POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Non-conforming mesh adaptivity for Hybrid High-Order methods

Laurea Magistrale in Mathematical Engineering - Ingegneria Matematica

Author: Alessandra Crippa

Advisor: Prof. Paola Antonietti

Co-advisor: Daniele Di Pietro

Academic year: 2021-2022

1. Introduction

The Hybrid High-Order (HHO) methods are discretization schemes for PDEs among the polytopal methods, whose key features are the arbitrary approximation orders, the hybrid nature (with unknowns both on cells and edges in 2D), and the properties related to polytopal methods in general, namely the freedom in discretizing the domain (see [3], [2] and [1]). They can indeed easily cope with elements with different numbers of edges, with hanging nodes, and with non-matching interfaces. As a direct consequence, it is possible to discretize the physical domain through non-conforming meshes, which is particularly relevant for physical phenomena occurring in small areas of the domain.

In this work, after introducing the HHO method, we will explain an automatic mesh adaptation procedure, based on a posteriori error estimates. Our ultimate goal will be to test it coupled with a local non-conforming refinement of the mesh, for which we will implement some methods within the HArD::Core library. The latter provides a suite of C++ tools to implement numerical schemes on general polytopal meshes and it lacks, up to now, methods to perform local non-conforming refinement of the mesh.

2. The HHO method

Let us see how HHO is built for the Poisson problem, that we recall in its weak formulation. Find $u \in H^1_0(\Omega)$ such that

$$a(u,v) = (f,v) \qquad \forall v \in H^1_0(\Omega). \tag{1}$$

where the bilinear form $a : H^1(\Omega) \times H^1(\Omega) \to \mathbb{R}$ is such that $a(u,v) \coloneqq (\boldsymbol{\nabla} u, \boldsymbol{\nabla} v)$.

The first ingredients to build the HHO method are *polynomial projectors*, in particular the L^2-orthogonal projector $\pi_X^{0,l} : L^1(X) \to \mathbb{P}^l(X)$ and the elliptic projector $\pi_X^{1,l} : W^{1,1}(X) \to \mathbb{P}^l(X)$. They will be necessary to inspire an appropriate way to re-write the left-hand side of problem 1. Indeed, by exploiting the integration by parts formula, namely,
given a function $v \in W^{1,1}(T)$ and a function $w \in C^\infty(\overline{T})$,

$$(\boldsymbol{\nabla} v, \boldsymbol{\nabla} w)_T = -(v, \Delta w)_T + \sum_{F \in \mathcal{F}_T} (v, \boldsymbol{\nabla} w \cdot \boldsymbol{n}_{TF})_F,$$

and by specializing it for a polynomial function $w \in \mathbb{P}^{k+1}(T)$, we discover a fundamental

relation between the projectors; namely, from $\pi_T^{0,k}$ e $\pi_F^{0,k}$, projectors of degree k it is possible to completely characterize the elliptic projector of higher degree k+1, $\pi_T^{1,k+1}$. The first idea of HHO methods arises here, namely to take as unknown some objects that can be interpreted as polynomial projectors: $((\pi_T^{0,k}v)_{T\in\mathcal{T}_h}, (\pi_F^{0,k}v)_{F\in\mathcal{F}_h})$.

2.1. Local contributions

Let us now therefore introduce the local HHO spaces, where we will look for a discrete solution.

$$\begin{aligned}\underline{U}_T^k \coloneqq \{\underline{v}_T = (v_T, (v_F)_{F\in\mathcal{F}_T}) \ : \\ v_T \in \mathbb{P}^k(T) \text{ and } v_F \in \mathbb{P}^k(F) \\ \forall F \in \mathcal{F}_T,\end{aligned} \tag{2}$$

Notice that the discrete space of the method is not included in the continuous one, $H_0^1(\Omega)$, differently from what happens in the standard finite elements method. This feature will be at the base for the more flexible properties of the mesh. (*verificare, è giusto?*)

Then we define the second main ingredient of the method: the local potential reconstruction operator, which is at the core of HHO methods. It is conceived such that it replicates the integration by parts formula that links the elliptic projector $\pi_T^{1,k+1}$ of degree $k+1$ to $\pi_T^{0,k}$ and $\pi_F^{0,k}$ projectors on the element and its faces, of degree k. As a result we can obtain an important relation:

$$\forall v \in W^{1,1}(T) \qquad \mathrm{p}_T^{k+1}\underline{I}_T^k v = \pi_T^{1,k+1} v, \tag{3}$$

where $\underline{I}_T^k$, the interpolator, is the operator that allows us to interpret the unknowns of HHO scheme as the L^2 projectors of a function v on the element and its faces.

$$\begin{aligned}\underline{I}_T^k : W^{1,1}(T) \to \underline{U}_T^k, \\ \underline{I}_T^k v \coloneqq (\pi_T^{0,k}v, (\pi_F^{0,k}v)_{F\in\mathcal{F}_T}).\end{aligned} \tag{4}$$

The result 3 will be fundamental for the HHO method to be consistent for polynomial exact solutions.

We can now arrange the final local problem, approximating the continuous bilinear form a on each element of the mesh T with the discrete bilinear form $\mathrm{a}_T : \underline{U}_T^k \times \underline{U}_T^k \to \mathbb{R}$ such that, for all $\underline{u}_T, \underline{v}_T \in \underline{U}_T^k$,

$$\begin{aligned}\mathrm{a}_T(\underline{u}_T, \underline{v}_T) \coloneqq (\boldsymbol{\nabla}\mathrm{p}_T^{k+1}\underline{u}_T, \boldsymbol{\nabla}\mathrm{p}_T^{k+1}\underline{v}_T)_T \\ +\mathrm{s}_T(\underline{u}_T, \underline{v}_T).\end{aligned} \tag{5}$$

The first term is responsible for consistency, indeed taking $\underline{u}_T$ as the interpolate of a polynomial function, exploiting therefore 3 and thanks to the definition of elliptic projection, we recover the continuous bilinear form a. The second is a stabilization term, other crucial ingredient for HHO methods. It is needed to recover the coercivity of the bilinear form, so that the method will be proved to be well-posed (thanks to Lax-Milgram theorem). Hence, s_T is assumed to satisfy some properties: it is symmetric and semidefinite positive, it makes the bilinear form coercive, and it satisfies polynomial consistency too.

2.2. Global discrete problem

Finally, the global discrete problem will be based on the following space of unknowns:

$$\begin{aligned}\underline{U}_h^k \coloneqq \Big\{\underline{v}_h = ((v_T)_{T\in\mathcal{T}_h}, (v_F)_{F\in\mathcal{F}_h}) \ : \\ v_T \in \mathbb{P}^k(T) \quad \forall T \in \mathcal{T}_h \text{ and} \\ v_F \in \mathbb{P}^k(F) \quad \forall F \in \mathcal{F}_h\Big\},\end{aligned} \tag{6}$$

which is a collection of polynomial functions on the cells and on the faces of the mesh. The global bilinear form will be given by $\mathrm{a}_h : \underline{U}_h^k \times \underline{U}_h^k \to \mathbb{R}$ and $\mathrm{s}_h : \underline{U}_h^k \times \underline{U}_h^k \to \mathbb{R}$ such that for all $\underline{u}_h, \underline{v}_h \in \underline{U}_h^k$,

$$\begin{aligned}\mathrm{a}_h(\underline{u}_h, \underline{v}_h) \coloneqq \sum_{T\in\mathcal{T}_h} \mathrm{a}_T(\underline{u}_T, \underline{v}_T), \\ \mathrm{s}_h(\underline{u}_h, \underline{v}_h) \coloneqq \sum_{T\in\mathcal{T}_h} \mathrm{s}_T(\underline{u}_T, \underline{v}_T).\end{aligned} \tag{7}$$

The global bilinear form enjoys the property of stability (hence coercivity), boundedness and consistency. It follows the well-posedness of the discrete Poisson problem

$$\mathrm{a}_h(\underline{u}_h, \underline{v}_h) = (f, v_h) \qquad \forall \underline{v}_h \in \underline{U}_{h,0}^k. \tag{8}$$

2.3. Convergence analysis

It is possible to prove some convergence estimates for the HHO methods, in two norms:

- the energy norm, induced by the discrete bilinear form a_h;

- the L2 norm.

Clearly we cannot expect to compute directly the error as the difference between the numerical solution $\underline{u}_T$ and the exact one, u, as they do not live in the same space (as we mentioned, $V_h \not\subset V$) so we must compare $\underline{u}_T$ to the interpolate of u, $\underline{I}_h^k u$ (or, as we see next, we can compare its component to the proper projection of u into polynomial spaces).
For sufficiently regular solutions and mesh sequences, it holds

$$\|\underline{u}_h - \underline{I}_h^k u\|_{\mathrm{a},h} \lesssim h^{r+1}|u|_{H^{r+2}(\mathcal{T}_h)}. \tag{9}$$

Further assuming elliptic regularity,

$$\|u_h - \pi_h^{0,k} u\|_{L^2} \lesssim \begin{cases} h^2\|f\|_{H^1(\mathcal{T}_h)} & \text{if } k = 0, \\ h^{r+2}|u|_{H^{r+2}(\mathcal{T}_h)} & \text{if } k \geq 1, \end{cases} \tag{10}$$

where u_h is the discontinuous polynomial function given by the components of $\underline{u}_h$ corresponding to the each element T. The last property is known as super convergence property of HHO methods.

2.4. Implementation

The implementation is carried out by fixing proper basis for the discrete spaces and deriving accordingly a linear algebraic system - which accounts also for the boundary conditions - equivalent to the discretized weak formulation. A crucial technique to be highlighted, used to assemble the final linear system, is the so-called *static condensation*. It allows to account only for face unknowns, making the HHO scheme a *skeletal* method, reducing therefore the size of the linear system to be solved and the computational effort.

3. A posterior Error Estimator

Since the aim of this work is to focus on adaptivity of the mesh, we briefly introduce some a posteriori error estimators, whose goal will be to drive the mesh refinements. Notice however that this aspect will be carried out in our upcoming work, while for now, in this thesis, we will exploit only the *true* error to drive the mesh refinement, considering a benchmark case with a known solution. The mesh adaptivity procedure gives some crucial advantages, such as the significant enhancement of the performance of problems with singular solutions, allowing to fully exploit the high-order of approximation of HHO schemes. Indeed, for smooth exact solutions, increasing the polynomial degree yields a corresponding increase in the convergence rate, as we saw in the previous section. However, if the exact solution does not satisfy the requested regularity assumptions, the order of convergence is limited by the poor regularity of the solution. In order to restore optimal order of convergence local mesh adaptation can help, typically using local a posteriori error estimators to mark the elements with the largest error, so that they will be refined.

The a posteriori error estimator on which we will focus is the so-called residual based. It can be proven that the discretization error - computed as the exact solution $u$minus the polynomial reconstruction of the numerical solution$\underline{u}_h$ - is bounded from above by a scalar quantity ε, the so-called *estimator*.

$$\|\boldsymbol{\nabla}_h(\mathrm{p}_h^{k+1}\underline{u}_h - u)\| \leq \varepsilon.$$

ε can be moreover proven to be computable only through the discrete solution and the problem data, hence the upper bound is said to be *reliable*. Moreover, since there are no undetermined constants in the upper bound, it is said to be *guaranteed* and *fully computable*. It is also important for an estimator to be locally and globally efficient - namely bounded from above by the discretisation error - to be sure that the estimators localise the error correctly and do not overstimate it. Indeed it can be proven for our residual based error estimator the following:

$$\varepsilon \lesssim \left(\|\boldsymbol{\nabla}_h(\mathrm{p}_h^{k+1}\underline{u}_h - u)\| + |\underline{u}_h|_{\mathrm{s},h}\right).$$

This implies that the a posteriori estimate is eligible to drive local mesh refinement.

The typical adaptive procedure is characterized by the iteration of the following four steps: solve for the numerical scheme on the current mesh, compute the error estimator, mark certain elements with precise values of the estimators, refine them to get the next mesh (see Algorithm 1).

Algorithm 1 Automatic mesh adaptation

Set tol > 0 and $N_{\max}$
Generate an initial coarse mesh $\mathcal{T}_h^{(0)}$, set $n \leftarrow 0$, and let $\mathcal{T}_h^{(n)} \leftarrow \mathcal{T}_h^{(0)}$
repeat
  Solve the HHO problem (8) on $\mathcal{T}_h^{(n)}$
  for $T \in \mathcal{T}_h^{(n)}$ **do**
    Compute and store the local estimator ε_T
  end for
  for $T \in \mathcal{T}_h^{(n)}$ **do**
    if T is among the 5% elements with the largest local estimator **then**
      mark the element T
    end if
  end for
  Set $n \leftarrow n+1$ and generate a novel mesh $\mathcal{T}_h^{(n)}$ by by refining the marked elements
until $\varepsilon <$ tol **or** $n > N_{\max}$

The adaptation procedure has been tested by [3] on the Fichera corner problem, whose solution is known to have a singularity in the origin. The authors showed that (see Fig. 1) uniformly refined meshes fail in achieve the optimal order of convergence (because of the poor regularity of the solution), however, exploiting instead the adaptation procedure (with a *conforming* refinement of the mesh), it is possible to recover the optimal convergence rates of $N_{\text{dof}}^{\frac{(k+1)}{d}}$ and $N_{\text{dof}}^{\frac{(k+2)}{d}}$ in both the energy-norm and L2-norm. This confirms that the above mentioned algorithm is able to restore optimal orders of convergence.

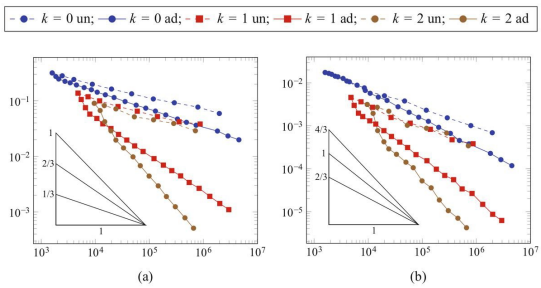

Figure 1: Error vs. N_{dof} for the test case of Fichera problem. "un"= uniformly refined meshes, "ad"= adaptively refined meshes, from [3].

4. Non-conforming mesh adaptivity

The last section of this work includes our original contribution, namely some methods to implement a local non-conforming adaptivity of the 2D mesh. The starting point it is the HArD::Core library, implemented by D. Di Pietro and J. Droniou, which provides a suite of C++ tools to implement numerical schemes on general polytopal meshes, whose unknowns are polynomials in the cells, both on the edges and on the faces (hybrid methods). The library was mainly designed for the Hybrid High-Order methods: several HHO methods are already implemented in this framework, however it provides tools which are useful for a range of numerical methods.

Within the library, the main class designed to handle the mesh is the `Mesh` class, which contains a vector of `Cell`, a vector of `Edge` and a vector of `Vertex` objects, based on classes that suitably represent all the entities of the mesh. The mesh class provides also methods to extract single elements, such as particular cells, edges, or vertices given their given. For instance, the function `Mesh::get_cells()` returns a vector with all the cells of the mesh, ordered according to their global indeces, while the function `Mesh::cell(size_t i)` returns the cell with global index `i`.

The goal of this work is to develop a new class which extends the classical `Mesh`, called `DynamicMesh`, able to handle the refinement (and coarsen) operations, and at the same time which mantains the same interface as `Mesh`. As a result the two classes can be used interchangeably and all the already existing schemes for solving partial differential equations, which use the `Mesh` class and its methods, will not need to be changed. Hence, the aim of the our `DynamicMesh` class will be to add some new methods to refine the mesh, and in the meantime to overload the already existing getter functions `get_cells()`, `cell(size_t i)`, etc. Clearly they will have to iterate over all the mesh entities properly, going through every element and every sub-element coming from the refinement operations. We will work only with triangular meshes, splitting at each refinement

the cell in 4 smaller trianglular cells joining the midpoints, as displayed in fig. 2.

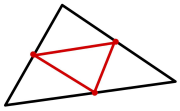

Figure 2: 2D triangular refined cell.

The mechanism of our refinement method is the following: when a cell (edge) is refined, it becomes *father* of four (two) new *children* cells (edges, respectively). In the `DynamicMesh` are contained two new vectors of objects based on the new classes `CellInfo` and `EdgeInfo`, whose goal is to store the *information* of the refinement. Each `CellInfo` represents a *virtual* cell of the refined mesh and each `EdgeInfo` a *virtual* edge and they basically contain pointers to their children and to their father in order to keep track of all the refined elements. However, notice that the vectors `cells`, `_edges` and `_vertices` will actually not change.

When the `refinement(CellInfo* c)` function is called, it creates properly the 4 new cells from `c`, add them to the children attribute of `c`, and adds `c` to their father attribute. In this way we create a tree-like structure of the mesh entities, and we will be able to iterate correctly through every element follow the branches of the tree, by mean of recursive functions.

The coarsen method inside `DynamicMehs` class, works exactly in the opposite direction. Its aim is to merge the `CellInfo` taken as argument to its brother, by eliminating all his father's children and their subchildren.

5. Numerical tests

Finally we tested the adaptation algorithm 1 on our new non conforming refinement methods, still considering the Fichera corner problem, which has a singularity in the origin.
We showed how the optimal convergence orders are recovered (see Fig. 5 with polynomial degree $k = 3$ as example).

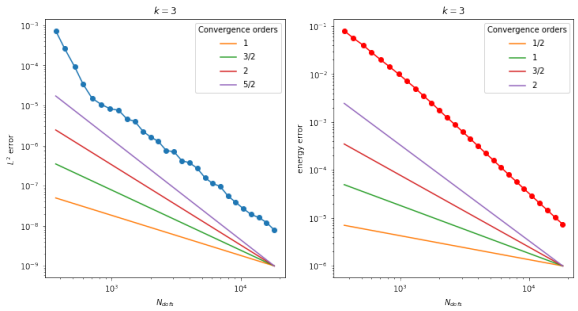

Figure 3: Error vs. N_{dof} for the Fichera solution with polynomial degree 3. Expected convergence rate: $N_{\text{dof}}^{-5/2}$ for the L^2 error, N_{dof}^{-2} energy error.

To conclude we display an example of refined meshes at different iterations of the mesh adaptation algorithm (see fig. 5).

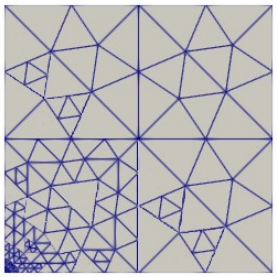

Figure 4: 10 iterations.

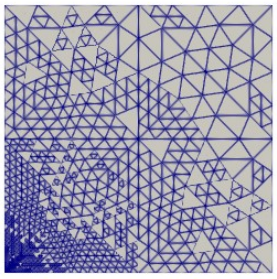

Figure 5: 25 iterations.

6. Conclusions

As we showed in the last section, a non-conforming adaptivity of the mesh is able to recover the optimal convergence rates also for poorly regular solutions. The novelty

introduced in our work, namely the non conformity of the refinement, can significantly reduce the computational cost required by an adaptation procedure, since it needs to re-build the mesh just locally at every refinement operation. It suits perfectly problems where physical phenomena occur in small areas of the domain, or problems which result in poorly regular solutions, like interface problems.

In upcoming work it is possible to incorporate a posteriori estimators into the mesh adaptation procedure, as well as the possibility to coarsen the mesh where the errors are low.

References

[1] Daniele A Di Pietro and Alexandre Ern. A hybrid high-order locking-free method for linear elasticity on general meshes. *Computer Methods in Applied Mechanics and Engineering*, 283:1–21, 2015.

[2] Daniele A Di Pietro and Ruben Specogna. An a posteriori-driven adaptive mixed high-order method with application to electrostatics. *Journal of Computational Physics*, 326:35–55, 2016.

[3] Daniele Antonio Di Pietro and Jérôme Droniou. The hybrid high-order method for polytopal meshes. *Design, analysis, and applications*, 19, 2019.