerned with the laminar flow over a flat plate at zero angle of attack, $M_\infty = 0.1$, and $Re_L = 10^5$. The geometry of the computational domain consists of a semicircle of radius $r = 3$ with the flat plate embedded on the lower boundary.

We are interested in comparing the accuracy of the solution for different approximation orders. For this comparison to be fair, we need to ensure that the resolution (i.e., number of degrees of freedom) is roughly the same independently of the order approximation. We can satisfy this condition by generating a sequence of meshes in which, at each step, the current mesh is uniformly refined and the polynomial order is divided by two. We note that this refinement is merely topological, because the r adaptivity is rerun at each time to adapt to the boundary layer. The coarsest case is computed using polynomials of order $p = 4$ with a total of 10 elements across the boundary layer.

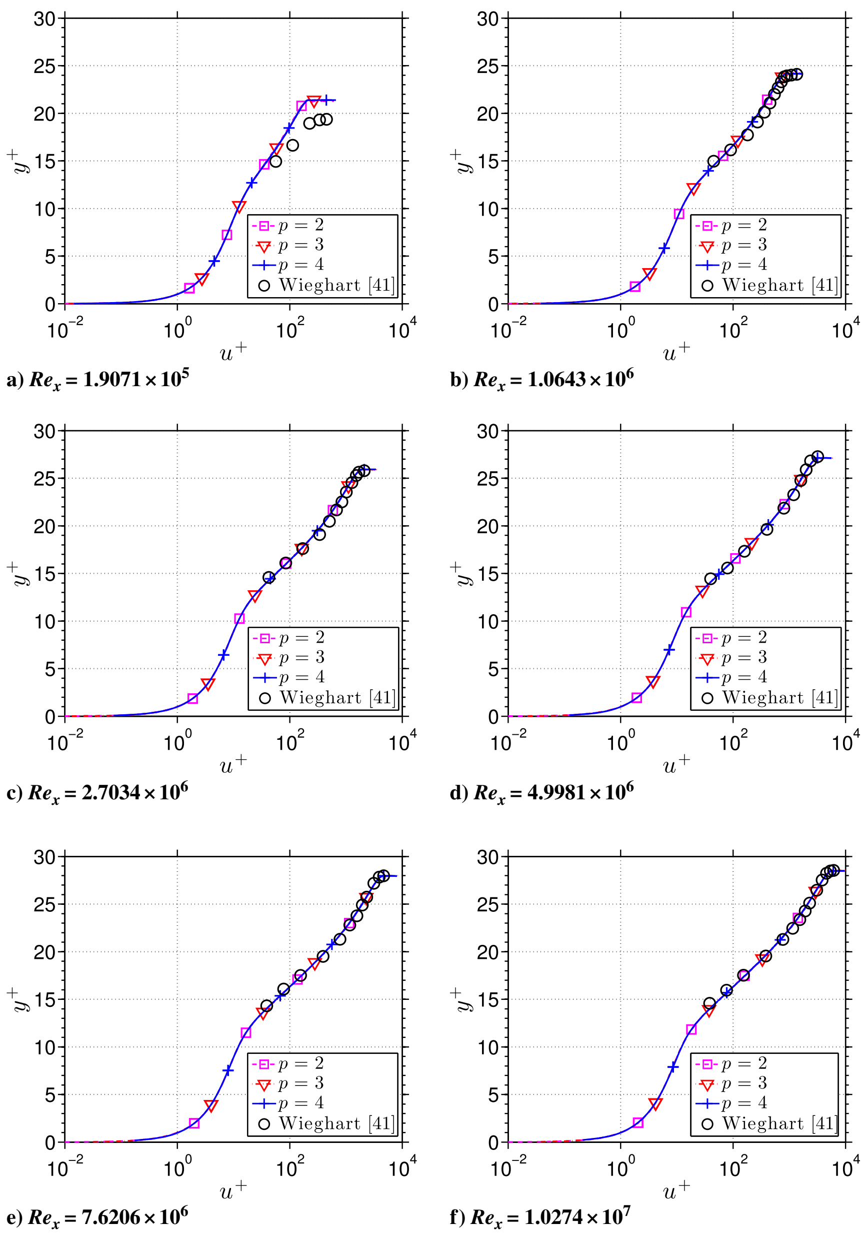


Fig. 7 Horizontal velocity profiles measured in wall units (y^+ vs u^+) at different stations along the flat plate. Agreement with experiments is excellent except for the case closest to the leading edge.

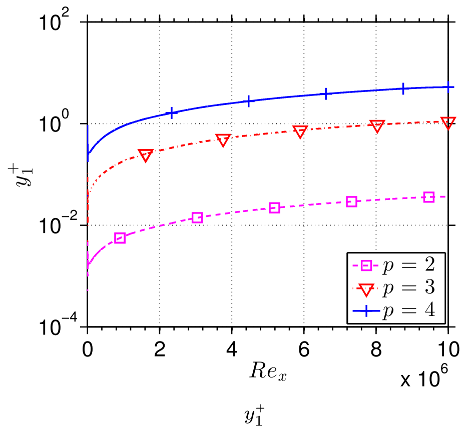


Fig. 8 Evolution of y_1^+ along the flat plate. Results indicate that the stretched grid in the normal direction and the adaptivity in the boundary-layer thickness are enough to control the growth of y_1^+.

We summarized the results obtained on this sequence of meshes in Fig. 4, which include the Blasius solution. The numerical results for the flowfield (i.e., velocity profiles and c_f) lie on top of the reference solution for any approximation order, with the exception of the normal velocity for $p = 1$, which presents jumps across element boundaries.

As a byproduct, this procedure returns a mesh that is adapted to the boundary layer along the wall. This is plotted in Fig. 5 for the case of $p = 4$. We observe how the normal scaling δ approximately follows a square root law, as is expected from the Blasius theory.

B. Turbulent Flow over a Flat Plate

Next, we test our method in the turbulent regime by applying it to the same geometry, at $M_\infty = 0.2$ and $Re_L = 5 \cdot 10^6$. In this case, the goal is to compare the solutions obtained with different orders of approximation to experimental results. We use a sequence of meshes given by $(p, n_{\text{norm}}) = \{(2, 20), (3, 13), (4, 10)\}$, which ensures approximately 40 degrees of freedom across the boundary layer. This is twice as much resolution as in the laminar case and responds to the need to capture the complexity of the turbulent velocity profiles, especially close to the wall.

The skin friction coefficient computed in these runs is plotted in Fig. 6 together with experimental results [41] and the Schultz-Grunow correlation. For comparison purposes, we also include the grid-converged results obtained with the CFL3D solver, which are available for download at NASA's Turbulence Modeling Resource database [42]. The agreement among the numerical solutions is excellent and very close to the experimental data, except around the leading edge. This seems to be an effect of the turbulence model rather than the numerics as other results on the Turbulence Modeling Resource database indicate.

A similar plot is drawn in Fig. 7 for the velocity profiles at different stations along the flat plate, compared with the experimental results by Wieghart [41]. These plots show a very strong agreement between the different runs and also with the experimental data, except for the first station at $Re_x = 1.9071 \times 10^5$ (Fig. 7a). The discrepancies found there are also present in other validation studies (e.g., the NPARC alliance database [43]) and can be partly attributed to the errors in c_f around the aforementioned leading edge.

One of the advantages of adapting to the boundary-layer thickness is the possibility to control the location of the first degree of freedom off the wall. This quantity of interest, denoted by y_1^+ in wall units, dictates the resolution in the viscous sublayer of the turbulent boundary layer and hence the accuracy of the computed friction coefficient. In the proposed r-apativity scheme, the value of y_1^+ is partially controlled by δ, hence it is interesting to verify that this is effectively the case. Notice that, in general, these two quantities do not necessarily grow at the same rate, which can be problematic for higher Reynolds numbers in the presence of pressure gradients. The plots in Fig. 8 show that adapting to the boundary-layer thickness is enough to keep y_1^+ under control, at least for the case of a zero pressure gradient flow. Furthermore, the fact that all results virtually coincide seems to indicate that a high-order discretization is not as sensitive to the value of y_1^+, as previously suggested by other studies [36]. Nevertheless, ensuring $y_1^+ \approx 1$ is considered a best practice, thus further development is required if this methodology is to be adopted in other solvers.

C. Turbulent Flow Around a NACA 0012 Airfoil

In this section, we show results obtained with the adaptive solver for the case of the turbulent flow around a NACA 0012 airfoil. The parameters for this case are $M_\infty = 0.15$ and $Re_c = 6 \cdot 10^6$. This configuration was chosen due to the availability of experimental and numerical data [42] for a variety of angles of attack in the range $\alpha \in [0, 15]$ deg.

In the following, we compare runs at different angles of attack using the same polynomial order $p = 4$ and the same topological mesh composed of $n_{\text{surf}} \times n_{\text{norm}} = 49 \times 10$ elements in the boundary-layer domain. Figure 9 contains the flowfield and mesh returned by the solver for the case of $\alpha = 15$ deg. We proceed to validate the solver by comparing the pressure and friction coefficients at three angles of attack ($\alpha = \{0, 10, 15\}$ deg) to experimental data (available only for c_p), as well as grid-converged results computed with CFL3D and Xfoil [44]. These are summarized in Fig. 10 and show an excellent agreement between the numerical solutions and the experiments.

A similar behavior is found for the integrated forces, such as lift or drag, as depicted in Fig. 11. Here, the r-apativity solver is run at intervals of 1 deg ($\alpha = 0, 1, 2, \ldots,$ deg) and an compared against experimental data [42], the results from CFL3D [42] for $\alpha = \{0, 10, 15\}$ deg, and a polar computed with Xfoil.

According to Rumsey [42], the CFL3D simulations were run using a structured C mesh of 897×257 nodes ($\approx$230,000 vertices). In comparison, the mesh used in the r-apativity solver was composed of

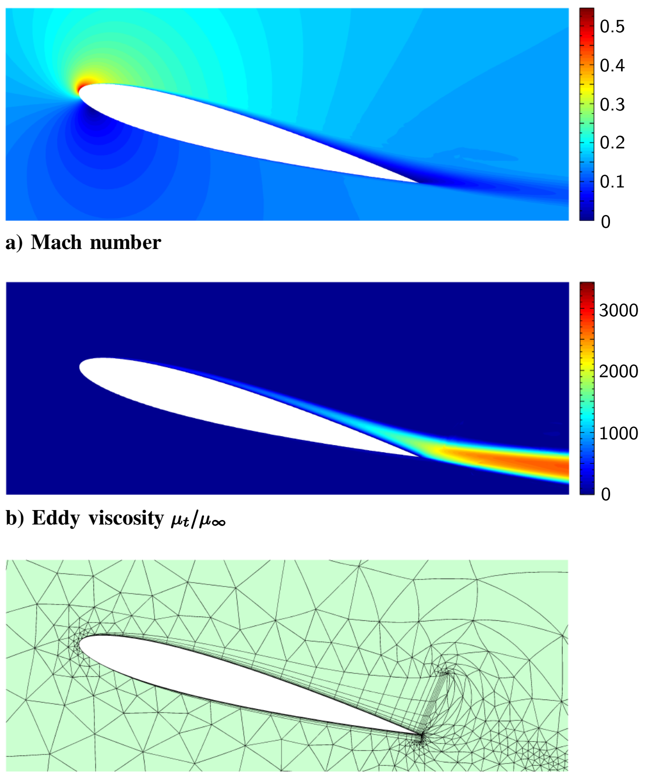
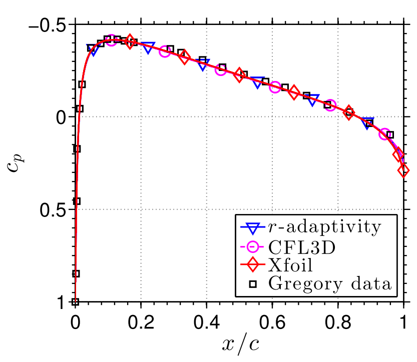
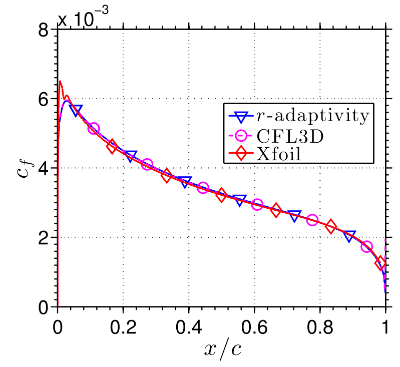


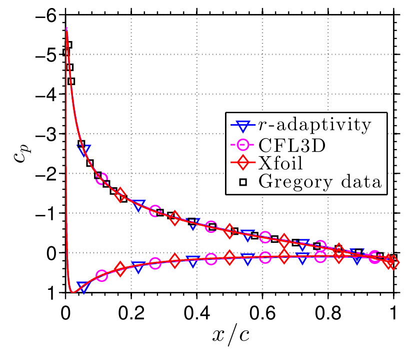
Fig. 9 Flowfield and mesh returned by the adaptive solver for the case of a NACA 0012 at $M_\infty = 0.15$, $Re_c = 6 \cdot 10^6$, and $\alpha = 15$ deg. The mesh adapts successfully to the thickness of the boundary layer.



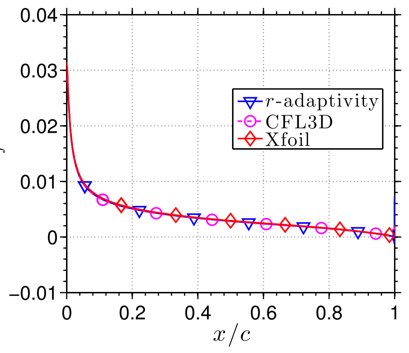
a) c_p at $\alpha = 0$ deg



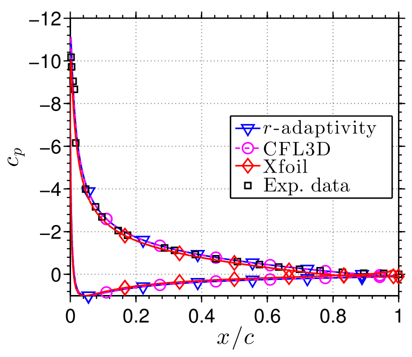
b) c_f at $\alpha = 0$ deg



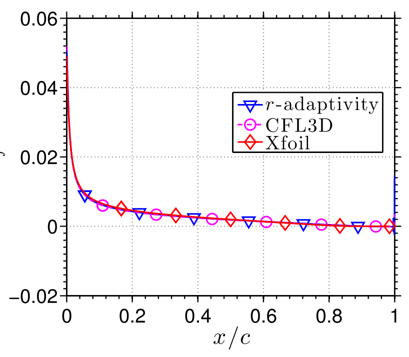
c) c_p at $\alpha = 10$ deg



d) c_f at $\alpha = 10$ deg

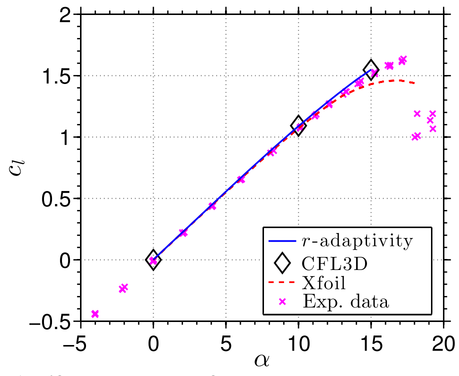


e) c_p at $\alpha = 15$ deg

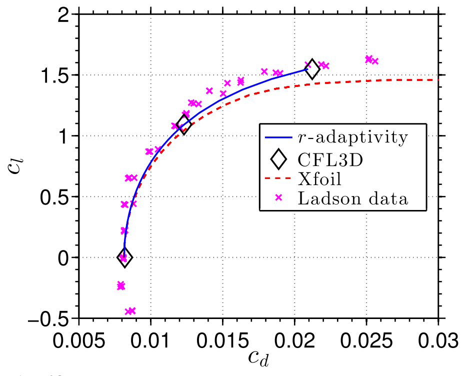


f) c_f at $\alpha = 15$ deg

Fig. 10 Comparison of pressure and friction coefficients computed using the *r*-apativity solver vs experimental data [42], CFL3D, and Xfoil. Agreement with experiments is excellent for all angles of attack.



a) Lift versus angle of attack

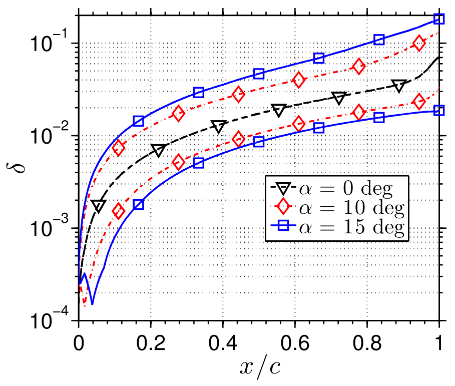


b) Lift versus drag

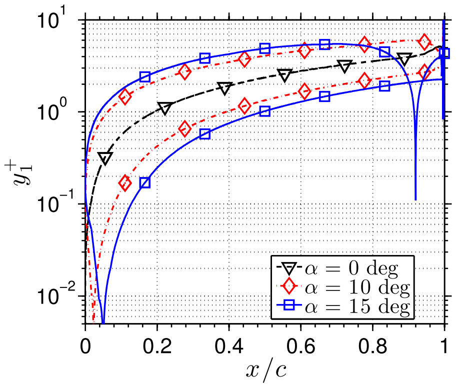
Fig. 11 Plots of lift and drag over a NACA 0012 at $M_\infty = 0.15$ and $Re_c = 6 \cdot 10^6$ for a range of angles of attack, compared against experimental data [42], CFL3D, and Xfoil.

38,500 high-order nodes. This represents significant savings in computational cost and serves to justify the use of a high-order approximation combined with *r* adaptivity.

The effect of *r* adaptivity is clearly visible on the meshes obtained as part of the solution (see Fig. 9c), but can be quantified by comparing δ and y_1^+ side-by-side as plotted in Fig. 12. First, we

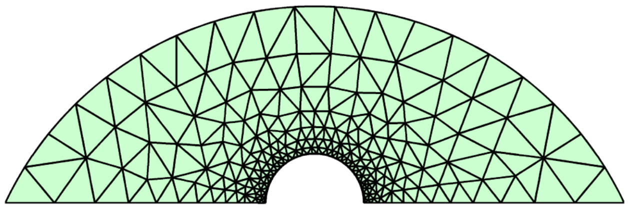


a) Normal scaling δ

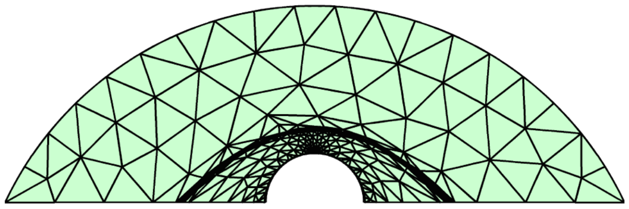


b) Distance of first node off the wall

Fig. 12 Chordwise evolution of the normal scaling δ and distance to the first degree of freedom off the wall (y_1^+) for meshes returned by the adaptive solver around a NACA 0012 airfoil at $M_\infty = 0.15$, $Re_c = 6 \cdot 10^6$, and three different angles of attack.

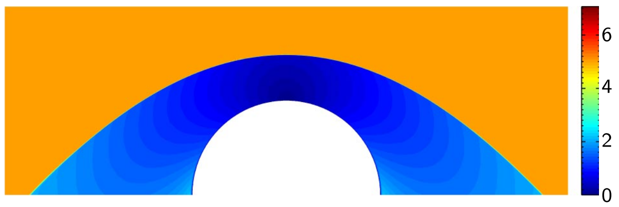


a) Initial mesh

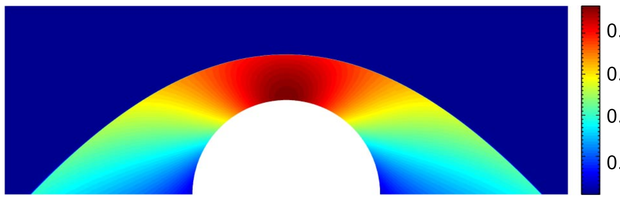


b) After 5 iterations of anisotropic refinement on the external mesh

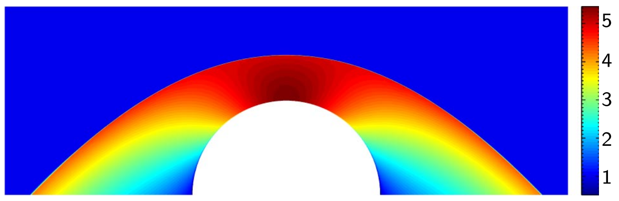
Fig. 13 Meshes used to compute high-speed flow around a cylinder. The combination of anisotropic refinement and r adaptivity yields meshes that are adapted to the boundary layer and shock wave using different mechanisms.



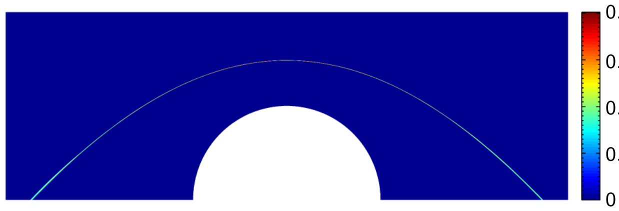
a) Mach number



b) Pressure



c) Density



d) Shock indicator

Fig. 14 Sample of the flowfield around a cylinder at $M_\infty = 5$ and $Re_R = 4 \cdot 10^4$. Results shown here correspond to five iterations of anisotropic refinement.

observe that the r-apativity scheme produces meshes, in which the thickness of the boundary-layer domain grows up to three orders of magnitude from the stagnation point (where it is minimal) to the trailing edge. This adaptivity happens automatically as part of the solution process and is oblivious to the existence of a stagnation point in the flow. Second, we note that this aggressive growth in the normal scaling δ is enough to maintain $y_1^+ = \mathcal{O}(1)$ along the surface.

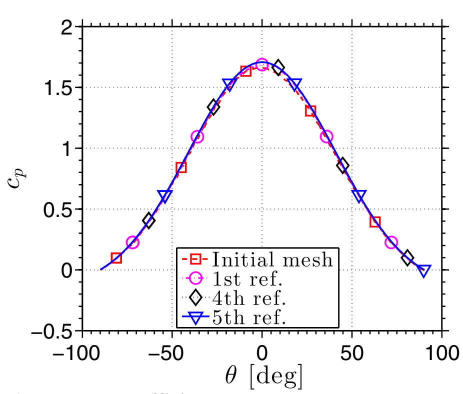
All in all, these results for the airfoil showcase the generality and efficiency of this scheme and serve as validation for low Mach number flows. Furthermore, the mesh generation is totally independent of the parameters of the problem such as Reynolds number, Mach number, or angle of attack, which allow us to reuse the mesh topology between runs.

D. Viscous High-Speed Flow over a Cylinder

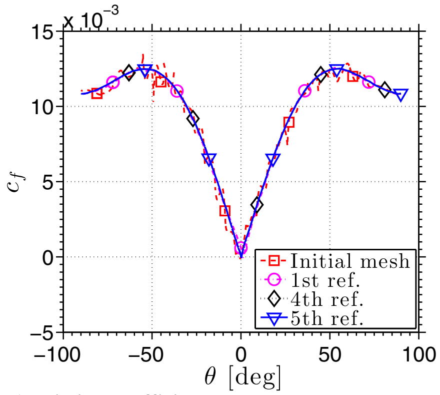
The previous test cases have served to showcase the benefits of r adaptivity for low Mach number flows, both in the laminar and turbulent regimes. We now extend our study to laminar compressible flows with the help of the shock-capturing model introduced in Sec. II.C.

We consider the simulation of the flow around the bow of a cylinder in a supersonic stream at $M_\infty = 5$ with a Reynolds number of $Re_R = 4 \cdot 10^4$. In this configuration, the flow exhibits a detached shock wave and a thin boundary layer. Although the r-apativity scheme can take care of the boundary layer, it is not designed to adapt around the shock. For this, we propose to use the bidimensional anisotropic mesh generator (BAMG) a posteriori anisotropic adaptivity framework [45] in an outer loop, with the r-apativity solver inside it. The mesh returned by the process is adapted to both the boundary layer and the shock wave, as shown in Fig. 13.

A sample of the flow solution obtained after five iterations of anisotropic refinement is contained in Fig. 14. For all these runs, $p = 3$ and $n_{\text{norm}} = 7$. As we expected, the use of adaptivity on the external domain yields sharper shocks and also removes the oscillations in the flow behind the shock, which are a direct cause for inaccuracies in the boundary layer. This is clearly visible in Fig. 15 where the pressure coefficient is compared with the friction

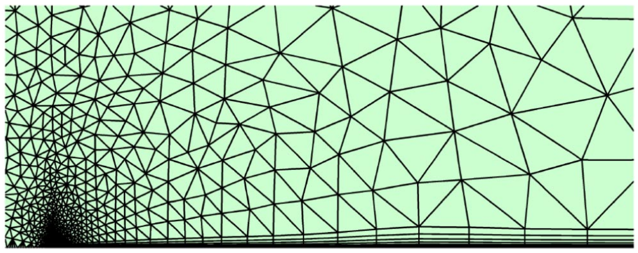


a) Pressure coefficient

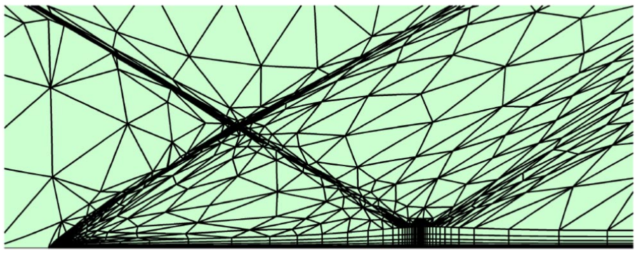


b) Friction coefficient

Fig. 15 Evolution of the pressure and friction coefficient around the cylinder with the adaptation cycle. Results highlight the importance of shock capturing and adaptivity in obtaining clean friction coefficients (1st ref., 1st mesh refinement).

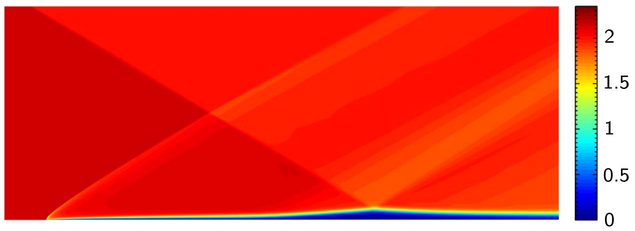


a) Initial mesh

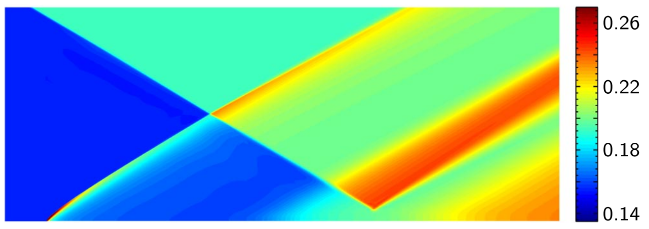


b) Final mesh after 9 iterations of anisotropic refinemen on the external mesh

Fig. 16 Evolution of the mesh around the separation bubble in the shock wave/boundary-layer interaction case.



a) Mach number



b) Pressure

Fig. 17 Mach number and pressure field around the flat plate. Separation and reattachment of the boundary layer due to the shock is visible as compression waves in the pressure field.

coefficient as the refinement evolves. We note that the viscous quantities (e.g., c_f) benefit from the refinement more than the inviscid quantities (e.g., c_p), which can be accurately computed on a coarser discretization.

Therefore, we conclude that the success of the r-apativity methodology in the case of high Mach number flows is constrained by the shock-capturing capabilities of the scheme, in addition to the proper adaptation mechanisms around the shock waves.

E. Shock Wave/Boundary-Layer Interaction

The last test case deals with a strong shock wave/boundary-layer interaction that happens when a shock hits the boundary layer and causes separation. When this happens, the shock does not reflect off the wall as it would in the inviscid case (or if the boundary layer does not separate), but rather turns into an expansion fan at the same time that other flow features develop. Our goal is to assess the behavior of the r adaptivity approach when the boundary layer separates.

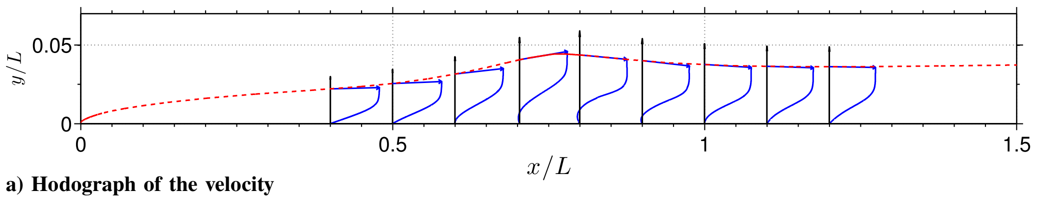
For this, we will use the test case by Degrez et al. [46], whose geometry is composed of a flat plate and a shock generator inside a stream at $M_\infty = 2.15$ and $Re_L = 10^5$. In our computations, we choose to round the leading edge of the flat plate to the dimensions measured by Degrez et al. [46] to avoid the singularity. The simulations use polynomials of order $p = 3$ and $n_{\text{norm}} = 7$ elements across the boundary layer.

The solver is run iteratively with BAMG in the outer loop and r adaptivity between iterations. This yields meshes like the ones shown in Fig. 16. A rendering of the flowfield obtained at the highest level of refinement is included in Fig. 17, along with plots of the boundary-layer profiles at different stations over the flat plate in Fig. 18. We notice how these show the presence of a laminar separation bubble, which is well handled by the r adaptivity.

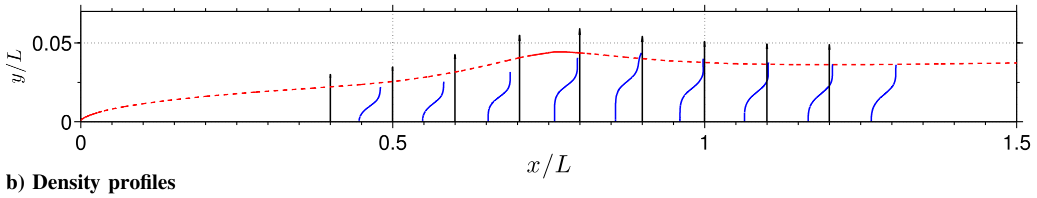
The importance of adaptivity can be assessed by looking at the evolution of the stresses at the wall with each iteration of BAMG, as plotted in Fig. 19. In principle, these seem insensitive to the refinement of the external mesh except at the region immediately adjacent to the point where the shock hits the wall (see right column of Fig. 19). This behavior is expected of a shock-induced separation bubble.

In qualitative terms, the computed results compare well with the experimental curve by Degrez et al. [46] for the pressure coefficient, but strongly disagree with the numerical simulations presented in the same paper. This was blamed on experimental errors, however, we believe they might also be attributed to the numerics of the scheme used to simulate the experiment.

All in all, this case has served to verify that r adaptivity also works in the case of moderately separated flows, this being a flow regime present in many problems of interest in aerodynamics. In addition, this serves to prove that the method can deal with shock/boundary-layer interactions in a robust way.

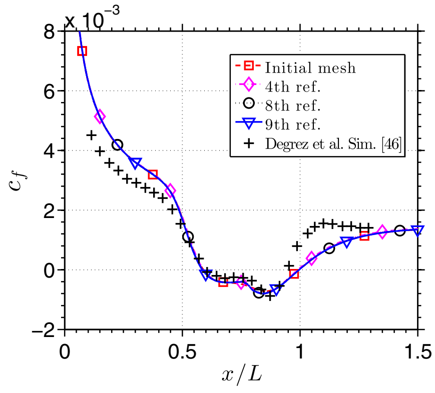


a) Hodograph of the velocity

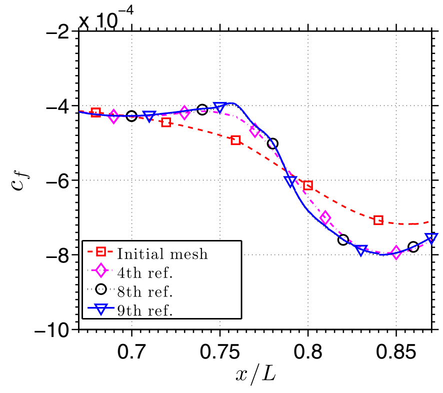


b) Density profiles

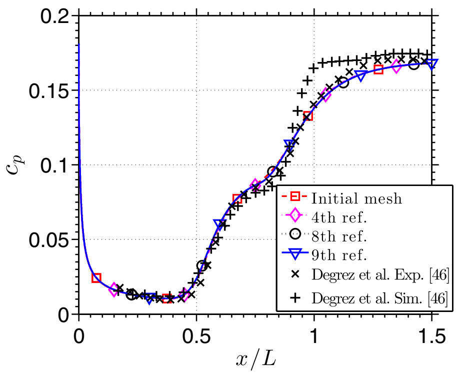
Fig. 18 Velocity and density profiles around the laminar separation bubble extracted from the boundary-layer domain. The dashed line denotes the edge of the boundary-layer domain.



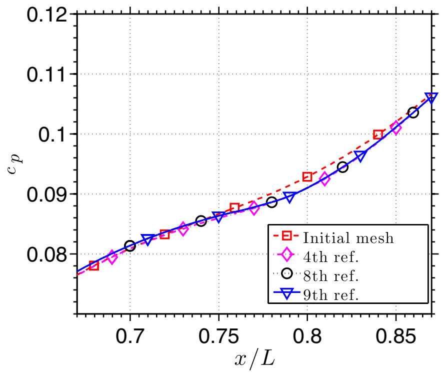
a) Friction coefficient



b) Detail of the friction coefficient



c) Pressure coefficient



d) Detail of the pressure coefficient

Fig. 19 Evolution of stresses at the wall with the external mesh adaptation cycle. The effect of sharper shock profiles can only be seen in the vicinity of the region where the shock impinges (right column).

V. Conclusions

A novel methodology has been presented to simulate viscous compressible flows based on a combination of adaptivity and a high-order discretization. The unique feature of this *r*-apativity method is the way in which the position of the nodes of the mesh is solved together with the flowfield. In this way, the mesh can be adapted to follow the boundary layer, which results in improved solution quality and robustness. The only requirement to apply this methodology is the availability of a structured mesh close to solid walls, which is a common practice in several commercial and research solvers. This allows one to extract boundary-layer profiles and determine the thickness of the viscous layer that drives the mesh deformation algorithm. This novel way of treating the mesh as part of the solution enables the accurate simulation of separation bubbles as well as high Reynolds number flows on highly anisotropic meshes.

Results have been produced for a variety of 2-D flows to evaluate the performance of the scheme in terms of accuracy per degree of freedom or grid control close to the wall. The results indicate that the use of *r* adaptivity can produce grid convergence independently of the order of approximation, with a moderate amount of degrees of freedom across the boundary layer. Also, the results compare favorably to general anisotropic strategies in the case of high Mach number flows, and without loss of generality, can be combined with them to produce meshes that are adapted to the boundary layer, as well as other features like shock waves.

It is believed that the method could be extended in a variety of ways, for example, by including more than one boundary-layer mesh in the domain (e.g., multi-element airfoils). The treatment of thin gaps in that case would require the extension of the mesh deformation algorithm to include the boundary-layer domain and the addition of a penalty term to drive the edge of the boundary layer, while preventing element inversion. In addition, the equations that govern the boundary-layer thickness should be extended to include explicit control over the viscous scale y_1^+. This would effectively treat the two relevant scales of the problem independently and prevent underresolution in more general cases. Similarly, an additional indicator could be introduced to approximate the thickness of the thermal boundary layer, which can be misgauged by the current definition of δ_{BL} in cases where the heat transfer dominates ($Pr < 1$). The extension of this methodology to three-dimensional flows would only require a reassessment of the thickness indicator when crossflow is present. Together with the application to three-dimensional flows, the single most important extension would be to include transition to turbulence, either in the form of a classical stability analysis for which the boundary-layer profiles are readily available or a combination with large-eddy simulation, which could yield significant savings in terms of computational cost.

Acknowledgments

The authors would like to acknowledge Xevi Roca for his feedback and discussion of various aspects of the solver, especially the mesh generation and deformation. D. Moro would like to acknowledge the financial support by the Fundacion La Caixa and the Zakhartchenko Fellowship. N. C. Nguyen and J. Peraire would like to acknowledge support by the U.S. Air Force Office of Scientific Research (AFOSR) (Grant FA9550-12-0357). M. Drela would like to acknowledge the support by the AFOSR (Grant F49620-78C-0084).

References

[1] Peraire, J., Vahdati, M., Morgan, K., and Zienkiewicz, O. C., "Adaptive Remeshing for Compressible Flow Computations," *Journal of Computational Physics*, Vol. 72, No. 2, 1987, pp. 449–466. doi:10.1016/0021-9991(87)90093-3

[2] Peraire, J., Peiro, J., and Morgan, K., "Adaptive Remeshing for Three-Dimensional Compressible Flow Computations," *Journal of Computational Physics*, Vol. 103, No. 2, 1992, pp. 269–285. doi:10.1016/0021-9991(92)90401-J

[3] Frey, P., and Alauzet, F., "Anisotropic Mesh Adaptation for CFD Computations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, Nos. 48–49, 2005, pp. 5068–5082. doi:10.1016/j.cma.2004.11.025

[4] Fidkowski, K. J., "A Simplex Cut-Cell Adaptive Method for High-Order Discretizations of the Compressible Navier–Stokes Equations," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2007.

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

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

[7] Bassi, F., Crivellini, A., Rebay, S., and Savini, M., "Discontinuous Galerkin Solution of the Reynolds-Averaged Navier-Stokes and $k - \omega$ Turbulence Model Equations," *Computers and Fluids Journal*, Vol. 34, No. 4, 2005, pp. 507–540. doi:10.1016/j.compfluid.2003.08.004

[8] Nguyen, N. C., Persson, P. O., and Peraire, J., "RANS Solutions Using High Order Discontinuous Galerkin Methods," *45th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA Paper 2007-0914, 2007.

[9] Moro, D., Nguyen, N. C., and Peraire, J., "Navier–Stokes Solution Using Hybridizable Discontinuous Galerkin Methods," *20th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-3407, 2011.

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

[11] Modisette, J. M., "An Automated Reliable Method for Two-Dimensional Reynolds-Averaged Navier–Stokes Simulations," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2011.

[12] Ceze, M., and Fidkowski, K. J., "Pseudo-Transient Continuation, Solution Update Methods, and CFL Strategies for DG Discretizations of the RANS-SA Equations," *21st AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2013-2686, 2013.

[13] Babuška, I., and Guo, B. Q., "The h, p and $h - p$ Version of the Finite Element Method; Basis Theory and Applications," *Advances in Engineering Software*, Vol. 15, Nos. 3–4, 1992, pp. 159–174. doi:10.1016/0965-9978(92)90097-Y

[14] Wang, L., and Mavriplis, D. J., "Adjoint-Based $h - p$ Adaptive Discontinuous Galerkin Methods for the 2D Compressible Euler Equations," *Journal of Computational Physics*, Vol. 228, No. 20, 2009, pp. 7643–7661. doi:10.1016/j.jcp.2009.07.012

[15] Giorgiani, G., Fernández-Méndez, S., and Huerta, A., "Hybridizable Discontinuous Galerkin p-Adaptivity for Wave Propagation Problems," *International Journal for Numerical Methods in Fluids*, Vol. 72, No. 12, 2013, pp. 1244–1262. doi:10.1002/fld.v72.12

[16] Budd, C. J., Huang, W., and Russell, R. D., "Adaptivity with Moving Grids," *Acta Numerica*, Vol. 18, 2009, pp. 111–241. doi:10.1017/S0962492906400015

[17] Tang, H., and Tang, T., "Adaptive Mesh Methods for One- and Two-Dimensional Hyperbolic Conservation Laws," *SIAM Journal on Numerical Analysis*, Vol. 41, No. 2, 2003, pp. 487–515. doi:10.1137/S003614290138437X

[18] Di, Y., Li, R., Tang, T., and Zhang, P., "Moving Mesh Finite Element Methods for the Incompressible Navier–Stokes Equations," *SIAM Journal on Scientific Computing*, Vol. 26, No. 3, 2005, pp. 1036–1056. doi:10.1137/030600643

[19] Drela, M., "A New Transformation and Integration Scheme for the Compressible Boundary Layer Equations, and Solution Behavior at Separation," M.Sc. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 1983.

[20] Allmaras, S. R., "A Coupled Euler/Navier–Stokes Algorithm for 2-D Unsteady Transonic Shock/Boundary-Layer Interaction," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 1989.

[21] Roca, X., Gargallo-Peiró, A., and Sarrate, J., "Defining Quality Measures for High-Order Planar Triangles and Curved Mesh Generation," *Proceedings of the 20th International Meshing Roundtable*, edited by W. R. Quadros, Springer, Berlin, 2011, pp. 365–383.

[22] Gargallo-Peiro, A., Roca, X., Peraire, J., and Sarrate, J., "Optimization of a Regularized Distortion Measure to Generate Curved High-Order Unstructured Tetrahedral Meshes," *International Journal for Numerical Methods in Engineering*, Vol. 103, No. 5, 2015, pp. 342–363. doi:10.1002/nme.v103.5

[23] Nguyen, N. C., and Peraire, J., "Hybridizable Discontinuous Galerkin Methods for Partial Differential Equations in Continuum Mechanics," *Journal of Computational Physics*, Vol. 231, No. 18, 2012, pp. 5955–5988. doi:10.1016/j.jcp.2012.02.033

[24] Peraire, J., Nguyen, N. C., and Cockburn, B., "A Hybridizable Discontinuous Galerkin Method for the Compressible Euler and Navier–Stokes Equations," *48th AIAA Aerospace Sciences Meeting*, AIAA Paper 2010-363, 2010.

[25] Chaurasia, H. K., "A Time-Spectral Hybridizable Discontinuous Galerkin Method for Periodic Flow Problems," Ph.D. Thesis, Massachusetts Inst. of Technology, Cambridge, MA, 2014.

[26] Moro, D., Nguyen, N. C., and Peraire, J., "Dilatation-Based Shock Capturing for High-Order Methods," *International Journal for Numerical Methods in Fluids*, Vol. 82, No. 7, 2016, pp. 398–416. doi:10.1002/fld.4223

[27] Persson, P. O., Bonet, J., and Peraire, J., "Discontinuous Galerkin Solution of the Navier-Stokes Equations on Deformable Domains," *Computer Methods in Applied Mechanics and Engineering*, Vol. 198, Nos. 17–20, 2009, pp. 1585–1595. doi:10.1016/j.cma.2009.01.012

[28] Spalart, P. R., and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aérospatiale*, Vol. 1, No. 1, 1994, pp. 5–21.

[29] Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, 1981, pp. 357–372. doi:10.1016/0021-9991(81)90128-5

[30] Kast, S. M., and Fidkowski, K. J., "Output-Based Mesh Adaptation for High Order Navier–Stokes Simulations on Deformable Domains," *Journal of Computational Physics*, Vol. 252, Nov. 2013, pp. 468–494. doi:10.1016/j.jcp.2013.06.007

[31] Cockburn, B., Gopalakrishnan, J., and Lazarov, R., "Unified Hybridization of Discontinuous Galerkin, Mixed and Continuous

Galerkin Methods for Second Order Elliptic Problems," *SIAM Journal on Numerical Analysis*, Vol. 47, No. 2, 2009, pp. 1319–1365. doi:10.1137/070706616

[32] Nguyen, N. C., and Peraire, J., "An Adaptive Shock-Capturing HDG Method for Compressible Flows," *20th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-3060, 2011.

[33] Drela, M., and Giles, M., "Viscous-Inviscid Analysis of Transonic and Low Reynolds Number Airfoils," *AIAA Journal*, Vol. 25, No. 10, 1987, pp. 1347–1355. doi:10.2514/3.9789

[34] Lighthill, M. J., "On Displacement Thickness," *Journal of Fluid Mechanics*, Vol. 4, No. 4, 1958, pp. 383–392. doi:10.1017/S0022112058000525

[35] Uranga, A., Persson, P. O., Drela, M., and Peraire, J., "Implicit Large Eddy Simulation of Transition to Turbulence at Low Reynolds Numbers Using a Discontinuous Galerkin Method," *International Journal for Numerical Methods in Engineering*, Vol. 87, Nos. 1–5, 2011, pp. 232–261. doi:10.1002/nme.v87.1/5

[36] Drosson, M., Hillewaert, K., and Essers, J.-A., "Stability and Boundary Resolution Analysis of the Discontinuous Galerkin Method Applied to the Reynolds-Averaged Navier–Stokes Equations Using the Spalart–Allmaras Model," *SIAM Journal on Scientific Computing*, Vol. 35, No. 3, 2013, pp. B666–B700. doi:10.1137/110834615

[37] Wang, L., Anderson, K., Erwin, T., and Kapadia, S., "High-Order Discontinuous Galerkin Method for Computation of Turbulent Flows," *AIAA Journal*, Vol. 53, No. 5, 2015, pp. 1159–1171. doi:10.2514/1.J053134

[38] Bassi, F., and Rebay, S., "High-Order Accurate Discontinuous Finite Element Solution of the 2D Euler Equations," *Journal of Computational Physics*, Vol. 138, No. 2, 1997, pp. 251–285. doi:10.1006/jcph.1997.5454

[39] Dziuk, G., and Elliott, C. M., "Finite Element Methods for Surface PDEs," *Acta Numerica*, Vol. 22, 2013, pp. 289–396. doi:10.1017/S0962492913000056

[40] Cantwell, C., Yakovlev, S., Kirby, R. M., Peters, N. S., and Sherwin, S. J., "High-Order Spectral/hp Element Discretisation for Reaction-Diffusion Problems on Surfaces: Application to Cardiac Electrophysiology," *Journal of Computational Physics*, Vol. 257, Pt. A, 2014, pp. 813–829. doi:10.1016/j.jcp.2013.10.019

[41] Coles, D. E., and Hirst, E. A., *Computation of Turbulent Boundary Layers — 1968 AFOSR-IFP-Stanford Conference*, Vol. 2, Stanford Univ., Stanford, CA, 1968.

[42] Rumsey, C. L., "NASA Langley Research Center Turbulence Modeling Resource," http://turbmodels.larc.nasa.gov [retrieved 24 March 2017].

[43] Slater, J., "NPARC Alliance CFD Verification and Validation," https://www.grc.nasa.gov/www/wind/valid/fpturb/fpturb.html [retrieved 24 March 2017].

[44] Drela, M., "XFOIL: An Analysis and Design System for Low Reynolds Number Airfoils," *Low Reynolds Number Aerodynamics*, edited by T. Mueller, Lecture Notes in Engineering, Springer–Verlag, New York, 1989, pp. 1–12.

[45] Hecht, F., "BAMG: Bidimensional Anisotropic Mesh Generator," Technical Rept., INRIA, Rocquencourt, France, 2006.

[46] Degrez, G., Boccadoro, C., and Wendt, J., "The Interaction of an Oblique Shock Wave with a Laminar Boundary Layer Revisited. An Experimental and Numerical Study," *Journal of Fluid Mechanics*, Vol. 177, April 1987, pp. 247–263. doi:10.1017/S0022112087000946

C. W. Rowley
Associate Editor