



# A Painless Automatic hp-Adaptive Strategy for Elliptic Problems

Vincent Darrigrand, David Pardo, Théophile Chaumont-Frelet, Ignacio Gómez-Revuelto, Emilio Luis Garcia-Castillo

## ► To cite this version:

Vincent Darrigrand, David Pardo, Théophile Chaumont-Frelet, Ignacio Gómez-Revuelto, Emilio Luis Garcia-Castillo. A Painless Automatic hp-Adaptive Strategy for Elliptic Problems. *Finite Elements in Analysis and Design*, 2020, 178, pp.103424. 10.1016/j.finel.2020.103424 . hal-02071427v2

**HAL Id: hal-02071427**

**<https://inria.hal.science/hal-02071427v2>**

Submitted on 29 Jan 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# A Painless Automatic $hp$ -Adaptive Strategy for Elliptic Problems<sup>\*</sup>

Vincent Darrigrand<sup>a,b,\*</sup>, David Pardo<sup>b,c,d</sup>, Théophile Chaumont-Frelet<sup>e,f</sup>, Ignacio Gómez-Revuelto<sup>g</sup>, Luis Emilio Garcia-Castillo<sup>g</sup>

<sup>a</sup>*Cerfacs, Toulouse, France*

<sup>b</sup>*University of the Basque Country (UPV-EHU), Leioa, Spain*

<sup>c</sup>*Basque Center for Applied Mathematics (BCAM), Bilbao, Spain*

<sup>d</sup>*Ikerbasque, Bilbao, Spain*

<sup>e</sup>*INRIA Sophia Antipolis, Méditerranée Research Centre, Nachos Project Team, France*

<sup>f</sup>*University of Nice, J. A. Dieudonné Mathematics Laboratory (UMR CNRS 6621), Nice, France*

<sup>g</sup>*Departamento de Teoría de la Señal y Comunicaciones. Universidad Carlos III de Madrid, Leganes (Madrid), Spain.*

---

## Abstract

In this work, we introduce a novel  $hp$ -adaptive strategy. The main goal is to minimize the complexity and implementational efforts hence increasing the robustness of the algorithm while keeping close to optimal numerical results. We employ a multi-level hierarchical data structure imposing Dirichlet nodes to manage the so-called hanging nodes. The  $hp$ -adaptive strategy is based on performing quasi-optimal unrefinements. Taking advantage of the hierarchical structure of the basis functions both in terms of the element size  $h$  and the polynomial order of approximation  $p$ , we mark those with the lowest contributions to the energy of the solution and remove them. This straightforward unrefinement strategy does not need from a fine grid or complex data structures, making the algorithm flexible to many practical situations and existing implementations. On the other side, we also identify some limitations of the proposed strategy, namely: (a) data structures only support isotropic  $h$ -refinements (although  $p$ -anisotropic refinements are enabled), (b) we assume certain quasi-orthogonality properties of the basis functions in the energy norm, and (c) in this work, we restrict to symmetric and positive definite problems. We illustrate these and other advantages and limitations of the proposed  $hp$ -adaptive strategy with several one-, two- and three-dimensional Poisson examples.

*Keywords:*  $hp$ -Adaptivity, Unrefinements, Elliptic problems, Multi-level

---

## 1. Introduction

Most engineering applications require a grid and an approximation space that accurately captures the most salient features of the solution and satisfies reasonable computational cost constraints. Such features may include boundary layers or singularities, which could be reproduced with small grid elements ( $h$ -refinements), and smooth solution areas that can be superbly approximated with high-order elements ( $p$ -refinements). Often, both  $h$  and  $p$  refinements need to be combined within the same problem. When no

---

<sup>\*</sup>The first two authors are supported by Projects of the Spanish Ministry of Economy and Competitiveness with reference MTM2016-76329-R (AEI/FEDER, EU), MTM2016-81697-ERC and the Basque Government Consolidated Research Group Grant IT649-13 on “Mathematical Modeling, Simulation, and Industrial Applications (M2SI)”, the BCAM “Severo Ochoa” accreditation of excellence SEV-2017-0718, and the Basque Government through the BERC 2018-2021 program and the European Union’s Horizon 2020, research and innovation program under the Marie Skłodowska-Curie grant agreement N° 777778. Last two authors have been partially supported by the Spanish Government through TEC2016-80386-P

\*Corresponding author

*Email addresses:* `vincent.darrigrand@gmail.com` (Vincent Darrigrand), `dzubiaur@gmail.com` (David Pardo), `theophile.chaumont@inria.fr` (Théophile Chaumont-Frelet), `igomez@diac.upm.es` (Ignacio Gómez-Revuelto), `legcasti@ing.uc3m.es` (Luis Emilio Garcia-Castillo)

information on the solution is known *a priori*, it is essential to have at our disposal automatic adaptive algorithms.

Refinement algorithms that simultaneously adapt element sizes  $h$  and polynomial orders of approximation  $p$  throughout the grid are known as  $hp$ -adaptive algorithms. When properly designed and implemented, they deliver exponential convergence rates (see [1, 2] for theoretical results on the subject, and, e.g., [3, 4] for numerical results along with an open-source software). Despite the great convergence properties exhibited by  $hp$ -adaptive algorithms, their industrial use is still somewhat limited, and only a handful of companies have adopted them for their daily computations. We believe this limited use is due to the high complexity exhibited by most existing  $hp$ -adaptive algorithms. Indeed, these algorithms frequently lead to overly complicated pieces of software whose debugging and upgrading requires from dedicated and highly-trained experts on the field. Moreover, they sometimes lack robustness probably due to the high-complexity of the adaptive algorithm and its associated data structures.

We classify existing  $hp$ -adaptive algorithms into two categories: those based on generating an entirely new mesh (re-meshing), and those that merely refine an existing mesh. The first option, depicted in Figure 1, and proposed, for instance, by Schoberl et. al. [5], needs from an efficient mesh generator, thus redirecting the implementation efforts to the mesh generator rather than to the adaptive data structures. The second option, based on refining an existing grid, requires complicated data structures to ensure the global continuity requirements of the solution, typically via the use of hanging nodes. To limit the implementation complexity, especially in higher dimensions, most authors restrict their algorithm to the *1-irregularity* rule (see, e.g., [3, 6]) that allows for at most one level of hanging nodes, as illustrated in Figure 2. Some exception to the 1-irregularity rule can be found in [7, 8].

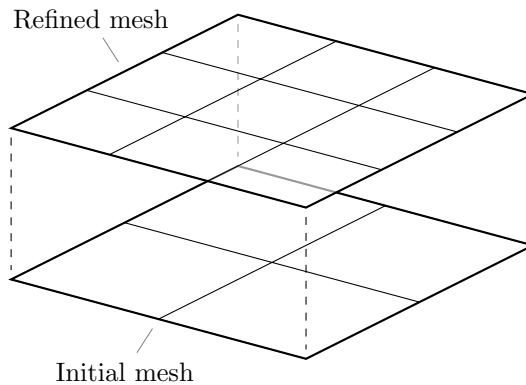


Figure 1: Re-meshing without hierarchy.

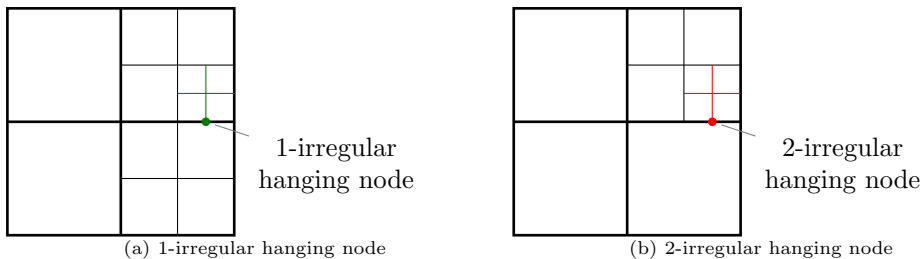


Figure 2: Meshes illustrating the 1-irregularity rule (panel a) and higher-irregularity rules (panel b).

Since 2015, Zander et al. proposed new data structures for supporting  $hp$ -discretizations [9–11]. Using a multi-level approach and hierarchical basis functions both in  $h$  and in  $p$ , they perform global uniform refine-

ments and impose homogeneous Dirichlet conditions (i.e., they remove the corresponding basis functions) at those nodes that need not be refined.

The design of those multi-level meshes with wide use of Dirichlet nodes overcome the hanging node difficulties *per se*: it ensures that the basis functions of the refined patch vanishes at its boundaries, see Figure 3. This method highly simplifies previously existing data structures for supporting *hp*-adaptive refinements. Moreover, it is possible to *easily* implement it in most of the existing Finite Element codes by simply generating global refinement trees and properly handling Dirichlet nodes. However, it also poses some limitations, namely: (a) anisotropic *h*-refinements are unsupported, (b) integration costs increase with the number of levels of *h*-refinements due to the hierarchical nature of *h*-basis functions, and (c) the design of scalable parallelization schemes may be more challenging, although some works on this area already exist (see, e.g., [12]). We can also encounter other multi-level implementations that support anisotropic *h*-refinements (see [13]) at the expense of considering slightly more complex data structures.

In this work, and driven by our “simple-to-implement” goal, we adopt the data structures proposed by Zander et al. [9–11], and we focus on developing an automatic and relatively simple *hp*-adaptive algorithm under these new data structures.

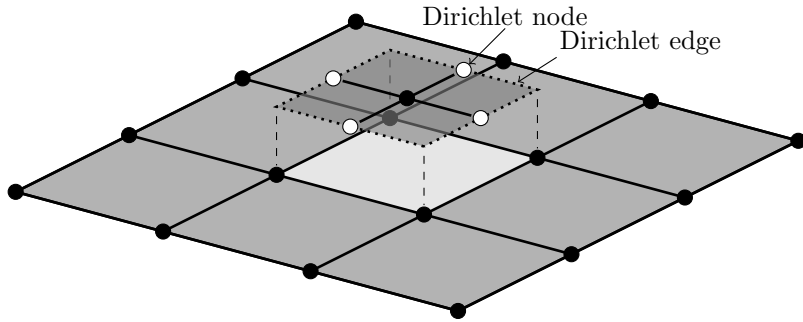


Figure 3: Multi-level 2D mesh without constraints on hanging nodes using Dirichlet nodes, see [9] for details.

There exists a variety of refinement-based *hp*-adaptive algorithms. Amongst others, we shall mention the works of: (a) Ainsworth et al. [14], which is simple to implement but it is only designed for isotropic refinements in *h* and *p* and its suitability for industrial applications is unclear; (b) the *Texas 3 Step strategy* [15] that performs first an *h*-adaptive step followed by a *p*-adaptive one that leads to non-optimal results; (c) the work of Demkowicz et al. presented in [3, 4, 16] and applied in several contexts, e.g. [17–28], that produces almost optimal meshes but needs from solving the problem over a globally refined  $(\frac{h}{2}, p+1)$ -grid, which is often prohibitively expensive, and also requires a sophisticated implementation; (d) the work of Houston et al. [29], which estimates the regularity of the solution with the Legendre coefficients [30] and, (e) the contribution of Zander et al. [31] and applications [32–35], which combine their multi-level data structure [9, 10] with a classic residual-based estimator [36]. We refer to [30] for a recent (Oct. 2014) review and comparison of some of the existing methods in terms of computational time versus the number of degrees of freedom (dofs). Note that the implementational effort is not accounted for in that survey. Finally, we also mention the family of *hp*-adaptive Discontinuous Galerkin (DG) methods, [37–45], which require from a specific implementation that is not always easily transferable to commercial FEM codes.

Inheriting some of the existing *hp*-adaptive algorithms is simply unfeasible with the new data structures employed by Zander et al. [31]. For example, the local projections considered by Demkowicz [3] in the context of hierarchical *h*-basis functions considered in our work would lead to an inefficient and overly complex implementation. On the other side, we may exploit the hierarchical structure of the basis functions to our advantage and design new *hp*-adaptive algorithms that could not be implemented with previous data structures. This is precisely the path we follow in this work.

Our main contribution is an easy-to-implement *hp*-adaptive strategy. The main idea is that given a mesh, we perform close to optimal unrefinements by taking advantage of the hierarchical data structures proposed by Zander et al. [9, 10]. We employ quadrilateral elements and select some of the basis functions that can

be easily removed to perform an unrefinement step (i.e., coarsening a given mesh similarly to [46, 47]). The use of optimal unrefinements rather than refinements delivers a unique advantage: the algorithm is capable of “correcting” (removing) undesired unknowns introduced during the pre-asymptotic regime. An unrefinement scheme is natural to implement with the hierarchical multi-level data structures employed here, but it would become rather challenging to use in combination with other data structures such as those proposed by Demkowicz [3]. For instance, the imposition of the 1-irregularity rule imposes important difficulties when implementing arbitrary unrefinements.

Let us devise an analogy of our proposed *hp*-adaptive algorithm in terms of workload in a business company. The multi-level basis function structure might be compared to a pyramidal hierarchy of workers and supervisors. The lowest level workers being the bubble basis functions (in charge of only one element) and the supervisors being the linear basis functions (in charge of several elements). The *hp*-algorithm abides by the following guidelines: (a) hire arbitrarily, either via a global *h*, *p* or *hp*-refinements or any other type of refinement selected by the user, see Appendix B, (b) evaluate the contribution of the last level workers, which correspond to those basis functions that can be removed without losing completeness, and (c) fire the ineffective workers: (i) either fire those lazy workers (local *p*-unrefinement), (ii) or, if all the last level workers are inefficient, fire the supervisor instead (local *h*-unrefinement). To determine the contribution of a basis function, we consider the difference in energy of the solution with and without the contribution of that basis function. In the context of *h*-adaptivity, such approach relates to the  $(h - h/2)$ -type error estimators, whose convergence has been proven in [48].

In this work, we focus on elliptic problems. We employ 1D, 2D and 3D numerical results to illustrate the observed convergence behaviour. However, we have no convergence proof of the proposed *hp*-adaptive algorithm. Due to the complexity of *hp*-adaptive algorithms, the lack of convergence proofs is customary in the field (see e.g., [3, 49]), although some convergence proofs exist for certain algorithms [47, 50–53]. To simplify the exposition, in the numerical results we consider Poisson problems with homogeneous Dirichlet and Neumann boundary conditions. The computational domain is the unit square (of dimension 1 or 2) or a scaled version of it unless specified (L-shaped domain). The load vector is selected so that the exact solution fits a specific manufactured solution.

The paper is organized as follows: Section 2 describes the data structures as well as the available mesh operations. Section 3 sets the definition of the *removable* basis function that is a key concept of our method. Section 4 defines the indicators we use to determine which basis functions may be removed (unrefined). The unrefinement algorithm is described in Section 5 as well as the algorithm performing the global refinements. Implementation details are exposed in Section 6. Section 7 numerically illustrates our method when applied to 1D, 2D and 3D examples. Finally, we draw some conclusions in Section 8. In Appendix A, we display additional 2D examples built from the one-directional solutions used in the 1D models. In Appendix B, we consider different refinement strategies illustrating the flexibility of the presented method when it comes to perform arbitrary refinements. Such alternative refinement strategies may be used to minimize the overall cost of the *hp*-adaptive algorithm.

## 2. Definitions

### 2.1. Abstract variational formulation

Given a domain  $\Omega \subset \mathbb{R}^d$ , where  $d=1, 2$ , or  $3$  is the spatial dimension, let  $\mathbb{H}(\Omega)$  be a Hilbert functional space. We denote  $\|\cdot\|_{\mathbb{H}(\Omega)}$  the norm of  $\mathbb{H}(\Omega)$ . Our problem is expressed in abstract variational form as:

$$\left| \begin{array}{l} \text{Find } u \in \mathbb{H} \text{ such that} \\ \\ b(u, \phi) = f(\phi) \quad \forall \phi \in \mathbb{H}, \end{array} \right. \quad (1)$$

where  $f$  is a linear continuous form on  $\mathbb{H}$  and  $b$  a bilinear continuous, symmetric and elliptic form such that problem (1) admits a unique solution in  $\mathbb{H}$ . Note that for simplicity of the presentation, we restrict ourselves to homogeneous Dirichlet boundary conditions that are directly incorporated into subspace  $\mathbb{H}$ .

## 2.2. Discretization

Let  $\mathcal{T}$  be a partition of  $\Omega$  into open active elements  $K_a$  such that  $\bar{\Omega} = \bigcup_{K \in \mathcal{T}} \bar{K}_a$ .  $\mathbb{H}_{\mathcal{T}}$  denotes a conforming finite element subspace of  $\mathbb{H}$  associated with partition  $\mathcal{T}$ . The finite element solution of (1) on  $\mathbb{H}_{\mathcal{T}}$  is denoted by  $u_{\mathcal{T}}$ .  $|\mathcal{T}|$  stands for the cardinal of  $\mathcal{T}$ .

**Basis functions:** We restrict to quadrilateral/hexahedral elements. A basis function of  $\mathbb{H}_{\mathcal{T}}$  whose support contains an element  $K_a$  is built by tensor product of 1D integrated Legendre polynomials basis functions. Specifically, a basis function  $\phi$  is a product of  $i$  1D linear functions and  $d - i$  bubble functions where  $i \in (0, \dots, d)$ .

**Nodes:** Following the notation of [3], we define an abstraction of a node  $e$  as a component of an element  $K$  that refers to either a vertex, an edge, a face, or a volume. We define the dimension  $d_e \in \{0, \dots, d\}$  of  $e$  according to the dimension of the component it denotes:  $d_e = 0$  for a vertex,  $d_e = 1$  for an edge,  $d_e = 2$  for a face, and  $d_e = 3$  for a volume. For instance, in 2D, a quadrilateral element contains four nodes of dimension 0 (vertices), four of dimension 1 (edges), and one of dimension 2 (face).

*Remark.* In the following, we abuse the notation by considering indistinctly an element  $K$  either as a set of nodes or as a subdomain of  $\Omega$ .

**Node directions:** A node  $e$  of dimension  $d_e \neq 0$  has a set  $\mathcal{D}_e \subset \{1, \dots, d\}$  of  $d_e$  directions. If the node  $e$  is a vertex,  $d_e = 0$ , then we set  $\mathcal{D}_e := \emptyset$ . The subset of nodes of  $K_a \in \mathcal{T}$  containing the direction  $i$  is denoted as  $K_{a,i}$ .

**Basis functions associated to a node:** A basis function is associated to a node  $e$  of dimension  $d_e$  if it contains the tensor product of exactly  $d - d_e$  1D linear functions whose support includes  $e$ .

*Example.* As illustrated in Figure 4a,  $P_1^x$  is associated with vertex 2,  $P_2^x$  is associated with vertex 1, and  $P_3^x$  is associated with edge 3. In Figure 4b, the basis function  $P_1^x P_1^y$  is associated with vertex 3,  $P_1^x P_3^y$  is associated with edge 6, and  $P_3^x P_3^y$  is associated with face 9.

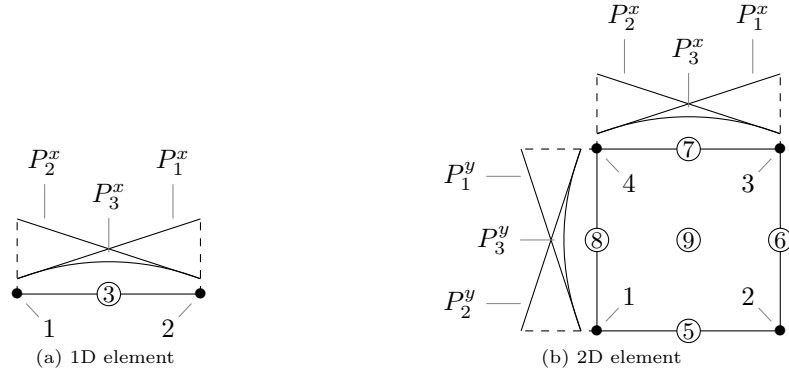


Figure 4: Nodes distribution and basis functions in one element

**Node orders:** For a given node  $e$  of dimension  $d_e \in \{1, \dots, d\}$  of an element  $K_a \in \mathcal{T}$ , the orders of a node is a vector containing the highest polynomial order of the basis functions along each of its directions and it is denoted as  $p = (p_1, \dots, p_{d_e}) \in (\mathbb{N} \setminus \{0\})^{d_e}$ . If  $e$  has no associated basis function, then we set  $p = 1$  in each direction. We designate the pair consisting of a node and its associated orders by  $e_p$ .

**Element orders:** The orders of an element  $K_a \in \mathcal{T}$ , denoted as  $\mathcal{O}(K_a)$ , is a set that gathers the orders of all its nodes.

### 2.3. Genealogical tree

Given an arbitrary element  $K$ , we define its following genealogy tree properties:

- *Level*: The level of an element is the number of successive ancestors different from itself that it has. For example, a level 0 element is a root element. The application  $\mathcal{L} : K \mapsto n \in \mathbb{N}$  returns the element level.
- *Children*: If  $K$  has been refined, then it admits a set of  $N_{\text{child}}$  children,  $\{K_C^j\}_{j=1}^{N_{\text{child}}}$ , as illustrated in Figure 5. We define the application  $\mathcal{C}$  that for  $K$  returns the set of children  $\mathcal{C}(K) = \{K_C^j\}_{j=1}^{N_{\text{child}}}$ . If  $K$  is unrefined, we define  $\mathcal{C}(K) = \emptyset$  unless  $K$  is a root active element, in which case we define  $\mathcal{C}(K) = \{K\}$ .
- *Parent*: If there exists a  $K_P$  such that  $K \in \mathcal{C}(K_P)$ , then  $K_P$  is called the parent of  $K$ . We define the application  $\mathcal{P}$  that for  $K$  returns its parent  $\mathcal{P}(K) = \{K_P\}$ .  $\mathcal{P}(\mathcal{T})$  is the set of all parents of the elements of  $\mathcal{T}$ .
- *Siblings*: The siblings of  $K$  are the elements that share the same parent element. We define the application  $\mathcal{S}$  that for a given  $K$  returns itself and its siblings  $\mathcal{S} := \mathcal{C} \circ \mathcal{P}$ . Note that if  $K$  is a root element, we set  $\mathcal{S}(K) := \{K\}$ .

*Example.* In Figure 5, we illustrate 1D and 2D genealogical trees.

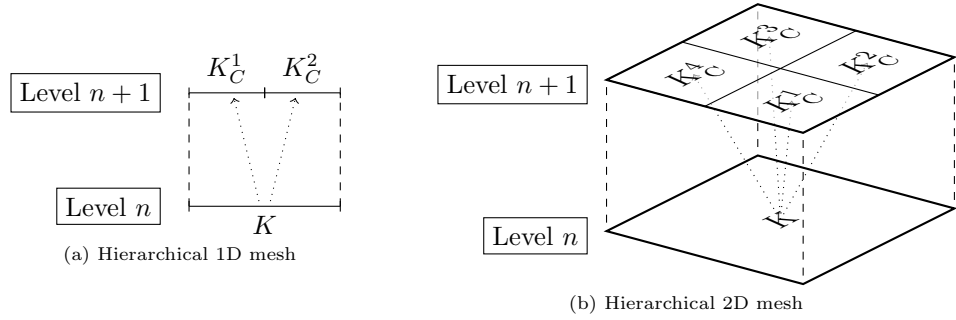


Figure 5: Hierarchical isotropic mesh element subdivision

### 2.4. Supported refinements and unrefinements

**Mesh operations:** For a given active element  $K_a \in \mathcal{T}$ , we define the following operations:

- *h-unrefinement* on  $K_a$ : setting its siblings  $\mathcal{S}(K_a)$  to inactive and  $\mathcal{P}(K_a)$  as active.  $\mathcal{P}(K_a)$  inherits the maximum orders of its children. Note that if  $\mathcal{L}(K_a) = \mathcal{L}(\mathcal{P}(K_a)) = 0$ , then an *h*-unrefinement has no effect since  $\mathcal{S}(K_a) = \mathcal{P}(K_a) = \{K_a\}$ .
- *h-refinement* on  $K_a$ : breaking  $K_a$  into a set of children  $\mathcal{C}(K_a)$ , and setting its children active. They inherit the orders of  $K_a$ .
- *p-unrefinement* of one node  $e_p$  of  $K_a$  in one direction  $i \in \mathcal{D}_e$ : setting  $p$  to  $\max(p - \mathbb{1}_i, 1)$  where  $\mathbb{1}_i$  is a vector of dimension  $d_e$  with 1 in the  $i^{\text{th}}$  component and 0 everywhere else.
- *p-refinement* of one node  $e_p$  of  $K_a$  in one direction  $i \in \mathcal{D}_e$ : setting  $p$  to  $p + \mathbb{1}_i$ .
- *p-(un)refinement* of  $K_a$  in one direction  $i$ : *p*-(un)refining, in the direction  $i$ , the interior node of  $K_a$  (i.e.,  $e \in K_a$  such that  $d_e = d$ ). The orders of the nodes of lower dimension containing the direction  $i$  are set as the minimum of the orders  $p_i$  of the contiguous interior nodes.

### 3. Removable basis functions

Our automatic adaptive unrefinement strategy takes full advantage of the hierarchical structure of our basis functions, both in terms of  $h$  and  $p$ . The idea is to identify individual basis functions that can be directly removed from the discretization without affecting the others and such that the remaining basis functions generate a *complete* polynomial subspace of  $\mathbb{H}_{\mathcal{T}}$  in the sense that such subspace restricted to each element is given by all polynomials of a given order. We denote them as *removable* basis functions ( $\text{Rm}_{\text{basis}}$ ). These will be the candidates for unrefinement (removal) at a given iteration. Thus, it is essential to identify them for any given mesh in our adaptive strategy.

*Example.* Let us illustrate the set of removable basis functions in the 1D cases shown in Figure 6 and the 2D scenario described by Figure 7. The removable basis functions (in green) are selected such that if individually removed, the remaining subspace still constitutes a complete polynomial subspace. If any other basis functions (drawn in red) would be removed, we would need to redefine the basis functions for the corresponding space to be complete.

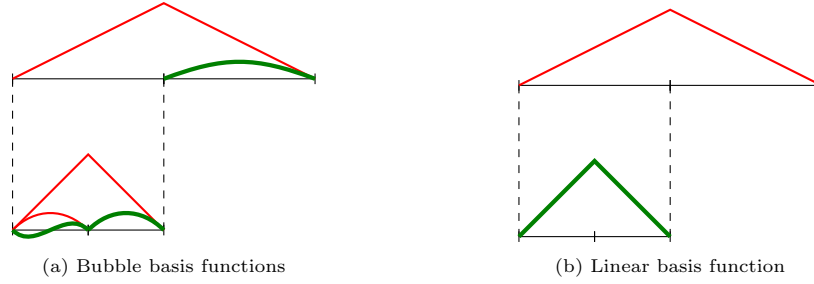


Figure 6: 1D removable basis functions (in green)

**Removable basis functions for the  $hp$ -adaptive case:** We define the set of removable basis functions of a node, namely  $\text{Rm}_{\text{basis}}(e_p)$ , as follows:

- If  $e_p$  is a vertex whose associated linear basis function has its support on one next-to-last level element whose children are of order  $p = 1$  (i.e., no bubbles), this basis function is *removable*.
- if  $e_p$  is a node of dimension  $d_e \neq 0$ , the *removable* basis functions in a given direction  $i \in \mathcal{D}_e$  are the basis functions whose cartesian product contains the highest degree in the direction  $i$ .

The set of removable basis functions in the direction  $i$  is denoted  $\text{Rm}_{\text{basis}}(e_p, i)$  and we define the set

$$\text{Rm}_{\text{basis}}(e_p) := \bigcup_{i \in \mathcal{D}_e} \text{Rm}_{\text{basis}}(e_p, i).$$

**Removable basis functions for the  $h$ -adaptive case:** When performing  $h$ -adaptivity with any order, we need to modify the definition of removable linear basis function: we consider as removable solely the linear basis functions whose support is entirely contained on a penultimate level element, *independently* of the polynomial order of its children.

With the above definitions, we obtain the following properties:

- The basis functions of an element  $K$  that are eliminated when performing a  $p$ -unrefinement in a given direction are removable.
- If there are only linear basis functions in  $K$ , then the ones that are eliminated when performing an  $h$ -unrefinement on  $\mathcal{P}(K)$  are removable.



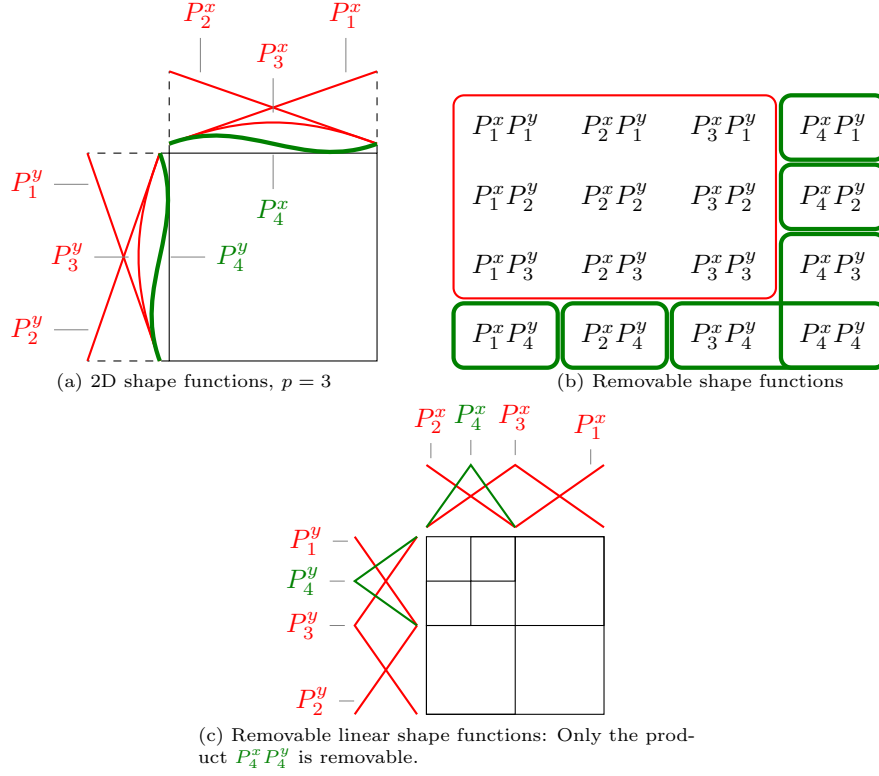


Figure 7: 2D removable shape functions (in green)

- When performing  $h$ -unrefinements only (for any  $p$ ), the removable basis functions coincide with the active linear ones.

**Notations around removable basis functions:** For the ease of the presentation, we introduce the following notations.

- For any given set of nodes  $K$ , we denote  $K^{\text{rm}}$  the subset of the nodes of  $K$  that contains removable basis functions as for example  $\mathcal{T}^{\text{rm}} \subset \mathcal{T}$  or  $K_a^{\text{rm}} \subset K_a \subset \mathcal{T}$ .
- For any given set of elements  $\mathcal{T}$ , we denote  $\mathcal{T}_K^{\text{rm}}$  the set of elements that contains at least one removable basis function.
- In particular, the subset of  $\mathcal{T}_K^{\text{rm}}$  that contains the nodes with a linear removable basis function is denoted  $\mathcal{T}_{K,h}^{\text{rm}}$  and  $\mathcal{T}_{K,p}^{\text{rm}}$  gathers those with at least one bubble removable basis function.
- We denote the cardinal of any given set  $K$  as  $n_K$ . For example,  $n_{\mathcal{T}^{\text{rm}}}$  or  $n_{K_a^{\text{rm}}}$ .

#### 4. Energy contribution indicators

Since the bilinear form  $b$  is symmetric and positive definite, we denote the energy  $J(v) := \frac{1}{2}b(v, v) - f(v)$ , for any  $v \in \mathbb{H}$ . The solution  $u$  of (1) satisfies

$$u = \underset{v \in \mathbb{H}}{\operatorname{argmin}} J(v) \quad (2)$$

For a given partition  $\mathcal{T}$ , the finite elements solution  $u_{\mathcal{T}}$  of (1) in  $\mathbb{H}_{\mathcal{T}}$  ( $u_{\mathcal{T}}$  also verifies (2) in  $\mathbb{H}_{\mathcal{T}}$ ) can be decomposed as  $u_{\mathcal{T}} = \sum_{i=1}^N u_i \phi_i$  where  $N$  is the dimension of  $\mathbb{H}_{\mathcal{T}}$ .

**Contribution in energy:** For a given  $\tilde{u} \in \mathbb{H}_{\mathcal{T}}$ , we define its energy contribution as:

$$\begin{aligned}\mathcal{R}(\tilde{u}) &:= J(u_{\mathcal{T}} - \tilde{u}) - J(u_{\mathcal{T}}) \\ &= \frac{1}{2}b(\tilde{u}, \tilde{u}) - b(u_{\mathcal{T}}, \tilde{u}) + f(\tilde{u}) \\ &= \frac{1}{2}b(\tilde{u}, \tilde{u})\end{aligned}$$

**Nodal and elemental indicators:** We define the error indicators for the nodes and the elements as follows:

- For a given node  $e$ , such that  $\text{Rm}_{\text{basis}}e \neq \emptyset$ , we define

$$\tilde{u}_e := \sum_{\phi_j \in \text{Rm}_{\text{basis}}(e)} u_j \phi_j. \quad (3)$$

- For a given node  $e$  such that  $d_e \neq 0$ , and  $i \in \mathcal{D}_e$ , and  $\text{Rm}_{\text{basis}}(e, i) \neq \emptyset$ , we define

$$\tilde{u}_{e,i} := \sum_{\phi_j \in \text{Rm}_{\text{basis}}(e, i)} u_j \phi_j. \quad (4)$$

- We define the element-wise error isotropic and anisotropic indicators of  $K_a \in \mathcal{T}_K^{\text{rm}}$  by accumulating the node-wise error indicator of the element, scaled as per removable basis function as in [3]:

$$\eta_{K_a}^{\mathcal{T}} := \frac{1}{n_{\text{Rm}_{\text{basis}}(K_a^{\text{rm}})}} \mathcal{R} \left( \sum_{e \in K_a^{\text{rm}}} \tilde{u}_e \right), \quad (5)$$

$$\eta_{K_a, i}^{\mathcal{T}} := \frac{1}{n_{\text{Rm}_{\text{basis}}(K_{a,i}^{\text{rm}})}} \mathcal{R} \left( \sum_{e \in K_{a,i}^{\text{rm}}} \tilde{u}_{e,i} \right), \quad \forall i \in \{1, \dots, d\}. \quad (6)$$

*Remark.* Note that  $u_{\mathcal{T}} - \tilde{u}$  is different from the solution  $u_{\tilde{\mathcal{T}}}$  for  $\tilde{\mathcal{T}}$  being the subset of  $\mathcal{T}$  without the basis function selected for the definition of  $\tilde{u}$ . They would only coincide if basis functions would be orthogonal in the norm prescribed by the bilinear form. However, we notice that if  $|J(u_{\mathcal{T}} - \tilde{u}) - J(u_{\mathcal{T}})|$  is small then  $|J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}})|$  is also small. Indeed, since  $J$  is monotonously decreasing as the space is enriching,  $J(u_{\mathcal{T}}) \leq J(u_{\tilde{\mathcal{T}}})$  and  $J(u_{\mathcal{T}}) \leq J(u_{\mathcal{T}} - \tilde{u})$  since  $u_{\mathcal{T}}$  is a solution of (1). Then,

$$\begin{aligned}& |J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}})| - |J(u_{\mathcal{T}} - \tilde{u}) - J(u_{\mathcal{T}})| \\ &= J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}}) - J(u_{\mathcal{T}} - \tilde{u}) + J(u_{\mathcal{T}}) \\ &= J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}} - \tilde{u})\end{aligned}$$

We notice that,  $J(u_{\tilde{\mathcal{T}}}) \leq J(u_{\mathcal{T}} - \tilde{u})$  since  $u_{\mathcal{T}} - \tilde{u} \in \mathbb{H}_{\tilde{\mathcal{T}}}$  and  $u_{\tilde{\mathcal{T}}}$  is solution of (1) in  $\mathbb{H}_{\tilde{\mathcal{T}}}$ . Thus

$$|J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}})| \leq |J(u_{\mathcal{T}} - \tilde{u}) - J(u_{\mathcal{T}})|.$$

Nonetheless, it may happen that  $|J(u_{\mathcal{T}} - \tilde{u}) - J(u_{\mathcal{T}})|$  is large whereas  $|J(u_{\tilde{\mathcal{T}}}) - J(u_{\mathcal{T}})|$  is small. In that case, our algorithm would not remove such basis function in that particular iteration, making the algorithm non-optimal. Later unrefinement iterations would correct such mistakes.

*Remark.* The indicators are local computable quantities. Indeed, each  $\tilde{u}_e$  involves a limited number of basis functions, and by definition of the residual  $\mathcal{R}$ , the nodal indicator  $\mathcal{R}(\tilde{u}_e)$  results in local computations and element-wise accumulations.

To determine which element contains *useless* basis function(s), we compute the average “work” per basis function. Then a basis function is considered useless if its work contribution is below some small percentage of the average.

**Average quantity:** The average contribution over all the elements is computed as:

$$W_{\text{avg}}^{\mathcal{T}} = \frac{1}{n_{\mathcal{T}_K^{\text{rm}}}} \sum_{K_a \in \mathcal{T}_K^{\text{rm}}} \eta_{K_a}^{\mathcal{T}}.$$

The local average contribution over the sons of a family is computed as: For all  $K_a \in \mathcal{T}_K^{\text{rm}}$ , such that  $\mathcal{L}(K_a) \neq 0$  and  $\mathcal{S}(K_a) \subset \mathcal{T}_K^{\text{rm}}$ ,

$$W_{\text{avg}}^{\mathcal{T}, \mathcal{S}(K_a)} = \frac{1}{n_{\mathcal{S}(K_a)}} \sum_{K \in \mathcal{S}(K_a)} \eta_K^{\mathcal{T}}.$$

## 5. Algorithm

The core of our approach is the  $hp$ -unrefinement step acting on any given mesh. This algorithm is then encapsulated in a two-steps procedure: 1) Arbitrary refinements that increases the number of degrees of freedom (dofs) 2)  $hp$ -unrefinements that decrease it. Let us present first the unrefinement step and then describe the entire adaptive process.

### 5.1. Firing (coarsening) policy

The  $hp$  firing policy has to decide whether unrefining a given element in polynomial order  $p$  or in element size  $h$ . The philosophy is the following: (a) if the selected removable basis function is a bubble, then we  $p$ -unrefine it unless the bubbles of the whole siblinghood are marked, in which case we  $h$ -unrefine the siblinghood; (b) if the selected removable basis function is linear, then the only choice is an  $h$ -unrefinement. The  $h$  or  $p$  dilemma is illustrated in the 1D case in Figure 8. Algorithm 1 describes it. We emphasize that the global average is computed only once on the initial (finest) mesh and remains fixed during the unrefinement process. By doing so, we ensure that the number of unrefinements decreases along the iterative coarsening process that will eventually stop.

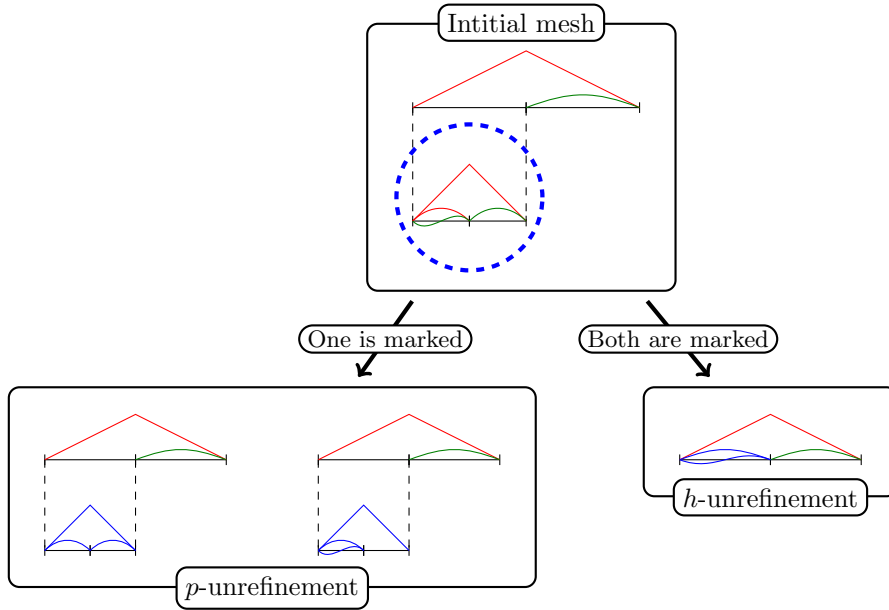


Figure 8: 1D  $hp$ -unrefinements

---

Algorithm 1: Firing policy (coarsening step)

---

**Inputs:**  $\mathcal{T}_0$  given mesh;  $\alpha_p, \alpha_h \geq 0$ .  
**Output:**  $\mathcal{T}$  coarsened mesh  
 $\mathcal{T} := \mathcal{T}_0$  initialize the mesh to be unrefined.  
**do**  
    Solve: solve the problem on  $\mathcal{T}$ .  
    Estimate: compute the indicators  $\eta_{K_a}^{\mathcal{T}}, \forall K_a \in \mathcal{T}^{\text{rm}}$  and the average  $W_{\text{avg}}^{\mathcal{T}_0}$ .  

$p$ -Mark:  $\forall K_a \in \mathcal{T}^{\text{rm}} \setminus \mathcal{T}_h^{\text{rm}}, \forall i \in \{1, \dots, d\}$ ,  
        **if**  $|\eta_{K_a, i}^{\mathcal{T}}| \leq \alpha_p |W_{\text{avg}}^{\mathcal{T}_0}|$   
        **then** mark  $K_a$  for  $p$ -unrefinement in the direction  $i$ .  

$h$ -Mark:  $\forall K_a \in \mathcal{T}^{\text{rm}}$  such that  $\mathcal{S}(K_a) \subset \mathcal{T}^{\text{rm}}$ ,  
        **if**  $|W_{\text{avg}}^{\mathcal{T}, \mathcal{S}(K_a)}| \leq \alpha_h |W_{\text{avg}}^{\mathcal{T}}|$ ,  
        **then** mark  $\mathcal{S}(K_a)$  for  $h$ -unrefinement.  
    Escape: **if** nothing has been marked, **return**  $\mathcal{T}$  as unrefined mesh.  
    Unrefine: update  $\mathcal{T}$ .  
**end**

---

### 5.2. Global hiring and firing policy

The refining algorithm that includes the unrefining algorithm is kept simple: One step of refinements and one of unrefinements until the requested precision is reached. The refinement step is up to the user discretion. We choose to alternate between global  $h$ -refinement and  $p$ -refinement. One could, for instance, perform refinements only on part of the mesh in which the error is suspected to be large. The procedure is described in Algorithm 2. The complete adaptive process is schematized in Figure 9.

---

Algorithm 2: Adaptive process

---

**Input:**  $\mathcal{T}$  initial coarse mesh  
**Output:**  $\mathcal{T}$  adapted mesh  
**do**  
    Refine: perform arbitrary refinements on  $\mathcal{T}$  ( $h$ ,  $p$  or both).  
    Unrefine: Run the *firing policy* to update  $\mathcal{T}$ .  
    Escape: **if** the unrefined  $\mathcal{T}$  satisfies a given precision, **return**  $\mathcal{T}$  as the final adapted mesh.  
**end**

---

*Remark.* In this work, global refinements are performed alternatively in  $p$  for one iteration and then in  $h$  for the subsequent iteration. However, when performing  $p$  refinements, in the context of  $hp$  unrefinements, we impose that the difference in polynomial orders of a given element and its active neighbors in each direction is below  $\Delta p = 6$ . Otherwise, instead of  $p$ -refining the element, we  $h$ -refine it. The reason behind this choice is that such gaps in polynomial orders occur in the presence of singularities that are better treated by  $h$ -refinements. Our unrefinement algorithm may fail to detect such cases since the bubble of higher degree (removable) may exhibit a large estimator and are not selected for unrefinements. We also impose that the maximum order should not exceed  $p = 11$ . If a  $p$ -refinement would provoke such a situation, we again apply an  $h$ -refinement instead.

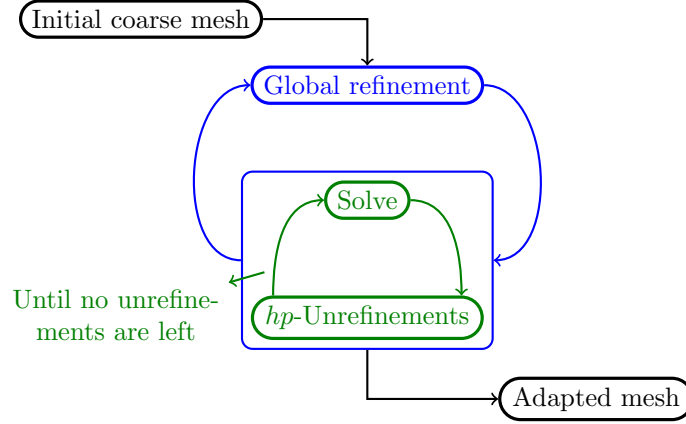


Figure 9: Adaptive Algorithm

## 6. Implementation details

An important feature of our method is its simplicity as on top of the multi-level data structure, our  $hp$ -adaptive algorithm requires below a thousand lines written in Fortran for its implementation. Furthermore, the implementation is independent of the spatial dimensionality of the problem. Additionally, the same algorithm serves for  $hp$ -adaptivity,  $p$ -adaptivity for any element-size, and  $h$ -adaptivity for any order.

Regarding computational cost, we compute the solution once per coarsening iteration, and the computational cost of the local indicators grows with the number of levels due to the hierarchical data structure although, in practical cases, the depth of the refinements is limited. Additionally, the method could also be parallelized following the strategy described in [12].

We provide a schematic code description for determining the removable basis functions in Algorithm 3 and to compute their contribution in Algorithm 4.

Algorithm 3: Determination of the removable basis function

---

**Inputs:**  $K \in \mathcal{T}^{\text{rm}}$ ,  $e \in K$ ,  $i \in d_e$

**Output:**  $\tilde{u}_{e,i}$

---

**if** performing  $h$ -unrefinement (any  $p$ ) **then**

Mark all last level active linear basis functions as removable.

**else**

**if**  $p_{e,i} \geq 2$  **then**

        Retrieve the indexes of the basis functions up to  $p_{e,i} - 1$ .

        Retrieve the indexes of the basis functions up to  $p_{e,i}$ .

Mark as removable functions all basis functions corresponding with the difference between both indices lists.

**else if** the maximum orders of all the siblings of  $K$  (ie.  $\mathcal{S}(K)$ ) is 1, **then**

Mark the linear basis functions as removable.

**else**

        The node  $e$  has no removable basis function.

**end**

**end**

---

---

**Input:**  $\mathcal{T}$  given mesh.  
**Outputs:**  $\eta_K$ , for all  $K$  in  $\mathcal{T}^{\text{rm}}$ .  
**do** for all  $K$  in  $\mathcal{T}^{\text{rm}}$   
     **do** for all  $e$  in  $K$  and for all  $i$  in  $d_e$   
         Find the removable basis functions  $\tilde{u}_e$  and  $\tilde{u}_{e,i}$ , Eqs. (3) and (4).  
     **end**  
     Compute their contribution:  $\eta_{K_a}^{\mathcal{T}}$ ,  $\eta_{K_a,i}^{\mathcal{T}}$  with Eqs. (5) and (6).  
**end**

---

## 7. Numerical Results

In this section, we solve the following problem: for  $\Omega := (a, b)^d$  where  $d$  is the dimension of the domain and  $\partial\Omega = \Gamma_D \cup \Gamma_N$ , and  $\Gamma_D \cap \Gamma_N = \emptyset$ :

$$\left| \begin{array}{l} \text{Find } u \text{ such that,} \\ \begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ \nabla u \cdot \hat{n} = g & \text{on } \Gamma_N, \end{cases} \end{array} \right.$$

where  $f, g$  and the definitions of  $\Gamma_D$  and  $\Gamma_N$  are set so that  $u$  fits certain given manufactured solutions and  $\hat{n}$  is the outward normal unit vector.

The relative error between the exact solution  $u$  and the approximated solution  $u_h$  is computed in percent against the exact solution in the  $H^1$  semi-norm:

$$e_h = \frac{|u - u_h|_{H^1}}{|u|_{H^1}} \cdot 100.$$

For every test case, we provide the convergence history for the  $hp$ -adaptive strategy  $\blacktriangleleft \blacktriangle$  against  $h$ -adaptivity with uniform  $p = 1$   $\blacktriangleleft \blacktriangle$  and  $p = 2$   $\blacktriangleleft \blacksquare \blacktriangle$ .

In the following numerical results, we set the parameters of the unrefinement process to  $\alpha_p = 0.1$  and  $\alpha_h = 0.3$ . Such choice provides the best results amongst the tested parameters. Furthermore, in the following examples (and for Appendix A), we perform first a global  $h$ -refinement and, for the next iteration, a global  $p = p + 2$  refinement.

### 7.1. 1D test problems

We consider the following three manufactured solutions in the domain  $\Omega = (0, 1)$ : A smooth solution, a singular solution, and a solution with a strong gradient. The initial mesh is composed of two equal-size elements with orders  $p = 1$ .

1. **Regular Solution:**  $u = \sin(2\pi x)$ ,  $\Gamma_D = \{0\} \cup \{1\}$ ,  $\Gamma_N = \emptyset$ . The smoothness of the solution is captured with a coarse mesh with uniform relatively elevated orders ( $p = 3$ ), as shown in Figure 10. We need 11 dofs to reach 1% error.
2. **Singular Solution:**  $u = x^{\frac{3}{5}}$  exhibits a singular behavior in  $H^1$  at 0,  $\Gamma_D = \{0\}$ ,  $\Gamma_N = \{1\}$ . The  $hp$ -adaptive strategy selects tiny elements nearby the singularity (of size  $10^{-10}$ ) and of growing size as we move away from the singularity (see Figure 11). The polynomial orders are high  $p = 9$  in the element closest to the singularity, while in the next two elements it decreases quickly to  $p = 3$ , and the rest are set to  $p = 4$ .

This non-optimal high order at the singularity is due to the alternated refinements we performed and because the contribution of the highest order basis function at the singularity is above average so the unrefinement algorithm will not remove it. To reach 1% of error, our algorithm needs around  $N = 104$  dofs whereas the algorithm of [3] require around  $N = 80$  dofs. However, for  $N = 145$  dofs, we obtain an error of 0.45% where they reached around 0.5%.

3. **High-Gradient Solution:**  $u = \text{atan}\left(a\left(x - \frac{1}{5}\right)\right) + \text{atan}\left(\frac{a}{5}\right)$  for  $a = 120$ ,  $\Gamma_D = \{0\}$ ,  $\Gamma_N = \{1\}$ , which is a regular solution with a strong gradient nearby  $x = 0.2$ , also known as a shock problem. The strong gradient (see Figure 12) is captured by element with both small  $h$  and high  $p$  around  $x = 0.2$ . To reach 2% Demkowicz [3] produced a mesh with around  $N = 23$  dofs when our algorithm needs  $N = 25$  dofs for 4% and  $N = 38$  dofs for 1.4%. Note, however, that they imposed Dirichlet boundary conditions on the whole boundary thus, our problem is less stable, which partly explains the need of extra dofs in the pre-asymptotical regime.

In all considered examples, the adapted mesh captures the main features of each solution correctly, and the convergence is exponential and competitive with respect to other existing methods (see, e.g., [3]).

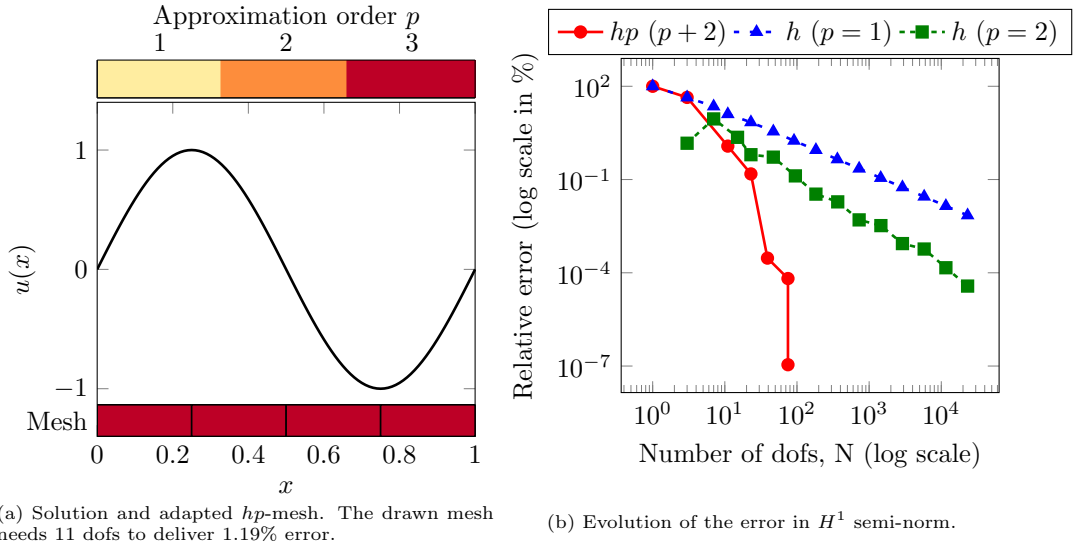
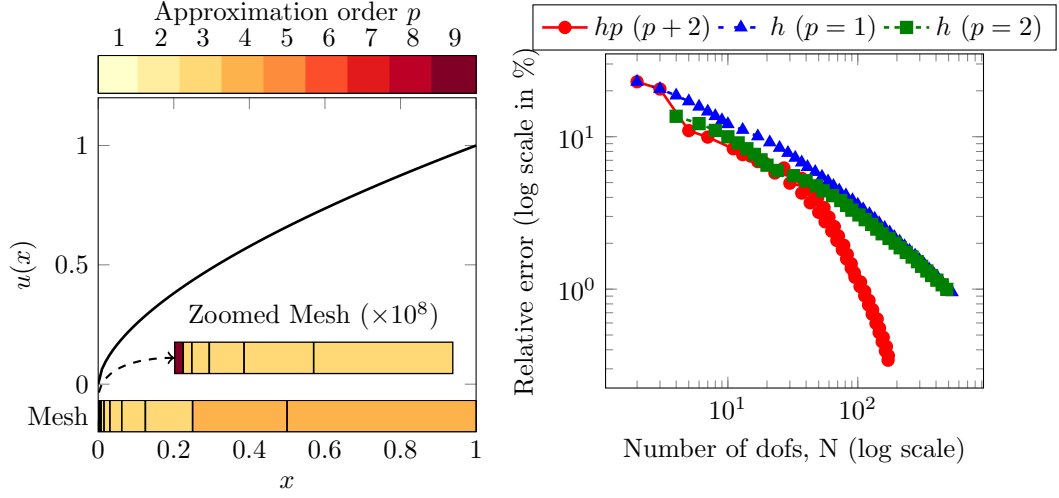


Figure 10: Regular solution.  $u = \sin(2\pi x)$ .

## 7.2. 2D test problems

Similarly to the 1D case, we consider the following three manufactured solutions: a smooth solution, a shock problem and a singular solution, where the singularity is generated by the non-convexity of the domain. The initial mesh has four (three for the L-shape domain) equal sized linear elements.

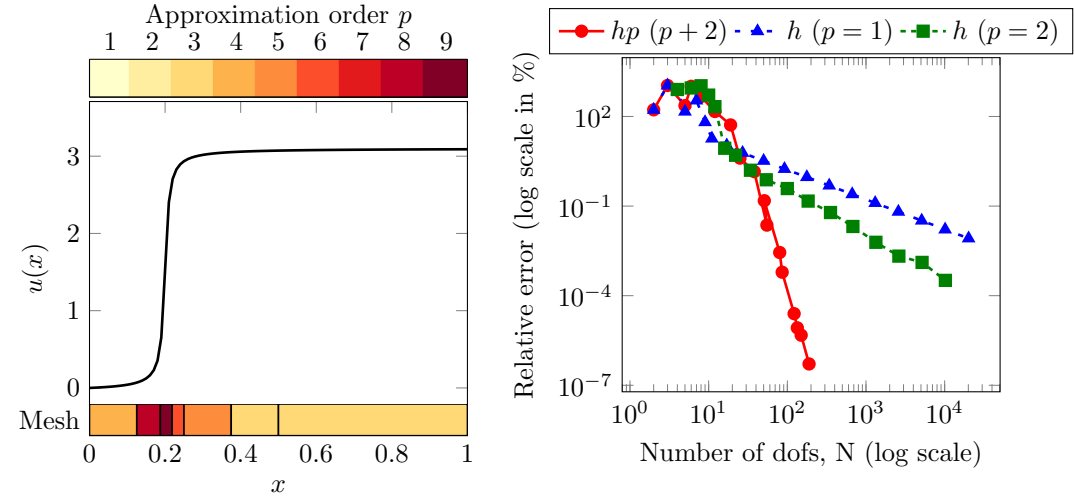
1. **Regular Solution:**  $u = \sin(2\pi x) \sin(2\pi y)$ , on  $\Omega = (0, 1)^2$  with  $\Gamma_D = \partial\Omega$ ,  $\Gamma_N = \emptyset$ . We display the results in Figure 13.
2. **Reentrant corner:**  $u = r^{\frac{2}{3}} \sin(\frac{2}{3}(\theta + \frac{\pi}{2}))$ ,  $r = \sqrt{x^2 + y^2}$ ,  $\theta = \text{atan2}(y, x)$  on  $\Omega = (-1, 1)^2 \setminus (-1, 0)^2$  with  $\Gamma_D = \{0\} \times (-1, 1) \cup (-1, 1) \times \{0\}$  and  $\Gamma_N = \partial\Omega \setminus \Gamma_D$ . The singularity in  $H^1$  of this solution comes from the non convexity of the domain. Figure 14 shows competitive convergence rates and close to optimal meshes. A zoom around the singularity of the  $hp$  mesh is displayed in Figure 15
3. **Shock problem:**  $u = \text{atan}(a(r - \frac{1}{2})) - \text{atan}(-\frac{a}{2})$ ,  $a = 60$  and  $r = \sqrt{x^2 + y^2}$ , on  $\Omega = (0, 1)^2$  with  $\Gamma_D = \{(0, 0)\}$  and  $\Gamma_N = \partial\Omega \setminus \Gamma_D$ . The solution exhibits a strong gradient around  $r = 0.5$ . Figure 16 gathers the adapted meshes and convergence history. Note that, although widely used, the problem is



(a) Solution and adapted  $hp$ -mesh. The drawn mesh needs 104 dofs to deliver 1.04% error.

(b) Evolution of the error in  $H^1$  semi-norm.

Figure 11: Singular solution.  $u = x^{\frac{3}{5}}$ .



(a) Solution and adapted  $hp$ -mesh. The drawn mesh needs 38 dofs to deliver 1.42% error.

(b) Evolution of the error in  $H^1$  semi-norm.

Figure 12: High gradient solution.  $u = \text{atan}(a(x - \frac{1}{5})) + \text{atan}(a \cdot \frac{1}{5})$ ,  $a = 120$ .

ill-posed with such boundary conditions and there might occur extra refinements around the point-wise Dirichlet condition. Such extra refinements are visible in Figures 16a and 16b. However, our algorithm is able to finally remove them, as shown in Figure 16c.

As for the 1D examples, we show that our algorithm generates almost optimal meshes and exponential convergence rates.

### 7.3. 3D test problems

Finally, we illustrate our algorithm on a manufactured 3D shock problem: We consider  $\Omega := (0, 1)^3$  and  $\Gamma_D := \partial\Omega$  and  $f$  is selected so that  $u = \text{atan}(60(r-1)) \cdot x(x-1)y(y-1)z(z-1)$ , where  $r = \sqrt{x^2 + y^2 + z^2}$ . The initial mesh is a partition of the unit cube into  $4^3 = 64$  cubes.



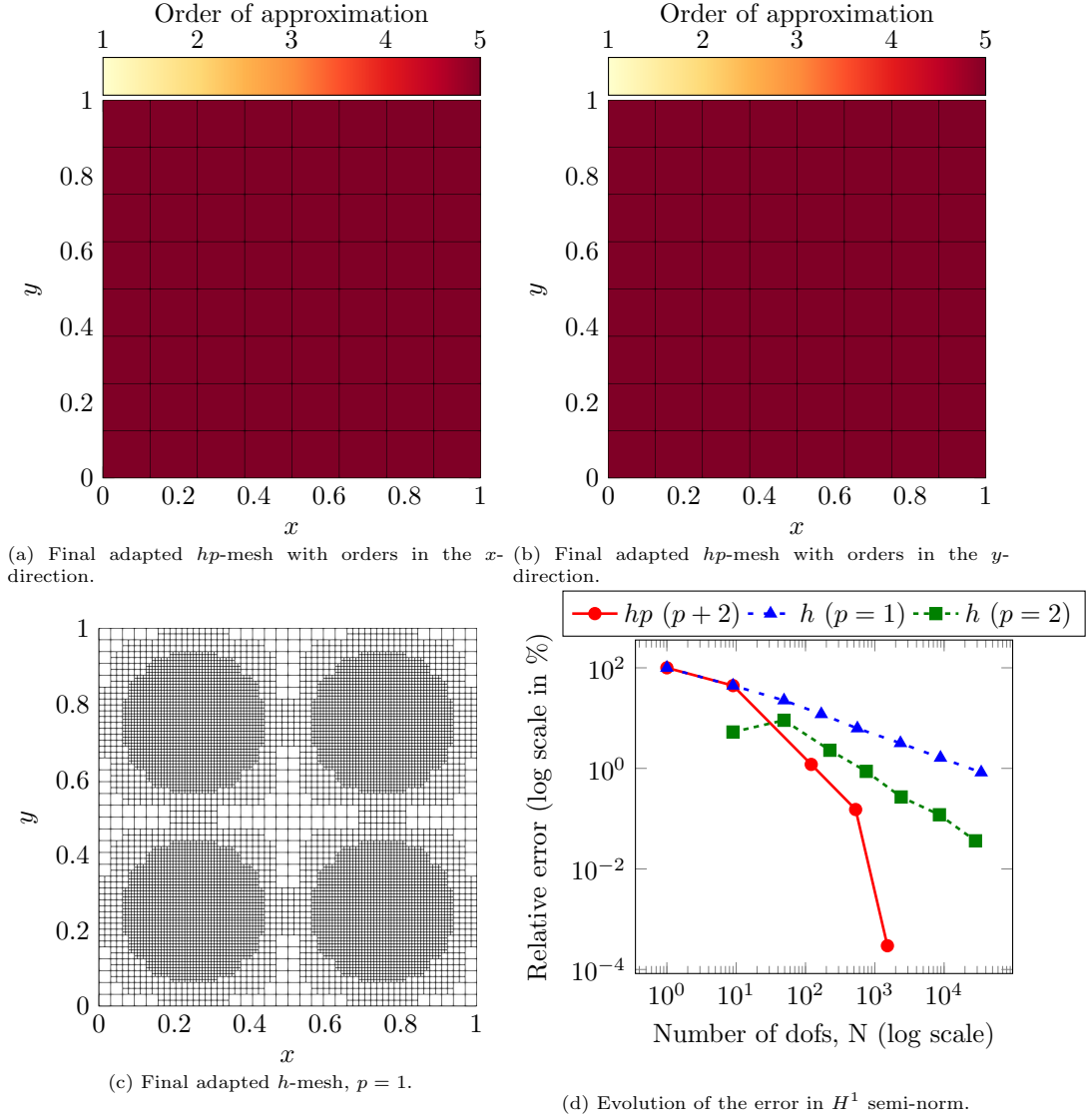


Figure 13: Regular solution. The final adapted  $hp$ -mesh, Figures 13a and 13b, needs 1521 dofs for delivering an error of  $2.9 \cdot 10^{-4}\%$ ; and the  $h$ -mesh, Figures 13c, delivers an error of  $3.621 \cdot 10^{-2}\%$  for 28369 dofs. Additionally, for 121 dofs, we obtain an error of 1.199%.

Figure 17 shows the solution computed on the finest  $hp$  mesh. Figure 18 displays three clips of the  $hp$ -adapted mesh. The clipping criteria is the refinement level of the elements. The shock is correctly captured via  $hp$ -refinements. Figure 19 shows the convergence results of the algorithm. We observe that the convergence of the  $h$ -adaptive process is linear while the  $hp$  version exhibits exponential-type convergence.

## 8. Conclusions

We introduce a novel  $hp$ -adaptive method for elliptic problems based on the multi-level data structure developed by Zander et al. [9, 10]. Our goal is to minimize the implementation efforts without undermining neither the quality of the adapted mesh nor the computational cost.

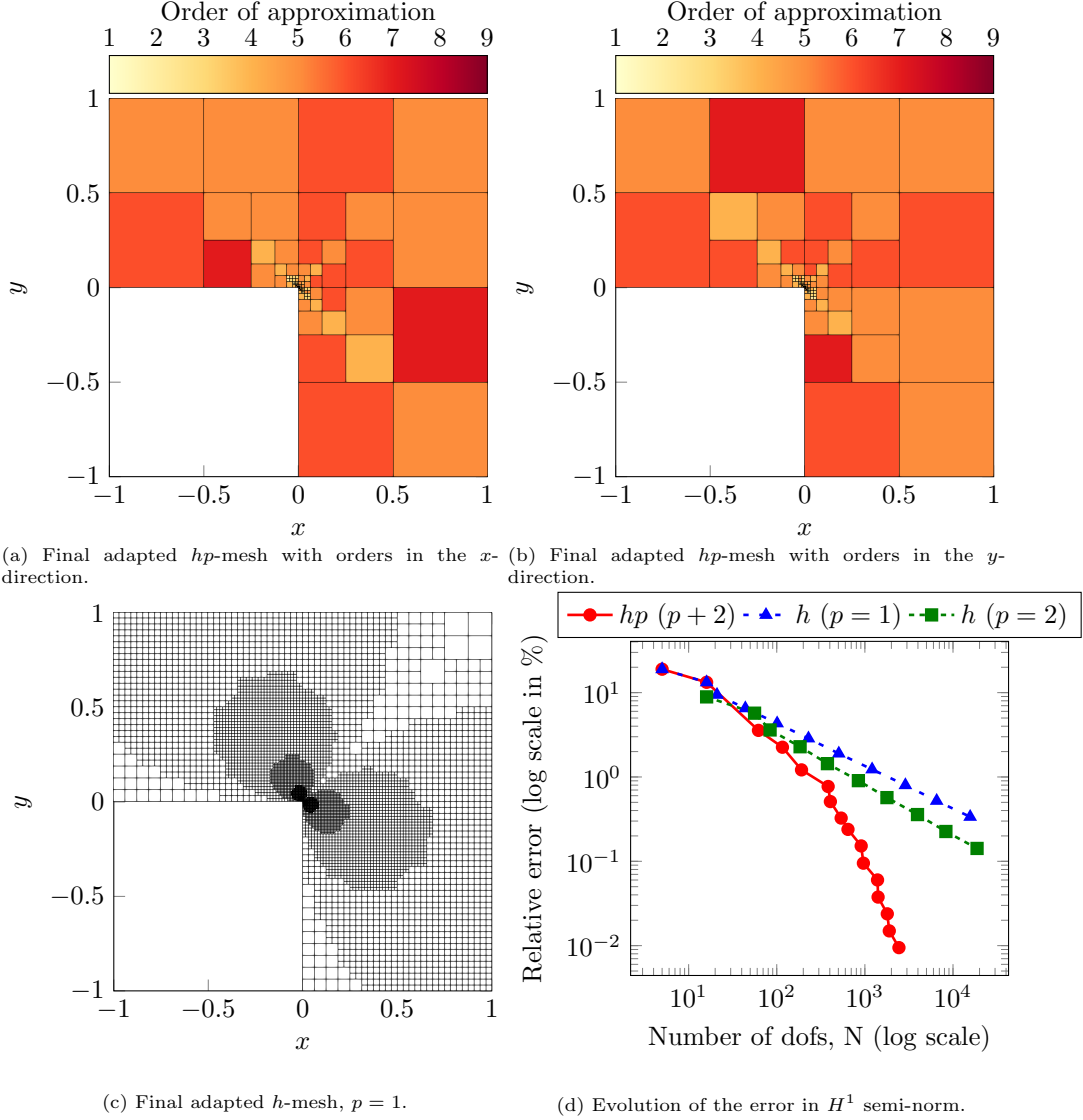


Figure 14: 2D singular reentrant corner problem. The final adapted  $hp$ -mesh, Figures 14a and 14b, needs 2435 dofs for delivering an error of  $9.46 \cdot 10^{-3}\%$ ; and the  $h$ -mesh, Figures 14c, delivers an error of 0.142% for 18752 dofs. Additionally, for 383 dofs, we obtain an error of 0.771%. The algorithm of Demkowicz [3] needs between 277 and 417 dofs to reach 1% of error.

We present a coarsening strategy that relies on the contribution of each *removable* basis function. If its contribution is small, then it is marked for unrefinement. Such unrefining policy does not rely on the initial mesh, thus providing great flexibility to build a fine mesh candidate for coarsening. Furthermore, by being able to unrefine a given mesh several times in a row, we can correct potential mistakes committed in previous iterations. This is especially advantageous since it allows to remove unnecessary refinements possibly introduced during the pre-asymptotic regime.

We numerically demonstrate using standard academic test problems that our method produces close to optimal meshes when compared with other existing techniques as, e.g. [3]. Indeed, we were able to detect and adjust the meshes properly for solutions that exhibit singularities, strong gradients, or even regular solutions in 1D and 2D and strong gradients in 3D.

The main advantages are both the limited implementation efforts and the flexibility of the refinement

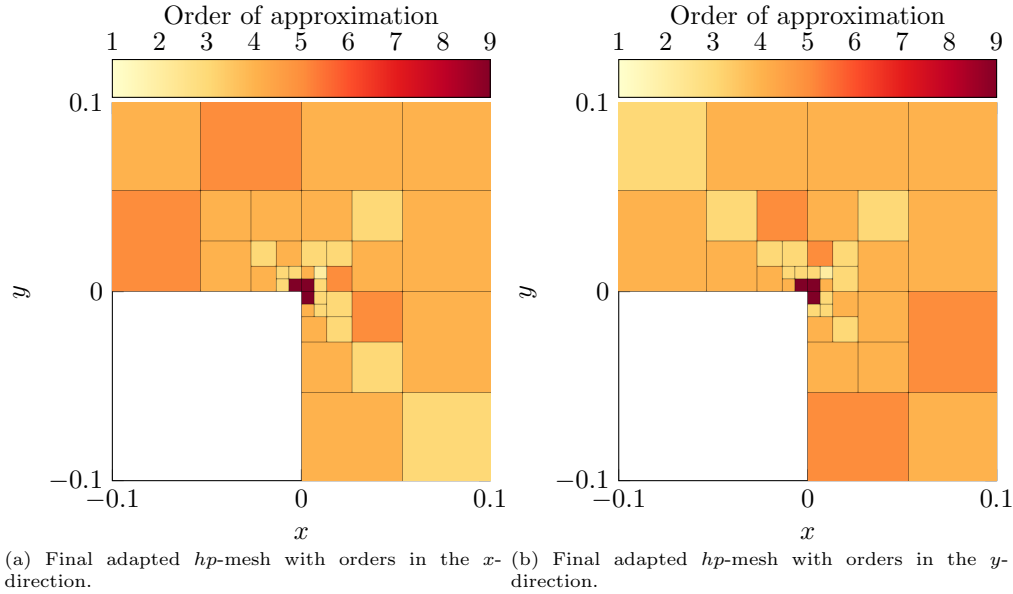


Figure 15: 2D singular reentrant corner problem. Zoom of the  $hp$  meshes around the singularity.

strategy. We also identified some limitations of the proposed  $hp$ -adaptive algorithm: (a) the lack of orthogonality of the basis functions may lead to non-optimal results, (b) the definition of the multi-level data structure forces isotropic  $h$ -(un)refinements, and (c) the current  $hp$ -adaptive algorithm is limited to elliptic problems due to the definition of the error indicators. The first limitation is a direct consequence of the lack of using projections, which also simplifies the implementation. The second one is due to the employed data structures. One possibility to overcome it would be the use of other data structures such as the ones discussed in [13]. To overcome the last one, in turn, we require a re-definition of the indicators that will be the object of future studies.

The continuation of this work includes: (a) the extension to non-elliptic problems perhaps via the use of a residual minimization method for providing an adequate error representation, (b) the use of triangular/tetrahedral meshes, and (c) the development of goal-oriented indicators following a similar element-wise strategy as in [54, 55].

## References

- [1] I. Babuška, B. Szabo, I. Katz, The  $p$ -version of the finite element method, *SIAM Journal on Numerical Analysis* 18 (3) (1981) 515–545. [arXiv:https://doi.org/10.1137/0718033](https://doi.org/10.1137/0718033), doi:10.1137/0718033. URL <https://doi.org/10.1137/0718033>
- [2] B. Guo, I. Babuška, The  $h$ - $p$  version of the finite element method, *Computational Mechanics* 1 (1) (1986) 21–41. doi:10.1007/BF00298636. URL <https://doi.org/10.1007/BF00298636>
- [3] L. Demkowicz, Computing with  $hp$ -adaptive finite elements. Vol. 1. One and two dimensional elliptic and Maxwell problems, *Applied Mathematics and Nonlinear Science Series*, Chapman & Hall/CRC, Boca Raton, FL, 2007. doi:10.1201/9781420011692. URL <http://dx.doi.org/10.1201/9781420011692>
- [4] L. Demkowicz, J. Kurtz, D. Pardo, M. Paszyński, W. Rachowicz, A. Zdunek, Computing with  $hp$ -adaptive finite elements. Vol. 2. Frontiers: three dimensional elliptic and Maxwell problems with applications, *Applied Mathematics and Nonlinear Science Series*, Chapman & Hall/CRC, Boca Raton, FL, 2008.
- [5] J. Schöberl, Netgen an advancing front 2d/3d-mesh generator based on abstract rules, *Computing and Visualization in Science* 1 (1) (1997) 41–52. doi:10.1007/s007910050004. URL <https://doi.org/10.1007/s007910050004>
- [6] A. Szymczak, A. Paszyńska, M. Paszyński, D. Pardo, Preventing deadlock during anisotropic 2D mesh adaptation in  $hp$ -adaptive FEM, *Journal of Computational Science* 4 (3) (2013) 170 – 179, agent-Based Simulations, Adaptive Algorithms,

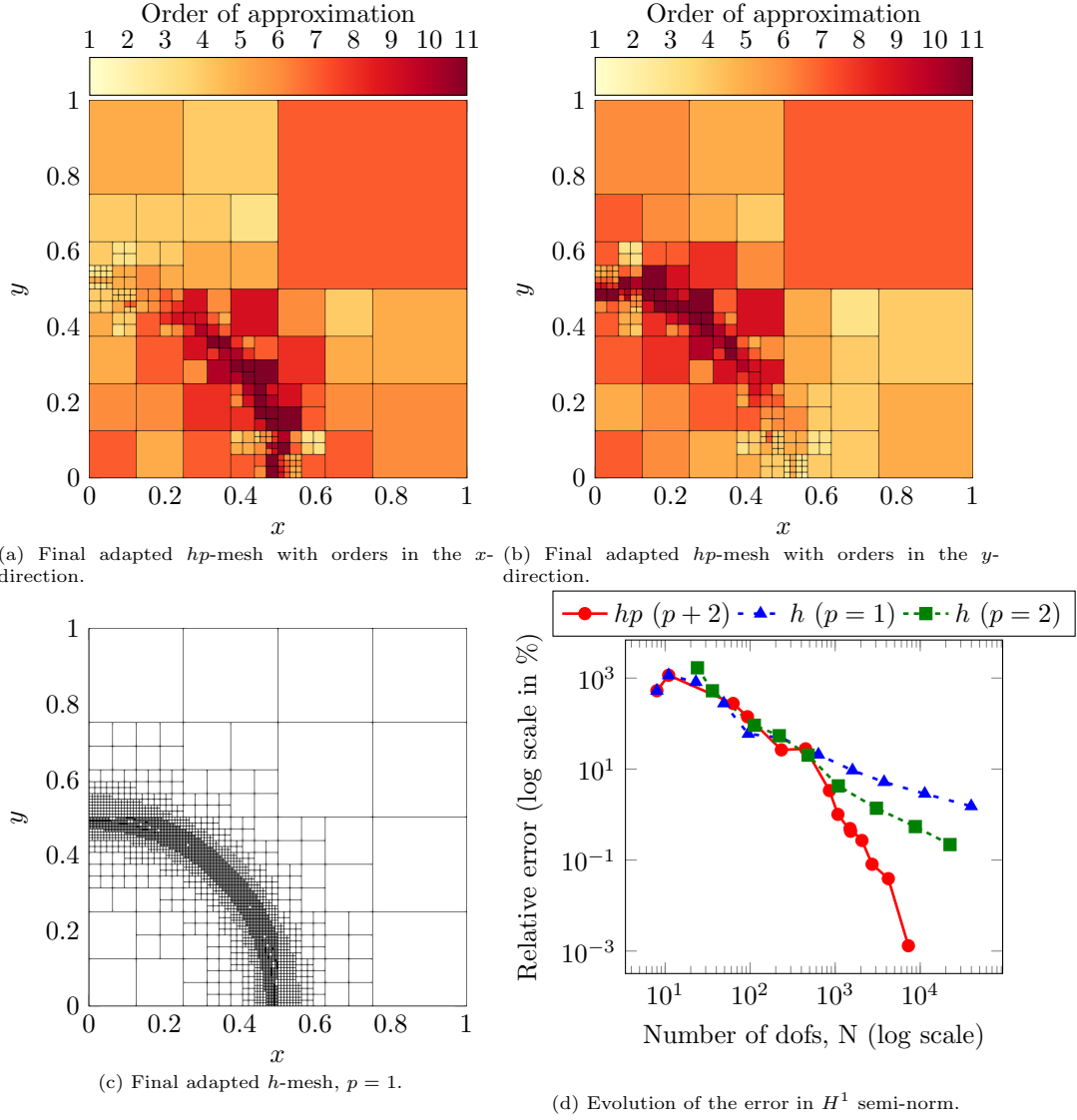


Figure 16: 2D shock problem. The final adapted  $hp$ -mesh, Figures 16a and 16b, needs 7260 dofs for delivering an error of  $1.3 \cdot 10^{-3}\%$ ; and the  $h$ -mesh, Figures 16c, delivers an error of 0.219% for 22404 dofs. Additionally, for 856 dofs, we obtain an error of 3.369%. The algorithm of Demkowicz [3] needs between 1261 and 1901 dofs to reach 1% of error.

ICCS 2011 Workshop. doi:<https://doi.org/10.1016/j.jocs.2011.09.001>.

URL <http://www.sciencedirect.com/science/article/pii/S187775031100086X>

- [7] P. Solin, S. Giani, An iterative adaptive  $hp$ -FEM method for non-symmetric elliptic eigenvalue problems, Computing 95 (1, suppl.) (2013) S183–S213.

URL <https://doi.org/10.1007/s00607-012-0251-7>

- [8] P. Solin, S. Giani, An iterative adaptive finite element method for elliptic eigenvalue problems, J. Comput. Appl. Math. 236 (18) (2012) 4582–4599.

URL <https://doi.org/10.1016/j.cam.2012.05.002>

- [9] N. Zander, T. Bog, S. Kollmannsberger, D. Schillinger, E. Rank, Multi-level  $hp$ -adaptivity: high-order mesh adaptivity without the difficulties of constraining hanging nodes, Computational Mechanics 55 (3) (2015) 499–517. doi:10.1007/s00466-014-1118-x.

URL <https://doi.org/10.1007/s00466-014-1118-x>

- [10] N. Zander, T. Bog, M. Elhaddad, F. Frischmann, S. Kollmannsberger, E. Rank, The multi-level  $hp$ -method for three-dimensional problems: Dynamically changing high-order mesh refinement with arbitrary hanging nodes, Computer Meth-

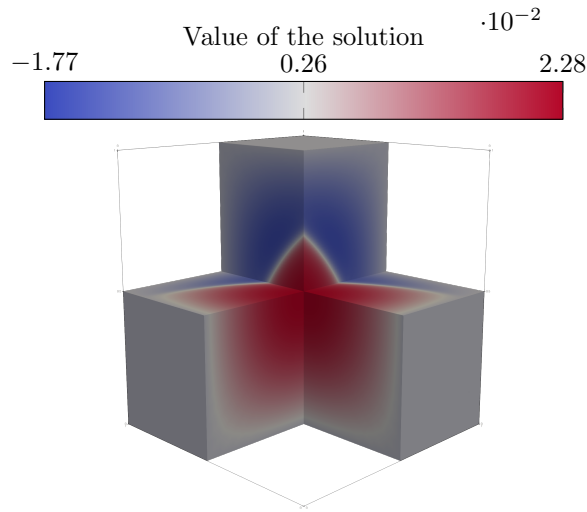


Figure 17: 3D shock problem: Solution computed on the final adapted *hp* mesh.

- ods in Applied Mechanics and Engineering 310 (2016) 252 – 277. doi:<https://doi.org/10.1016/j.cma.2016.07.007>. URL <http://www.sciencedirect.com/science/article/pii/S0045782516307289>
- [11] N. D. Zander, Multi-level *hp*-fem: dynamically changing high-order mesh refinement with arbitrary hanging nodes, Ph.D. thesis, Technische Universität München (2017).
- [12] J. N. Jomo, N. Zander, M. Elhaddad, A. Özcan, S. Kollmannsberger, R.-P. Mundani, E. Rank, Parallelization of the multi-level *hp*-adaptive finite cell method, Computers & Mathematics with Applications 74 (1) (2017) 126 – 142, 5th European Seminar on Computing ESCO 2016. doi:<https://doi.org/10.1016/j.camwa.2017.01.004>. URL <http://www.sciencedirect.com/science/article/pii/S0898122117300147>
- [13] P. D. Stolfo, A. Schröder, N. Zander, S. Kollmannsberger, An easy treatment of hanging nodes in *hp*-finite elements, Finite Elements in Analysis and Design 121 (2016) 101 – 117. doi:<https://doi.org/10.1016/j.finel.2016.07.001>. URL <http://www.sciencedirect.com/science/article/pii/S0168874X16301299>
- [14] M. Ainsworth, B. Senior, An adaptive refinement strategy for *hp*-finite element computations, Applied Numerical Mathematics 26 (1) (1998) 165 – 178. doi:[https://doi.org/10.1016/S0168-9274\(97\)00083-4](https://doi.org/10.1016/S0168-9274(97)00083-4). URL <http://www.sciencedirect.com/science/article/pii/S0168927497000834>
- [15] J. Oden, A. Patra, A parallel adaptive strategy for *hp* finite element computations, Computer Methods in Applied Mechanics and Engineering 121 (1) (1995) 449 – 470. doi:[https://doi.org/10.1016/0045-7825\(94\)00705-R](https://doi.org/10.1016/0045-7825(94)00705-R). URL <http://www.sciencedirect.com/science/article/pii/004578259400705R>
- [16] L. Demkowicz, W. Rachowicz, P. Devloo, A fully automatic *hp*-adaptivity, in: Proceedings of the Fifth International Conference on Spectral and High Order Methods (ICOSAHOM-01) (Uppsala), Vol. 17, 2002, pp. 117–142. doi:10.1023/A:1015192312705. URL <http://dx.doi.org/10.1023/A:1015192312705>
- [17] D. Pardo, Integration of *hp*-adaptivity with a two grid solver: applications to electromagnetics, Ph.D. thesis, The University of Texas at Austin (2004). URL <https://www.lib.utexas.edu/etd/d/2004/pardod042/pardod042.pdf>
- [18] M. Paszyński, L. Demkowicz, D. Pardo, Verification of goal-oriented *hp*-adaptivity, Comput. Math. Appl. 50 (8-9) (2005) 1395–1404. doi:10.1016/j.camwa.2005.03.018. URL <http://dx.doi.org/10.1016/j.camwa.2005.03.018>
- [19] D. Pardo, L. García-Castillo, L. Demkowicz, C. Torres-Verdín, A two-dimensional self-adaptive *hp* finite element method for the characterization of waveguide discontinuities. II. Goal-oriented *hp*-adaptivity, Comput. Methods Appl. Mech. Engrg. 196 (49-52) (2007) 4811–4822. doi:10.1016/j.cma.2007.06.023. URL <http://dx.doi.org/10.1016/j.cma.2007.06.023>
- [20] L. E. García-Castillo, D. Pardo, I. Gómez-Revuelto, L. F. Demkowicz, A two-dimensional self-adaptive *hp* finite element method for the characterization of waveguide discontinuities. I. Energy-norm based automatic *hp*-adaptivity, Comput. Methods Appl. Mech. Engrg. 196 (49-52) (2007) 4823–4852. doi:10.1016/j.cma.2007.06.024. URL <http://dx.doi.org/10.1016/j.cma.2007.06.024>
- [21] D. Pardo, L. Demkowicz, C. Torres-Verdín, C. Michler, PML enhanced with a self-adaptive goal-oriented *hp*-finite element method: simulation of through-casing borehole resistivity measurements, SIAM J. Sci. Comput. 30 (6) (2008) 2948–2964. doi:10.1137/070689796. URL <http://dx.doi.org/10.1137/070689796>
- [22] L. E. Garcia-Castillo, D. Pardo, L. F. Demkowicz, Energy-norm-based and goal-oriented automatic adaptivity for electromagnetics: Application to waveguide discontinuities, Microwave Theory and Techniques, IEEE Transactions on 56 (12)

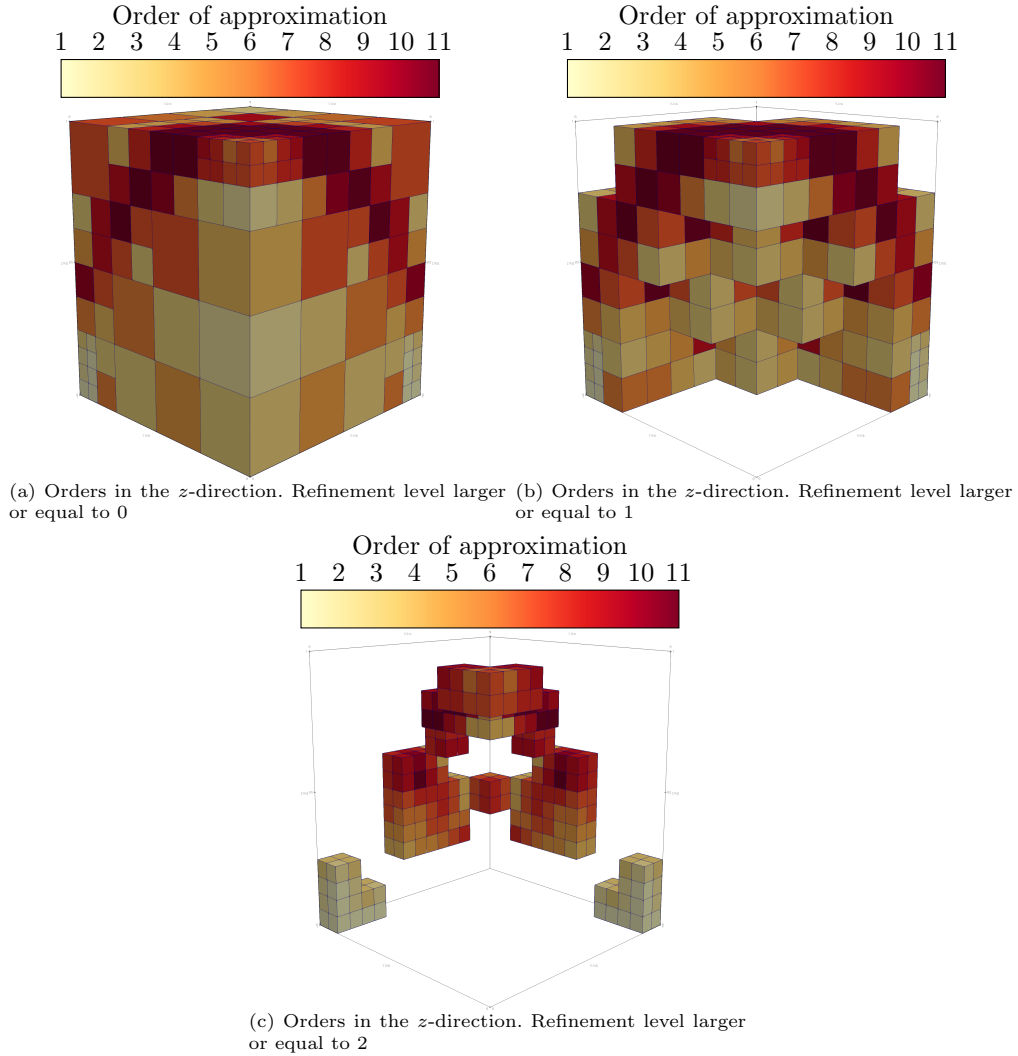


Figure 18: 3D shock problem: Final adapted  $hp$  mesh. Since the solution is symmetric, we show only the values of the polynomial order in the  $z$  component. We filtered the elements according to their refinement level, from the largest (0) to smaller elements (higher levels).

- (2008) 3039–3049.
- [23] D. Pardo, Multigoal-oriented adaptivity for  $hp$ -finite element methods, *Procedia Computer Science* 1 (1) (2010) 1953 – 1961. doi:<http://dx.doi.org/10.1016/j.procs.2010.04.219>.  
URL <http://www.sciencedirect.com/science/article/pii/S1877050910002206>
- [24] V. M. Calo, D. Pardo, M. R. Paszyński, Goal-oriented self-adaptive  $hp$  finite element simulation of 3D DC borehole resistivity simulations, *Procedia Computer Science* 4 (0) (2011) 1485 – 1495, proceedings of the International Conference on Computational Science, ICCS 2011. doi:<http://dx.doi.org/10.1016/j.procs.2011.04.161>.  
URL <http://www.sciencedirect.com/science/article/pii/S1877050911002195>
- [25] I. Gomez-Revuelto, L. E. Garcia-Castillo, S. Llorente-Romano, D. Pardo, A three-dimensional self-adaptive  $hp$  finite element method for the characterization of waveguide discontinuities, *Comput. Methods Appl. Mech. Engrg.* 249/252 (2012) 62–74. doi:10.1016/j.cma.2012.05.013.  
URL <http://dx.doi.org/10.1016/j.cma.2012.05.013>
- [26] M. Paszyński, D. Pardo, V. Calo, Parallel simulations of 3D DC borehole resistivity measurements with goal-oriented self-adaptive  $hp$  finite element method, *Journal of the Serbian Society for Computational Mechanics/Vol 6* (2) (2012) 1–18.
- [27] J. Alvarez-Aramberri, D. Pardo, H. Barucq, Inversion of magnetotelluric measurements using multigoal oriented  $hp$ -adaptivity, *Procedia Computer Science* 18 (2013) 1564 – 1573. doi:<http://dx.doi.org/10.1016/j.procs.2013.05.324>.

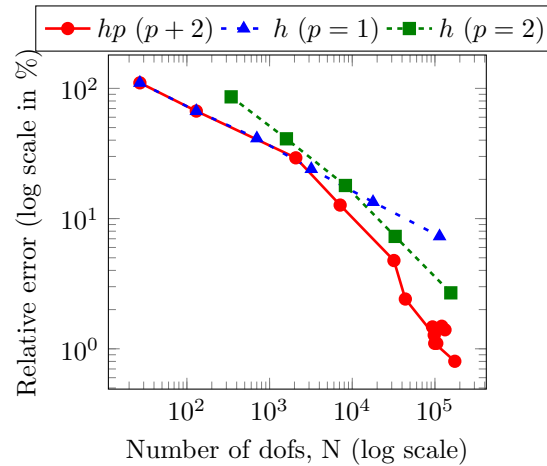


Figure 19: 3D shock problem: Convergence history.

- URL <http://www.sciencedirect.com/science/article/pii/S1877050913004675>
- [28] J. Alvarez-Aramberri, D. Pardo, H. Barucq, A secondary field based *hp*-Finite Element Method for the simulation of magnetotelluric measurements, *Journal of Computational Science* 11 (2015) 137–144. doi:<http://doi.org/10.1016/j.jocs.2015.02.005>.  
URL <http://www.sciencedirect.com/science/article/pii/S1877750315000150>
- [29] P. Houston, B. Senior, E. Süli, Sobolev regularity estimation for *hp*-adaptive finite element methods, in: *Numerical mathematics and advanced applications*, Springer Italia, Milan, 2003, pp. 631–656.
- [30] W. F. Mitchell, M. A. McClain, A comparison of *hp*-adaptive strategies for elliptic partial differential equations, *ACM Trans. Math. Softw.* 41 (1) (2014) 2:1–2:39. doi:10.1145/2629459.  
URL <http://doi.acm.org/10.1145/2629459>
- [31] D. D’Angella, N. Zander, S. Kollmannsberger, F. Frischmann, E. Rank, A. Schröder, A. Reali, Multi-level *hp*-adaptivity and explicit error estimation, *Advanced Modeling and Simulation in Engineering Sciences* 3 (1) (2016) 33. doi:10.1186/s40323-016-0085-5.  
URL <http://dx.doi.org/10.1186/s40323-016-0085-5>
- [32] N. Zander, M. Ruess, T. Bog, S. Kollmannsberger, E. Rank, Multi-level *hp*-adaptivity for cohesive fracture modeling, *International Journal for Numerical Methods in Engineering* 109 (13) (2017) 1723–1755. arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1002/nme.5340>, doi:10.1002/nme.5340.  
URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.5340>
- [33] M. Elhaddad, N. Zander, T. Bog, L. Kudela, S. Kollmannsberger, J. Kirschke, T. Baum, M. Ruess, E. Rank, Multi-level *hp*-finite cell method for embedded interface problems with application in biomechanics, *International Journal for Numerical Methods in Biomedical Engineering* 34 (4) (2018) e2951, e2951 cnm.2951. arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1002/cnm.2951>, doi:10.1002/cnm.2951.  
URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/cnm.2951>
- [34] S. Kollmannsberger, A. Özcan, M. Carraturo, N. Zander, E. Rank, A hierarchical computational model for moving thermal loads and phase changes with applications to selective laser melting, *Computers & Mathematics with Applications* 75 (5) (2018) 1483–1497. doi:<https://doi.org/10.1016/j.camwa.2017.11.014>.  
URL <http://www.sciencedirect.com/science/article/pii/S0898122117307289>
- [35] T. Bog, N. Zander, S. Kollmannsberger, E. Rank, Weak imposition of frictionless contact constraints on automatically recovered high-order, embedded interfaces using the finite cell method, *Computational Mechanics* 61 (4) (2018) 385–407. doi:10.1007/s00466-017-1464-6.  
URL <https://doi.org/10.1007/s00466-017-1464-6>
- [36] I. Babuška, W. C. Rheinboldt, A-posteriori error estimates for the finite element method, *International Journal for Numerical Methods in Engineering* 12 (10) (1978) 1597–1615. doi:10.1002/nme.1620121010.  
URL <http://dx.doi.org/10.1002/nme.1620121010>
- [37] P. Houston, B. Senior, E. Süli, *hp*-discontinuous Galerkin finite element methods for hyperbolic problems: error analysis and adaptivity, *Internat. J. Numer. Methods Fluids* 40 (1-2) (2002) 153–169, iCFD Conference on Numerical Methods for Fluid Dynamics (Oxford, 2001).  
URL <https://doi.org/10.1002/flid.271>
- [38] S. Congreve, P. Houston, Two-grid *hp*-DGFEM for second order quasilinear elliptic PDEs based on an incomplete Newton iteration, in: *Recent advances in scientific computing and applications*, Vol. 586 of *Contemp. Math.*, Amer. Math. Soc., Providence, RI, 2013, pp. 135–142.

- URL <https://doi.org/10.1090/conm/586/11629>
- [39] S. Giani, E. Hall, An a-posteriori error estimate for  $hp$ -adaptive DG methods for elliptic eigenvalue problems on anisotropically refined meshes, *Computing* 95 (1, suppl.) (2013) S319–S341.  
URL <https://doi.org/10.1007/s00607-012-0261-5>
  - [40] S. Giani, D. Schötzau, L. Zhu, An a-posteriori error estimate for  $hp$ -adaptive DG methods for convection-diffusion problems on anisotropically refined meshes, *Comput. Math. Appl.* 67 (4) (2014) 869–887.  
URL <https://doi.org/10.1016/j.camwa.2012.10.015>
  - [41] P. Houston, I. Perugia, D. Schötzau,  $hp$ -DGFEM for Maxwell’s equations, in: F. Brezzi, A. Buffa, S. Corsaro, A. Murli (Eds.), *Numerical mathematics and advanced applications*, Springer Italia, Milan, 2003, pp. 785–794.
  - [42] A. Cangiani, E. H. Georgoulis, P. Houston,  $hp$ -version discontinuous Galerkin methods on polygonal and polyhedral meshes, *Math. Models Methods Appl. Sci.* 24 (10) (2014) 2009–2041.  
URL <https://doi.org/10.1142/S0218202514500146>
  - [43] A. Cangiani, Z. Dong, E. H. Georgoulis, P. Houston,  $hp$ -version discontinuous Galerkin methods for advection-diffusion-reaction problems on polytopic meshes, *ESAIM Math. Model. Numer. Anal.* 50 (3) (2016) 699–725.  
URL <https://doi.org/10.1051/m2an/2015059>
  - [44] L. Demkowicz, J. Gopalakrishnan, A. H. Niemi, A class of discontinuous Petrov–Galerkin methods. Part III: Adaptivity, *Applied Numerical Mathematics* 62 (4) (2012) 396 – 427. doi:<http://dx.doi.org/10.1016/j.apnum.2011.09.002>.  
URL <http://www.sciencedirect.com/science/article/pii/S0168927411001656>
  - [45] S. Petrides, L. F. Demkowicz, An adaptive DPG method for high frequency time-harmonic wave propagation problems, *Computers & Mathematics with Applications* 74 (8) (2017) 1999 – 2017. doi:<https://doi.org/10.1016/j.camwa.2017.06.044>.  
URL <http://www.sciencedirect.com/science/article/pii/S0898122117304029>
  - [46] P. Binev, Instance optimality for  $hp$ -type approximation, *Oberwolfach Reports* 39 (2013) 14–16. doi:10.4171/OWR/2013/39.  
URL <https://doi.org/10.4171/OWR/2013/39>
  - [47] C. Canuto, R. H. Nochetto, R. Stevenson, M. Verani, Convergence and optimality of  $hp$ -afem, *Numer. Math.* 135 (4) (2017) 1073–1119.  
URL <https://doi.org/10.1007/s00211-016-0826-x>
  - [48] C. Erath, G. Gantner, D. Praetorius, Optimal convergence behavior of adaptive FEM driven by simple  $(h-h/2)$ -type error estimators, *arXiv e-prints* (2018) arXiv:1805.00715arXiv:1805.00715.
  - [49] W. Rachowicz, D. Pardo, L. Demkowicz, Fully automatic  $hp$ -adaptivity in three dimensions, *Comput. Methods Appl. Mech. Engrg.* 195 (37-40) (2006) 4816–4842. doi:10.1016/j.cma.2005.08.022.  
URL <http://dx.doi.org/10.1016/j.cma.2005.08.022>
  - [50] M. Bürg, W. Dörfler, Convergence of an adaptive  $hp$  finite element strategy in higher space-dimensions, *Applied Numerical Mathematics* 61 (11) (2011) 1132 – 1146. doi:<https://doi.org/10.1016/j.apnum.2011.07.008>.  
URL <http://www.sciencedirect.com/science/article/pii/S0168927411001267>
  - [51] C. Canuto, R. H. Nochetto, R. Stevenson, M. Verani, On  $p$ -robust saturation for  $hp$ -AFEM, *Comput. Math. Appl.* 73 (9) (2017) 2004–2022.  
URL <https://doi.org/10.1016/j.camwa.2017.02.035>
  - [52] P. Binev, Tree approximation for  $hp$ -adaptivity, *SIAM Journal on Numerical Analysis* 56 (6) (2018) 3346–3357. arXiv: <https://doi.org/10.1137/18M1175070>, doi:10.1137/18M1175070.  
URL <https://doi.org/10.1137/18M1175070>
  - [53] P. Daniel, A. Ern, I. Smears, M. Vohralík, An adaptive  $hp$ -refinement strategy with computable guaranteed bound on the error reduction factor, *Computers & Mathematics with Applications* 76 (5) (2018) 967 – 983. doi:<https://doi.org/10.1016/j.camwa.2018.05.034>.  
URL <http://www.sciencedirect.com/science/article/pii/S0898122118303109>
  - [54] S. Prudhomme, J. T. Oden, On goal-oriented error estimation for elliptic problems: application to the control of pointwise errors, *Comput. Methods Appl. Mech. Engrg.* 176 (1-4) (1999) 313–331. doi:10.1016/S0045-7825(98)00343-0.  
URL [http://dx.doi.org/10.1016/S0045-7825\(98\)00343-0](http://dx.doi.org/10.1016/S0045-7825(98)00343-0)
  - [55] J. T. Oden, S. Prudhomme, Goal-oriented error estimation and adaptivity for the finite element method, *Comput. Math. Appl.* 41 (5-6) (2001) 735–756. doi:10.1016/S0898-1221(00)00317-5.  
URL [http://dx.doi.org/10.1016/S0898-1221\(00\)00317-5](http://dx.doi.org/10.1016/S0898-1221(00)00317-5)



## Appendix A. 2D results with one dimensional solutions

This appendix describes additional 2D results: we consider the 1D examples of section 7.1 and we extend them to a 2D domain by considering a constant solution in  $y$  with homogeneous Neumann boundary conditions at the edges  $y = 0$  and  $y = 1$ . In the case of the shock problem, we now centered the shock around  $x = 0.5$  instead of  $x = 0.2$  as in the 1D example. The results are displayed in Figures A.20, A.21, and A.22. In every test case, we observe that the meshes seem optimal and as expected: one dimensional both in  $h$  and  $p$ , and the  $y$ -directional orders are constant and equal to 1. As for the other 2D example of section 7.2, the convergence is exponential. However, we observe that the singular solution A.21 does not provide exponential convergence. This examples serve to expose the limitation due to the lack of anisotropic refinements in  $h$ . Without such limitation, we would have been able to recover the exponential convergence rate as for the 1D case.

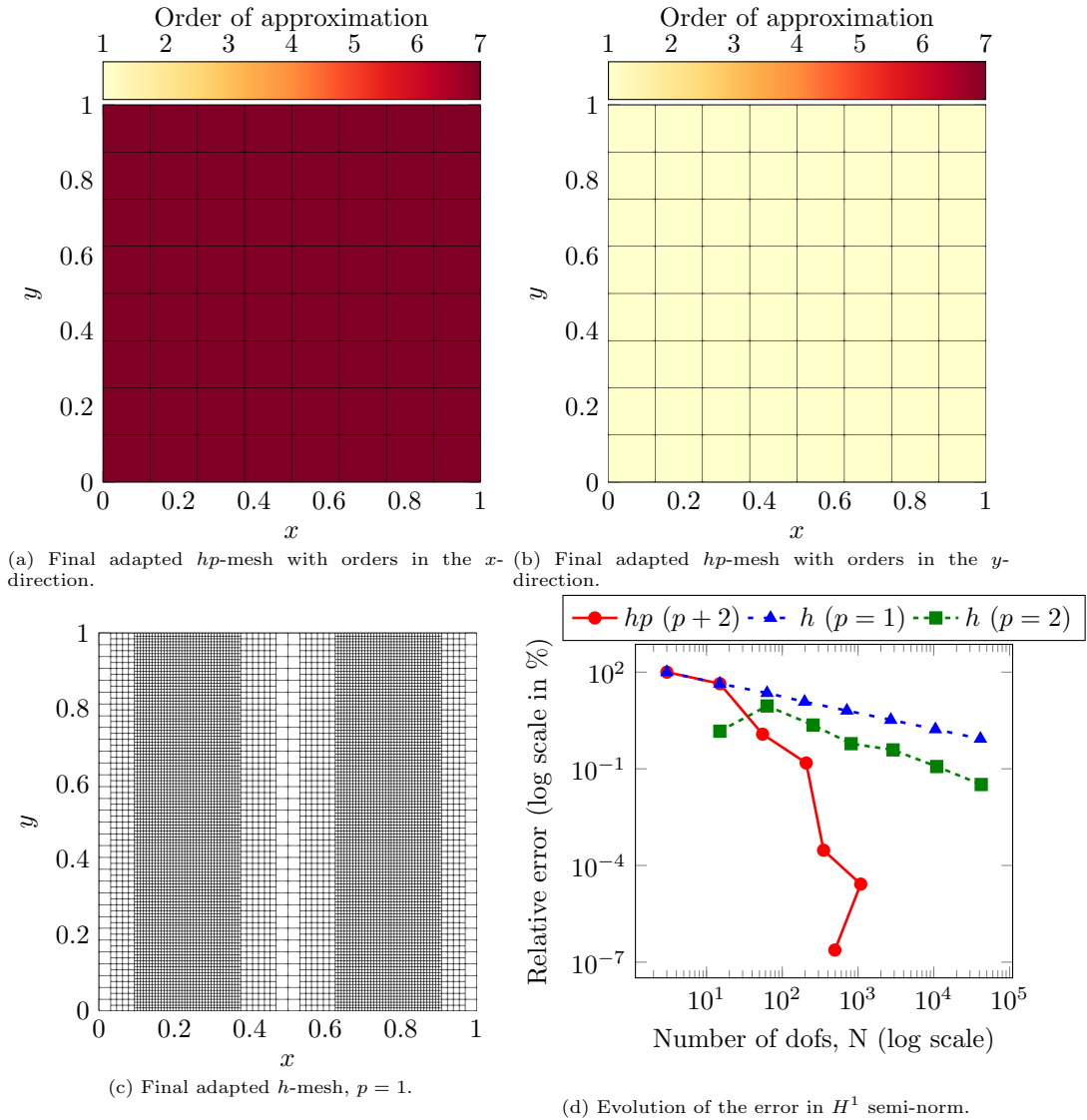


Figure A.20: One dimensional regular solution solved in a 2D mesh.

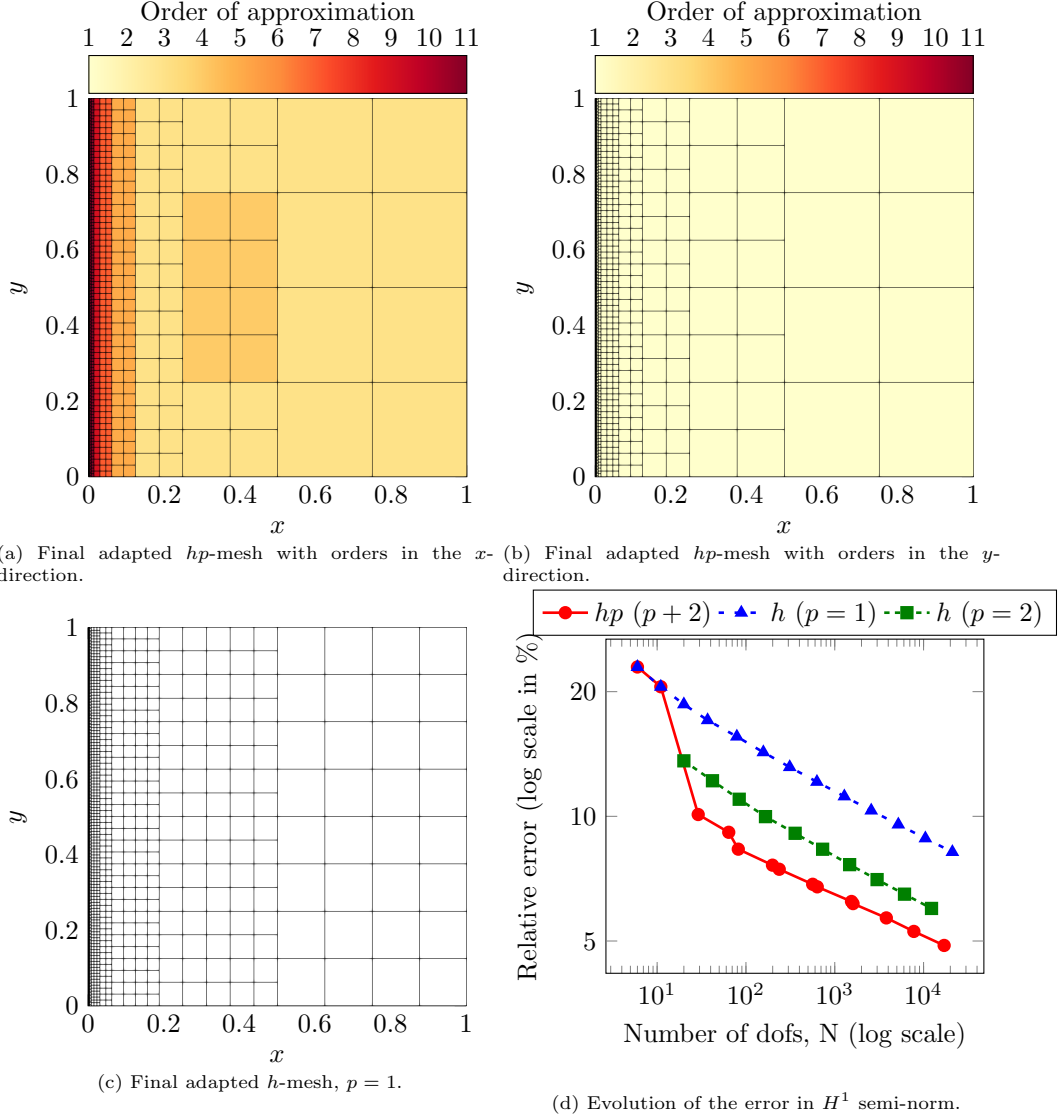


Figure A.21: One dimensional singular solution solved in a 2D mesh. The final adapted  $hp$ -mesh, Figures A.21a and A.21b, needs 17041 dofs to deliver an error of 4.878%; and the  $h$ -mesh, Figure A.21c, has an associated solution with error equal 5.993% for 12284 dofs.

## Appendix B. Different refinement strategy

This section is devoted to show the flexibility of the refinement step. In the following, instead of performing global refinements alternating between  $h$  and  $p + 2$  globally refined grids, as used in the previously exposed numerical results in Section 7, we  $h$ -refine half of the mesh (all descendants of two initial mesh elements) and  $p + 2$  refine the other half (all descendants of the other two initial elements), switching their role at the next refinement iteration. Such alternative refinement process illustrates one of the different options that one could employ when performing refinements. For example, it is also possible to refine only half of the mesh at each step and/or employ other alternatives suitable for a particular application. Some of them may considerably reduce the overall cost of the adaptive algorithm. Alternatively, one may also use a partially converged iterative solver to solve the fine grid problem (following a similar strategy as in [17]).

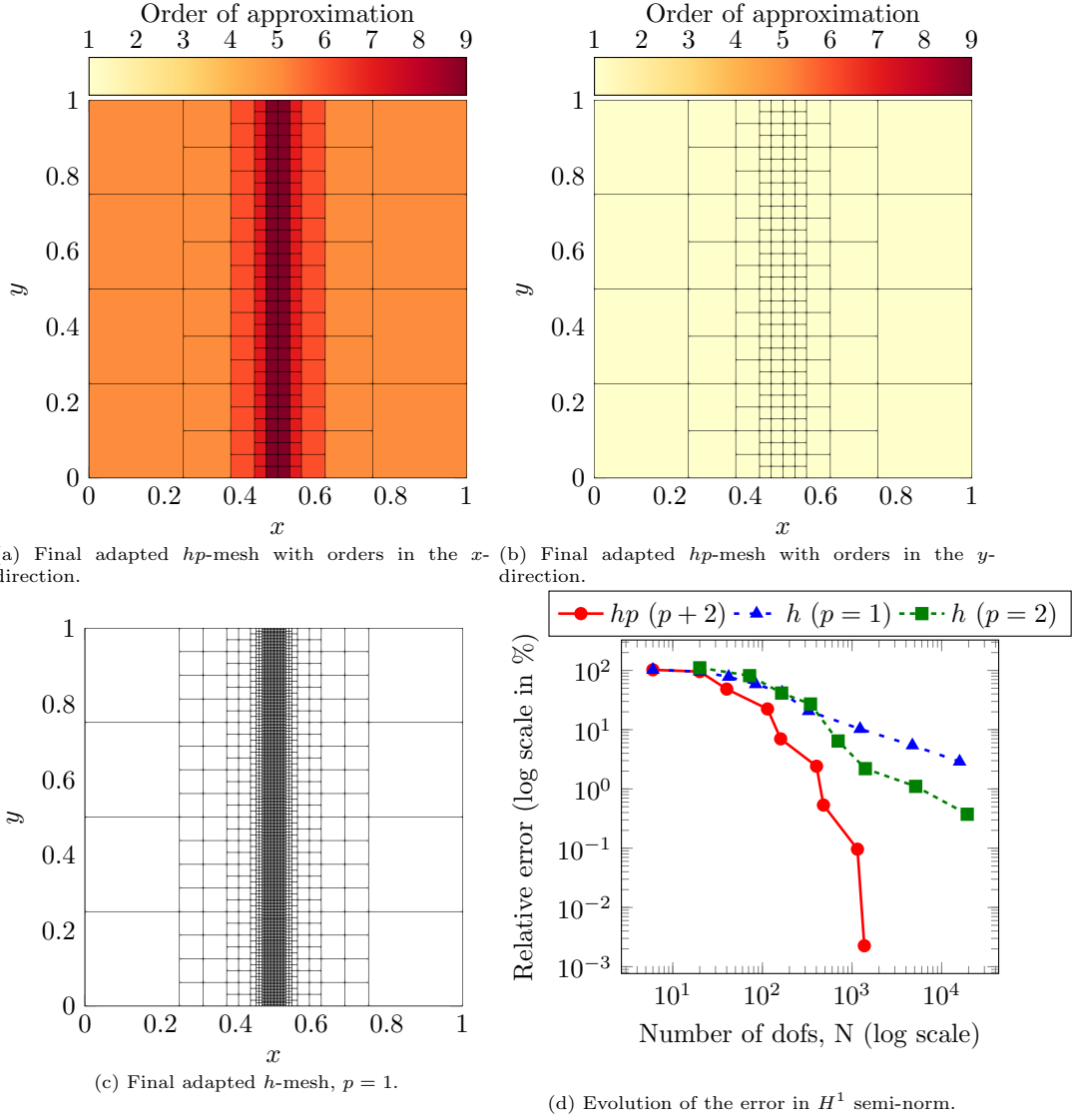
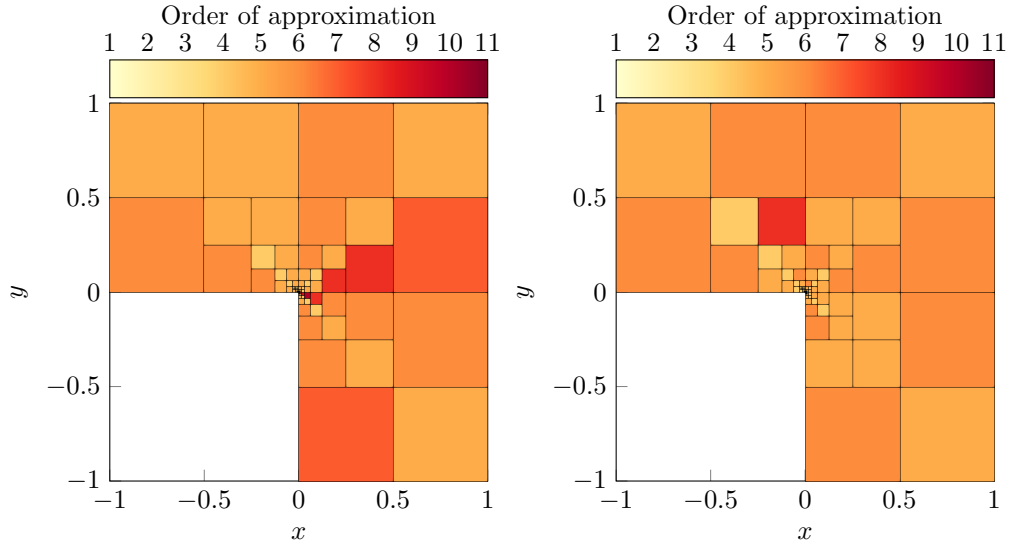
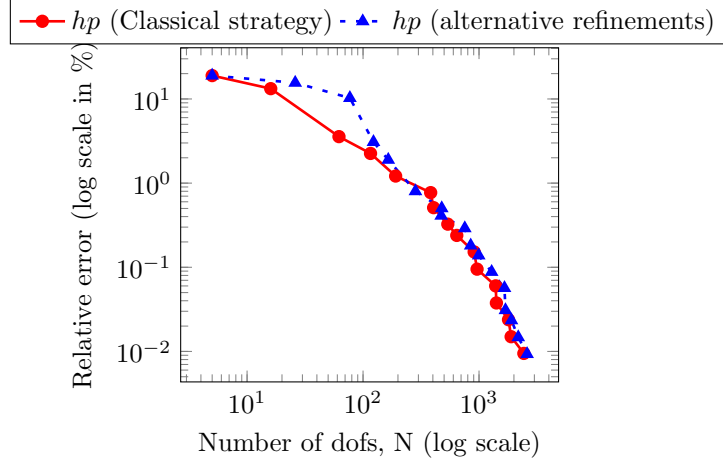


Figure A.22: One dimensional strong gradient solution solved in a 2D mesh. The final adapted  $hp$ -mesh, Figures A.22a and A.22b, needs 1372 dofs to deliver an error of  $2.24 \cdot 10^{-3}\%$ ; and the  $h$ -mesh, Figure A.22c, has an associated solution with error equal 0.374% for 19304 dofs.

We illustrate the aforementioned strategy with the 2D L-shape problem (Figure B.23) and the 2D shock problem (Figure B.24). The resulting convergence curves exhibit a similar behaviour as with the original refinement process.

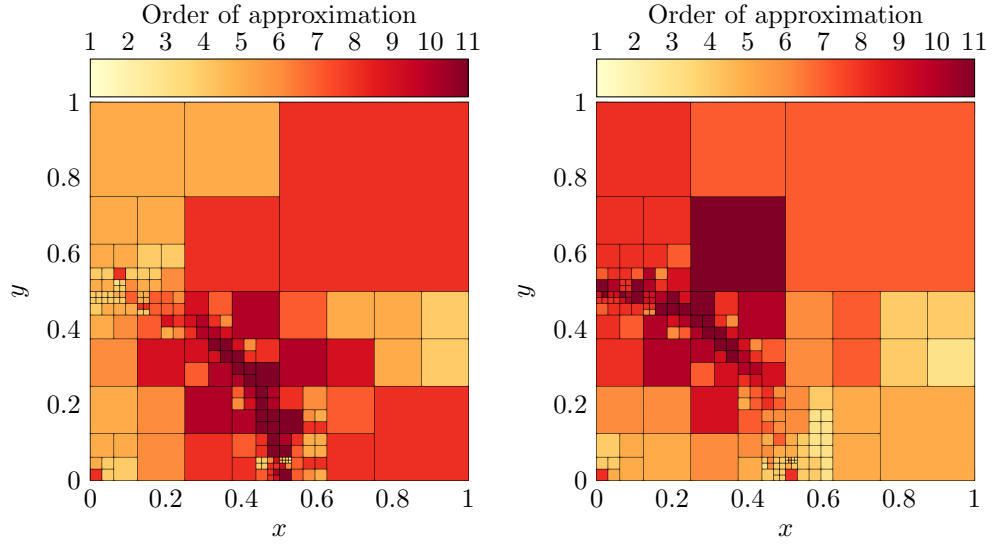


(a) Final adapted  $hp$ -mesh with orders in the  $x$ -direction. (b) Final adapted  $hp$ -mesh with orders in the  $y$ -direction.

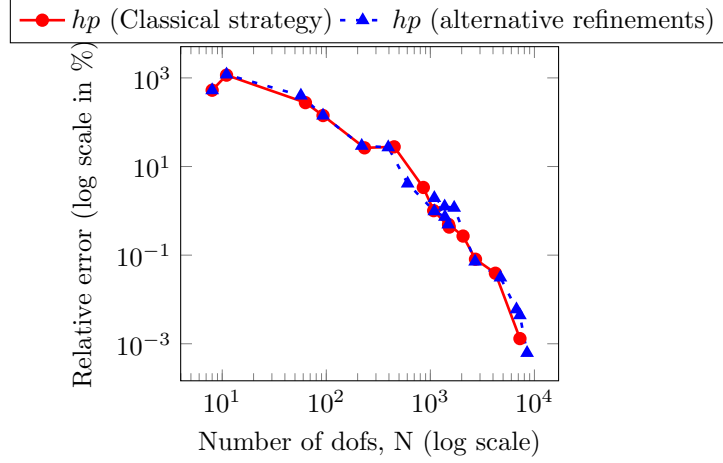


(c) Evolution of the error in  $H^1$  semi-norm.

Figure B.23: 2D L-shape problem. Alternative  $hp$ -adaptive refinement strategy where the global refinement steps combine simultaneously  $p$ - and  $h$ -refinements.



(a) Final adapted  $hp$ -mesh with orders in the  $x$ -direction. (b) Final adapted  $hp$ -mesh with orders in the  $y$ -direction.



(c) Evolution of the error in  $H^1$  semi-norm.

Figure B.24: 2D shock problem. Alternative  $hp$ -adaptive refinement strategy where the global refinement steps combine simultaneously  $p$ - and  $h$ -refinements.