INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 5, Number 4, Pages 704–728

A PRE–PROCESSING MOVING MESH METHOD FOR DISCONTINUOUS GALERKIN APPROXIMATIONS OF ADVECTION–DIFFUSION–REACTION PROBLEMS

PAOLA F. ANTONIETTI AND PAUL HOUSTON

Abstract. We propose a pre–processing mesh re-distribution algorithm based upon harmonic maps employed in conjunction with discontinuous Galerkin approximations of advection–diffusion–reaction problems. Extensive two–dimensional numerical experiments with different choices of monitor functions, including monitor functions derived from goal–oriented *a posteriori* error indicators are presented. The examples presented clearly demonstrate the capabilities and the benefits of combining our pre–processing mesh movement algorithm with both uniform, as well as, adaptive isotropic and anisotropic mesh refinement.

Key Words. Discontinuous Galerkin methods, advection–diffusion problems, adaptivity.

1. Introduction

The modeling of the interaction between advective and diffusive processes is of fundamental importance in many areas of applied mathematics. Typically, in applications, advection essentially dominates diffusion, which leads to a 'nearly' hyperbolic set of governing partial differential equations. Moreover, solutions to these equations exhibit localized phenomena, such as propagating 'near–shocks' and sharp transition layers, and their numerical approximation presents a challenging computational task; indeed, it is well documented that many standard numerical methods, developed for diffusion–dominated processes, often behave very poorly when applied to these types of problems. Additionally, the presence of local singularities in the solution may lead to a global deterioration of the numerical approximation. Indeed, when uniform meshes are employed, the computational cost to obtain accurate numerical solutions is typically very high, particularly for three-dimensional problems. Therefore, the development of effective and robust adaptive methods for these types of problems becomes a computational necessity. The successful implementation of adaptive strategies, on the one hand, can increase the accuracy of the numerical approximation and, on the other hand, decrease the computational cost. The adaptive strategies developed within the context of finite element methods, can be broadly classified as follows:

Received by the editors December 11, 2007.
2000 *Mathematics Subject Classification.* 65N30.
The first author acknowledges the financial support of the European Union, under the ADIGMA project. The second author acknowledges the financial support of both the EPSRC, under grant GR/R76615, and the European Union, under the ADIGMA project.

h-method: This involves the automatic refinement or coarsening of the computational mesh based on suitable *a posteriori* error estimates or error indicators;

p-method: This involves the enrichment of the local (elemental) polynomial degree;

hp-method: This combines both local h– and p–refinement based on a local decision taken on each element of the computational mesh as to which refinement strategy (*i.e.*, h–refinement or p–refinement) should be employed on the element in order to obtain the greatest reduction in the error per unit cost. It exploits suitable control techniques which assess the local regularity of the underlying analytical solution; for example, one may determine the analyticity of a function by writing it in terms of a convergent Legendre series expansion, and assessing the rate at which the Legendre coefficients tend to zero, cf. [18, 8];

r-method (moving mesh method): This approach *relocates* (without affecting the mesh topology) the grid points of a mesh, keeping the number of nodes fixed, in such a way that the nodes become concentrated in regions of the computational domain where the analytical solution undergoes rapid variation.

On the one hand, considerable progress has been made on both the *a posteriori* error analysis of finite element methods for a wide range of partial differential equations of practical interest, and the development of reliable and robust automatic h–, p– and hp–strategies (see, for example, [1, 4, 9, 25, 26, 28, 8], and the references therein). On the other hand, the state of development of "optimal" mesh modification strategies which are capable of delivering the greatest reduction in the error for the least amount of computational cost, is far less advanced.

In recent years considerable work has been devoted to the development of r–adaptive finite element algorithms, which, for a fixed polynomial order at least, seek to re-distribute the nodes of a given mesh in an optimal fashion; see, for example, [29, 23, 21, 23, 20, 19, 22], and the references cited therein. The moving mesh method is very well suited for dynamical problems, and indeed problems with moving boundaries, though such approaches may also be employed to optimize a mesh for a stationary PDE by employing a nonlinear iteration similar to that employed in h–adaptive methods. An r–refinement method usually contains two key steps: a mesh selection algorithm and a solution algorithm. In some of the existing r-methods, these two parts are strongly associated with each other, and any change of the underlying partial differential equation will result in the rewriting of large parts of the computational code. The success of a mesh adaptation strategy using a variational approach hinges on choosing an appropriate monitor function (cf. [6], for a study of this aspect of the adaptive mesh generation problem). For example, for linear finite elements, the monitor function is often given in terms of some first–/second–order derivatives of the computed solution.

One major drawback of r–refinement techniques is that they are often very expensive, particularly when the underlying mesh is extremely fine; in this case any iterative approach employed to move the mesh may converge very slowly, if at all. In this article, we aim exploit the numerous advantages of the original r-method based on harmonic maps to develop an optimal pre–processing algorithm to be employed in conjunction with discontinuous Galerkin (DG, for short) approximations

of steady state advection–diffusion–reaction problems. Here, the pre–processing mesh movement is only applied to very coarse initial meshes, which can be done very efficiently; the resulting optimized mesh is then used as an initial grid for subsequent h-refinement. A logical domain is used as a reference domain and the mesh movement is implemented according to an appropriate domain transformation based on employing a harmonic mapping. A good feature of the adaptive methods based on harmonic maps is that existence, uniqueness, and non-singularity for the continuous map can be guaranteed from the theory of harmonic maps (*e.g.*, see [7, 13]). Our aim is to combine the advantages of the r–method (*e.g.*, the resolution of the mesh in regions of interest can be rapidly attained) and those of the h–method (*e.g.*, greater accuracy can be attained through the introduction of additional elements into the computational mesh), while keeping the mesh generator and the finite element solver independent of one another. In the proposed pre–processing algorithm we construct the harmonic map between the physical mesh and the logical mesh by employing an *iteration* procedure; at each iteration the mesh is moved "closer" to the harmonic map. At the limit, we obtain a grid which is aligned with the layers of the underlying solution and take into account possible anisotropies of the coefficients of the problem. This idea of iterating the harmonic mapping was proposed in [30, 24]. The pre–processing technique is applied to a number of test problems in two–dimensions; here, we shall demonstrate that the proposed mesh-redistribution strategy adapts the mesh extremely well to strong layers present in the solution without producing skew elements.

The paper is structured as follows. In Section 2 and Section 3 we introduce the model problem and formulate its discontinuous Galerkin finite element approximation, respectively. The pre–processing moving mesh algorithm is presented in Section 4, and its competitiveness is validated by means of numerical experiments in Section 5. Finally, in Section 6 we summarize the work presented and draw some conclusions.

2. Notation and model problem

In this section, we define some notation and introduce the model problem to be considered.

Throughout the paper we use the following standard function space notation. For a bounded domain D in $\mathbb{R}^d$, $d = 1, 2, 3$, we denote by $H^s(D)$ the standard real Sobolev space of order $s \in \mathbb{N}_0$. For $s = 0$ we write $L^2(D)$ in lieu of $H^0(D)$. Furthermore, we write $H_0^1(D)$ for the subspace of $H^1(D)$ of functions with zero trace on ∂D.

Let Ω be an open, bounded polyhedral domain in $\mathbb{R}^d$, $d = 2, 3$, and let Γ signify the union of its $(d-1)$-dimensional open faces; by $\boldsymbol{n}(x) = \{n_i\}_{i=1}^d$ we denote the unit outward normal vector to Γ at $x \in \Gamma$. We consider the following advection–diffusion–reaction problem:

$$\mathcal{L}u \equiv -\nabla \cdot (\boldsymbol{\varepsilon} \nabla u) + \nabla \cdot (\boldsymbol{\beta} u) + \gamma u = f \text{ in } \Omega, \quad u = 0 \text{ on } \Gamma, \tag{1}$$

where $f \in L^2(\Omega)$ and $\gamma \in L^\infty(\Omega)$ are real–valued functions, $\boldsymbol{\beta} = \{\beta_i\}_{i=1}^d$ is a vector whose entries are Lipschitz continuous real–valued functions on $\overline{\Omega}$, and $\boldsymbol{\varepsilon} = \{\boldsymbol{\varepsilon}_{ij}\}_{i,j=1}^d$ is a *symmetric* matrix whose entries $\boldsymbol{\varepsilon}_{ij}$ are bounded, piecewise continuous real–valued functions defined on $\overline{\Omega}$, with

$$\eta^T \boldsymbol{\varepsilon}(x) \eta > 0 \quad \forall \, \mathbf{0} \neq \eta \in \mathbb{R}^d \quad \forall x \in \overline{\Omega}.$$

We define the sets Γ_- and Γ_+, referred to as the inflow and outflow boundary, respectively, as

$$\Gamma_- = \{x \in \Gamma \ : \ \boldsymbol{\beta}(x) \cdot \boldsymbol{n}(x) < 0\}, \quad \Gamma_+ = \{x \in \Gamma \ : \ \boldsymbol{\beta}(x) \cdot \boldsymbol{n}(x) \geq 0\}.$$

Finally, we assume that the following (standard) hypothesis holds:

$$\gamma(x) + \frac{1}{2} \nabla \cdot \boldsymbol{\beta}(x) > 0 \quad \forall x \in \Omega.$$

The weak formulation of the model problem (1) is defined by: find $u \in H_0^1(\Omega)$ such that

$$\mathcal{B}(u, v) = \int_\Omega f v \, \mathrm{dx} \quad \forall v \in H_0^1(\Omega), \tag{2}$$

where the bilinear form $\mathcal{B}(\cdot, \cdot) : H_0^1(\Omega) \times H_0^1(\Omega)$ is defined by

$$\mathcal{B}(u, v) = \int_\Omega \boldsymbol{\varepsilon} \nabla u \cdot \nabla v \, \mathrm{dx} - \int_\Omega u \, \boldsymbol{\beta} \cdot \nabla v \, \mathrm{dx} + \int_\Omega \gamma \, uv \, \mathrm{dx}.$$

The well-posedness theory (in the case of weaker assumptions on the coefficients) of the boundary value problem (2) is provided in [15, 17], for example.

3. Discontinuous Galerkin discretization

In this section, we define some notation, and introduce the DG discretization of model problem (1).

3.1. Meshes, finite element spaces and trace operators. Let $\mathcal{T}_h$ be a *shape-regular* (not necessarily matching) partition of Ω into disjoint open elements K such that $\overline{\Omega} = \cup_{K \in \mathcal{T}_h} \overline{K}$, where each $K \in \mathcal{T}_h$ is the image of a fixed master element $\widehat{K}$, *i.e.*, $K = F_K(\widehat{K})$, and $\widehat{K}$ is either the open unit d-simplex or the open unit hypercube in $\mathbb{R}^d$, $d = 2, 3$. We denote by h_K the diameter of $K \in \mathcal{T}_h$, and set $h = \max_{K \in \mathcal{T}_h} h_K$.

We define the $(d-1)$–dimensional faces (if $d = 2$, "face" means "edge") of $\mathcal{T}_h$ as follows. An *interior face* of $\mathcal{T}_h$ is the (non-empty) interior of $\partial K_1 \cap \partial K_2$, where K_1 and K_2 are two adjacent elements of $\mathcal{T}_h$, not necessarily matching (cf. Figure 1). Similarly, a *boundary face* of $\mathcal{T}_h$ is the (non-empty) interior of $\partial K \cap \Gamma$, where K is a boundary element of $\mathcal{T}_h$. We denote by $\mathcal{F}_h^I$ and $\mathcal{F}_h^B$ the sets of all interior and boundary faces of $\mathcal{T}_h$, respectively, and set $\mathcal{F}_h = \mathcal{F}_h^I \cup \mathcal{F}_h^B$. Implicit in these definitions is the assumption that $\mathcal{T}_h$ respects the decomposition of Γ in the sense that each $F \in \mathcal{F}_h$ that lies on Γ belongs to the interior of exactly one of Γ_+, Γ_-.

We shall assume that the mesh $\mathcal{T}_h$ is *regular* or *1–irregular*; *i.e.*, each face of $K \in \mathcal{T}_h$ has at most one hanging node. Moreover, writing m_K to denote the d–dimensional measure of the element $K \in \mathcal{T}_h$, we assume that the following *bounded local variation* property holds: there exists a constant $C \geq 1$ such that, for any pair of elements K_1 and K_2 which share a $(d-1)$–dimensional face,

$$C^{-1} \leq m_{K_1} / m_{K_2} \leq C \, ,$$

cf. [10].

We also introduce the *local mesh size* function $\mathtt{h}$ in $L^\infty(\mathcal{F}_h)$ defined as

$$\mathtt{h}(x) = \begin{cases} \dfrac{\min\{m_{K_1}, m_{K_2}\}}{m_F} & \text{if } x \text{ is in the interior of } \overline{F} = \overline{\partial K_1 \cap \partial K_2} \, , \\ \dfrac{m_K}{m_F} & \text{if } x \text{ is in the interior of } \overline{F} = \overline{\partial K \cap \Gamma} \, , \end{cases} \tag{3}$$

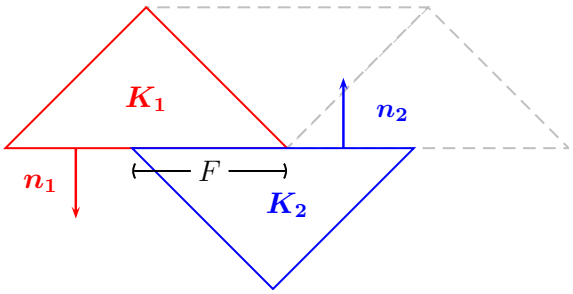


FIGURE 1. A sample of neighboring elements K_1 and K_2, and their shared face $F \in \mathcal{F}_h^I$ in two–dimensions.

where, m_F denotes the $(d-1)$–dimensional measure of the face $F \in \mathcal{F}_h$.

For piecewise smooth vector– and scalar–valued functions $\boldsymbol{\tau}$ and v, respectively, we introduce the following trace operators. Let $F \in \mathcal{F}_h^I$ be an interior face shared by two elements K_1 and K_2 with outward unit normal vectors $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$, respectively (cf. Figure 1). For $i = 1, 2$, let $\boldsymbol{\tau}_i$ and v_i be the traces of $\boldsymbol{\tau}$ and v on ∂K_i taken within the interior of K_i, respectively. We define the *jump* and the *average* across F by

$$[\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau}_1 \cdot \boldsymbol{n}_1 + \boldsymbol{\tau}_2 \cdot \boldsymbol{n}_2, \qquad [\![v]\!] = v_1 \boldsymbol{n}_1 + v_2 \boldsymbol{n}_2,$$
$$\{\!\{\boldsymbol{\tau}\}\!\} = (\boldsymbol{\tau}_1 + \boldsymbol{\tau}_2)/2, \qquad \{\!\{\boldsymbol{v}\}\!\} = (\boldsymbol{v}_1 + \boldsymbol{v}_2)/2,$$

respectively. On a boundary face $F \in \mathcal{F}_h^B$, we set, analogously,

$$[\![\boldsymbol{\tau}]\!] = \boldsymbol{\tau} \cdot \boldsymbol{n}, \quad [\![v]\!] = v\,\boldsymbol{n}, \quad \{\!\{\boldsymbol{\tau}\}\!\} = \boldsymbol{\tau}, \quad \{\!\{v\}\!\} = v.$$

Given that K is an element in the subdivision $\mathcal{T}_h$, we denote by ∂K the union of $(d-1)$–dimensional open faces of K. Let $x \in \partial K$ and suppose that $\mathbf{n}_K(x)$ denotes the unit outward normal vector to ∂K at x. With these conventions, we define the inflow and outflow parts of ∂K, respectively, by

$$\partial_- K = \{x \in \partial K : \ \boldsymbol{\beta}(x) \cdot \mathbf{n}_K(x) < 0\}\ , \ \partial_+ K = \{x \in \partial K : \ \boldsymbol{\beta}(x) \cdot \mathbf{n}_K(x) \geq 0\}\ .$$

Given a piecewise smooth scalar function v and an element $K \in \mathcal{T}_h$, we denote by v^+ (respectively, v^-) the interior (respectively, exterior) trace of v defined on ∂K (respectively, $\partial K \setminus \Gamma$).

Finally, for a non negative approximation order ℓ, we denote by $\mathcal{P}^\ell(\widehat{K})$ the set of polynomials of total degree ℓ on $\widehat{K}$; when $\widehat{K}$ is the unit hypercube in $\mathbb{R}^d$, we also consider $\mathcal{Q}^\ell(\widehat{K})$, the set of all tensor–product polynomials on $\widehat{K}$ of degree ℓ in each coordinate direction. We collect the F_K in the vector $\boldsymbol{F} = \{F_K \ : \ K \in \mathcal{T}_h\}$, and define the following DG finite element space

$$V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F}) = \{u \in L^2(\Omega) \ : \ u \circ F_K \in \mathcal{M}^\ell(\widehat{K}) \quad \forall\ K \in \mathcal{T}_h\},$$

where $\mathcal{M}$ is either $\mathcal{P}$ or $\mathcal{Q}$.

3.2. Discontinuous Galerkin approximation. For simplicity of presentation, we suppose that the entries of the matrix $\boldsymbol{\varepsilon}$ are constant on each element $K \in \mathcal{T}_h$, *i.e.*,

$$\boldsymbol{\varepsilon} \in [V_h^0(\Omega, \mathcal{T}_h, \boldsymbol{F})]_{\rm sym}^{d\times d}.$$

The extension to general $\boldsymbol{\varepsilon} \in [L^\infty(\Omega)]^{d\times d}_{\rm sym}$ follows analogously based on employing the modified DG method proposed in [14]. With the above assumption, the matrix function $\boldsymbol{\varepsilon}$ admits a unique square root $\sqrt{\boldsymbol{\varepsilon}} \in [V_h^0(\Omega, \mathcal{T}_h, \boldsymbol{F})]^{d\times d}_{\rm sym}$. In the following, we denote by $|\cdot|_2$ the matrix norm subordinate to the l_2–vector norm on $\mathbb{R}^d$, set $\overline{\varepsilon} = \left|\sqrt{\boldsymbol{\varepsilon}}\right|_2^2$, and write $\overline{\varepsilon}_K = \overline{\varepsilon}|_K$ for any $K \in \mathcal{T}_h$. By defining ∇_h as the elementwise application of the operator ∇, it will be assumed that

$$\boldsymbol{\beta} \cdot \nabla_h v \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F}) \quad \forall v_h \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F}).$$

For a given partition $\mathcal{T}_h$ we denote by $H^2(\mathcal{T}_h)$ the space of functions that belong to $H^2(K)$ for any $K \in \mathcal{T}_h$, and we define the bilinear form $\mathcal{B}_h(\cdot,\cdot) : H^2(\mathcal{T}_h) \times H^2(\mathcal{T}_h) \longrightarrow \mathbb{R}$ as

$$\begin{aligned}
\mathcal{B}_h(u,v) \;=\; & \int_\Omega \boldsymbol{\varepsilon}\nabla_h u \cdot \nabla_h v \,\mathrm{dx} - \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\boldsymbol{\varepsilon}\nabla_h u\}\!\} \cdot [\![v]\!] \,\mathrm{ds} \\
& - \sum_{F\in\mathcal{F}_h} \int_F [\![u]\!] \cdot \{\!\{\boldsymbol{\varepsilon}\nabla_h v\}\!\} \,\mathrm{ds} + \sum_{F\in\mathcal{F}_h} \int_F \sigma\, [\![u]\!] \cdot [\![v]\!] \,\mathrm{ds} \\
& - \int_\Omega u\, \boldsymbol{\beta} \cdot \nabla_h v \,\mathrm{dx} + \sum_{K\in\mathcal{T}_h} \int_{\partial_+ K} (\boldsymbol{\beta}\cdot\mathbf{n}_K)\, u^+ v^+ \,\mathrm{ds} \\
& + \sum_{K\in\mathcal{T}_h} \int_{\partial_- K\setminus\Gamma} (\boldsymbol{\beta}\cdot\mathbf{n}_K)\, u^- v^+ \,\mathrm{ds} + \int_\Omega \gamma\, uv \,\mathrm{dx},
\end{aligned} \tag{4}$$

where σ is the *discontinuity penalization* term defined by $\sigma|_F = \sigma_F$ for $F \in \mathcal{F}_h$, with

$$\sigma_F = \alpha \frac{\epsilon}{\mathtt{h}} \quad \text{on } F \in \mathcal{F}_h. \tag{5}$$

Here, α is a positive real number at our disposal; for reasons of stability, this must be selected sufficiently large, cf. [10]. The *local mesh size* $\mathtt{h}$ in $L^\infty(\mathcal{F}_h)$ is defined according to (3), and the function ϵ in $L^\infty(\mathcal{F}_h)$ is defined as

$$\epsilon(x) = \begin{cases} \max\{\overline{\varepsilon}_{K_1}, \overline{\varepsilon}_{K_2}\} & \text{if } x \text{ is in the interior of } \overline{F} = \overline{\partial K_1 \cap \partial K_2}\ , \\ \overline{\varepsilon}_K & \text{if } x \text{ is in the interior of } \overline{F} = \overline{\partial K \cap \Gamma}\ . \end{cases}$$

The DG approximation of problem (1) reads as follows: find $u_h \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F})$ such that

$$\mathcal{B}_h(u_h, v) = \int_\Omega f v \,\mathrm{dx} \quad \forall v \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F}). \tag{6}$$

Remark 3.1. *From a close inspection of the expression of the bilinear form $\mathcal{B}_h(\cdot,\cdot)$ given in* (4) *it is clear that we have used the* symmetric interior penalty *method proposed in* [2] *for the approximation of the diffusion term (first two lines of* (4)*), and a DG method with* upwinded fluxes *(see* [5]*) for the approximation of the advection term (the third line and the first term in the fourth line in* (4)*). With minor changes, we could also consider any other* symmetric *DG approximations of the second order terms arising in the model problem* (1) *(see* [3]*).*

4. A pre–processing moving mesh method

In this section we describe our pre–processing moving mesh algorithm. We first recall the main features of harmonic function theory and its application to mesh generation; then we present our numerical pre–processing technique and we describe some implementation issues of the numerical scheme.

4.1. Harmonic maps between Riemannian manifolds. The harmonic function theory can provide a general framework for developing useful mesh generators (cf. [7]). A good feature of adaptive methods based on harmonic mappings is that existence, uniqueness, and nonsingularity for the continuous map can be guaranteed from the theory of harmonic maps.

Let Ω and Ω_C be compact Riemannian manifolds of dimension d with metric tensors d_{ij} and $r_{\nu\mu}$ defined in the local coordinate systems denoted by $\boldsymbol{x}$ and $\boldsymbol{\xi}$, respectively. We will refer to Ω_C as the logical domain. Following Dvinsky [7], we define the energy for a map $\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{x})$ as

$$E(\boldsymbol{\xi}) = \frac{1}{2}\int_\Omega \sqrt{d}\, d^{ij} r_{\nu\mu} \frac{\partial \xi^\nu}{\partial x^i}\frac{\partial \xi^\mu}{\partial x^j}\,\mathrm{d}\boldsymbol{x},$$

where $d = \det(d_{ij})$, $d_{ij} = (d^{ij})^{-1}$, and the standard summation convention is assumed. With a Euclidean metric on the logical domain Ω_C, the energy functional becomes

$$E(\boldsymbol{\xi}) = \frac{1}{2}\sum_k \int_\Omega G^{ij}\,\frac{\partial \xi^k}{\partial x^i}\frac{\partial \xi^k}{\partial x^j}\,\mathrm{d}\boldsymbol{x}, \tag{7}$$

where we have set $G^{ij} = \sqrt{d}\, d^{ij}$. The Euler–Lagrange equations, whose solution minimize the above energy, are given by

$$\frac{\partial}{\partial x^i}\left(G^{ij}\frac{\partial \xi^k}{\partial x^j}\right) = 0. \tag{8}$$

The inverse of G^{ij} is called the *monitor function* and plays a key role in the development of moving mesh algorithms.

As an example, let us denote as $(x^1(\xi^1,\xi^2), x^2(\xi^1,\xi^2))$ the harmonic mapping in two–dimensions. Here (x^1, x^2) and (ξ^1, ξ^2) are the physical and the computational coordinates, respectively. Then, the energy functional (7) can be rewritten as

$$E(\xi^1,\xi^2) = \frac{1}{2}\int_\Omega \sqrt{\det(D)}\left(\nabla {\xi^1}^\top D^{-1}\nabla\xi^1 + \nabla {\xi^2}^\top D^{-1}\nabla\xi^2\right)\mathrm{d}x^1\mathrm{d}x^2,$$

where $D = \left\{d^{ij}\right\}_{i,j}$ is a symmetric positive matrix depending on (x^1, x^2). In this case, the corresponding Euler–Lagrange equations are given by

$$\nabla\cdot(\sqrt{\det(D)}D^{-1}\nabla\xi^1) = 0, \quad \nabla\cdot(\sqrt{\det(D)}D^{-1}\nabla\xi^2) = 0.$$

The existence and uniqueness of harmonic maps are established in [13, 23] under suitable assumptions on Ω_C, *i.e.*, the Riemannian curvature of Ω_C is non-positive and its boundary is convex. Since Ω_C is obtained by construction, both requirements can usually be satisfied. Solutions to (8) are harmonic functions giving a continuous, one-to-one mapping with continuous inverse, which is differentiable and has a nonzero Jacobian. The idea is that one is free to specify G^{ij} as a function of physical coordinates when defining the energy (7), and that minimizing this energy will result in a harmonic mapping with the desired properties.

To solve the Euler–Lagrange equations (8) numerically, one usually interchanges dependent and independent variables. The solution of (8) requires evaluating derivatives of $\boldsymbol{\xi}$ with respect to the physical coordinates $\boldsymbol{x}$. In moving mesh computations, one usually specifies the logical arrangement of grid points and computes the physical coordinates of the (mapped) grid points. More precisely, we solve the inverse mapping of $\boldsymbol{\xi}$, $\boldsymbol{x}(\boldsymbol{\xi})$, because it directly defines the mesh on Ω.

4.2. The numerical pre–processing scheme. In this section, we will describe the moving mesh pre–processing strategy.

In order to design a sequence of appropriate meshes for the DG scheme (6), we separate the mesh generation algorithm into two parts: mesh–movement and adaptive h–refinement (subdivision). The mesh–movement is an iterative procedure to construct the harmonic map between the physical mesh and the logical mesh. Given an initial (uniform) mesh, at each iteration step the aim is to move the mesh closer to the one defined by the harmonic map. In the numerical computation, we always keep the initial mesh in the logical domain fixed. This mesh is not used to approximate the advection–diffusion–reaction problem (1); however, the difference between the grid points defined in the logical domain and the corresponding ones computed by numerically solving the system of second–order elliptic PDEs defined in (8), is used to move the mesh in the physical domain. More precisely, in the first step we chose an initial mesh $\boldsymbol{\xi}^0$ in the logical domain Ω_C that is used as a reference grid only, and that will be kept unchanged throughout the computation. We also choose an initial mesh $\boldsymbol{x}^0$ on the physical domain Ω. Once the solution u_h of problem (6) is computed on the initial mesh $\boldsymbol{x}^0$, the inverse matrix of the monitor function, G^{ij} (which in general depends on u_h), can be computed. By approximating the solution of the Euler–Lagrange equation (8) with a conforming finite element method, we obtain a new mesh in the logical domain, denoted by $\boldsymbol{\xi}^*$. If the difference (in a suitable norm) between $\boldsymbol{\xi}^*$ and the initial mesh $\boldsymbol{\xi}^0$ is not small, we move the mesh in the physical space by using the computed error between $\boldsymbol{\xi}^*$ and $\boldsymbol{\xi}^0$, and obtain a new grid. This procedure is repeated until the difference between $\boldsymbol{\xi}^*$ and the initial mesh $\boldsymbol{\xi}^0$ is sufficiently small. We describe more precisely our pre–processing scheme in the following algorithm.

Algorithm 1. *Suppose that an approximate solution* $u_h \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F})$ *of the problem* (6) *on the initial mesh* $\boldsymbol{x}^0$*, and a prescribed tolerance* `TOL` *are given. The pre–processing moving mesh algorithm is described in Figure 2.*

In order to guarantee the quality of the harmonic map, the mesh–moving process is repeated until the l_2–norm of the distance between the computed numerical solution of the harmonic map (8) and the initial logical mesh $\boldsymbol{\xi}^0$ is smaller than a preassigned tolerance `TOL` (cf. Algorithm 1, part `B`). In Algorithm 1, part `E`, the nodal values of the new grid are computed based on the assumption that the surface of u_h on Ω is fixed; more precisely, we assume that *the surface of* u_h *on* Ω *will not move* though the nodes of the mesh may be moved to new locations.

Next, we will discuss in detail the key ingredients for our numerical scheme. To this end, let $\mathcal{T}_h^0$ be a triangulation of Ω into a set of elements, each denoted by K. The finite element space employed is chosen to be the usual *continuous* finite element space consisting of piecewise linear polynomials defined on $\mathcal{T}_h^0$. Let $\boldsymbol{X}^0 = \{\boldsymbol{x}_i^0\}$ denote the nodes of the initial mesh $\mathcal{T}_h^0$, and $\Psi^0 = \{\psi_i^0\}$ be the associated *conforming* linear finite element basis functions defined so that $\psi_i^0(\boldsymbol{x}_j^0) = \delta_{ij}$, where

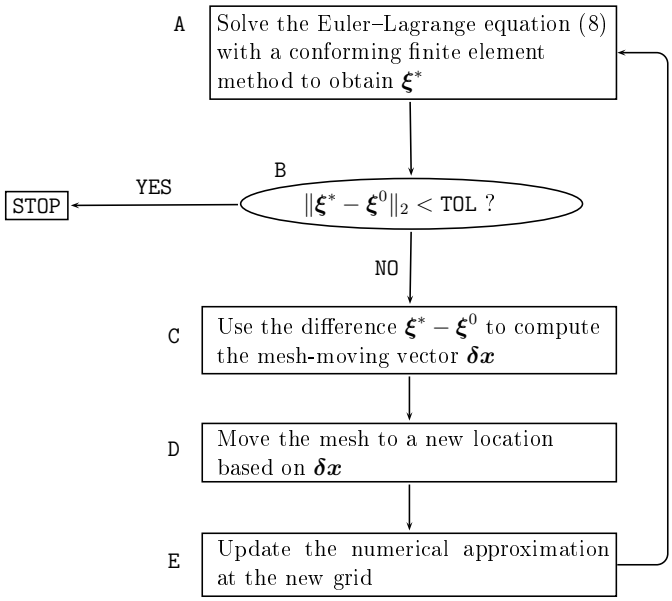


FIGURE 2. The moving mesh pre–processing algorithm.

δ_{ij} denotes the Kronecker delta symbol. Suppose also that we have computed the approximate solution $u_h \in V_h^\ell(\Omega, \mathcal{T}_h^0, \boldsymbol{F})$ of problem (6) on the initial mesh $\mathcal{T}_h^0$ consisting of the nodes $\boldsymbol{X}^0 = \{\boldsymbol{x}_i^0\}$. Analogously, we fix a logical domain Ω_C, and an initial mesh $\boldsymbol{\xi}^0$ in Ω_C, with nodes $\boldsymbol{\xi}^0 = \{\boldsymbol{\xi}_i^0\}$.

Remark 4.1. *In practical applications, whenever the physical domain Ω is convex and is of regular shape (say a convex polygon/polyhedron), we can simply identify the logical domain with the physical one,* i.e., *$\Omega_C \equiv \Omega$ and we can choose a uniform initial mesh,* i.e., *$\boldsymbol{\xi}^0 \equiv \boldsymbol{X}^0$.*

We now need to obtain a new mesh $\mathcal{T}_h^*$ with nodes $\boldsymbol{X}^* = \{\boldsymbol{x}_i^*\}$. This can be done in the following two steps:

i) **Compute the error.** Without lose of generality, let $G = \left\{G^{ij}\right\}_{i,j}$ be a function of u_h only, *i.e.*, $G = G(u_h)$; we will describe later how $G = G(u_h)$ can be chosen. We first approximate with a conforming finite element method the following generalized Poisson problems subject to appropriate Dirichlet boundary conditions:

$$\frac{\partial}{\partial x^i}\left(G\frac{\partial \xi^k}{\partial x^j}\right) = 0, \quad \boldsymbol{\xi}|_{\partial\Omega} = \boldsymbol{\xi}_b, \ k = 1, \ldots, d. \tag{9}$$

By doing so we obtain a new mesh $\boldsymbol{\xi}^*$ with nodes $\boldsymbol{\xi}^* = \{\boldsymbol{\xi}_i^*\}$. The error function, which will play a key role in the prediction of the movement of the numerical grid in the physical space Ω, is defined as $\boldsymbol{\delta\xi} = \boldsymbol{\xi}^0 - \boldsymbol{\xi}^*$.

ii) **Compute the movement.** For any given element $K \in \mathcal{T}_h$, the piecewise linear element transformation from the logical domain Ω_C to the computational domain Ω such that $\boldsymbol{\xi}_i^* \longrightarrow \boldsymbol{x}_i$ has constant gradient on K and

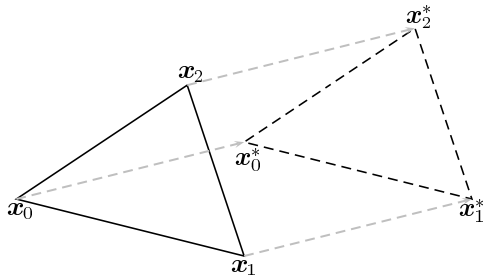


FIGURE 3. A sample of the element motion.

satisfies the following linear system

$$\mathtt{B}^{\mathtt{H}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} = \mathtt{B},$$

where, for any element $K \in \mathcal{T}_h$, $\mathtt{B}^{\mathtt{H}}$ is the Jacobi matrix of the transformation between the initial coordinates of the logical domain $\boldsymbol{\xi}^0$ and the current coordinates $\boldsymbol{\xi}^*$, *i.e.*, the approximate solution of the harmonic map (9). Analogously, $\mathtt{B}$ is the Jacobi matrix of the transformation between the initial coordinates $\boldsymbol{X}^0$ in the computational domain and the current coordinates $\boldsymbol{X}$.

Solving the above linear system gives $\partial \boldsymbol{x}/\partial \boldsymbol{\xi}$ in K. If we take the volume of the element $|K|$ as the weight, the weighted average error of $\boldsymbol{X}$ at the i–th node is defined by

$$\boldsymbol{\delta x}_i = \frac{\sum_{K\in\mathcal{N}_i} |K| \, \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}|_K \, \boldsymbol{\delta \xi}_i}{\sum_{K\in\mathcal{N}_i} |K|},$$

where $\mathcal{N}_i$ denotes the set of elements $K \in \mathcal{T}_h$ surrounding the node $\boldsymbol{x}_i$. It can be shown that the above volume–weighted average converges to a smooth solution in measure when the size of the mesh goes to 0. The location of the nodes in the new mesh on the physical domain is taken as

$$\boldsymbol{x}_i^* = \boldsymbol{x}_i + \tau_i \, \boldsymbol{\delta x}_i, \tag{10}$$

where τ_i is the *length of the movement* and is defined by

$$\boldsymbol{\tau}_i = \theta \min_{K\in\mathcal{N}_i} \left\{ \frac{|K|}{\max\limits_{\substack{F\in\mathcal{F}_h \\ F\subset\partial K}} |F|} \right\},$$

with $\theta \in (0,1)$. A sample of the motion of an element based on (10) is illustrated in Figure 3.

In general, the l_2–norm of $\boldsymbol{\delta \xi}$ decreases at each iteration step, but, whenever the solution u_h is very singular (*e.g.*, the solution exhibits very large gradients), the decrease of $\|\boldsymbol{\delta \xi}\|_2$ could become slow.

5. Numerical experiments

In Section 4.2 we have proposed an adaptive grid adaptation procedure based on moving the mesh using a harmonic mapping approach. Here, we report some

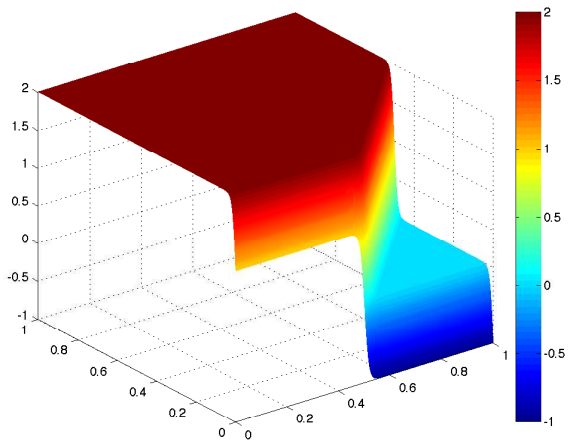


FIGURE 4. Example 1. Elevation plot of the analytical solution $u(x,y) = \tanh(60y) - \tanh(60x - 60y - 30)$.

numerical experiments indicating that our pre–processing technique is indeed efficient and robust. Throughout the section we have set $\theta = 0.5$, $\texttt{TOL} = 10^{-2}$ (cf. Algorithm 1), and $\alpha = 10$ (cf. (5)). Let $\mathcal{T}_h^0$ denote the initial (uniform) mesh on Ω, and $\#K$ be the total number of elements of $\mathcal{T}_h^0$. The monitor function G has been chosen as

$$(G|_K)^{-1} = \sqrt{\overline{\eta} + \delta\,\eta_K}\, I, \tag{11}$$

where, $\eta_K \in V_h^0(\Omega, \mathcal{T}_h, \boldsymbol{F})$ is a suitable error indicator whose definition will be specified in the following, and the average error $\overline{\eta}$ is given by

$$\overline{\eta} = \frac{\sum_{K\in\mathcal{T}_h} \eta_K}{\#K},$$

cf. [29, 23, 21, 23], for example. Here, δ is called the *intensity parameter* and is used to emphasize or deemphasize the influence of the error function on the mesh concentration. For larger δ the mesh distribution is more closely influenced by η_K, which generally results in more computational effort being expended in solving the pre–processing moving mesh method. Smaller δ gives less variation in G, resulting in less mesh adaptation. The influence of the choice of δ on our pre–processing scheme will be addressed in Section 5.2.

5.1. Example 1. Poisson's equation. Here, we let $\boldsymbol{\varepsilon} = I$, $\boldsymbol{\beta} = \mathbf{0}$, $\gamma = 0$, and select the source term $f(x,y)$, and the non-homogeneous Dirichlet boundary conditions, so that the analytical solution of the corresponding Poisson equation on $\Omega = (0,1)^2$ is given by $u(x,y) = \tanh(60y) - \tanh(60x - 60y - 30)$ (cf. Figure 4). The analytical solution exhibits sharp layers along the straight lines $y = 0$ and $y = x - 0.5$.

In this set of experiments, we have set $\delta = 1$ (cf. (11)), and

$$\eta_K = \frac{\|u - u_h\|_{0,K}^2}{|K|} \quad \forall K \in \mathcal{T}_h. \tag{12}$$

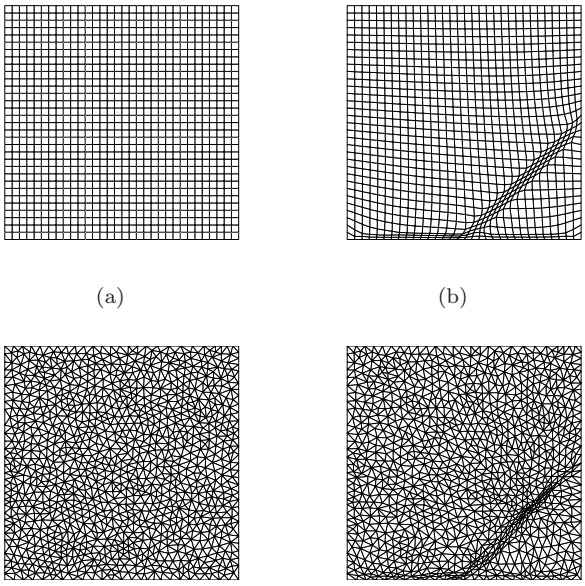


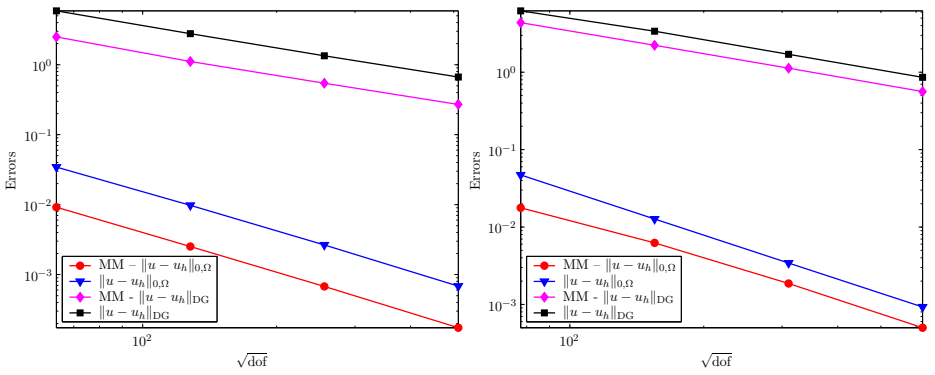
FIGURE 5. Example 1. Left: Initial quadrilateral and triangular meshes, respectively; Right: Corresponding meshes after application of the moving mesh method, respectively.

We have tested our pre–processing moving mesh method on both quadrilateral and triangular meshes, cf. Figure 5. More precisely, in Figures 5(a) & 5(c) we show the initial uniform quadrilateral and triangular meshes employed, respectively, while Figures 5(b) & 5(d) show the corresponding meshes after the mesh movement algorithm has been applied. Here, we clearly observe that the grids produced using the mesh movement algorithm have moved points in such a manner as to resolve the layers present within the analytical solution.

We now assess the improvement in the computed error when the mesh is first pre–processed using the mesh movement algorithm proposed in this article, before mesh adaptation is performed; for simplicity, here we consider simply global uniform refinement of the initial meshes, both with and without the pre–processing mesh movement step, see Figure 6. Here, we have computed the L^2–norm of the error between the analytical solution u and the approximate (DG) piecewise linear solution u_h, *i.e.*, $\|u - u_h\|_{0,\Omega}$. According to the *a priori* error analysis (see, *e.g.*, [15]), we expect to observe quadratic convergence as the mesh-size goes to zero. We have also computed the error between u and u_h measured in terms of the DG–norm, *i.e.*,

$$\|u - u_h\|^2_{\mathrm{DG}} = \sum_{K \in \mathcal{T}_h} |u - u_h|^2_{1,K} + \sum_{F \in \mathcal{F}_h} \|\sigma^{1/2} [\![u - u_h]\!]\|^2_{0,F};$$

in this case we expect to observe linear convergence of the DG-norm of the error as the mesh-size goes to zero. In Figures 6(a) & 6(b) we have reported the computed errors (loglog scale) as a function of the square root of the total number of degrees



(a) Quadrilateral meshes (b) Triangular meshes

FIGURE 6. Example 1. Computed errors versus the square root of the total number of degrees of freedom (loglog scale) employing uniform refinement.

step	dof	$\|u-u_h\|_{0,\Omega}$	ratio	$\|u-u_h\|_{\text{DG}}$	ratio
1	4096	0.9183E-02 (0.3437E-01)	3.7430	2.496 (5.890)	2.3597
2	16384	0.2519E-02 (0.9760E-02)	3.8743	1.111 (2.779)	2.5012
3	65536	0.6769E-03 (0.2653E-02)	3.9191	0.5436 (1.342)	2.4682
4	262144	0.1742E-03 (0.6828E-03)	3.9198	0.2700 (0.6655)	2.4645

TABLE 1. Example 1. Computed errors both with and without (in parenthesis) the pre–processing moving mesh method, and the ratio of these errors (ratio): quadrilateral meshes.

of freedom (dof) both with and without the pre–processing moving mesh method. The results in Figure 6(a) refer to the quadrilateral meshes, cf. Figures 5(a) & 5(b), whereas the analogous results obtained on triangular grids (cf. Figures 5(c) & 5(d)) are shown in Figure 6(b). Here, we observe the expected convergence rates are achieved; moreover we see that the use of the pre–processing technique clearly leads to a consistent reduction in the computed error as the meshes are uniformly refined. More precisely, on quadrilateral meshes, the use of the pre–processing technique reduces the error by a factor of around 4 when the error is computed in the L^2–norm, and by a factor of around 2.5 when the error is measured in the DG–norm. On triangular grids the L^2–norm of the error is reduced by a factor of around 2, while for the DG-norm of the error the improvement is by a factor of about 1.5. For completeness, in Table 1 we summarize the computed errors on quadrilateral meshes and the ratio between the norms of the error with and without the pre–processing moving mesh method (ratio). The analogous results obtained on triangular grids are shown in Table 2.

We have run the same set of experiments by substituting the error indicator η_K given in (12) with $\eta_K = |u-u_h|^2_{1,K}$, and the same behavior has been observed. These results are not reported here for the sake of brevity.

step	dof	$\|u-u_h\|_{0,\Omega}$	ratio	$\|u-u_h\|_{\mathrm{DG}}$	ratio
1	6012	0.1770E-01 (0.4701E-01)	2.6563	4.367 (6.218)	1.4237
2	24048	0.6229E-02 (0.1270E-01)	2.0392	2.229 (3.384)	1.5184
3	96192	0.1865E-02 (0.3429E-02)	1.8382	1.127 (1.702)	1.5107
4	384768	0.5004E-03 (0.9275E-03)	1.8534	0.5627 (0.8597)	1.5278

TABLE 2. Example 1. Computed errors both with and without (in parenthesis) the pre–processing moving mesh method, and the ratio of these errors (ratio): triangular meshes.

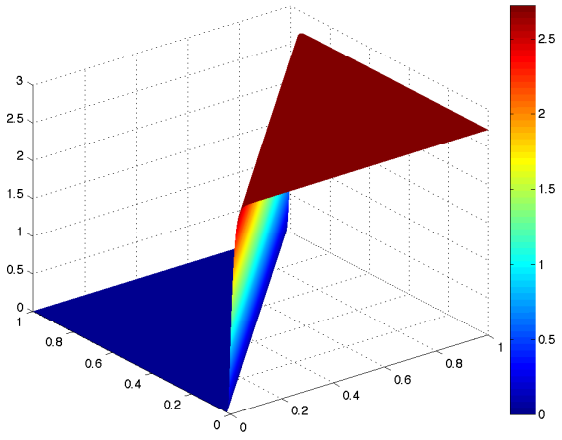


FIGURE 7. Example 2. Elevation plot of the analytical solution $u(x,y)$ given in (13) with $\varepsilon = 10^{-2}$.

5.2. Example 2. Advection–diffusion problem with constant coefficients: Straight internal layer. For a given $\varepsilon \in \mathbb{R}^+$, we let $\boldsymbol{\varepsilon} = \varepsilon I$, $\boldsymbol{\beta} = [1,1]^\top$, $\gamma = 0$, and consider an advection–diffusion problem with constant coefficients on $\Omega = (0,1)^2$. We choose the source term $f(x,y)$ and the non-homogeneous Dirichlet boundary conditions such that the corresponding analytical solution of problem (1) is given by

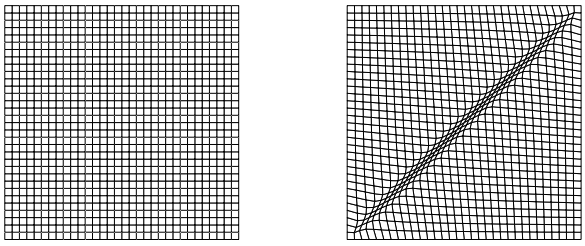
$$u(x,y) = \exp\left(\frac{1-\exp(-\frac{x-y}{\varepsilon})}{1-\exp(-\frac{1}{\varepsilon})}\right). \tag{13}$$

For $\varepsilon \searrow 0$, the analytical solution exhibits a sharp internal layer along the straight line $y = x$. Firstly, we choose $\varepsilon = 10^{-2}$ (cf. Figure 7 where the elevation plot of the analytical solution is shown).

Here, we take $\delta = 1$ (cf. (11)), and the error indicator η_K is chosen as

$$\eta_K = |u-u_h|^2_{1,K} \quad \forall K \in \mathcal{T}_h. \tag{14}$$

For the sake of simplicity, we focus here on quadrilateral meshes. The initial uniform mesh is shown in Figure 8(a); the corresponding one obtained by using our pre–processing moving mesh method is reported in Figure 8(b). At each step



(a) (b)

FIGURE 8. Example 2. (a) Initial quadrilateral mesh; (b) Quadrilateral mesh obtained with the pre–processing moving mesh method.

`step`	`dof`	$\|u - u_h\|_{0,\Omega}$	`ratio`
1	4096	0.4102E-01 (0.6514E-01)	1.5880
2	16384	0.6699E-02 (0.2093E-01)	3.1243
3	65536	0.1846E-02 (0.5580E-02)	3.0228

TABLE 3. Example 2. Computed L^2–errors both with and without (in parenthesis) the pre–processing moving mesh method, and the ratio of these errors (`ratio`).

of refinement `step` $= 0, 1, 2, 3$ we have considered a global uniform refinement of these initial grids, and, in order to compare the performance of our pre–processing algorithm, we have computed the error in the L^2–norm, namely $\|u - u_h\|_{0,\Omega}$, where u_h is the piecewise linear approximation of the analytical solution u. According to the theory developed in [15], we expect that the error goes to zero as $O(h^2)$ whenever $h \searrow 0$.

In Table 3 we report the computed errors in the L^2–norm. The results between parenthesis refer to the errors computed without the pre–processing moving mesh method. To compare the performance we have also reported the ratio of the L^2–norm of the errors computed with and without the pre–processing method (`ratio`). We clearly observe that the pre–processing moving mesh algorithm reduces the computed error by a factor of around 3.

We now investigate the effects of the choice of the *intensity parameter* δ (cf. (11)) on the performance of the pre–processing moving mesh method. In Figure 9 we show the grids obtained with the pre–processing moving mesh method for $\delta = 5, 10, 20$. The corresponding computed errors in the L^2–norm for the first three steps of refinement are reported in Table 4. It can be observed from the results reported in Table 4 that increasing the value of δ may lead to a degradation in the size of the computed error.

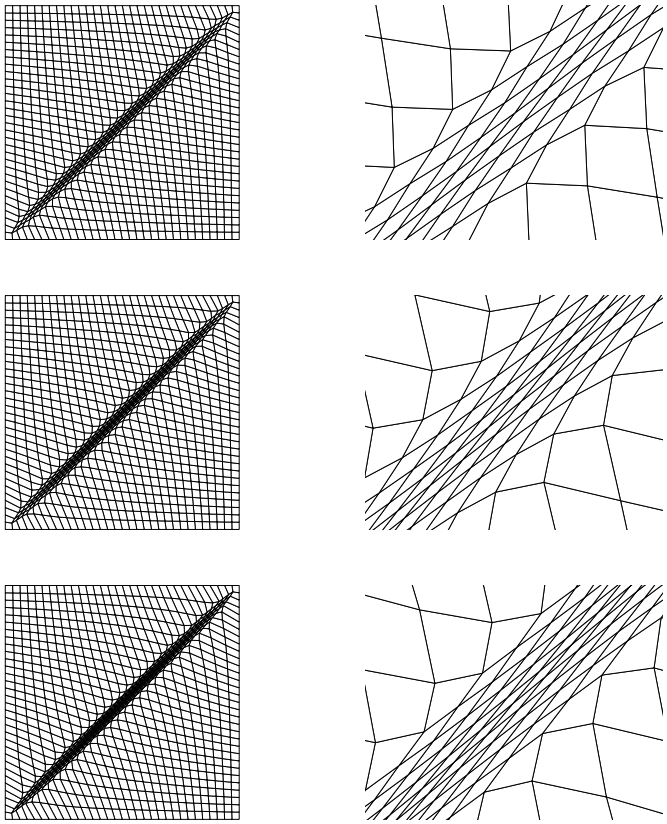


FIGURE 9. Example 2. Quadrilateral meshes (and a zoom near the point $(0.5, 0.5)$) obtained with the pre–processing moving mesh method for $\delta = 5, 10, 20$, respectively.

step	dof	$\delta = 5$	$\delta = 10$	$\delta = 20$
1	4096	0.9512E-01	0.1378E-00	0.2026E-00
2	16384	0.8281E-02	0.9885E-02	0.1230E-01
3	65536	0.2511E-02	0.3201E-02	0.4277E-02

TABLE 4. Example 2. Computed errors $\|u - u_h\|_{0,\Omega}$ using the pre–processing moving mesh method for $\delta = 5, 10, 20$, respectively.

5.3. Example 3. Advection–diffusion problem with constant coefficients: Circular internal layer. For a given $\varepsilon \in \mathbb{R}^+$, we let $\boldsymbol{\varepsilon} = \varepsilon I$, $\boldsymbol{\beta} = [2, 3]^\top$, $\gamma = 0$, and consider an advection–diffusion problem with constant coefficients on $\Omega = (0, 1)^2$. We choose as the analytical solution

(15)

$$u(x, y) = 16(x - x^2)(y - y^2)\left(\frac{1}{2} + \frac{\arctan(2\sqrt{\varepsilon^{-1}}\left[1/16 - (x - 1/2)^2 - (y - 1/2)^2\right])}{\pi}\right),$$

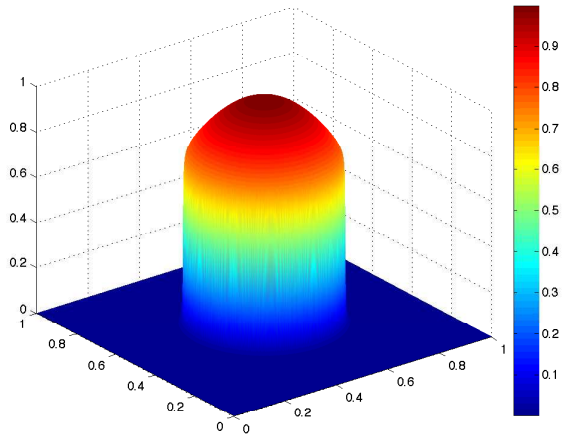
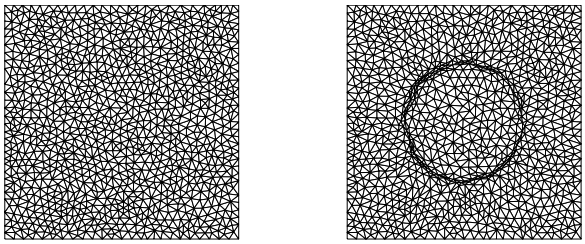


FIGURE 10. Example 3. Elevation plot of the analytical solution $u(x, y)$ given in (15) with $\varepsilon = 10^{-6}$.



(a) (b)

FIGURE 11. Example 3. (a) Initial triangular mesh; (b) Mesh obtained with the pre–processing moving mesh method.

and we adjust the source term $f(x, y)$ and the non-homogeneous Dirichlet boundary conditions accordingly. For $\varepsilon \searrow 0$, the analytical solution exhibits a sharp circular internal layer. We choose $\varepsilon = 10^{-6}$ (cf. Figure 10 where the elevation plot of the analytical solution is shown).

Firstly, we take $\delta = 1$ (cf. (11)), and the error indicator η_K is chosen again as

$$\eta_K = |u - u_h|^2_{1,K} \quad \forall K \in \mathcal{T}_h. \tag{16}$$

The initial triangular mesh and that obtained with our pre–processing moving mesh method are reported in Figure 11. We clearly observe that, by employing the pre–processing algorithm, many elements are concentrated near the internal circular layer present in the underlying analytical solution, as we would expect.

We now turn to another choice of the monitor function based on an *a posteriori* error estimator. To this end, we note that in many practical applications the

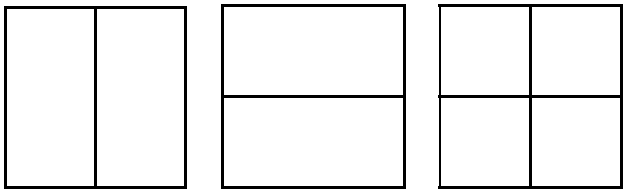


FIGURE 12. Example 3. Trial Cartesian refinements in 2D: anisotropic refinements 12(a)–12(b); isotropic refinement 12(c).

quantity of interest is an output or target functional $J(\cdot)$ of the solution (*e.g.*, the flux through the outflow boundary of the computational domain). Following [16, 27, 10] we recall the following dual or adjoint problem: find $z \in H^2(\Omega, \mathcal{T}_h)$ such that

$$\mathcal{B}_h(w, z) = J(w) \quad \forall w \in H^2(\Omega, \mathcal{T}_h),$$

and assume that the choice of the linear functional under consideration guarantees that the above problem possesses a unique solution. By choosing $w = u - u_h$, recalling the linearity of $J(\cdot)$ and the Galerkin orthogonality property, we obtain the following error representation formula

$$J(u) - J(u_h) = \mathcal{B}_h(u - u_h, z - z_h) \quad \forall z_h \in V_h^\ell(\Omega, \mathcal{T}_h, \boldsymbol{F}).$$

Using the consistency of the DG formulation, and applying the divergence theorem, we can rewrite the above equality as

$$J(u) - J(u_h) = \sum_{K \in \mathcal{T}_h} \eta_K, \tag{17}$$

where the analytic expression for η_K is given in [10, Eq. 5.4]. The triangle inequality gives

$$|J(u) - J(u_h)| \leq \sum_{K \in \mathcal{T}_h} |\eta_K|\,. \tag{18}$$

Here, we choose as the monitor function the *a posteriori* error indicator given in (18), and, for a user-defined tolerance `tol`, we now consider the problem of designing an appropriate finite element mesh $\mathcal{T}_h$ such that $|J(u) - J(u_h)| \leq \texttt{tol}$, subject to the constraint that the total number of elements in $\mathcal{T}_h$ is minimized. As in [11, 12] we exploit the *a posteriori* error bound (18) with z replaced by a discontinuous Galerkin approximation $\hat{z}_h$ computed on the same mesh $\mathcal{T}_h$ used for the primal solution u_h, but with a higher degree polynomial, *e.g.*, $\hat{z}_h \in V_h^{\ell+1}(\Omega, \mathcal{T}_h, \boldsymbol{F})$. We then use the fixed fraction mesh refinement algorithm, with refinement and derefinement fractions set to 25% and 10%, respectively. To subdivide the elements which have been flagged for refinement, we employ either an isotropic refinement strategy (cf. Figure 12(c)) or the anisotropic refinement strategy proposed in [10] which is based on choosing the most competitive subdivision of K from a series of trial refinements (cf. Figure 12).

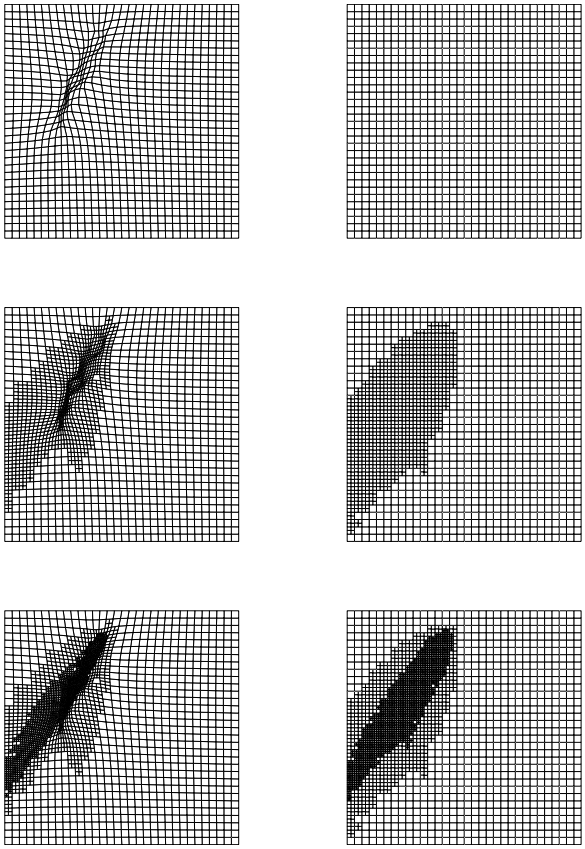


FIGURE 13. Example 3. Point value evaluation. First three steps of the isotropic adaptively refined meshes with (left) and without (right) the pre–processing moving mesh strategy.

5.3.1. Point value evaluation. In the first example, $J(u)$ is simply the evaluation of the solution u at the point $(0.43, 0.9)$, therefore $J(u) = 5.4856E-04$.

In Figure 13 and Figure 14 we show the first three meshes generated using isotropic and anisotropic mesh adaptation, respectively. In both cases we compare the adaptively refined meshes obtained when the pre–processing moving mesh strategy is applied to the initial mesh (left) and the case when adaptive refinement of the mesh is employed directly on the initial uniform mesh (right).

In Figure 15 we compare the error in the computed target functional $J(\cdot)$ obtained both with and without the pre–processing moving mesh strategy, based on employing both the adaptive isotropic and anisotropic mesh refinement strategies. As in the previous section, we clearly observe the superiority of employing the pre–processing moving mesh strategy. Indeed, we see that the error $|J(u) - J(u_h)|$

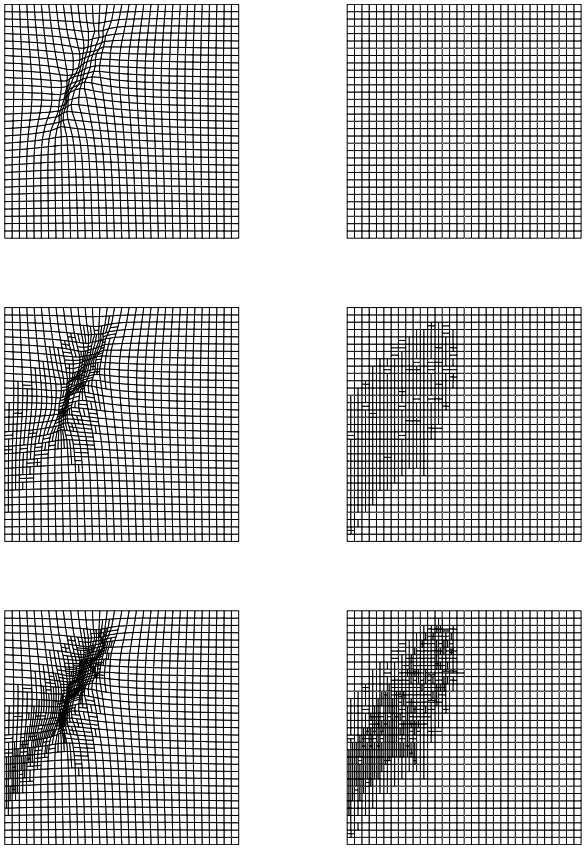


FIGURE 14. Example 3. Point value evaluation. First three steps of the anisotropic adaptively refined meshes with (left) and without (right) the pre–processing moving mesh strategy.

computed on the series of isotropically or anisotropically refined meshes is always less than the corresponding quantity computed without employing the pre–processing strategy to the initial mesh. In particular, we observe that the isotropic and anisotropic refinement algorithms perform in a similar fashion, since the mesh is not automatically aligned with the layers present in the underlying solution.

5.3.2. Meanflow value evaluation. We now suppose that the aim of the computation is to calculate the value of the (weighted) mean value of u over Ω, *i.e.*,

$$J(u) = \int_\Omega u\psi \,\mathrm{dx}, \quad \psi = 1 + \tanh\left(100\left[-\left(x-\frac{1}{2}\right)^2 - \left(y-\frac{3}{4}\right)^2 + \frac{1}{64}\right]\right).$$

The (approximate) true value of the functional is given by $J(u) = 0.036920059604442$. Here, we again consider the use of the isotropic and anisotropic refinement strategies considered in the previous section based on employing a uniform and a pre–processed moved mesh. Firstly, we show the computed meshes obtained at the

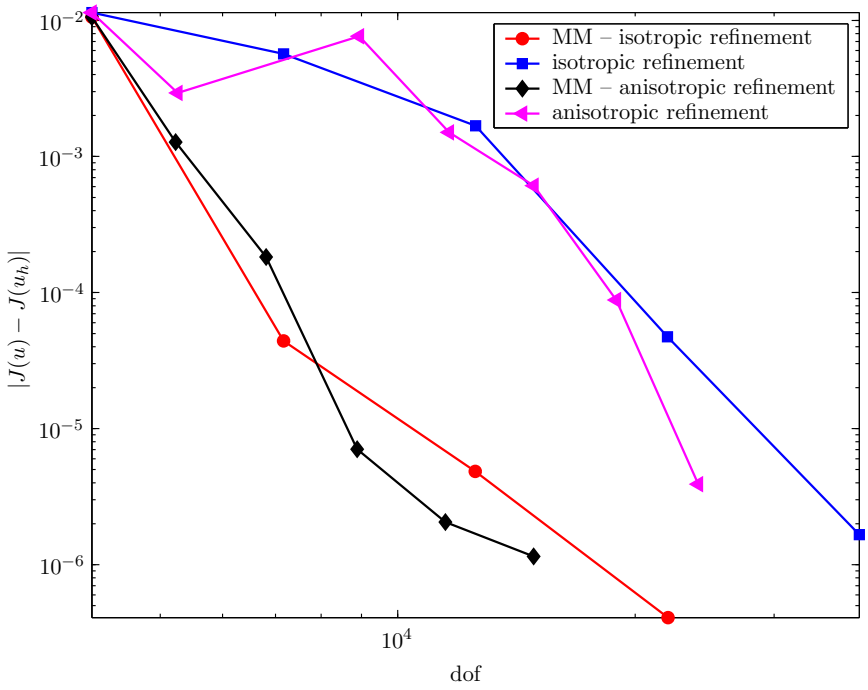


FIGURE 15. Example 3. Point value evaluation. The computed error $|J(u) - J(u_h)|$ versus the total number of degrees of freedom (loglog scale). Comparison between the errors obtained with and without the pre–processing moving mesh strategy.

first three steps of refinement; the results reported in Figure 16 (left) refers to the meshes computed by employing an isotropic mesh refinement strategy coupled with the pre–processing moving mesh technique; whereas the corresponding ones obtained without the moving mesh strategy are shown in Figure 16 (right). The analogous results obtained with the anisotropic refinement strategy are shown in Figure 17. The errors in the computed target functional $|J(u) - J(u_h)|$ for all the cases considered are reported in Figure 18 (loglog scale). As in the previous example, we clearly observe the advantages of employing the moving mesh strategy.

6. Conclusions

In this article, we have developed a mesh movement algorithm based on employing harmonic maps between a logical computational domain, and the corresponding physical domain of interest. This approach has been employed primarily as a pre–processing step within an adaptive h–refinement algorithm. Indeed, our numerical experiments clearly indicate that if an initial coarse mesh is first pre–processed using the moving mesh algorithm, then subsequent adaptive refinement can lead to significant efficiency gains in comparison with the corresponding adaptive algorithm applied to an initially uniform mesh. The performance of the proposed pre–processing mesh movement algorithm has been demonstrated in conjunction with the discontinuous Galerkin finite element method for the numerical approximation

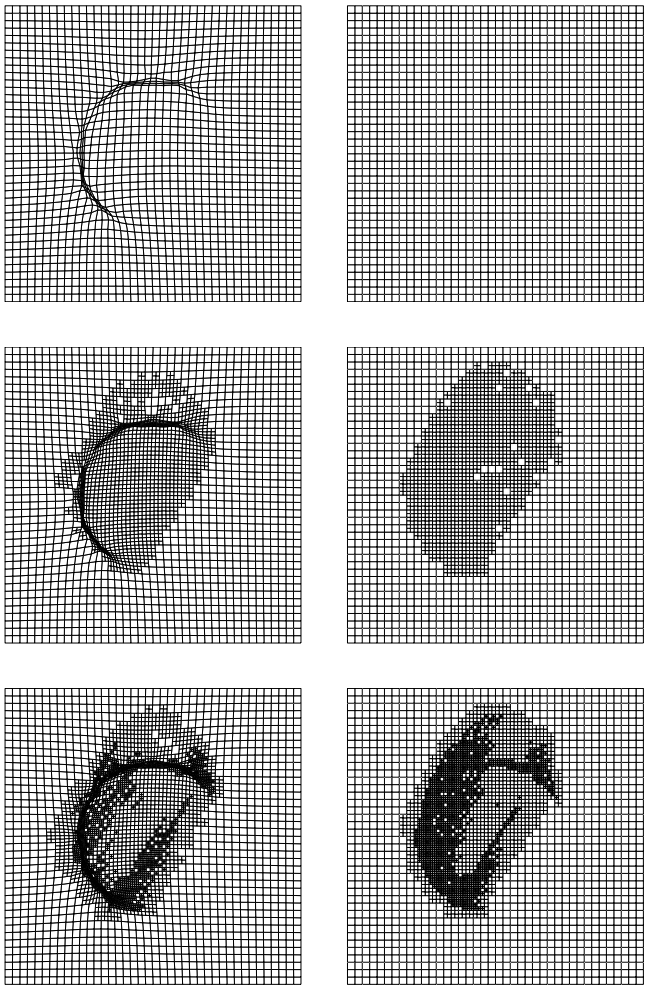


FIGURE 16. Example 3. Meanflow value evaluation. First three steps of the isotropic adaptively refined meshes with (left) and without (right) the pre–processing moving mesh strategy.

of advection–diffusion–reaction problems. Future work will be devoted to the application of the proposed adaptive algorithm to both two– and three–dimensional compressible fluid flows.

References

[1] M. Ainsworth and J. T. Oden. A posteriori error estimation in finite element analysis. *Comput. Methods Appl. Mech. Engrg.*, 142(1-2):1–88, 1997.

[2] D. N. Arnold. An interior penalty finite element method with discontinuous elements. *SIAM J. Numer. Anal.*, 19(4):742–760, 1982.

[3] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM J. Numer. Anal.*, 39(5):1749–1779 (electronic), 2001/02.

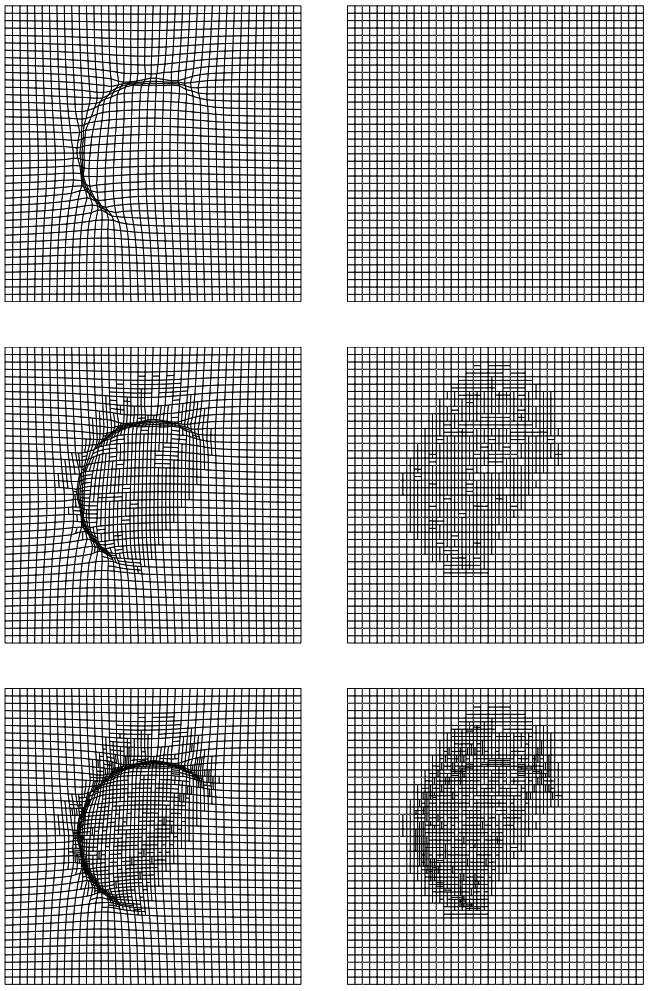


FIGURE 17. Example 3. Meanflow value evaluation. First three steps of the anisotropic adaptively refined meshes with (left) and without (right) the pre–processing moving mesh strategy.

[4] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. *Acta Numer.*, 10:1–102, 2001.
[5] F. Brezzi, L. D. Marini, and E. Süli. Discontinuous Galerkin methods for first-order hyperbolic problems. *Math. Models Methods Appl. Sci.*, 14(12):1893–1903, 2004.
[6] W. Cao, W. Huang, and R. D. Russell. A study of monitor functions for two-dimensional adaptive mesh generation. *SIAM J. Sci. Comput.*, 20(6):1978–1994 (electronic), 1999.
[7] A. S. Dvinsky. Adaptive grid generation from harmonic maps on riemannian manifolds. *J. Comput. Phys.*, 95:45–476, 1991.
[8] T. Eibner and J. M. Melenk. An adaptive strategy for *hp*-FEM based on testing for analyticity. *Comput. Mech.*, 39(5):575–595, 2007.
[9] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for differential equations. In *Acta numerica, 1995*, Acta Numer., pages 105–158. Cambridge Univ. Press, Cambridge, 1995.

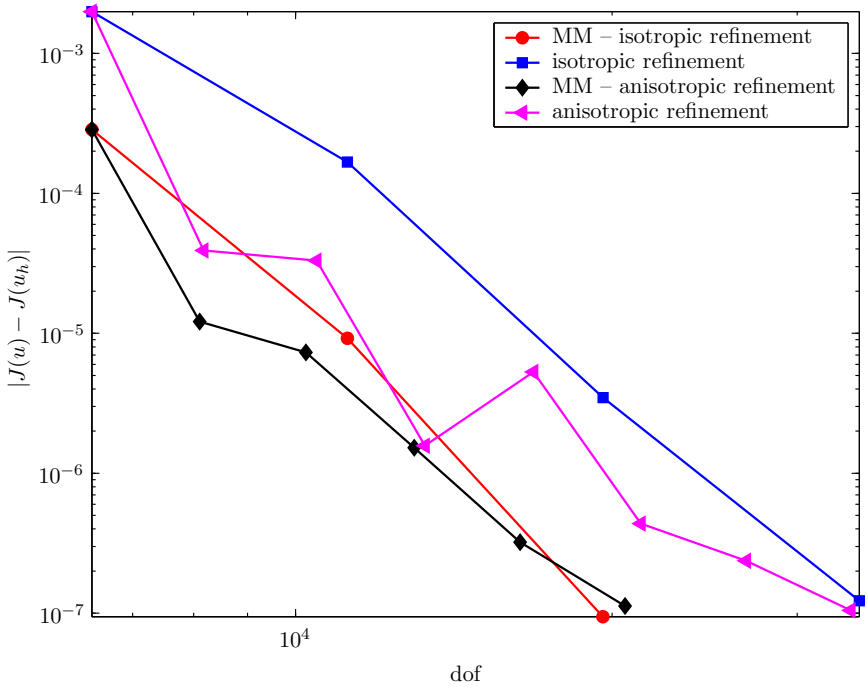


FIGURE 18. Example 3. Meanflow value evaluation. The computed error $|J(u) - J(u_h)|$ versus the total number of degrees of freedom (loglog scale). Comparison between the errors obtained with and without the pre–processing moving mesh strategy.

[10] E. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods for advection-diffusion-reaction problems on anisotropically refined meshes. *SIAM J. Sci. Comput.* (in press).

[11] E. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods on *hp*–anisotropic meshes I: A priori error analysis. *Int. J. Comp. Sci. Math.* (in press).

[12] E. Georgoulis, E. Hall, and P. Houston. Discontinuous Galerkin methods on *hp*–anisotropic meshes II: A posteriori error analysis and adaptivity. Technical report, University of Leicester Mathematics Research Report MA-07-010, 2007. Submitted for publication.

[13] R. S. Hamilton. *Harmonic maps of manifolds with boundary.* Springer-Verlag, Berlin, 1975. Lecture Notes in Mathematics, Vol. 471.

[14] K. Harriman, D. Gavaghan, and E. Süli. The importance of adjoint consistency in the approximation of linear functionals using the discontinuous Galerkin finite element method. Technical report, NA-04/18, Oxford University Computing Laboratory, 2004.

[15] P. Houston, C. Schwab, and E. Süli. Discontinuous *hp*-finite element methods for advection-diffusion-reaction problems. *SIAM J. Numer. Anal.*, 39(6):2133–2163, 2002.

[16] P. Houston and E. Süli. *hp*-adaptive discontinuous Galerkin finite element methods for first-order hyperbolic problems. *SIAM J. Sci. Comput.*, 23(4):1226–1252, 2001.

[17] P. Houston and E. Süli. Stabilized *hp*-finite element approximation of partial differential equations with non-negative characteristic form. *Computing*, 66:99–119, 2001.

[18] 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.

[19] W. Huang. Anisotropic mesh adaptation and movement. Lecture Notes for the workshop *Adaptive Methods, Theory and Application*, Peking University, Beijing, China, June-August 2005.

[20] J. Lang, W. Cao, W. Huang, and R. D. Russell. A two-dimensional moving finite element method with local refinement based on a posteriori error estimates. *Appl. Numer. Math.*, 46(1):75–94, 2003.

[21] R. Li, W. Liu, T. Tang, and P. Zhang. Moving mesh finite element methods based on harmonic maps. In *Scientific computing and applications (Kananaskis, AB, 2000)*, volume 7 of *Adv. Comput. Theory Pract.*, pages 143–156. Nova Sci. Publ., Huntington, NY, 2001.

[22] R. Li and T. Tang. Moving mesh discontinuous Galerkin method for hyperbolic conservation laws. *J. Sci. Comput.*, 27(1-3):347–363, 2006.

[23] R. Li, T. Tang, and P. Zhang. Moving mesh methods in multiple dimensions based on harmonic maps. *J. Comput. Phys.*, 170(2):562–588, 2001.

[24] W. Ren and X.-P. Wang. An iterative grid redistribution method for singular problems in multiple dimensions. *J. Comput. Phys.*, 159(2):246–273, 2000.

[25] C. Schwab. *p- and hp-finite element methods*. Numerical Mathematics and Scientific Computation. The Clarendon Press Oxford University Press, New York, 1998. Theory and applications in solid and fluid mechanics.

[26] E. Süli and P. Houston. Adaptive finite element approximation of hyperbolic problems. In *Error estimation and adaptive discretization methods in computational fluid dynamics*, volume 25 of *Lect. Notes Comput. Sci. Eng.*, pages 269–344. Springer, Berlin, 2003.

[27] E. Süli and P. Houston. Adaptive finite element approximation of hyperbolic problems. In *Error estimation and adaptive discretization methods in computational fluid dynamics*, volume 25 of *Lect. Notes Comput. Sci. Eng.*, pages 269–344. Springer, Berlin, 2003.

[28] E. Süli, P. Houston, and C. Schwab. *hp*-finite element methods for hyperbolic problems. In *The mathematics of finite elements and applications, X, MAFELAP 1999 (Uxbridge)*, pages 143–162. Elsevier, Oxford, 2000.

[29] T. Tang. Moving mesh methods for computational fluid dynamics. In *Recent advances in adaptive computation*, volume 383 of *Contemp. Math.*, pages 141–173. Amer. Math. Soc., Providence, RI, 2005.

[30] T. Tang and M. R. Trummer. Boundary layer resolving pseudospectral methods for singular perturbation problems. *SIAM J. Sci. Comput.*, 17(2):430–438, 1996.

Dipartimento di Matematica, Università degli Studi di Pavia, Via Ferrata 1, 27100 Pavia, Italy
E-mail: paola.antonietti@unipv.it
URL: http://www-dimat.unipv.it/antonietti/

School of Mathematical Sciences, University of Nottingham, University Park, Nottingham, NG7 2RD, UK.
E-mail: Paul.Houston@nottingham.ac.uk
URL: http://www.maths.nottingham.ac.uk/personal/ph/