INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 00, Number 0, Pages 000–000

ANISOTROPIC hp–ADAPTIVE DISCONTINUOUS GALERKIN FINITE ELEMENT METHODS FOR COMPRESSIBLE FLUID FLOWS

STEFANO GIANI AND PAUL HOUSTON

Abstract. In this article we consider the construction of general isotropic and anisotropic adaptive mesh refinement strategies, as well as hp–mesh refinement techniques, for the numerical approximation of the compressible Euler and Navier–Stokes equations. To discretize the latter system of conservation laws, we exploit the (adjoint consistent) symmetric version of the interior penalty discontinuous Galerkin finite element method. The *a posteriori* error indicators are derived based on employing the dual-weighted-residual approach in order to control the error measured in terms of general target functionals of the solution; these error estimates involve the product of the finite element residuals with local weighting terms involving the solution of a certain adjoint problem that must be numerically approximated. This general approach leads to the design of economical finite element meshes specifically tailored to the computation of the target functional of interest, as well as providing efficient error estimation. Numerical experiments demonstrating the performance of the proposed adaptive algorithms will be presented.

Key words. Discontinuous Galerkin methods, a posteriori error estimation, adaptivity, anisotropic hp–refinement, compressible flows

1. Introduction

The development of Discontinuous Galerkin (DG) methods for the numerical approximation of the compressible Euler and Navier-Stokes equations is an extremely exciting research topic which is currently being developed by a number of groups all over the world, cf. [1, 2, 3, 4, 6, 10, 11, 16, 20, 21, 22, 32, 33, 34], for example. DG methods have several important advantages over well established finite volume methods. The concept of higher-order discretization is inherent to the DG method. The stencil is minimal in the sense that each element communicates only with its direct neighbors. In particular, in contrast to the increasing stencil size needed to increase the accuracy of classical finite volume methods, the stencil of DG methods is the same for any order of accuracy, which has important advantages for the implementation of boundary conditions and for the parallel efficiency of the method. Moreover, due to the simple communication at element interfaces, elements with so–called hanging nodes can be easily treated, a fact that simplifies local mesh refinement (h–refinement). Additionally, the communication at element interfaces is identical for any order of the method, which simplifies the use of methods with different polynomial orders p in adjacent elements. This allows for the variation of the order of polynomials over the computational domain (p–refinement), which in combination with h–refinement leads to so–called hp–adaptivity.

Mesh adaptation in finite element discretizations should be based on rigorous *a posteriori* error estimates; for hyperbolic/nearly–hyperbolic equations such estimates should reflect the inherent mechanisms of error propagation (see [26, 27]). These considerations are particularly important when local quantities such as point values, local averages or flux integrals of the analytical solution are to be computed

with high accuracy. In the context of aerodynamic flow simulations, it is of vital importance that certain force coefficients, such as the drag, lift and moment on a body immersed within a compressible fluid, are reliably and efficiently computed. Selective error estimates of this kind can be obtained by the optimal control technique proposed in [8] and [5] which is based on duality arguments analogous to those from the *a priori* error analysis of finite element methods. In the resulting *a posteriori* error estimates, the element-residuals of the computed solution are multiplied by local weights involving the adjoint solution. These weights represent the sensitivity of the relevant error quantity with respect to variations of the local mesh size. Since the adjoint solution is usually unknown analytically, it has to be approximated numerically. On the basis of the resulting *a posteriori* error estimate the current mesh is locally adapted and then new approximations to the primal and adjoint solution are computed. This feed-back process is repeated, for instance, until the required error tolerance is reached. In this way, optimal meshes, or in the *hp*–setting, optimal finite element spaces can be obtained for various kinds of error measures, where *optimal* can mean *most economical for achieving a prescribed accuracy TOL* or *most accurate for a given maximum number N_{max} of degrees of freedom.* This approach is quite universal as it can, in principle, be applied to almost any problem, as long as it is posed in a variational setting.

In this work, we consider the *a posteriori* error estimation and adaptive mesh design of the *hp*–version of the DG finite element method applied to compressible flows on general finite element spaces consisting of an anisotropic computational mesh with anisotropic polynomial degree approximation orders. Here, we shall be interested in the reliable and efficient approximation of certain target functionals of the underlying analytical solution of practical interest. In particular, (weighted) Type I *a posteriori* error bounds are derived, based on employing the dual-weighted-residual approach, cf. [5, 19, 28, 29], for example. Based on the *a posteriori* error bound we design and implement a series of adaptive algorithms to efficiently design the underlying finite element space. Inspired by our recent articles [12, 13], we consider adaptive mesh refinement algorithms based on utilizing anisotropic *h*–refinement, isotropic *hp*–refinement, and finally anisotropic *hp*–refinement. Within this latter strategy, once elements have been marked for refinement/derefinement, on the basis of the size of the local error indicators, the proposed adaptive algorithm consists of two key steps: (a) Determine whether to undertake *h*– or *p*–refinement/derefinement; (b) Select a locally optimal anisotropic/isotropic refinement. Step (a) is based on assessing the local analyticity of the underlying primal and adjoint solutions, on the basis of the decay rates of Legendre series coefficients; see our previous articles [16, 30, 29], together with [7]. Step (b) is based on employing a competitive refinement strategy, whereby the "optimal" refinement is selected from a series of trial refinements. This entails the numerical solution of a series of local primal and adjoint problems which is relatively cheap and fully parallelizable, cf. [13]. The work presented in this paper is a complete and improved account of our recent work announced in the book chapter [14].

This article is structured as follows. In Section 2 we introduce the three–dimensional compressible Navier–Stokes equations. Then, in Section 3 we formulate its discontinuous Galerkin finite element approximation, based on employing the adjoint consistent symmetric interior penalty method introduced in [23]. Then, in

Section 4 we derive an error representation formula together with the corresponding (weighted) Type I a posteriori error bound for general target functionals of the solution. The error representation formula stems from a duality argument and includes computable residual terms multiplied by local weights involving the adjoint solution; the inclusion of the adjoint solution in the Type I bound ensures that the error creation and error propagation mechanisms inherent in compressible fluid flows are reflected by the resulting local error indicators. In Section 5 we outline an anisotropic h–version adaptive algorithm, which is based on employing a competitive refinement strategy; see [12, 15] for the application of this approach to 2D convection–diffusion problems. Numerical experiments for both 2D and 3D viscous flows will be undertaken. Before we embark on anisotropic hp–refinement, in Section 6 we first study the application of standard isotropic hp–refinement to both inviscid and viscous flows. Finally, Section 7 considers the application of the anisotropic hp–refinement algorithm developed in [13] to a viscous flow problem.

2. Compressible Navier-Stokes equations

In this article, we consider both two– and three–dimensional inviscid and laminar compressible flow problems. With this in mind, for generality, in this section we introduce the stationary compressible Navier-Stokes equations in three-dimensions:

$$\nabla \cdot (\mathcal{F}^c(\mathbf{u}) - \mathcal{F}^v(\mathbf{u}, \nabla \mathbf{u})) = 0 \quad \text{in } \Omega, \tag{1}$$

where Ω is an open bounded domain in $\mathbb{R}^d$ with boundary Γ; for the purposes of this section, we set $d = 3$. The vector of conservative variables $\mathbf{u}$ is given by $\mathbf{u} = (\rho, \rho v_1, \rho v_2, \rho v_3, \rho E)^\top$ and the convective flux $\mathcal{F}^c(\mathbf{u}) = (\mathbf{f}_1^c(\mathbf{u}), \mathbf{f}_2^c(\mathbf{u}), \mathbf{f}_3^c(\mathbf{u}))^\top$ is defined by

$$\mathbf{f}_1^c(\mathbf{u}) = \begin{bmatrix} \rho v_1 \\ \rho v_1^2 + p \\ \rho v_1 v_2 \\ \rho v_1 v_3 \\ \rho H v_1 \end{bmatrix}, \ \mathbf{f}_2^c(\mathbf{u}) = \begin{bmatrix} \rho v_2 \\ \rho v_2 v_1 \\ \rho v_2^2 + p \\ \rho v_2 v_3 \\ \rho H v_2 \end{bmatrix}, \text{ and } \mathbf{f}_3^c(\mathbf{u}) = \begin{bmatrix} \rho v_3 \\ \rho v_3 v_1 \\ \rho v_3 v_2 \\ \rho v_3^2 + p \\ \rho H v_3 \end{bmatrix}. \tag{2}$$

Furthermore, writing $\mathcal{F}^v(\mathbf{u}) = (\mathbf{f}_1^v(\mathbf{u}), \mathbf{f}_2^v(\mathbf{u}), \mathbf{f}_3^v(\mathbf{u}))^\top$, we have

$$\mathbf{f}_k^v(\mathbf{u}, \nabla \mathbf{u}) = \begin{pmatrix} 0 \\ \tau_{1k} \\ \tau_{2k} \\ \tau_{3k} \\ \tau_{kl} v_l + \mathcal{K} T_{x_k} \end{pmatrix}, \quad k = 1, 2, 3.$$

Here, ρ, $\mathbf{v} = (v_1, v_2, v_3)^\top$, p, E and T denote the density, velocity vector, pressure, specific total energy, and temperature, respectively. Moreover, $\mathcal{K}$ is the thermal conductivity coefficient and H is the total enthalpy given by

$$H = E + \frac{p}{\rho} = e + \tfrac{1}{2}\mathbf{v}^2 + \frac{p}{\rho},$$

where e is the specific static internal energy, and the pressure is determined by the equation of state of an ideal gas

$$p = (\gamma - 1)\rho e, \tag{3}$$

where $\gamma = c_p/c_v$ is the ratio of specific heat capacities at constant pressure, c_p, and constant volume, c_v; for dry air, $\gamma = 1.4$. For a Newtonian fluid, the viscous stress tensor is given by

$$\tau = \mu \left(\nabla \mathbf{v} + (\nabla \mathbf{v})^\top - \tfrac{2}{3}(\nabla \cdot \mathbf{v}) I \right),$$

where μ is the dynamic viscosity coefficient; the temperature T is given by

$$\mathcal{K} T = \tfrac{\mu \gamma}{Pr} \left(E - \tfrac{1}{2}\mathbf{v}^2 \right),$$

where $\mathrm{Pr} = 0.72$ is the Prandtl number. For the purposes of discretization, we rewrite the compressible Navier–Stokes equations (1) in the following (equivalent) form:

$$\nabla \cdot (\mathcal{F}^c(\mathbf{u}) - G(\mathbf{u}) \nabla \mathbf{u}) \equiv \frac{\partial}{\partial x_k} \left(\mathbf{f}_k^c(\mathbf{u}) - G_{kl}(\mathbf{u}) \frac{\partial \mathbf{u}}{\partial x_l} \right) = 0 \quad \text{in } \Omega.$$

Here, the matrices $G_{kl}(\mathbf{u}) = \partial \mathbf{f}_k^v(\mathbf{u}, \nabla \mathbf{u}) / \partial u_{x_l}$, for $k, l = 1, 2, 3$, are the homogeneity tensors defined by $\mathbf{f}_k^v(\mathbf{u}, \nabla \mathbf{u}) = G_{kl}(\mathbf{u}) \partial \mathbf{u} / \partial x_l$, $k = 1, 2, 3$.

Given that $\Omega \subset \mathbb{R}^3$ is a bounded region, with boundary Γ, the system of conservation laws (1) must be supplemented by appropriate boundary conditions. For simplicity of presentation, we assume that Γ may be decomposed as follows

$$\Gamma = \Gamma_{\text{D,sup}} \cup \Gamma_{\text{D,sub-in}} \cup \Gamma_{\text{D,sub-out}} \cup \Gamma_{\text{W}} \cup \Gamma_{\text{sym}},$$

where $\Gamma_{\text{D,sup}}$, $\Gamma_{\text{D,sub-in}}$, $\Gamma_{\text{D,sub-out}}$, Γ_{W}, and Γ_{sym} are distinct subsets of Γ representing Dirichlet (supersonic), Dirichlet (subsonic-inflow), Dirichlet (subsonic-outflow), solid wall boundaries, and symmetry boundaries, respectively, cf. [21]. We remark that as in [21, 22], Neumann boundary conditions may also be considered; for clarity of presentation, we neglect this case and refer to our earlier articles for details.

Thereby, we may specify the following boundary conditions:

$$\mathcal{B}(\mathbf{u}) = \mathcal{B}(\mathbf{g}) \quad \text{on} \quad \Gamma_{\text{D,sup}} \cup \Gamma_{\text{D,sub-in}} \cup \Gamma_{\text{D,sub-out}},$$

where $\mathbf{g} = (g_1, \ldots, g_5)^\top$ is a prescribed Dirichlet condition. Here, $\mathcal{B}$ is a boundary operator employed to enforce appropriate Dirichlet conditions on $\Gamma_{\text{D,sup}} \cup \Gamma_{\text{D,sub-in}} \cup \Gamma_{\text{D,sub-out}}$. For simplicity of presentation, we assume that

$$\mathcal{B}(\mathbf{u}) = \begin{cases} \mathbf{u} & \text{on } \Gamma_{\text{D,sup}}, \\ (u_1, u_2, u_3, u_4, 0)^\top & \text{on } \Gamma_{\text{D,sub-in}}, \\ \left(0, 0, 0, 0, (\gamma - 1)(u_5 - (u_2^2 + u_3^2 + u_4^2)/(2u_1))\right)^\top & \text{on } \Gamma_{\text{D,sub-out}}; \end{cases}$$

we note that this latter condition enforces a specific pressure $p_{\text{out}} = (\mathcal{B}(\mathbf{g}))_5$ on $\Gamma_{\text{D,sub-out}}$.

For solid wall boundaries, we consider *isothermal* and *adiabatic* conditions; to this end, decomposing $\Gamma_W = \Gamma_{\text{iso}} \cup \Gamma_{\text{adia}}$, we set

$$\mathbf{v} = \mathbf{0} \quad \text{on} \quad \Gamma_{\text{W}}, \qquad T = T_{\text{wall}} \quad \text{on} \quad \Gamma_{\text{iso}}, \qquad \mathbf{n} \cdot \nabla T = 0 \quad \text{on} \quad \Gamma_{\text{adia}},$$

where T_{wall} is a given wall temperature; see [3, 1, 6, 9] and the references cited therein for further details. On the symmetry boundary, we simply impose that the normal component of the velocity is zero; see below for further details.

3. DG Discretization

In this section we introduce the adjoint-consistent interior penalty DG discretization of the compressible Navier–Stokes equations (1), cf. [23] for further details. First, we begin by introducing some notation. We assume that $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, can be subdivided into a mesh $\mathcal{T}_h = \{\kappa\}$ consisting of tensor-product (quadrilaterals, $d = 2$, and hexahedra, $d = 3$) open element domains κ. For each $\kappa \in \mathcal{T}_h$, we denote by $\mathbf{n}_\kappa$, the unit outward normal vector to the boundary $\partial\kappa$. We assume that each $\kappa \in \mathcal{T}_h$ is an image of a fixed reference element $\hat{\kappa}$, that is, $\kappa = \sigma_\kappa(\hat{\kappa})$ for all $\kappa \in \mathcal{T}_h$, where $\hat{\kappa}$ is the open unit hypercube in $\mathbb{R}^d$, and σ_κ is a smooth bijective mapping. On the reference element $\hat{\kappa}$ we define the polynomial space $\mathcal{Q}_{\mathbf{p}}$ with respect to the anisotropic polynomial degree vector $\mathbf{p} := \{p_i\}_{i=1,\ldots,d}$ as follows:

$$\mathcal{Q}_{\mathbf{p}} = \text{span}\{\Pi_{i=1}^d \hat{x}_i^j : 0 \leq j \leq p_i\}.$$

With this notation, we introduce the following (anisotropic) finite element space.

Definition 3.1. *Let $\mathbf{p} = (\mathbf{p}_\kappa : \kappa \in \mathcal{T}_h)$ be the composite polynomial degree vector of the elements in a given finite element mesh $\mathcal{T}_h$. We define the finite element space with respect to Ω, $\mathcal{T}_h$, and $\mathbf{p}$ by*

$$\mathbf{V}_{h,\mathbf{p}} = \{u \in L_2(\Omega) : u|_\kappa \circ \sigma_\kappa \in [\mathcal{Q}_{\mathbf{p}_\kappa}]^{d+2}\}.$$

In the case when the elemental polynomial degree vector $\mathbf{p}_\kappa = \{p_{\kappa,i}\}_{i=1,\ldots,d}$, $\kappa \in \mathcal{T}_h$, is isotropic in the sense that

$$p_{\kappa,1} = p_{\kappa,2} = \ldots = p_{\kappa,d} \equiv p_\kappa$$

for all elements κ in the finite element mesh $\mathcal{T}_h$, then we write $\mathbf{V}_{h,\mathbf{p}_{\text{iso}}}$ in lieu of $\mathbf{V}_{h,\mathbf{p}}$, where $\mathbf{p}_{\text{iso}} = (p_\kappa : \kappa \in \mathcal{T}_h)$. Additionally, in the case when the polynomial degree is both isotropic and uniformly distributed over the mesh $\mathcal{T}_h$, i.e., when $p_\kappa = p$ for all κ in $\mathcal{T}_h$, then we simply denote the finite element space by $\mathbf{V}_{h,p}$.

An *interior face* of $\mathcal{T}_h$ is defined as the (non-empty) $(d-1)$–dimensional interior of $\partial\kappa^+ \cap \partial\kappa^-$, where κ^+ and κ^- are two adjacent elements of $\mathcal{T}_h$, not necessarily matching. A *boundary face* of $\mathcal{T}_h$ is defined as the (non-empty) $(d-1)$–dimensional interior of $\partial\kappa \cap \Gamma$, where κ is a boundary element of $\mathcal{T}_h$. We denote by $\Gamma_{\mathcal{I}}$ the union of all interior faces of $\mathcal{T}_h$. Let κ^+ and κ^- be two adjacent elements of $\mathcal{T}_h$, and $\mathbf{x}$ an arbitrary point on the interior face $f = \partial\kappa^+ \cap \partial\kappa^-$. Furthermore, let $\mathbf{v}$ and $\underline{\tau}$ be vector- and matrix-valued functions, respectively, that are smooth inside each element $\kappa^\pm$. By $(\mathbf{v}^\pm, \underline{\tau}^\pm)$, we denote the traces of $(\mathbf{v}, \underline{\tau})$ on f taken from within the interior of $\kappa^\pm$, respectively. Then, the averages of $\mathbf{v}$ and $\underline{\tau}$ at $\mathbf{x} \in f$ are given by $\{\!\{\mathbf{v}\}\!\} = (\mathbf{v}^+ + \mathbf{v}^-)/2$ and $\{\!\{\underline{\tau}\}\!\} = (\underline{\tau}^+ + \underline{\tau}^-)/2$, respectively. Similarly, the jump of $\mathbf{v}$ at $\mathbf{x} \in f$ is given by $\underline{[\![\mathbf{v}]\!]} = \mathbf{v}^+ \otimes \mathbf{n}_{\kappa^+} + \mathbf{v}^- \otimes \mathbf{n}_{\kappa^-}$, where we denote by $\mathbf{n}_{\kappa^\pm}$ the unit outward normal vector of $\kappa^\pm$, respectively. On $f \subset \Gamma$, we set $\{\!\{\mathbf{v}\}\!\} = \mathbf{v}$, $\{\!\{\underline{\tau}\}\!\} = \underline{\tau}$ and $\underline{[\![\mathbf{v}]\!]} = \mathbf{v} \otimes \mathbf{n}$, where $\mathbf{n}$ denotes the unit outward normal vector to Γ. For matrices $\underline{\sigma}, \underline{\tau} \in \mathbb{R}^{m \times n}$, $m, n \geq 1$, we use the standard notation $\underline{\sigma} : \underline{\tau} = \sum_{k=1}^m \sum_{l=1}^n \sigma_{kl}\tau_{kl}$; additionally, for vectors $\mathbf{v} \in \mathbb{R}^m, \mathbf{w} \in \mathbb{R}^n$, the matrix $\mathbf{v} \otimes \mathbf{w} \in \mathbb{R}^{m \times n}$ is defined by $(\mathbf{v} \otimes \mathbf{w})_{kl} = v_k\, w_l$.

The DG discretization of (1) is given by: find $\mathbf{u}_h \in \mathbf{V}_{h,\mathbf{p}}$ such that

$$\begin{aligned}\mathcal{N}(\mathbf{u}_h,\mathbf{v}) \equiv & -\int_\Omega \mathcal{F}^c(\mathbf{u}_h):\nabla_h\mathbf{v}\,\mathrm{d}\mathbf{x}+\sum_{\kappa\in\mathcal{T}_h}\int_{\partial\kappa\setminus\Gamma}\mathcal{H}(\mathbf{u}_h^+,\mathbf{u}_h^-,\mathbf{n}^+)\cdot\mathbf{v}^+\,\mathrm{d}s\\ &+\int_\Omega \mathcal{F}^v(\mathbf{u}_h,\nabla_h\mathbf{u}_h):\nabla_h\mathbf{v}\,\mathrm{d}\mathbf{x}-\int_{\Gamma_\mathcal{I}}\{\!\{\mathcal{F}^v(\mathbf{u}_h,\nabla_h\mathbf{u}_h)\}\!\}:\underline{[\![\mathbf{v}]\!]}\,\mathrm{d}s\\ &-\int_{\Gamma_\mathcal{I}}\{\!\{G^\top(\mathbf{u}_h)\nabla_h\mathbf{v}\}\!\}:\underline{[\![\mathbf{u}_h]\!]}\,\mathrm{d}s+\int_{\Gamma_\mathcal{I}}\underline{\delta}(\mathbf{u}_h):\underline{[\![\mathbf{v}]\!]}\,\mathrm{d}s\\ &+\mathcal{N}_{\Gamma\setminus\Gamma_{\mathrm{sym}}}(\mathbf{u}_h,\mathbf{v})+\mathcal{N}_{\Gamma_{\mathrm{sym}}}(\mathbf{u}_h,\mathbf{v})=0\end{aligned}\tag{4}$$

for all $\mathbf{v}$ in $\mathbf{V}_{h,\mathbf{p}}$. The subscript h on the operator ∇_h is used to denote the discrete counterpart of ∇, defined elementwise. Here, $\mathcal{H}(\cdot,\cdot,\cdot)$ denotes the (convective) numerical flux function; this may be chosen to be any two–point monotone Lipschitz function which is both consistent and conservative. For the purposes of this article, we employ the Vijayasundaram flux.

In order to define the penalization function $\underline{\delta}(\cdot)$ arising in the DG method (4), we first introduce the local (anisotropic) mesh and polynomial functions $\mathtt{h}$ and $\mathtt{p}$, respectively. To this end, the function $\mathtt{h}$ in $L_\infty(\Gamma_\mathcal{I}\cup\Gamma)$ is defined as $\mathtt{h}(\mathbf{x}) = \min\{m_{\kappa^+},m_{\kappa^-}\}/m_f$, if $\mathbf{x}$ is in the interior of $f=\partial\kappa^+\cap\partial\kappa^-$ for two neighboring elements in the mesh $\mathcal{T}_h$, and $\mathtt{h}(\mathbf{x})=m_\kappa/m_f$, if $\mathbf{x}$ is in the interior of $f=\partial\kappa\cap\Gamma$. Here, for a given (open) bounded set $\omega\subset\mathbb{R}^s$, $s\geq 1$, we write m_ω to denote the s–dimensional measure (volume) of ω. In a similar fashion, we define $\mathtt{p}$ in $L_\infty(\Gamma_\mathcal{I}\cup\Gamma)$ by $\mathtt{p}(\mathbf{x})=\max\{p_{\kappa^+,i},p_{\kappa^-,j}\}$ for κ^+, κ^- as above, where the indices i and j are chosen such that $\sigma_{\kappa^+}^{-1}(f)$ and $\sigma_{\kappa^-}^{-1}(f)$ are orthogonal to the ith–, respectively, jth–coordinate direction on the reference element $\hat\kappa$. For $\mathbf{x}$ in the interior of a boundary face $f=\partial\kappa\cap\Gamma$, we write $\mathtt{p}(\mathbf{x})=p_{\kappa,i}$, when $\sigma_\kappa^{-1}(f)$ is orthogonal to the ith–coordinate direction on $\hat\kappa$. With this notation the penalization term is given by

$$\underline{\delta}(\mathbf{u}_h) = C_{\mathrm{IP}}\frac{\mathtt{p}^2}{\mathtt{h}}\{\!\{G(\mathbf{u}_h)\}\!\}\underline{[\![\mathbf{u}_h]\!]},$$

where C_{IP} is a (sufficiently large) positive constant, cf. [13].

Finally, we define the boundary terms present in the forms $\mathcal{N}_{\Gamma\setminus\Gamma_{\mathrm{sym}}}(\cdot,\cdot)$ and $\mathcal{N}_{\Gamma_{\mathrm{sym}}}(\cdot,\cdot)$. To this end, we write

$$\begin{aligned}\mathcal{N}_{\Gamma\setminus\Gamma_{\mathrm{sym}}}(\mathbf{u}_h,\mathbf{v}) = & \int_{\Gamma\setminus\Gamma_{\mathrm{sym}}}\mathcal{H}_\Gamma(\mathbf{u}_h^+,\mathbf{u}_\Gamma(\mathbf{u}_h^+),\mathbf{n}^+)\cdot\mathbf{v}^+\,\mathrm{d}s+\int_{\Gamma\setminus\Gamma_{\mathrm{sym}}}\underline{\delta}_\Gamma(\mathbf{u}_h^+):\mathbf{v}\otimes\mathbf{n}\,\mathrm{d}s\\ &-\int_{\Gamma\setminus\Gamma_{\mathrm{sym}}}\mathbf{n}\cdot\mathcal{F}_\Gamma^v(\mathbf{u}_\Gamma(\mathbf{u}_h^+),\nabla_h\mathbf{u}_h^+)\,\mathbf{v}^+\,\mathrm{d}s\\ &-\int_{\Gamma\setminus\Gamma_{\mathrm{sym}}}\left(G_\Gamma^\top(\mathbf{u}_h^+)\nabla_h\mathbf{v}_h^+\right):\left(\mathbf{u}_h^+-\mathbf{u}_\Gamma(\mathbf{u}_h^+)\right)\otimes\mathbf{n}\,\mathrm{d}s,\end{aligned}$$

where

$$\underline{\delta}_\Gamma(\mathbf{u}_h)=C_{\mathrm{IP}}\frac{\mathtt{p}^2}{\mathtt{h}}G_\Gamma(\mathbf{u}_h^+)\left(\mathbf{u}_h-\mathbf{u}_\Gamma(\mathbf{u}_h)\right)\otimes\mathbf{n}.$$

Here, the viscous boundary flux $\mathcal{F}_\Gamma^v$ and the corresponding homogeneity tensor G_Γ are defined by

$$\mathcal{F}_\Gamma^v(\mathbf{u}_h,\nabla\mathbf{u}_h)=\mathcal{F}^v(\mathbf{u}_\Gamma(\mathbf{u}_h),\nabla\mathbf{u}_h)=G_\Gamma(\mathbf{u}_h)\nabla\mathbf{u}_h=G(\mathbf{u}_\Gamma(\mathbf{u}_h))\nabla\mathbf{u}_h.$$

Furthermore, on portions of the boundary Γ where adiabatic boundary conditions are imposed, $\mathcal{F}_\Gamma^v$ and G_Γ are modified such that $\mathbf{n}\cdot\nabla T=0$. The convective

boundary flux $\mathcal{H}_\Gamma$ is defined by

$$\mathcal{H}_\Gamma(\mathbf{u}_h^+, \mathbf{u}_\Gamma(\mathbf{u}_h^+), \mathbf{n}) = \mathbf{n} \cdot \mathcal{F}^c(\mathbf{u}_\Gamma(\mathbf{u}_h^+)).$$

The boundary function $\mathbf{u}_\Gamma(\mathbf{u})$ is given according to the type of boundary condition imposed. To this end, we set

$$\mathbf{u}_\Gamma(\mathbf{u}) = \begin{cases} \mathbf{g} & \text{on } \Gamma_{\text{D,sup}}, \\ (g_1, g_2, g_3, g_4, \frac{p(\mathbf{u})}{\gamma-1} + (g_2^2+g_3^2+g_4^2)/(2g_1))^\top & \text{on } \Gamma_{\text{D,sub-in}}, \\ (u_1, u_2, u_3, u_4, \frac{p_{\text{out}}}{\gamma-1} + (u_2^2+u_3^2+u_4^2)/(2u_1))^\top & \text{on } \Gamma_{\text{D,sub-out}}. \end{cases}$$

Here, $p \equiv p(\mathbf{u})$ denotes the pressure evaluated using the equation of state (3). On Γ_{iso}, we set $\mathbf{u}_\Gamma(\mathbf{u}) = (u_1, 0, 0, 0, u_1 c_v T_{\text{wall}})^\top$, while $\mathbf{u}_\Gamma(\mathbf{u}) = (u_1, 0, 0, 0, u_5)^\top$ on Γ_{adia}, cf. [23], for example.

On the symmetry boundary, we employ the same technique introduced in [25]. To this end, we define

$$\mathbf{u}_\Gamma(\mathbf{u}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1-2n_1^2 & -2n_1n_2 & -2n_1n_3 & 0 \\ 0 & -2n_1n_2 & 1-2n_2^2 & -2n_2n_3 & 0 \\ 0 & -2n_1n_3 & -2n_2n_3 & 1-2n_3^2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{u} \quad \text{on} \quad \Gamma_{\text{sym}}, \tag{5}$$

where $\mathbf{n} = (n_1, n_2, n_3)^\top$ is the unit outward normal vector to the boundary. Additionally, it is necessary to introduce a suitable approximation of $\nabla \mathbf{u}_h^-$; to this end, we introduce the following gradient operator

$$(\nabla u)_{\Gamma,jl}(\mathbf{u}_h) = \partial_{u_m} u_\Gamma^j(\mathbf{u}_h) \partial_{x_k} u_h^m (\delta_{kl} - 2n_k n_l).$$

With this notation, the form $\mathcal{N}_{\Gamma_{\text{sym}}}(\cdot,\cdot)$ is defined as follows

$$\begin{aligned} \mathcal{N}_{\Gamma_{\text{sym}}}(\mathbf{u}_h, \mathbf{v}) \quad = \quad & \int_{\Gamma_{\text{sym}}} \mathcal{H}_\Gamma(\mathbf{u}_h^+, \mathbf{u}_\Gamma(\mathbf{u}_h^+), \mathbf{n}^+) \cdot \mathbf{v}^+ \, \mathrm{d}s + \int_\Gamma \underline{\delta}_{\Gamma_{\text{sym}}}(\mathbf{u}_h^+) : \mathbf{v}^+ \otimes \mathbf{n} \, \mathrm{d}s \\ & - \frac{1}{2} \int_{\Gamma_{\text{sym}}} \left(\mathcal{F}^v(\mathbf{u}_h^+, \nabla_h \mathbf{u}_h^+) + \mathcal{F}^v(\mathbf{u}_\Gamma(\mathbf{u}_h^+), (\nabla \mathbf{u})_\Gamma(\mathbf{u}_h^+)) \right) : \mathbf{v}^+ \otimes \mathbf{n} \, \mathrm{d}s \\ & - \frac{1}{2} \int_\Gamma \left(G^\top(\mathbf{u}_h^+) \nabla_h \mathbf{v}_h^+ \right) : \left(\mathbf{u}_h^+ - \mathbf{u}_\Gamma(\mathbf{u}_h^+) \right) \otimes \mathbf{n} \, \mathrm{d}s, \end{aligned}$$

where

$$\underline{\delta}_{\Gamma_{\text{sym}}}(\mathbf{u}_h) = C_{\text{IP}} \frac{\mathtt{p}^2}{\mathtt{h}} \frac{1}{2} \left(G(\mathbf{u}_h^+) + G(\mathbf{u}_\Gamma(\mathbf{u}_h^+)) \right) (\mathbf{u}_h - \mathbf{u}_\Gamma(\mathbf{u}_h)) \otimes \mathbf{n}.$$

4. *A posteriori* error estimation

In this section we briefly outline the derivation of an adjoint-based *a posteriori* bound on the error in a given computed target functional $J(\cdot)$ of practical interest, such as the drag, lift, or moment on a body immersed within a compressible fluid, for example; see [8, 5] for further details.

Assuming that the functional of interest $J(\cdot)$ is differentiable, we write $\bar{J}(\cdot;\cdot)$ to denote the mean value linearization of $J(\cdot)$ defined by

$$\bar{J}(\mathbf{u}, \mathbf{u}_h; \mathbf{u} - \mathbf{u}_h) = J(\mathbf{u}) - J(\mathbf{u}_h) = \int_0^1 J'[\theta \mathbf{u} + (1-\theta)\mathbf{u}_h](\mathbf{u} - \mathbf{u}_h) \, \mathrm{d}\theta,$$

where $J'[\mathbf{w}](\cdot)$ denotes the Fréchet derivative of $J(\cdot)$ evaluated at some $\mathbf{w}$ in $\mathbf{V}$. Here, $\mathbf{V}$ is some suitably chosen function space such that $\mathbf{V}_{h,\mathbf{p}} \subset \mathbf{V}$.

Analogously, for $\mathbf{v}$ in $\mathbf{V}$, we define the mean–value linearization of $\mathcal{N}(\cdot,\mathbf{v})$ by

$$\mathcal{M}(\mathbf{u},\mathbf{u}_h;\mathbf{u}-\mathbf{u}_h,\mathbf{v})=\mathcal{N}(\mathbf{u},\mathbf{v})-\mathcal{N}(\mathbf{u}_h,\mathbf{v})=\int_0^1 \mathcal{N}'[\theta\mathbf{u}+(1-\theta)\mathbf{u}_h](\mathbf{u}-\mathbf{u}_h,\mathbf{v})\,\mathrm{d}\theta.$$

Here, $\mathcal{N}'[\mathbf{w}](\cdot,\mathbf{v})$ denotes the Fréchet derivative of $\mathbf{u}\mapsto\mathcal{N}(\mathbf{u},\mathbf{v})$, for $\mathbf{v}\in\mathbf{V}$ fixed, at some $\mathbf{w}$ in $\mathbf{V}$. Let us now introduce the adjoint problem: find $\mathbf{z}\in\mathbf{V}$ such that

$$\mathcal{M}(\mathbf{u},\mathbf{u}_h;\mathbf{w},\mathbf{z})=\bar{J}(\mathbf{u},\mathbf{u}_h;\mathbf{w})\quad\forall\mathbf{w}\in\mathbf{V}. \tag{6}$$

With this notation, we may state the following error representation formula

$$J(\mathbf{u})-J(\mathbf{u}_h)=\mathcal{R}_\Omega(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)\equiv\sum_{\kappa\in\mathcal{T}_h}\eta_\kappa, \tag{7}$$

where $\mathcal{R}_\Omega(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)=-\mathcal{N}(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)$ includes primal residuals multiplied by the difference of the adjoint solution $\mathbf{z}$ and an arbitrary discrete function $\mathbf{z}_h\in\mathbf{V}_{h,\mathbf{p}}$, and η_κ denotes the local elemental indicators; see [20, 22] for details. Upon application of the triangle inequality, we deduce that

$$|J(\mathbf{u})-J(\mathbf{u}_h)|\le\mathcal{R}_{|\Omega|}(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)\equiv\sum_{\kappa\in\mathcal{T}_h}|\eta_\kappa|\,. \tag{8}$$

We note that the error representation formula (7) depends on the unknown analytical solution $\mathbf{z}$ to the adjoint problem (6) which in turn depends on the unknown analytical solution $\mathbf{u}$. Thus, in order to render these quantities computable, both $\mathbf{u}$ and $\mathbf{z}$ must be replaced by suitable approximations. Here, the linearizations leading to $\mathcal{M}(\mathbf{u},\mathbf{u}_h;\cdot,\cdot)$ and $\bar{J}(\mathbf{u},\mathbf{u}_h;\cdot)$ are performed about $\mathbf{u}_h$ and the adjoint solution $\mathbf{z}$ is approximated by computing the DG approximation $\bar{\mathbf{z}}_{\bar{h}}\in\mathbf{V}_{\bar{h},\mathbf{p}_d}$, where $\mathbf{V}_{\bar{h},\mathbf{p}_d}$ is an *adjoint* finite element space consisting of (discontinuous) piecewise polynomials of composite degree $\mathbf{p}_d$ represented on a computational mesh $\mathcal{T}_{\bar{h}}$ of granularity $\bar{h}$. On the basis of numerical experimentation, in this article we set $\mathbf{V}_{\bar{h},\mathbf{p}_d}=\mathbf{V}_{h,\mathbf{p}_d}$, where $\mathbf{p}_d=\mathbf{p}+\mathbf{1}$, cf. [19, 29]; thereby, in this setting, we write $\bar{\mathbf{z}}_h$, in lieu of $\bar{\mathbf{z}}_{\bar{h}}$, to denote the approximate adjoint solution sought from the finite element space $\mathbf{V}_{h,\mathbf{p}_d}$.

In the following sections we consider the development of a variety of adaptive mesh refinement algorithms in order to efficiently control the error in the computed target functional of interest.

5. Anisotropic mesh adaptation

In this section we first consider the automatic design of anisotropic finite element meshes $\mathcal{T}_h$, assuming that the underlying polynomial degree distribution is both uniform and fixed, i.e., when $\mathbf{u}_h\in\mathbf{V}_{h,p}$. Thereby, for a user-defined tolerance $\mathtt{TOL}$, we now consider the problem of designing an appropriate finite element mesh $\mathcal{T}_h$ such that

$$|J(\mathbf{u})-J(\mathbf{u}_h)|\le\mathtt{TOL}\,,$$

subject to the constraint that the total number of elements in $\mathcal{T}_h$ is minimized. Following the discussion presented in [28], we exploit the *a posteriori* error bound (8) with $\mathbf{z}$ replaced by the numerical approximation $\bar{\mathbf{z}}_h\in\mathbf{V}_{h,p_d}$, with $p_d=p+1$, cf. above. Thereby, in practice we enforce the stopping criterion

$$\mathcal{R}_{|\Omega|}(\mathbf{u}_h,\bar{\mathbf{z}}_h-\mathbf{z}_h)\le\mathtt{TOL}\,. \tag{9}$$

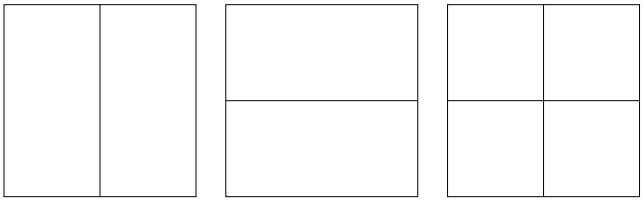


FIGURE 1. Cartesian refinement in 2D: (a) & (b) Anisotropic refinement; (c) Isotropic refinement.

If (9) is not satisfied, then the elements are marked for refinement/derefinement according to the size of the (approximate) error indicators $|\bar{\eta}_\kappa|$, based on employing a fixed fraction strategy, for example. Here, $\bar{\eta}_\kappa$ is defined analogously to η_κ in (7) with $\mathbf{z}$ replaced by $\bar{\mathbf{z}}_h$. To subdivide the elements which have been flagged for refinement, we employ a simple Cartesian refinement strategy; here, elements may be subdivided either anisotropically or isotropically according to the three refinements (in two–dimensions, i.e., $d = 2$) depicted in Figure 1. In order to determine the optimal refinement, we exploit the following strategy based on choosing the most competitive subdivision of κ from a series of trial refinements, whereby an approximate local error indicator on each trial patch is determined, cf. [12, 15].

Algorithm 5.1. *Given an element κ in the computational mesh $\mathcal{T}_h$ (which has been marked for refinement), we first construct the mesh patches $\mathcal{T}_{h,i}$, $i = 1, 2, 3$, based on refining κ according to Figures 1(a), (b), & (c), respectively. On each mesh patch, $\mathcal{T}_{h,i}$, $i = 1, 2, 3$, we compute the approximate error estimators*

$$\mathcal{R}_{\kappa,i}(\mathbf{u}_{h,i}, \bar{\mathbf{z}}_{h,i} - \mathbf{z}_h) = \sum_{\kappa' \in \mathcal{T}_{h,i}} \eta_{\kappa',i},$$

for $i = 1, 2, 3$, respectively. Here, $\mathbf{u}_{h,i}$, $i = 1, 2, 3$, is the DG approximation computed on the mesh patch $\mathcal{T}_{h,i}$, $i = 1, 2, 3$, respectively, based on enforcing appropriate boundary conditions on $\partial\kappa$ computed from the original DG solution $\mathbf{u}_h$ on the portion of the boundary $\partial\kappa$ of κ which is interior to the computational domain Ω, i.e., where $\partial\kappa \cap \Gamma = \emptyset$. Similarly, $\bar{\mathbf{z}}_{h,i}$ denotes the DG approximation to $\mathbf{z}$ computed on the local mesh patch $\mathcal{T}_{h,i}$, $i = 1, 2, 3$, respectively, with polynomials of degree p_d, based on employing suitable boundary conditions on $\partial\kappa \cap \Gamma = \emptyset$ derived from $\bar{\mathbf{z}}_h$. Finally, $\eta_{\kappa',i}$, $i = 1, 2, 3$, is defined in an analogous manner to η_κ, cf. above, with $\mathbf{u}_h$ and $\mathbf{z}$ replaced by $\mathbf{u}_{h,i}$ and $\bar{\mathbf{z}}_{h,i}$, respectively.

The element κ is then refined according to the subdivision of κ which satisfies

$$\min_{i=1,2,3} \frac{|\eta_\kappa| - |\mathcal{R}_{\kappa,i}(\mathbf{u}_{h,i}, \bar{\mathbf{z}}_{h,i} - \mathbf{z}_h)|}{\#dofs(\mathcal{T}_{h,i}) - \#dofs(\kappa)},$$

where $\#dofs(\kappa)$ and $\#dofs(\mathcal{T}_{h,i})$, $i = 1, 2, 3$, denote the number of degrees of freedom associated with κ and $\mathcal{T}_{h,i}$, $i = 1, 2, 3$, respectively, cf. [12].

The extension of this approach to the case when $\mathcal{T}_h$ is a hexahedral mesh in three-dimensions follows in an analogous fashion. Indeed, in this setting, we again employ a Cartesian refinement strategy whereby elements may be subdivided either isotropically or anisotropically according to the four refinements depicted in

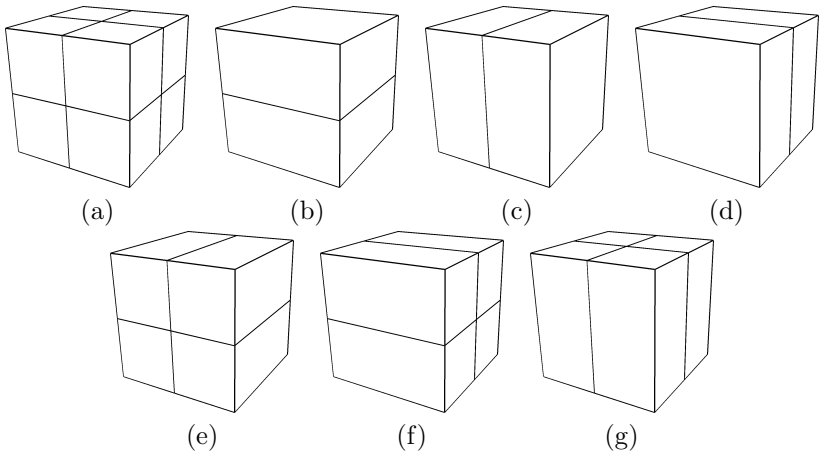


FIGURE 2. Cartesian refinement in 3D.

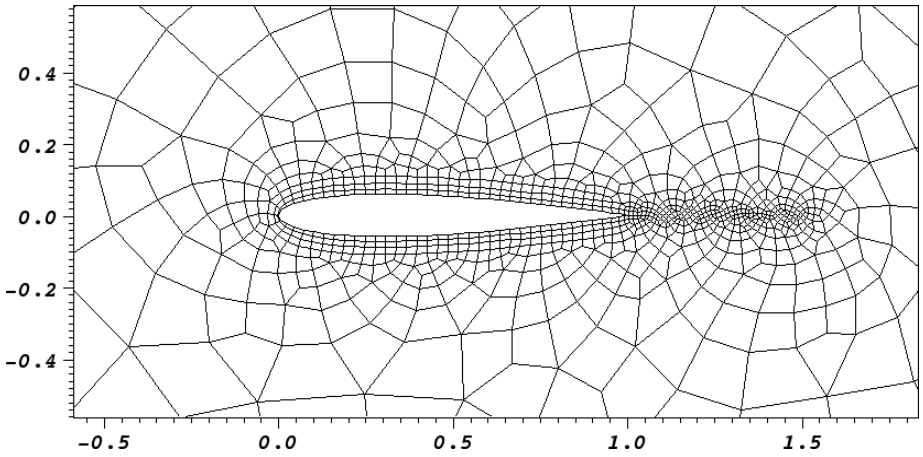


FIGURE 3. Example 1: Zoom of initial mesh with 1134 elements.

Figures 2(a)–(d). We remark that we assume that a face in the computational mesh is a complete face of at least one element. This assumption means that the refinements depicted in Figures 2(b)–(d) may be inadmissible. In this situation, we replace the selected refinement by either one of the anisotropic mesh refinements depicted in Figures 2(e)–(g), or if necessary, an isotropic refinement is performed.

5.1. Numerical experiments. In this section we present a number of experiments to numerically demonstrate the performance of the anisotropic adaptive algorithm outlined in the previous section.

5.1.1. Example 1: Laminar flow around a NACA0012 airfoil. In this example, we consider the subsonic viscous flow around a NACA0012 airfoil; here, the upper and lower surfaces of the airfoil geometry are specified by the function $g^{\pm}$,

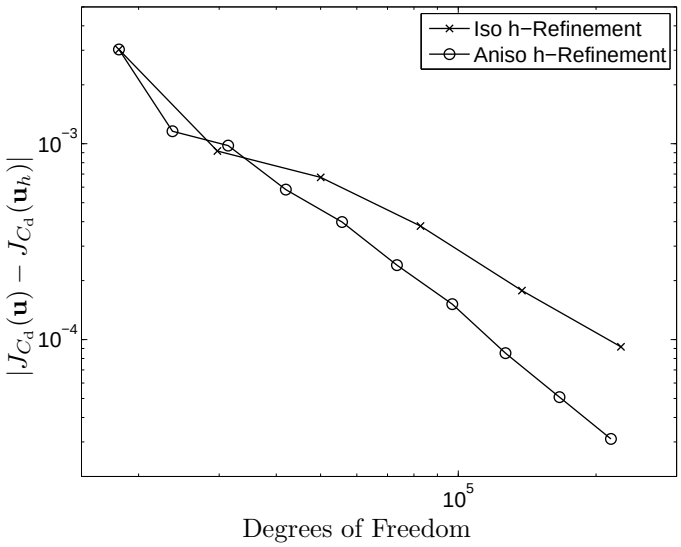


FIGURE 4. Example 1: Comparison between adaptive isotropic and anisotropic mesh refinement.

respectively, where

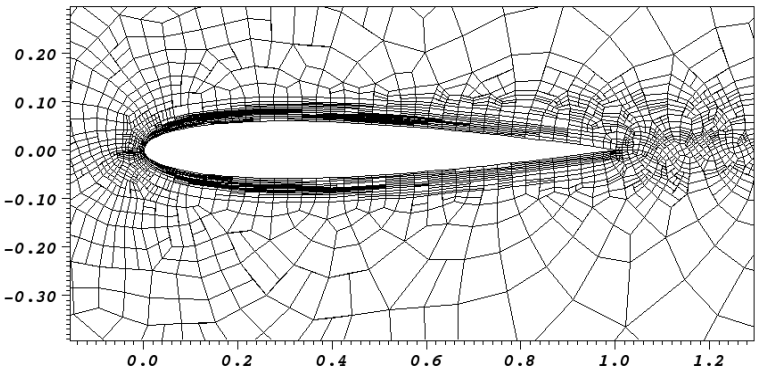
$$g^{\pm}(s) = \pm 5 \times 0.12 \times (0.2969 s^{1/2} - 0.126 s - 0.3516 s^2 + 0.2843 s^3 - 0.1015 s^4).$$

As the chord length l of the airfoil is $l \approx 1.00893$ we use a rescaling of g in order to yield an airfoil of unit (chord) length. At the farfield (inflow) boundary we specify a Mach 0.5 flow at an angle of attack $\alpha = 2^\circ$, with Reynolds number $\mathrm{Re} = 5000$; on the walls of the airfoil geometry, we impose a zero heat flux (adiabatic) no-slip boundary condition. This is a standard laminar test case which has been investigated by many other authors, cf. [1, 21], for example, and serves as one of the test cases for the EU project ADIGMA [31].

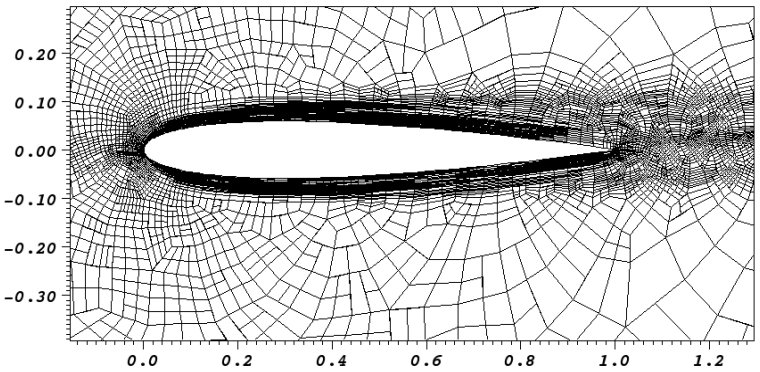
Here, we consider the estimation of the drag coefficient C_d; i.e., the target functional of interest is given by

$$J(\cdot) \equiv J_{C_\mathrm{d}}(\cdot),$$

where $J_{C_\mathrm{d}}(\cdot)$ is defined as the adjoint consistent approximation to C_d, cf. [17, 18, 24]. The initial starting mesh is taken to be an unstructured quadrilateral–dominant hybrid mesh consisting of both quadrilateral and triangular elements; here, the total number of elements is 1134; see Figure 3. Furthermore, curved boundaries are approximated by piecewise quadratic polynomials. In Figure 4 we plot the error in the computed target functional $J_{C_\mathrm{d}}(\cdot)$ using both an isotropic (only) mesh refinement algorithm, together with the anisotropic refinement strategy outlined in Section 5. From Figure 4, we observe the superiority of employing the anisotropic mesh refinement algorithm in comparison with standard isotropic subdivision of the elements. Indeed, the error $|J_{C_\mathrm{d}}(\mathbf{u}) - J_{C_\mathrm{d}}(\mathbf{u}_h)|$ computed on the series of anisotropically refined meshes designed using the proposed algorithm outlined in Section 5 is (almost) always less than the corresponding quantity computed on the isotropic grids. Indeed, on the final mesh anisotropic mesh refinement leads to an improvement in $|J_{C_\mathrm{d}}(\mathbf{u}) - J_{C_\mathrm{d}}(\mathbf{u}_h)|$ of over 60% compared with the same quantity



(a)



(b)

FIGURE 5. Example 1: Anisotropic mesh after (a) 4 adaptive refinements, with 3485 elements; (b) 8 adaptive refinements, with 10410 elements.

computed using isotropic mesh refinement. The meshes generated after 4 and 8 anisotropic adaptive mesh refinements are shown in Figures 5(a) & (b), respectively. Here, we clearly observe significant anisotropic refinement of the viscous boundary layer, as we would expect.

5.1.2. Example 2: Laminar flow around a streamlined body. In this second example we consider laminar flow past a streamlined three–dimensional body. Here, the geometry of the body is based on a 10 percent thick airfoil with boundaries constructed by a surface of revolution. More precisely, the (half) geometry is given by the following expression

$$
\begin{array}{ll}
16(x-1/4)^2+400z^2=1, & 0\le x\le 1/3,\ 0\le y\le 1/100,\\
z=1/(10\sqrt{2})\,(1-x), & 1/3<x\le 1,\ 0\le y\le 1/100,\ z>0,\\
z=-1/(10\sqrt{2})\,(1-x), & 1/3<x\le 1,\ 0\le y\le 1/100,\ z<0,\\
16(x-1/4)^2+400(z^2+(y-1/100)^2)=1, & 0\le x\le 1/3,\ y>1/100,\\
200(z^2+(y-1/100)^2)-(1-x)^2=0, & 1/3\le x\le 1,\ y>1/100,
\end{array}
$$

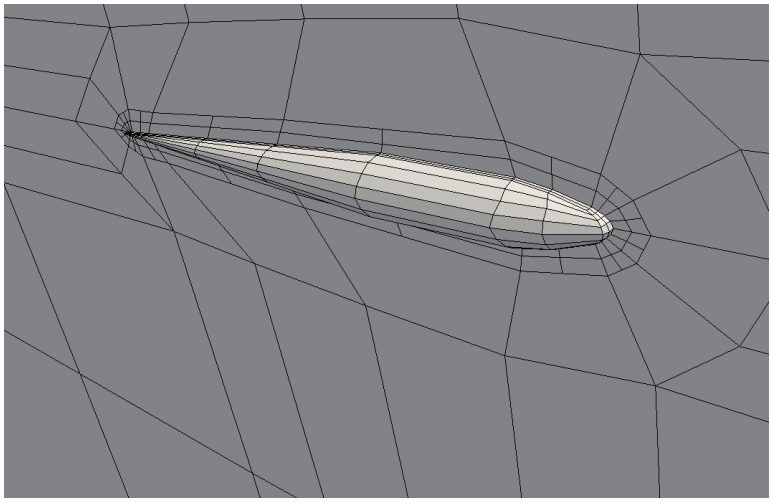


FIGURE 6. Example 2: Initial coarse mesh on the body surface and the symmetry plane.

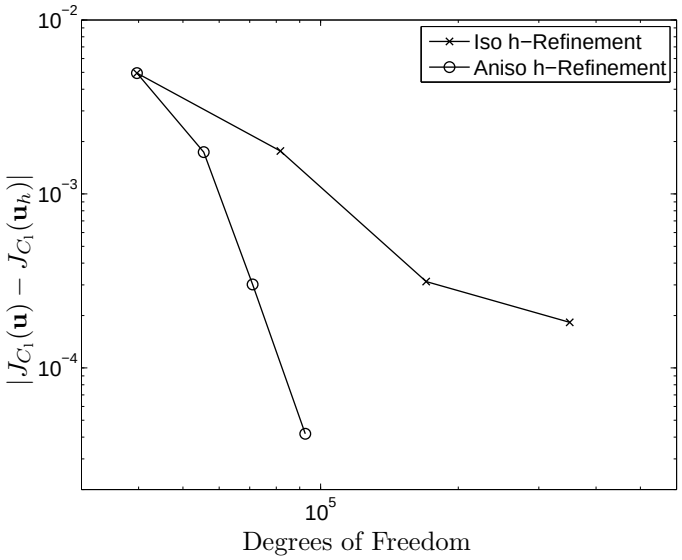
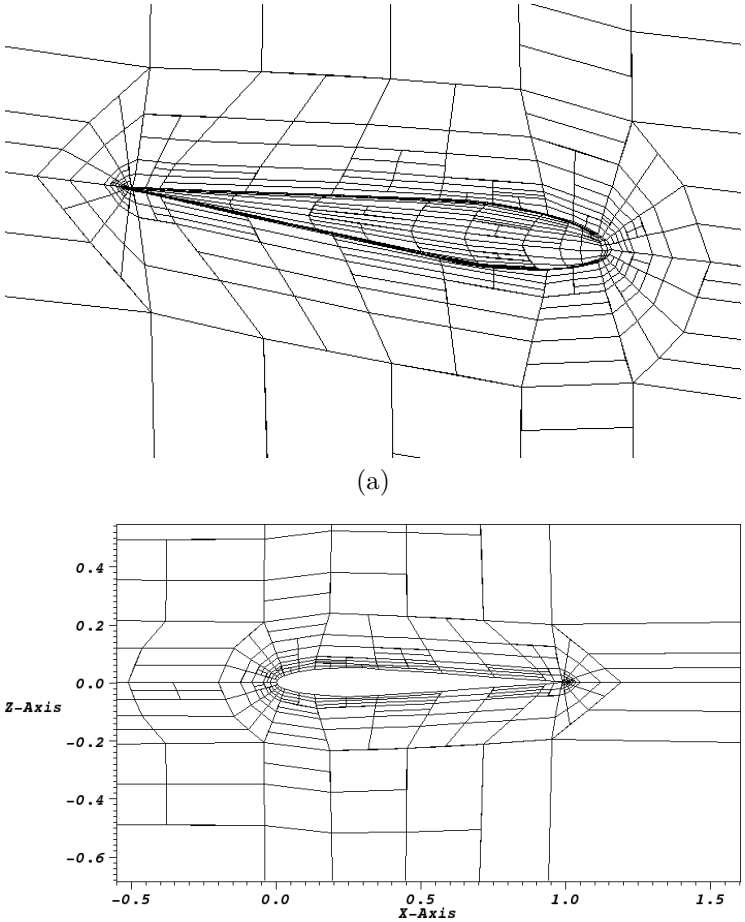


FIGURE 7. Example 2: Comparison between adaptive isotropic and anisotropic mesh refinement.

cf. Figure 6.

This geometry is considered at laminar conditions with inflow Mach number equal to 0.5, at an angle of attack $\alpha = 1°$, and Reynolds number $\mathrm{Re} = 5000$ with adiabatic no-slip wall boundary condition imposed. Here, we suppose that the aim of the computation is to calculate the lift coefficient C_l; i.e.,

$$J(\cdot) \equiv J_{C_l}(\cdot).$$



(a)

(b)

FIGURE 8. Example 2. Anisotropic mesh after 3 adaptive refinements, with 2314 elements: (a) Boundary mesh; (b) Symmetry plane.

In this example, the initial starting mesh is taken to be an unstructured hexahedral mesh with 992 elements, cf. Figure 6. In Figure 7 we plot the error in the computed target functional $J_{C_1}(\cdot)$ using both an isotropic (only) mesh refinement algorithm, together with the anisotropic refinement strategy outlined in Section 5. From Figure 7, we again observe the superiority of employing the anisotropic mesh refinement algorithm in comparison with standard isotropic subdivision of the elements. Indeed, the error $|J_{C_1}(\mathbf{u}) - J_{C_1}(\mathbf{u}_h)|$ computed on the series of anisotropically refined meshes designed using Algorithm 5.1 is always less than the corresponding quantity computed on the isotropic grids. Indeed, on the final mesh the true error between $J_{C_1}(\mathbf{u})$ and $J_{C_1}(\mathbf{u}_h)$ using anisotropic mesh refinement is over an order of magnitude smaller than the corresponding quantity when isotropic h–refinement

# Ele	# Dof	$J_{C_{\rm dp}}(\mathbf{u}) - J_{C_{\rm dp}}(\mathbf{u}_h)$	$\sum_{\kappa\in\mathcal{T}_h}\bar{\eta}_\kappa$	θ_1	$\sum_{\kappa\in\mathcal{T}_h}\lvert\bar{\eta}_\kappa\rvert$	θ_2
448	7168	-0.4844E-02	-0.4411E-02	0.91	0.4453E-02	0.92
562	10252	-0.1197E-02	-0.1111E-02	0.93	0.1126E-02	0.94
685	14912	-0.5029E-03	-0.4631E-03	0.92	0.4707E-03	0.94
784	19360	-0.3923E-03	-0.3685E-03	0.94	0.3749E-03	0.96
838	23928	-0.1541E-03	-0.1433E-03	0.93	0.1500E-03	0.97
970	31780	-0.7443E-04	-0.6990E-04	0.94	0.7720E-04	1.04
1018	38132	-0.3061E-04	-0.2893E-04	0.95	0.3295E-04	1.08
1045	45616	-0.3010E-04	-0.2770E-04	0.92	0.3009E-04	1.00
1120	56684	-0.7940E-05	-0.7772E-05	0.98	0.9242E-05	1.16
1201	73200	-0.2481E-05	-0.2341E-05	0.94	0.3868E-05	1.56

TABLE 1. Example 3: hp–Refinement algorithm based on an initial structured quadrilateral mesh.

is employed alone. The mesh generated after 3 anisotropic adaptive mesh refinements is shown in Figures 8(a) & (b). Here, we again observe significant anisotropic refinement of the viscous boundary layer.

6. hp–Adaptivity on isotropically refined meshes

In this section we now consider the case when both the underlying finite element mesh $\mathcal{T}_h$ and the polynomial distribution are isotropic; thereby, $\mathbf{u}_h \in \mathbf{V}_{h,\mathbf{p}_{\rm iso}}$. The extension to general anisotropic finite element spaces will be considered in the following section. In this setting, once an element has been selected for refinement/derefinement the key step in the design of such an (isotropic) hp–adaptive algorithm is the local decision taken on each element κ in the computational mesh as to which refinement strategy (i.e., h-refinement *via* local mesh subdivision or p-refinement by increasing the degree of the local polynomial approximation) should be employed on κ in order to obtain the greatest reduction in the error per unit cost. To this end, we employ the technique for assessing local smoothness developed in the article [30], which is based on monitoring the decay rate of the sequence of coefficients in the Legendre series expansion of a square–integrable function. The extension of this analyticity estimation procedure to higher–dimensions is based on the application of these techniques in each coordinate direction on a reference element, assuming that a quadrilateral/hexahedral finite element mesh has been employed. For the case of triangular and tetrahedral meshes, we refer to [7].

6.1. Example 3: Inviscid flow around a NACA0012 airfoil. In this section we consider the performance of the goal–oriented hp–refinement algorithm outlined above for the inviscid compressible flow around a NACA0012 airfoil with inflow Mach number equal to 0.5, at an angle of attack $\alpha = 2^\circ$. Here, we suppose that the aim of the computation is to calculate the pressure induced drag coefficient $C_{\rm dp}$; i.e., $J(\cdot) \equiv J_{C_{\rm dp}}(\cdot)$. In Tables 1 & 2 we show the performance of the proposed adaptive finite element algorithm employing hp–refinement based on exploiting a structured and unstructured (hybrid) starting mesh, respectively. In each case, we show the number of elements and degrees of freedom (Dof) in $\mathbf{V}_{h,\mathbf{p}_{\rm iso}}$, the true error in the functional $J_{C_{\rm dp}}(\mathbf{u}) - J_{C_{\rm dp}}(\mathbf{u}_h)$, the computed error representation formula $\sum_{\kappa\in\mathcal{T}_h}\bar{\eta}_\kappa$, the approximate *a posteriori* error bound $\sum_{\kappa\in\mathcal{T}_h}\lvert\bar{\eta}_\kappa\rvert$,

# Ele	# Dof	$J_{C_{\rm dp}}(\mathbf{u}) - J_{C_{\rm dp}}(\mathbf{u}_h)$	$\sum_{\kappa\in\mathcal{T}_h}\bar{\eta}_\kappa$	θ_1	$\sum_{\kappa\in\mathcal{T}_h}\lvert\bar{\eta}_\kappa\rvert$	θ_2
365	5816	-0.1570E-01	-0.1276E-01	0.81	0.1292E-01	0.82
476	8612	-0.4385E-02	-0.3488E-02	0.80	0.3522E-02	0.80
530	11540	-0.8699E-03	-0.7229E-03	0.83	0.7335E-03	0.84
593	14556	-0.2288E-03	-0.2052E-03	0.90	0.2174E-03	0.95
650	18756	-0.6131E-04	-0.5476E-04	0.90	0.5862E-04	0.96
728	24456	-0.2285E-04	-0.2043E-04	0.89	0.2254E-04	0.99
809	30104	-0.8102E-05	-0.7065E-05	0.87	0.9337E-05	1.15
839	36188	-0.3086E-05	-0.2655E-05	0.86	0.4745E-05	1.54
881	45428	-0.1620E-05	-0.1456E-05	0.90	0.3153E-05	1.95
923	55592	-0.4111E-06	-0.4111E-06	1.00	0.1690E-05	4.11

TABLE 2. Example 3: *hp*–Refinement algorithm based on an initial unstructured hybrid mesh.

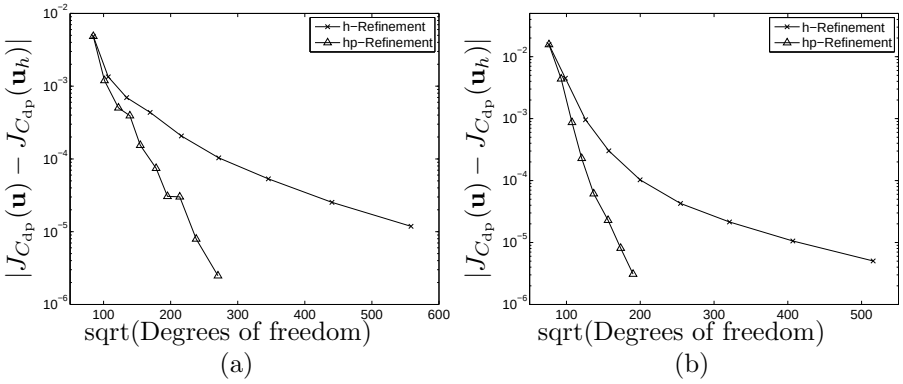
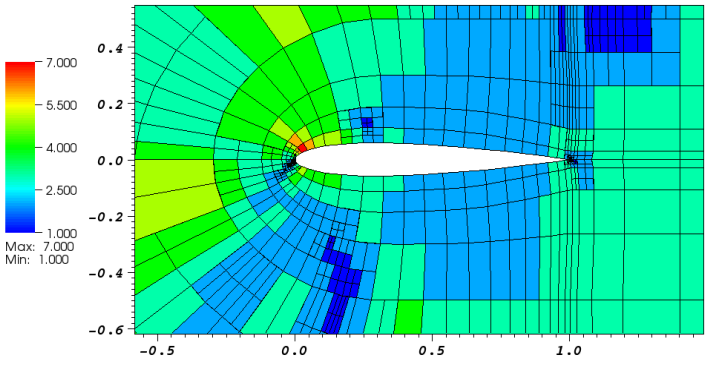


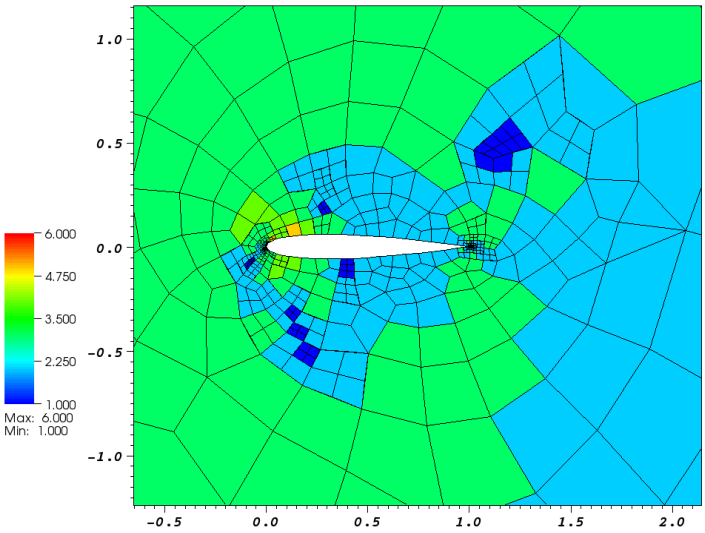
FIGURE 9. Example 3: Comparison between adaptive *h*– and *hp*–mesh refinement. (a) Structured initial mesh; (b) Unstructured initial mesh.

and their respective effectivity indices $\theta_1 = \sum_{\kappa\in\mathcal{T}_h}\bar{\eta}_\kappa/(J_{C_{\rm dp}}(\mathbf{u}) - J_{C_{\rm dp}}(\mathbf{u}_h))$ and $\theta_2 = \sum_{\kappa\in\mathcal{T}_h}|\bar{\eta}_\kappa|/|J_{C_{\rm dp}}(\mathbf{u}) - J_{C_{\rm dp}}(\mathbf{u}_h)|$. Here, we see that the quality of the computed error representation formula is extremely good, with $\theta_1 \approx 1$ even on very coarse meshes.

In Figure 9 we plot the error in the computed target functional $J_{C_{\rm dp}}(\cdot)$, using both *h*– and *hp*–refinement against the square–root of the number of degrees of freedom on a linear–log scale in the case of both a structured and unstructured initial mesh. In both cases, we see that after the initial transient, the error in the computed functional using *hp*–refinement becomes (on average) a straight line, thereby indicating exponential convergence of $J_{C_{\rm dp}}(\mathbf{u}_h)$ to $J_{C_{\rm dp}}(\mathbf{u})$. Figure 9 also demonstrates the superiority of the adaptive *hp*–refinement strategy over the standard adaptive *h*–refinement algorithm. In each case, on the final mesh the true error between $J_{C_{\rm dp}}(\mathbf{u})$ and $J_{C_{\rm dp}}(\mathbf{u}_h)$ using *hp*–refinement is almost 2 orders of magnitude smaller than the corresponding quantity when *h*–refinement is employed alone. Finally, in Figure 10 we show the *hp*–mesh distributions based on employing a structured and unstructured initial mesh after 9 and 7 adaptive refinement steps, respectively.



(a)



(b)

FIGURE 10. Example 3: hp–Mesh distribution. (a) Structured initial mesh after 9 adaptive refinements; (b) Unstructured initial mesh after 7 adaptive refinements.

6.2. Example 1 (Revisited): Laminar flow around a NACA0012 airfoil. Secondly, we again consider Example 1; i.e., laminar compressible flow around a NACA0012 airfoil with inflow Mach number equal to 0.5, at an angle of attack $\alpha = 2°$, and Reynolds number Re $= 5000$ with adiabatic no-slip wall boundary condition imposed on the airfoil geometry. As before, we suppose that the aim of the computation is to calculate the drag coefficient $C_{\rm d}$.

In Figure 11 we plot the error in the computed target functional $J_{C_{\rm d}}(\cdot)$, using both h– and hp–refinement against the square–root of the number of degrees of freedom on a linear–log scale in the case of both a structured and unstructured initial mesh. As before, in both cases, we see that after the initial transient, the error in the computed functional using hp–refinement becomes (on average) a straight

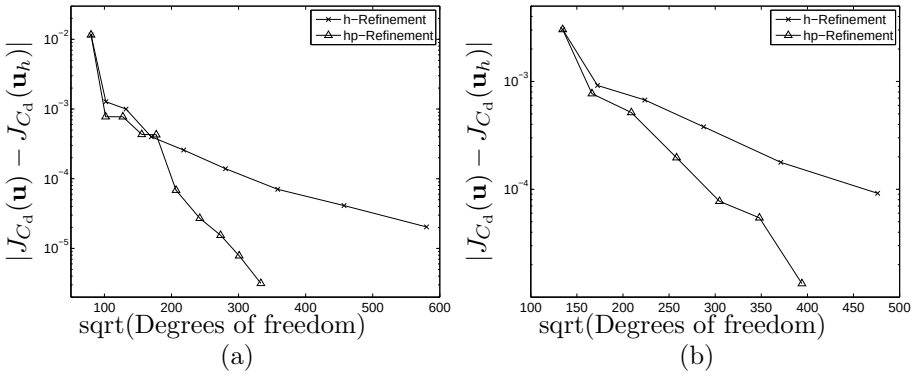


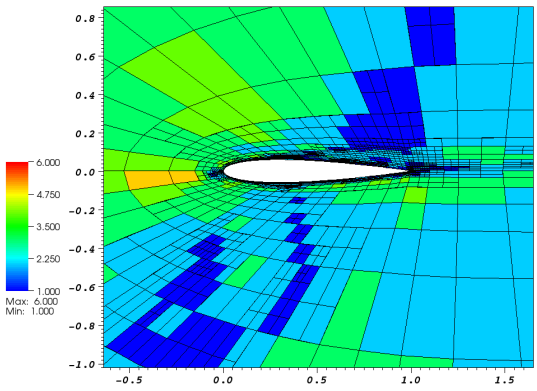
FIGURE 11. Example 1 (revisited): Comparison between adaptive h– and hp–mesh refinement. (a) Structured initial mesh; (b) Unstructured initial mesh.

line, thereby indicating exponential convergence of $J_{C_d}(\mathbf{u}_h)$ to $J_{C_d}(\mathbf{u})$. Figure 11 also demonstrates the superiority of the adaptive hp–refinement strategy over the standard adaptive h–refinement algorithm. In each case, on the final mesh the true error between $J_{C_d}(\mathbf{u})$ and $J_{C_d}(\mathbf{u}_h)$ using hp–refinement is over an order of magnitude smaller than the corresponding quantity when h–refinement is employed alone.

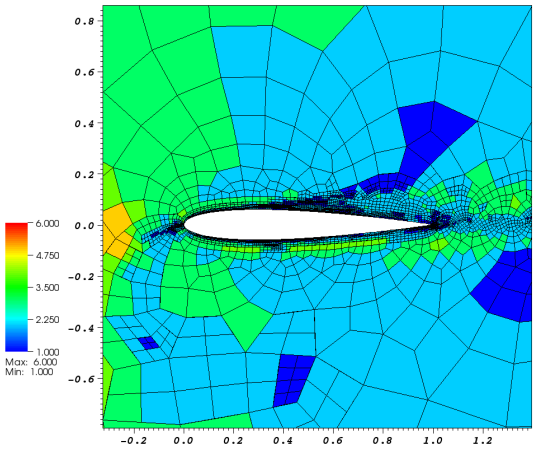
In Figure 12 we show the hp–mesh distributions based on employing a structured and unstructured initial mesh after 8 and 7 adaptive refinement steps, respectively. In each case we observe that some h–refinement has been undertaken in the vicinity of the boundary layers as we would expect. However, once the h–mesh has adequately captured the structure of the primal and adjoint solutions, the hp–adaptive algorithm subsequently performs p–refinement.

7. Anisotropic hp–mesh adaptation

Finally, in this section we consider the general case of automatically generating anisotropically refined computational meshes, together with an anisotropic polynomial degree distribution. With this in mind, once an element has been selected for refinement/derefinement a decision is first made whether to carry out an h-refinement/derefinement or p-enrichment/derefinement based on the technique outlined in Section 6, whereby the analyticity of the solutions $\mathbf{u}$ and $\mathbf{z}$ is assessed by studying the decay rates of their underlying Legendre coefficients. Once the h– and p–refinement flags have been determined on the basis of the above strategy, a decision regarding the type of refinement to be undertaken — isotropic or anisotropic — must be made. Motivated by the work in Section 5, we employ a competitive refinement technique, whereby the "optimal" refinement is selected from a series of trial refinements. In the h–version setting, we again exploit the algorithm outlined in Section 5. For the case when an element has been selected for polynomial enrichment we consider the p–version counterpart of Algorithm 5.1 and solve local problems based on increasing the polynomial degrees anisotropically in one direction at a time by one degree, or isotropically by one degree; see [13] for details.



(a)



(b)

FIGURE 12. Example 1 (revisited): hp–Mesh distribution. (a) Structured initial mesh after 8 adaptive refinements; (b) Unstructured initial mesh after 7 adaptive refinements.

7.1. Example 1 (Revisited): Laminar flow around a NACA0012 airfoil. In this section we again consider the test case outlined in Example 1 and again suppose that the aim of the computation is to calculate the drag coefficient $C_{\rm d}$, cf. Section 5.1.1. In Figure 13 we plot the error in the computed target functional $J_{C_{\rm d}}(\cdot)$, using a variety of h–/hp–adaptive algorithms against the square–root of the number of degrees of freedom on a linear–log scale in the case when an unstructured initial mesh is employed. In particular, here we consider the performance of the following adaptive mesh refinement strategies: isotropic h–refinement, anisotropic h–refinement, isotropic hp-refinement, anisotropic h–/isotropic p–refinement, and anisotropic hp–refinement. Here, we clearly observe that as the flexibility of the underlying adaptive strategy is increased, thereby allowing for greater flexibility in the construction of the finite element space $V_{h,\mathbf{p}}$, the error in the computed target functional of interest is improved in the sense that the error in the computed

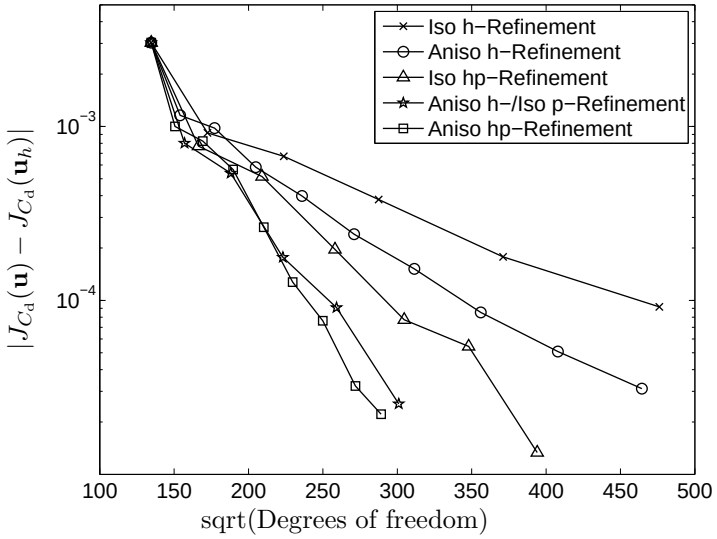


FIGURE 13. Example 1 (revisited): Comparison between different adaptive refinement strategies.

value of $J_{C_d}(\cdot)$ is decreased for a fixed number of degrees of freedom. However, we point out that in the initial stages of refinement, all of the refinement algorithms perform in a similar manner. Indeed, it is not until the structure of the underlying analytical solution is resolved that we observe the benefits of increasing the complexity of the adaptive refinement strategy. Finally, we point out that the latter three refinement strategies incorporating p–refinement all lead to exponential convergence of $J_{C_d}(\mathbf{u}_h)$ to $J_{C_d}(\mathbf{u})$. Figures 14(a) & (b) show the resultant hp–mesh distribution when employing anisotropic hp–refinement after 5 adaptive steps; here, Figures 14(a) & (b) show the (approximate) polynomial degrees employed in the x– and y–directions, respectively. We observe that anisotropic h–refinement has been employed in order to resolve the boundary layer and anisotropic p-refinement has been utilized further inside the computational domain. In particular, we notice that the polynomial degrees have been increased to a higher level in the orthogonal direction to the curved geometry, as we would expect.

Acknowledgement

The authors acknowledge the support of the EU under the ADIGMA project.

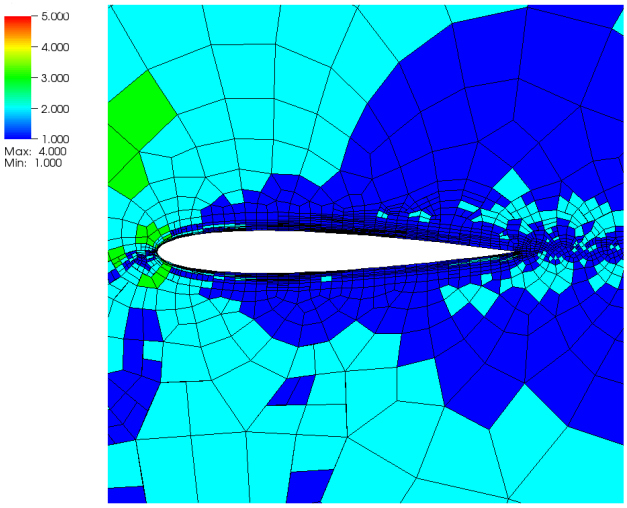
References

[1] F. Bassi and S. Rebay. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations. *J. Comp. Phys.*, 131:267–279, 1997.

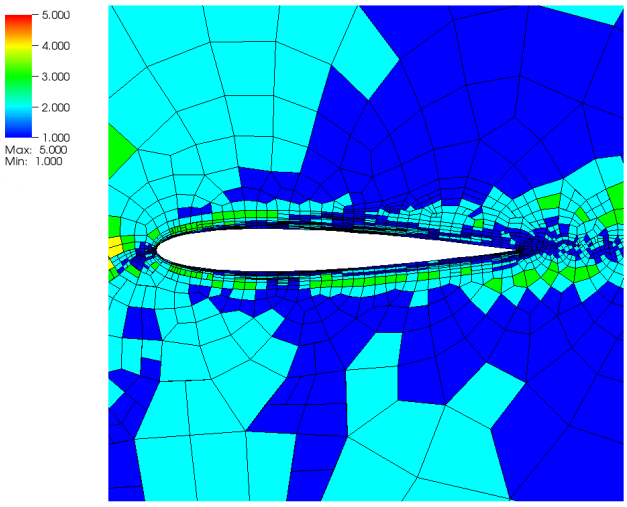
[2] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2d Euler equations. *J. Comp. Phys.*, 138:251–285, 1997.

[3] C. Baumann and J. Oden. A discontinuous *hp* finite element method for the Euler and Navier-Stokes equations. *Int. J. Numer. Methods Fluids*, 31:79–95, 1999.

[4] C. Baumann and J. Oden. An adaptive-order discontinuous Galerkin method for the solution of the Euler equations of gas dynamics. *Int. J. Numer. Methods Engrg.*, 47:61–73, 2000.



(a)



(b)

FIGURE 14. Example 1 (revisited). Mesh distribution after 5 adaptive anisotropic hp–refinements, with 2200 elements and 52744 degrees of freedom: (a) h–/p_x–mesh distribution; (b) h–/p_y–mesh distribution.

[5] R. Becker and R. Rannacher. An optimal control approach to a-posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*, pages 1–102. Cambridge University Press, 2001.

[6] V. Dolejší. On the discontinuous Galerkin method for the numerical solution of the Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 45:1083–1106, 2004.

[7] T. Eibner and J. M. Melenk. An adaptive strategy for hp-FEM based on testing for analyticity. *Comput. Mech.*, 39(5):575–595, 2007.

[8] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for differential equations. In A. Iserles, editor, *Acta Numerica*, pages 105–158. Cambridge University Press, 1995.

[9] M. Feistauer, J. Felcman, and I. Straškraba. *Mathematical and Computational Methods for Compressible Flow*. Clarendon Press, Oxford, 2003.

[10] K. Fidkowski and D. Darmofal. A triangular cut-cell adaptive method for high-order discretizations of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 225:1653–1672, 2007.

[11] K. Fidkowski, T. Oliver, J. Lu, and D. Darmofal. *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier-Stokes equations. *J. Comput. Phys.*, 207(1):92–113, July 2005.

[12] E. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods for advection–diffusion–reaction problems on anisotropically refined meshes. *SIAM J. Sci. Comput.*, 30(1):246–271, 2007.

[13] E. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods on *hp*–anisotropic meshes II: A posteriori error analysis and adaptivity. *Appl. Numer. Math.*, 59(9):2179–2194, 2009.

[14] S. Giani and P. Houston. High-order *hp*–adaptive discontinuous Galerkin finite element methods for compressible fluid flows. In N. Kroll, H. Bieler, H. Deconinck, V. Couallier, H. van der Ven, and K. Sorensen, editors, *ADIGMA – A European Initiative on the Development of Adaptive Higher–Order Variational Methods for Aerospace Applications*, volume 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, pages 399–411. Springer, 2010.

[15] E. J. C. Hall. *Anisotropic Adaptive Refinement For Discontinuous Galerkin Methods*. PhD thesis, Department of Mathematics, University of Leicester, 2007.

[16] K. Harriman, P. Houston, B. Senior, and E. Süli. *hp*–Version discontinuous Galerkin methods with interior penalty for partial differential equations with nonnegative characteristic form. In C.-W. Shu, T. Tang, and S.-Y. Cheng, editors, *Recent Advances in Scientific Computing and Partial Differential Equations. Contemporary Mathematics Vol. 330*, pages 89–119. AMS, 2003.

[17] R. Hartmann. Adjoint consistency analysis of discontinuous Galerkin discretizations. *SIAM J. Numer. Anal.*, 45(6):2671–2696, 2007.

[18] R. Hartmann. Numerical analysis of higher order discontinuous Galerkin finite element methods. In H. Deconinck, editor, *VKI LS 2008-08: CFD - ADIGMA course on very high order discretization methods, Oct. 13-17, 2008*. Von Karman Institute for Fluid Dynamics, Rhode Saint Genèse, Belgium, 2008.

[19] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for nonlinear hyperbolic conservation laws. *SIAM J. Sci. Comput.*, 24:979–1004, 2002.

[20] R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *J. Comput. Phys.*, 183(2):508–532, 2002.

[21] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible Navier–Stokes equations I: Method formulation. *Int. J. Num. Anal. Model.*, 3(1):1–20, 2006.

[22] R. Hartmann and P. Houston. Symmetric interior penalty DG methods for the compressible Navier–Stokes equations II: Goal–oriented a posteriori error estimation. *Int. J. Num. Anal. Model.*, 3(2):141–162, 2006.

[23] R. Hartmann and P. Houston. An optimal order interior penalty discontinuous Galerkin discretization of the compressible Navier–Stokes equations. *J. Comput. Phys.*, 227(22):9670–9685, 2008.

[24] R. Hartmann and P. Houston. Error estimation and adaptive mesh refinement for aerodynamic flows. In H. Deconinck, editor, *VKI LS 2010-01: 36th CFD/ADIGMA Course on hp-Adaptive and hp-Multigrid Methods, Oct. 26-30, 2009*. Von Karman Institute for Fluid Dynamics, Rhode Saint Genèse, Belgium, 2010.

[25] R. Hartmann and T. Leicht. Error estimation and anisotropic mesh refinement for 3d aerodynamic flow simulations. *J. Comput. Phys.*, 229(19), 2010.

[26] P. Houston, J. Mackenzie, E. Süli, and G. Warnecke. A posteriori error analysis for numerical approximations of Friedrichs systems. *Numer. Math.*, 82:433–470, 1999.

[27] P. Houston and E. Süli. Local mesh design for the numerical solution of hyperbolic problems. In M. Baines, editor, *Numerical methods for Fluid Dynamics VI, ICFD*, pages 17–30, 1998.

[28] P. Houston and E. Süli. hp–Adaptive discontinuous Galerkin finite element methods for hyperbolic problems. *SIAM J. Sci. Comput.*, 23:1225–1251, 2001.

[29] P. Houston and E. Süli. Adaptive finite element approximation of hyperbolic problems. In T. Barth and H. Deconinck, editors, *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics. Lect. Notes Comput. Sci. Engrg.*, volume 25, pages 269–344. Springer, 2002.

[30] P. Houston and E. Süli. A note on the design of hp–adaptive finite element methods for elliptic partial differential equations. *Comput. Methods Appl. Mech. Engrg.*, 194(2-5):229–243, 2005.

[31] N. Kroll. ADGIMA – A European project on the development of adaptive higher-order variational methods for aerospace applications. 47th AIAA Aerospace Sciences Meeting, 2009. AIAA 2009-176.

[32] S. Prudhomme, F. Pascal, J. Oden, and A. Romkes. Review of *a priori* error estimation for discontinuous Galerkin methods. TICAM Report 00-27, University of Texas, 2000.

[33] J. van der Vegt and H. van der Ven. Space-time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows, I. General formulation. *J. Comp. Phys.*, 182:546–585, 2002.

[34] K. G. van der Zee. An $H^1(P^h)$-coercive discontinuous Galerkin formulation for the Poisson problem: 1-d analysis. Master's thesis, TU Delft, 2004.

School of Mathematical Sciences, University of Nottingham, University Park, Nottingham NG7 2RD, UK.

E-mail: Stefano.Giani@nottingham.ac.uk

School of Mathematical Sciences, University of Nottingham, University Park, Nottingham NG7 2RD, UK.

E-mail: Paul.Houston@nottingham.ac.uk