A Painless Automatic hp-Adaptive Strategy for Elliptic Problems

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

A Painless Automatic *hp*-Adaptive Strategy for Elliptic Problems*

Vincent Darrigrand[a,*], David Pardo[a,b,c], Théophile Chaumont-Frelet[d,e], Ignacio Gómez-Revuelto[f], Luis Emilio Garcia-Castillo[f]

[a] *University of the Basque Country (UPV-EHU), Leioa, Spain*
[b] *Basque Center for Applied Mathematics (BCAM), Bilbao, Spain*
[c] *Ikerbasque, Bilbao, Spain*
[d] *INRIA Sophia Antipolis, Méditerranée Research Centre, Nachos Project Team, France*
[e] *University of Nice, J. A. Dieudonné Mathematics Laboratory (UMR CNRS 6621), Nice, France*
[f] *Departamento de Teoria 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 quasi-optimal 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 require 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- and two-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 information on the solution is known *a priori*, it is essential to have at our disposal automatic adaptive algorithms.

*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 Sklodowska-Curie grant agreement N° 777778.

*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), luise@tsc.uc3m.es (Luis Emilio Garcia-Castillo)

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). 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 lack of 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].

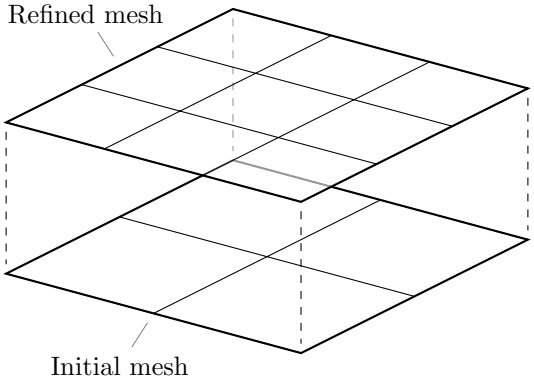
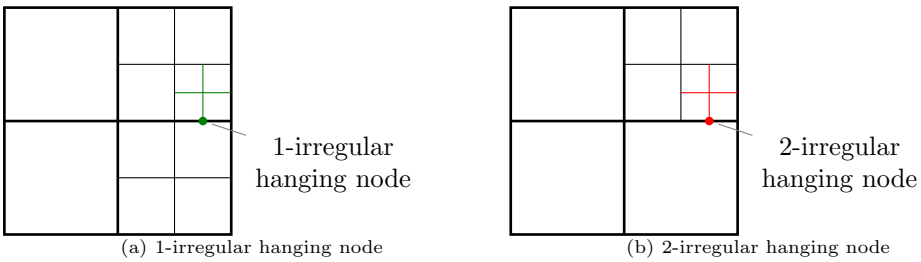


Figure 1: Re-meshing without hierarchy.



(a) 1-irregular hanging node (b) 2-irregular hanging node

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

In 2015, Rank and his collaborators proposed new data structures for supporting hp-discretizations [9, 10]. Using a multi-level approach and hierarchical basis functions both in h and in p, they perform global uniform refinements and massively 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 massive 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, and following the *no free lunch* theorem, 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.

On this work, and driven by our "simple-to-implement" goal, we adopt the data structures proposed by Rank and his collaborators [9, 10], and we focus on developing an automatic and relatively simple *hp*-adaptive algorithm under these new data structures.

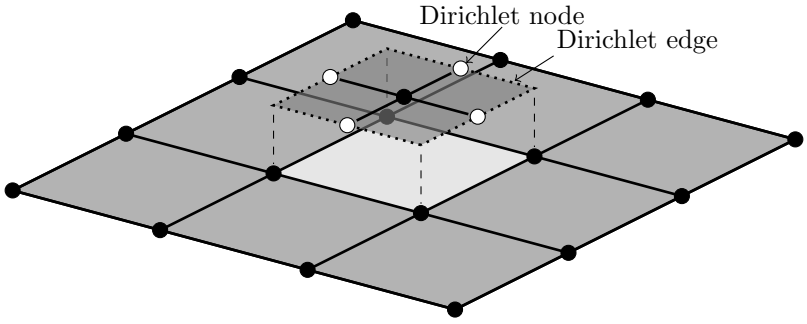


Figure 3: Multi-level 2D mesh without constraints on hanging nodes using Dirichlet nodes, see [9] for details.

There exists a variety of refinement-based *hp*-adaptive algorithms. Amongst others, we shall mention the works of: (a) Ainsworth et al. [11], 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* [12] that performs first an h-adaptive step followed by a p-adaptive one that leads to suboptimal results; (c) the work of Demkowicz et al. presented in [3, 4, 13] and applied in several contexts, e.g. [14–25], that produces almost optimal meshes but requires 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. [26], which estimates the regularity of the solution with the Legendre coefficients [27] and, (e) the contribution of Rank et al. [28] and applications [29–31], which combine their multi-level data structure [9, 10] with a classic residual-based estimator [32]. We refer to [27] 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 eco-system of *hp*-adaptive Discontinuous Galerkin (DG) method, [33–41] that requires a complex and specific implementation, which 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 Rank et. al. For example, the local projections considered by Demkowicz et. al [3] in the context of the 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 what we exploit in this work.

Our main contribution is an easy-to-implement (and robust) *hp*-adaptive strategy. The main idea is that given a mesh, we perform quasi-optimal unrefinements by taking advantage of the hierarchical data structures proposed by Rank et al. [9, 10]. We employ quadrilateral elements and select some of the basis functions that can be removed without losing the completeness of the approximation space to perform an unrefinement step (i.e., coarsening a given mesh similarly to [42, 43]). 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 et. al. [3].

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, (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 this work, we focus on elliptic problems. 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 and 2D 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.

2. Definitions

2.1. Abstract variational formulation

Given a domain $\Omega \subset \mathbb{R}^d$, where d 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:

> Find $u \in \mathbb{H}$ such that
>
> $$b(u,\phi) = f(\phi) \quad \forall \phi \in \mathbb{H}, \tag{1}$$

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 $\mathbb{H}$.

2.2. Discretization

Let $\mathcal{T}$ be a partition of Ω 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 ϕ is a product of i 1D linear functions and $d-i$ bubble functions where $i \in (0, \cdots, d)$.

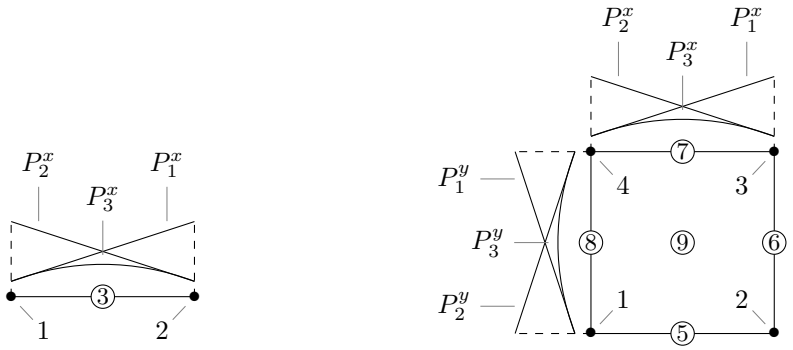
Nodes: Following the notation of [3], we define an abstraction of a node e as a component of an element K that refers to either a vertex, an edge, a face, or a volume. We define the dimension $d_e \in \{0, \cdots, d\}$ of e according to the dimension of the component it denotes: $d_e = 0$ for a vertex, $d_e = 1$ 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 Ω.

Node directions: A node e of dimension $d_e \neq 0$ has a set $\mathcal{D}_e \subset \{1, \cdots, d\}$ of d_e directions. If the node e is a vertex, $d_e = 0$, then we set $\mathcal{D}_e \coloneqq \varnothing$. 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.



(a) 1D element (b) 2D element

Figure 4: Nodes distribution and basis functions in one element

Node orders: For a given node e of dimension $d_e \in \{1, \cdots, d\}$ 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 $p = (p_1, \cdots, p_{d_e}) \in (\mathbb{N}\backslash\{0\})^{d_e}$. 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.

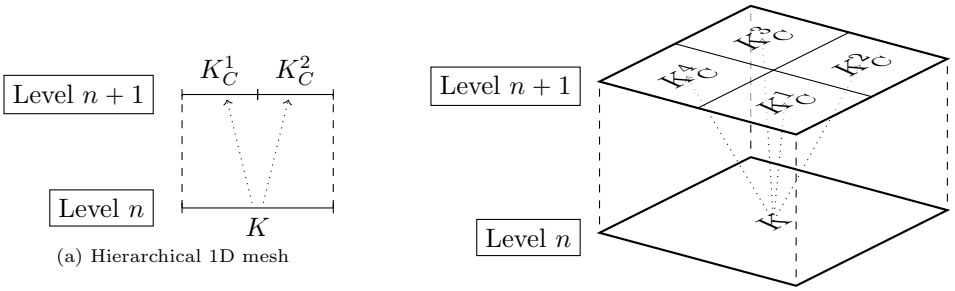
Element orders: The orders of an element $K_a \in \mathcal{T}$, denoted as $\mathcal{O}(K_a)$, is a set that gathers the orders of all its nodes.

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 $\mathcal{L} : K \longmapsto n \in \mathbb{N}$ returns the element level.
- *Children:* If K has been refined, then it admits a set of N_{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 $\mathcal{C}(K) = \varnothing$ 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 $\mathcal{S} \coloneqq \mathcal{C} \circ \mathcal{P}$. Note that if K is a root element, we set $\mathcal{S}(K) \coloneqq \{K\}$.

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



(a) Hierarchical 1D mesh

(b) Hierarchical 2D mesh

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^{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.

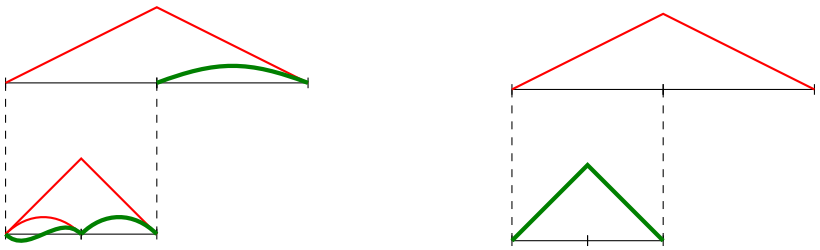
3. Removable basis functions

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 $\mathbb{H}_{\mathcal{T}}$. We denote them as *removable* basis functions (Rm_{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 in order to preserve the *completeness* of the Galerkin 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.

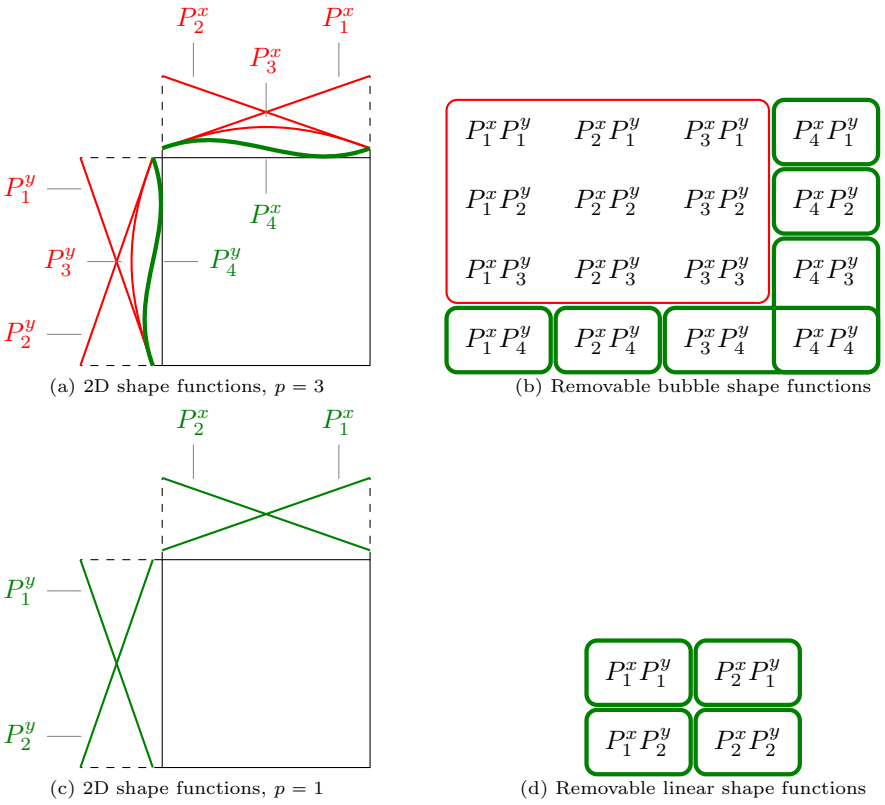
Removable basis functions: We define the set of removable basis functions of a node, namely $\text{Rm}_{\text{basis}}(e_p)$, as follows:

- 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 bubble functions whose cartesian product contains the highest degree in the direction i.



(a) Bubble basis functions (b) Linear basis function

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



(a) 2D shape functions, $p = 3$ (b) Removable bubble shape functions

(c) 2D shape functions, $p = 1$ (d) Removable linear shape functions

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

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

$$\mathrm{Rm}_{\mathrm{basis}}(e_p) \coloneqq \bigcup_{i \in \mathcal{D}_e} \mathrm{Rm}_{\mathrm{basis}}(e_p, i).$$

Remark. Since we want to generate a complete subspace of $\mathbb{H}_{\mathcal{T}}$, we cannot remove most of the basis functions existing in a given mesh. In particular, a basis function is considered as non-removable if any of the elements of its support contains a higher degree basis function.

Remark. When performing h-adaptivity with any order, we need to extend the definition of removable linear basis function: we consider as removable solely the linear basis functions whose support is on a penultimate level element *independently* of the order of its children.

Notations around removable basis functions: For the ease of the presentation, we introduce the fol-

lowing notations.

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

4. Energy contribution indicators

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

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

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 $\widetilde{u} \in \mathbb{H}_{\mathcal{T}}$, we define its energy contribution as:

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

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

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

$$\begin{aligned}\widetilde{u}_e &\coloneqq \sum_{\phi_j \in \mathrm{Rm}_{\mathrm{basis}}(e)} u_j \phi_j. \\ \eta_e^{\mathcal{T}} &\coloneqq \mathcal{R}(\widetilde{u}_e)\end{aligned}$$

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

$$\begin{aligned}\widetilde{u}_{e,i} &\coloneqq \sum_{\phi_j \in \mathrm{Rm}_{\mathrm{basis}}(e,i)} u_j \phi_j. \\ \eta_{e,i}^{\mathcal{T}} &\coloneqq \mathcal{R}(\widetilde{u}_{e,i})\end{aligned}$$

- We define the element-wise error isotropic and anisotropic indicators of $K_a \in \mathcal{T}^{\mathrm{rm}}{}_K$ by accumulating the node-wise error indicator of the element:

$$\begin{aligned}\eta_{K_a}^{\mathcal{T}} &\coloneqq \frac{1}{n_{\mathrm{Rm}_{\mathrm{basis}}(K_a^{\mathrm{rm}})}} \sum_{e \in K_a^{\mathrm{rm}}} \eta_e^{\mathcal{T}}, \\ \eta_{K_a,i}^{\mathcal{T}} &\coloneqq \frac{1}{n_{\mathrm{Rm}_{\mathrm{basis}}(K_{a,i}^{\mathrm{rm}},i)}} \sum_{e \in K_{a,i}^{\mathrm{rm}}} \eta_{e,i}^{\mathcal{T}}, \quad \forall i \in \{1, \cdots, d\}.\end{aligned}$$

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

$$\begin{aligned}\left|J(u_{\widetilde{\mathcal{T}}}) - J(u_{\mathcal{T}})\right| - |J(u_{\mathcal{T}} - \widetilde{u}) - J(u_{\mathcal{T}})| \\ = J(u_{\widetilde{\mathcal{T}}}) - J(u_{\mathcal{T}}) - J(u_{\mathcal{T}} - \widetilde{u}) + J(u_{\mathcal{T}}) \\ = J(u_{\widetilde{\mathcal{T}}}) - J(u_{\mathcal{T}} - \widetilde{u})\end{aligned}$$

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

$$\left|J(u_{\widetilde{\mathcal{T}}}) - J(u_{\mathcal{T}})\right| \leqslant |J(u_{\mathcal{T}} - \widetilde{u}) - J(u_{\mathcal{T}})|\,.$$

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

Remark. The indicators are local computable quantities. Indeed, each $\widetilde{u}_e$ involves a limited number of basis functions, and by definition of the residual $\mathcal{R}$, the nodal indicator $\mathcal{R}(\widetilde{u}_e)$ 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:

$$W^{\mathcal{T}}_{\text{avg}} = \frac{1}{n_{\mathcal{T}^{\text{rm}}_K}} \sum_{K_a \in \mathcal{T}^{\text{rm}}_K} \eta^{\mathcal{T}}_{K_a}.$$

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

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

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.

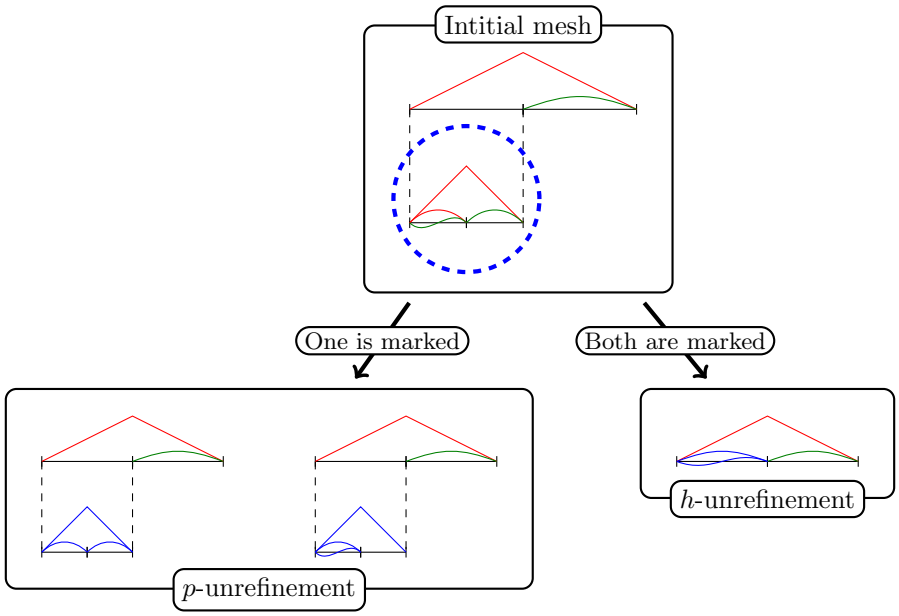


Figure 8: 1D *hp*-unrefinements

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.

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.

Algorithm 1: Firing policy (coarsening step)

Inputs: $\mathcal{T}_0$ given mesh; $\alpha_p, \alpha_h \geqslant 0$.
Output: $\mathcal{T}$ coarsened mesh
$\mathcal{T} \coloneqq \mathcal{T}_0$ initialize the mesh to be unrefined.
do
  Solve: solve the problem on $\mathcal{T}$.
  Estimate: compute the indicators $\eta^{\mathcal{T}}_{K_a}, \forall K_a \in \mathcal{T}^{\mathrm{rm}}$ and the average $W^{\mathcal{T}_0}_{\mathrm{avg}}$.
  $p-$Mark: $\forall K_a \in \mathcal{T}^{\mathrm{rm}} \backslash \mathcal{T}^{\mathrm{rm}}_h$, $\forall i \in \{1, \cdots, d\}$,
    if $\left|\eta^{\mathcal{T}}_{K_a,i}\right| \leqslant \alpha_p \left|W^{\mathcal{T}_0}_{\mathrm{avg}}\right|$
    then mark K_a for $p-$unrefinement in the direction i.
  $h-$Mark: $\forall K_a \in \mathcal{T}^{\mathrm{rm}}$ such that $\mathcal{S}(K_a) \subset \mathcal{T}^{\mathrm{rm}}$,
    if $\left|W^{\mathcal{T},\mathcal{S}(K_a)}_{\mathrm{avg}}\right| \leqslant \alpha_h \left|W^{\mathcal{T}}_{\mathrm{avg}}\right|$,
    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

Algorithm 2: Adaptive process

Inputs: $\mathcal{T}$ 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

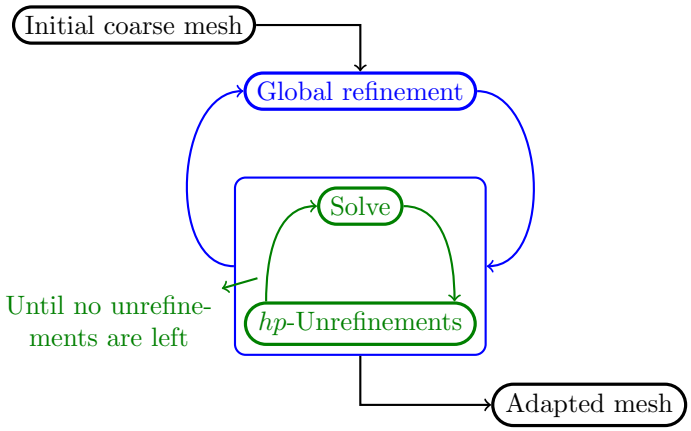


Figure 9: Adaptive Algorithm

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 is suitable for large-scale parallelization.

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 $\Gamma_D \cap \Gamma_N = \varnothing$:

> Find u such that,
>
> $$\begin{cases} -\Delta u = f & \text{in } \Omega, \\ u = 0 & \text{on } \Gamma_D, \\ u' = g & \text{on } \Gamma_N, \end{cases}$$

where f, g and the definitions of Γ_D and Γ_N are set so that u fits certain given manufactured solutions.

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 against *h*-adaptivity with uniform $p = 1$ and $p = 2$.

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.

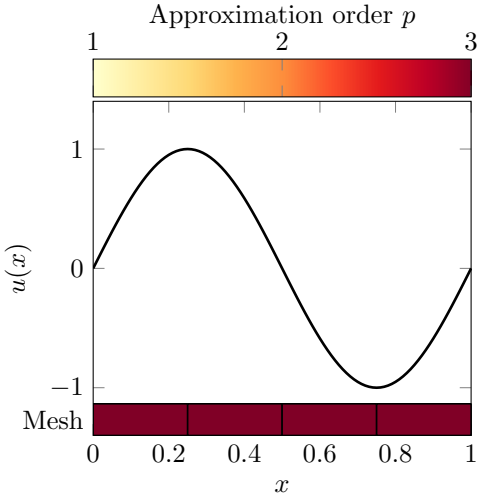
1. **Regular Solution**: $u = \sin(2\pi x)$, $\Gamma_D = \{0\} \cup \{1\}$, $\Gamma_N = \varnothing$. The smoothness of the solution is captured with a coarse mesh with uniform relatively elevated orders ($p = 3$), as shown in Figure 10. We need 11 dofs to reach 1% error.

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 the singularity, while in the next two elements it decreases quickly to $p = 3$, and the rest are set to $p = 4$.

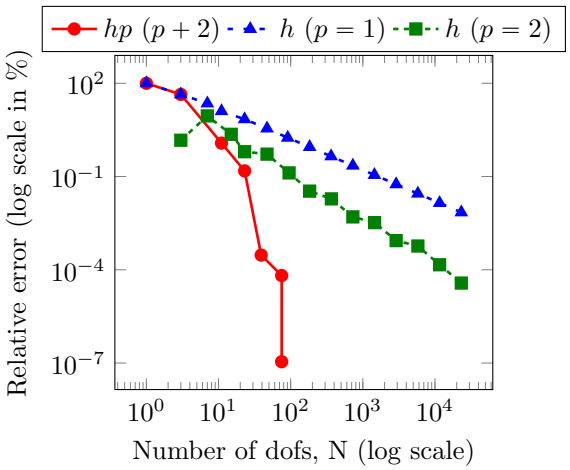
   This suboptimal high order at the singularity is due to the alternated refinements we performed and because the contribution of the highest order basis function at the singularity is above average so the unrefinement algorithm will not remove it. To reach 1% of error, our algorithm needs around $N = 104$ 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**: $u = \operatorname{atan}\left(a\left(x - \frac{1}{5}\right)\right) + \operatorname{atan}\left(\frac{a}{5}\right)$ 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 et al. [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]).

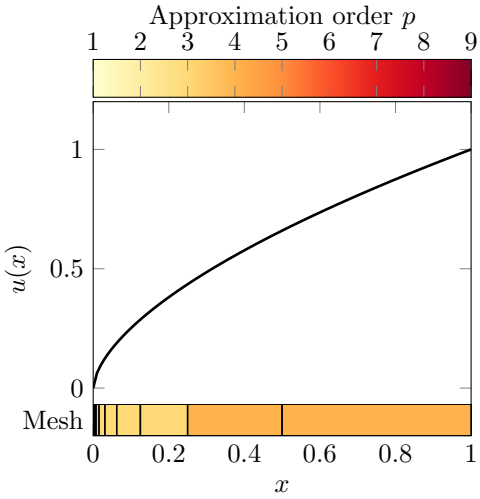


(a) Solution and adapted hp-mesh. The drawn mesh needs 11 dofs to deliver 1.19% error.

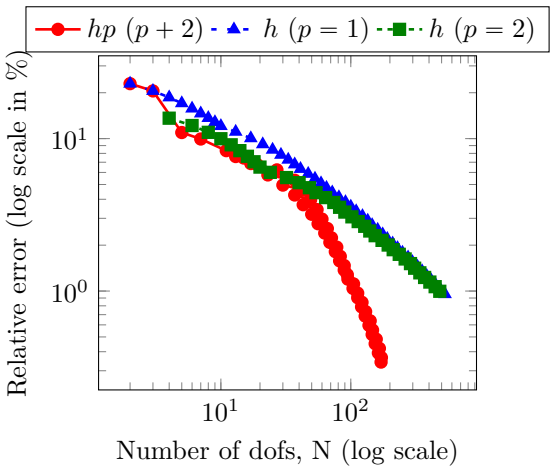


(b) Evolution of the error in H^1 semi-norm.

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

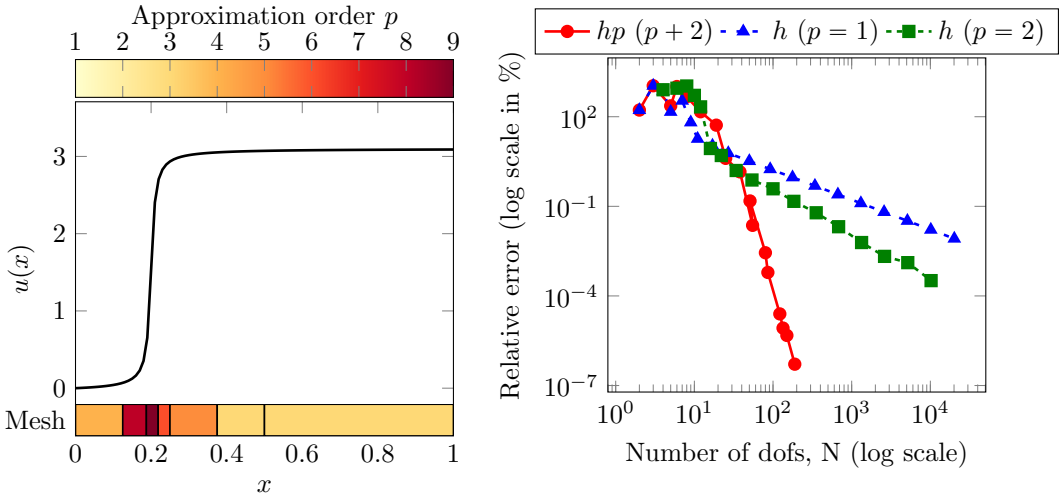


(a) Solution and adapted hp-mesh. The drawn mesh needs 104 dofs to deliver 1.04% error.



(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. $u = \operatorname{atan}\left(a\left(x - \frac{1}{5}\right)\right) + \operatorname{atan}\left(a \cdot \frac{1}{5}\right)$, $a = 120$.

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.

1. **Regular Solution**: $u = \sin(2\pi x)\sin(2\pi y)$, on $\Omega = (0,1)^2$ with $\Gamma_D = \partial\Omega$, $\Gamma_N = \varnothing$. We display the results in Figure 13.

2. **Reentrant corner**: $u = r^{\frac{2}{3}}\sin(\frac{2}{3}(\theta + \frac{\pi}{2}))$, $r = \sqrt{x^2+y^2}$, $\theta = \operatorname{atan2}(y,x)$ on $\Omega = (-1,1)^2\backslash(-1,0)^2$ with $\Gamma_D = \{0\} \times (-1,1) \cup (-1,1) \times \{0\}$ and $\Gamma_N = \partial\Omega\backslash\Gamma_D$. The singularity in H^1 of this solution comes from the non convexity of the domain. Figure 14 shows competitive convergence rates with quasi-optimal meshes.

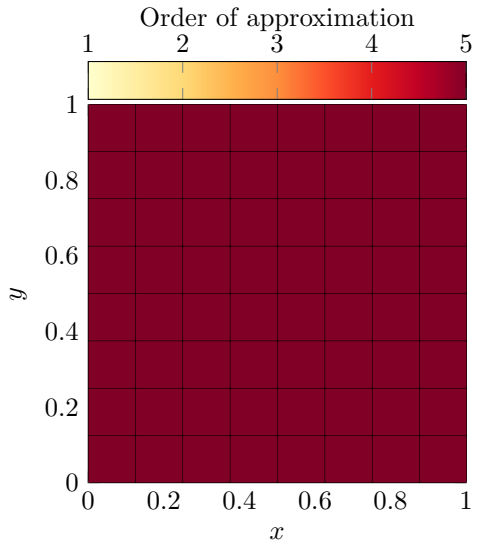
3. **Shock problem**: $u = \operatorname{atan}(a(r - \frac{1}{2}) - \operatorname{atan}(-\frac{a}{2}))$, $a = 60$ and $r = \sqrt{x^2+y^2}$, on $\Omega = (0,1)^2$ with $\Gamma_D = \{(0,0)\}$ and $\Gamma_N = \partial\Omega\backslash\Gamma_D$. The solution exhibits a strong gradient around $r = 0.5$. Figure 15 gathers the adapted meshes and convergence history.

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

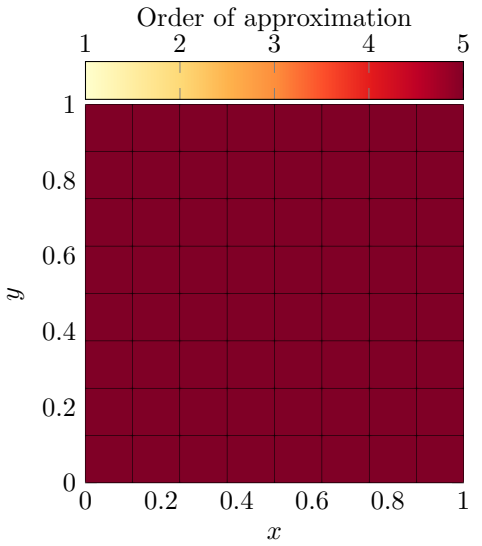
8. Conclusions

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

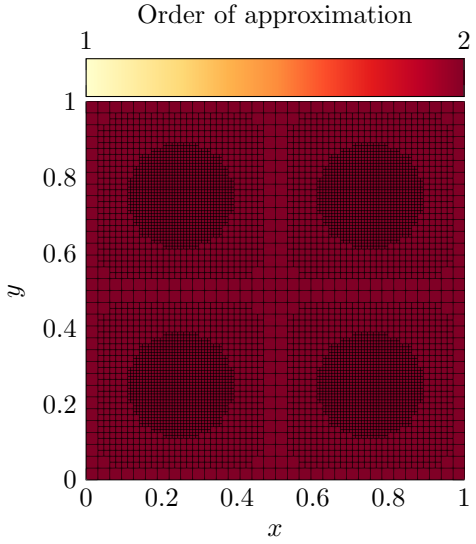
We present a coarsening strategy that relies on the contribution of each *removable* basis function. If its contribution is small, then it is marked for unrefinement. Such unrefining policy does not rely on the initial mesh, thus providing great flexibility to build a fine mesh candidate for coarsening. 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.



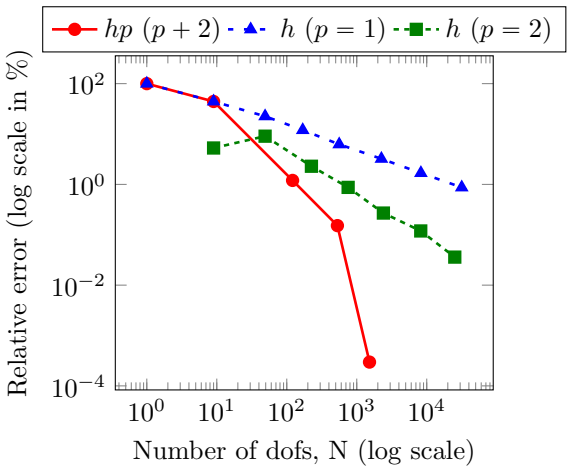
(a) Final adapted hp-mesh with orders in the x-direction.



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



(c) Final adapted h-mesh, $p = 2$.



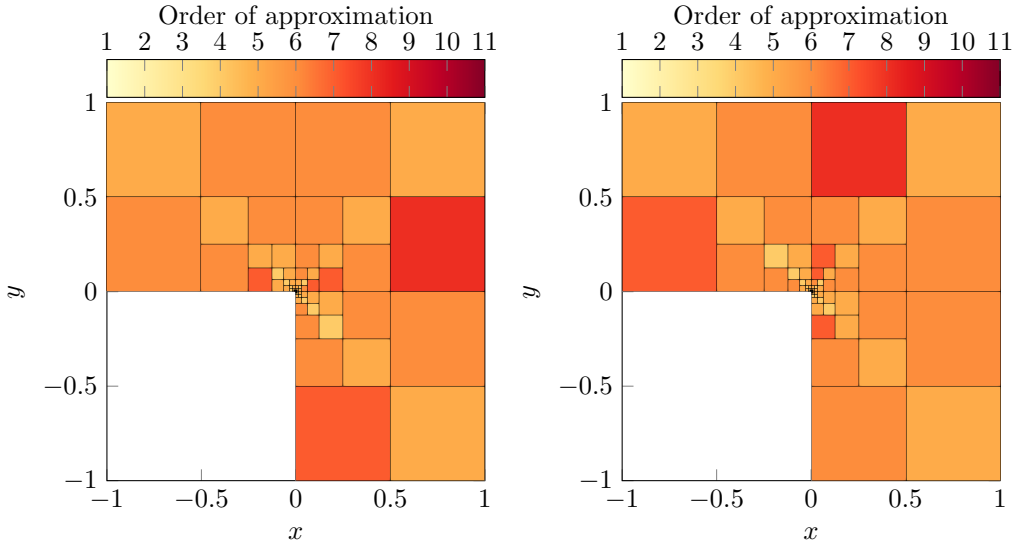
(d) Evolution of the error in H^1 semi-norm.

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 $3.6 \cdot 10^{-2}$% for 25153 dofs. Additionally, for 121 dofs, we obtain an error of 1.199%.

We demonstrate using standard academic test problems that our method produces quasi-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.

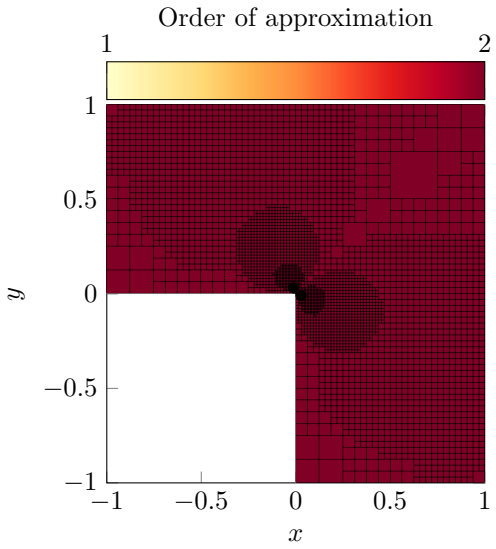
The main advantages are both the painless implementation efforts and the flexibility of the refinement strategy. However, we identified some limitations to our method: (a) the definition of the multi-level data structure forces isotropic h-(un)refinements, (b) we are yet limited to elliptic problems due to the definition of the indicators, and (c) the lack of orthogonality of the basis functions may lead to suboptimal results.

The continuation of this work includes: (a) a 3D demonstration of the method, (b) the extension to non-elliptic problems perhaps via the use of a residual minimization method for providing an adequate

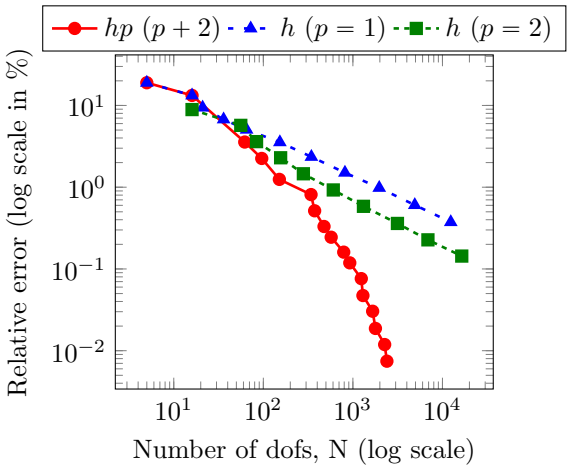


(a) Final adapted hp-mesh with orders in the x-direction.

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



(c) Final adapted h-mesh, $p = 2$.



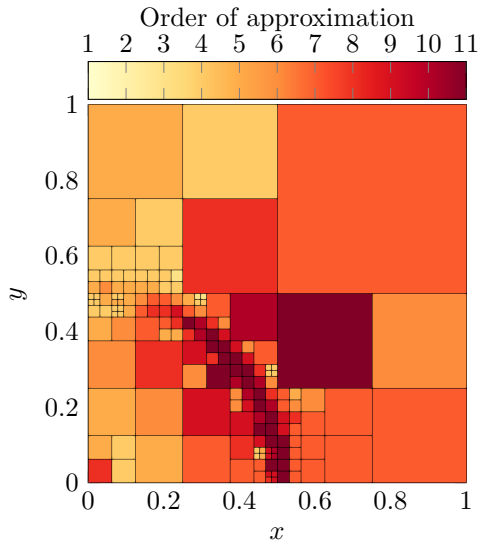
(d) Evolution of the error in H^1 semi-norm.

Figure 14: 2D singular reentrant corner problem. The final adapted hp-mesh, Figures 14a and 14b, needs 2397 dofs for delivering an error of $7.43 \cdot 10^{-3}$%; and the h-mesh, Figures 14c, delivers an error of 0.144% for 16428 dofs. Additionally, for 341 dofs, we obtain an error of 0.812%. The algorithm of Demkowicz et al [3] needs between 277 and 417 dofs to reach 1% of error.

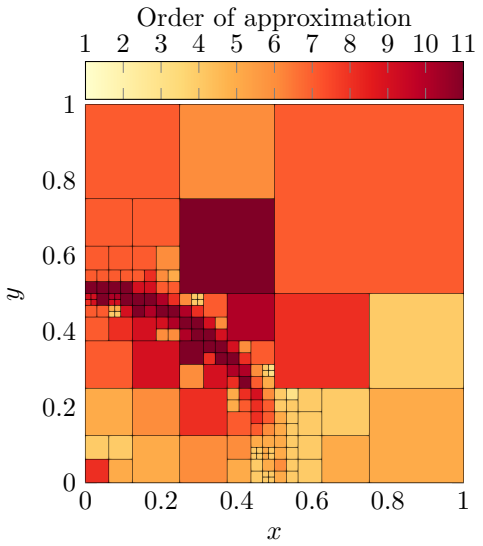
error representation, (c) the use of triangular/tetrahedral meshes, and (d) the development of goal-oriented indicators following a similar element-wise strategy as in [44, 45].

References

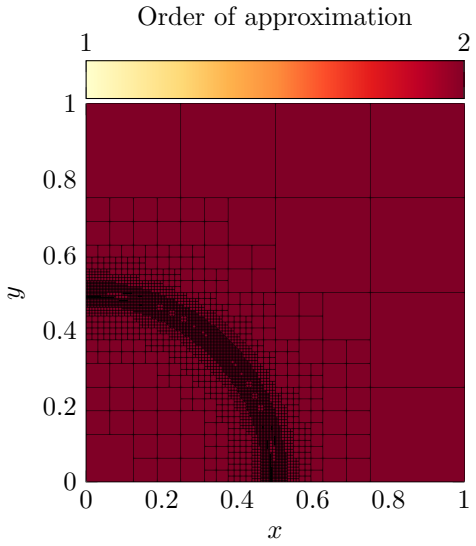
[1] I. Babuska, B. Szabo, I. Katz, The p-version of the finite element method, SIAM Journal on Numerical Analysis 18 (3) (1981) 515–545. arXiv:https://doi.org/10.1137/0718033, doi:10.1137/0718033.
URL https://doi.org/10.1137/0718033



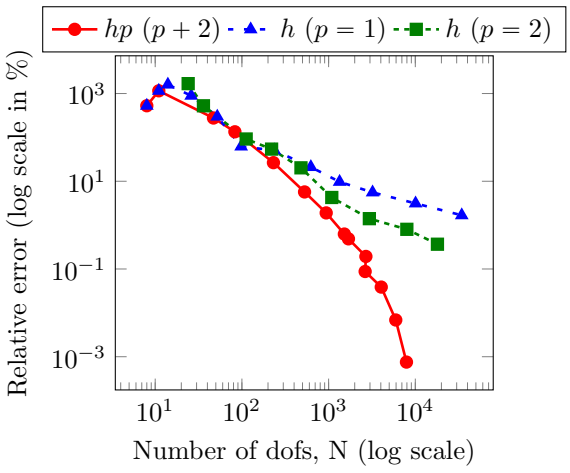
(a) Final adapted *hp*-mesh with orders in the x-direction.



(b) Final adapted *hp*-mesh with orders in the y-direction.



(c) Final adapted h-mesh, $p = 2$.



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

Figure 15: 2D shock problem. The final adapted *hp*-mesh, Figures 15a and 15b, needs 7908 dofs for delivering an error of $7.5 \cdot 10^{-4}$%; and the h-mesh, Figures 15c, delivers an error of 0.367% for 18004 dofs. Additionally, for 938 dofs, we obtain an error of 1.899%. The algorithm of Demkowicz et al [3] needs between 1261 and 1901 dofs to reach 1% of error.

[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, 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 Methods 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] 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.
URL http://www.sciencedirect.com/science/article/pii/S0168927497000834
[12] 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.
URL http://www.sciencedirect.com/science/article/pii/004578259400705R
[13] 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
[14] 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
[15] 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
[16] 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
[17] 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
[18] 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
[19] 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) (2008) 3039–3049.
[20] 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
[21] 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
[22] 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
[23] 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.
[24] 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.
URL http://www.sciencedirect.com/science/article/pii/S1877050913004675

[25] 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

[26] 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.

[27] 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

[28] 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

[29] 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

[30] 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

[31] 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

[32] 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

[33] 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/fld.271

[34] 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

[35] 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

[36] 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

[37] 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.

[38] 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

[39] 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

[40] 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

[41] 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

[42] 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

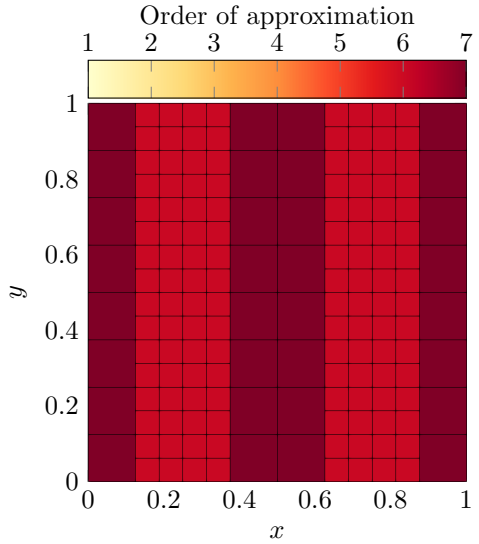
[43] C. Canuto, R. H. Nochetto, R. Stevenson, M. Verani, Convergence and optimality of *hp*-afem, Numer. Math. 135 (4) (2017) 1073–1119.
URL https://doi.org/10.1007/s00211-016-0826-x

[44] 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

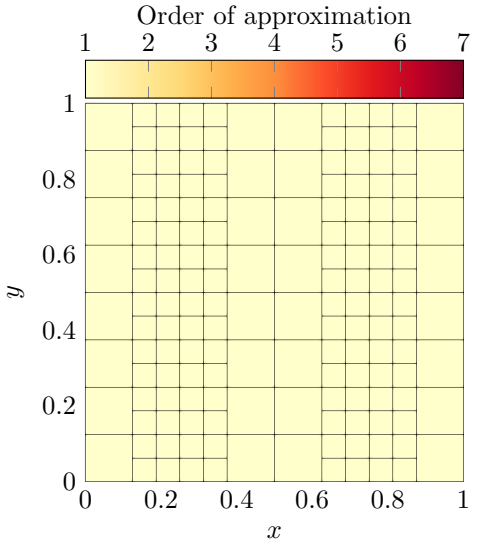
[45] J. T. Oden, S. Prudhomme, Goal-oriented error estimation and adaptivity for the finite element method, Comput. Math. Appl. 41 (5-6) (2001) 735–756. doi:10.1016/S0898-1221(00)00317-5.
URL http://dx.doi.org/10.1016/S0898-1221(00)00317-5

Appendix A. 2D results with one dimensional solutions

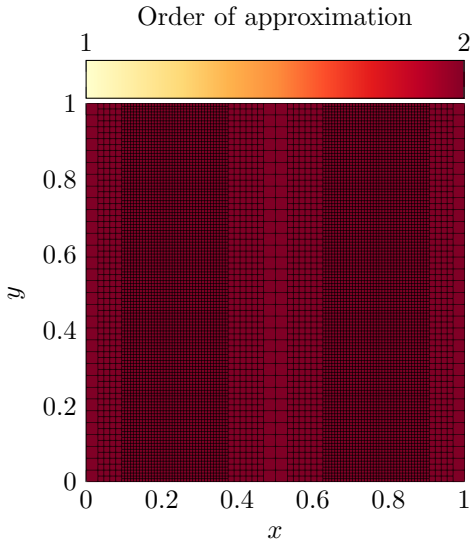
This appendix describes additional 2D results: we consider the 1D examples of section 7.1 and we extend them to a 2D domain by considering a constant solution in y with homogeneous Neumann boundary conditions at the edges $y = 0$ and $y = 1$. In the case of the shock problem, we now centered the shock around $x = 0.5$ instead of $x = 0.2$ as in the 1D example. The results are displayed in Figures A.16, A.17, and A.18. 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.17 does not provide exponential convergence. Indeed, we see here the limitation of isotropic refinements in h.



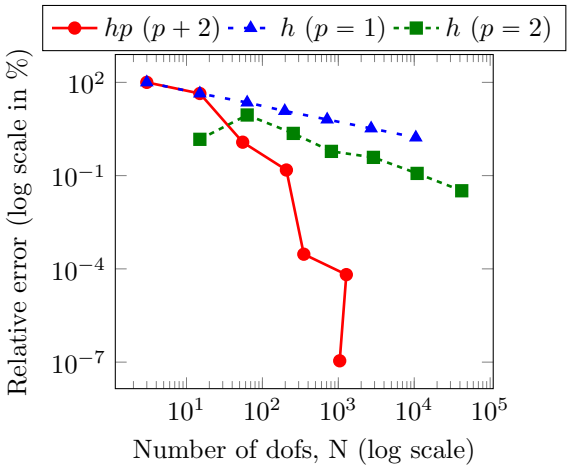
(a) Final adapted hp-mesh with orders in the x-direction.



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

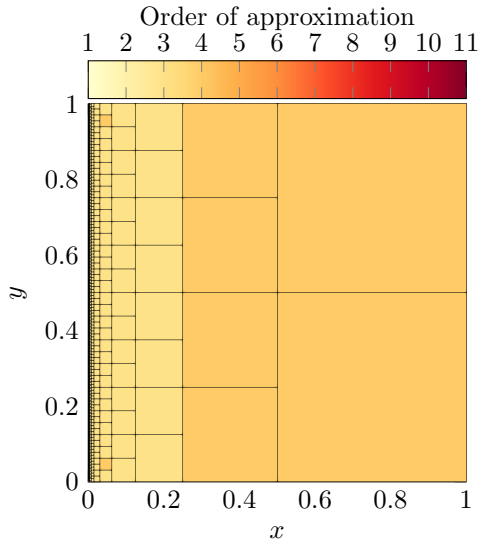


(c) Final adapted h-mesh, $p = 2$.

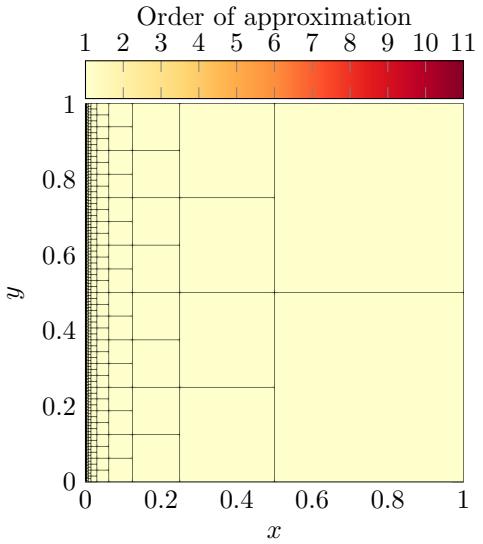


(d) Evolution of the error in H^1 semi-norm.

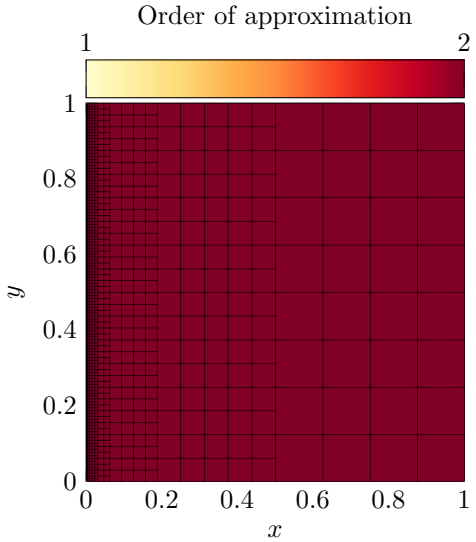
Figure A.16: One dimensional regular solution solved in a 2D mesh.



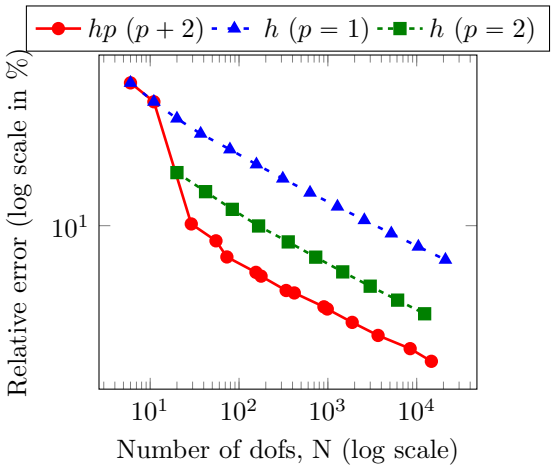
(a) Final adapted *hp*-mesh with orders in the x-direction.



(b) Final adapted *hp*-mesh with orders in the y-direction.

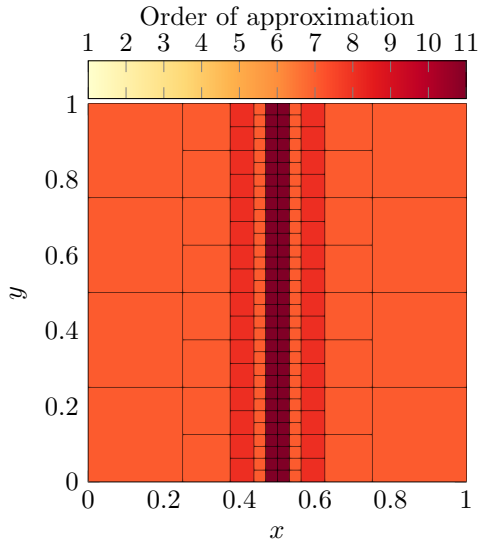


(c) Final adapted h-mesh, $p = 2$.

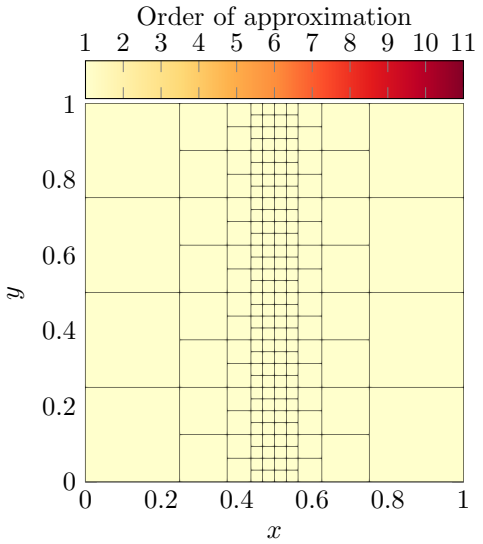


(d) Evolution of the error in H^1 semi-norm.

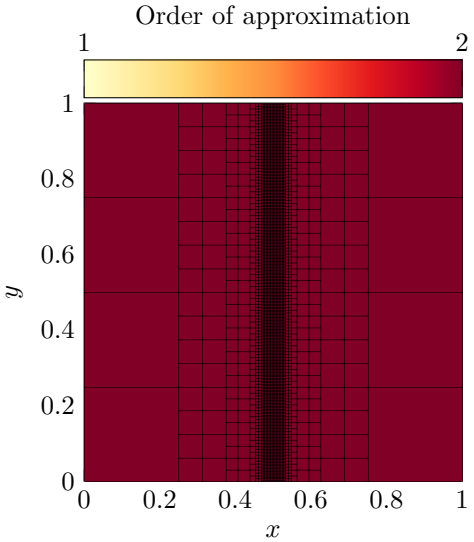
Figure A.17: One dimensional singular solution solved in a 2D mesh. The final adapted *hp*-mesh, Figures A.17a and A.17b, needs 14630 dofs to deliver an error of 4.546%; and the h-mesh, Figure A.17c, has an associated solution with error equal 5.993% for 12284 dofs.



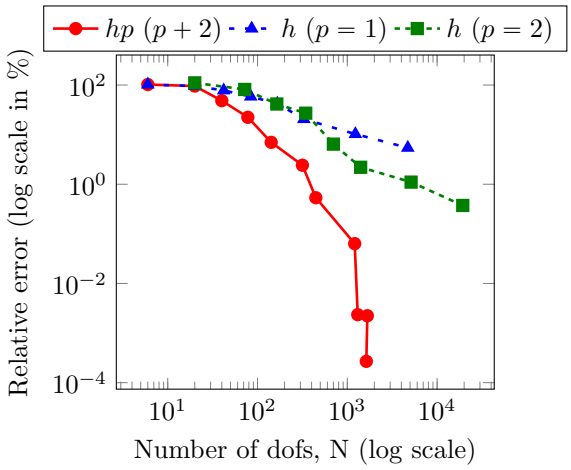
(a) Final adapted *hp*-mesh with orders in the x-direction.



(b) Final adapted *hp*-mesh with orders in the y-direction.



(c) Final adapted h-mesh, $p = 2$.



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

Figure A.18: One dimensional strong gradient solution solved in a 2D mesh. The final adapted *hp*-mesh, Figures A.18a and A.18b, needs 1628 dofs to deliver an error of $2.6 \cdot 10^{-4}$%; and the h-mesh, Figure A.18c, has an associated solution with error equal 0.374% for 19304 dofs.