POLITECNICO
MILANO 1863

A Hybrid High Order method for magnetostatics

M.Sc. Thesis of
Aurelio Edoardo Spadotto

Supervisor:
Prof. Luca Formaggia

Co-supervisor:
Prof. Daniele Antonio Di Pietro

Programme:
Mathematical Engineering

Department of Mathematics
Politecnico di Milano

Contents

Abstract

The Hybrid High Order method (HHO) is a discretization method for PDEs designed to support general polyhedral meshes and provide arbitrary approximation order. The method's strength is the great freedom it gives in choosing a discretization of the physical domain. No shape of the elements is imposed, and elements with a different number of faces can coexist in the same mesh. Yet, a stability and convergence analysis can be provided *a priori*. Moreover, despite an increased complexity, the method shows itself prone to several techniques alleviating the computational burden.

The method's unknowns are conceived as collections of polynomials attached to both cells and faces of a polyhedral mesh, whence the term *hybrid* in the name. The first cornerstone of the method is the concept of *local reconstruction*. It is a technique to design local, easily invertible and embarassingly parallel problems that allow to obtain higher degree cell-wise polynomial projections of functions starting from lower degree polynomial projections on both cells and faces. The second main ingredient of the method are suitable built-in *stabilisation terms*, added to guarantee stability while preserving convergence properties.

After several proofs of performance in the domain of computational mechanics, the latest boundary in HHO development is the numerical analysis of electromagnetism models. Electromagnetic phenomena are fully described by the celebrated Maxwell equations. In stationary conditions the magnetic component is independent, and must respect a linear second order **curl curl** problem. Starting from an original implementation of the HHO method for this problem, several optimization techniques are designed together with an extension to more general boundary conditions. The method performs stable and convergent on a heterogeneous variety of polyhedral meshes.

Acknowledgments

This work has been realized during an internship at the *Institut Montepelliérain Alexander Grothendieck* (IMAG), under the supervision of prof. Daniele Di Pietro of the University of Montpellier and prof. Simon Lemaire, from INRIA Lille. Implementation is almost entirely based on the library HArD::Core3D (`https://github.com/jdroniou/HArDCore3D-release.git`), by prof. Jérôme Droniou from Monash University of Melbourne and by prof. Di Pietro. Numerical tests have been performed with the support of MESO@LR-Platform at the University of Montpellier. Many thanks to anybody involved during the development of this project and to prof. Luca Formaggia for the final revision of this report.

Milano, 07 Oct 2021 A. S.

1 Introduction

Hybrid High Order methods belong to the family of numerical schemes for computational mechanics whose most established member is the Finite Element Method. The design and analysis of these methods is based on a variational formulation of differential problems in a Hilbert setting, which is the source for a finite dimensional recasting, leading eventually to an algebraic problem faced numerically. The discretization step always relies on a suitable decomposition of the domain into *elements*. The necessity to apply Finite Element schemes in the most general scenarios, with irregular domains, fractures and multiple scale geometries has driven the work in computational mechanics towards new versions of the method relaxing the constraints on the mesh features. In this sense, the Discontinuous Galerkin Method is one of the most known schemes. With DG, vertices of neighbouring triangular elements are not required to match. Polytopal methods push this trend beyond, allowing the support of elements with an arbitrary number of flat faces. The most promising aspect of polyhedral schemes such as the HHO method is that they can help working around typical drawbacks of classical unstructured meshes of FEM. Geometric microstructures in a domain can be caught by large agglomerated polyhedral elements instead of a greater number of small unstructured elements, without jeopardizing the accuracy. Moreover, treatment of interfaces and mesh refinement suffers fewer constraints and limitations than the case of conforming Finite Element Method.

Another distinguishing feature of the HHO method is being hybrid. The so-called *degrees of freedom* are attached to both cells and faces of the mesh. Instead of considering broken cell-wise polynomials to discretize functions as it is the case with continuous and discontinuous Finite Element methods, with HHO collections of cell-wise and face-wise polynomials are stored as unknowns. The cornerstone of the method is the idea of *reconstructing* higher-degree broken polynomials starting from these collections of heterogeneous objects. It is important to remark that such recostruction is the sum of independent cell-wise reconstruction operations. From the implementation point of view, this translates into the possibility to introduce a parallelization of the reconstruction step. The latter is one of the strategies that can be put in action to prevent the scheme from becoming computationally unbearable. Indeed, if we compare HHO method with the linear Finite Element method having a DOF per vertex, we realize that the total dimension of the overall system tends to be huge, revealing the main drawback of non-conforming schemes. Actually, despite being an hybrid method, at the implementation level HHO can be considered a *skeletal* scheme, as it is possible to assemble a linear system accounting only for face unknowns. That is true thanks to a local static condensation and computation of a local Schur complement. Again, the

local nature of this tecnhique allows its application in parallel at the element level.

The first apperance of the HHO method dates back to 2014 [DS14], when it was first applied to linear elasticity problems. Since then, HHO approach has been proposed and validated for several linear and nonlinear problems, both in solid and fluid mechanics. The results of the present work refer to a line of research with the aim of expanding the analysis to computational electromagnetism. In particular the problem of magnetostatics has been considered. The problem is linear and second order, and it is characterized by the presence of the **curl curl** operator. In [FS20] an effort is shown to perform a reconstruction of the **curl** operator by reducing as much as possible the number of degrees of freedom required without losing accuracy with respect to a more straight-forward but heavier reconstruction. This work describes how an implementation of the optimized method has been performed and offers a comparison between the results with a pre-optimized scheme and those obtained by following the scheme architecture shown in [FS20]. Moreover, the case of general boundary conditions is explored. The original HHO scheme for the magnetostatics problem had been designed and analised for the case of fully Dirichlet boundary conditions. Here, a similar scheme is proposed to deal with the case of Neumann boundary conditions, with a proposal to discretize the border contribution of the right hand side and an alternative stabilisation scheme. A stability and convergence analysis is brought on following the steps of the Dirichlet case. The choice of the stabilisation is crucial to define a seminorm which must turn out to be a norm on a subspace to which we restrict the search of the solution. The most important theoretical results supporting the analysis are Weber inequalities, which play a similar role to Poincaré inequalities when curl is considered in place of gradient. An HHO scheme for Mixed boundary conditions is showed and tested too.

Local reconstructions are at the core of HHO schemes, but the way it can be conceived is not unique. The present exposition is mostly related to discrete reconstructions of differential operators, but an alternative approach consists in the opposite direction, a potential reconstruction. Even though it was not implemented numerically, some space is devoted to expose the problem of reconstructing a potential. The task is quite straightforward in case of a scalar potential, but some care must be taken when reconstructing a vector potential. This difference is the discrete counterpart of the fact that a scalar potential is fixed up to a constant, whereas a vector potential is fixed up to a gradient. A possible closed reconstruction of a vector potential is proposed that may be the source for future developments of the scheme. Another difficulty emerging when dealing with potential functions is that their construction requires special care with non trivial topologies, which can be instead very interesting in real-world applications. The validity of the results showed assumes a connected domain with connected boundary. In future generalized schemes may make use of topology detecting algorithms as proposed in [P D13]. These should be coupled with topology-sensitive reconstruction of potentials.

From the implementation point of view, most of the work has been devoted to define suitable classes to represent the articulated classes of polynomial families that are used

in the optimized reconstruction. The library provided already a good number of ways to represent polynomial spaces as objects, but still a way to represent composed basis and ways to extract bases out of the image of differential operators like the **curl** was missing. This made up the contribution touching the core of the project. After these changes, the implementation of the scheme could go on by relying on a general routine, adding a solver method specific for the magnetostatics problem. Most of the steps of the solver are standard and share the same philosophy as Finite Element Methods. The peculiar step involves the building of the cell-wise reconstruction matrices, the computation of the discrete local matrices and the assembly of the statically condensed local contribution for each cell. Once the schemes have been implemented, they have been tested on different families of meshes to check the independence of the scheme from the shape of the polyhedra.

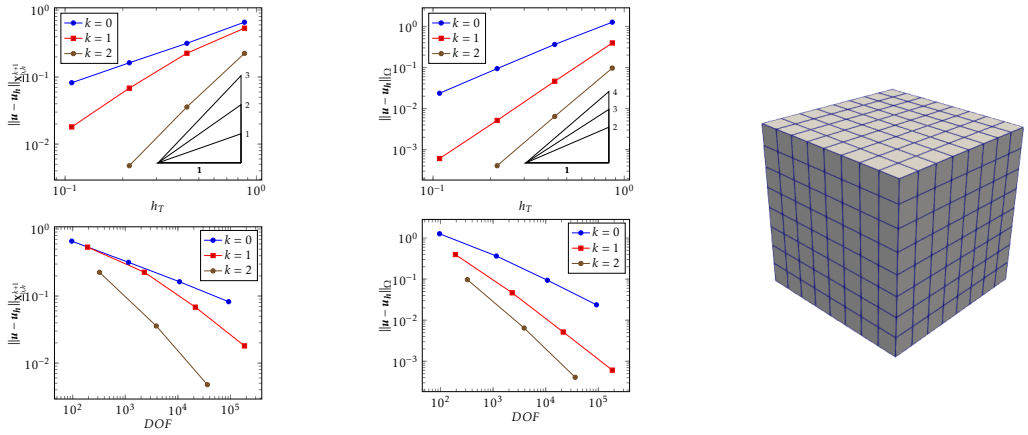


Figure 1.1: HHO performance on a family of hexahedral meshes

This exposition is organized as follows: after a framing of the physical model leading to the equations of magnetostatics the HHO method is presented for general boundary conditions. A stability and convergence analysis is presented, for the particular case of Neumann conditions. After the theoretical discussion, the most relevating implementation features are shown, and an overview of the software is given, together with a focus on the original contribution added to cope with new boundary conditions. Eventually, a gallery of results is exposed, with an emphasis on independence of the convergence trend from the chosen polytopal geometry of the mesh.

2 Model Framing

Electromagnetic interaction is one of the four fundamental ways matter exchanges forces, together with gravitation and the two kinds of nuclear interaction. Starting from the description of forces acting on objects that acquired a property called *electric charge* when rubbed, a long chain of apparently independent phenomena have been found to be linked by a common physical entity, the electromagnetic field. By the end of the XIX century the most relevant physical models for the microscopic nature of matter and light were based on electromagnetism. From the point of view of applications, exploitation of electricity has driven the most revolutionary technological outbreak after the advent of steam machines. Nevertheless, the entire theory of electromagnetism is completely described by a set of four principles, that mathematically translate as four linear partial differential equations. Informally they read:

$$\begin{cases} \nabla \cdot E & = \rho \\ \nabla \times E & = -\frac{\partial B}{\partial t} \\ \nabla \cdot B & = 0 \\ \nabla \times B & = J + \frac{\partial E}{\partial t}, \end{cases}$$

where E and B indicate respectively the electric and magnetic fields, ρ is the charge density, and J the current density. This is known as Maxwell system. It is such a solid set of physical laws that the fundamental concepts of space and time have been reformulated to find consistence with one of its consequences: light travels in vacuum at a constant speed for any observer. Einstein's assumption of universal validity of these equations lead to his groundbreaking work on relativity.
Set aside the theoretical importance of Maxwell equations, this work is mainly concerned with the task of finding solutions in specific configurations. Computer-assisted calculations based on approximations of the continuous laws by large systems of easy to program discrete equations have been one of the very first tasks electronic calculators were tested with. Since the half of the XX century, computational physics has produced a great variety of different methods to discretize equations which helped and deal with complex configurations that are beyond the scope of the analytical approach. Polyhedral methods such as the Virtual Element Method or HHO Method are part of the effort to get simulations that are reliable and sufficiently accurate trying to minimize the computational burden in terms of time and computer memory.
In order to apply a method that was originally introduced for mechanical problems to the domain of electromagnetism, a step-by-step approach is in progress, starting from simple hypotheses. First of all, we focus on the stationary case, where all time derivatives vanish. In static conditions, the electric phenomena are independent from

the magnetic ones. The electric equations, through the introduction of an electric potential field, reduce to the well-known Laplace equation, which has been extensively treated both analitically and numerically in the context of the harmonic theory. On the other hand, the magnetic side of the decoupled Maxwell system reads:

$$\begin{cases} \mathbf{curl}\, B & = J \\ \mathbf{div}\, B & = 0. \end{cases}$$

From the mathematical point of view, sets of equations valid everywhere in a given domain Ω are coupled with boundary conditions, required to be valid on the boundary of Ω. This is often the way to obtain a closed well-posed problem, which can be proved to have one and only one solution. In this context, the first condition one could add is a *Dirichlet condition*, namely:

$$\boldsymbol{n} \cdot B = \boldsymbol{n} \cdot \Psi \quad \text{on } \partial\Omega,$$

where $\boldsymbol{n}$ is the outgoing vector normal to the boundary, and Ψ is a given vector-valued field on $\partial\Omega$. This condition, other than providing a mathematical closure to the system, models the physical condition of a perfect conductor enclosing the domain. Having a perfect conductor imposes that no impinging plane waves can cross the boundary, and the energy of the system is not spread outside Ω. It is the electromagnetic equivalent of the reflection condition for a vibrating rope with a fixed end. It is often the case that an alternative formulation for this problem is considered, emerging from the fact that $B \in \boldsymbol{H}(\mathbf{div}^0;\Omega)$, that is B is a solenoidal field. Indeed, under the hypothesis of trivial topology for Ω, it is true that:

$$\boldsymbol{H}(\mathbf{div}^0;\Omega) = \mathbf{curl}(\boldsymbol{H}(\mathbf{curl};\Omega)).$$

In other words it is possible to find a field $\boldsymbol{u}$ such that $\mathbf{curl}\, \boldsymbol{u} = B$. $\boldsymbol{u}$ is called *vector potential* for the magnetic field. The vector potential is not uniquely defined, unless the value of its divergence and normal component at the boundary is prescribed. Assigning these two values constitues a choice for a *gauge*. The most simple is the Coulomb gauge, assuming divergence free potential with vanishing normal component at boundary. When $\Psi = 0$, translating the null divergence condition as the existence of a potential and fixing the gauge leads to the problem in potential formulation:

$$\begin{cases} \mathbf{curl}(\mathbf{curl}\, \boldsymbol{u}) = \boldsymbol{J} & \text{in}\, \Omega \\ \mathbf{div}(\boldsymbol{u}) = 0 & \text{in}\, \Omega \\ \boldsymbol{n} \times (\boldsymbol{u} \times \boldsymbol{n}) = \boldsymbol{0} & \text{on}\, \partial\Omega. \end{cases}$$

This is the problem our numeric analysis starts from. A further manipulation leads to a weak form, suitable for discretization. That is where many discretization schemes are introduced, HHO not being an exception.

Of course, Dirichlet condition is only one of the possibilities to obtain a well-posed problem with meaningful physical interpretation. An alternative is a *Neumann condition*, namely:

$$\boldsymbol{n} \times (B \times \boldsymbol{n}) = \boldsymbol{n} \times (\Phi \times \boldsymbol{n}),$$

where Φ is a vector-valued field on $\partial\Omega$. This kind of condition is suitable to take into account the presence of dissipative walls that allow energy transmission. There are eventually configurations in which some walls are of type Dirichlet, and others of type Neumann. Such mixed scenarios are to be considered too. In the end, after a suitable analysis, an HHO solver is available to provide numerical simulation of the magnetic field in closed domains, with sufficiently general conditions at the boundary.

3 The HHO Method

3.1 Discrete setting

Before introducing the method, it is useful to expose a list of definitions and conventions that will be used through the rest of the presentation, starting from the notion of polytopal mesh.

Definition 1. *Let an integer $d > 2$ be fixed. Given a set of vertices $\mathcal{P} := \{\boldsymbol{P}_0, ..., \boldsymbol{P}_d\} \subset \mathbb{R}^d$ such that the family of vectors $\{\boldsymbol{P}_1 - \boldsymbol{P}_0, ..., \boldsymbol{P}\_\boldsymbol{P}_0\}$ is linearly independent, the interior of the convez hull of $\mathcal{P}$ is a **simplex** of $\mathbb{R}^d$. For each integer $i \in 0, ..., d$, the convex hull of $\mathcal{P} \setminus \boldsymbol{P}_i$ is a **simplicial face**. A **polytope** is a connected set that is the interior of a finite union of closures of simplices.*

Definition 2. *A **polytopal mesh** of Ω is a couple $\mathcal{M}_h = (\mathcal{T}_h, \mathcal{F}_h)$ where:*

- *The set of* mesh elements *(or mesh cells) $\mathcal{T}_h$ is a finite collection of nonempty disjoint polytopes T with boundary ∂T and diameter h_T such that the meshsize h satisfies*

$$h = \max_{T \in \mathcal{T}_h} h_T$$

  and it holds

$$\overline{\Omega} = \bigcup_{T \in \mathcal{T}_h} \overline{T}$$

- *The set of mesh faces $\mathcal{F}_h$ is a finite collection of disjoint subsets of $\overline{\Omega}$ such that, for any $F \in \mathcal{F}_h$, F is a non-empty open connected subset of a hyperplane of $\mathbb{R}^d$ and the $(d-1)$-dimensional Hausdorff measure of its relative boundary $\overline{F} \setminus F$ is zero. We denote by h_F the diameter of F. Further assume that:*
  - *For each $F \in \mathcal{F}_h$, either there exist distinct mesh elements $T_1, T_2 \in \mathcal{T}_h$ such that $F \subset \partial T_1 \cap \partial T_2$ and F is called an interface, or there exists one mesh element $T \in \mathcal{T}_h$ such that $F \subset \partial T \cap \partial\Omega$ and F is called a boundary face;*
  - *The set of mesh faces is a partition of the mesh skeleton, i.e.,*

$$\bigcup_{T \in \mathcal{T}_h} \partial T = \bigcup_{F \in \mathcal{F}_h} \overline{F}.$$

Interfaces are collected in the set $\mathcal{F}_h^i$ and boundary faces in $\mathcal{F}_h^b$, so that

$$\mathcal{F}_h = \mathcal{F}_h^i \cup \mathcal{F}_h^b.$$

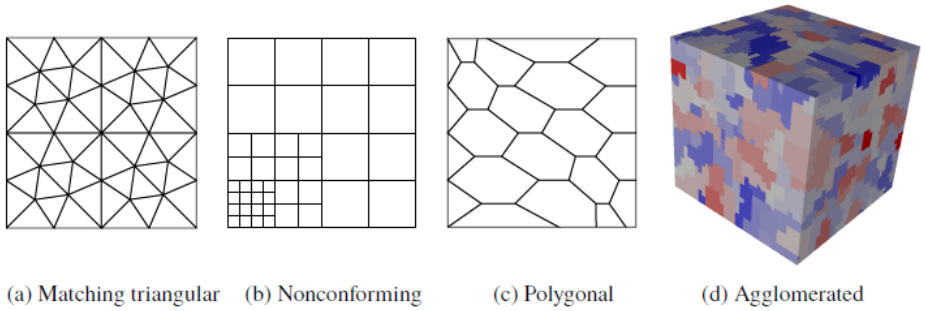
For any mesh element $T \in \mathcal{T}_h$,

$$\mathcal{F}_T := \{F \in \mathcal{F}_h : F \subset \partial T\}$$

denotes the set of faces contained in ∂T. *Symmetrically, for any mesh face* $F \in \mathcal{F}_h$,

$$\mathcal{T}_F := \{T \in \mathcal{T}_h : F \subset \partial T\}$$

is the set containing the one or two mesh elements sharing F. Finally, for all $T \in \mathcal{T}_h$ *and all* $F \in \mathcal{F}_T$, $\boldsymbol{n}_{TF}$ *denotes the unit normal vector to F pointing out of T.*



(a) Matching triangular (b) Nonconforming (c) Polygonal (d) Agglomerated

Figure 3.1: Examples of polytopal meshes supported by HHO in 2D and 3D

The method for a tridimensional problem, so from this point on $d = 3$ will be considered. We'll need to consider several local polynomial spaces attached to the faces and cells of a polytopal mesh:

- $\mathcal{P}^k(T)$: scalar polynomials of degree up to k on the cell T,
- $\boldsymbol{\mathcal{P}}^k(T)$: 3-valued polynomials up to degree k on the cell T,
- $\mathcal{P}^k(F)$: scalar polynomials of degree up to k on the face F,
- $\boldsymbol{\mathcal{P}}^k(F)$: 2-valued polynomials up to degree k on the cell T,
- $\boldsymbol{\mathcal{R}}^k(T) := \mathbf{curl}(\boldsymbol{\mathcal{P}}^{k+1}(T))$
- $\boldsymbol{\mathcal{G}}^k(T) := \mathbf{grad}(\mathcal{P}^{k+1}(T))$
- $\boldsymbol{\mathcal{G}}^k(F) := \mathbf{grad}_\tau(\mathcal{P}^{k+1}(F)) \subseteq \boldsymbol{\mathcal{P}}^k(T)$
- $\widetilde{\mathcal{P}}^{k+2}(F)$: homogeneous polynomials of degree up to $k+2$ on face F
- $\mathcal{P}_{\mathfrak{b}}^{k+1}(F) := \mathcal{P}^k(F) \oplus \mathbf{grad}_\tau(\widetilde{\mathcal{P}}^{k+2}(F))$

Let X be a cell or a face of a polytopal mesh. We use the notation $(f,g)_X$ to express the L_2 product $\int_X fg dX$ and $\|f\|_X$ to indicate the L_2 norm on X. In the design and analysis of the HHO method L_2 projectors on polynomial spaces play a fundamental role.

Definition 3. *Let $d = l = 2, 3$ and $X \subseteq \mathbb{R}^d$. Let U be a subspace of $L^2(X)$. The operator $\pi_U : L^2(X) \to U$ such that:*

$$(\pi_U g, u)_U = (g, u)_U \qquad \forall\, u \in U$$

*is called L_2 **orthogonal projector** on U.*

Let $d = l = 2, 3$ and $X \subseteq \mathbb{R}^d$. Let U be a subspace of $L^2(X, \mathbb{R}^d)$. The operator $\boldsymbol{\pi}_U : L^2(X, \mathbb{R}^d) \to U$ such that:

$$(\boldsymbol{\pi}_U g, u)_U = (g, u)_U \qquad \forall\, v \in U$$

*is called (tridimensional) L_2 **orthogonal projector** on U.*

When $U = \mathcal{P}^k(T)$ the synthetic notation π_T^k will be used, and similarly we will use: $\pi_F^k, \boldsymbol{\pi}_T^k$ and $\boldsymbol{\pi}_F^k$.

To introduce the concept of local reconstruction we need to consider collection of polynomes on a cell and on all the neighbouring faces.

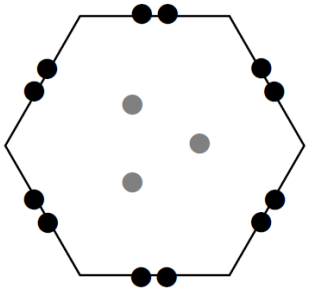


Figure 3.2: Degrees of freedom collocation on a 2D hexagonal cell. Distinct colors are used for cell and face unknowns

Definition 4.

$$\underline{\mathrm{Y}}_T^{k+1} := \{\underline{q}_T = (q_T, \{q_F\}_{F \in \mathcal{F}_T}) \qquad : \begin{cases} q_T \in \mathcal{P}^k(T) \\ q_F \in \mathcal{P}^{k+1}(F) \end{cases}\}$$

*is the **local HHO space** for scalar variables.*
*An **interpolator** is an operator $\underline{\mathrm{I}}_T^{k+1} : L_2(T) \to \underline{\mathrm{Y}}_T^{k+1}$ defined such that:*

$$\underline{\mathrm{I}}_T^{k+1} q = \underline{q}_T = (\pi_T^k g, \{\pi_F^{k+1} g|_F\}_{F \in \mathcal{F}_T}).$$

*The local **gradient reconstruction operator** $\boldsymbol{G}_T^{k+1} : \underline{\mathrm{Y}}_T^{k+1} \to \mathcal{P}^{k+1}(T)$ associates to $\underline{q}_T \in \underline{\mathrm{Y}}_T^{k+1}$ a reconstruction $\boldsymbol{G}_T^{k+1} \underline{q}_T$ such that:*

$$\left(\boldsymbol{G}_T^{k+1} \mathrm{q}_{T,F}, \boldsymbol{w}\right)_T = -(\mathrm{q}_T, \mathbf{div}\, \boldsymbol{w})_T + \sum_{F \in \mathcal{F}_T} (\mathrm{q}_F, \boldsymbol{w}_{|F} \cdot \boldsymbol{n}_{TF})_F \quad \forall \boldsymbol{w} \in \boldsymbol{\mathcal{P}}^{k+1}(T). \tag{3.1}$$

Definition 5.

$$\underline{\mathbf{X}}_T^{k+1} := \{\underline{\boldsymbol{u}}_T = (\boldsymbol{u}_T, \{\boldsymbol{u}_F\}_{F\in\mathcal{F}_T}) \qquad : \begin{cases} \boldsymbol{u}_T \in \boldsymbol{\mathcal{P}}^k(T) \\ \boldsymbol{u}_F \in \mathcal{P}_{\mathfrak{b}}^{k+1}(F) \end{cases}\}$$

*is the **local HHO space** for vector variables.*
*An **interpolator** is an operator $\underline{\mathbf{I}}_T^{k+1} : L_2(T,\mathbb{R}^3) \to \underline{\mathbf{X}}_T^{k+1}$ defined such that:*

$$\underline{\mathbf{I}}_T^{k+1}\boldsymbol{u} = \underline{\boldsymbol{u}}_T = (\pi_T^k\boldsymbol{u}, \{\pi_{\mathcal{P}_{\mathfrak{b}}^{k+1}(F)}\boldsymbol{u}|_F\}_{F\in\mathcal{F}_T}).$$

*The local **gradient reconstruction operator** $\boldsymbol{C}_T^k : \underline{\mathbf{Y}}_T^{k+1} \to \boldsymbol{\mathcal{P}}^{k+1}(T)$ associates to $\underline{\boldsymbol{u}}_T \in \underline{\mathbf{X}}_T^{k+1}$ a reconstruction $\boldsymbol{C}_T^k\underline{\boldsymbol{u}}_T$ such that:*

$$\left(\boldsymbol{C}_T^k\underline{\mathbf{u}}_T, \mathbf{w}\right)_T = (\mathbf{u}_T, \mathbf{curl}\,\mathbf{w})_T + \sum_{F\in\mathcal{F}_T}(\mathbf{u}_F, \gamma_{\tau,F}(\mathbf{w}\times\boldsymbol{n}_{TF}))_F \quad \forall\mathbf{w}\in\mathcal{R}^k(T). \tag{3.2}$$

A special role is played by the following commutation properties, valid for any T:

$$\boldsymbol{G}_T^{k+1}\underline{\mathbf{I}}_T^{k+1}q = \pi_T^{k+1}(\mathbf{grad}\,q|_T), \tag{3.3a}$$
$$\boldsymbol{C}_T^k\underline{\mathbf{I}}_T^{k+1}\boldsymbol{v} = \pi_{\boldsymbol{C}_T}^{k+1}(\mathbf{curl}\,\boldsymbol{v}|_T). \tag{3.3b}$$

We can then define the global hybrid spaces that appear in the HHO method:

$$\underline{\mathbf{X}}_{\mathfrak{b},h}^{k+1} := \left\{\underline{\mathbf{v}}_h = \left((\mathbf{v}_T)_{T\in\mathcal{T}_h}, (\mathbf{v}_F)_{F\in\mathcal{F}_h}\right) : \begin{array}{ll} \mathbf{v}_T \in \boldsymbol{\mathcal{P}}^{k+1}(T) & \forall T\in\mathcal{T}_h, \\ \mathbf{v}_F \in \mathcal{P}_{\mathfrak{b}}^{k+1}(F) & \forall F\in\mathcal{F}_h \end{array}\right\}, \tag{3.4a}$$

$$\underline{\mathbf{Y}}_h^{k+1} := \left\{\underline{\mathrm{q}}_h = \left((\mathrm{q}_T)_{T\in\mathcal{T}_h}, (\mathrm{q}_F)_{F\in\mathcal{F}_h}\right) : \begin{array}{ll} \mathrm{q}_T \in \mathcal{P}^k(T) & \forall T\in\mathcal{T}_h, \\ \mathrm{q}_F \in \mathcal{P}^{k+1}(F) & \forall F\in\mathcal{F}_h \end{array}\right\}. \tag{3.4b}$$

It is also useful to consider global spaces of broken polynomials:

$$\mathcal{P}^k(\mathcal{T}_h) := \{q_h \in L^2(\Omega) : q_h|_T \in \mathcal{P}^k(T) \quad \forall T\in\mathcal{T}_h\},$$
$$\boldsymbol{\mathcal{P}}^k(\mathcal{T}_h) := \{\boldsymbol{v}_h \in L^2(\Omega) : \boldsymbol{v}_h|_T \in \boldsymbol{\mathcal{P}}^k(T) \quad \forall T\in\mathcal{T}_h\}.$$

To any element $\underline{\boldsymbol{v}}_h \in \underline{\mathbf{X}}_{\mathfrak{b},h}^{k+1}$ it is possible to associate a broken polynomial $\boldsymbol{v}_h \in \boldsymbol{\mathcal{P}}^(\mathcal{T}_h)$ such that $\boldsymbol{v}_h|_T = \boldsymbol{v}_T\ \forall T$. Similarly, we consider the broken scalar polynomial q_h such that $q_h|_T = q_T\ \forall T$, for a given element $\underline{q}_h$ of $\underline{\mathrm{Y}}_h^{k+1}$.
Finally we can extend to the global setting the notion of reconstruction:

Definition 6. *The **global gradient reconstruction** operator $\boldsymbol{G}_h^{k+1} : \underline{\mathrm{Y}}_h^{k+1} \to \boldsymbol{\mathcal{P}}^{k+1}(\mathcal{T}_h)$ associates to $\underline{q}_h \in \underline{\mathrm{Y}}_h^{k+1}$ a broken polynomial $\boldsymbol{G}_h^{k+1}\underline{q}_h$ such that:*

$$\boldsymbol{G}_h^{k+1}\underline{q}_h|_T = \boldsymbol{G}_T^{k+1}\underline{q}_T.$$

*The **global curl reconstruction** operator $\boldsymbol{C}_h^k : \underline{\mathbf{X}}_{\mathfrak{b},h}^{k+1} \to \boldsymbol{\mathcal{R}}^k(\mathcal{T}_h)$ associates to $\underline{\boldsymbol{v}}_h \in \underline{\mathbf{X}}_{\mathfrak{b},h}^{k+1}$ a broken polynomial $\boldsymbol{C}_h^k\underline{\boldsymbol{v}}_h$ such that:*

$$\boldsymbol{C}_h^k\underline{\boldsymbol{v}}_h|_T = \boldsymbol{C}_T^k\underline{\boldsymbol{v}}_T.$$

The commutation properties of 3.3 can be recast globally:

$$\boldsymbol{G}_h^{k+1}\underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1}q = \pi_h^{k+1}(\mathbf{grad}\, q), \tag{3.5a}$$

$$\boldsymbol{C}_h^{k}\underline{\mathbf{I}}_{\mathbf{X},h}^{k+1}\boldsymbol{v} = \pi_{C_h}^{k+1}(\mathbf{curl}\, \boldsymbol{v}). \tag{3.5b}$$

The analysis of the present application of the HHO method relies on a discrete version of the second Weber inequality, a theoretical tool that can be used in demonstrations of well-posedness of the magnetostatics problems in a variational setting. It states a result similar to Poincaré inequality, allowing to bound the L_2 norm with the curl seminorm for a certain category of functions. At the continuous level, the following theorem holds:

Theorem 3.1. *Given the Sobolev spaces:*

$$\begin{aligned} H(\mathbf{curl};\Omega) &= \{\boldsymbol{u}\in L^2(\Omega,\mathbb{R}^3) : \mathbf{curl}\,\boldsymbol{u}\in L^2(\Omega,\mathbb{R}^3)\}\\ H(\mathbf{div};\Omega) &= \{\boldsymbol{u}\in L^2(\Omega,\mathbb{R}^3) : \mathbf{div}\,\boldsymbol{u}\in L^2(\Omega)\}\\ H_0(\mathbf{div}^0;\Omega) &= \{\boldsymbol{u}\in H(\mathbf{div};\Omega) : \mathbf{div}\,\boldsymbol{u}=0\,,\ \boldsymbol{u}\cdot\boldsymbol{n}=0 \quad \text{on } \partial\Omega\} \end{aligned}$$

for any $\boldsymbol{u}\in H(\mathbf{curl};\Omega)\cap H_0(\mathbf{div}^0;\Omega)$*:*

$$\|\boldsymbol{u}\|_\Omega \le C_w \|\mathbf{curl}\,\boldsymbol{u}\|_\Omega$$

for some $C_w>0$ *only depending on the domain* Ω*.*

To show the validity of a discrete version of theorem (3.1) we will need the following technical result.

Lemma 3.2 (Polynomial decomposition)**.** *Let* $q\in\mathbb{N}$*. Let* $T\in\mathcal{T}_h$ *and* $\boldsymbol{p}\in\boldsymbol{\mathcal{P}}^q(T)$*. Then, there exist* $g\in\mathcal{P}^{q+1}(T)$ *and* $\boldsymbol{c}\in\boldsymbol{\mathcal{P}}^q(T)$ *such that* $\boldsymbol{p}=\mathbf{grad}\, g+(\boldsymbol{x}-\boldsymbol{x}_T)\times\mathbf{curl}\,\boldsymbol{c}$ *and*

$$\|\boldsymbol{p}-\mathbf{grad}\, g\|_T \le 2h_T\|\mathbf{curl}\,\boldsymbol{p}\|_T. \tag{3.6}$$

The next step is to equip the space $\underline{\mathbf{X}}_{\flat,h}^{k+1}$ with the seminorm $\|\cdot\|_{\mathbf{X},h,\flat}$ defined by

$$\|\underline{\mathbf{v}}_h\|_{\mathbf{X},h,\flat}^2 := \|\mathbf{curl}_h\mathbf{v}_h\|_\Omega^2 + \sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T} h_F^{-1}\|\boldsymbol{\pi}_{\mathcal{P},\flat}^{k+1}\big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\big)\|_F^2. \tag{3.7}$$

This seminorm is the discrete counterpart of the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-norm of the **curl** and is composed of two contributions: the first one is the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-norm of the broken **curl** of the function obtained patching the polynomials attached to mesh elements; the second one accounts for the difference between the face unknowns and the (gradient part of the) tangential trace of the element unknowns. $\boldsymbol{\gamma}_{\tau,F}$ has been used to indicate the tangential trace operator.

The discret result which is going to be discussed is valid on the following counterpart of the space $\boldsymbol{H}(\mathbf{curl};\Omega)\cap\boldsymbol{H}_0(\mathbf{div}^0;\Omega)$ defined by

$$\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}:=\left\{\underline{\mathbf{w}}_h\in\underline{\mathbf{X}}_{\flat,h}^{k+1}\ :\ \left(\mathbf{w}_h,\boldsymbol{G}_h^{k+1}\underline{\mathrm{q}}_h\right)_\Omega=0\quad\forall\underline{\mathrm{q}}_h\in\underline{\mathrm{Y}}_h^{k+1}\right\},\tag{3.8}$$

We call it a discrete version of $\boldsymbol{H}(\mathbf{curl};\Omega)\cap\boldsymbol{H}_0(\mathbf{div}^0;\Omega)$ as the condition in (3.8) is the discrete version of requiring $(\boldsymbol{w},\mathbf{grad}\,q)_\Omega=0\quad\forall\,q\in H_0^1(\Omega)$, which entails $\mathbf{div}\,\boldsymbol{w}=0$, $\boldsymbol{w}\cdot\boldsymbol{n}=0\quad$ *on* $\partial\Omega$. Here and in what follows, the circle overset will be reserved to those hybrid spaces that incorporate a discrete divergence-free property (see, in particular, (3.42)).

Theorem 3.3 (Discrete Weber inequality)**.** *There exists a constant $c_{\mathrm{W}}>0$ independent of h such that, for all $\underline{\mathbf{v}}_h\in\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$, one has*

$$\|\mathbf{v}_h\|_\Omega\le c_{\mathrm{W}}\|\underline{\mathbf{v}}_h\|_{\mathbf{X},h,\flat},\tag{3.9}$$

where we recall that the term in the left-hand side is the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-norm of the broken polynomial vector $\mathbf{v}_h$ such that $(\mathbf{v}_h)_{|T}=\mathbf{v}_T$ for all $T\in\mathcal{T}_h$.

Proof. Let $\underline{\mathbf{v}}_h\in\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$. Recall the following standard $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-orthogonal Helmholtz decomposition (*2nd Helmholtz decomposition*) (cf., e.g., [ACL18, Proposition 3.7.3]):

$$\boldsymbol{L}^2(\Omega;\mathbb{R}^3)=\mathbf{grad}\Big(H^1_{zmv}(\Omega)\Big)\overset{\perp}{\oplus}\boldsymbol{H}_0(\mathbf{div}^0;\Omega).$$

By the characterization of divergence-free functions from [ACL18, Theorem 3.5.1], and since $\mathbf{v}_h\in\boldsymbol{\mathcal{P}}^{k+1}(\mathcal{T}_h)\subset\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$ and Ω is simply connected, we can write

$$\mathbf{v}_h=\mathbf{grad}\,\varphi+\mathbf{curl}\,\boldsymbol{\psi},\tag{3.10}$$

for some $\varphi\in H^1(\Omega)_{zmv}$, and some $\boldsymbol{\psi}\in\boldsymbol{H_0(\mathbf{curl};\Omega)}$ such that $\mathbf{div}\,\boldsymbol{\psi}=0$. Furthermore,

$$\|\boldsymbol{\psi}\|_{\boldsymbol{H(\mathbf{curl};\Omega)}}\lesssim\|\mathbf{curl}\,\boldsymbol{\psi}\|_\Omega.\tag{3.11}$$

Using the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-orthogonal decomposition (3.10) of $\mathbf{v}_h$, we have that

$$\|\mathbf{v}_h\|_\Omega^2=(\mathbf{v}_h,\mathbf{grad}\,\varphi)_\Omega+\left(\mathbf{v}_h,\mathbf{curl}\,\boldsymbol{\psi}\right)_\Omega=:\mathcal{I}_1+\mathcal{I}_2.\tag{3.12}$$

For the first term in 3.12, setting $\underline{\varphi}_h:=\underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1}\varphi\in\underline{\mathrm{Y}}_h^{k+1}$, we infer that

$$\mathcal{I}_1=\left(\mathbf{v}_h,\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},h}^{k+1}(\mathbf{grad}\,\varphi)\right)_\Omega=\left(\mathbf{v}_h,\boldsymbol{G}_h^{k+1}\underline{\varphi}_h\right)_\Omega=0,\tag{3.13}$$

where we have used that $\mathbf{v}_h\in\boldsymbol{\mathcal{P}}^{k+1}(\mathcal{T}_h)$ to insert $\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},h}^{k+1}$, followed by the commutation property (3.5a) of $\boldsymbol{G}_h^{k+1}$ and the fact that $\underline{\mathbf{v}}_h\in\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$.

Let us now estimate the second term in (3.12). We have

$$\begin{aligned}\mathcal{I}_2 &= \sum_{T\in\mathcal{T}_h}\Big(\mathbf{v}_T,\mathbf{curl}\,\psi\Big)_T = \sum_{T\in\mathcal{T}_h}\left(\Big(\mathbf{curl}\,\mathbf{v}_T,\psi\Big)_T - \sum_{F\in\mathcal{F}_T}\Big(\psi_{|F}\times\boldsymbol{n}_{TF},\mathbf{v}_{T|F}\Big)_F\right)\\ &= \sum_{T\in\mathcal{T}_h}\left(\Big(\mathbf{curl}\,\mathbf{v}_T,\psi\Big)_T - \sum_{F\in\mathcal{F}_T}\Big(\boldsymbol{\gamma}_{\tau,F}(\psi\times\boldsymbol{n}_{TF}),\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\Big)_F\right),\end{aligned}$$

where we have used an integration by parts formula on each mesh element $T\in\mathcal{T}_h$ in the first line, and the fact that the jumps of ψ vanish on interfaces along with $\psi_{|F}\times\boldsymbol{n}_{TF}=0$ for all $F\in\mathcal{F}_h^{\mathrm{b}}$ to insert $\mathbf{v}_F$ into the second term in the second line. Applying Cauchy–Schwarz inequalities to the right-hand side, we obtain

$$\begin{aligned}\mathcal{I}_2 \le &\left(\sum_{T\in\mathcal{T}_h}\Big(\|\mathbf{curl}\mathbf{v}_T\|_T^2 + \sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\|_F^2\Big)\right)^{1/2}\\ &\times\left(\sum_{T\in\mathcal{T}_h}\Big(\|\psi\|_T^2+\sum_{F\in\mathcal{F}_T}h_F\|\psi_{|F}\times\boldsymbol{n}_{TF}\|_F^2\Big)\right)^{1/2}.\end{aligned}\tag{3.14}$$

We focus on the first factor on the right-hand side of (3.14). For $T\in\mathcal{T}_h$ and $F\in\mathcal{F}_T$, decomposing $\mathbf{v}_T\in\mathcal{P}^{k+1}(T)$ along (3.6) as $\mathbf{v}_T=\mathbf{grad}\,g+(\boldsymbol{x}-\boldsymbol{x}_T)\times\mathbf{curl}\,\boldsymbol{c}$ with $g\in\mathcal{P}^{k+2}(T)$ and $\boldsymbol{c}\in\mathcal{P}^{k+1}(T)$, inserting into the norm $\pm\Big[\boldsymbol{\gamma}_{\tau,F}(\mathbf{grad}\,g)+\pi_{\mathcal{P},\mathrm{b}}^{k+1}\Big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)\Big)\Big]$, and using the triangle inequality, we infer, since $\mathbf{v}_F\in\mathcal{P}_{\mathrm{b}}^{k+1}(F)$, since $\boldsymbol{\gamma}_{\tau,F}(\mathbf{grad}\,g)=\mathbf{grad}_\tau(g_{|F})\in\boldsymbol{\mathcal{G}}^{k+1}(F)$, and recalling $\boldsymbol{G}^{k+1}(F)\subset\boldsymbol{\mathcal{P}}_{\mathrm{b}}^{k+1}(F)$:

$$\begin{aligned}\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\|_F^2 \lesssim &\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}_{\mathcal{P},\mathrm{b}}^{k+1}\Big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\Big)\|_F^2\\ &+\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\gamma}_{\tau,F}\Big(\mathbf{v}_T-\mathbf{grad}\,g\Big)\|_F^2\\ &+\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}_{\mathcal{P},\mathrm{b}}^{k+1}\Big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T-\mathbf{grad}\,g)\Big)\|_F^2.\end{aligned}\tag{3.15}$$

Using the $\boldsymbol{L}^2(F;\mathbb{R}^2)$-boundedness of $\boldsymbol{\pi}_{\mathcal{P}}^{k+1}$, a discrete trace inequality (cf., e.g., [DD20, Lemma 1.32]), and Lemma 3.2 with $\boldsymbol{p}=\mathbf{v}_T$, we infer

$$\begin{aligned}&\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\|_F^2\\ &\qquad\lesssim\sum_{T\in\mathcal{T}_h}\Big(\|\mathbf{curl}\,\mathbf{v}_T\|_T^2+\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}_{\mathcal{P}}^{k+1}\Big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T)-\mathbf{v}_F\Big)\|_F^2\Big).\end{aligned}\tag{3.16}$$

Now, for the second factor on the right-hand side of (3.14), using that $|\psi_{|F}\times\boldsymbol{n}_{TF}|\le|\psi_{|F}|$, a continuous trace inequality (cf., e.g., [DD20, Lemma 1.31]), the fact that $h_T\le\mathrm{diam}(\Omega)$

for all $T \in \mathcal{T}_h$, and concluding with (3.11), one has

$$\left(\sum_{T\in\mathcal{T}_h}\left(\|\boldsymbol{\psi}\|_T^2 + \sum_{F\in\mathcal{F}_T} h_F\|\boldsymbol{\psi}_{|F}\times\boldsymbol{n}_{TF}\|_F^2\right)\right)^{1/2} \lesssim \|\boldsymbol{\psi}\|_{\boldsymbol{H}(\mathbf{curl};\Omega)} \lesssim \|\mathbf{curl}\,\boldsymbol{\psi}\|_\Omega. \tag{3.17}$$

Plugging (3.16) and (3.17) into (3.14), and recalling the definition (3.7) of the $\|\cdot\|_{\mathbf{X},h,\flat}$-seminorm yields

$$\mathcal{I}_2 \lesssim \|\underline{\mathbf{v}}_h\|_{\mathbf{X},h}\|\mathbf{curl}\,\boldsymbol{\psi}\|_\Omega \le \|\underline{\mathbf{v}}_h\|_{\mathbf{X},h}\|\mathbf{v}_h\|_\Omega,$$

where we have used the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-orthogonality of the decomposition (3.10) in the last bound. We conclude by combining (3.12), (3.13), and this last estimate. □

A direct consequence of the theorem is the following corollary:

Corollary 3.4 (Norm $\|\cdot\|_{\mathbf{X},h,\flat}$)**.** *The map $\|\cdot\|_{\mathbf{X},h,\flat}$ defines a norm on $\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$ defined by (3.8).*

Proof. This is a direct consequence of Theorem (3.3) and of the definition (3.7). For all $\underline{\mathbf{v}}_h \in \mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$, if $\|\underline{\mathbf{v}}_h\|_{\mathbf{X},h,\flat} = 0$, then $\mathbf{v}_h = \mathbf{0}$, i.e. $\mathbf{v}_T = \mathbf{0}$ for all $T \in \mathcal{T}_h$. Then, for all $F \in \mathcal{F}_h$, $\|\boldsymbol{\pi}_{\mathcal{P},\flat}^{k+1}(\mathbf{v}_F)\|_F = \|\mathbf{v}_F\|_F = 0$, i.e. $\mathbf{v}_F = \mathbf{0}$, whence $\underline{\mathbf{v}}_h = \underline{\mathbf{0}}_h$. □

This last corollary will be instrumental in the analysis of the HHO method of Section **??**

Corollary 3.5 (Generalized discrete Weber inequality)**.** *Let $\mathrm{d}_h : \underline{\mathrm{Y}}_h^{k+1} \times \underline{\mathrm{Y}}_h^{k+1} \to \mathbb{R}$ be a symmetric positive semi-definite bilinear form such that, for all $\varphi \in H^1_{zmv}(\Omega)$, letting $\underline{\varphi}_h := \underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1}\varphi \in \underline{\mathrm{Y}}_h^{k+1}$,*

$$\mathrm{d}_h\big(\underline{\varphi}_h, \underline{\varphi}_h\big)^{1/2} \lesssim \|\mathbf{grad}\,\varphi\|_\Omega. \tag{3.18}$$

Then, there is $c_{\mathrm{W}} > 0$ independent of h such that, for all $(\underline{\mathbf{v}}_h, \underline{\mathrm{r}}_h) \in \underline{\mathbf{X}}_{\flat,h}^{k+1}\times\underline{\mathrm{Y}}_h^{k+1}$ satisfying

$$-\big(\mathbf{v}_h, \boldsymbol{G}_h^{k+1}\underline{\mathrm{q}}_h\big)_\Omega + \mathrm{d}_h\big(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h\big) = 0 \qquad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1}, \tag{3.19}$$

one has

$$\|\mathbf{v}_h\|_\Omega \le c_{\mathrm{W}}\Big(\|\underline{\mathbf{v}}_h\|_{\mathbf{X},h}^2 + \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h)\Big)^{1/2}. \tag{3.20}$$

Proof. We follow the steps of the proof of Theorem (3.3). If $(\underline{\mathbf{v}}_h, \underline{\mathrm{r}}_h) \in \underline{\mathbf{X}}_{\flat,h}^{k+1}\times\underline{\mathrm{Y}}_h^{k+1}$ satisfies (3.19), then (3.13) becomes

$$\mathcal{I}_1 = (\mathbf{v}_h, \mathbf{grad}\,\varphi)_\Omega = \big(\mathbf{v}_h, \boldsymbol{G}_h^{k+1}\underline{\varphi}_h\big)_\Omega = \mathrm{d}_h\big(\underline{\mathrm{r}}_h, \underline{\varphi}_h\big).$$

By the Cauchy–Schwarz inequality and (3.18), we infer

$$\mathcal{I}_1 \lesssim \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h)^{1/2}\|\mathbf{grad}\,\varphi\|_\Omega \le \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h)^{1/2}\|\mathbf{v}_h\|_\Omega, \tag{3.21}$$

where we have used the $\boldsymbol{L}^2(\Omega;\mathbb{R}^3)$-orthogonality of the decomposition (3.10) in the last bound. The rest of the proof is unchanged provided we substitute (3.21) to (3.13). □

3.2 HHO schemes for magnetostatics

Let $\Omega \in \mathbb{R}^3$ be an open, connected domain with a connected border. We consider the potential formulation of the problem of magnetostatics, namely that of determining the vector field $\boldsymbol{u} : \Omega \to \mathbb{R}^3$ representing the *magnetic vector potential* subject to the following constraints:

$$\begin{cases} \mathbf{curl}(\mathbf{curl}\,\boldsymbol{u}) = \boldsymbol{J} & \text{in}\,\Omega \\ \mathbf{div}(\boldsymbol{u}) = 0 & \text{in}\,\Omega \\ \boldsymbol{n} \times (\boldsymbol{u} \times \boldsymbol{n}) = \boldsymbol{0} & \text{on}\,\partial\Omega \end{cases} \tag{3.22}$$

The first line of (3.22) expresses *Ampère's law* (cfr [ACL18, Eq 1.38]), where the vector field $\boldsymbol{J} : \Omega \to \mathbb{R}^3$ represents the *current density* in the domain, acting as a source for the magnetic field. The second equation requires $\boldsymbol{u}$ to be solenoidal, a condition known as *Coulomb gauge*. Finally, an homogeneous Dirichlet boundary condition fixes the value of the component of $\boldsymbol{u}$ tangential to $\partial\Omega$.

This set of equations can be recast in a variational formulation via the usual product by test function and integration by part technique, leading to the following second-order well posed problem (see [ACL18, sec. 7.2]):
Find $(\boldsymbol{u}, p) \in \boldsymbol{H}_0(\mathbf{curl}, \Omega) \times H^1_0(\Omega)$ such that:

$$(\mathbf{curl}\,\boldsymbol{u}, \mathbf{curl}\,\boldsymbol{v})_\Omega + (\boldsymbol{v}, \mathbf{grad}\,p)_\Omega = (\boldsymbol{J}, \mathbf{curl}\,\boldsymbol{v}) \forall\, \boldsymbol{v} \in \boldsymbol{H_0}(\mathbf{curl}; \Omega) \tag{3.23a}$$

$$(\boldsymbol{u}, \mathbf{grad}\,q)_\Omega = 0 \qquad \forall\, q \in H^1_0(\Omega), \tag{3.23b}$$

where

$$\boldsymbol{H_0}(\mathbf{curl}; \Omega) = \{w \in \boldsymbol{L}^2(\Omega, \mathbb{R}^3) : \mathbf{curl}\, w \in \boldsymbol{L}^2(\Omega, \mathbb{R}^3),\ \boldsymbol{n} \times (w \times \boldsymbol{n}) = \boldsymbol{0} \qquad \text{on}\ \partial\Omega\}.$$

In problem (3.23) p represents an artificial scalar unknown serving as Lagrange multiplier to impose the divergence-free constraint on $\boldsymbol{u}$. If $\mathbf{div}\,\boldsymbol{J} = \boldsymbol{0}$ in Ω and $\boldsymbol{J} \cdot \boldsymbol{n} = \boldsymbol{0}$ on $\partial\Omega$, it can be easily shown that $p = 0$ (cfr [FS20] and [Kik89]).

On the entire $\partial\Omega$ or on a portion $\Gamma_H \subset \partial\Omega$ an alternative Neumann boundary condition can be imposed. In case of fully Neumann boundary conditions the strong form of the problem reads as follows:

$$\mathbf{curl}\Big(\mathbf{curl}\,\boldsymbol{u}\Big) = \boldsymbol{J} \qquad \text{in}\ \Omega, \tag{3.24a}$$

$$\mathbf{div}\,\boldsymbol{u} = 0 \qquad \text{in}\ \Omega, \tag{3.24b}$$

$$\boldsymbol{u} \cdot \boldsymbol{n} = \boldsymbol{0} \qquad \text{on}\ \partial\Omega, \tag{3.24c}$$

$$\boldsymbol{n} \times (\mathbf{curl}\,\boldsymbol{u} \times \boldsymbol{n}) = \boldsymbol{n} \times (\mathbf{curl}\,\boldsymbol{\Phi} \times \boldsymbol{n}) \qquad \text{on}\ \partial\Omega, \tag{3.24d}$$

where $\boldsymbol{\Phi} : \Omega \to \mathbb{R}^3$ is a provided field. The corresponding weak formulation reads:
Find $(\boldsymbol{u}, p) \in \boldsymbol{H}(\mathbf{curl}, \Omega) \times H^1(\Omega)$ such that:

$$(\mathbf{curl}\,\boldsymbol{u},\mathbf{curl}\,\boldsymbol{v})_\Omega + (\boldsymbol{v},\mathbf{grad}\,p)_\Omega = (\boldsymbol{J},\boldsymbol{v}) \qquad \forall\,\boldsymbol{v}\in\boldsymbol{H}(\mathbf{curl};\Omega) \tag{3.25a}$$

$$(\boldsymbol{u},\mathbf{grad}\,q)_\Omega + (p,q)_\Omega = (\boldsymbol{\Phi}\times\boldsymbol{n},\boldsymbol{n}\times(\boldsymbol{v}\times\boldsymbol{n}))_{\partial\Omega} \qquad \forall\,q\in H^1(\Omega). \tag{3.25b}$$

Finally the magnetostatics problem with mixed boundary conditions reads in its strong formulation:

$$\mathbf{curl}\Big(\mathbf{curl}\,\boldsymbol{u}\Big) = \boldsymbol{J} \qquad \text{in } \Omega, \tag{3.26a}$$

$$\mathbf{div}\,\boldsymbol{u} = 0 \qquad \text{in } \Omega, \tag{3.26b}$$

$$\boldsymbol{u}\cdot\boldsymbol{n} = \boldsymbol{0} \qquad \text{on } \Gamma_H, \tag{3.26c}$$

$$\boldsymbol{n}\times(\mathbf{curl}\,\boldsymbol{u}\times\boldsymbol{n}) = \boldsymbol{n}\times(\mathbf{curl}\,\boldsymbol{\Phi}\times\boldsymbol{n}) \qquad \text{on } \Gamma_H, \tag{3.26d}$$

$$\boldsymbol{n}\times(\boldsymbol{u}\times\boldsymbol{n}) = \boldsymbol{0} \qquad \text{on } \Gamma_B, \tag{3.26e}$$

where $\partial\Omega = \Gamma_H\cup\Gamma_B$, $\Gamma_H\cap\Gamma_B = \emptyset$ and the field $\boldsymbol{\Phi}\in\boldsymbol{H}(\mathbf{curl},\Omega)$ is provided. The weak formulation reads as follows:

Find $(\boldsymbol{u},p)\in\boldsymbol{H}_{\Gamma_B}(\mathbf{curl},\Omega)\times H^1_{\Gamma_B}(\Omega)$ such that:

$$(\mathbf{curl}\,\boldsymbol{u},\mathbf{curl}\,\boldsymbol{v})_\Omega + (\boldsymbol{v},\mathbf{grad}\,p)_\Omega = (\boldsymbol{J},\boldsymbol{v}) + (\boldsymbol{\Phi}\times\boldsymbol{n},\boldsymbol{n}\times(\boldsymbol{v}\times\boldsymbol{n}))_{\partial\Omega} \qquad \forall\,\boldsymbol{v}\in\boldsymbol{H}_{\Gamma_B}(\mathbf{curl},\Omega) \tag{3.27a}$$

$$(\boldsymbol{u},\mathbf{grad}\,q)_\Omega = 0 \qquad \forall\,q\in H^1_{\Gamma_B}(\Omega), \tag{3.27b}$$

where

$$\boldsymbol{H}_{\Gamma_B}(\mathbf{curl},\Omega) := \{\boldsymbol{w}\in\boldsymbol{L}^2(\Omega,\mathbb{R}^3) : \mathbf{curl}\,\boldsymbol{w}\in\boldsymbol{L}^2(\Omega,\mathbb{R}^3), \boldsymbol{n}\times(\boldsymbol{w}\times\boldsymbol{n}) = \boldsymbol{0} \qquad \text{on } \Gamma_B\}$$

Proof of well-posedness of the problem for varying boundary conditions can be found in [ACL18]. It has to be remarked that in the analysis it is fundamental to assume the topological assumption on Ω. If the domain is not connected or its boundary is not connected well-posedness may not be granted.
To HHO method is built as a discretization of the weak problem (3.23). Such discretization starts considering a polyhedral mesh $\mathcal{M}_h = (\mathcal{T}_h,\mathcal{F}_h)$, as defined of section 3.

The discrete unknowns of the HHO method are collections of polynomials local to both cells and faces. In particular we recall that the following hybrid spaces are considered:

$$\underline{\mathbf{X}}_h^{k+1} := \left\{\underline{\mathbf{v}}_h = \Big((\mathbf{v}_T)_{T\in\mathcal{T}_h},(\mathbf{v}_F)_{F\in\mathcal{F}_h}\Big) : \begin{matrix}\mathbf{v}_T\in\boldsymbol{\mathcal{P}}^{k+1}(T) & \forall T\in\mathcal{T}_h,\\ \mathbf{v}_F\in\mathcal{P}_{\mathrm{b}}^{k+1}(F) & \forall F\in\mathcal{F}_h\end{matrix}\right\}, \tag{3.28a}$$

$$\underline{\mathrm{Y}}_h^{k+1} := \left\{\underline{\mathrm{q}}_h = \Big((\mathrm{q}_T)_{T\in\mathcal{T}_h},(\mathrm{q}_F)_{F\in\mathcal{F}_h}\Big) : \begin{matrix}\mathrm{q}_T\in\mathcal{P}^{k}(T) & \forall T\in\mathcal{T}_h,\\ \mathrm{q}_F\in\mathcal{P}^{k+1}(F) & \forall F\in\mathcal{F}_h\end{matrix}\right\}. \tag{3.28b}$$

We will as well consider subspaces incorporating strongly homogeneous Dirichlet boundary conditions on the entire boundary, or a portion Γ_B:

$$\begin{aligned}
\underline{\mathbf{X}}_{\mathrm{b},h,\mathbf{0}}^{k+1} &:= \left\{\underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} : \mathbf{v}_F = \mathbf{0} \quad \forall F \in \mathcal{F}_h^b\right\},\\
\underline{\mathrm{Y}}_{h,0}^{k+1} &:= \left\{\underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1} : \mathrm{q}_F = 0 \quad \forall F \in \mathcal{F}_h^b\right\},\\
\underline{\mathbf{X}}_{h,\Gamma_B,\mathrm{b}}^{k+1} &:= \left\{\underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} : \mathbf{v}_F = \mathbf{0} \quad \forall F \in \mathcal{F}_h^b \cap \Gamma_B\right\},\\
\underline{\mathrm{Y}}_{h,\Gamma_B}^{k+1} &:= \left\{\underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1} : \mathrm{q}_F = 0 \quad \forall F \in \mathcal{F}_h^b \cap \Gamma_B\right\}.
\end{aligned} \tag{3.29}$$

Given a mesh element $T \in \mathcal{T}_h$, we denote by $\underline{\mathbf{X}}_T^{k+1}$ and $\underline{\mathrm{Y}}_T^{k+1}$ the restrictions of respectively $\underline{\mathbf{X}}_{\mathrm{b},h}^{k+1}$ and $\underline{\mathrm{Y}}_h^{k+1}$ to T, and by $\underline{\mathbf{v}}_T := \left(\mathbf{v}_T, (\mathbf{v}_F)_{F\in\mathcal{F}_T}\right) \in \underline{\mathbf{X}}_T^{k+1}$ and $\mathrm{q}_{T,F} := \left(\mathrm{q}_T, (\mathrm{q}_F)_{F\in\mathcal{F}_T}\right) \in \underline{\mathrm{Y}}_T^{k+1}$ the respective restrictions to element T of generic vectors of polynomials $\underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1}$ and $\underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1}$.
We let $\mathbf{v}_h$ and q_h (not underlined) be the broken polynomial functions in $\boldsymbol{\mathcal{P}}^{k+1}(\mathcal{T}_h)$ and in $\mathcal{P}^k(\mathcal{T}_h)$ such that

$$(\mathbf{v}_h)_{|T} := \mathbf{v}_T \quad \text{and} \quad (\mathrm{q}_h)_{|T} := \mathrm{q}_T \quad \text{for all } T \in \mathcal{T}_h.$$

We are then ready to state the HHO discretization of problem (3.23):
Find $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbf{X}}_{\mathrm{b},h,\mathbf{0}}^{k+1} \times \underline{\mathrm{Y}}_{h,0}^{k+1}$ such that:

$$\mathrm{a}_h(\underline{\mathbf{u}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{\mathrm{p}}_h) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega \qquad \forall \underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h,\mathbf{0}}^{k+1}, \tag{3.30a}$$

$$-\mathrm{b}_h(\underline{\mathbf{u}}_h, \underline{\mathrm{q}}_h) + \mathrm{d}_h(\underline{\mathrm{p}}_h, \underline{\mathrm{q}}_h) = 0 \qquad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_{h,0}^{k+1}. \tag{3.30b}$$

where we define the discrete bilinear forms $\mathrm{a}_h : \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \to \mathbb{R}$, $\mathrm{b}_h : \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathrm{Y}}_h^{k+1} \to \mathbb{R}$, and $\mathrm{c}_h : \underline{\mathrm{Y}}_h^{k+1} \times \underline{\mathrm{Y}}_h^{k+1} \to \mathbb{R}$ as follows:

$$\mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h) := \left(\boldsymbol{C}_h^k \underline{\boldsymbol{w}}_h, \boldsymbol{C}_h^k \underline{\boldsymbol{v}}_h\right)_\Omega + \mathrm{s}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h), \tag{3.31a}$$

$$\mathrm{b}_h(\underline{\mathbf{w}}_h, \underline{\mathrm{q}}_h) := \left(\mathbf{w}_h, \boldsymbol{G}_h^{k+1} \underline{\mathrm{q}}_h\right)_\Omega, \tag{3.31b}$$

$$\mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h) := \sum_{T\in\mathcal{T}_h} \sum_{F\in\mathcal{F}_T} h_F (\mathrm{r}_F, \mathrm{q}_F)_F. \tag{3.31c}$$

$\mathrm{s}_h : \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \to \mathbb{R}$ is the stabilization bilinear form:

$$\mathrm{s}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h) := \sum_{T\in\mathcal{T}_h} \sum_{F\in\mathcal{F}_T} h_F^{-1} \left(\boldsymbol{\pi}_{\mathcal{G},F}^{k+1}\left(\boldsymbol{\gamma}_{\tau,F}(\mathbf{w}_T) - \mathbf{w}_F\right), \boldsymbol{\pi}_{\mathcal{G},F}^{k+1}\left(\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T) - \mathbf{v}_F\right)\right)_F. \tag{3.32}$$

as it is proposed in [FS20]. One can notice the distinguishing features of the HHO method: occurrences of differential operators **curl** and **grad** are substituted with the corresponding discrete reconstruction operators introduced in 3.5. Moreover extra stabilisation terms s_h and d_h are added, which are symmetric, and positive semidefinite.

By setting: $\underline{\mathbb{Z}}_{\flat,h,\mathbb{0}}^{k+1} := \underline{\mathbf{X}}_{\flat,h,\mathbf{0}}^{k+1} \times \underline{\mathrm{Y}}_{h,0}^{k+1}$ Problem (3.30) can be reformulated as follows: find $\underline{\mathbb{z}}_h = (\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}$ such that:

$$\mathrm{A}_h\Big(\underline{\mathbb{z}}_h, \mathrm{s}_h\Big) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega \qquad \forall\, \mathrm{s}_h = (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) \in \underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}, \tag{3.33}$$

where the bilinear form $\mathrm{A}_h : \underline{\mathbb{Z}}_h^{k+1} \times \underline{\mathbb{Z}}_h^{k+1} \to \mathbb{R}$ is defined by

$$\mathrm{A}_h\Big((\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h), (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\Big) := \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{\mathrm{r}}_h) - \mathrm{b}_h(\underline{\mathbf{w}}_h, \underline{\mathrm{q}}_h) + \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h). \tag{3.34}$$

The analysis brought on in [FS20] for the case of the Dirichlet problem establishes that problem (4.2) admits a unique solution, stable with respect to data. Moreover the following result is proved, providing the convergence order estimate with respect to the local polynomial degree.

Theorem 3.6 (Energy-error estimate)**.** *Assume that*

$$\boldsymbol{u} \in \boldsymbol{H}_{\mathbf{0}}(\mathbf{curl};\Omega) \cap \boldsymbol{H}^1(\Omega;\mathbb{R}^3) \cap \boldsymbol{H}^{k+2}(\mathcal{T}_h;\mathbb{R}^3), \qquad p \in H_0^1(\Omega) \cap H^{k+1}(\mathcal{T}_h).$$

Consider $\widehat{\underline{\mathbf{u}}}_h := \underline{\mathbf{I}}_{\mathbf{X},h}^{k+1} \boldsymbol{u} \in \underline{\mathbf{X}}_{\flat,h,\mathbf{0}}^{k+1}$ *and* $\widehat{\underline{\mathrm{p}}}_h := \underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1} p \in \underline{\mathrm{Y}}_{h,0}^{k+1}$ *and define the errors*

$$\underline{\mathbf{X}}_{h,\mathbf{0}}^{k+1} \ni \underline{\mathbf{e}}_h := \underline{\mathbf{u}}_h - \widehat{\underline{\mathbf{u}}}_h, \qquad \underline{\mathrm{Y}}_{h,0}^{k+1} \ni \underline{\epsilon}_h := \underline{\mathrm{p}}_h - \widehat{\underline{\mathrm{p}}}_h,$$

Then, the following holds true:

$$\|(\underline{\mathbf{e}}_h, \underline{\epsilon}_h)\|_{\mathbb{Z},h,\flat} \lesssim \left[\sum_{T\in\mathcal{T}_h} h_T^{2(k+1)} \left(|\boldsymbol{u}|^2_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)} + |p|^2_{H^{k+1}(T)}\right)\right]^{1/2}, \tag{3.35}$$

where we equip the spaces $\underline{\mathbf{X}}_{\flat,h}^{k+1}$ *and* $\underline{\mathrm{Y}}_h^{k+1}$ *with the seminorms*

$$\|\underline{\mathbf{w}}_h\|_{\mathbf{X},h,\flat} := \left(\|\mathbf{curl}_h \mathbf{w}_h\|^2_\Omega + \mathrm{s}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h)\right)^{1/2}, \tag{3.36a}$$

$$\|\underline{\mathrm{r}}_h\|_{\mathrm{Y},h} := \left(\sum_{T\in\mathcal{T}_h} h_T^2 \|\mathbf{grad}\, \mathrm{r}_T\|^2_T + \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h)\right)^{1/2}, \tag{3.36b}$$

and the space $\underline{\mathbb{Z}}_{\flat,h}^{k+1}$ *with the seminorm*

$$\|(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h)\|_{\mathbb{Z},h} := \left(\|\underline{\mathbf{w}}_h\|^2_{\mathbf{X},h} + \|\underline{\mathrm{r}}_h\|^2_{\mathrm{Y},h}\right)^{1/2}. \tag{3.37}$$

Let's now consider general boundary conditions. On the basis of the analysis brought on in [FS20], we will adapt the steps of the demonstration to deal with fully Neumann boundary conditions, show stability and provide a similar convergence theorem.

The proposed HHO scheme for the Neumann case reads:
Find $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathrm{Y}}_h^{k+1}$ such that

$$\mathrm{a}_h(\underline{\mathbf{u}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{\mathrm{p}}_h) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega + \sum_{F \in \mathcal{F}_h^b} (\boldsymbol{\Phi} \times \boldsymbol{n}, \boldsymbol{n} \times (\boldsymbol{v}_F \times \boldsymbol{n}))_F \qquad \forall \underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1}, \tag{3.38a}$$

$$-\mathrm{b}_h(\underline{\mathbf{u}}_h, \underline{\mathrm{q}}_h) + \mathrm{c}_h(\underline{\mathrm{p}}_h, \underline{\mathrm{q}}_h) = 0 \qquad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1}. \tag{3.38b}$$

where

$$c_h(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h) := (\mathrm{r}_h, \mathrm{q}_h)_\Omega + \sum_{T \in \mathcal{T}_h} \sum_{F \in \mathcal{F}_T} h_F (\mathrm{r}_F, \mathrm{q}_F)_F,$$

is a slightly modified stabilisation term.

Numerical results will be showed also for mixed boundary conditions. The corresponding discrete problem reads as follows:
Find $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbf{X}}_{h,\Gamma_B,\mathrm{b}}^{k+1} \times \underline{\mathrm{Y}}_{h,\Gamma_B}^{k+1}$ such that:

$$\mathrm{a}_h(\underline{\mathbf{u}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{\mathrm{p}}_h) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega + \sum_{F \in \mathcal{F}_h^b \cap \Gamma_H} (\boldsymbol{\Phi} \times \boldsymbol{n}, \boldsymbol{n} \times (\boldsymbol{v}_F \times \boldsymbol{n}))_F \qquad \forall \underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{h,\mathbf{0},\Gamma_B}^{k+1}, \tag{3.39a}$$

$$-\mathrm{b}_h(\underline{\mathbf{u}}_h, \underline{\mathrm{q}}_h) + \mathrm{c}_h(\underline{\mathrm{p}}_h, \underline{\mathrm{q}}_h) = 0 \qquad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_{h,\Gamma_B}^{k+1}. \tag{3.39b}$$

3.3 Stability analysis

Let's start the analysis of the Neumann problem by getting precise on the notion of norms that will be involved. By setting: $\underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1} := \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathrm{Y}}_h^{k+1}$ Problem (3.38) can be reformulated as follows:
Find $\underline{\mathrm{z}}_h = (\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}$ such that:

$$\mathrm{A}_h\Big(\underline{\mathrm{z}}_h, \mathrm{s}_h\Big) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega + \sum_{F \in \mathcal{F}_h^b} (\mathbf{v}_F \times \boldsymbol{n}, \boldsymbol{n} \times (\boldsymbol{\Phi} \times \boldsymbol{n}))_F \qquad \forall\, \mathrm{s}_h = (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}, \tag{3.40}$$

where the bilinear form $\mathrm{A}_h : \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1} \times \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1} \to \mathbb{R}$ is defined by

$$\mathrm{A}_h\Big((\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h), (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\Big) := \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{\mathrm{r}}_h) - \mathrm{b}_h(\underline{\mathbf{w}}_h, \underline{\mathrm{q}}_h) + \mathrm{c}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h). \tag{3.41}$$

We also give an adapted version of dicrete divergence-free property in light of the stabilisation:

$$\begin{aligned} \mathring{\underline{\mathbb{Z}}}_{\mathrm{b},h}^{k+1} &:= \Big\{(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1} \,:\, -\mathrm{b}_h(\underline{\mathbf{w}}_h, \underline{\mathrm{q}}_h) + \mathrm{c}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{q}}_h) = 0 \quad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1}\Big\} \\ &= \Big\{(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1} \,:\, \mathrm{A}_h\Big((\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h), (\underline{\mathbf{0}}_h, \underline{\mathrm{q}}_h)\Big) = 0 \quad \forall \underline{\mathrm{q}}_h \in \underline{\mathrm{Y}}_h^{k+1}\Big\}. \end{aligned} \tag{3.42}$$

We equip the spaces $\underline{\mathbf{X}}_{\flat,h}^{k+1}$ and $\underline{\mathrm{Y}}_h^{k+1}$ with the seminorms

$$\|\underline{\mathbf{w}}_h\|_{\mathbf{X},\flat,h} := \left(\|\mathbf{curl}_h \mathbf{w}_h\|_\Omega^2 + \mathrm{s}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h)\right)^{1/2}, \tag{3.43a}$$

$$\|\underline{\mathrm{r}}_h\|_{\mathrm{Y},h} := \mathrm{c}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h)^{1/2}. \tag{3.43b}$$

One can easily verify that $\|\cdot\|_{\mathrm{Y},h}$ defines a norm on $\underline{\mathrm{Y}}_h^{k+1}$. We now equip $\underline{\mathbb{Z}}_{\flat,h}^{k+1}$ with the seminorm

$$\|(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h)\|_{\mathbb{Z},\flat,h} := \left(\|\underline{\mathbf{w}}_h\|_{\mathbf{X},\flat,h}^2 + \|\underline{\mathrm{r}}_h\|_{\mathrm{Y},\flat,h}^2\right)^{1/2}. \tag{3.44}$$

Lemma 3.7 (Norm $\|\cdot\|_{\mathbb{Z},\flat,h}$)**.** *The map $\|\cdot\|_{\mathbb{Z},\flat,h}$ defines a norm on $\mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$.*

Proof. The seminorm property being straightforward, we only need to prove that, for all couples $(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$, $\|(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h)\|_{\mathbb{Z},\flat,h} = 0$ implies $(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) = (\underline{\mathbf{0}}_h, \underline{0}_h)$. Let then $(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$ be such that $\|(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h)\|_{\mathbb{Z},\flat,h} = 0$. We infer that $\|\underline{\mathbf{w}}_h\|_{\mathbf{X},\flat,h} = 0$ and $\|\underline{\mathrm{r}}_h\|_{\mathrm{Y},\flat,h} = 0$. Since $\|\cdot\|_{\mathrm{Y},\flat,h}$ is a norm on $\underline{\mathrm{Y}}_h^{k+1}$, we directly get from the second relation that $\underline{\mathrm{r}}_h = \underline{0}_h$. Now, owing to the definitions (3.42) of $\mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$, (3.31b) of b_h, and to the fact that $\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$ is defined as in (3.8), we infer from $(\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$ and $\underline{\mathrm{r}}_h = \underline{0}_h$ that $\underline{\mathbf{w}}_h \in \mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$. By Corollary (3.5), $\|\cdot\|_{\mathbf{X},\flat,h}$ defines a norm on $\mathring{\underline{\mathbf{X}}}_{\flat,h}^{k+1}$, hence $\underline{\mathbf{w}}_h = \underline{\mathbf{0}}_h$, which concludes the proof. □

We now state some preliminary results for the stability analysis of Problem (3.30).

Lemma 3.8 (Equivalences of seminorms)**.** *The following holds true:*

$$\|\underline{\mathbf{w}}_h\|_{\mathbf{X},\flat,h}^2 \lesssim \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h) \lesssim \|\underline{\mathbf{w}}_h\|_{\mathbf{X},\flat,h}^2 \qquad \forall \underline{\mathbf{w}}_h \in \underline{\mathbf{X}}_{\flat,h}^{k+1} \tag{3.45}$$

Proof. Let $\underline{\mathbf{w}}_h \in \underline{\mathbf{X}}_{\flat,h}^{k+1}$, and $T \in \mathcal{T}_h$. By the definition (3.2) of $\boldsymbol{C}_T^k$, testing with $\boldsymbol{w} = \mathbf{curl}\,\mathbf{w}_T \in \boldsymbol{\mathcal{R}}^k(T)$, integrating by parts, and using the fact that $\boldsymbol{\gamma}_{\tau,F}\big(\mathbf{curl}\,\mathbf{w}_T \times \boldsymbol{n}_{TF}\big) \in \boldsymbol{\mathcal{P}}^k(F) \subset \boldsymbol{\mathcal{P}}_\flat^{k+1}(F)$, we infer

$$\|\mathbf{curl}\,\mathbf{w}_T\|_T^2 = \left(\boldsymbol{C}_T^k \underline{\mathbf{w}}_T, \mathbf{curl}\,\mathbf{w}_T\right)_T + \sum_{F\in\mathcal{F}_T} \left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}}_\flat,F}^{k+1}\big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{w}_T) - \mathbf{w}_F\big), \boldsymbol{\gamma}_{\tau,F}(\mathbf{curl}\,\mathbf{w}_T \times \boldsymbol{n}_{TF})\right)_F.$$

By the Cauchy–Schwarz inequality, a discrete trace inequality (see, e.g., [DD20, Lemma 1.32]), and recalling the definition (3.31a) of a_h, we get $\|\mathbf{curl}_h \mathbf{w}_h\|_\Omega^2 \lesssim \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h)$, and the first inequality in (3.45) follows by adding $\mathrm{s}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h)$ to both sides. To prove the second inequality, we test (3.2) with $\boldsymbol{w} = \boldsymbol{C}_T^k \underline{\mathbf{w}}_T \in \boldsymbol{\mathcal{R}}^k(T)$ to infer

$$\|\boldsymbol{C}_T^k \underline{\mathbf{w}}_T\|_T^2 = \left(\mathbf{curl}\,\mathbf{w}_T, \boldsymbol{C}_T^k \underline{\mathbf{w}}_T\right)_T - \sum_{F\in\mathcal{F}_T} \left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}}_\flat,F}^{k+1}\big(\boldsymbol{\gamma}_{\tau,F}(\mathbf{w}_T) - \mathbf{w}_F\big), \boldsymbol{\gamma}_{\tau,F}(\boldsymbol{C}_T^k \underline{\mathbf{w}}_T \times \boldsymbol{n}_{TF})\right)_F,$$

and we conclude by the same arguments. □

We are now in position to show well-posedness for Problem (3.30).

Lemma 3.9 (Well-posedness). *For all* $\underline{z}_h \in \underline{\mathbb{Z}}_{\flat,h}^{k+1}$*:*

$$\mathrm{A}_h\Big(\underline{z}_h, \underline{z}_h\Big) \gtrsim \|\underline{z}_h\|_{\mathbb{Z},\flat,h}^2. \tag{3.46}$$

Hence, Problem (3.30) *is well-posed, and the following a priori bound holds true:*

$$\|(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)\|_{\mathbb{Z},\flat,h} \lesssim \{\|\boldsymbol{f}\|_\Omega^2 + \|\boldsymbol{\Phi}\|_{\partial\Omega}^2\}^{\frac{1}{2}}. \tag{3.47}$$

Proof. Let $\underline{z}_h = (\underline{\mathbf{w}}_h, \underline{\mathrm{r}}_h) \in \underline{\mathbb{Z}}_{\flat,h}^{k+1}$. One has

$$\|\underline{z}_h\|_{\mathbb{Z},\flat,h}^2 \lesssim \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{w}}_h) + \mathrm{d}_h(\underline{\mathrm{r}}_h, \underline{\mathrm{r}}_h) = \mathrm{A}_h\Big(\underline{z}_h, \underline{z}_h\Big). \tag{3.48}$$

To prove well-posedness, since the linear system associated to Problem (3.30) is square, it is sufficient to prove injectivity. Assume that $\mathrm{A}_h\Big((\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h), (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\Big) = 0$ for all $(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) \in \underline{\mathbb{Z}}_{\flat,h}^{k+1}$. Taking $(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) = (\underline{\mathbf{0}}_h, \underline{\mathrm{q}}_h)$ and using(3.42), we first infer that $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \mathring{\underline{\mathbb{Z}}}_{\flat,h}^{k+1}$. Taking $(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) = (\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)$ and using (3.46), we then get

$$\|(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)\|_{\mathbb{Z},\flat,h}^2 \lesssim \mathrm{A}_h\Big((\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h), (\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)\Big) = 0,$$

which, by Lemma(3.7), eventually yields $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) = (\underline{\mathbf{0}}_h, \underline{0}_h)$. To prove the a priori bound (3.47), we take $\underline{z}_h = (\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)$ in (**??**) and we use (4.1). We get, by the Cauchy–Schwarz inequality,

$$\begin{aligned}
\|(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)\|_{\mathbb{Z},\flat,h}^2 &\lesssim (\boldsymbol{f}, \mathbf{u}_h)_\Omega + \sum_{F\in\mathcal{F}_h^b} (\boldsymbol{\Phi}\times\boldsymbol{n}, \boldsymbol{n}\times(\boldsymbol{u}_F\times\boldsymbol{n}))_F \\
&\le \|\boldsymbol{f}\|_\Omega\|\mathbf{u}_h\|_\Omega + \|\boldsymbol{\Phi}\|_{\partial\Omega}\{\sum_{F\in\mathcal{F}_H^b}\|\boldsymbol{u}_F\|_F^2\}^{\frac{1}{2}} \\
&\le \{\|\boldsymbol{f}\|_\Omega^2 + \|\boldsymbol{\Phi}\|_{\partial\Omega}^2\}^{\frac{1}{2}}\{\|\boldsymbol{u}\|_\Omega^2 + \sum_{F\in\mathcal{F}_h^b}\|\boldsymbol{u}_F\|_F^2\}^{\frac{1}{2}}.
\end{aligned}$$

The conclusion follows from the combination of a generalized discrete Weber inequality (3.20) of Corollary (3.5) applied to $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h)$ satisfying (3.19) (one can easily check that d_h satisfies (3.18), thanks to an application of Poincaré-Wirtinger inequality, that is Poincaré inequality on $H^1_{zmv}(\Omega)$ to bound $\|\mathbf{u}_h\|_\Omega$. □

3.4 Error analysis

We recall that $(\boldsymbol{u}, p) \in \boldsymbol{H}(\mathbf{curl};\Omega)\times H^1(\Omega)$ denotes the unique solution to Problem (3.25). We assume from now on that $\boldsymbol{u}$ possesses the additional regularity $\boldsymbol{u} \in \boldsymbol{H}^1(\Omega;\mathbb{R}^3)$, and we let $\widehat{\underline{\mathbf{u}}}_h := \underline{\mathbf{I}}_{\mathbf{X},\flat,h}^{k+1}\boldsymbol{u} \in \underline{\mathbf{X}}_{\flat,h}^{k+1}$ and $\widehat{\underline{\mathrm{p}}}_h := \underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1}p \in \underline{\mathrm{Y}}_h^{k+1}$. We define the errors

$$\underline{\mathbf{X}}_{\flat,h}^{k+1} \ni \underline{\mathbf{e}}_h := \underline{\mathbf{u}}_h - \widehat{\underline{\mathbf{u}}}_h, \qquad \underline{\mathrm{Y}}_h^{k+1} \ni \underline{\epsilon}_h := \underline{\mathrm{p}}_h - \widehat{\underline{\mathrm{p}}}_h, \tag{3.49}$$

where $(\underline{\mathbf{u}}_h, \underline{\mathrm{p}}_h) \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1} \times \underline{\mathrm{Y}}_h^{k+1}$ is the unique solution to Problem (3.30). Recalling (4.1) and (4.2), the errors $(\underline{\mathbf{e}}_h, \underline{\epsilon}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}$ solve

$$\mathrm{A}_h\Big((\underline{\mathbf{e}}_h, \underline{\epsilon}_h), (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\Big) = \mathrm{l}_h(\underline{\mathbf{v}}_h) + \mathrm{m}_h\Big(\underline{\mathrm{q}}_h\Big) \qquad \forall\, (\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}, \tag{3.50}$$

where we have defined the consistency error linear forms

$$\mathrm{l}_h(\underline{\mathbf{v}}_h) := (\boldsymbol{f}, \mathbf{v}_h)_\Omega + \sum_{F \in \mathcal{F}_h^b} (\boldsymbol{\Phi} \times \boldsymbol{n}, \boldsymbol{n} \times (\boldsymbol{v}_F \times \boldsymbol{n}))_F - \mathrm{a}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h) - \mathrm{b}_h(\underline{\mathbf{v}}_h, \widehat{\underline{\mathrm{p}}}_h), \tag{3.51a}$$

$$\mathrm{m}_h\Big(\underline{\mathrm{q}}_h\Big) := \mathrm{b}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathrm{q}}_h) - \mathrm{c}_h(\widehat{\underline{\mathrm{p}}}_h, \underline{\mathrm{q}}_h). \tag{3.51b}$$

Theorem 3.10 (Energy-error estimate)**.** *Assume that*

$$\boldsymbol{u} \in \boldsymbol{H}(\mathbf{curl};\Omega) \cap \boldsymbol{H}^1(\Omega;\mathbb{R}^3) \cap \boldsymbol{H}^{k+2}(\mathcal{T}_h;\mathbb{R}^3), \qquad p \in H^1(\Omega) \cap H^{k+1}(\mathcal{T}_h).$$

Then, the following holds true, with $(\underline{\mathbf{e}}_h, \underline{\epsilon}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}$ *defined by* (3.49)*:*

$$\|(\underline{\mathbf{e}}_h, \underline{\epsilon}_h)\|_{\mathbb{Z},\mathrm{b},h} \lesssim \left[\sum_{T \in \mathcal{T}_h} h_T^{2(k+1)} \left(|\boldsymbol{u}|^2_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)} + |p|^2_{H^{k+1}(T)}\right)\right]^{1/2}. \tag{3.52}$$

Proof. Since $(\underline{\mathbf{e}}_h, \underline{\epsilon}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}$, by (3.46) with $\underline{z}_h = (\underline{\mathbf{e}}_h, \underline{\epsilon}_h)$ and (3.50), we infer

$$\|(\underline{\mathbf{e}}_h, \underline{\epsilon}_h)\|_{\mathbb{Z},\mathrm{b},h} \lesssim \max_{(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h) \in \underline{\mathbb{Z}}_{\mathrm{b},h}^{k+1}, \|(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\|_{\mathbb{Z},\mathrm{b},h}=1} \Big(\mathrm{l}_h(\underline{\mathbf{v}}_h) + \mathrm{m}_h\Big(\underline{\mathrm{q}}_h\Big)\Big). \tag{3.53}$$

Let us first focus on $\mathrm{l}_h(\underline{\mathbf{v}}_h)$ for $\underline{\mathbf{v}}_h \in \underline{\mathbf{X}}_{\mathrm{b},h}^{k+1}$. By (3.51a), the fact that $\boldsymbol{f} = \mathbf{curl}(\mathbf{curl}\,\boldsymbol{u}) + \mathbf{grad}\,p$ in Ω, and element-by-element integration by parts, we infer

$$\begin{aligned}
\mathrm{l}_h(\underline{\mathbf{v}}_h) &= \sum_{T \in \mathcal{T}_h} \left[\Big(\mathbf{curl}\,\boldsymbol{u}, \mathbf{curl}\,\mathbf{v}_T\Big)_T - \sum_{F \in \mathcal{F}_T} \Big(\mathbf{curl}\,\boldsymbol{u}_{|F} \times \boldsymbol{n}_{TF}, \mathbf{v}_{T|F}\Big)_F\right] \\
&\quad + \Big(\mathbf{grad}\,p, \mathbf{v}_h\Big)_\Omega - \mathrm{a}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h) - \mathrm{b}_h(\underline{\mathbf{v}}_h, \widehat{\underline{\mathrm{p}}}_h) \\
&= \sum_{T \in \mathcal{T}_h} \Bigg(\Big(\mathbf{curl}\,\boldsymbol{u}, \mathbf{curl}\,\mathbf{v}_T\Big)_T \\
&\quad - \sum_{F \in \mathcal{F}_T} \Big(\gamma_{\tau,F}\Big(\mathbf{curl}\,\boldsymbol{u} \times \boldsymbol{n}_{TF}\Big), \gamma_{\tau,F}(\mathbf{v}_T) - \mathbf{v}_F\Big)_F\Bigg) - \sum_{F \in \mathcal{F}_h^b} (\mathbf{curl}\,\boldsymbol{\Phi} \times \boldsymbol{n}_{TF}, \boldsymbol{v}_T|_F)_F - \mathrm{a}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h),
\end{aligned} \tag{3.54}$$

where we have used the fact that the tangential component of $\mathbf{curl}\,\boldsymbol{u}$ is continuous across interfaces (as a consequence of the fact that $\mathbf{curl}\,\boldsymbol{u} \in \boldsymbol{H}(\mathbf{curl};\Omega) \cap \boldsymbol{H}^1(\mathcal{T}_h;\mathbb{R}^3)$) along with $\mathbf{curl}\,\boldsymbol{u} \times \boldsymbol{n}_{TF} = \mathbf{curl}\,\boldsymbol{\Phi} \times \boldsymbol{n}_{TF}$ for all $F \in \mathcal{F}_h^{\mathrm{b}}$ to insert $\mathbf{v}_F$ into the boundary

term, together with the fact that $\left(\mathbf{grad}\, p, \mathbf{v}_h\right)_\Omega = \mathrm{b}_h(\underline{\mathbf{v}}_h, \widehat{\underline{\mathrm{p}}}_h)$ as a consequence of the commutation property (3.5a). Using the definitions (3.31a) of a_h and (3.5b) of $\boldsymbol{C}_T^k$ for $T \in \mathcal{T}_h$ (testing with $\boldsymbol{w} = \boldsymbol{C}_T^k \widehat{\underline{\mathbf{u}}}_T \in \boldsymbol{\mathcal{R}}^k(T)$), and integrating by parts, we have

$$\mathrm{a}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h) = \sum_{T\in\mathcal{T}_h}\Bigg[\left(\boldsymbol{C}_T^k\widehat{\underline{\mathbf{u}}}_T, \mathbf{curl}\,\mathbf{v}_T\right)_T - \sum_{F\in\mathcal{F}_T}\left(\boldsymbol{\gamma}_{\tau,F}(\boldsymbol{C}_T^k\widehat{\underline{\mathbf{u}}}_T\times\boldsymbol{n}_{TF}), \boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T) - \mathbf{v}_F\right)_F\Bigg] - \sum_{F\in\mathcal{F}_h^b}(\pi_{\boldsymbol{\mathcal{R}},T}^k\,\mathbf{curl}\,\boldsymbol{\Phi}, \boldsymbol{\gamma}_{\tau,F}(\boldsymbol{v}_T))_F + \mathrm{s}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h). \tag{3.55}$$

Since, by Lemma (3.5b), $\boldsymbol{C}_T^k\widehat{\underline{\mathbf{u}}}_T = \boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k\left(\mathbf{curl}\,\boldsymbol{u}_{|T}\right)$ for all $T \in \mathcal{T}_h$, a combination of (3.54) and (3.55) yields (recall that $\mathbf{v}_T \in \boldsymbol{\mathcal{P}}^{k+1}(T)$)

$$\mathrm{l}_h(\underline{\mathbf{v}}_h) = \sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}\left(\boldsymbol{\gamma}_{\tau,F}\Big((\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u})\times\boldsymbol{n}_{TF}\Big), \boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T) - \mathbf{v}_F\right)_F - \sum_{F\in\mathcal{F}_h^b}(\mathbf{curl}\,\boldsymbol{\Phi} - \pi_{\boldsymbol{\mathcal{R}},T}^k\,\mathbf{curl}\,\boldsymbol{\Phi}, \boldsymbol{\gamma}_{\tau,F}\boldsymbol{v}_t|_F)_F - \mathrm{s}_h(\widehat{\underline{\mathbf{u}}}_h, \underline{\mathbf{v}}_h).$$

Applying the triangle and Cauchy–Schwarz inequalities, we get

$$|\mathrm{l}_h(\underline{\mathbf{v}}_h)| \le \sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}\|\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u}\|_F\,\|\boldsymbol{\gamma}_{\tau,F}(\mathbf{v}_T) - \mathbf{v}_F\|_F + \|\mathbf{curl}\,\boldsymbol{\Phi} - \pi_{\boldsymbol{\mathcal{R}},T}^k\,\mathbf{curl}\,\boldsymbol{\Phi}\|_{\partial\Omega}\Big\{\sum_{F\in\mathcal{F}_h^b}\|\boldsymbol{\gamma}_{\tau,F}(\boldsymbol{v}_T)\|_F\Big\}^{\frac12} + \mathrm{s}_h(\widehat{\underline{\mathbf{u}}}_h, \widehat{\underline{\mathbf{u}}}_h)^{1/2}\mathrm{s}_h(\underline{\mathbf{v}}_h, \underline{\mathbf{v}}_h)^{1/2}. \tag{3.56}$$

Let us focus on $\|\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u}\|_F$ for $F \in \mathcal{F}_T$. Adding/subtracting $\mathbf{curl}\left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}(\boldsymbol{u}_{|T})\right)$, using a triangle inequality, a discrete trace inequality (see, e.g., [DD20, Lemma 1.32]) on $\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}(\boldsymbol{u}_{|T})\right)$, and the approximation properties of $\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}$ on mesh faces (see, e.g., [DD20, Theorem 1.45]) for $\mathbf{curl}\left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}(\boldsymbol{u}_{|T})\right) - \mathbf{curl}\,\boldsymbol{u}$, we infer

$$h_F^{1/2}\|\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u}\|_F \lesssim \|\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u}\|_T + \|\mathbf{curl}\,\boldsymbol{u} - \mathbf{curl}\left(\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}(\boldsymbol{u}_{|T})\right)\|_T + h_T^{k+1}|\boldsymbol{u}|_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)},$$

where we have used yet another triangle inequality to insert $\mathbf{curl}\,\boldsymbol{u}$. The second term on the right-hand side is readily estimated using again the approximation properties of $\boldsymbol{\pi}_{\boldsymbol{\mathcal{P}},T}^{k+1}$. As far as the first term is concerned, we have

$$\|\boldsymbol{\pi}_{\boldsymbol{\mathcal{R}},T}^k(\mathbf{curl}\,\boldsymbol{u}_{|T}) - \mathbf{curl}\,\boldsymbol{u}\|_T = \min_{\boldsymbol{p}\in\boldsymbol{\mathcal{P}}^{k+1}(T)}\|\mathbf{curl}\,\boldsymbol{p} - \mathbf{curl}\,\boldsymbol{u}\|_T,$$

which finally yields

$$h_F^{1/2}\|\boldsymbol{\pi}^k_{\boldsymbol{\mathcal{R}},T}(\mathbf{curl}\,\boldsymbol{u}_{|T})-\mathbf{curl}\,\boldsymbol{u}\|_F\lesssim h_T^{k+1}|\boldsymbol{u}|_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)}.$$

Following the same steps for $\boldsymbol{\Phi}$ one concludes:

$$h_F^{1/2}\|\boldsymbol{\pi}^k_{\boldsymbol{\mathcal{R}},T}(\mathbf{curl}\,\boldsymbol{\Phi}_{|T})-\mathbf{curl}\,\boldsymbol{\Phi}\|_F\lesssim h_T^{k+1}|\boldsymbol{\Phi}|_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)}.$$

As for $\mathrm{s}_h(\widehat{\underline{\mathbf{u}}}_h,\widehat{\underline{\mathbf{u}}}_h)$ we have:

$$\begin{aligned}\mathrm{s}_h(\widehat{\underline{\mathbf{u}}}_h,\widehat{\underline{\mathbf{u}}}_h)&=\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{G}},F}\Big(\boldsymbol{\gamma}_{\tau,F}(\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}(\boldsymbol{u}_{|T})-\boldsymbol{u})\Big)\|_F^2\\&\le\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}(\boldsymbol{u}_{|T})-\boldsymbol{u}\|_F^2\lesssim\sum_{T\in\mathcal{T}_h}h_T^{2(k+1)}|\boldsymbol{u}|^2_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)},\end{aligned}\tag{3.57}$$

Plugging these last estimates into (3.56), applying a discrete Cauchy–Schwarz inequality, and using (3.16) as well as(3.57), we infer

$$|\mathrm{l}_h(\underline{\mathbf{v}}_h)|\lesssim\left(\sum_{T\in\mathcal{T}_h}h_T^{2(k+1)}(|\boldsymbol{u}|^2_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)}+|\boldsymbol{\Phi}|_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)})\right)^{1/2}\|\underline{\mathbf{v}}_h\|_{\mathbf{X},\mathrm{b},h}.\tag{3.58}$$

Let us now focus on $\mathrm{m}_h\big(\underline{\mathrm{q}}_h\big)$ for $\underline{\mathrm{q}}_h\in\underline{\mathrm{Y}}_h^{k+1}$. Since $\widehat{\mathrm{p}}_h=p=0$, $m_h(\underline{\mathrm{q}}_h)=b_h(\widehat{\underline{\mathbf{u}}}_h,\underline{\mathrm{q}}_h)$. Starting from(3.51b), performing an element-by-element integration by parts in (3.1), and using that $\mathbf{grad}\,\mathrm{q}_T\in\boldsymbol{\mathcal{G}}^{k-1}(T)\subset\boldsymbol{\mathcal{P}}^{k+1}(T)$, we infer

$$\begin{aligned}m_h(\underline{\mathrm{q}}_h)&=\sum_{T\in\mathcal{T}_h}\left((\mathbf{grad}\,\mathrm{q}_T,\boldsymbol{u})_T+\sum_{F\in\mathcal{F}_T}\Big(\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}(\boldsymbol{u}_{|T})_{|F}\cdot\boldsymbol{n}_{TF},\mathrm{q}_F-\mathrm{q}_{T|F}\Big)_F\right)\\&=\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}\Big((\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}(\boldsymbol{u}_{|T})-\boldsymbol{u})_{|F}\cdot\boldsymbol{n}_{TF},\mathrm{q}_F-\mathrm{q}_{T|F}\Big)_F,\end{aligned}$$

where the last identity follows from another element-by-element integration by parts, and from the fact that $\mathbf{div}\,\boldsymbol{u}=0$ in Ω, and that $\boldsymbol{u}\in\boldsymbol{H}^1(\Omega;\mathbb{R}^3)$ along with $\boldsymbol{u}\cdot\boldsymbol{n}=0$ on $\partial\Omega$. By the triangle and Cauchy–Schwarz inequalities, one then gets

$$\begin{aligned}|\mathrm{m}_h\big(\underline{\mathrm{q}}_h\big)|\le&\left(\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F^{-1}\|\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}(\boldsymbol{u}_{|T})-\boldsymbol{u}\|_F^2\right)^{1/2}\\&\times\left(\sum_{T\in\mathcal{T}_h}\sum_{F\in\mathcal{F}_T}h_F\|\mathrm{q}_F-\mathrm{q}_{T|F}\|_F^2\right)^{1/2}.\end{aligned}\tag{3.59}$$

Using, for all $T\in\mathcal{T}_h$, the approximation properties of $\boldsymbol{\pi}^{k+1}_{\boldsymbol{\mathcal{P}},T}$ on the faces of T for the first factor on the right-hand side, and the triangle inequality along with a discrete trace

inequality (see, e.g., [DD20, Lemma 1.32]) for the second factor, we infer

$$|\mathrm{m}_h(\underline{\mathrm{q}}_h)| \lesssim \left(\sum_{T\in\mathcal{T}_h} h_T^{2(k+1)} |\boldsymbol{u}|^2_{\boldsymbol{H}^{k+2}(T;\mathbb{R}^3)} \right)^{1/2} \|\underline{\mathrm{q}}_h\|_{\mathrm{Y},h}. \tag{3.60}$$

Plugging (3.58) and (3.60) into (3.53) for $(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)$ such that $\|(\underline{\mathbf{v}}_h, \underline{\mathrm{q}}_h)\|_{\mathrm{Z},\mathfrak{b},h} = 1$ finally yields (3.52).

□

3.5 Notes on local reconstruction

The characteristic feature of HHO schemes is local reconstruction. With this technique a new polynomial on a cell is determined starting from polynomial defined on the same cell plus all the polynomials local to the neighbouring faces. The idea at the base of local reconstruction is that it is possible to obtain higher order polynomial projection of a function on a cell by exploiting lower order polynomial projections on the faces and on the cell.
In the version of the method that has been exposed reconstruction is implemented for the operators **curl** and **grad**. An alternative scheme may be designed considering a reconstruction for the potential.
To give an introductory example, we'll show the reconstruction of a scalar potential, that may be used to discretize the bilinear form b_h of (3.31b). Consider a cell T and the set $\mathcal{F}_T$ of its neighbouring faces. We recall the definition of L_2-orthogonal projection:

Definition 7. *Let $g : T \to \mathbb{R}$ be in $L^1(T)$. The L_2 projection $\pi_0^k g \in \mathcal{P}^K(T)$ of g is the k-degree polynomial satisfying:*

$$(\pi_0^k g, w)_T = (g, w) \qquad \forall w \in \mathcal{P}^k(T). \tag{3.61}$$

We also consider the different definition of projection:

Definition 8. *Let $g : T \to \mathbb{R}$ be in $H^1(T)$. The elliptic projection $\pi_1^k g \in \mathcal{P}^K(T)$ of g is the k-degree polynomial satisfying:*

$$(\mathbf{grad}\, \pi_1^k g, \mathbf{grad}\, w)_T = (\mathbf{grad}\, g, \mathbf{grad}\, w) \qquad \forall w \in \mathcal{P}^k(T) \tag{3.62a}$$

$$(\pi_1^k g - g, 1) = 0 \tag{3.62b}$$

The condition (3.62b) is necessary to provide a consistent definition, as the first condition (3.62b) only prescribes properties of its gradient, and two scalar fields with the same gradient are equal up to a constant. The approximation properties of these projectors are expressed by the following theorem (see also [DD20, Theor. 1.45, 1.48]):

Theorem 3.11. *Let $(\mathcal{M})_{h\in\mathcal{H}}$ be a mesh sequence. Let a polynomial degree $l \geq 0$ and an integer $s \in \{1, \dots, l+1\}$. Then, for any X element or face of $\mathcal{M}_h$, all $v \in H^s(X)$, and all $m \in 0, \dots, s$,*

$$|v - \pi_X^{0,l} v|_{H^m(X)} \lesssim h_T^{s-m} |v|_{H^s(X)} \tag{3.63}$$

And for any $T \in \mathcal{T}_h$,

$$|v - \pi_T^{1,l} v|_{H^m(T)} \lesssim h_T^{s-m} |v|_{H^s(X)} \tag{3.64}$$

With these definitions at hand, let's consider the following integration by parts formula, valid for any $g \in L^2$ and $w \in H^2(T)$:

$$(\mathbf{grad}\, g, \mathbf{grad}\, w)_T = -(g, \Delta w)_T + \sum_{F \in \mathcal{F}_T} (u, \mathbf{grad}\, w \cdot \boldsymbol{n})_F \tag{3.65}$$

If we restrict this property to any $w \in \mathcal{P}^{k+1}(T)$ we can easily recognize on the left a quantity that remains the same if g is changed with its elliptic projection of degree $k+1$, whereas on the right g can be substituted with the L^2 projection of degree k whenever it appears:

$$(\mathbf{grad}\, \pi_1^{k+1} g, \mathbf{grad}\, w)_T = -(\pi_0^k g, \Delta w)_T + \sum_{F \in \mathcal{F}_T} (\pi_0^k g, \mathbf{grad}\, w \cdot \boldsymbol{n})_F. \tag{3.66}$$

If we add the second line in definition (3.61) to (3.66), we deduce the following local well-posed problem to retrieve the elliptic projection:
Find $\pi_1^{k+1} g \in \mathcal{P}_T^{k+1}$ such that:

$$\begin{aligned} (\mathbf{grad}\, \pi_1^{k+1} g, \mathbf{grad}\, w)_T &= -(\pi_0^k g, \Delta w)_T + \sum_{F \in \mathcal{F}_T} (\pi_0^k g, \mathbf{grad}\, w \cdot \boldsymbol{n})_F \forall w \in \mathcal{P}^{k+1} \\ (\pi_1^k g - g, 1) &= 0 \end{aligned} \tag{3.67}$$

We consider a reconstruction operator $p_T^{k+1} : \mathcal{P}^k(T) \bigtimes \{ \bigtimes_{F \in \mathcal{F}_T} \mathcal{P}^k(F) \} \to \mathcal{P}^{k+1}(T)$ that maps a k-degree projection of g into a $k+1$-degree projection. We can define it such that $p_T^{k+1} \underline{\mathbf{u}}_T$ is the solution to problem (3.67). We can also collect the L_2-orthogonal projections of a a function on a cell and its neighbouring faces defining an interpolation operator:

$$\underline{\mathbf{I}}_T^k : L_2(T) \to \underline{\mathbf{X}}_T : \underline{\mathbf{I}}_T^k v = (\pi_T^{0,k} v, (\pi_F^{0,k})_{\mathcal{F}_T}) \tag{3.68}$$

By doing so, we can state a commutation property:

$$p_T^{k+1} \underline{\mathbf{I}}_T^k v = \pi_T^{1,k+1} v \tag{3.69}$$

Once introduced suitable problem-dependent local reconstructions the HHO method discretizes bilinear forms appearing in the problem by discrete bilinear forms acting on some reconstruction obtained from the dicrete unkowns rather than directly on the discrete unknowns. In this example, bilinear form b_h may be discretized like this:

$$b(v, q) = (\boldsymbol{v}, \mathbf{grad}\, q)_\Omega \rightsquigarrow b_h(\underline{\boldsymbol{v}}_h, \mathbf{grad}\, \pi_{1,h}^{k+1} \underline{q}_h) = \sum_{T \in \mathcal{T}_h} (\underline{\boldsymbol{v}}_T, \mathbf{grad}\, \pi_{1,h}^{k+1} \underline{q}_T)_T$$

Symmetrically, it would be interesting to obtain a similar reconstruction for the **curl** potential. We are interested in a **curl**-version $\pi_T^{c,k+1}$ of the elliptic projector , together with a reconstruction operator g_T^{k+1} that, composed with an interpolator on a suitable

hybrid local space can make a statement like the following true for a function $v \in \boldsymbol{H}(\mathbf{curl}, \Omega)$:

$$g_T^{k+1} \underline{\mathbf{I}}_T^k \boldsymbol{v} = \pi_T^{c,k+1} \boldsymbol{v} \tag{3.70}$$

The delicate point here resides in the way the **curl**-elliptic projection is defined. In the case of the gradient a suitable polynomial is designated, such that its gradient coincides with the gradient of the function to be projected. Here requiring the same is simple, but closing the system to provide a consistent definition requires more than a scalar condition. This reflects the continuous version of the problem. Functions that have the same gradient differ by a constant. Fixing this constant is sufficient to uniquely define a potential. On the other hand, functions that share the same curl are identical up to the gradient of some scalar field. Then, in the continuous case, this would amount to an infinitely dimensional condition to fix the potential. The way this is done is called *gauge fixing*.
Working with polynomial spaces, it is however possible to reduce to a finite number the conditions to reconstruct the potential. For this purpose we can consider the polynomial decomposition introduced in lemma (3.2). Any three-valued k-degreed polynomial in $\boldsymbol{\mathcal{P}}^k(T)$ has its gradient part in the finite dimensional space $\boldsymbol{\mathcal{G}}^k(T)$. Then, if we define a projector by imposing a condition on the curl, since $\mathbf{curl}(\mathbf{grad}) = 0$, the gradient part is filtered out. To retrieve it we should impose a number of conditions equal to the dimension of $\boldsymbol{\mathcal{G}}^k(T)$. These considerations lead to the following definition of the **curl**-elliptic projection:

Definition 9. *Let $\boldsymbol{g} : T \to \mathbb{R}^3$ be in $\boldsymbol{H}(\mathbf{curl}, T)$. The* **curl***-elliptic projection $\pi_c^k \boldsymbol{g} \in \mathcal{P}^K(T)$ of $\boldsymbol{g}$ is the k-degree polynomial satisfying:*

$$\begin{aligned} (\mathbf{curl}\, \pi_c^k \boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w})_T &= (\mathbf{curl}\, \boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w}) \qquad \forall \boldsymbol{w} \in \boldsymbol{\mathcal{P}}^k(T) \\ (\pi_c^k \boldsymbol{g} - \boldsymbol{g}, \boldsymbol{z}) &= 0 \qquad \forall \boldsymbol{z} \in \boldsymbol{\mathcal{G}}^k(T) \end{aligned} \tag{3.71}$$

Similarly to what we have done for the **grad** case we start from a Green formula, where we notice that $\boldsymbol{g}$ can be substituted by its projections in the scalar products. Consider $\boldsymbol{w} \in \boldsymbol{\mathcal{P}}^{k+1}(T)$

$$\begin{aligned} (\mathbf{curl}\, \boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w})_T &= (\boldsymbol{g}, \mathbf{curl}\, \mathbf{curl}\, \boldsymbol{w})_T + \sum_{F \in \mathcal{F}_T} (\boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w} \times \boldsymbol{n})_F \\ (\mathbf{curl}\, \pi_c^{k+1} \boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w})_T &= (\boldsymbol{\pi}_0^k \boldsymbol{g}, \mathbf{curl}\, \mathbf{curl}\, \boldsymbol{w})_T + \sum_{F \in \mathcal{F}_T} (\boldsymbol{\pi}_0^k \boldsymbol{g}, \mathbf{curl}\, \boldsymbol{w} \times \boldsymbol{n})_F. \end{aligned}$$

Finally, the curl potential reconstructor can be defined:

Definition 10. *Consider a local hybrid space of collections of polynomials on a cell T and its faces: $\underline{X}^k(T) = \boldsymbol{\mathcal{P}}^k(T) \times \{ \bigtimes_{F \in \mathcal{F}_T} \boldsymbol{\mathcal{P}}^k(F) \}$. Let $p_c^{k+1} : \underline{X}^k(T) \to \boldsymbol{\mathcal{P}}^{k+1}(T)$ be the* ***curl potential***

reconstruction *operator such that, for an element* $(\boldsymbol{g}_T, (\boldsymbol{g}_F)_{\mathcal{F}_T}) = \underline{\boldsymbol{g}}_T \in \underline{X}^k(T)$, $p_c^{k+1}\underline{\boldsymbol{g}}_T$ *solves:*

$$
\begin{aligned}
(\mathbf{curl}\,\pi_c^{k+1}\underline{\boldsymbol{g}}, \mathbf{curl}\,\boldsymbol{w})_T &= (\boldsymbol{g}_T, \mathbf{curl}\,\mathbf{curl}\,\boldsymbol{w})_T + \sum_{F\in\mathcal{F}_T}(\boldsymbol{g}_F, \mathbf{curl}\,\boldsymbol{w}\times\boldsymbol{n})_F \qquad \forall \boldsymbol{w}\in\boldsymbol{\mathcal{P}}^{k+1}(T)\\
(\pi_c^{k+1}\underline{\boldsymbol{g}}_T - \boldsymbol{g}_T, \boldsymbol{z})_T &= 0 \qquad \forall \boldsymbol{z}\in\boldsymbol{\mathcal{G}}^k(T).
\end{aligned}
$$

For the implementation of the method for the problem of magnetostatics the operators **curl** and **grad** are reconstructed rather than a potential, but in a similar fashion. The operators introduced in (3.1) and (3.2) indeed join the following commutation properties:

$$
\begin{aligned}
\boldsymbol{C}_T^k \underline{\mathbf{I}}_{\mathbf{X},h}^{k+1}\boldsymbol{v} &= \pi_{\boldsymbol{C}_h}^k(\mathbf{curl}\,\boldsymbol{v})\\
\boldsymbol{G}_T^{k+1}\underline{\mathrm{I}}_{\mathrm{Y},h}^{k+1} q &= \pi_h^k(\mathbf{grad}\,q)
\end{aligned}
\tag{3.72}
$$

From an implementation point of view, whenever some kind of reconstruction is introduced, what is done is building the so called *local reconstruction matrix*. Consider for example the curl reconstruction defined in (3.2). We can take a basis $\{\boldsymbol{\psi}_i\}_i$ of the curl reconstruction space $\mathcal{R}^k(T)$. Given a generic $\underline{\mathbf{w}_T} \in \underline{\mathbf{X}}_T^{k+1}$ its local reconstruction $\boldsymbol{C}_T^k\mathbf{w}_T$ can be expressed through a finite collection of coefficients using the basis $\{\boldsymbol{\psi}_i\}_i$ as $\boldsymbol{C}_T^k\underline{\mathbf{w}_T} = \sum_i U_i\boldsymbol{\psi}_i$. We can store these coefficient in a column vector $\underline{U}$. With an analogous reasoning we can also consider a basis $\{\boldsymbol{\phi}_m^T\}_m$ for $\boldsymbol{\mathcal{P}}^k(T)$ and a basis $\{\boldsymbol{\phi}_n^{F_k}\}_n$ for $\mathcal{P}_{\mathrm{b}}^k(F_k)$ of any $F_k \in \mathcal{F}_T$. $\underline{\mathbf{w}_T}$ is the represented as a vector of coefficients $\underline{W} = [\underline{W}^T, \underline{W}^{F_1}, ..., \underline{W}^{F_N}]^T$. Taking back definition (3.2) we can substitute the new expression for $\underline{\mathbf{w}_T}$ and $\boldsymbol{C}_T^k\underline{\mathbf{w}_T}$, and observe that the variational condition is verified for any $\mathbf{v}_T \in \boldsymbol{\mathcal{R}}^k(T)$ iff it is verified for any element of $\{\boldsymbol{\psi}_i\}_i$. We obtain:

$$
\begin{aligned}
\sum_j \big(\boldsymbol{\psi}_j, \boldsymbol{\psi}_i\big)_T U_j &= \sum_m (\boldsymbol{\phi}_m^T, \mathbf{curl}\,\boldsymbol{\psi}_i)_T W_m^T + \sum_{F_k\in\mathcal{F}_T}\{\sum_n (\boldsymbol{\phi}_m^{F_k}, \gamma_{\tau,F}(\boldsymbol{\psi}_i\times\boldsymbol{n}_{TF}))_{F_k} W_n^{F_k}\} \quad \forall \boldsymbol{\psi}_i :\\
\sum_j MCT_{ij}\cdot U_j &= \sum_m A_{im}\cdot W_m^T + \sum_l B_{il}^1\cdot W_l^{F_1} + \cdots + \sum_l B_{il}^N\cdot W_l^{F_N} \quad \forall i :\\
MCT\cdot\underline{U} &= [A|B_1|...|B_N]\cdot\underline{W}\\
\underline{U} &= MCT^{-1}\cdot[A|B_1|...|B_N]\cdot\underline{W}
\end{aligned}
\tag{3.73}
$$

In (3.73) $MCT^{-1}\cdot[A|B_1|...|B_N]$ represents the local reconstruction matrix which allows to obtain the coefficients of the local curl reconstruction from the HHO unknowns.

4 Implementation aspects

4.1 Local static condensation

As it happens in similar methods, the practical implementation of the method relies on the equivalence between the discretized weak formulation and an algebraic linear system. Even though the HHO method is characterized by degrees of freedom which are attached to both cells and faces, it is a common practice to assemble a global system accounting only for face unknowns. The way the system is reduced is called static condensation. When this technique is applied the HHO scheme becomes a *skeletal* method, as the unknowns of the solution refer only to face degrees of freedom. Staic condensation can be implemented locally, that is cell-wise, before global assembly. This allows to perform it in parallel. To discuss the algebraic formulation of the method we make reference to the problem set in the cartesian product space:
Find $\underline{z}_h = (\underline{\mathbf{u}}_h, \underline{p}_h) \in \underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}$ such that:

$$\mathrm{A}_h\Big(\underline{z}_h, \underline{s}_h\Big) = (\boldsymbol{f}, \mathbf{v}_h)_\Omega \qquad \forall\, \underline{s}_h = (\underline{\mathbf{v}}_h, \underline{q}_h) \in \underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}, \tag{4.1}$$

where the bilinear form $\mathrm{A}_h : \underline{\mathbb{Z}}_h^{k+1} \times \underline{\mathbb{Z}}_h^{k+1} \to \mathbb{R}$ is defined by

$$\mathrm{A}_h\Big((\underline{\mathbf{w}}_h, \underline{r}_h), (\underline{\mathbf{v}}_h, \underline{q}_h)\Big) := \mathrm{a}_h(\underline{\mathbf{w}}_h, \underline{\mathbf{v}}_h) + \mathrm{b}_h(\underline{\mathbf{v}}_h, \underline{r}_h) - \mathrm{b}_h(\underline{\mathbf{w}}_h, \underline{q}_h) + \mathrm{d}_h(\underline{r}_h, \underline{q}_h). \tag{4.2}$$

It is possible to express a generic element $\underline{s}_h = (\underline{\mathbf{v}}_h, \underline{q}_h) \in \underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}$ using suitable bases for $\underline{\mathbf{X}}_{h,\mathbf{0}}^{k+1}$ and $\underline{Y}_h^{k+1}$ by storing the collection of Hilbert coefficients for $\underline{z}_h$ in a vector:

$$\underline{s}_h \to \underline{S} = \begin{bmatrix} \underline{S}_F \\ \underline{S}_T \end{bmatrix} = \begin{bmatrix} \underline{V}_F \\ \underline{Q}_F \\ \underline{V}_T \\ \underline{Q}_T \end{bmatrix} \tag{4.3}$$

The variational condition of problem (4.1) must be valid for any element of the cartesian product space $\underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}$, thus it is necessary and sufficient that it is verified for each element of the global basis. Let's consider a basis $\mathcal{B} = \{\xi_i\}_i$ of $\underline{\mathbb{Z}}_{h,\mathbb{0}}^{k+1}$. By imposing (4.1) to be verified for any element of the basis it is found that the vector of degrees of freedom of the discrete solution must solve the linear system:

$$\mathcal{A} \cdot \underline{S} = \underline{B}, \tag{4.4}$$

where $\mathcal{A}_{i,j} = \mathrm{A}_h(\xi_j, \xi_i)$ and $\underline{B}_i = (\boldsymbol{f}, \underline{\boldsymbol{\phi}}_i)_\Omega$, being $\xi_i = (\underline{\boldsymbol{\phi}}_i, \underline{\phi}_i) \in \mathcal{B}$ element of a basis for the hybrid space.

As it is evident from their definition, any of the bilinear forms that appear in definition (4.2) is the sum of cell-wise contributions. This means that the algebraic global system can be assembled starting from independent local contributions that only involve the degrees of freedom related to a cell and all of its neighbouring faces. Arranging the deegrees of freedom by putting in order all the face unknowns and at the bottom all the cell unknowns, a local contribution can be represented with the following block-wise structure:

$$A^T = \begin{bmatrix} A_{FF} & A_{FT} \\ A_{TF} & A_{TT} \end{bmatrix}, B^T = \begin{bmatrix} 0 \\ B_T \end{bmatrix}. \tag{4.5}$$

On their side, each of the three blocks which have at least one F in their label can be represented with an inner block structure referred to single faces. We can call A_{mn} the block of A_{FF} associated to faces m and n. Similarly, A_{mT} will be used to indicate the block in A_{FT} related to face m and A_{Tn} the block of A_{TF} associated with face n.
With reference to a suitable mapping from local to global indexing each of the blocks of a local contribution can be added to the global system with a procedure which is common in the implementation of similar methods.
However, it is possible to reduce the size of the contribution by applying static condensation. Consider in the global system the lines associated to cell T. One can recognize the blocks from $T-$local contribution:

$$\begin{aligned} \sum_{F \in \mathcal{F}_T} A_{TF} z_F + A_{TT} z_T = B_T \\ z_T = A_{TT}^{-1} \{ B_T - \sum_{F \in \mathcal{F}_T} A_{TF} z_F \}. \end{aligned} \tag{4.6}$$

Thus, local cell unknowns are expressible in function of the neighbouring face unknowns. Inverting A_{TT} is an affordable task, as this block embodies cell-cell bilinear forms that turn out to be symmetric and definite positive for both the vector and the scalar component. Consider now a generic internal face with label l, which is at the interface of cells labelled respectively with 1 and 2. The block of lines referred to the unknowns on l can be formulated substituting the expression derived for the cell unknowns on T_1 and T_2:

$$\begin{aligned} \{ \sum_{m \in \mathcal{F}_1} [A^1_{lm} - A^1_{l1} A^{-1}_{11} A^1_{1,m}] z_m + A^1_{1l} A^{-1}_{11} B_1 \} + \\ \{ \sum_{m \in \mathcal{F}_2} [A^2_{lm} - A^2_{l2} A^{-1}_{22} A^2_{2,m}] z_m + A^2_{2l} A^{-1}_{22} B_2 \}, \end{aligned} \tag{4.7}$$

By making the substitution a system is obtained where the only unknowns are the face coefficients z_F. In (4.7), one can recognize the blocks of a new compact local contribution referred only to face unknowns to work out cell-wise before global assembly:

$$A_F = A_{FF} - A_{FT} A_{TT}^{-1} A_{TF}, \qquad B_F = -A_{FT} A_{TT}^{-1} B_T. \tag{4.8}$$

4.2 The HArD::Core3D library

The library `HArD::Core3D` (Hybrid Arbitrary Degree::Core 3D) provides classes and methods necessary to implement hybrid numerical schemes, where polynomial unknowns can be attached to several types of geometric entity (cells, faces, edges or vertices) of a generic tridimensional mesh. In particular it provides the instruments to build HHO schemes, where unknowns are associated to cells and faces.
The main ingredients are the following:

- An environment of classes to represent a polyhedral mesh and its associated entities. Meshes are generally initialized from meshfiles in RF format. Suitable classes are defined to represent vertices, edges, faces and cells. Moreover meshes are are corredated with methods to provide number and address of neighbouring entities as well as flags to distinguish internal and boundary. A mesh object reduces to a collection of its composing elements. Among its methods, there is one to reorder unknowns to cope with strongly imposed boundary conditions. Notice that a mesh object designed like this is suitable for methods potentially dealing with unknowns attached to any geometric entity, even though HHO methods only have degrees of freedom attached to cells and faces.

```
#include "mesh.hpp"
#include "mesh_builder.hpp"

using namespace HArDCore3D;
int main() {
// Mesh file to read
std::string default_mesh =
    "../../meshes/Voro-small-0/RF_fmt/voro-4";
// Build the mesh
MeshBuilder meshbuilder = MeshBuilder(mesh_file);
std::unique_ptr<Mesh> mesh_ptr = meshbuilder.build_the_mesh();

std::cout << "There␣are␣"
          << mesh_ptr->n_cells() << "␣cells␣in␣the␣mesh.\n";
std::cout << "There␣are␣"
          << mesh_ptr->n_faces() << "␣faces␣in␣the␣mesh.\n";
          ...
}
```

Listing 4.1: Loading a mesh

- Quadrature rules. Local assembly for an HHO scheme scales down to computing quantities in the form: $(\mathcal{L}\phi_i^{F/T}, \mathcal{L}\psi_j^{F/T})_{F/T}$ where $\mathcal{L}$ is some first-order differential operator, and $\phi_i^{F/T}$, $\psi_j^{F/T}$ are two elements of a polynomial basis attached to a face or an cell. Quadrature rules shift from the need for computig local integrals to the need of providing evaluation of polynomials on quadrature nodes. `HArD::Core3D` makes use of the Dunavant rule to calibrate nodes and weights.

```
  #include <quadraturerule.hpp>
  ...
```

```
using HArDCore3D;
size_t iT = 0;    // cell index
unsigned m_K = 1; // degree of exactness
const Cell & T = mesh_ptr->cell(iT);
QuadratureRule quad_2k_T =
      generate_quadrature_rule(T, 2 * m_K);
```

Listing 4.2: Declaring a quadrature rule for a cell

- Suitable classes to represent families or bases of polynomial functions in $\mathbb{R}^d$ with $d = 2, 3$. The kind of spaces that can be attached to cells and faces in hybrid high order methods can cover a large variety. Then, the gamma of classes provided in `basis.hpp` is quite large. Starting from the most simple bases, more abstract classes are built by composing simpler classes.
  - The most simple classes to define are those for scalar polynomials. Monomials are chosen to span a basis for $\mathcal{P}^k(\mathbb{R}^2)$ and $\mathcal{P}^k(\mathbb{R}^3)$. Monomials are completely described by lists of respectively two and three integers, representing the exponents assigned to each coordinate. Generating sequence of exponents to span a basis up to a given deegree is relatively straightforward. Starting from these encoding of monomials it is possible not only to evaluate them, but also to evaluate their derivatives, by implementing automatic rules of derivation for the lists of exponents.
  - The next step consists in the definition of classes to represent local bases $\mathcal{P}^k(T)$ and $\mathcal{P}^k(F)$. As it is costumary when dealing with finite elements, local basis functions are defined as affine transformations of reference basis functions of $\mathcal{P}^k(\mathbb{R}^d)$. Cell basis functions are defined as:

$$\phi_T^i(x) := \frac{1}{h_T}\phi_*^i(x - x_T)$$

    where x_T, h_T are respectively the barycentre and diameter of the cell, and ϕ_*^i is a reference monomial. Since evaluation up to first derivation order of reference functions is provided it is straightforward to extend these methods to the transformed basis functions.

```
class MonomialScalarBasisCell
{
public:
  /// Constructor
  MonomialScalarBasisCell(
      const Cell &T, ///< A mesh cell
      size_t degree  ///< The maximum
                     ///polynomial degree to be considered
  );
  ...
  /// Evaluate the i-th basis function at point x
  FunctionValue function(size_t i, const VectorRd &x) const;

  /// Evaluate the gradient of the i-th
```

```
    /// basis function at point x
    GradientValue gradient(size_t i, const VectorRd &x) const;

  private:
    ....
    size_t m_degree;
    VectorRd m_xT;
    double m_hT;
    std::vector<VectorZd> m_powers; //list of exponents
                                    //of monomials
};
```

Listing 4.3: Representation of a basis of monomials

– Scalar polynomials are then enough to consider vectors of polynomials. To generate spaces of l-valued polynomials it is sufficient to tensorize a scalar basis:

$$\phi^i \to \begin{bmatrix} \phi^i \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ \phi^i \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ \phi^i \end{bmatrix}$$

– The last fundamental notion is that of *family*, a general collection of functions in the space spanned by a provided basis. Given access to a basis object, a family defines its members through a matrix with as many rows as the basis cardinality and as many columns as the family cardinality. Each column contains the coefficients of the expansion of a family member onto the basis. By exploiting linearity, evaluation of family members scales down to the evaluation of basis functions.

```
template <typename BasisType>
class Family {
  ...
public:
Family(
 const BasisType &basis,
        ///< The basis in which the family is expressed
 const Eigen::MatrixXd &matrix
        ///< The coefficient matrix whose
        ///<i-th line contains the coefficient
        ///<of the expansion of the i-th function
        ///<of the family in the basis
    )
    : m_basis(basis),
      m_matrix(matrix){...};
}
```

Listing 4.4: The family template class

At this evel, starting from the primitive class representing a basis for $\mathcal{P}^k(\mathbb{R}^d)$, it has been possible to build classes representing more and more complex families of l-valued polynomials on face and cells. According to necessity it

is possible to develop further this section of the library, for example to generate orthonormal basis or to consider bases for the image of a differential operator like the gradient or the curl.
Even though all the basis classes that are declared in `basis.hpp` rely on different implementations, all of them share a common utility, that is the evaluation up to first order of derivation at quadrature nodes. It is then desirable to have a common interface to evaluate families of functions regardless of the actual type of the family. Evaluation traits are designed to serve this purpose. Traits serve the role of proxy to the basis objects, so that the evaluation step in an HHO solver can be made uniform and independent of the specific chosen basis. In `HArD::Core3D` this is achieved through the function template `evaluate_quad` which can be directly used to work out local contributions. This is the case when calculating gradient or curl reconstruction matrices, to perform local static condensation, and finally to assemble the global system.

```
// Create basis (f_1,...,f_r) of degree k in cell T
MonomialScalarBasisCell basisT(T, k);
// Create quadrature rules of degree 2*k in cell T
QuadratureRule quadT = generate_quadrature_rule(T, 2*k);
// Compute values of gradients
//of basis functions at the quadrature nodes
boost::multi_array<VectorRd, 2> gradbasis_on_quadT
      = evaluate_quad<Gradient>::compute(basisT, quadT);
// Create Gram-like matrix (here, a stiffness matix)
// of (\nabla f_i,\nabla f_j)
Eigen::MatrixXd M
= compute_gram_matrix
  (gradbasis_on_quadT, gradbasis_on_quadT, quadT, true);
```

Listing 4.5: Evaluating a basis at quadrature nodes

- Classes defining HHO spaces. Classes for degrees of freedom are necessary to provide an interface between local indexing and global indexing. Each HHO space is built starting from a mesh object, and it holds the collection of cells and faces, each corredated with a suitable basis. Moreover, at construction time, the local reconstruction matrices are worked out cell-wise in parallel.

```
namespace HArDCore3D {
class YSpace : public HHOSpace {
  public:
  typedef Family<MonomialScalarBasisCell>
          CellBasisType;
  typedef Family<MonomialScalarBasisFace>
          FaceBasisType;
  typedef TensorizedVectorFamily<CellBasisType, 3>
          GradientReconstructionBasisType;
  // constructor
  YSpace(const Mesh & mesh,
        size_t K,
        bool use_threads = true,
```

```
            std::ostream & output = std::cout):... {
            ...
            m_output << "Computing␣gradient␣reconstructions"
                     << std::endl;
        // building local reconstruction matrices
            parallel_for(mesh.n_cells(),
                         construct_all_gradients,
                         m_use_threads);
        }
        ...
        private:
        // Cell bases
        std::vector<std::unique_ptr<CellBasisType> >
            m_cell_bases;
        // Face bases
        std::vector<std::unique_ptr<FaceBasisType> >
            m_face_bases;
       // Gradient reconstruction basis
        std::vector<std::unique_ptr<GradientReconstructionBasisType>>
            m_gradient_reconstruction_bases;

        // Container for local gradients
        std::vector<Eigen::MatrixXd>
            m_cell_gradients;
        std::vector<Eigen::MatrixXd>
            m_cell_gradients_rhs;
        ...}}
```

Listing 4.6: Class representing $\underline{Y}_h^{k+1}$

- The last ingredient is an HHO scheme class, with a `solve` method that performs all the steps in order:
  - Loading of the meshfile
  - Reordering of the elements to deal with strongly imposed boundary conditions
  - Instantiation of the problem-dependent HHO spaces and parallel building of the local reconstruction matrices
  - Computation of the statically condensed local contribution to the global system
  - Assembly of the global system
  - Enforcement of boundary conditions (for Dirichlet or mixed boundary conditions)
  - Solution of the global system

During the solution there are two important processes of the solution algorithm that are performed element-wise. The first is the assembly of the local reconstruction matrix performed by the constructor of any HHO space, and the second is the computation

of the local statically condensed contribution of each element to the global system. This configures an embarassingly parallel problem. The parallelism is implemented by incapsulating any cell-wise operation in a functor which can be passed as parameter to a parallel for-cycle as in the following example:

```
...
bool m_use_threads = true;
// a functor performing a cell-wise
// operation within a given range
// of indexes
std::function<void(size_t, size_t)> construct_all_cell_bases
= [this](size_t start, size_t end)->void
  {
    for (size_t iT = start; iT < end; iT++) {
      const Cell & T = *this->mesh().cell(iT);
      ...///cell-wise operations
      }
  };
m_output << "[XSpace] Constructing element bases" << std::endl;
parallel_for(mesh.n_cells(), construct_all_cell_bases, m_use_threads);
```

4.3 Original contribution

`HArD::Core3D` provides the general tools to load a mesh, specify quadrature rules and assemble a global system starting from local contributions. Then, to provide a new scheme for magnetostatics essentially five problem-dependent tasks have to be faced:

- Provide a class describing the hybrid HHO spaces $\underline{\mathbf{X}}_h^{k+1}$ an $\underline{\mathrm{Y}}_h^{k+1}$
- Define bases for cell unknowns, face unknowns and recontruction unknowns
- Describe how to build a local reconstruction matrix
- Define the local contribution specific of the problem
- Take care of strongly imposed boundary conditions in the Dirichlet and mixed case

We briefly recall that for the magnetostatics problem an HHO space discretizing the space of vector potentials is considered:

$$\underline{\mathbf{X}}_h^{k+1} = \begin{cases} \text{local cell space: } \boldsymbol{\mathcal{P}}^{k+1}(T) \\ \text{local face space: } \mathcal{P}_{\flat}^{k+1}(F) := \mathcal{P}^k(F) \oplus \mathbf{grad}_\tau(\widetilde{\mathcal{P}}^{k+2}(F)) \\ \text{local curl reconstruction space: } \mathcal{R}^k(T) := \mathbf{curl}(\mathcal{P}^{k+1}(T)) \end{cases} \tag{4.9}$$

together with another HHO space discretizing the space of pressures:

$$\underline{Y}_h^{k+1} = \begin{cases} \text{local cell space: } \mathcal{P}^k(T) \\ \text{local face space: } \mathcal{P}^{k+1}(F) \\ \text{local curl reconstruction space: } \mathcal{P}^{k+1}(T). \end{cases} \tag{4.10}$$

When defining classes for these spaces the key step is to provide properly the definition of the local bases, choosing from the gamma provided in `basis.hpp`. For $\underline{Y}_h^{k+1}$ it is relatively easy, as bases for full polynomial spaces are already provided. The task is more complex for $\underline{\mathbf{X}}_h^{k+1}$, as bases for the requested polynomial spaces aren't provided in `basis.hpp`. Actually, a simpler version of the method can be given by considering full polynomial local spaces for $\underline{\mathbf{X}}_h^{k+1}$. This choice however, other than increasing the number of local unknowns, and so the weight of the method, leads to worse convergence performance (see Remark 17, [FS20]).
As a general remark on how `HArD::Core3D` is developed an emphasis should be put on the independence of the section of code related to basis representation. Every tool in `basis.hpp` can be potentially used even beyond the frame of HHO schemes. However, new methods are added to this section according to necessity for specific problems. As a result, the developer is not provided with the most general possible library to treat polynomial spaces. It is expected that `basis.hpp` is extended when a new method is implemented. In particular it is necessary to provide a class to represent direct sums of bases, as well as a method to extract independent bases out of the image of the curl applied to a full polynomial space. Accordingly, a class template and and a template function were added to the set of utilities to represent polynomial bases:

```
/// class to represent a basis of
/// a direct sum of finite dim. spaces given
/// a basis of each one
    template <typename BasisType1, typename BasisType2>
    class BasisDirectSum {
      public:
      BasisDirectSum (BasisType1 &basis1, BasisType2 &basis2);
      ...
    }

/// function to flush out from  a family null
/// and repeated or linearly correlated elements
/// (based on numerical evaluation)
/// works for vector valued basis
///(originally thought to filter a curl rec. basis)

template <typename BasisType>
Family<BasisType> Filter (const BasisType &broadbasis,
                          const QuadratureRule &quad_rule);
```

So, in the definition of class `XSpace` representing $\underline{\mathbf{X}}_h^{k+1}$ the instantiation of local bases works like this:

```
    XSpace::XSpace(
        const Mesh & mesh,
        size_t K,
```

```
        bool use_threads,
        std::ostream & output) {
            ...
        /// cell-wise instantiation as an object function
        std::function<void(size_t, size_t)> construct_all_cell_bases
            = [this](size_t start, size_t end)->void
              {
              for (size_t iT = start; iT < end; iT++) {
                const Cell & T = *this->mesh().cell(iT);
                MonomialScalarBasisCell basis_Pk_T(T, m_K);
                QuadratureRule quad_2k_T =
                generate_quadrature_rule(T, 2 * m_K);
                auto basis_Pk_T_quad =
                evaluate_quad<Function>::compute(basis_Pk_T, quad_2k_T);
                ///storing cell basis (after orthonormalization)
                this->m_cell_bases[iT].reset
                  ( new CellBasisType(l2_orthonormalize(basis_Pk_T,
                    quad_2k_T,
                    basis_Pk_T_quad)) );

                typedef CurlBasis
                <TensorizedVectorFamily<MonomialScalarBasisCell, 3>>
                temp_basis_type;
                CurlBasis
                <TensorizedVectorFamily<MonomialScalarBasisCell, 3>>
                    basis_curl =
                        CurlBasis(TensorizedVectorFamily
                                <MonomialScalarBasisCell,3>
                                (MonomialScalarBasisCell(T, m_K)));
                   ///take out null or repeated elements
                   ///store the curl recontruction basis
                  this->m_curl_reconstruction_bases[iT].reset
                    ( new CurlReconstructionBasisType (
                          Filter<temp_basis_type>
                          (basis_curl, quad_2k_T)));
                 }

              };
        /// parallel instantiation
        parallel_for(mesh.n_cells(),
                     construct_all_cell_bases,
                     m_use_threads);
        }
```

The following issue regards the building of the curl reconstruction matrix. We recall that the curl reconstruction $\boldsymbol{C}_T^k \underline{\mathbf{v}_T} \in \mathcal{R}^k(T)$ is defined as:

$$\left(\boldsymbol{C}_T^k \underline{\mathbf{u}}_T, \mathbf{w}\right)_T = (\mathbf{u}_T, \mathbf{curl}\, \mathbf{w})_T + \sum_{F \in \mathcal{F}_T} (\mathbf{u}_F, \gamma_{\tau,F}(\mathbf{w} \times \boldsymbol{n}_{TF}))_F \quad \forall \mathbf{w} \in \mathcal{R}^k(T). \tag{4.11}$$

A problem emerges when realizing that to compute the quantity $(\mathbf{u}_T, \mathbf{curl}\, \phi_i)_T$ for a generic element ϕ_i of the basis for $\mathcal{R}^k(T)$ it is necessary to provide second order derivatives of polynomials on a cell. This is true because curl bases essentially hold a

scalar basis on the same element, such that the evaluation of a curl basis function is provided as the evaluation of the curl of its scalar ancestor. Then, evaluating the curl of a curl basis function implies applying two times the curl operator to the ancestor, which eventually requires computing second order derivatives. However computation of second order derivatives is not a method provided by scalar polynomials. The direct solution to this problem is extending the class `MonomialScalarBasisCell` to provide the hessian matrix of each basis function. On top of this it is possible to definine a **curl curl** operator. An indirect solution resides in finding an equivalent formulation of definition (4.11) for the curl reconstruction. This is possible by applying an integration by parts to term $(\mathbf{u}_T, \mathbf{curl}\,\phi_i)_T$. This leads to an alternative formulation, given by:

$$\left(\boldsymbol{C}_T^k \underline{\mathbf{u}}_T, \mathbf{w}\right)_T = (\mathbf{curl}\,\mathbf{u}_T, \mathbf{w})_T + \sum_{F \in \mathcal{F}_T} (\mathbf{u}_F - \mathbf{u}_T, \gamma_{\tau,F}(\mathbf{w} \times \boldsymbol{n}_{TF}))_F \quad \forall \mathbf{w} \in \mathcal{R}^k(T). \tag{4.12}$$

With this work-around, no evaluation of derivatives is required for curl basis functions, eliminating the need to manipulate the class `MonomialScalarBasisCell`. As for the `YSpace` representing $\underline{Y}_h^{k+1}$, it can instead rely completely on tools that are already provided by the original library.

Once the definition of the HHO spaces is given the next step is assigning the local contribution. The construction of local contribution is provided as a method of the solver class `HHO-magnetostatics`. The rest of the algorithm follows standard steps. From dense local contributions a sparse global system is assembled. A standard lifting technique is used to enforce strong boundary conditions, and a suitable method can be chosen to solve the linear system.

5 Numerical results

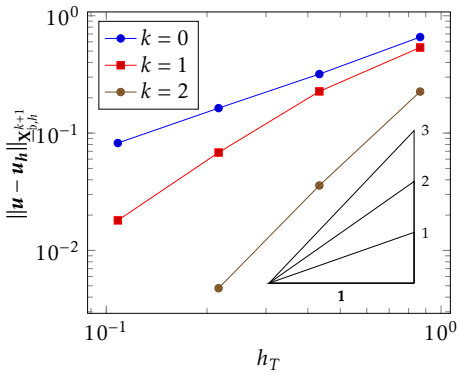
In this section the results of convergence tests are displayed. For each mesh sequence the method is applied varying the maximum polynomial degree and reporting the trend of the error with respect to the meshsize h_T and the total number of degrees of freedom. Also, estimates of the convergence rates are provided. The norm of the error was considered both in the sense of (3.10) and in the sense L_2. Several kinds of polytopal mesh families were tested.

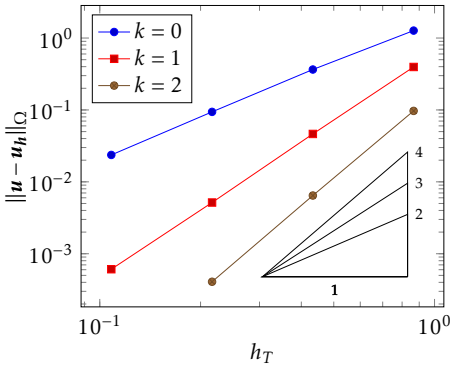
Dirichlet Boundary Conditions

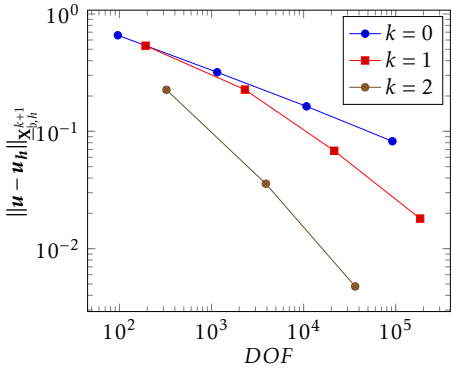
To realize numerical tests on Dirichlet boundary conditions, the method was applied to the problem with the following data:

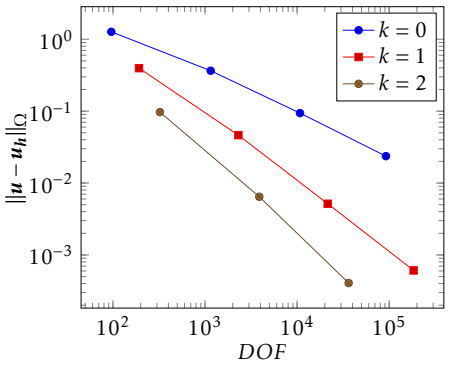
- $\Omega = [0,1]^3$
- $\boldsymbol{u} = \begin{bmatrix} \sin(\pi y) sin(\pi z) \\ \sin(\pi x) * sin(\pi z) \\ \sin(\pi x) * sin(\pi y) \end{bmatrix}$
- $p = \sin(\pi x)\sin(\pi y)\sin(\pi z)$
- $\boldsymbol{f} = 2\pi^2 \begin{bmatrix} \sin(\pi y)\sin(\pi z) \\ \sin(\pi x) sin(\pi z) \\ \sin(\pi x)\sin(\pi y) \end{bmatrix} + \mathbf{grad}\, p$

Figure 5.1: Dirichlet-Mesh Family: CubicCells









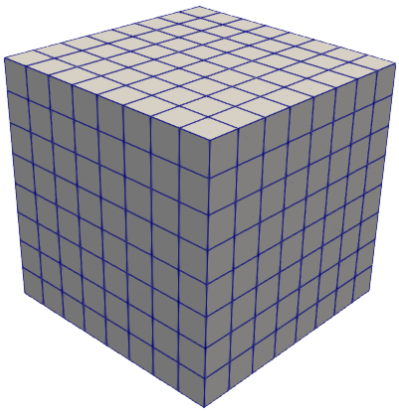
---- Convergence Rates: K=0 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	6.59e-01	
4.33e-01	3.19e-01	1.048
2.17e-01	1.63e-01	0.965
1.08e-01	8.23e-02	0.988

L2 error:

Mesh size	Error	Rate
8.66e-01	1.27e+00	
4.33e-01	3.65e-01	1.799
2.17e-01	9.40e-02	1.957
1.08e-01	2.37e-02	1.990



---- Convergence Rates: K=1 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	5.37e-01	
4.33e-01	2.27e-01	1.243
2.17e-01	6.83e-02	1.731
1.08e-01	1.80e-02	1.921

L2 error:

Mesh size	Error	Rate
8.66e-01	3.96e-01	
4.33e-01	4.64e-02	3.094
2.17e-01	5.16e-03	3.169
1.08e-01	6.09e-04	3.082

---- Convergence Rates: K=2 ----

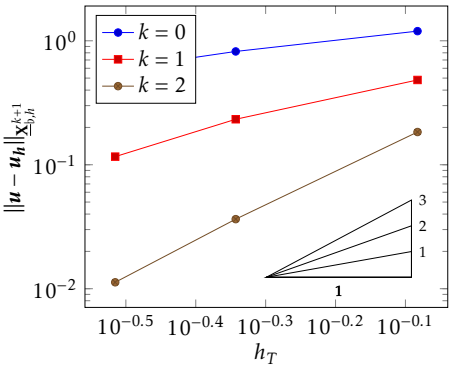
Energy error:

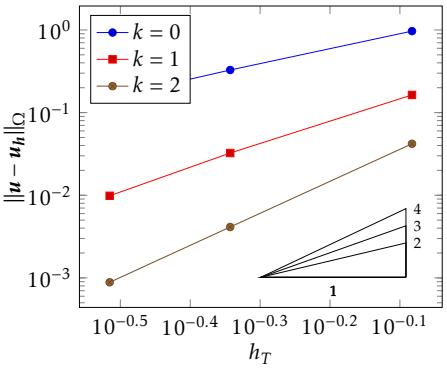
Mesh size	Error	Rate
8.66e-01	2.26e-01	
4.33e-01	3.58e-02	2.659
2.17e-01	4.77e-03	2.908

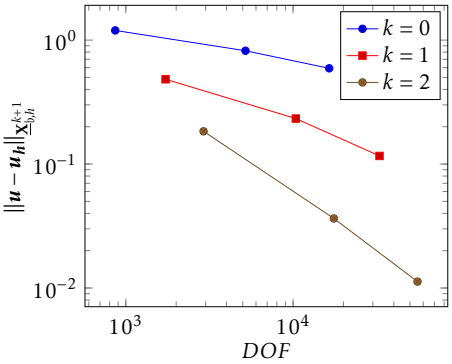
L2 error:

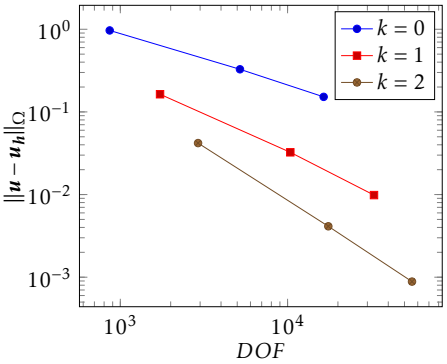
Mesh size	Error	Rate
8.66e-01	9.70e-02	
4.33e-01	6.45e-03	3.910
2.17e-01	4.08e-04	3.984

Figure 5.3: Dirichlet - Mesh Family: VoroSmall



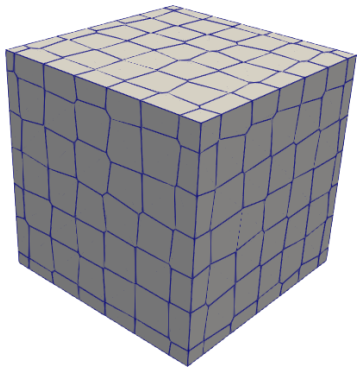






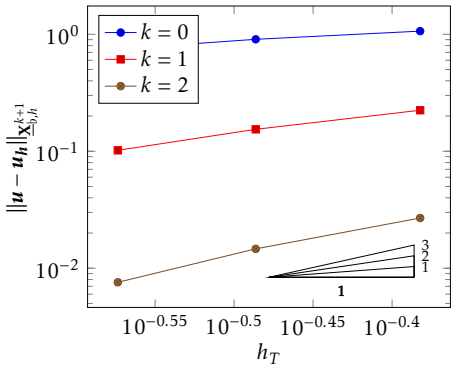

```
 ---- Compute convergence rates ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     1.20e+00
    4.54e-01     8.20e-01        0.630
    3.05e-01     5.92e-01        0.822
L2 error:
   Mesh size        Error         Rate
    8.27e-01     9.68e-01
    4.54e-01     3.28e-01        1.807
    3.05e-01     1.52e-01        1.934
```

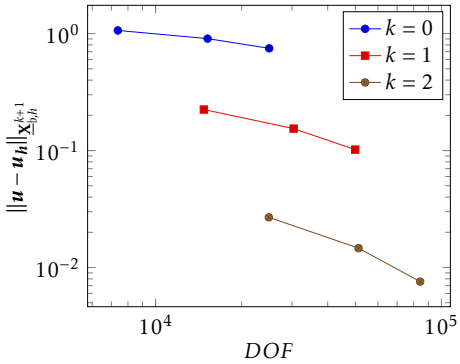
```
 ---- Compute convergence rates ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     4.83e-01
    4.54e-01     2.33e-01        1.219
    3.05e-01     1.16e-01        1.749
L2 error:
   Mesh size        Error         Rate
    8.27e-01     1.64e-01
    4.54e-01     3.24e-02        2.701
    3.05e-01     9.85e-03        3.002
```

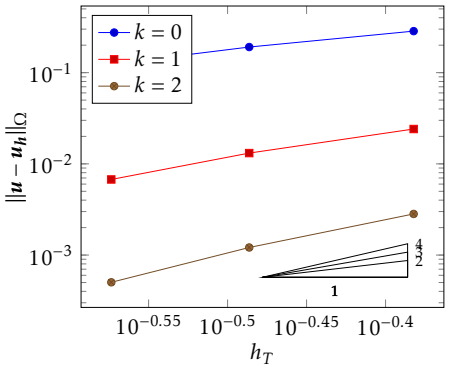


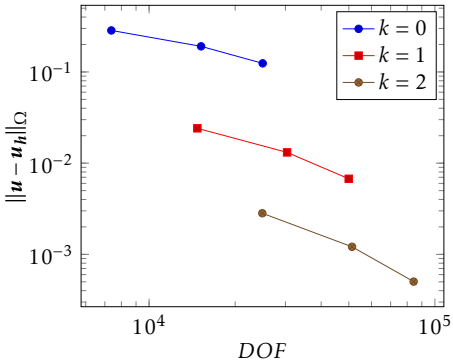
```
 ---- Compute convergence rates ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     1.84e-01
    4.54e-01     3.64e-02        2.702
    3.05e-01     1.13e-02        2.955
L2 error:
   Mesh size        Error         Rate
    8.27e-01     4.20e-02
    4.54e-01     4.13e-03        3.871
    3.05e-01     8.85e-04        3.881
```

Figure 5.5: Dirichlet - Mesh Family: VoroTets

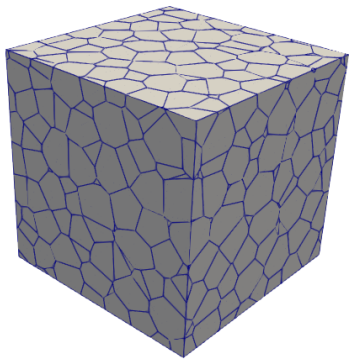






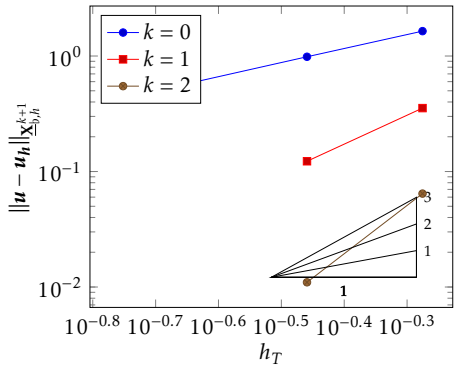


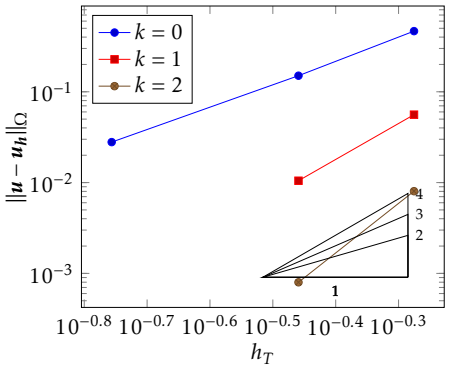

```
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    4.15e-01    1.06e+00
    3.26e-01    9.06e-01       0.672
    2.67e-01    7.48e-01       0.952
L2 error:
   Mesh size       Error        Rate
    4.15e-01    2.85e-01
    3.26e-01    1.91e-01       1.675
    2.67e-01    1.24e-01       2.136
```

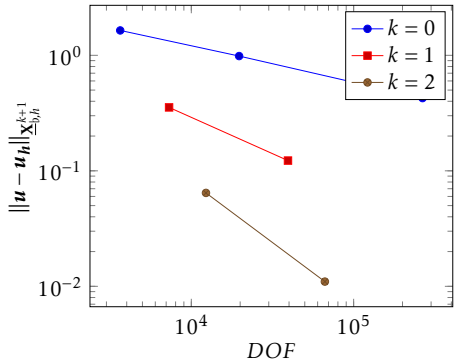


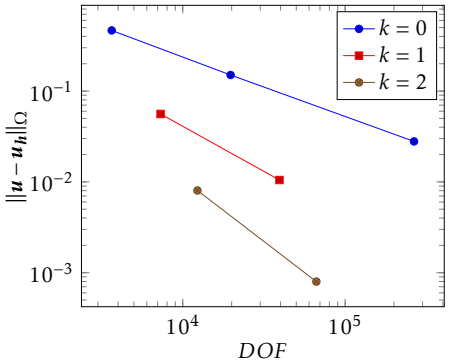
```
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    4.15e-01    2.24e-01
    3.26e-01    1.54e-01       1.567
    2.67e-01    1.02e-01       2.056
L2 error:
   Mesh size       Error        Rate
    4.15e-01    2.41e-02
    3.26e-01    1.31e-02       2.535
    2.67e-01    6.74e-03       3.312
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    4.15e-01    2.69e-02
    3.26e-01    1.46e-02       2.534
    2.67e-01    7.58e-03       3.275
L2 error:
   Mesh size       Error        Rate
    4.15e-01    2.82e-03
    3.26e-01    1.21e-03       3.535
    2.67e-01    5.03e-04       4.371
```

Figure 5.7: Dirichlet - Mesh Family: RandomHexahedra



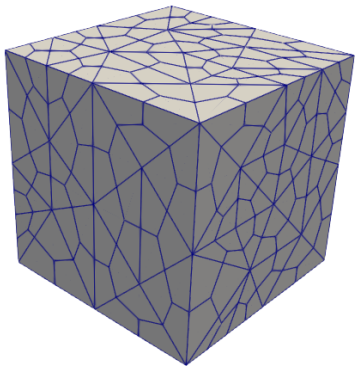







```
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    5.30e-01    1.64e+00
    3.47e-01    9.85e-01       1.208
    1.75e-01    4.26e-01       1.224
L2 error:
   Mesh size       Error        Rate
    5.30e-01    4.65e-01
    3.47e-01    1.50e-01       2.671
    1.75e-01    2.79e-02       2.461
```

```
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    5.30e-01    3.54e-01
    3.47e-01    1.23e-01       2.505
L2 error:
   Mesh size       Error        Rate
    5.30e-01    5.60e-02
    3.47e-01    1.05e-02       3.958
```



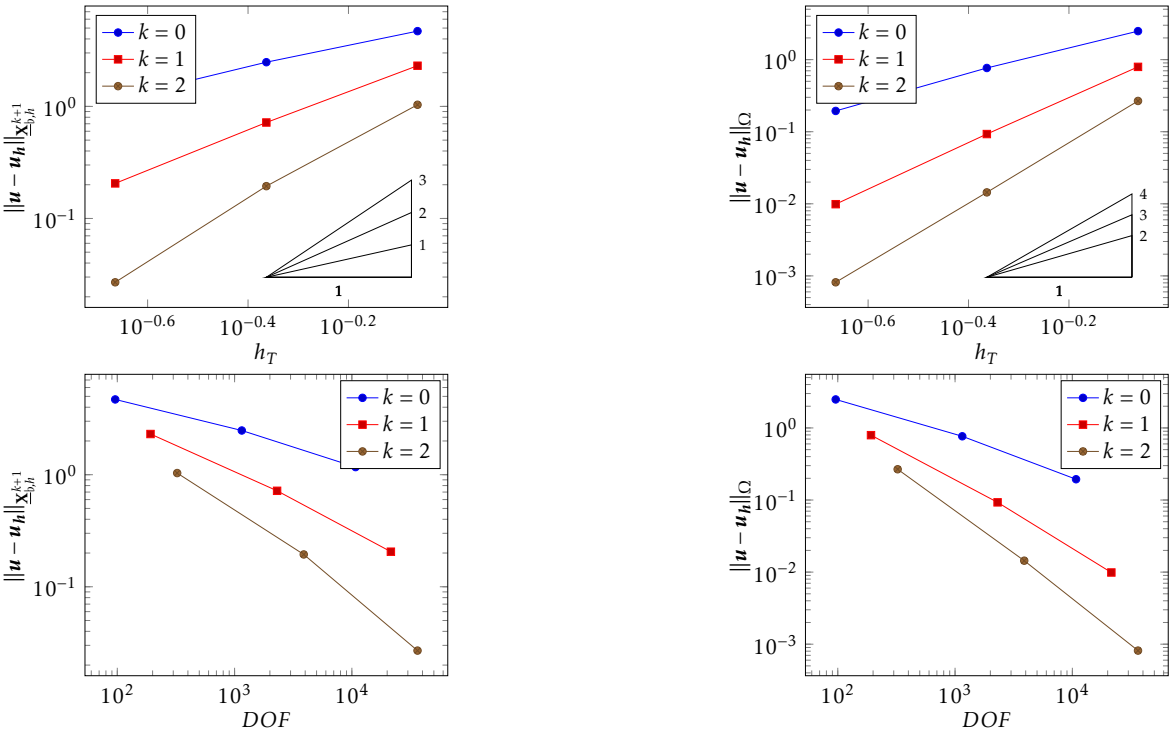
```
 ---- Compute convergence rates ----
Energy error:
   Mesh size       Error        Rate
    5.30e-01    6.44e-02
    3.47e-01    1.10e-02       4.179
L2 error:
   Mesh size       Error        Rate
    5.30e-01    8.04e-03
    3.47e-01    7.98e-04       5.460
```

Neumann boundary conditions

To realize numerical tests on Neumann boundary conditions, the method was applied to the problem with the following data:

- $\Omega = [0,1]^3$
- $\boldsymbol{u} = \begin{bmatrix} \sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \cos(\pi x)\sin(\pi x)\cos(\pi z) \\ -2\cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix}$
- $p = 0$
- $\boldsymbol{f} = 3\pi^2 \begin{bmatrix} \sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \cos(\pi x)\sin(\pi x)\cos(\pi z) \\ -2\cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix}$

Figure 5.9: Neumann - Mesh Family: CubicCells



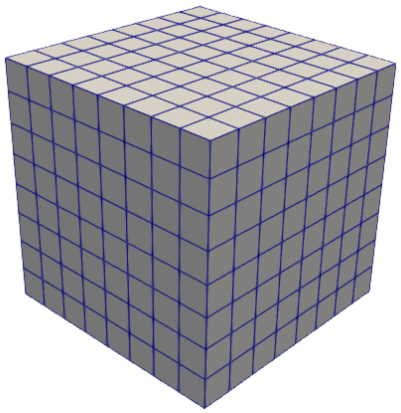
---- K=0 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	4.70e+00	
4.33e-01	2.48e+00	0.921
2.17e-01	1.17e+00	1.085

L2 error:

Mesh size	Error	Rate
8.66e-01	2.49e+00	
4.33e-01	7.66e-01	1.698
2.17e-01	1.94e-01	1.980



---- K=1 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	2.31e+00	
4.33e-01	7.18e-01	1.683
2.17e-01	2.06e-01	1.805

L2 error:

Mesh size	Error	Rate
8.66e-01	7.94e-01	
4.33e-01	9.28e-02	3.096
2.17e-01	9.88e-03	3.232

---- K=2 ----

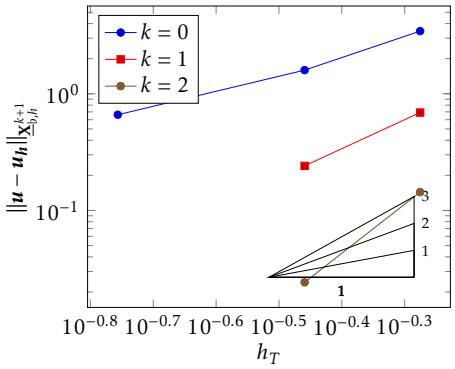
Energy error:

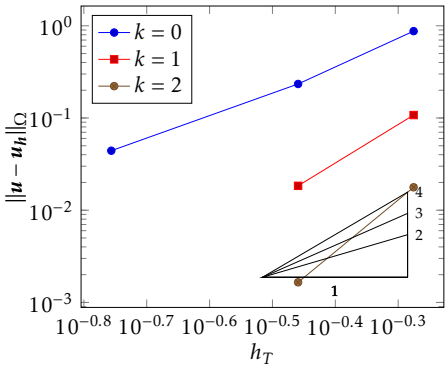
Mesh size	Error	Rate
8.66e-01	1.03e+00	
4.33e-01	1.94e-01	2.413
2.17e-01	2.69e-02	2.851

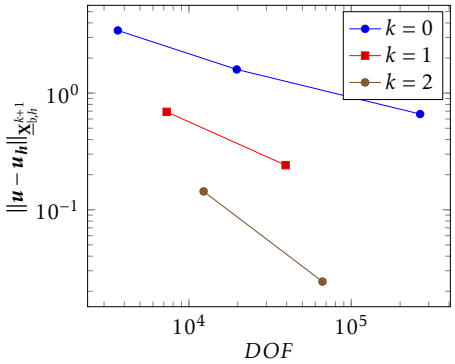
L2 error:

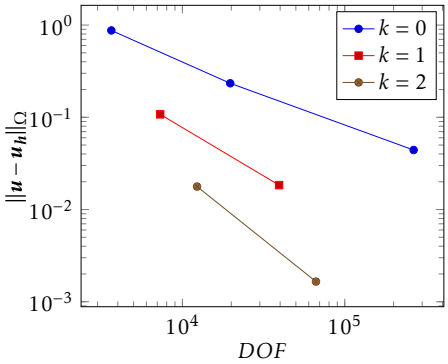
Mesh size	Error	Rate
8.66e-01	2.67e-01	
4.33e-01	1.44e-02	4.213
2.17e-01	8.15e-04	4.144

Figure 5.11: Neumann - Mesh Family: RandomHexahedra









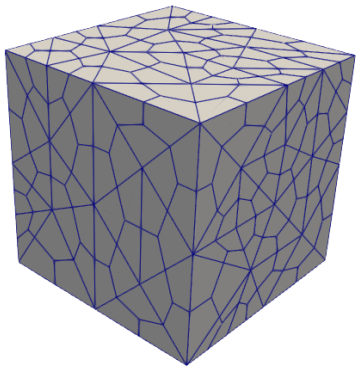
---- K=0 ----

Energy error:

Mesh size	Error	Rate
5.30e-01	3.45e+00	
3.47e-01	1.60e+00	1.820
1.75e-01	6.62e-01	1.287

L2 error:

Mesh size	Error	Rate
5.30e-01	8.75e-01	
3.47e-01	2.34e-01	3.119
1.75e-01	4.42e-02	2.435



---- K=1 ----

Energy error:

Mesh size	Error	Rate
5.30e-01	6.91e-01	
3.47e-01	2.41e-01	2.487

L2 error:

Mesh size	Error	Rate
5.30e-01	1.08e-01	
3.47e-01	1.84e-02	4.176

---- K=2 ----

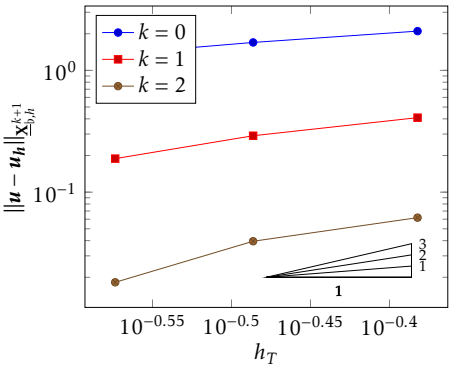
Energy error:

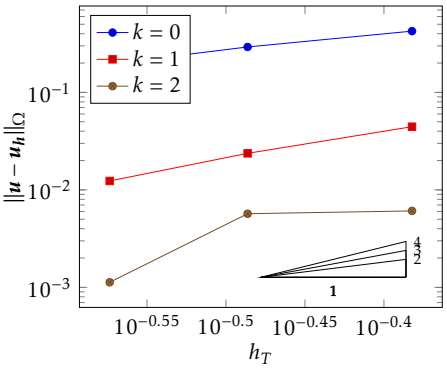
Mesh size	Error	Rate
5.30e-01	1.44e-01	
3.47e-01	2.42e-02	4.207

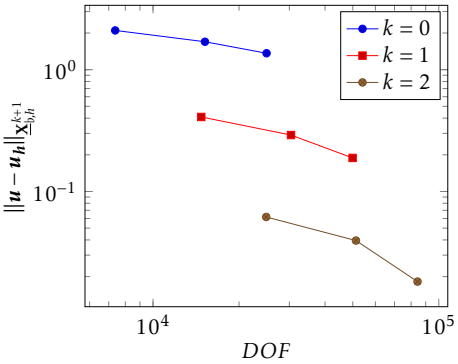
L2 error:

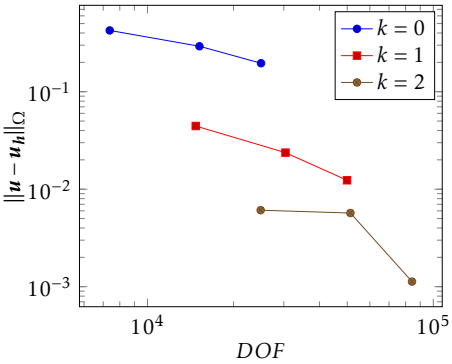
Mesh size	Error	Rate
5.30e-01	1.77e-02	
3.47e-01	1.65e-03	5.608

Figure 5.13: Neumann - Mesh Family: VoroTets









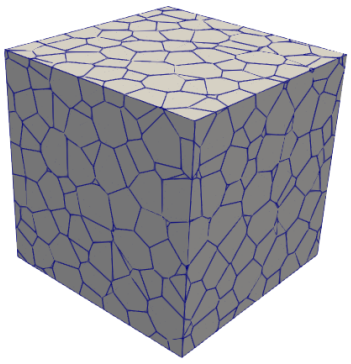
---- K=0 ----

Energy error:

Mesh size	Error	Rate
4.15e-01	2.10e+00	
3.26e-01	1.70e+00	0.893
2.67e-01	1.36e+00	1.083

L2 error:

Mesh size	Error	Rate
4.15e-01	4.25e-01	
3.26e-01	2.93e-01	1.560
2.67e-01	1.96e-01	1.993



---- K=1 ----

Energy error:

Mesh size	Error	Rate
4.15e-01	4.09e-01	
3.26e-01	2.91e-01	1.427
2.67e-01	1.89e-01	2.151

L2 error:

Mesh size	Error	Rate
4.15e-01	4.45e-02	
3.26e-01	2.37e-02	2.625
2.67e-01	1.23e-02	3.248

---- K=2 ----

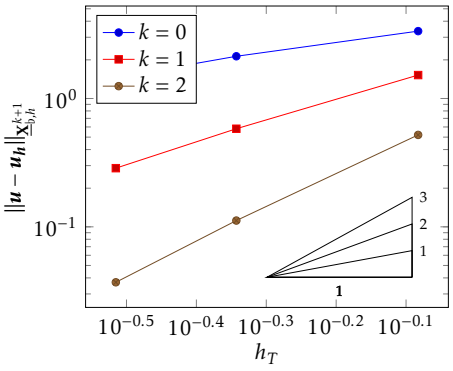
Energy error:

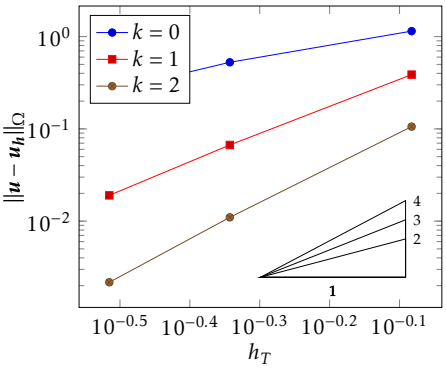
Mesh size	Error	Rate
4.15e-01	6.18e-02	
3.26e-01	3.95e-02	1.861
2.67e-01	1.82e-02	3.860

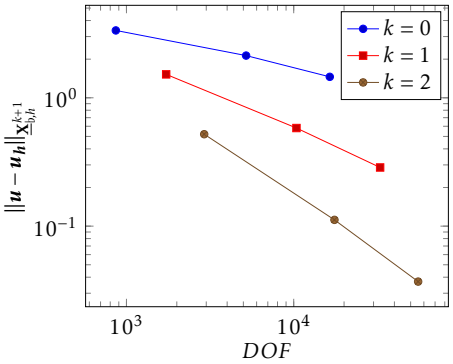
L2 error:

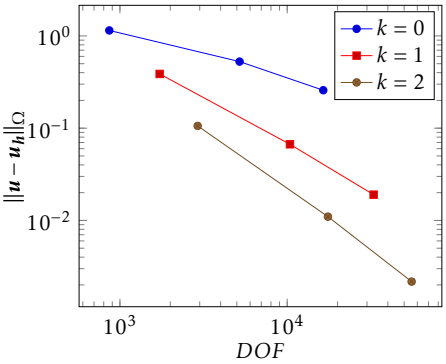
Mesh size	Error	Rate
4.15e-01	6.09e-03	
3.26e-01	5.70e-03	0.278
2.67e-01	1.13e-03	8.058

Figure 5.15: Neumann - Mesh Family: VoroSmall









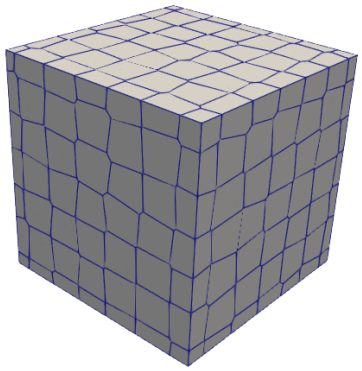
---- K=0 ----

Energy error:

Mesh size	Error	Rate
8.27e-01	3.35e+00	
4.54e-01	2.13e+00	0.753
3.05e-01	1.46e+00	0.960

L2 error:

Mesh size	Error	Rate
8.27e-01	1.15e+00	
4.54e-01	5.27e-01	1.298
3.05e-01	2.58e-01	1.801



---- K=1 ----

Energy error:

Mesh size	Error	Rate
8.27e-01	1.52e+00	
4.54e-01	5.81e-01	1.607
3.05e-01	2.86e-01	1.782

L2 error:

Mesh size	Error	Rate
8.27e-01	3.87e-01	
4.54e-01	6.69e-02	2.932
3.05e-01	1.90e-02	3.166

---- K=2 ----

Energy error:

Mesh size	Error	Rate
8.27e-01	5.20e-01	
4.54e-01	1.12e-01	2.566
3.05e-01	3.70e-02	2.790

L2 error:

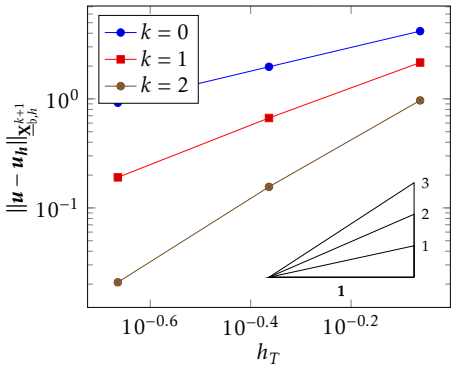
Mesh size	Error	Rate
8.27e-01	1.06e-01	
4.54e-01	1.10e-02	3.781
3.05e-01	2.17e-03	4.082

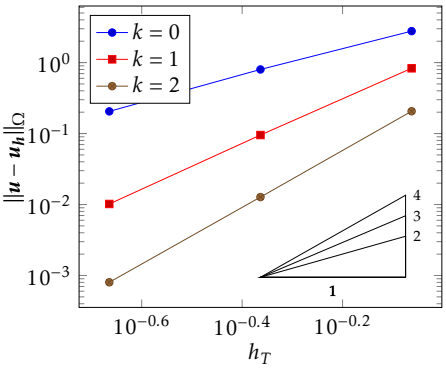
Mixed boundary conditions

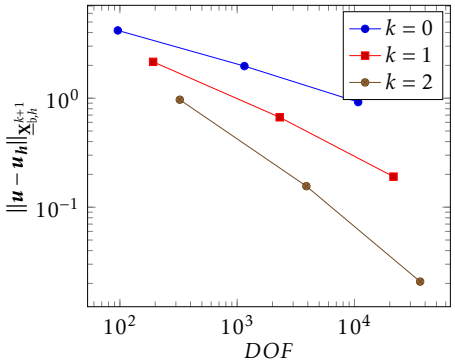
To realize numerical tests on mixed conditions, the method was applied to the problem with the following data:

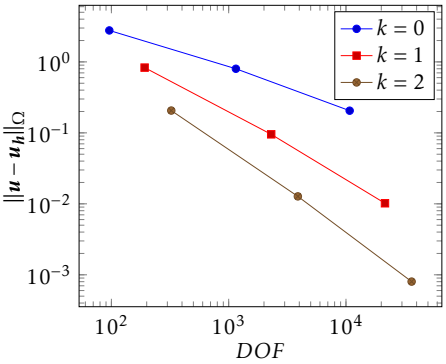
- $\Omega = [0,1]^3$
- $\boldsymbol{u} = \begin{bmatrix} -4\sin(0.5\pi(x-1))\cos(\pi y)\cos(\pi z) \\ \cos(0.5\pi(x-1))\sin(\pi y)\cos(\pi z) \\ \cos(0.5\pi(x-1))\cos(\pi y)\sin(\pi z) \end{bmatrix}$
- $p = 0$
- $\boldsymbol{f} = \frac{9}{4}\pi^2 \begin{bmatrix} -4\sin(0.5\pi(x-1))\cos(\pi y)\cos(\pi z) \\ \cos(0.5\pi(x-1))\sin(\pi y)\cos(\pi z) \\ \cos(0.5\pi(x-1))\cos(\pi y)\sin(\pi z) \end{bmatrix}$

Figure 5.17: Mixed - Mesh Family: CubicCells









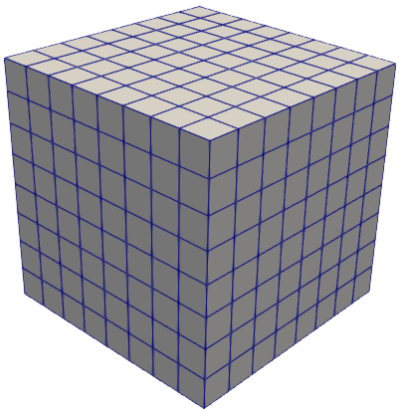
---- K=0 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	4.18e+00	
4.33e-01	1.97e+00	1.086
2.17e-01	9.21e-01	1.097

L2 error:

Mesh size	Error	Rate
8.66e-01	2.77e+00	
4.33e-01	8.00e-01	1.794
2.17e-01	2.05e-01	1.963



---- K=1 ----

Energy error:

Mesh size	Error	Rate
8.66e-01	2.15e+00	
4.33e-01	6.68e-01	1.689
2.17e-01	1.91e-01	1.809

L2 error:

Mesh size	Error	Rate
8.66e-01	8.31e-01	
4.33e-01	9.53e-02	3.124
2.17e-01	1.02e-02	3.226

---- K=2 ----

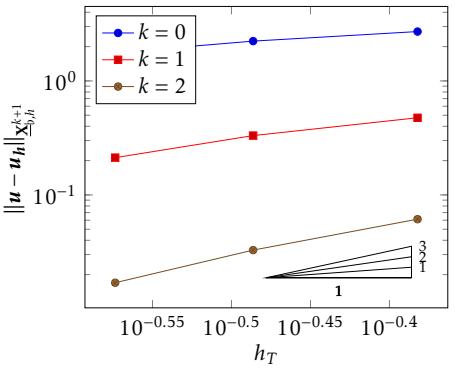
Energy error:

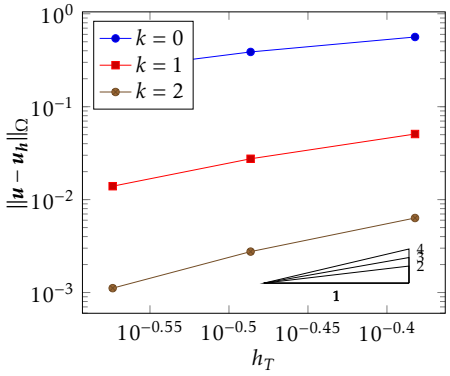
Mesh size	Error	Rate
8.66e-01	9.68e-01	
4.33e-01	1.56e-01	2.634
2.17e-01	2.08e-02	2.908

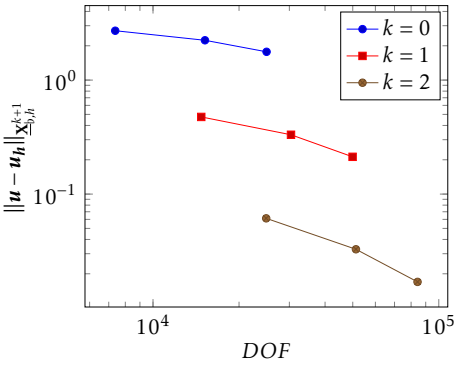
L2 error:

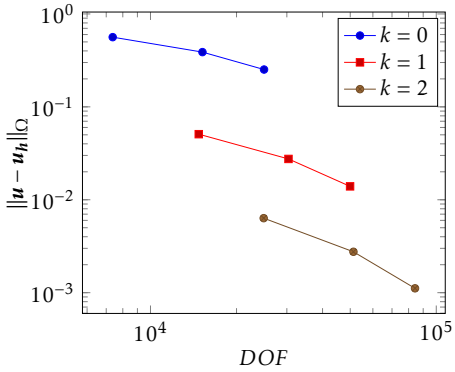
Mesh size	Error	Rate
8.66e-01	2.06e-01	
4.33e-01	1.27e-02	4.017
2.17e-01	8.02e-04	3.989

Figure 5.19: Mixed - Mesh Family: VoroTets









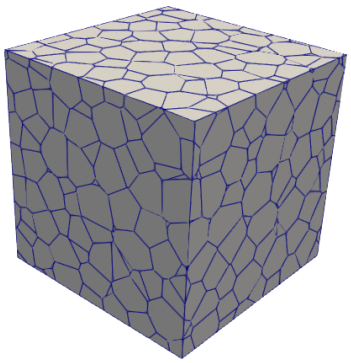
---- K=0 ----

Energy error:

Mesh size	Error	Rate
4.15e-01	2.71e+00	
3.26e-01	2.23e+00	0.806
2.67e-01	1.77e+00	1.158

L2 error:

Mesh size	Error	Rate
4.15e-01	5.63e-01	
3.26e-01	3.88e-01	1.546
2.67e-01	2.52e-01	2.150



---- K=1 ----

Energy error:

Mesh size	Error	Rate
4.15e-01	4.75e-01	
3.26e-01	3.31e-01	1.503
2.67e-01	2.12e-01	2.216

L2 error:

Mesh size	Error	Rate
4.15e-01	5.08e-02	
3.26e-01	2.75e-02	2.560
2.67e-01	1.39e-02	3.382

---- K=2 ----

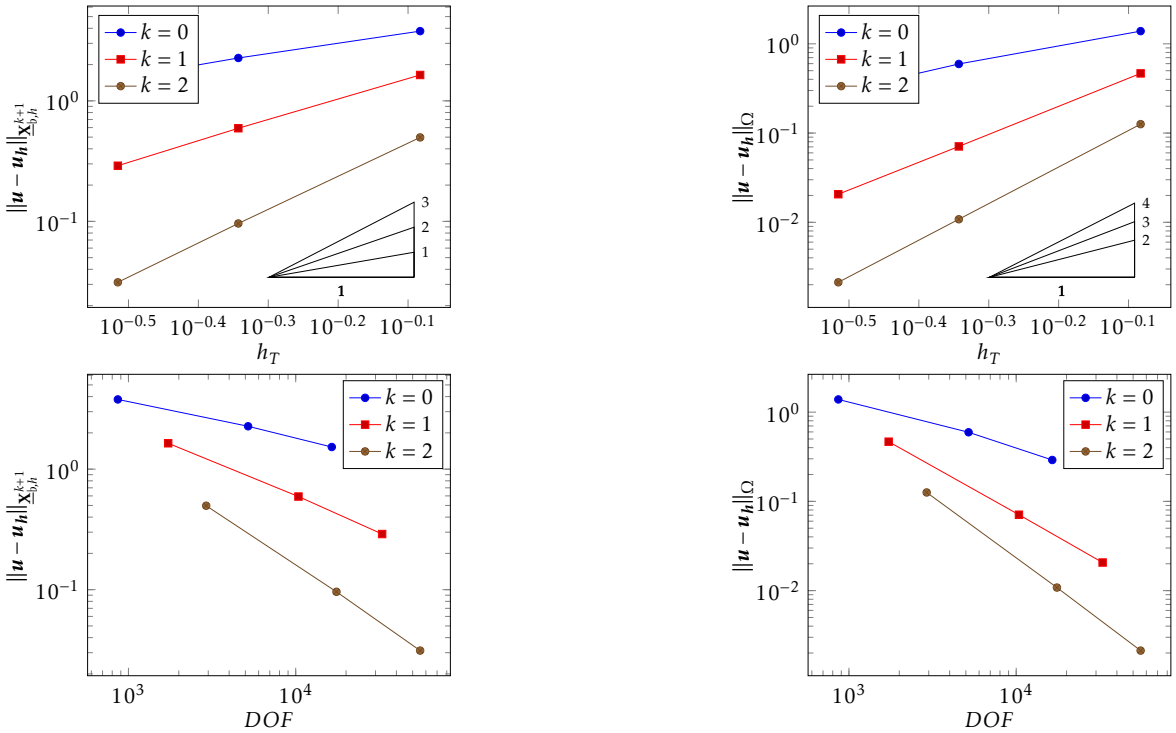
Energy error:

Mesh size	Error	Rate
4.15e-01	6.13e-02	
3.26e-01	3.28e-02	2.608
2.67e-01	1.70e-02	3.277

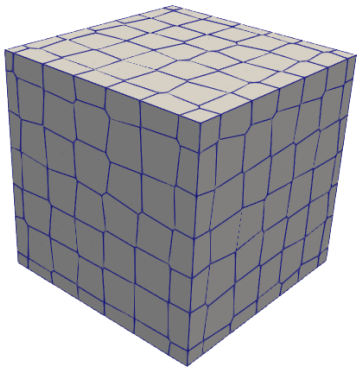
L2 error:

Mesh size	Error	Rate
4.15e-01	6.34e-03	
3.26e-01	2.76e-03	3.476
2.67e-01	1.11e-03	4.505

Figure 5.21: Mixed - Mesh Family: VoroSmall




```
 ---- K=0 ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     3.79e+00
    4.54e-01     2.27e+00        0.857
    3.05e-01     1.53e+00        0.998
L2 error:
   Mesh size        Error         Rate
    8.27e-01     1.39e+00
    4.54e-01     5.95e-01        1.415
    3.05e-01     2.91e-01        1.801
```

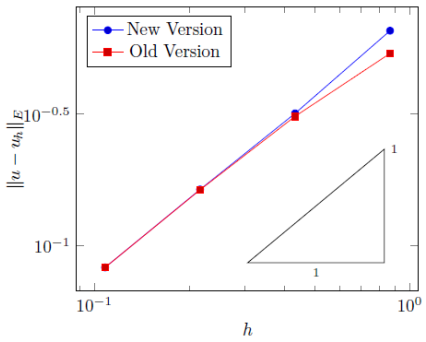


```
 ---- K=1 ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     1.64e+00
    4.54e-01     5.93e-01        1.699
    3.05e-01     2.90e-01        1.804
L2 error:
   Mesh size        Error         Rate
    8.27e-01     4.66e-01
    4.54e-01     7.09e-02        3.146
    3.05e-01     2.06e-02        3.107
 ---- K=2 ----
Energy error:
   Mesh size        Error         Rate
    8.27e-01     4.97e-01
    4.54e-01     9.60e-02        2.745
    3.05e-01     3.12e-02        2.830
L2 error:
   Mesh size        Error         Rate
    8.27e-01     1.26e-01
    4.54e-01     1.08e-02        4.097
    3.05e-01     2.13e-03        4.096
```

6 Conclusions

The simulation output matches the expected convergence rate, and this is true independently of the chosen mesh family. Even the results for Neumann and mixed conditions show encouraging results. We can conclude that after this work a tool is available to approximate a magnetic potential field inside a finite domain with rather general boundary conditions. For mixed conditions this was at least shown numerically.

To give an idea of how the method's performance is improved after considering reduced local bases for the vector unknown reconstruction we show a comparison between a scheme with optimized **curl** reconstruction and a raw scheme where full polynomial spaces are chosen to define the space of vector unknowns. The result is a reduction of around 11% of the dimension of the global system with no deterioration in the error. On the other hand this result should be balanced by the overhead brought by the filtering of the curl basis which is a step added to the optimized version. Also, full polynomial spaces may be more performing for coarser meshes.



k=0			k=1		
DOFS old	DOFS new	saved %	DOFS old	DOFS new	saved %
104	96	7%	216	192	11%
1296	1152	11%	2592	2304	11%
12096	10752	11%	24192	21504	11%
103680	92160	11%	207360	184320	11%

Figure 6.1: *Dirichlet BCs - Cubic Cells: optimization gain*

In future the method's performance may be improved on several grounds. A more organic representation of the variety of classes used to represent polynomial bases could be the next goal. Also, optimizatin on the algebraic side could be explored more deeply, possibly by looking for preconditioning strategies or multigrid approaches. Also, a different reconstruction basis based on potential reconstruction may lead to even less degrees of freedom with no accuracy degradation. Probably the next challenge will be a generalized method which shows stability even when the domain topology is not trivial. This would have a positive effect for application with complex geometries. The next achievements will be presumably the reuslts of merging HHO theory with

typical methods of computational algebraic topology.

Bibliography

[ACL18] F. Assous, P. Ciarlet Jr., and S. Labrunie. *Mathematical foundations of computational electromagnetism*. Vol. 198. Applied Mathematical Sciences. Springer, Cham, 2018. ISBN: 978-3-319-70841-6; 978-3-319-70842-3. DOI: 10.1007/978-3-319-70842-3.

[D A21] J. D. D. A. Di Pietro. "An arbitrary-order discrete de Rham complex on polyhedral meshes: Exactness, Poincaré inequalities, and consistency". In: (2021). DOI: arXiv:2101.04940.

[DD20] D. A. Di Pietro and J. Droniou. *The Hybrid High-Order method for polytopal meshes. Design, analysis, and applications*. Modeling, Simulation and Applications 19. Springer International Publishing, 2020. ISBN: 978-3-030-37202-6 (Hardcover) 978-3-030-37203-3 (eBook). DOI: 10.1007/978-3-030-37203-3.

[DS14] A. D.A. Di Pietro and S.Lemaire. "An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators". In: *Comput. Meth. Appl. Math* 14.4 (2014), pp. 461–472. DOI: 10.1515/cmam-2014-0018.

[FS20] D. D. P. F. Chave and S.Lemaire. "A discrete Weber inequality on three-dimensional hybrid spaces with application to the HHO approximation of magnetostatics". In: (2020). DOI: hal.02892526.

[Kik89] F. Kikuchi. "Mixed formulations for finite element analysis of magnetostatic and electrostatic problems". In: *Japan J. Appl. Math.* 6.2 (1989), pp. 209–221. DOI: 10.1007/BF03167879.

[LDL60] E. L. L.D.Landau. *Electrodynamics of continuous media*. Vol. 198. Pergamon Press, 1960. ISBN: 0-08-030276-9.

[Mon03] P. Monk. "Finite Element Methods for Mawxwell's Equations". In: Numerical Mathematics and Scientific Computation (2003). DOI: 10.1093/acprof:oso/9780198508885.001.0001.

[P D13] R. S. P. Dłotko. "A novel technique for cohomology computations in engineering practice". In: *Comput. Methods Appl. Mech. Engrg.* 253 (2013), pp. 530–542. DOI: 10.1016/j.cma.2012.08.009.

[Qua] A. Quarteroni. *Numerical Models for Differential Models*. Vol. 8. Modeling, Simulations Applications. Springer, pp. 267–273. ISBN: 978-88-470-5533-3. DOI: 10.1007/978-88-470-5522-3.

[Sal] S. Salsa. *Equazioni a derivate parziali: Metodi, modelli e applicazioni*. Vol. 97. Unitext. Springer. ISBN: 978-88-470-5783-8. DOI: 10.1007/978-470-5782-2.