High–Order *hp*–Adaptive Discontinuous Galerkin Finite Element Methods for Compressible Fluid Flows

Stefano Giani and Paul Houston

Abstract This article is concerned with the construction of general isotropic and anisotropic adaptive strategies, as well as *hp*–mesh refinement techniques, in combination with dual–weighted–residual *a posteriori* error indicators for the discontinuous Galerkin finite element discretization of compressible fluid flow problems.

1 Introduction

The development of Discontinuous Galerkin (DG) methods for the numerical approximation of the 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, 5, 8, 9, 10, 14], 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 this 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.

Stefano Giani
School of Mathematical Sciences, University of Nottingham, University Park, Nottingham NG7 2RD, UK. e-mail: Stefano.Giani@nottingham.ac.uk

Paul Houston
School of Mathematical Sciences, University of Nottingham, University Park, Nottingham NG7 2RD, UK. e-mail: Paul.Houston@nottingham.ac.uk

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 [12]). 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. Selective error estimates of this kind can be obtained by the optimal control technique proposed in [4] and [3] 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 article develops duality-based *a posteriori* error estimation of DG finite element methods, together with the application of these computable bounds within automatic adaptive finite element algorithms. Here, a variety of isotropic and anisotropic adaptive strategies, as well as hp–mesh refinement will be investigated.

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:

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

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 $\mathscr{F}^c(\mathbf{u}) = \left(\mathbf{f}_1^c(\mathbf{u}), \mathbf{f}_2^c(\mathbf{u}), \mathbf{f}_3^c(\mathbf{u})\right)^\top$ is given by $\mathbf{f}_1^c(\mathbf{u}) = (\rho v_1, \rho v_1^2 + p, \rho v_1 v_2, \rho v_1 v_3, \rho H v_1)^\top$, $\mathbf{f}_2^c(\mathbf{u}) = (\rho v_2, \rho v_2 v_1, \rho v_2^2 + p, \rho v_2 v_3, \rho H v_2)^\top$, and $\mathbf{f}_3^c(\mathbf{u}) = (\rho v_3, \rho v_3 v_1, \rho v_3 v_2, \rho v_3^2 + p, \rho H v_3)^\top$. Furthermore, $\mathbf{f}_k^v(\mathbf{u}, \nabla \mathbf{u}) = (0, \tau_{1k}, \tau_{2k}, \tau_{3k}, \tau_{kl} v_l + \mathscr{K} T_{x_k})^\top$, $k = 1, 2, 3$. 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, $\mathscr{K}$ is the thermal conductivity coefficient and H is the total enthalpy given by $H = E + \frac{p}{\rho} = e + \frac{1}{2}\mathbf{v}^2 + \frac{p}{\rho}$, where e is the specific static internal energy, and the pressure is determined by the equation of state of an ideal gas $p = (\gamma - 1)\rho e$, 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 $\tau = \mu\left(\nabla\mathbf{v} + (\nabla\mathbf{v})^\top - \frac{2}{3}(\nabla\cdot\mathbf{v})I\right)$, where μ is the dynamic viscosity coefficient; the temperature T is given by $\mathscr{K}T = \frac{\mu\gamma}{Pr}\left(E - \frac{1}{2}\mathbf{v}^2\right)$, 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:

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

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 $\mathbf{f}_k^v(\mathbf{u},\nabla\mathbf{u}) = G_{kl}(\mathbf{u})\partial\mathbf{u}/\partial x_l$, $k = 1,2,3$.

3 DG Discretization

In this section we introduce the adjoint-consistent interior penalty DG discretization of the compressible Navier–Stokes equations (1), cf. [11] 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 $\mathscr{T}_h = \{\kappa\}$ consisting of tensor-product (quadrilaterals, $d = 2$, and hexahedra, $d = 3$) open element domains κ. For each $\kappa \in \mathscr{T}_h$, we denote by $\mathbf{n}_\kappa$ the unit outward normal vector to the boundary $\partial\kappa$. We assume that each $\kappa \in \mathscr{T}_h$ is an image of a fixed reference element $\hat{\kappa}$, that is, $\kappa = \sigma_\kappa(\hat{\kappa})$ for all $\kappa \in \mathscr{T}_h$, 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 $\mathscr{Q}_\mathbf{p}$ with respect to the anisotropic polynomial degree vector $\mathbf{p} := \{p_i\}_{i=1,\ldots,d}$ as follows: $\mathscr{Q}_\mathbf{p} := \mathrm{span}\{\Pi_{i=1}^d \hat{x}_i^j : 0 \le j \le p_i\}$. With this notation, we introduce the following (anisotropic) finite element space.

Definition 1. Let $\mathbf{p} = (\mathbf{p}_\kappa : \kappa \in \mathscr{T}_h)$ be the composite polynomial degree vector of the elements in a given finite element mesh $\mathscr{T}_h$. We define the finite element space with respect to Ω, $\mathscr{T}_h$, and $\mathbf{p}$ by $\mathbf{V}_{h,\mathbf{p}} = \{u \in L_2(\Omega) : u|_\kappa \circ \sigma_\kappa \in [\mathscr{Q}_{\mathbf{p}_\kappa}]^{d+2}\}$.

In the case when the elemental polynomial degree vector $\mathbf{p}_\kappa = \{p_{\kappa,i}\}_{i=1,\ldots,d}$, $\kappa \in \mathscr{T}_h$, 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 $\mathscr{T}_h$, then we write $\mathbf{V}_{h,\mathbf{p}_{\mathrm{iso}}}$ in lieu of $\mathbf{V}_{h,\mathbf{p}}$, where $\mathbf{p}_{\mathrm{iso}} = (p_\kappa : \kappa \in \mathscr{T}_h)$. Additionally, in the case when the polynomial degree is both isotropic and uniformly distributed over the mesh $\mathscr{T}_h$, i.e., when $p_\kappa = p$ for all κ in $\mathscr{T}_h$, then we simply denote the finite element space by $\mathbf{V}_{h,p}$.

An *interior face* of $\mathscr{T}_h$ 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 $\mathscr{T}_h$, not necessarily matching. A *boundary face* of $\mathscr{T}_h$ is defined as the (non-empty) $(d-1)$–dimensional interior of $\partial\kappa \cap \Gamma$, where κ is a boundary element of $\mathscr{T}_h$. We denote by $\Gamma_\mathscr{I}$ the union of all interior faces of $\mathscr{T}_h$. Let κ^+ and κ^- be two adjacent elements of $\mathscr{T}_h$, 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.

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

$$\begin{aligned}\mathscr{N}(\mathbf{u}_h,\mathbf{v}) \;\equiv & -\int_\Omega \mathscr{F}^c(\mathbf{u}_h) : \nabla_h \mathbf{v}\,\mathrm{d}\mathbf{x} + \sum_{\kappa\in\mathscr{T}_h}\int_{\partial\kappa\setminus\Gamma} \mathscr{H}(\mathbf{u}_h^+,\mathbf{u}_h^-,\mathbf{n}^+)\cdot\mathbf{v}^+\,\mathrm{d}s \\ & + \int_\Omega \mathscr{F}^v(\mathbf{u}_h,\nabla_h\mathbf{u}_h) : \nabla_h\mathbf{v}\,\mathrm{d}\mathbf{x} - \int_{\Gamma_\mathscr{I}} \{\{\mathscr{F}^v(\mathbf{u}_h,\nabla_h\mathbf{u}_h)\}\} : \underline{[\![\mathbf{v}]\!]}\,\mathrm{d}s \\ & - \int_{\Gamma_\mathscr{I}} \{\{G^\top(\mathbf{u}_h)\nabla_h\mathbf{v}\}\} : \underline{[\![\mathbf{u}_h]\!]}\,\mathrm{d}s + \int_{\Gamma_\mathscr{I}} \underline{\delta}(\mathbf{u}_h) : \underline{[\![\mathbf{v}]\!]}\,\mathrm{d}s + \mathscr{N}_\Gamma(\mathbf{u}_h,\mathbf{v}) = 0 \quad (2)\end{aligned}$$

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, $\mathscr{H}(\cdot,\cdot,\cdot)$ 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 (2), we first introduce the local (anisotropic) mesh and polynomial functions $\mathtt{h}$ and $\mathtt{p}$, respectively. To this end, the function $\mathtt{h}$ in $L_\infty(\Gamma_\mathscr{I} \cup \Gamma)$ 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 $\mathscr{T}_h$, 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 $L_\infty(\Gamma_\mathscr{I} \cup \Gamma)$ 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. [7].

Finally, we define the boundary terms present in the form $\mathscr{N}_\Gamma(\cdot,\cdot)$ by

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

where $\underline{\delta}_\Gamma(\mathbf{u}_h) = C_{\mathrm{IP}}\frac{\mathtt{p}^2}{\mathtt{h}}G_\Gamma(\mathbf{u}_h^+)\,(\mathbf{u}_h - \mathbf{u}_\Gamma(\mathbf{u}_h))\otimes\mathbf{n}$. Here, the viscous boundary flux $\mathscr{F}_\Gamma^v$ and the corresponding homogeneity tensor G_Γ are defined by

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

Furthermore, on portions of the boundary Γ where adiabatic boundary conditions are imposed, $\mathscr{F}_\Gamma^v$ and G_Γ are modified such that $\mathbf{n}\cdot\nabla T=0$. The convective boundary flux $\mathscr{H}_\Gamma$ is defined by $\mathscr{H}_\Gamma(\mathbf{u}_h^+,\mathbf{u}_\Gamma(\mathbf{u}_h^+),\mathbf{n})=\mathbf{n}\cdot\mathscr{F}^c(\mathbf{u}_\Gamma(\mathbf{u}_h^+))$. Finally, the boundary function $\mathbf{u}_\Gamma(\mathbf{u})$ is given according to the type of boundary condition imposed; for details, we refer to [11], for example.

4 *A posteriori* error estimation

In this section we consider 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.

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 $\mathscr{N}(\cdot,\mathbf{v})$ by

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

Here, $\mathscr{N}'[\mathbf{w}](\cdot,\mathbf{v})$ denotes the Fréchet derivative of $\mathbf{u}\mapsto\mathscr{N}(\mathbf{u},\mathbf{v})$, 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

$$\mathscr{M}(\mathbf{u},\mathbf{u}_h;\mathbf{w},\mathbf{z})=\bar{J}(\mathbf{u},\mathbf{u}_h;\mathbf{w})\quad\forall\mathbf{w}\in\mathbf{V}. \tag{3}$$

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

$$J(\mathbf{u})-J(\mathbf{u}_h)=\mathscr{R}(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)\equiv\sum_{\kappa\in\mathscr{T}_h}\eta_\kappa, \tag{4}$$

where $\mathscr{R}(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)=-\mathscr{N}(\mathbf{u}_h,\mathbf{z}-\mathbf{z}_h)$ 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 [8, 10] for details.

We note that the error representation formula (4) depends on the unknown analytical solution $\mathbf{z}$ to the adjoint problem (3) 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 $\mathscr{M}(\mathbf{u},\mathbf{u}_h;\cdot,\cdot)$ 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 $\bar{\mathbf{z}}_h\in\bar{\mathbf{V}}_{h,\mathbf{p}}$, where $\bar{\mathbf{V}}_{h,\mathbf{p}}$ is an *ad-*

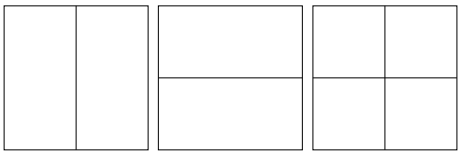


Fig. 1 Cartesian refinement in 2D: (a) & (b) Anisotropic refinement; (c) Isotropic refinement.

joint finite element space from which the approximate adjoint solution $\overline{\mathbf{z}}_h$ is sought. For the purposes of this article, we set $\overline{\mathbf{V}}_{h,\mathbf{p}} = \mathbf{V}_{h,\mathbf{p}_d}$, where $\mathbf{p}_d = \mathbf{p} + \mathbf{1}$.

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 $\mathscr{T}_h$, assuming that the underlying polynomial degree distribution is both uniform and fixed, i.e., when $\mathbf{u}_h \in \mathbf{V}_{h,p}$. To this end, elements are marked for refinement/derefinement according to the size of the (approximate) error indicators $|\overline{\eta}_\kappa|$, based on employing a fixed fraction strategy, for example. Here, $\overline{\eta}_\kappa$ is defined analogously to η_κ in (4) with $\mathbf{z}$ replaced by $\overline{\mathbf{z}}_h$.

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 propose 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.

Algorithm 5.1 *Given an element κ in the computational mesh $\mathscr{T}_h$ (which has been marked for refinement), we first construct the mesh patches $\mathscr{T}_{h,i}$, $i = 1,2,3$, based on refining κ according to Figures 1(a), (b), & (c), respectively. On each mesh patch, $\mathscr{T}_{h,i}$, $i = 1,2,3$, we compute the approximate error estimators $\mathscr{R}_{\kappa,i}(\mathbf{u}_{h,i}, \overline{\mathbf{z}}_{h,i} - \mathbf{z}_h) = \sum_{\kappa' \in \mathscr{T}_{h,i}} \eta_{\kappa',i}$, for $i = 1,2,3$, respectively. Here, $\mathbf{u}_{h,i}$, $i = 1,2,3$, is the DG approximation computed on the mesh patch $\mathscr{T}_{h,i}$, $i = 1,2,3$, 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, $\overline{\mathbf{z}}_{h,i}$ denotes the DG approximation to $\mathbf{z}$ computed on the local mesh patch $\mathscr{T}_{h,i}$, $i = 1,2,3$, respectively, with polynomials of degree p_d, based on employing suitable boundary conditions on $\partial\kappa \cap \Gamma = \emptyset$ derived*

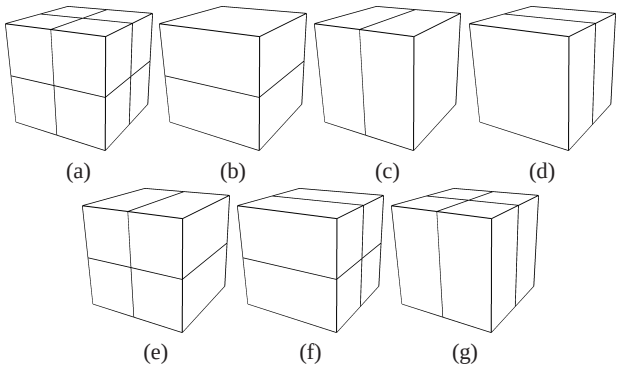


Fig. 2 Cartesian refinement in 3D.

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*

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

where $\#dofs(\kappa)$ *and* $\#dofs(\mathscr{T}_{h,i})$*,* $i=1,2,3$*, denote the number of degrees of freedom associated with* κ *and* $\mathscr{T}_{h,i}$*,* $i=1,2,3$*, respectively, cf. [6].*

The extension of this approach to the case when $\mathscr{T}_h$ 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 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 1(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.

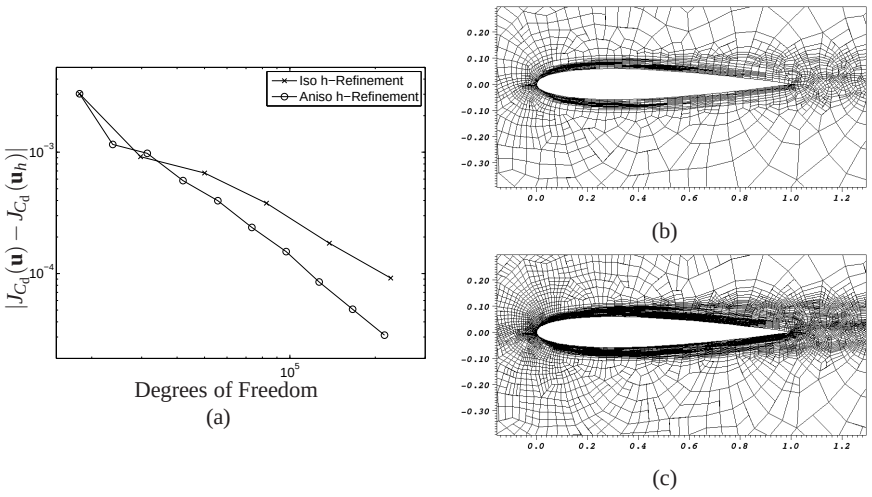


Fig. 3 ADIGMA MTC3 test case: (a) Comparison between adaptive isotropic and anisotropic mesh refinement; Anisotropic mesh after (b) 4 adaptive refinements, with 3485 elements; (c) 8 adaptive refinements, with 10410 elements.

5.1.1 ADIGMA MTC3: Laminar flow around a NACA0012 airfoil

In this example, we consider the subsonic viscous flow around a NACA0012 airfoil. 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. Here, we consider the estimation of the drag coefficient C_d; i.e., the target functional of interest is given by $J(\cdot) \equiv J_{C_\mathrm{d}}(\cdot)$. 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. Furthermore, curved boundaries are approximated by piecewise quadratic polynomials. In Figure 3(a) 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 3(a), 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 computed using isotropic mesh refinement. The meshes generated after 4 and 8 anisotropic adaptive mesh refinements are shown in Figures 3(b) & (c), respectively. Here, we clearly observe significant anisotropic refinement of the viscous boundary layer, as we would expect.

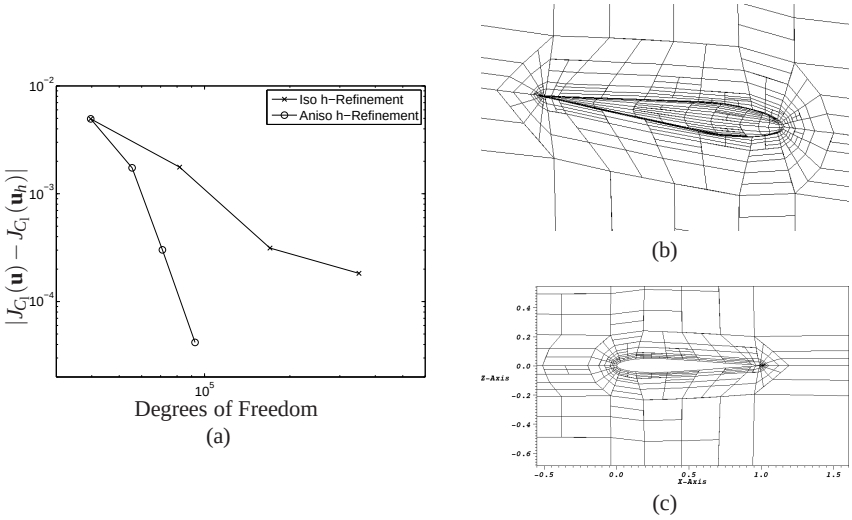


Fig. 4 ADIGMA BTC0 test case (laminar): (a) Comparison between adaptive isotropic and anisotropic mesh refinement; Anisotropic mesh after 3 adaptive refinements, with 2314 elements: (b) Boundary mesh; (c) Symmetry plane.

5.1.2 ADIGMA BTC0: 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. The BTC0 geometry is considered at laminar conditions with inflow Mach number equal to 0.5, at an angle of attack $\alpha = 1^\circ$, 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., $J(\cdot) \equiv J_{C_l}(\cdot)$. In this example, the initial starting mesh is taken to be an unstructured hexahedral mesh with 992 elements. In Figure 4(a) we plot the error in the computed target functional $J_{C_l}(\cdot)$ using both an isotropic (only) mesh refinement algorithm, together with the anisotropic refinement strategy outlined in Section 5. From Figure 4(a), we again observe the superiority of employing the anisotropic mesh refinement algorithm in comparison with standard isotropic subdivision of the elements. Indeed, the error $|J_{C_l}(\mathbf{u}) - J_{C_l}(\mathbf{u}_h)|$ 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 $J_{C_l}(\mathbf{u})$ and $J_{C_l}(\mathbf{u}_h)$ using anisotropic mesh refinement is over an order of magnitude smaller than the corresponding quantity when isotropic *h*–refinement is employed alone. The mesh generated after 3 anisotropic adaptive mesh refinements is shown in Figures 4(b) & (c). Here, we again observe significant anisotropic refinement of the viscous boundary layer.

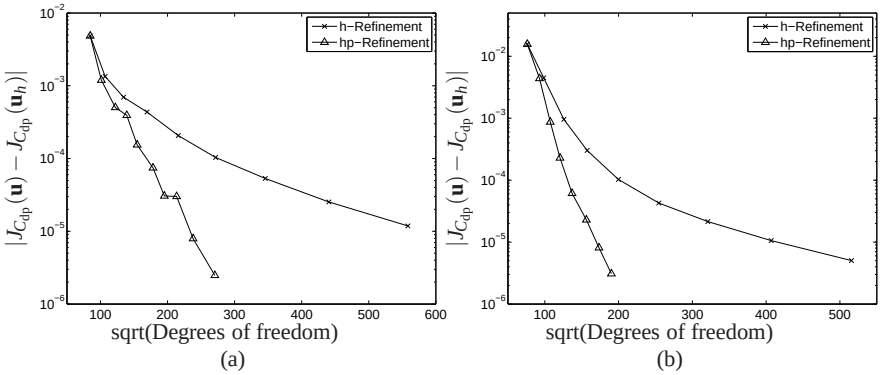


Fig. 5 ADIGMA MTC1 test case: Comparison between adaptive *hp*– and *h*–mesh refinement. (a) Structured initial mesh; (b) Unstructured initial mesh.

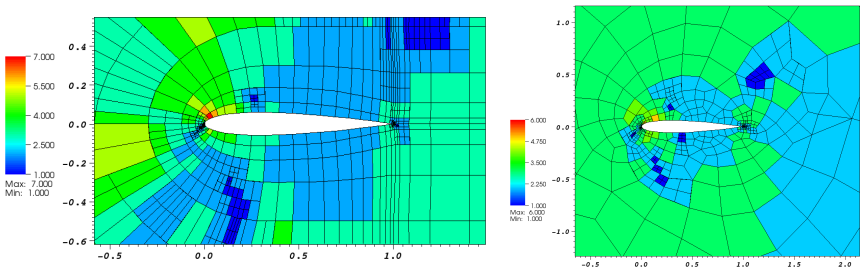
6 *hp*–Adaptivity on isotropically refined meshes

In this section we now consider the case when both the underlying finite element mesh $\mathscr{T}_h$ 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 [13], which is based on monitoring the decay rate of the sequence of coefficients in the Legendre series expansion of a square–integrable function.

6.1 ADIGMA MTC1: Inviscid flow around a NACA0012 airfoil

In this section we consider the performance of the goal–oriented *hp*–refinement algorithm outlined above for the ADIGMA MTC1 test case: 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 Figure 5 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



(a) (b)

Fig. 6 ADIGMA MTC1 test case: *hp*–Mesh distribution. (a) Structured initial mesh after 9 adaptive refinements; (b) Unstructured initial mesh after 7 adaptive refinements.

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 5 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 6 we show the *hp*–mesh distributions based on employing a structured and unstructured initial mesh after 9 and 7 adaptive refinement steps, respectively.

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 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 [7] for details.

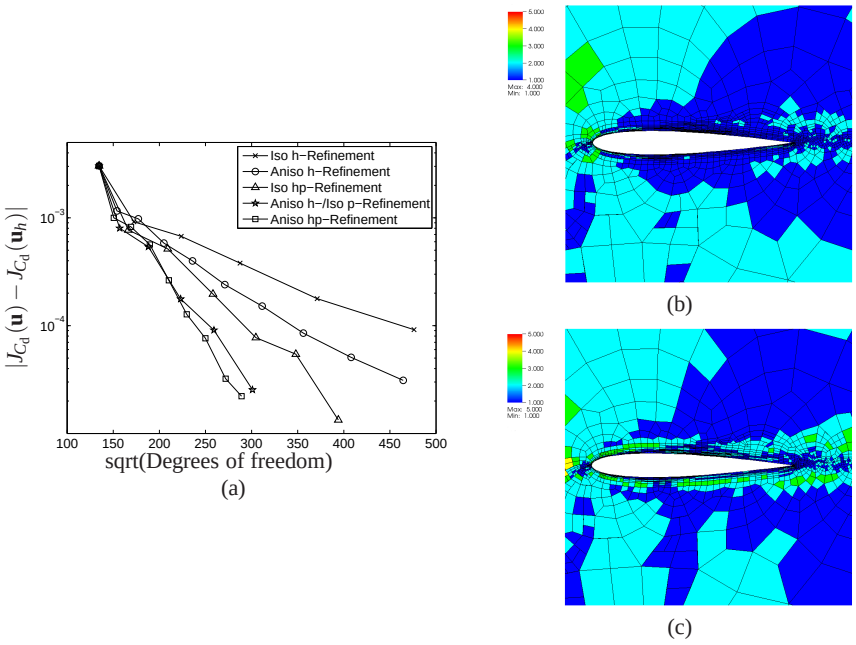


Fig. 7 ADIGMA MTC3 test case: (a) Comparison between different adaptive refinement strategies. Mesh distribution after 5 adaptive anisotropic *hp*–refinements, with 2200 elements and 52744 degrees of freedom: (b) *h*–/p_x–mesh distribution; (c) *h*–/p_y–mesh distribution.

7.1 ADIGMA MTC3: Laminar flow around a NACA0012 airfoil

In this section we again consider the ADIGMA MTC3 test case 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 7(a) 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 value of $J_{C_{\rm d}}(\cdot)$ 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 refine-

ment strategies incorporating *p*–refinement all lead to exponential convergence of $J_{C_d}(\mathbf{u}_h)$ to $J_{C_d}(\mathbf{u})$. Figures 7(b) & (c) show the resultant *hp*–mesh distribution when employing anisotropic *hp*–refinement after 5 adaptive steps; here, Figures 7(b) & (c) 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.

Acknowledgements 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. C. Baumann and J. Oden. A discontinuous *hp* finite element method for the Euler and Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 31:79–95, 1999.
3. R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numerica*, 10:1–102, 2001.
4. 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.
5. K. J. Fidkowski and D. L. Darmofal. A triangular cut-cell adaptive method for high-order discretizations of the compressible Navier-Stokes equations. *J. Comput. Physics*, 225:1653–1672, 2007.
6. 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.
7. 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.
8. R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *J. Comput. Phys.*, 183(2):508–532, 2002.
9. 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.
10. 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.
11. 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.
12. P. Houston, J. Mackenzie, E. Süli, and G. Warnecke. A posteriori error analysis for numerical approximations of Friedrichs systems. *Numerische Mathematik*, 82:433–470, 1999.
13. 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.
14. 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.