Communications to SIMAI Congress,
ISSN 1827-9015, Vol. 3 (2009) 244 (12pp)

DOI: 10.1685/CSC09244

# An $hr$–adaptive discontinuous Galerkin method for advection–diffusion problems

Paola F. Antonietti[1], Paul Houston[2]

[1] *MOX–Dipartimento di Matematica "F. Brioschi",*
*Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, ITALY*
*E-mail: paola.antonietti@polimi.it*
*http://mox.polimi.it/∼antonietti*

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

## Abstract

We propose an adaptive mesh refinement strategy based on exploiting a combination of a pre–processing mesh re-distribution algorithm employing a harmonic mapping technique, and standard (isotropic) mesh subdivision for discontinuous Galerkin approximations of advection–diffusion problems. Numerical experiments indicate that the resulting adaptive strategy can efficiently reduce the computed discretization error by clustering the nodes in the computational mesh where the analytical solution undergoes rapid variation.

*Keywords:* `Discontinuous Galerkin methods, advection-diffusion problems, moving mesh methods.`

## 1. Introduction

In many areas of applied mathematics the development of effective and robust adaptive numerical methods is indeed a computational necessity. Adaptive finite element methods assess the quality of the computed solution based on exploiting computable *a posteriori* error bounds, and subsequently enrich the underlying finite element space during the solution process in order to deliver numerical approximations to a given underlying continuum problem to within a specified accuracy. Mesh adaptivity may employ a variety of techniques to enrich the underlying finite element space; these may be broadly classified as follows:

- local refinement and/or coarsening of the mesh ($h$–adaptivity);
- locally varying the polynomial degree of the basis ($p$–adaptivity);

*Received 10/02/2009, in final form 16/06/2009*
*Published 31/07/2009*

- combinations of the above ($hp$–adaptivity);
- mesh movement ($r$–adaptivity).

The idea behind $r$–adaptivity (moving mesh methods) is to *relocate* or *redistribute* the grid points of a mesh, keeping the number of nodes fixed without changing the mesh topology.

We point out that, while 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 [1,9,10], on the other hand, the state of development of so-called *optimal* mesh modification strategies is far less advanced [2,8,11]. In this article, we exploit the numerous advantages of the original $r$–refinement method based on exploiting harmonic maps to construct an optimal pre–processing algorithm, employed in conjunction with discontinuous Galerkin (DG) approximations of steady state advection–diffusion problems. The pre–processing mesh movement algorithm 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 adaptive $h$–refinement.

## 2. Model Problem and its DG Discretization

Given a bounded polyhedral domain $\Omega \subseteq \mathbb{R}^n$, $n = 2, 3$, $f \in L^2(\Omega)$, and $g \in H^{1/2}(\partial\Omega)$, we consider the following model problem:

$$-\varepsilon\Delta u + \nabla \cdot (\boldsymbol{\beta} u) = f \quad \text{in } \Omega, \qquad u = g \quad \text{on } \Gamma \equiv \partial\Omega, \tag{1}$$

where $\varepsilon > 0$ is the diffusion coefficient and $\boldsymbol{\beta} = \{\beta_i\}_{i=1}^n$ is a vector function whose entries $\beta_i$, $1 \le i \le n$, are Lipschitz continuous real–valued functions on $\bar{\Omega}$.

We consider regular partitions $\mathcal{T}_h$ of $\Omega$ of granularity $h > 0$, where each $K \in \mathcal{T}_h$ is the image of a fixed master element $\mathcal{K}$, *i.e.*, $K = F_K(\mathcal{K})$, where $\mathcal{K}$ is either the open unit $n$–simplex or the open unit $n$–hypercube in $\mathbb{R}^n$, $n = 2, 3$. We denote by $\mathcal{F}_h$ the set of all faces of $\mathcal{T}_h$, and by $\mathcal{F}_h^B$ the set of all faces that lie on the boundary. For a given approximation order $\ell \ge 1$, we define the DG finite element space $V_h = \{v \in L^2(\Omega) : v|_K \circ F_K \in \mathcal{M}^\ell(\mathcal{K}) \quad \forall K \in \mathcal{T}_h\}$, where $\mathcal{M}^\ell(\mathcal{K})$ is a suitable space of polynomials of degree at most $\ell$ on $\mathcal{K}$. We denote by $\nabla_h$ the elementwise application of the operator $\nabla$, and, for $v \in V_h$ and $K \in \mathcal{T}_h$, we write $v^+$ (respectively, $v^-$) to denote the interior (respectively, exterior) trace of $v$ defined on $\partial K$ (respectively, $\partial K \setminus \Gamma$). Given $K \in \mathcal{T}_h$, the inflow and outflow parts of $\partial K$ are defined as $\partial_- K := \{x \in \partial K : \boldsymbol{\beta}(x) \cdot \mathbf{n}_K(x) < 0\}$, and $\partial_+ K := \{x \in \partial K : \boldsymbol{\beta}(x) \cdot \mathbf{n}_K(x) \ge 0\}$, respectively, where $\boldsymbol{n}_K$ denotes the unit outward normal vector to $\partial K$.

For a parameter $\alpha \geq \alpha_{\min} > 0$ (at our disposal), adopting the standard notation $\{\!\{\cdot\}\!\}$ for the face-average and $[\![\cdot]\!]$ for the jump operator [3], we define the bilinear form $B_h(\cdot,\cdot) : V_h \times V_h \to \mathbb{R}$ as

$$
\begin{aligned}
B_h(u,v) &= \int_\Omega \varepsilon \nabla_h u \cdot \nabla_h v \,\mathrm{d}x - \sum_{F\in\mathcal{F}_h} \int_F \{\!\{\varepsilon \nabla_h u\}\!\} \cdot [\![v]\!] \,\mathrm{d}s \\
&- \sum_{F\in\mathcal{F}_h} \int_F [\![u]\!] \cdot \{\!\{\varepsilon \nabla_h v\}\!\} \,\mathrm{d}s + \sum_{F\in\mathcal{F}_h} \int_F \alpha\, \varepsilon\, h_F^{-1} [\![u]\!] \cdot [\![v]\!] \,\mathrm{d}s - \int_\Omega u\, \boldsymbol{\beta} \cdot \nabla_h v \,\mathrm{d}x \\
&+ \sum_{K\in\mathcal{T}_h} \int_{\partial_+ K} (\boldsymbol{\beta}\cdot \mathbf{n}_K)\, u^+ v^+ \,\mathrm{d}s + \sum_{K\in\mathcal{T}_h} \int_{\partial_- K\setminus\Gamma} (\boldsymbol{\beta}\cdot \mathbf{n}_K)\, u^- v^+ \,\mathrm{d}s.
\end{aligned}
$$

Then, the DG approximation of problem (1) reads as follows:

$$
\text{Find } u_h \in V_h \text{ such that } \quad B_h(u_h, v) = F_h(v) \quad \forall v \in V_h, \tag{2}
$$

where the functional $F_h(\cdot) : V_h \to \mathbb{R}$ is given by

$$
\begin{aligned}
F_h(v) &= \int_\Omega f v \,\mathrm{d}x + \sum_{F\in\mathcal{F}_h^B} \int_F \varepsilon\, g\, \nabla v^+ \cdot \mathbf{n}_K \,\mathrm{d}s \\
&+ \sum_{F\in\mathcal{F}_h^B} \int_F \varepsilon\, \alpha\, h_F^{-1} g\, v^+ \,\mathrm{d}s + \sum_{K\in\mathcal{T}_h} \int_{\partial_- K\cap\Gamma} (\boldsymbol{\beta}\cdot \mathbf{n}_K)\, g v^+ \,\mathrm{d}s.
\end{aligned}
$$

Here, $h_F$ is a local mesh size function associated with each face $F$ in $\mathcal{F}_h$, cf. [2] for details.

## 3. A Pre–processing Moving Mesh Method

In this section we present the pre–processing moving mesh method based on employing a harmonic mapping technique. We point out that the existence of a unique harmonic map operator is a key motivation for exploiting this approach within mesh movement algorithms, cf. [7,8,11], for example. The proposed method is implemented by employing a finite element approximation of the node locations in the computational mesh, within an iterative procedure. The moving mesh algorithm can be viewed as a black box routine added to the whole numerical solver for the approximation of the PDE under consideration; with this in mind, it is very convenient for coding since no modifications to the application solver for the PDE are required.

Let $\Omega$ and $\Omega_C$ (the latter being referred to as the logical domain) be compact Riemannian manifolds of dimension $n$ with associated metric tensors $d_{ij}$ and $r_{\nu\mu}$, respectively, defined in the local coordinate systems denoted by $\boldsymbol{x}$ and $\boldsymbol{\xi}$, respectively. Following [4], with a Euclidean metric on the logical domain $\Omega_C$, the Euler–Lagrange equations, whose solution minimize the energy functional, are given by

$$\frac{\partial}{\partial x^i}\left(G^{ij}\frac{\partial \xi^k}{\partial x^j}\right) = 0, \quad k = 1, \dots, n, \tag{3}$$

where we have set $G^{ij} = \sqrt{d}\, d^{ij}$, $d = \det(d_{ij})$, $d_{ij} = (d^{ij})^{-1}$, and the standard summation convention is assumed. The inverse of the matrix $G = \{G^{ij}\}_{i,j=1}^{n}$ is called the *monitor function* and plays a key role in the development of moving mesh algorithms. Solutions to (3) 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 $G$, or as is usually the case, the inverse of $G$, as a function of physical coordinates, and that minimizing the energy will result in a harmonic mapping with the desired properties. To solve the Euler–Lagrange equations (3) numerically, one usually interchanges dependent and independent variables. The solution of (3) 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.

Next, we describe the key points of the pre–processing algorithm (cf. Figure 1).

**Initialization.** We choose an initial conforming mesh $\Sigma_h^0$ in the logical domain $\Omega_C$, with nodes $\boldsymbol{\xi}^0 = \{\boldsymbol{\xi}_i^0\}$: the initial logical mesh is used as a reference grid only, and will be kept unchanged throughout the computation. We also choose an initial conforming mesh $\mathcal{T}_h^0$ of the physical domain $\Omega$ with nodes $\boldsymbol{x}^0 = \{\boldsymbol{x}_i^0\}$. We remark that, whenever the physical domain $\Omega$ is of regular shape, we can simply identify the logical domain with the physical one, *i.e.*, $\Omega_C \equiv \Omega$, and we can choose $\boldsymbol{\xi}^0 \equiv \boldsymbol{x}^0$.

**Monitor function.** We compute, once and for all, the solution $u_h$ of problem (2) on the initial physical mesh $\mathcal{T}_h^0$. Additionally, given that the monitor function is typically a function of $u_h$ only, the corresponding inverse matrix may be computed elementwise, which thereby defines $G|_K$ for all $K$ in the physical mesh.

**Relocation of the grid points.** At the generic step of the iterative pro-

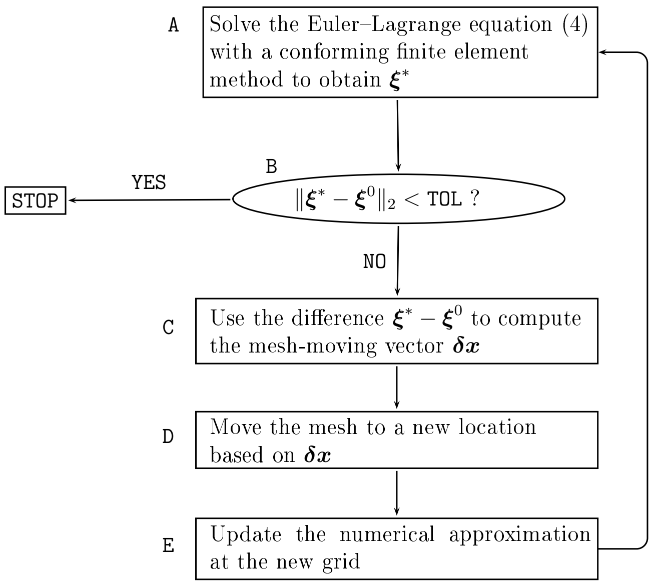

A
Solve the Euler–Lagrange equation (4) with a conforming finite element method to obtain $\boldsymbol{\xi}^*$
B
$\|\boldsymbol{\xi}^* - \boldsymbol{\xi}^0\|_2 <$ TOL ?
YES
STOP
NO
C
Use the difference $\boldsymbol{\xi}^* - \boldsymbol{\xi}^0$ to compute the mesh-moving vector $\boldsymbol{\delta x}$
D
Move the mesh to a new location based on $\boldsymbol{\delta x}$
E
Update the numerical approximation at the new grid

Fig. 1. The pre-processing moving mesh algorithm.

cedure, we first approximate with a conforming finite element method the following generalized Poisson problems subject to appropriate Dirichlet boundary conditions:

$$\frac{\partial}{\partial x^i}\left(G^{ij}\frac{\partial \xi^k}{\partial x^j}\right) = 0, \quad \boldsymbol{\xi}|_{\partial\Omega} = \boldsymbol{\xi}_b, \quad k = 1, \ldots, n. \tag{4}$$

By doing so we obtain a new mesh $\Sigma_h^*$ 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 $\Omega$, is defined as $\boldsymbol{\delta\xi} = \boldsymbol{\xi}^0 - \boldsymbol{\xi}^*$. If the norm of computed error $\|\boldsymbol{\delta\xi}\|_{\ell^2}$ is not smaller than a preassigned tolerance `TOL`, we use the computed error to move the nodes $\boldsymbol{x} = \{\boldsymbol{x}_i\}$ of the current mesh $\mathcal{T}_h$ in the physical space to new nodes $\boldsymbol{x}^* = \{\boldsymbol{x}_i^*\}$ and obtain a new grid $\mathcal{T}_h^*$. The nodal values of the new grid are computed based on the assumption that the surface of $u_h$ on $\Omega$ will not change when the nodes of the mesh are moved to new locations. We next describe in detail how we relocate the nodes in the physical space. For any node $\boldsymbol{x}_i$ of the current physical mesh $\mathcal{T}_h$, let $\mathcal{N}_i$ be the patch of elements in $\mathcal{T}_h$ surrounding $\boldsymbol{x}_i$. For every element $K$ in $\mathcal{N}_i$ we evaluate at the vertex $\boldsymbol{x}_i$ the Jacobi matrix $\mathtt{B}$ of the transformation between the initial coordinates $\boldsymbol{x}^0$ in the computational domain and the current coordinates $\boldsymbol{x}$. We remark that whenever the mesh is made by $n$–simplices the Jacobian is constant on $K$. Analogously, we denote by $\mathtt{B}^{\mathtt{H}}$ the evaluation at the corresponding node $\boldsymbol{\xi}_i$ in the logical mesh of the Jacobi matrix of the transformation between the initial coordinates of the logical domain $\boldsymbol{\xi}^0$ and the current coordinates $\boldsymbol{\xi}^*$. We next solve the following linear system

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

whose solution gives $\partial \boldsymbol{x}/\partial \boldsymbol{\xi}$ in $K$. 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 patch of elements in $\mathcal{T}_h$ surrounding the node $\boldsymbol{x}_i$. Then, the location of the nodes in the new mesh on $\Omega$ is taken as $\boldsymbol{x}_i^* = \boldsymbol{x}_i + \tau_i \, \boldsymbol{\delta x}_i$, where $\tau_i$ is the *length of the movement* which is defined by

$$\boldsymbol{\tau}_i = \frac{1}{2} \min_{K \in \mathcal{N}_i} \left\{ \frac{|K|}{\max_{\substack{F \in \mathcal{F}_h \\ F \subset \partial K}} |F|} \right\}.$$

## 4. Numerical Experiments

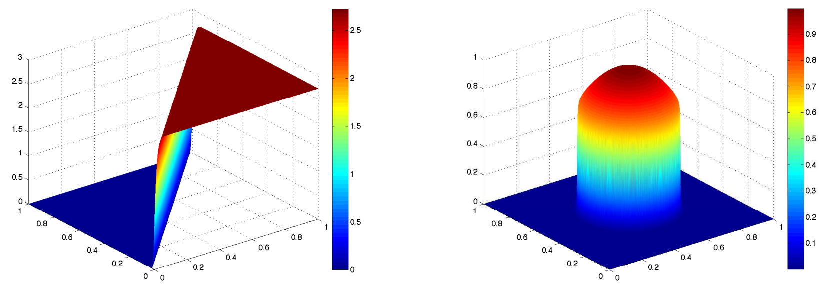
In this section we report some numerical experiments indicating that our pre–processing technique is indeed efficient and robust. Throughout this section we have set $\mathtt{TOL} = 10^{-2}$ (cf. Figure 1), and $\alpha = 10$ (cf. Section 2). In all the computations, we have set $\Omega = (0,1)^2$, and have chosen an analytical solution of problem (1) and adjusted the source term $f(x,y)$ and the non-homogeneous Dirichlet boundary conditions accordingly. The monitor function has been chosen as

$$(G|_K)^{-1} = \sqrt{\overline{\eta} + \eta_K} \, I, \tag{5}$$

where $\eta_K$ is a suitable error indicator whose definition will be specified in the following, and the average error $\overline{\eta}$ is given by

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

where $\#K$ denotes the number of elements in the mesh (cf. [7,8,11]).



(a) $\varepsilon = 10^{-2}$. (b) $\varepsilon = 10^{-6}$.

Fig. 2. Elevation plots of the analytical solutions for the analytical solutions given in (6) and (7), respectively.

In the first example, we let $\varepsilon \in \mathbb{R}^+$ and $\boldsymbol{\beta} = [1,1]^\top$. We choose the analytical solution of problem (1) to be the function

$$u(x,y) = \exp\left(\frac{1-\exp(-\frac{x-y}{\varepsilon})}{1-\exp(-\frac{1}{\varepsilon})}\right). \tag{6}$$

For $\varepsilon \searrow 0$, the analytical solution exhibits a sharp internal layer along the straight line $y = x$. We choose $\varepsilon = 10^{-2}$ (cf. Figure 2(a) where the elevation plot of the analytical solution is shown). We first assess the improvement in the computed error when the mesh is first pre–processed using the mesh movement algorithm. To this end, we select the error indicator appearing in (5) as follows: $\eta_K = |u - u_h|^2_{1,K}$ for all $K \in \mathcal{T}_h$. Here, we consider simply global uniform refinement of the initial meshes, both with and without the pre–processing mesh movement step (cf. Figure 3(a) and Figure 3(b), respectively). At each step of refinement, $\texttt{step} = 0,1,2,3$, 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 DG finite element approximation of the analytical solution $u$ computed using (discontinuous) piecewise linear polynomials.

In Table 1 we report the computed errors in the $L^2$–norm. The results in 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`). Here, we clearly observe that the pre–processing moving mesh algorithm reduces the computed error by a factor of around 3.

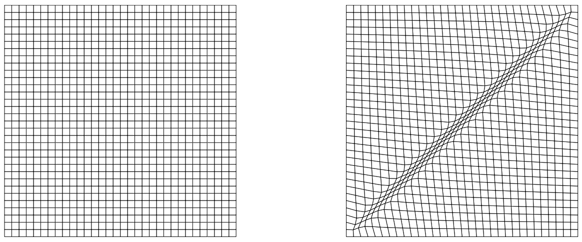


Fig. 3. (a) Initial quadrilateral mesh; (b) Quadrilateral mesh obtained with the pre–processing moving mesh method.

| 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 |

In the second example, we let $\varepsilon \in \mathbb{R}^+$ and $\boldsymbol{\beta} = [2,3]^\top$. The analytical solution is given by

$$u(x,y) = 16(x-x^2)(y-y^2)\left(\frac{1}{2} + \frac{\arctan(2\sqrt{\varepsilon^{-1}}\left[1/16-(x-1/2)^2-(y-1/2)^2\right])}{\pi}\right), \tag{7}$$

and, for $\varepsilon \searrow 0$, it exhibits a sharp circular internal layer. Here, we choose $\varepsilon = 10^{-6}$ (cf. Figure 2(b)). Firstly, we take again $\eta_K = |u-u_h|^2_{1,K}$ for all $K \in \mathcal{T}_h$. The initial triangular and quadrilateral meshes obtained with our pre–processing moving mesh method are reported in Figure 4. 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.

Next, we choose the monitor function based on an *a posteriori* error estimator. We suppose that the quantity of interest is the (weighted) mean value of $u$ over $\Omega$, *i.e.*,

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

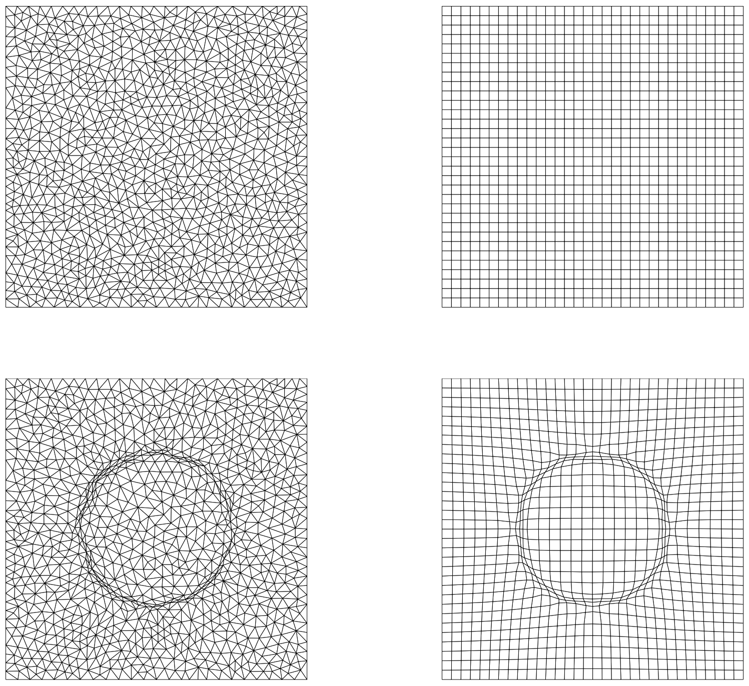


Fig. 4. Initial triangular and Cartesian meshes (top); meshes obtained with the pre–processing moving mesh method (bottom).

The (approximate) true value of the target functional is given by $J(u) = 0.036920059604442$. From [6] we have the following dual-weighted-residual error estimate

$$|J(u) - J(u_h)| \leq \sum_{K \in \mathcal{T}_h} |\eta_K| \,, \tag{8}$$

where the analytic expression for $\eta_K$ may be found in [5, Eq. 5.4], for example. Thereby, we select the monitor function based on employing the elementwise error indicators present in the *a posteriori* error bound stated in (8). Furthermore, for a user-defined tolerance `tol`, we now consider the problem of designing an appropriate finite element mesh $\mathcal{T}_h$ such that $|J(u) - J(u_h)| \leq$ `tol`, subject to the constraint that the total number of elements in $\mathcal{T}_h$ is minimized. To this end, we exploit the fixed fraction mesh refinement algorithm, with refinement and derefinement fractions set to 25% and 10%, respectively. Within this adaptive strategy, elements which have been flagged for refinement are subdivided by employing an isotropic

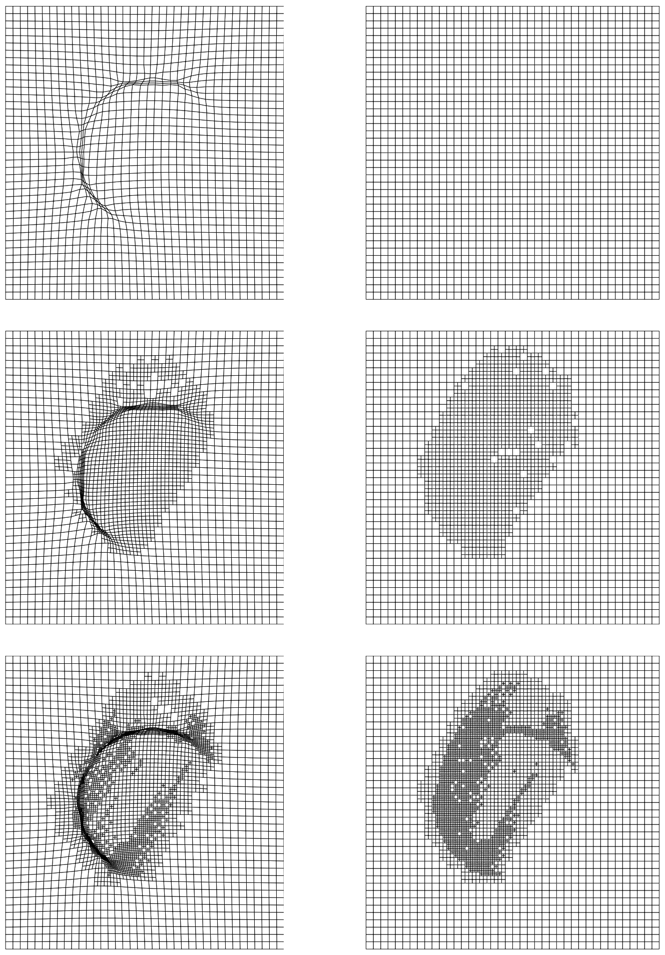


Fig. 5. First three steps of the isotropic adaptively refined meshes with (left) and without (right) the pre–processing moving mesh strategy.

refinement technique. Firstly, we show the computed meshes obtained at the first three steps of refinement (Figure 5, left) by employing an isotropic mesh refinement strategy coupled with the pre–processing moving mesh technique; the analogous ones obtained without the moving mesh strategy are shown in Figure 5 (right). The errors in the computed target functional $|J(u) - J(u_h)|$ are reported in Figure 6 (loglog scale). As in the previous example, here we clearly observe the superiority of employing the moving mesh strategy to pre-process the initial computational grid, before undertaking subsequent adaptive $h$–refinement.

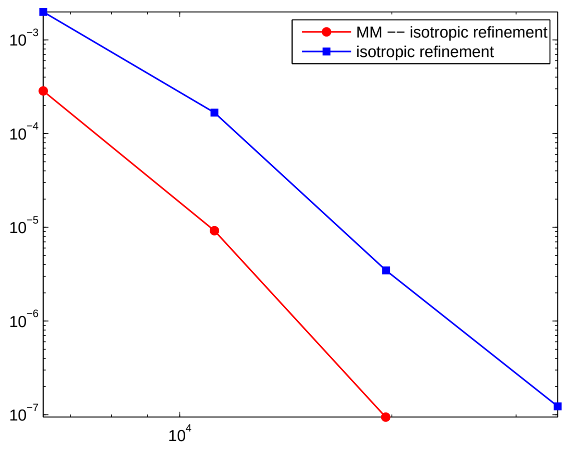


Fig. 6. The computed error $|J(u) - J(u_h)|$ versus the total number of degrees of freedom (loglog scale). Comparison between the errors obtained with and without the pre–processing moving mesh strategy.

## Acknowledgments

The work has been carried out while the first author was a research fellow at the University of Nottingham, financed by the European Union, under the `ADIGMA` project.

## 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. P. F. Antonietti and P. Houston. A pre-processing moving mesh method for discontinuous Galerkin approximations of advection-diffusion-reaction problems. *Int. J. Numer. Anal. Model.*, 5(4):704–728, 2008.
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.
4. A. S. Dvinsky. Adaptive grid generation from harmonic maps on riemannian manifolds. *J. Comput. Phys.*, 95:45–476, 1991.
5. 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.

6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. 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.