Practical unstructured splines: Algorithms, multi-patch spline spaces, and some applications to numerical analysis

Stefano Frambati, Hélène Barucq, Henri Calandra, Julien Diaz

Practical unstructured splines: algorithms, multi-patch spline spaces, and some applications to numerical analysis

Stefano Frambati*[1], Hélène Barucq[1], Henri Calandra[1], and Julien Diaz[1]

[1]Makutu, Inria, TotalEnergies, Universite de Pau et des Pays de l'Adour, CNRS, Avenue de l'Université, F64000 Pau

Abstract

In this work, we show how some recent advances on simplex spline spaces can be used to construct a polynomial-reproducing space of unstructured splines on multi-patch domains of arbitrary shape and topology. The traces of these functions on the subdomain boundaries reproduce the usual traces of standard polynomial bases used in discontinuous Galerkin (DG) approximations, allowing to borrow many theoretical and practical tools from these methods. Concurrently, we recast some theoretical results on the construction and evaluation of spaces of simplex splines into an explicit, algorithmic form. Together, these efforts allow to formulate a practical, efficient and fully unstructured multi-patch discontinuous Galerkin - isogeometric analysis (DG-IGA) scheme that bridges the gap between some current multi-patch isogeometric analysis (IGA) approaches and the more traditional mesh-based interior penalty discontinuous Galerkin (IPDG) method. We briefly discuss the advantages of this unified framework for time-explicit hyperbolic problems, and we present some interesting numerical examples using the acoustic wave equation.

1 Introduction

Standard finite element (FE) analysis for the numerical solution of partial differential equations (PDEs) is usually formulated over a domain $\Omega \subset \mathbb{R}^d$ by introducing an underlying geometric subdivision (a *mesh*) of Ω, a set of polynomial basis functions over each mesh element, and using suitable linear combinations to impose a strong global C^0 regularity over Ω. From here, over time, two seemingly opposing tendencies have arisen.

In discontinuous Galerkin (DG) schemes, continuity is not imposed strongly, making the solution discontinuous on mesh faces. Instead, weak continuity is restored via the imposition of numerical fluxes and penalty terms between elements. This fully unstructured approach offers a great deal of flexibility and combines a good modelization of complex geometries with local h (mesh size), k (polynomial degree) and even t (timestep)-adaptivity, which is especially appreciated in the physical sciences (see e.g. [1] for a recent application). In the case of time-explicit discretization schemes, the block-diagonal nature of the mass matrix also allows to use efficient explicit time discretization schemes. As the degree k of the basis increases, the Courant-Friedrichs-Lewy (CFL) restriction on the timestep scales like $O(h/k^2)$, where h is the spatial discretization step [2].

Going in a seemingly opposite direction, isogeometric analysis (IGA) [3, 4] replaces the standard FE basis by B-splines, i.e., piecewise-polynomial functions of degree k with increased regularity, up to order C^{k-1}. Since these functions (and their rational counterparts, Non-Uniform Rational B-Splines or NURBS) are routinely used in engineering to represent the (exact) geometry of mechanical pieces, IGA

*Corresponding author: stefano.frambati@totalenergies.com

obviates the need to mesh the simulation domain before performing analysis, eliminating the associated potential discrepancy as a source of error. Moreover, IGA has been proven to possess superior numerical properties, including a CFL condition timestep for wave propagation that scales like $O(h/k)$ [5].

Aiming at bridging the chasm between these two worlds, some recent approaches have focused on formulating independent IGA schemes over multiple B-spline patches, which are then coupled through the introduction of DG-like fluxes and penalties. These attempts, which are often named multi-patch DG-IGA schemes, have proven fruitful, with a recent work [5] highlighting how its application to time-domain wave propagation allows to retain the parallelization potential of DG (and its associated block-diagonal mass matrix) with the improved CFL condition typical of IGA.

However, the parameterization of IGA patches suitable for numerical analysis is far from a trivial task, and even ensuring its injectivity requires a careful placement of control points (see, e.g., [6, 7]). This problem is even more relevant in many applications to natural sciences, where the geometry of discontinuities can be complex and have arbitrary topology, while pre-existing CAD models are lacking or nonexistent. Recent advances in unstructured spline theory [8] have provided some clarity on the construction of spaces of simplex (unstructured) spline functions, and the technology seems ripe for a fully-formed numerical simulation scheme based on these functions.

In this work, we present a simple but highly flexible approach to this goal, which reproduces the features of multi-patch DG-IGA approaches but relies on unstructured multivariate splines. The corresponding IGA patches can have arbitrary topology, and are built starting from a simple set of points, without further structure, simplifying considerably their use in inverse problems. Moreover, our basis functions reproduce the usual FE and DG bases as special cases, making it very easy to seamlessly couple all three numerical schemes in a single simulation.

In Section 2 we give an introduction to the main properties of simplex splines and their polynomial-reproducing spaces that we use in this work. In Section 3, we recast some theoretical results of [8] into an explicit form, allowing to formulate some algorithms for the construction of spline spaces and the evaluation of all spline functions supported at a point. In Section 4, we refine this result to allow the definition of spline spaces over a set of subdomains of arbitrary shape, and we show that the spline functions with nonzero trace on subdomain boundaries have an especially simple form. Finally, in Section 5 we show how these spaces can be exploited to formulate a fully unstructured multi-patch DG-IGA scheme with promising numerical properties for time-explicit formulations, which we apply to the propagation of acoustic waves. We present a few numerical results in Section 6, before drawing some conclusions and discussing some avenues for improvement in Section 7.

2 Background

We recall in this section the basic properties of simplex splines and the associated polynomial-reproducing spaces that we will use of in the rest of this work.

2.1 Simplex spline functions

Simplex spline functions were first introduced by Curry and Schoenberg [9]. Let us select a vector of n points in $\mathbb{R}^d$, $A \coloneqq (a_1, \dots, a_n)$, possibly with repetitions. Given a set Q of $k+d+1$ indices between 1 and n, the normalized multivariate spline function $M(x \mid Q)$ can be defined for $x \in \mathbb{R}^d$ via the following recurrence formula, first derived by Micchelli [10],

$$M(x \mid Q) \coloneqq \begin{cases} \dfrac{1}{\operatorname{vol}(\sigma_Q)} \mathbf{1}_{\sigma_Q}(x) & \text{if } |Q| = d+1, \qquad (2.1a) \\ \dfrac{k+d}{k} \displaystyle\sum_{b \in B} \lambda_b(x) M(x \mid Q \setminus \{b\}) & \text{otherwise.} \qquad (2.1b) \end{cases}$$

Here, $\sigma_Q \coloneqq \operatorname{conv}(\{a_q\}_{q \in Q})$ indicates the d-dimensional simplex having as vertices the points indexed by Q, $\mathbf{1}_{\sigma_Q}(x)$ is the indicator function of σ_Q, whose value is one if $x \in \sigma_Q$ and zero otherwise, B is any subset

$B \subseteq Q$ of size $d+1$ such that the simplex $\sigma_B \coloneqq \mathrm{conv}((a_b)_{b\in B})$ satisfies $\mathrm{vol}(\sigma_B) > 0$, and $\{\lambda_b(x)\}_{b\in B}$ are the barycentric coordinates of x with respect to each of the vertices of σ_B.

If the affine rank of the points indexed by Q is less than $d+1$, the spline function is zero almost everywhere. Else, $M(x \mid Q)$ is a multivariate piecewise-polynomial function of $x \in \mathbb{R}^d$ with maximum degree k, regularity C^{k-1} if all sub-vectors of $d+1$ points of $(a_q)_{q\in Q}$ are affinely independent, and with locally lower regularity otherwise. It is well-known that the superposition integral of a continuous function $f(x) : \mathbb{R}^d \mapsto \mathbb{R}$ with a spline $M(x \mid Q)$ can be expressed directly as the following *Dirichlet average* [11, 12],

$$\int_{\mathbb{R}^d} f(x)M(x \mid Q)\,\mathrm{d}x = \frac{1}{(k+d)!}\int_{\Sigma^{k+d}} f\left(\sum_{q\in Q}\lambda_q a_q\right)(\mathrm{d}\lambda_q)_{q\in Q}, \tag{2.2}$$

where $\Sigma^{k+d} \subset \mathbb{R}^{k+d}$ is a $(k+d)$-dimensional simplex spanned by the barycentric coordinates $(\lambda_q)_{q\in Q}$ satisfying $0 \le \lambda_q \le 1$ and $\sum_{q\in Q}\lambda_q = 1$.

We will make use of two additional expressions, also derived in [10]. The first one is the *derivative formula* for a simplex spline function, which determines the first derivatives of a degree-k spline in terms of $d+1$ simplex splines of degree $k-1$,

$$(\nabla M(x \mid Q))_i = (k+d)\sum_{b\in B}\nu_{i,b}M(x \mid Q \setminus \{b\}). \tag{2.3}$$

Here, for each $i = 1,\ldots,d$, $(\nu_{i,b})_{b\in B}$ are the (vector) barycentric coordinates of the i-th unit direction with respect to σ_B, i.e., $(d+1)$ real numbers such that $\sum_{b\in B}\nu_{i,b}(a_b)_j = 1$ if $i = j$ and 0 if $i \neq j$, and satisfying $\sum_{b\in B}\nu_{i,b} = 0$. Another expression that we use in this work is the *knot insertion* formula. If $|Q| \ge d+2$ (i.e., if $k \ge 1$), we can select another index $c \in Q \setminus B$. We then have

$$M(x \mid Q \setminus \{c\}) = \sum_{b\in B}\lambda_b(a_c)M(x \mid Q \setminus \{b\}), \tag{2.4}$$

where again $\lambda_b(a_c)$ represents the usual barycentric coordinate of a_c with respect to the vertex a_b in the simplex σ_B. Notice that (2.4) relates splines of the same degree $k-1$. We show a few multivariate simplex splines in Figure 1.

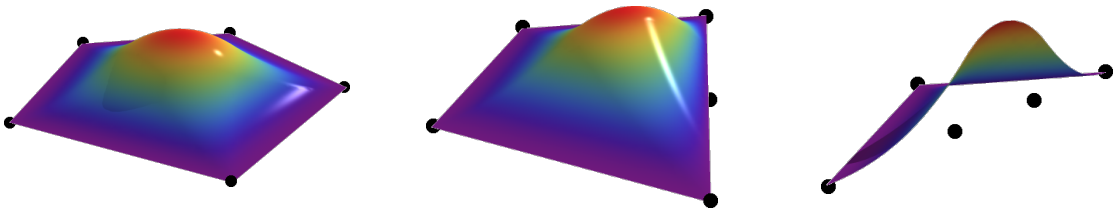

Figure 1: Three examples of bi-variate simplex splines of degree $k = 2$, built on $k+d+1 = 5$ points. As the points become collinear, the regularity of the spline function decreases.

2.2 Polynomial-reproducing spaces of simplex spline functions

Polynomial-reproducing spaces, i.e., spaces of functions containing all the polynomials up to a given degree k in their linear span, are essential for numerical schemes based on the Galerkin formulation. In [13], Neamtu showed that polynomial-reproducing simplex spline spaces can be built using higher-order *Delaunay triangulations* over the point configuration A. Limited to the case of two dimensions, and under the condition that all sub-vectors of $d+1$ points of A are affinely independent, Liu and Snoeyink [14, 15] have also provided an explicit and practical construction algorithm for these spaces.

Recently, these results were generalized in [8], where some combinatorial objects known as *zonotopal tilings* are employed to overcome both the complexities associated to higher dimensions and the

geometrical degeneracy of affinely dependent point configurations. When restricted to Delaunay triangulations (equivalently, to regular fine zonotopal tilings), the results of [8] provide enough material to create some explicit algorithms for the construction of spline spaces and the evaluation of the simplex splines supported on a given location. These algorithms are the object of the next section.

Let $A = (a_1, \dots, a_n)$ be, as above, a point configuration in $\mathbb{R}^d$, possibly including some repeated points, and let $h \coloneqq (h_1, \dots, h_n) \in \mathbb{R}^n$ be any *height vector* over A, associating to the index i the height value h_i. Let $B = \{b_1, \dots, b_{d+1}\}$ be a set of $d+1$ indices between 1 and n, select another index $i \notin B$, $1 \le i \le n$ and consider the following expressions,

$$S(b_1, \dots, b_{d+1}) \coloneqq \operatorname{sign} \begin{vmatrix} a_{b_1} & 1 \\ \vdots & \vdots \\ a_{b_{d+1}} & 1 \end{vmatrix}, \; D(b_1, \dots, b_{d+1}, i) \coloneqq \operatorname{sign} \begin{vmatrix} a_{b_1} & h_{b_1} & 1 \\ \vdots & \vdots & \vdots \\ a_{b_{d+1}} & h_{b_{d+1}} & 1 \\ a_i & h_i & 1 \end{vmatrix} \cdot S(b_1, \dots, b_{d+1}), \quad (2.5)$$

where $\left|\cdot\right|$ denotes the determinant, $\operatorname{sign}(x) \coloneqq +1$ or -1 if $x > 0$ or $x < 0$, respectively, and $\operatorname{sign}(0) \coloneqq 0$. We call a set of indices B such that the points $\{a_{b_1}, \dots, a_{b_{d+1}}\}$ are affinely independent (i.e., $S(b_1, \dots, b_{d+1}) \neq 0$) an *affinely independent set*. Notice that $D(b_1, \dots, b_{d+1}, i)$ does not depend on the order of the indices in B, and thus we will often use the shorthand $D(B, i)$. We adopt the following standard definitions.

Definition 2.1 (Weighted Delaunay condition)**.** A simplex σ_B, whose vertices are indexed by the affinely independent set $B = \{b_1, \dots, b_{d+1}\}$, satisfies the *weighted Delaunay condition* if $D(B, i) < 0$ whenever $i \notin B$.

Definition 2.2 (Generic height vector)**.** A height vector h is called *generic* if $D(B, i) \neq 0$ for every choice of affinely independent set $B = \{b_1, \dots, b_{d+1}\}$ and every choice of index $i \notin B$.

Definition 2.2 is equivalent to [8, Definition 3.2]. If h is generic, then the simplices σ_B satisfying the weighted Delaunay condition form a triangulation of $\operatorname{conv}(A)$, the *weighted Delaunay triangulation* of A. If there is no set of $d+1$ points in A that are co-planar, and no set of $d+2$ points that are co-spherical, then the choice $h_i = \|a_i\|^2$ yields the usual Delaunay triangulation of A (see, e.g., [16]). The above formulation is however valid for all point sets, including the case of repeated points in A.

Many algorithms and tools exist to compute the (weighted) Delaunay triangulation of a set of points, especially in two and three dimensions. In this work, we assume that a routine is available to compute the weighted Delaunay triangulation a set of points *without* repetitions, given a generic height vector h (see, e.g., [17]). However, since repeated points in A play a crucial role in our formulation, we need the following additional definition of a *fine-grained* height vector.

Definition 2.3 (Fine-grained height vector)**.** A height vector h is called *fine-grained* if, for every affinely independent set $B = \{b_1, \dots, b_{d+1}\}$, and any index $j \notin B$, either $a_j = a_b$ for some $b \in B$, or $D(B, i)$ has the same sign for all indices i such that $a_i = a_j$.

In other words, with a fine-grained height vector, repeated points in A have very close height values, so much so that the corresponding signs in (2.5) can only be distinguished by independent sets B that use one of the copies. In practice, this property is easy to achieve using symbolic perturbation, as shown in the following proposition.

Proposition 2.4. *Let U be the set indexing the first appearance of each point in A, and let h be a generic height vector over the (unique) points of A indexed by U. Let ε be a real number smaller than the smallest positive difference between these height values. For each repeated point of A, indexed by $i \notin U$ and such that $a_i = a_u$ for some $u \in U$, define $h(a_i) \coloneqq h(a_u) + (i-u)/|A|\,\varepsilon$. Then, the resulting height vector on A is both generic and fine-grained.*

Proof. The definition of h is equivalent to the application of the following perturbation rule whenever $a_i = a_j$,

$$\operatorname{sign}(h_i - h_j) = \operatorname{sign}(i - j), \quad (2.6)$$

with the understanding that, for any index k such that $a_k \neq a_i$, we have $\text{sign}(h_j - h_k) = \text{sign}(h_i - h_k)$. This shows that h is fine-grained. Let now B be an affinely independent set of indices. If a point a_i indexed by $i \notin B$ coincides with a point a_b for some $b \in B$, then one can subtract the row containing a_b from the last row in the first determinant defining $D(B, i)$ in (2.5), obtaining a row of zeros except for the $(d+2)$-th column, whose value is simply $h_i - h_b$. Expanding the determinant in this row and exploiting (2.6), as well as the fact that B is an affinely independent index set, shows that the determinant cannot be zero, and thus that h is generic. □

In practical terms, the height vector of Proposition 2.4 can simply be obtained by assigning the same height value to all the copies of a point in A, and comparing the point indices instead of the height values when the comparison is performed on two coincident points. Given (2.6), the expression (2.5) for $D(B, i)$ simplifies significantly in the case of repeated points,

$$D(b_1, \ldots, b_{d+1}, i) = \text{sign}(b_j - i) \text{ when } a_i = a_{b_j} \text{ for some } 1 \leq j \leq d+1.$$

In the following, we always assume that the height vector h is both generic and fine-grained.

Given a generic height vector h, the polynomial-reproducing simplex splines spaces of degree k, associated to order-k Delaunay configurations [8, 13–15], can be easily characterized as follows,

$$\mathcal{S}^{(k)} \coloneqq \{M(\cdot \mid I \sqcup B) : |I| = k, D(B, i) > 0 \text{ (respectively, } < 0) \text{ if and only if } i \in I \text{ (resp. } i \notin I \sqcup B)\}, \tag{2.7}$$

where we denote by $\sqcup$ the disjoint union of two sets, i.e., the union of sets with empty intersection. In the following, we refer I as the set of *interior knots* of M, and to B as the set of *boundary knots* of M.

3 Algorithms for simplex spline spaces

In this section, we review some theoretical results of [8] in order to formulate a set of explicit, comprehensive algorithms for the construction and evaluation of a space of polynomial-reproducing simplex splines over $\text{conv}(A)$.

3.1 Simplex spline space construction

A first algorithm to construct a polynomial-reproducing simplex spline space over $\text{conv}(A)$, valid in the case $d = 2$ and without repeated or affinely independent points, was provided by Liu and Snoeyink [14, 15] and proved to be valid at all degrees k by Schmitt [18]. This approach relies on the construction and subsequent triangulation, without added points, of a series of two-dimensional (non-convex) polygonal regions, called *vertex links*. For $d \geq 3$, however, there exist polytopal regions that cannot be triangulated in this sense, such as for example Schönhardt's polyhedron [19], hindering the extension of this approach to dimension 3 and above.

This result is extended in [8], where it is proved that, if the same weighted Delaunay triangulation is used at every step, and subject to a slight redefinition of vertex links [8, Definition 2.7], the triangulability of every link region is guaranteed, and the algorithm can be used successfully for all space dimensions, all spline degrees, and all point configurations. Furthermore, if the vertices located on the boundary of $\text{conv}(A)$ are repeated at least $k+1$ in A, the resulting simplex spline space reproduces all polynomials over $\text{conv}(A)$ [8, Corollary 2.4], similarly to the behavior of clamped (or open) knot sets in standard B-spline bases.

We extract from these results a simple, practical algorithm for the construction of a polynomial-reproducing spline space $\mathcal{S}^{(r)}$, for $r = 0, \ldots, k$, over any point configuration A, which we present in Algorithm 1. We apply here a few additional requirements to the process of [8, Theorem 2.10] and [8, Theorem 3.3]. In particular, we assume that all the points lying on $\text{conv}(A)$ are repeated $k+1$ times in A, so that polynomials are reproduced on the whole convex hull $\text{conv}(A)$, and furthermore we assume that repeated points appear consecutively in A, in order to simplify our point indexing conventions. Finally, we apply the perturbation rule (2.6). Since standard Delaunay triangulation algorithms do not always allow symbolically-perturbed height vectors, we explicitly introduce this feature in the algorithm, and only rely on (weighted) Delaunay triangulations of sets of unique points.

Algorithm 1 builds polynomial-reproducing simplex spline spaces of degree up to k over a point configuration.

Input:

A any point configuration in $\mathbb{R}^d$;
h a generic height vector;
k maximum spline space degree to be computed.

Output:

$\mathcal{S}^{(r)}$ set of simplex spline functions spanning the polynomial-reproducing space of degree r for all $0 \leq r \leq k$.

Assumptions: Points lying on the boundary of $\text{conv}(A)$ are repeated at least $k+1$ times in A. All the copies of a given point are consecutive in A.

Auxilary procedures: DELAUNAY(P, h) computes the weighted Delaunay triangulation of the points P with height vector h, MULT(a, A) returns the number of points in A that are equal to a, FIRSTINDEX(a, A) returns the index in A of the first copy of a in A.

procedure BUILDSPLINESPACE(A, h, k)
  $\mathcal{S}^{(r)} \leftarrow \varnothing$, $\mathcal{Q}_r \leftarrow \varnothing$, for $r = 0, \ldots, k$.
  $\mathcal{D} \leftarrow$DELAUNAY(unique points in A, h)
  for all simplices σ in $\mathcal{D}$ **do**
    $B \leftarrow \{$FIRSTINDEX(a, A): a vertex of $\sigma\}$
    Add $M(\cdot \mid \varnothing \sqcup B)$ to $\mathcal{S}_0$
  for all unique point a in A **do**
    Add $\{$FIRSTINDEX(a, A)$\}$, i.e., the singleton set containing just FIRSTINDEX(a, A), to $\mathcal{Q}_1$.
  for $r = 1, \ldots, k$ **do**
    for all sets of indices $Q \in \mathcal{Q}_r$ **do**
      $\mathcal{M}_0 \leftarrow \{M(\cdot \mid \varnothing \sqcup B) \in \mathcal{S}_0 : B \cap Q \neq \varnothing\}$
      $\mathcal{R}_0 \leftarrow$ geometric union of all simplices σ_B associated to splines $M(\cdot \mid \varnothing \sqcup B) \in \mathcal{M}_0$
      $\mathcal{M}_1 \leftarrow \{M(\cdot \mid I \sqcup B) \in \mathcal{S}^{(q)} : 1 \leq q < r, I \subset Q, B \cap Q = \varnothing\}$
      $\mathcal{R}_1 \leftarrow$ geometric union of all simplices σ_B associated to splines $M(\cdot \mid I \sqcup B) \in \mathcal{M}_1$
      $\mathcal{R}_Q \leftarrow$ geometric difference $\mathcal{R}_0 \setminus \mathcal{R}_1$
      if $\mathcal{R}_Q$ is not empty **then**
        $P_0 \leftarrow$ points on the boundary of $\mathcal{R}_Q$
        $P \leftarrow P_0 \sqcup \{a_i : i \in Q,$ MULT(a_i, A)$>$MULT(a_i, $(a_q)_{q \in Q}$)$\}$
        $\mathcal{D}_Q \leftarrow$DELAUNAY(unique points in P, h)
        for all simplices $\sigma \in \mathcal{D}_Q$ lying inside $\mathcal{R}_Q$ **do**
          $B \leftarrow \{$FIRSTINDEX(a, A)+MULT(a, $(a_q)_{q \in Q}$): a vertex of $\sigma\}$
          Add $M(\cdot \mid Q \sqcup B)$ to $\mathcal{S}^{(r)}$
          for all index $b \in B$ **do** add $Q \sqcup \{b\}$ to $\mathcal{Q}_{r+1}$ if not already present
  return $\{\mathcal{S}^{(0)}, \ldots, \mathcal{S}^{(k)}\}$.

Notice that, in Algorithm 1, the calculation of the region $\mathcal{R}_Q$ associated to a set of indices Q only involves simplices located near the points $(a_q)_{q \in Q}$, although the number of such simplices increases with the degree of the space. Also, notice that, as for Liu and Snoeyink's original approach, Algorithm 1 needs to perform an initial Delaunay triangulation over all (unique) points in A, followed by the triangulation of each region $\mathcal{R}_Q$. The number of these triangulations is bounded by the total number of spline functions in the result. Even though the computation of Delaunay triangulations over n points has a worst-case complexity of $O(n^2)$, aside from the first triangulation of A, all other triangulations only involve a very limited number of points.

Notice also that each spline $M(\cdot \mid I \sqcup B)$ produced by the algorithm, for all degrees $1 \leq r \leq k$, satisfies the weighted Delaunay condition (2.7), i.e., $D(B, i) < 0$ for all $i \notin I \sqcup B$ and $D(B, i) > 0$ for all $i \in I$, even when $a_i = a_b$ for some $i \in I$ and $b \in B$. This is due to the symbolic perturbation rule (2.6), which is implicitly enforced by the indexing rules chosen in Algorithm 1.

Finally, notice that, in addition to points on $\text{conv}(A)$, nothing prevents repeating some internal points

of A as well, with the consequence of locally reducing the regularity of the spline space. We exploit this property in Section 4 to show an interesting special case of this construction. In all cases, when $d = 1$ and h is the usual choice for Delaunay weights, i.e., $h_i \coloneqq \|a_i\|^2$, Algorithm 1 reproduces the usual B-spline basis [20] over the interval $\mathrm{conv}(A)$.

A graphical depiction of the Algorithm 1 is given in Figure 2. In the figure, we construct some spline functions over a set of points A in two dimensions, using the standard Delaunay height vector $h_i \coloneqq \|a_i\|^2$. The point a_1 is only represented once in A, while the two points a_2 and a_3 are coincident. Panel (a) shows the triangles of the Delaunay triangulation of A, corresponding to splines of degree $k = 0$ via (2.1b). When constructing the splines of degree $k = 1$, one has to compute the spline functions associated to the point a_1 ($Q = \{1\}$ in Algorithm 1). The corresponding region $\mathcal{R}_0$ consists of the union of all the triangles touching $\{a_1\}$, and is shown in panel (b). Notice that, in this case, $\mathcal{R}_1 = \varnothing$. Triangulating the region $\mathcal{R}_{\{1\}} = \mathcal{R}_0 \setminus \mathcal{R}_1 = \mathcal{R}_0$, as shown in panel (c), yields three triangles. Each of these is associated to a piecewise-linear spline, defined, via (2.1a) and (2.1b), by the union of the three vertices of the triangle with the point a_1. Each of these splines is added to $\mathcal{S}^{(1)}$. Notice that, for the repeated vertex $a \coloneqq a_2 = a_3$, one chooses to include the vertex indexed by $\textsc{FirstIndex}(a, A)+\textsc{Mult}(a, \{a_1\}) = 2 + 0 = 2$, i.e., a_2. For each triangle vertex b, the candidate set $Q = \{a_1, b\}$ is added to $\mathcal{Q}_2$, and is later used to construct splines of degree 2.

When evaluating the spline functions associated to the point a_2 ($Q = \{2\}$), one can easily take into account the fact that a_2 is repeated twice in A. Algorithm 1 prescribes in this case that the region $\mathcal{R}_0$, shown in panel (d), must be triangulated including a_2 as a vertex, as shown in panel (e). The ambiguity on the indexing of $a \coloneqq a_2 = a_3$ is once again resolved, since $\textsc{FirstIndex}(a, A)+\textsc{Mult}(a, \{a_2\}) = 2+1 = 3$, and therefore the index 2 is never used more than once.

The candidate set $Q = \{a_1, a_2\}$ produced in panel (c) must be processed when constructing the spline functions of degree $k = 2$. The corresponding regions $\mathcal{R}_0$ and $\mathcal{R}_1$ are shown in panels (f) and (g), respectively. Triangulating $\mathcal{R}_0 \setminus \mathcal{R}_1$ yields two triangles, as shown in panel (h), each yielding a quadratic spline defined by the union of its three vertices and the points a_1 and a_2. Ambiguities on repeated points are resolved by Algorithm 1, analogously as the previous case.

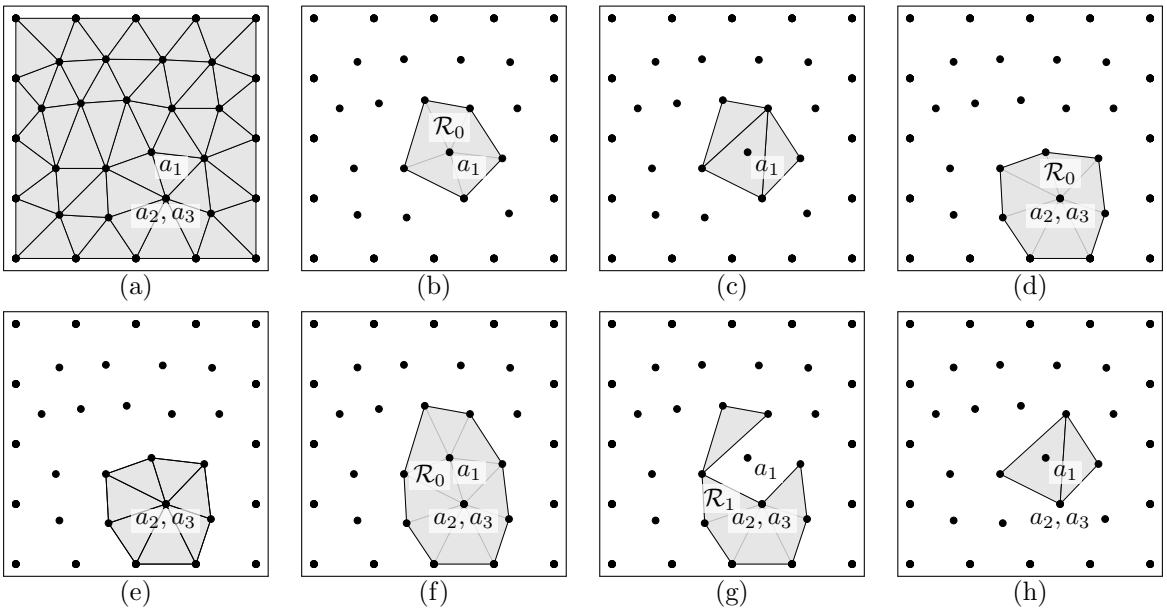

Figure 2: A set of points in two dimensions and a graphical depiction of some steps of Algorithm 1. Notice that $a_2 = a_3$. The description of these steps is given at the end of Section 3.1. (a) Delaunay triangles, representing splines of degree $k = 0$. (b) The region $\mathcal{R}_0$ corresponding to $Q = \{1\}$, and its triangulation (c). The corresponding region (d), and its triangulation (e) in the case of a repeated point, $Q = \{2\}$. (f) The region $\mathcal{R}_0$, the region $\mathcal{R}_1$ (g) and the triangulation of $\mathcal{R}_{\{a_1,a_2\}} = \mathcal{R}_0 \setminus \mathcal{R}_1$ (h), yielding the splines of degree $k = 2$, corresponding to $Q = \{a_1, a_2\}$.

3.2 Simplex spline evaluation

One of the best practical aspects of the usual one-dimensional B-spline basis is the availability of an efficient algorithm capable of computing the value of all the spline functions in the basis supported at a given point $x \in \mathbb{R}$, optimally reusing intermediate results [20, Chapter X, Algorithm 8]. For this task, limited to $d = 1$, only the spline functions $\mathcal{S}^{(r)}$, $r = 0, \ldots, k-1$ computed in Algorithm 1 are needed. When $d > 1$, these functions are only sufficient to determine which spline functions of $\mathcal{S}^{(k)}$ are nonzero on $x \in \mathbb{R}^d$, following [8, Proposition 3.7], but the evaluation of these splines at a given $x \in \mathbb{R}^d$ requires the introduction of auxiliary functions.

In practical applications, determining which functions are supported at a point can be performed efficiently using suitable bounding volume hierarchies (BVH, see, e.g., [21]). Thus, we do not focus on this task here. Instead, we reformulate in algorithmic form a combinatorial strategy presented in [8, Section 3.3] which can be used to construct a suitable set of auxiliary functions for the evaluation of all spline functions in $\mathcal{S}^{(k)}$, and their first derivatives, at a given location $x \in \mathbb{R}^d$. This can be achieved via the following steps.

First, one builds and stores a *base oriented evaluation graph* $\mathcal{G}$, whose vertices contain all the splines in $\mathcal{S}^{(k)}$ as well as a set of auxiliary spline functions, and where each spline $M(\cdot \mid I \sqcup B)$ of degree $k \geq 1$ has at most $2(d+1)$ incoming edges, two for each $b \in B$.

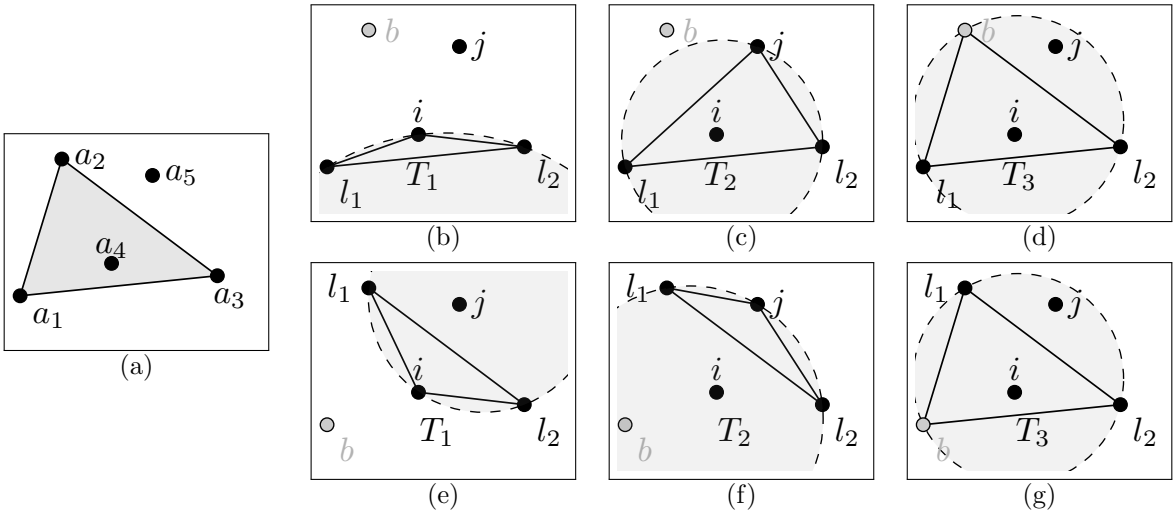

Figure 3: Some steps in the construction of some auxiliary functions for a spline of degree $k = 2$ in two dimensions via Algorithm 2. The spline is described by the five points $\{a_1, \ldots, a_5\}$ (a), and some auxiliary functions obtained by removing the vertex a_2 and a_1 are shown in panels (b–d) and (e–g), respectvely. A description of these steps is given in Section 3.2.

Once this structure has been computed, given a location $x \in \mathbb{R}^d$, one selects a suitable subgraph $\mathcal{G}_x$ of $\mathcal{G}$ containing no cycles, i.e., a directed acyclic graph (DAG). The splines of degree 0 have no incoming vertices in $\mathcal{G}_x$, and therefore can be used as starting points for the evaluation process. Using an efficient search structure, such as for example an $R^\star$-tree [21, 22] the splines of degree 0 supported at x are determined, and evaluated directly via (2.1a). All other degree zero splines are given a value of 0. The corresponding nodes in $\mathcal{G}_x$ are marked as visited. Then, one iteratively follows the topological ordering of $\mathcal{G}_x$, using for example Kahn's algorithm [23], until all the reachable nodes have been visited.

At each node, one can evaluate the corresponding spline function $M(\cdot \mid I \sqcup B)$ of degree $k \geq 1$ once all its incoming edges have been visited. In particular, each visited edge can be used to compute one of the $d+1$ values $M(x \mid I \sqcup B \setminus \{b\})$, $b \in B$. After all the incoming contributions have been evaluated, one can use (2.1b) to evaluate $M(x \mid I \sqcup B)$ itself, and (2.3) to evaluate its first derivatives, after which the corresponding node in $\mathcal{G}_x$ is marked as visited, and the process continues until all the active edges have been visited. The unevaluated spline functions are given a value of zero.

Algorithm 2 builds the auxiliary functions and base oriented evaluation graph.

Input:

A generic point configuration, as in Algorithm 1;

h a generic height vector.

Output:

$\mathcal{G}$ the base evaluation graph

Auxilary procedures: PUSH($\mathcal{Q}$, M) inserts M at the end of the queue $\mathcal{Q}$, POP($\mathcal{Q}$) removes and returns the element at the end of the queue $\mathcal{Q}$.

procedure BUILDAUXFUNCTIONS(A, h, $\mathcal{S}^{(k)}$)
  $\mathcal{G} \leftarrow \varnothing$, $\mathcal{Q} \leftarrow \varnothing$
  for all $M(\,\cdot \mid I \sqcup B) \in \mathcal{S}^{(k)}$ **do**
    if $M(\,\cdot \mid I \sqcup B)$ is not a vertex of $\mathcal{G}$ **then**
      Add $M(\,\cdot \mid I \sqcup B)$ as a vertex of $\mathcal{G}$
      PUSH($\mathcal{Q}$, $M(\,\cdot \mid I \sqcup B)$)
      while $\mathcal{Q}$ is not empty **do**
        $M(\,\cdot \mid I \sqcup B) \leftarrow$POP($\mathcal{Q}$)
        for all $b \in B$ **do**
          $L \leftarrow B \setminus \{b\}$, and choose a fixed ordering $L = (l_1, \ldots, l_d)$
          $\mathcal{C}_0 \leftarrow \{(l_1, \ldots, l_d, i) : i \in I\} \sqcup \{(l_1, \ldots, l_d, b)\}$
          $\mathcal{C}_1 \leftarrow \{C \in \mathcal{C}_0 : \mathrm{vol}(\sigma_C) > 0\}$
          Sort $\mathcal{C}_1$ using the total order defined by the following rule,

$$(l_1, \ldots, l_d, i) < (l_1, \ldots, l_d, j) \Leftrightarrow D(l_1, \ldots, l_d, i, j) \cdot S(l_1, \ldots, l_d, j) < 0. \tag{3.1}$$

          $\mathcal{N} \leftarrow$ elements immediately before and immediately after $(l_1, \ldots, l_d, b)$ in $\mathcal{C}_1$, if any
          for $B' \in \mathcal{N}$ **do** ▷ *Note: $\mathcal{N}$ may contain two, one or zero elements.*
            if $D(B', b) > 0$ **then**
              $X \leftarrow I \sqcup B$
            else
              $X \leftarrow (I \sqcup B) \setminus \{b\}$
            $I' \leftarrow X \setminus B'$
            if $M(\,\cdot \mid I' \sqcup B')$ is not a vertex of $\mathcal{G}$ **then**
              Add $M(\,\cdot \mid I' \sqcup B')$ as a vertex of $\mathcal{G}$
              PUSH($\mathcal{Q}$, $M(\,\cdot \mid I' \sqcup B')$)
            Add an edge from $M(\,\cdot \mid I' \sqcup B')$ to $M(\,\cdot \mid I \sqcup B)$ in $\mathcal{G}$
  return $\mathcal{G}$

For each spline $M(x \mid I \sqcup B)$ and each index $b \in B$, the corresponding incoming edge in $\mathcal{G}_x$ connects $M(x \mid I \sqcup B)$ to another spline $M(x \mid I' \sqcup B')$ satisfying $|B \cap B'| = d$ and either $I' \sqcup B' = I \sqcup B \setminus \{b\}$ or $I' \sqcup B' = I \sqcup B$. In the first case, the knowledge of $M(x \mid I' \sqcup B')$ yields directly $M(x \mid I \sqcup B \setminus \{b\})$. In the second case, the values of $M(x \mid I' \sqcup B') = M(x \mid I \sqcup B)$ and $M(x \mid I' \sqcup B' \setminus \{b'\})$ for all $b' \in B'$ are known, since the corresponding node in $\mathcal{G}_x$ has already been visited. These values can then be used to compute $M(x \mid I \sqcup B \setminus \{b\}) = M(x \mid I' \sqcup B' \setminus \{b\})$ through a single application of (2.4). If no incoming edge is present in $\mathcal{G}_x$ for some $b \in B$, then the corresponding value of $M(x \mid I \sqcup B \setminus \{b\})$ is simply set to zero.

We give in Algorithm 2 a step-by-step construction of the base graph $\mathcal{G}$, and we illustrate in Figure 3 some steps of the construction of auxiliary functions for a two-dimensional spline $M_1 \coloneqq M(\,\cdot \mid I \sqcup B)$ of degree $k = 2$ defined over the five points $\{a_1, \ldots, a_5\}$, depicted in panel (a). The boundary and interior knot indices are, respectively, $B = \{1, 2, 3\}$ and $I = \{4, 5\}$. Algorithm 2 prescribes that one of the points indexed by $b \in B$ is selected. Let us select $b = 2$, and let us denote the other indices in B by l_1 and l_2, respectively, and the indices in I by i and j. The simplices T_1, T_2 and T_3, defined by points indexed by $\{l_1, l_2, i\}$, $\{l_1, l_2, j\}$ and $\{l_1, l_2, b\}$ respectively, are shown in panels (b), (c) and (d).

Algorithm 3 evaluates all the spline functions supported at a given location x.

Input:

A generic point configuration, as in Algorithm 1;
x a generic point in $\mathbb{R}^d$;
$\mathcal{G}$ the base evaluation graph computed by Algorithm 2.

Output: The values of $M(x \mid I \sqcup B)$ and $M(x \mid I \sqcup B \setminus \{b\})$, $b \in B$, for all splines $M(\cdot \mid I \sqcup B) \in \mathcal{S}^{(k)}$

Auxilary procedures: FindSupported(x, $\mathcal{G}$, A) returns the splines $M(\cdot \mid \varnothing \sqcup B)$ of degree 0 in $\mathcal{G}$ such that $x \in \sigma_B$; Push and Pop defined as in Algorithm 2.

procedure EvaluateSplineFunctions(A, h, x, $\mathcal{G}$)
  $\mathcal{Q} \leftarrow \varnothing$, $\mathcal{G}_x \leftarrow \varnothing$ ▷ *Determine the active edges, building the acyclic graph $\mathcal{G}_x$.*
  $\mathcal{Z} \leftarrow$FindSupported(x, $\mathcal{G}$)
  for $M(\cdot \mid \varnothing \sqcup B) \in \mathcal{Z}$ **do** Push($\mathcal{Q}$, $M(\cdot \mid \varnothing \sqcup B)$)
  while $\mathcal{Q}$ is not empty **do**
    $M(\cdot \mid I \sqcup B) \leftarrow$Pop($\mathcal{Q}$)
    add $M(\cdot \mid I \sqcup B)$ as a vertex in $\mathcal{G}_x$
    for all $M(\cdot \mid I' \sqcup B')$ such that there is an edge e from $M(\cdot \mid I \sqcup B)$ to $M(\cdot \mid I' \sqcup B')$ in $\mathcal{G}$ **do**
      $L \leftarrow B \cap B'$
      $\{b'\} \leftarrow B' \setminus L$
      if x and $a_{b'}$ are on the same side of $\mathrm{conv}((a_l)_{l \in L})$ **then**
        add edge e to $\mathcal{G}_x$
        if $M(\cdot \mid I' \sqcup B')$ is not a vertex in $\mathcal{G}_x$ **then** Push($\mathcal{Q}$, $M(\cdot \mid I' \sqcup B')$)
  Evaluate $M(x \mid \varnothing \sqcup B)$ for all splines in $\mathcal{Z}$ via (2.1a)
  for $M(\cdot \mid \varnothing \sqcup B) \in \mathcal{Z}$ **do** Push($\mathcal{Q}$, $M(\cdot \mid \varnothing \sqcup B)$)
  while $\mathcal{Q}$ is not empty **do** ▷ *Follow the topological ordering of $\mathcal{G}_x$ to evaluate splines.*
    $M(\cdot \mid I \sqcup B) \leftarrow$Pop($\mathcal{Q}$)
    for all $M(\cdot \mid I' \sqcup B')$ such that there is an edge e from $M(\cdot \mid I \sqcup B)$ to $M(\cdot \mid I' \sqcup B')$ in $\mathcal{G}_x$ **do**
      $\{b'\} \leftarrow B' \setminus B$
      if $(I \sqcup B) \subset (I' \sqcup B')$ **then**
        $M(x \mid I' \sqcup B' \setminus \{b'\}) \leftarrow M(x \mid I \sqcup B)$
      else if $I \sqcup B = I' \sqcup B'$ **then**
        compute $M(x \mid I' \sqcup B' \setminus \{b'\})$ via (2.4), using $M(x \mid I \sqcup B)$ and $M(x \mid I \sqcup B \setminus \{b\})$, $b \in B$
      mark the edge e as visited
      if $M(\cdot \mid I' \sqcup B')$ has no more unvisited incoming edges **then**
        compute $M(\cdot \mid I' \sqcup B')$ via (2.1b)
        Push($\mathcal{Q}$, $M(\cdot \mid I' \sqcup B')$)
  return the computed values of visited splines $M(x \mid I \sqcup B) \in \mathcal{S}^{(k)}$, zero for all other spline functions in $\mathcal{S}^{(k)}$.

According to Algorithm 2, these three simplices must be sorted following the criterion (3.1), and the simplices not directly adjacent to T_3 must be discarded. In this case, one finds the ordering $T_1 < T_2 < T_3$, and thus only T_2 is kept. For the standard Delaunay height vector, this can be determined visually using circumscribed circles. In fact, both a_4 and a_5 lie inside the circle circumscribed to T_3, a_4 (but not a_1) lies inside the circle circumscribed to T_2, and neither a_5 nor a_1 lie inside the circle circumscribed to T_1. This inclusion hierarchy yields the ordering of the triangles. After defining $B' \coloneqq \{l_1, l_2, j\} = \{1, 3, 5\}$, one can easily verify that $D(B', b) < 0$. In fact, with the standard choice of Delaunay height vector, this corresponds to the condition that b lies outside the circle circumscribed to T_2, i.e., the standard Delaunay condition. Thus, according to Algorithm 2, a new spline $M_2 = M(\cdot \mid I' \sqcup B')$ is created with boundary and interior knots indexed by B' and $I' = (I \sqcup B \setminus \{b\}) \setminus B' = \{l_1, l_2, i, j\}$. Notice that M_2 is of degree $k = 1$. Both M_1 and M_2 are added to the graph $\mathcal{G}$ as vertices, if not already present, and an oriented

edge is added from M_2 to M_1.

The same procedure is then applied to another boundary knot vector of M_1, as shown in panels (e), (f) and (g). In this case, $\{b, l_1, l_2, i, j\} = \{1, 2, 3, 4, 5\}$, and three simplices T_1, T_2 and T_3 are created similarly to the previous case. Since the criterion (3.1) yields the ordering $T_1 < T_3 < T_2$, both T_1 and T_2 are kept. Given that $D(2, 3, 4, 1) < 0$ and $D(2, 3, 5, 1) > 0$, we must add to $\mathcal{G}$ the spline functions $M_3 \coloneqq M(\cdot \mid I'' \sqcup B'')$ and $M_4 \coloneqq M(\cdot \mid I''' \sqcup B''')$ defined by $B'' = \{2, 3, 4\}$, $I'' = \{5\}$, $B''' = \{2, 3, 5\}$ and $I''' = \{1, 4\}$. Notice that M_3 has degree $k = 1$, but M_4 has degree $k = 2$. The corresponding edges are added to $\mathcal{G}$. The process of Algorithm 2 continues recursively, until splines of degree $k = 0$ are reached, at which point the process stops.

We detail in Algorithm 3 the spline evaluation process, based on $\mathcal{G}$, and we present an example of its application in Figure 4. Here, we illustrate some steps in the evaluation of two spline functions of degree $k = 2$ in two dimensions, defined over the sets of 5 knots shown in panel (a). The boundary and interior knot indices (B, I) for the two splines are given by $(\{1, 2, 3\}, \{4, 5\})$ and $(\{2, 5, 6\}, \{1, 4\})$. Panel (b) shows the splines of degree $k = 0$ (each corresponding to a triangle) produced by Algorithm 2, and inserted into the base graph $\mathcal{G}$, while the full base graph $\mathcal{G}$ itself is shown in panel (d). Here, each spline function is identified by the couple (B, I) of boundary and interior knot indices, respectively. Empty arrowheads represent connections between splines of the same degree, and filled arrowheads represent connections between splines whose degrees differ by one.

When a point $x \in \mathbb{R}^2$ is selected, the evaluation via Algorithm 3 starts by choosing all the splines of degree $k = 0$ in $\mathcal{G}$ that are supported at x, as shown in panel (c). The subgraph $\mathcal{G}_x$ of $\mathcal{G}$ built by the first part of Algorithm 3 is shown in panel (e). Notice that $\mathcal{G}_x$ is indeed acyclic, and can therefore be topologically sorted. Starting from the two splines of degree $k = 0$ supported at x, namely $(\{1, 2, 4\}, \varnothing)$ and $(\{1, 4, 5\}, \varnothing)$, one is able to compute the values appearing on the right hand side of (2.1a), either directly (full arrowheads) or applying 2.4 (empty arrowheads). When all incoming arrows have been computed for a given spline, a simple application of (2.1b) yields the value of the spline itself at x. Notice that in $\mathcal{G}_x$ each spline has at most $(d + 1)$ incoming edges. Splines of degree zero have none, since in this case, $\mathcal{N} = \varnothing$ in Algorithm 2. Notice also that, in general, the set of splines of degree zero produced by Algorithm 2 does not form a triangulation, since the corresponding simplices may overlap.

Together, Algorithm 1, Algorithm 2 and Algorithm 3 allow to construct, given an arbitrary point configuration A, a polynomial-reproducing spline space over $\operatorname{conv}(A)$, and to compute the numerical value of all the spline functions that are nonzero at a given location x. In the following section, we will discuss how one can construct a multi-patch spline space over a domain decomposition of $\Omega \subseteq \operatorname{conv}(A)$.

4 Multi-patch spline spaces

In the previous section, we have reformulated algorithmically the processes described in [8] in order to build and evaluate a polynomial-reproducing spline space over a general point configuration A of points in $\mathbb{R}^d$. In particular, if the points lying directly on the convex hull of A are repeated $k + 1$ times, then the spline space of degree k reproduces all the polynomials up to degree k over the whole convex hull $\operatorname{conv}(A)$, including therefore its boundary $\partial(\operatorname{conv}(A))$. If the domain Ω is convex, introducing a point cloud A such that $\Omega = \operatorname{conv}(A)$ is therefore sufficient, and allows the imposition of boundary conditions on Ω.

In this section, we wish to extend this construction to more general, non-convex multi-patch spline spaces. To this end, we subdivide the (open) simulation domain $\Omega \subset \mathbb{R}^d$ into a finite set of open n_d subdomains $(\Omega_i)_{i=1}^{n_d}$, and we construct a set of compatible, polynomial-reproducing spline spaces over each subdomain Ω_i. In general, the subdomains are not assumed to have a particular shape, nor even to be convex. In practical applications, we wish to be able to build complex domains whose shape follows the irregularities of the problem geometry. We do not even assume that the domains are simply connected, and we allow the presence of internal boundaries. However, we still assume that the domains are *polytopal* (e.g., polyhedral in three dimensions and polygonal in two dimensions), and that they form

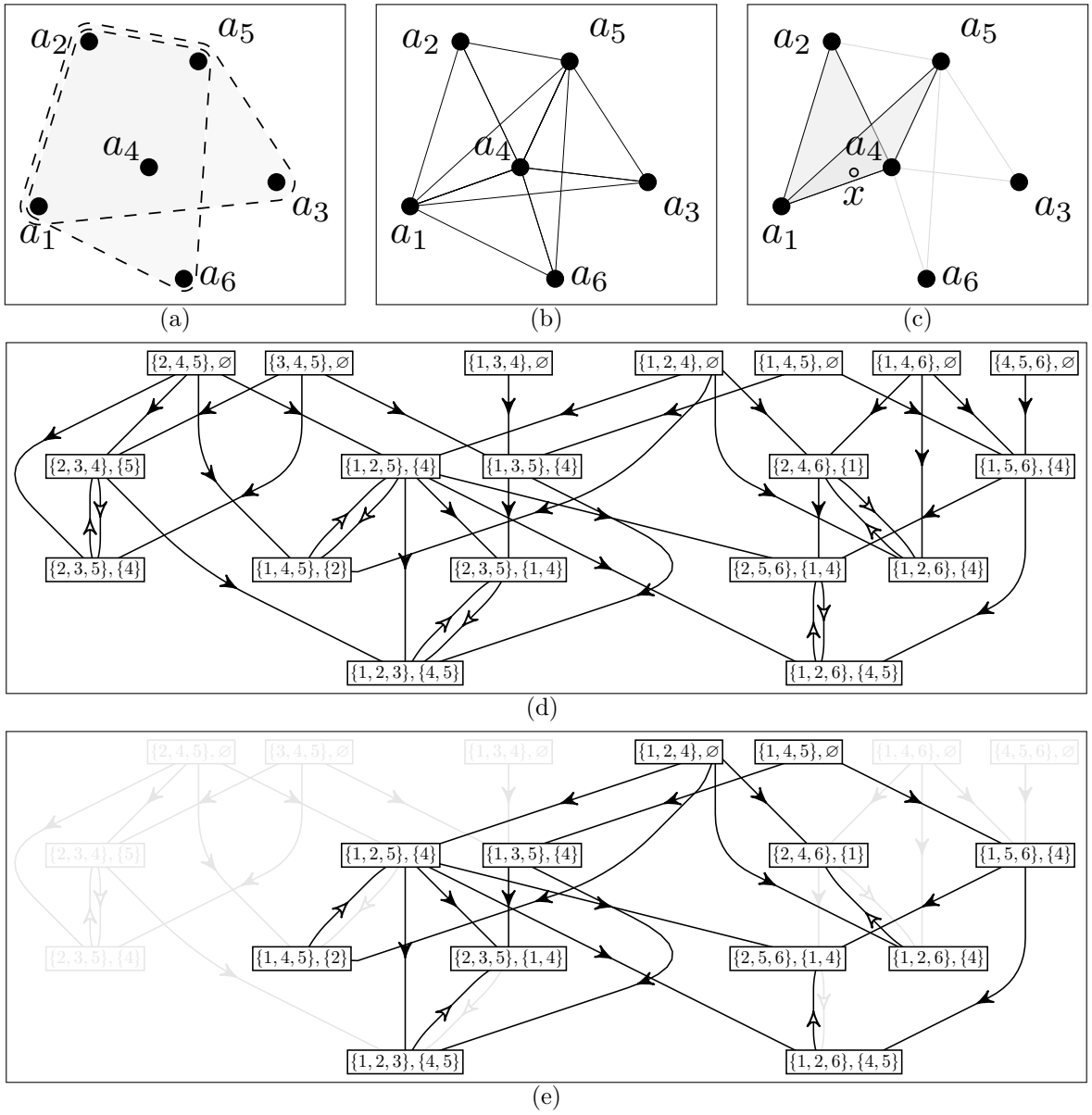

Figure 4: Evaluation process for two spline functions in two dimensions, according to Algorithm 3. (a) The two sets of 5 knots defining the two splines of degree $k = 2$. (b) The splines of degree $k = 0$ (each corresponding to a triangle) produced by Algorithm 2, and the corresponding full base graph $\mathcal{G}$ (d). (c) The spline functions of degree $k = 0$ in $\mathcal{G}$ that are supported at a point x, and the given evaluation acyclic graph $\mathcal{G}_x$ (e). In the graphs, a spline with boundary and interior knots indices B and I, respectively, is represented by the ordered couple B, I. A description of these steps is given in Section 3.2.

a proper *subdivision* of Ω, i.e., denoting by $\overline{\cdot}$ the closure of a set,

$$\Omega_i \cap \Omega_j = \varnothing \text{ for } i \neq j, \ \bigcup_{i=1}^{n_d} \overline{\Omega}_i = \overline{\Omega}.$$

We denote by $\mathcal{F}$ the set of all internal and external boundary facets of the compatible domains Ω_i, i.e.,

$$\mathcal{F} \coloneqq \bigcup_{i=1}^{n_d} \{F : F \in \partial\Omega_i\},$$

Given that we are not assuming that the domains Ω_i are convex, the algorithms proposed in the last section need some specialization, since one cannot simply introduce a point cloud in each domain and build a spline space on its convex hull. We therefore introduce in this section the assumptions on the domain boundaries and on the height vector that are required in order to produce a suitable space of multi-patch splines.

4.1 Gabriel property

We wish to create polynomial-reproducing spline spaces that follow the constrained facets $\mathcal{F}$ delimiting our subdomains. One potential roadblock lies with the fact that the construction Algorithm 1 is based on successive weighted Delaunay triangulation steps. In order for the corresponding spline spaces to follow the domain decomposition, the faces $F \in \mathcal{F}$ lying on the domain boundaries need to be included in the triangulation. The issue of building a Delaunay triangulation that respects a given set of constraints (constrained Delaunay triangulation, or CDT) is notoriously hard [24], and even determining whether a domain is triangulable or not is in general NP-hard (non polynomial hardness, see, e.g., [25]). For this reason, we shift this burden to a pre-processing step, and we require that a suitable point configuration and a suitable height vector have been selected.

Let h be a generic height vector in the sense of Definition 2.2, and $\mathcal{T}_A(h)$ be the associated weighted Delaunay triangulation of $\mathrm{conv}(A)$ (Definition 2.1). We introduce the following weighted version of the well-known Gabriel property of Delaunay facets, see, e.g. [26, 27].

Definition 4.1 (Gabriel property)**.** Let C be a set of indices between 1 and n such that $|C| = d$ and the points $(a_c)_{c\in C}$ are affinely independent. Suppose that there exists a point $\gamma \in \mathrm{aff}((a_c)_{c\in C})$ (i.e., satisfying $\det((a_c, 1)_{c\in C}, (\gamma, 1)) = 0$) such that

$$\gamma \cdot (a_i - a_c) - \frac{h_i - h_c}{2} \leq 0, \tag{4.1}$$

for all $c \in C$ and for all $i = 1, \ldots, n$, with equality if and only if $i \in C$. Then the facet $\mathrm{conv}((a_c)_{c\in C})$ is said to have the Gabriel property.

Gabriel facets are useful because they are automatically included in the weighted Delaunay triangulation. To see this, let N be a normal to the facet $\sigma_C \coloneqq \mathrm{conv}((a_c)_{c\in C})$, and let ε be the real number of smallest absolute value for which $(\gamma + \varepsilon N) \cdot (a_b - a_c) - (h_b - h_c)/2 = 0$ for some $b \notin C$. Since N is normal to σ_C, a_b must be affinely independent from the points $(a_c)_{c\in C}$. The coordinates of $\gamma' \coloneqq \gamma + \varepsilon N$ can be then computed explicitly by solving the d linearly independent equations $(a_i - a_c) \cdot \gamma' = (h_i - h_c)/2$ for $i \in B \coloneqq C \sqcup \{b\}$. It is then easy to check, comparing with (2.5), that $\mathrm{sign}(\gamma' \cdot (a_i - a_b) - (h_i - h_b)/2) = D(B, i)$ for all $i = 1, \ldots, n$ and $b \in B$. Recalling that h is generic, this implies that σ_B (and therefore its facet σ_C) appears in the corresponding weighted Delaunay triangulation.

Given an arbitrary facet σ_C, one can find a candidate point γ by choosing an index $\overline{c} \in C$ and finding a solution to the d linearly independent equations $\det((a_c, 1)_{c\in C}, (\gamma, 1)) = 0$ and $\gamma \cdot (a_i - a_{\overline{c}}) = (h_i - h_{\overline{c}})/2$ for $i \in C$, $i \neq \overline{c}$. The point γ represents a generalization of the center of the diametrical sphere circumscribed to σ_C. If γ satisfies (4.1), then the facet is already Gabriel, else one may proceed to *subdivide* σ_C by selecting some barycentric coordinates λ_C on σ_C and introducing a new point $e = \sum_{c\in C} \lambda_c a_c$ into A,

and a corresponding height value $h_e < \sum_{c\in C} \lambda_c h_c$. The sublinearity of the height value then makes it more likely that the Gabriel condition (4.1) is satisfied. The subdivision process can be further iterated if needed, until all the required facets are Gabriel. In the case of the standard Delaunay triangulation, this process is known as making the triangulation *conforming Gabriel*, see, e.g., [28, 29] and Figure 5. We do not describe here how one can most efficiently refine the constrained facets so that they are conforming Gabriel, and we assume instead that this pre-processing step has already been performed.

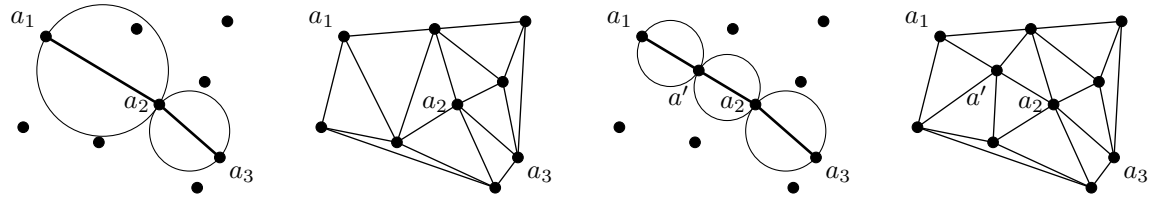

Figure 5: Example of Gabriel facets in two dimensions for the standard Delaunay height vector. (Left) the facet a_2a_3 is Gabriel, since its diametral sphere is empty, but the facet a_1a_2 is not, and therefore it does not appear in the Delaunay triangulation (center left). (Center right) The Gabriel property can however be restored by refining the facet a_1a_2, introducing the additional point a'. The facet a_1a_2 is then guaranteed to be represented in the modified Delaunay triangulation (right).

After making sure that all the constrained facets $F \in \mathcal{F}$ are Gabriel, one still has to make sure that they are included not only in the order-zero Delaunay triangulation of A, but also in all the iterated triangulations that are required for the construction of splines of degrees $1 \geq r \leq k$ in Algorithm 1. We show in the following subsections that repeating all the points belonging to constrained facets $k+1$ times in A is sufficient, provided that the height vector h is fine-grained (Definition 2.3).

Notice that the notion of a fine-grained height vector is compatible with the Gabriel property in the presence of repeated points. In fact, in the definition of a Gabriel facet, one can use for each vertex the copy that has the lowest index in A. Then, (4.1) is satisfied automatically for any point coincident with a facet vertex, thanks to (2.6).

4.2 Domain decomposition of spline spaces

We now prove that, if the properties discussed in the preceding section are satisfied, the spline space built via Algorithm 1 can indeed be used to create a spline space on each subdomain $\Omega_i \subseteq \Omega$.

First, we prove a proposition characterizing the behavior of these spline spaces near the constrained facets $F \in \mathcal{F}$, i.e., near the boundary of Ω and the interfaces between subdomains. In the following, $\binom{a}{b} \coloneqq a!/(b!(a-b)!)$ always denotes the binomial coefficient.

Proposition 4.2. *Let h be a generic, fine-grained height vector, and let F be a Gabriel facet, arbitrarily oriented, whose vertices are repeated at least $k+1$ times in A. If there are at least $k+1$ points in A on a given side of F, then there are exactly $\binom{k+d}{k}$ spline functions $M(x \mid I \sqcup B)$ of degree k such that $\sigma_B \coloneqq \mathrm{conv}((a_b)_{b\in B})$ lies on that side of F and contains F as a facet.*

Proof. By hypothesis, each of the d vertices of F is repeated at least $k+1$ times in A. Choose one copy for each vertex, and let C be the corresponding set of indices, with $|C| = d$. For each $c \in C$, define

$$R_c \coloneqq \{1 \leq i \leq n : a_i = a_c,\ h_i < h_c\}, \tag{4.2}$$

let $R_C \coloneqq \bigsqcup_{c\in C} R_c$ and let $k' \coloneqq |R_C|$. Notice that there are exactly $\binom{k+d}{k}$ distinct ways to choose the copies of vertices of F such that $k' \leq k$. For any such choice, we can complete the set R_C to a set $I \supseteq R_C$ with $|I| = k$ as follows.

Since F is Gabriel, there exists a point $\gamma \in \mathbb{R}^d$ such that, for all $c \in C$, $\gamma \cdot (a_i - a_c) - (h_i - h_c)/2$ is zero if $i \in C$ and negative if $i \notin C$. Pick an open halfspace of F and suppose that there are at least $k+1$ points of A (not necessarily distinct) lying in it. Denote by J the indices of the points of A in this open halfspace, and let N be the normal of σ_C pointing towards this side. For each index $j \in J$, the expression

$(\gamma + \theta N) \cdot (a_j - a_c) - (h_j - h_c)/2$ is linear in θ and negative for $\theta = 0$, and since $N \cdot (a_j - a_c) > 0$, it must cross zero and become positive for some value $\theta = \theta_j > 0$. Since h is generic, all the values θ_j must be distinct. Let

$$S \coloneqq \{j : (\gamma + \theta_j N) \cdot (a_j - a_c) - (h_j - h_c)/2 = 0 \text{ for all } c \in C \text{ and some } \theta_j > 0\}.$$

By hypothesis, S contains at least $k+1$ elements. Sort the indices in S by increasing value of the corresponding parameter θ_j, and let $S_{k-k'}$ contain the first $k-k'$ elements of S, and let b correspond to its $(k-k'+1)$-th element.

Define $B \coloneqq C \sqcup \{b\}$ and $I \coloneqq R_C \sqcup S_{k-k'}$, so that $|I| = k$. It is clear from (2.5), (4.2) and the definition of $S_{k-k'}$ that $D(B,i) < 0$ for $i \notin I \sqcup B$ and $D(B,i) > 0$ for $i \in I$. Thus, by (2.7), the simplex spline $M(x \mid I \sqcup B)$ is in $\mathcal{S}^{(k)}$. We can build in this way $\binom{k+d}{k}$ distinct simplex splines, whose simplices σ_B are all incident on the same side of the facet $F = \sigma_C$.

□

Notice that, if the facet F has at least $k+1$ points of A on each side, then Proposition 4.2 can be applied to both sides of F. An example of the simplices of Proposition 4.2 is shown in Figure 6, top right.

Proposition 4.2 is a generalization of a property of the usual mesh-based discontinuous Galerkin methods, where for every facet F in the mesh having a simplicial mesh element on its positive side, there are exactly $\binom{k+d}{k}$ linearly independent polynomials supported on the mesh element adjacent to F. We show in the next subsection that this is not an accident.

Proposition 4.2 immediately allows to subdivide the spline functions in the global spline space over Ω between the subdomains $(\Omega_i)_{i=1}^{n_d}$.

Corollary 4.3. *Let h satisfy the hypotheses of Proposition 4.2, and suppose that each facet $F \in \mathcal{F}$ is Gabriel and its vertices are repeated at least $k+1$ times in A. Then, for every spline $M(x \mid I \sqcup B)$ of degree k, the interior of the simplex $\operatorname{conv}((a_b)_{b \in B})$ cannot intersect the boundary of any subdomain.*

Proof. For every facet $F \in \mathcal{F}$ on the boundary of a subdomain Ω_i, select the side of F on which Ω_i lies. Then, there are at least $k+1$ points on the positive side of F, and Proposition 4.2 ensures that there are $\binom{k+d}{k}$ spline functions $M(x \mid I \sqcup B)$ whose simplex $\operatorname{conv}((a_b)_{b \in B})$ is adjacent to F on its positive side. Notice that these simplices do not cross F. Let $\mathcal{R}$ be the intersection of their interiors, and let $M(x \mid J \sqcup B')$ be a distinct simplex spline whose associated simplex $\sigma_{B'} \coloneqq \operatorname{conv}((a_b)_{b \in B'})$ intersects F. Then, the interior of $\sigma_{B'}$ must also intersect $\mathcal{R}$, in contradiction with [8, Proposition 2.12], which states that only $\binom{k+d}{k}$ such simplices cover the region $\mathcal{R}$. Thus, the interior of $\sigma_{B'}$ cannot cross F. The same reasoning can be repeated for all the facets on the boundary of any subdomain Ω_i, since they are all Gabriel by hypothesis, proving the corollary. □

If one chooses a set of domain boundaries and a height vector that satisfy the hypotheses of Corollary 4.3, then one can use the process detailed in the previous section to build a spline space over the whole Ω, and assign each spline function to a subdomain. This yields one spline $\mathcal{S}_i^{(k)}$ space for each subdomain Ω_i, determined according to the following criterion,

$$M(x \mid I \sqcup B) \in \mathcal{S}_i^{(k)} \text{ if and only if } \operatorname{conv}((a_b)_{b \in B}) \subseteq \Omega_i. \tag{4.3}$$

This is the criterion that we choose to construct our discontinuous spline space. Notice that some splines of $\mathcal{S}^{(k)}$ might have associated simplices lying outside Ω, and are therefore discarded. Notice also that the interior knots of the splines in $\mathcal{S}_i^{(k)}$ are also contained in Ω_i. This is trivially true for splines of degree 0, and since Algorithm 1 states that the set of internal points of splines of degree k is a subset of the set of points $I \sqcup B$ of splines of order $k-1$, it is true by induction at all degrees k.

Finally, notice that Corollary 4.3 has as an interesting special case the usual discontinuous Galerkin (DG) polynomial basis over a simplicial mesh.

Corollary 4.4. *Let the height vector h and the subdomains $(\Omega_i)_{i=1}^{n_d}$ satisfy the hypotheses of Corollary 4.3, and suppose furthermore that each subdomain Ω_i is a simplex, so that $(\Omega_i)_{i=1}^{n_d}$ forms a triangulation of Ω. Then, the corresponding spline space $\mathcal{S}^{(k)}$ is the usual Bernstein-Bézier discontinuous Galerkin basis of degree k over each simplex Ω_i.*

Proof. For any facet F of Ω_i, choose the side of F on which Ω_i lies, and follow the proof of Proposition 4.2. Notice that in this case, since all the points in Ω_i are repeated $k+1$ times and h is fine-grained (Definition 2.3), the points indexed by the set $S_{k-k'}$ must be all coincident. Thus, the knot vector of the resulting spline $M(\,\cdot \mid I \sqcup B)$ consists of $d+1$ distinct points $(a_b)_{b\in B} \subset \mathbb{R}^d$, each repeated a number of times $1 \le r_b \le k+1$, with $\sum_{b\in B} r_b = k+d+1$. Expression (2.2) then shows that, for any continuous function f,

$$\begin{aligned}\int_{\mathbb{R}^d} f(x) M(x \mid I \sqcup B)\,\mathrm{d}x = \frac{1}{(k+d)!} &= \int_{\Sigma^{k+d}} f\left(\sum_{i\in I\sqcup B} \lambda_i a_i\right)(\,\mathrm{d}\lambda_i)_{i\in I\sqcup B}, \\ &= \frac{\prod_{b\in B}(r_b-1)!}{(k+d)!} \int_{\Sigma^d} f\left(\sum_{b\in B} \lambda_b a_b\right)\left(\lambda_b^{r_b-1}\right)_{b\in B}(\,\mathrm{d}\lambda_b)_{b\in B},\end{aligned} \tag{4.4}$$

where we have used in the last step a known property of Dirichlet averages (see, e.g., [30, Theorem 5.2-4]) to reformulate the integral over the $(k+d)$-dimensional simplex Σ^{k+d} as an integral over the d-dimensional simplex Σ^d. The product $(\lambda_b^{r_b-1})_{b\in B}$ can be recognized as the barycentric representation of a Bernstein polynomial of degree $\sum_{b\in B}(r_b-1) = k$, see, e.g., [31]. Consequently, the last expression in (4.4) is simply the integral over σ_B of the product of $f(x)$ with a Bernstein polynomial. Moreover, the $\binom{k+d}{k}$ splines supported over σ_B correspond to the $\binom{k+d}{k}$ combinations of multiplicities $(r_b)_{b\in B}$ summing to $k+d+1$, as shown in the proof of Proposition 4.2. Thus, the whole Bernstein-Bézier basis over the simplex is obtained. Finally, since the facets of Ω_i satisfy the Gabriel property and Ω_i is a simplex, the simplex σ_B must coincide with Ω_i, hence completing the proof. □

This result proves that one can obtain the usual Bernstein-Bézier discontinuous Galerkin (DG) mesh-based polynomial basis as a special case of our method. Thus, with our formulation, one expects to be able to easily and naturally mix unstructured spline-based patches and DG patches with simplicial mesh elements in the same numerical scheme. We show that this is indeed the case with some numerical experiments in Section 6.

4.3 Trace of spline functions on domain boundaries

The spline spaces produced under the assumptions of Corollary 4.3 are indeed discontinuous at the domain interfaces. In fact, they include splines of degree k with $k+d$ points (out of the total $k+d+1$) lying on the same facet F on the boundary of the domain. Using the recurrence relation (2.1), it is easy to show that these functions have a non-zero trace on F, i.e., on the boundary of the subdomain, cf. Figure 1. These functions can therefore be used to evaluate boundary conditions, inter-domain fluxes and penalty terms for discontinuous-Galerkin approaches, as we do in Section 5.

We show in the next Proposition that the spline functions having a nonzero trace over a subdomain have a particularly simple form.

Proposition 4.5. *In the hypotheses of Proposition 4.2, the simplex splines supported in a subdomain and having a nonzero trace on a subdomain facet $F \in \mathcal{F}$ are exactly the Bernstein-Bézier polynomials with nonzero trace on F and supported on a common simplex adjacent to F.*

Proof. Due to the regularity of simplex spline functions (cf. (2.1) and Figure 1), a simplex spline $M(x \mid I \sqcup B)$ of degree k has a nonzero trace on F if and only if exactly $k+d$ of its knots lie on $\mathrm{aff}(F)$. The knots $(a_b)_{b\in B}$ describe a non-degenerate d-dimensional simplex. Thus, there is exactly one $b \in B$ such that $a_b \notin \mathrm{aff}(F)$, and letting $C \coloneqq B \setminus \{b\}$, one concludes that $F = \sigma_C \coloneqq \mathrm{conv}((a_c)_{c\in C})$ is a facet of $\sigma_B \coloneqq \mathrm{conv}((a_b)_{b\in B})$. All the knots $(a_i)_{i\in I}$ must lie on F. We need to prove that they are actually vertices of F.

Let $i \in I$, and suppose that a_i does not coincide with any point $(a_c)_{c\in C}$, so that the perturbation rule (2.6) does not apply. Let $(\lambda_c)_{c\in C}$ be the barycentric coordinates of a_i with respect to σ_C, such that $\sum_{c\in C}\lambda_c = 1$ and $\sum_{c\in C}\lambda_c a_c = a_i$. The fact that $i \in I$ implies $D(B,i) > 0$, via (2.7). Rewrite the last row in the first determinant in the definition of $D(B,i)$, (2.5), as

$$(a_i, h_i, 1) = \sum_{c\in C} \lambda_c (a_c, h_c, 1) + \sum_{c\in C} \lambda_c (0, h_i - h_c, 0), \tag{4.5}$$

and expand the determinant by exploiting its linearity with respect to this row. Since $C \subset B$ and thus $D(B, c) = 0$ for all $c \in C$, the terms arising from the first sum in (4.5) vanish. Direct evaluation of the remaining terms yields

$$0 < D(B, i) = \operatorname{sign}(\sum_{c \in C} \lambda_c h_c - h_i) \left(S(B)\right)^2 .$$

Therefore, $\sum_{c \in C} \lambda_c h_c > h_i$. But since F satisfies the Gabriel property, one can multiply (4.1) by λ_c and sum over c, yielding $h_i > \sum_{c \in C} \lambda_c h_c$ and contradicting (4.2). Thus, a_i must coincide with a vertex of σ_C.

We conclude the proof by noticing that the point a_b (i.e. the only point with index $b \in B$ that does not lie on F) must be the same for all these splines, as it corresponds to the first index in the set $S_{k-k'}$ in the proofs of Proposition 4.2 and Corollary 4.4. Thus, all these splines are supported on the same simplex $\operatorname{conv}((a_b)_{b \in B})$. □

Proposition 4.5 shows that, when the domain boundaries are composed of Gabriel facts whose vertices are repeated at least $k + 1$ times in A, the simplex spline functions in the spline sace $\mathcal{S}_i^{(k)}$ have exactly the same trace on the boundaries of Ω_i as the usual Bernstein-Bézier polynomials used in standard discontinuous Galerkin methods over a simplicial mesh of Ω_i. We exploit this fundamental property in the next Section 5 to show how two important features of the standard discontinuous Galerkin methods can be seamlessly carried over to unstructured spline spaces.

4.4 From simplex splines to multivariate B-splines

Using [8, Corollary 2.4], the explicit expansion of a polynomial $q(x)$ of degree k over a subdomain Ω_i can be expressed in terms of the spline functions of the corresponding space $\mathcal{S}_i^{(k)}$ as

$$q(x) = \binom{k+d}{k}^{-1} \sum_{M(\,\cdot\,|I \sqcup B) \in \mathcal{S}_i^{(k)}} Q((a_i)_{i \in I}) \operatorname{vol}(\sigma_B) M(x \mid I \sqcup B), \tag{4.6}$$

where $Q((a_i)_{i \in I})$ is a symmetric multi-affine function (the *polar form* of q) of the interior knots $(a_i)_{i \in I}$. Since, in our construction, many different splines can share the same interior knot indices I (namely all those obtained from the triangulation of the link region $\mathcal{R}_I$ of Algorithm 1), the decomposition (4.6) of the polynomial $q(x)$ is not unique.

We remove this unwanted freedom by fixing a given linear combination of splines sharing any given set of interior knots. Specifically, as done, e.g., in [15, Chapter 8], we define for every set I the *multivariate B-spline function* of degree k

$$N(\,\cdot \mid I) \coloneqq \binom{k+d}{k}^{-1} \sum_{\{B : M(\,\cdot\,|I \sqcup B) \in \mathcal{S}_i^{(k)}\}} \operatorname{vol}(\sigma_B) M(\,\cdot \mid I \sqcup B). \tag{4.7}$$

It is important to remark that, in order to preserve the multi-patch property of the spline space, the sum (4.7) is done independently for every subdomain Ω_i, and it only includes spline functions from the corresponding space $\mathcal{S}_i^{(k)}$. Simplex splines with the same interior knots but belonging to different patches $\mathcal{S}_i^{(k)}$ are not summed, to preserve the domain decomposition. We illustrate this sum in Figure 7.

Finally, multivariate B-splines can also be used to recover, as a special case, the usual finite element (FE) mesh-based polynomial basis.

Corollary 4.6. *Suppose that the height vector h satisfies the hypotheses of Corollary 4.4, and there is only one subdomain $\Omega_1 = \Omega$. Suppose furthermore that all the points have multiplicity $k + 1$ in A. Then the multivariate B-spline functions defined via* (4.7) *correspond to the usual C^0 Bernstein-Bézier finite element basis on the weighted Delaunay triangulation $\mathcal{T}(h)$ of Ω with height vector h.*

Proof. Since all the points in A are repeated $k + 1$ times, the same reasoning as in the proof of Corollary 4.4 can be applied to show that the spline spaces correspond to a complete basis of Bernstein-Bézier

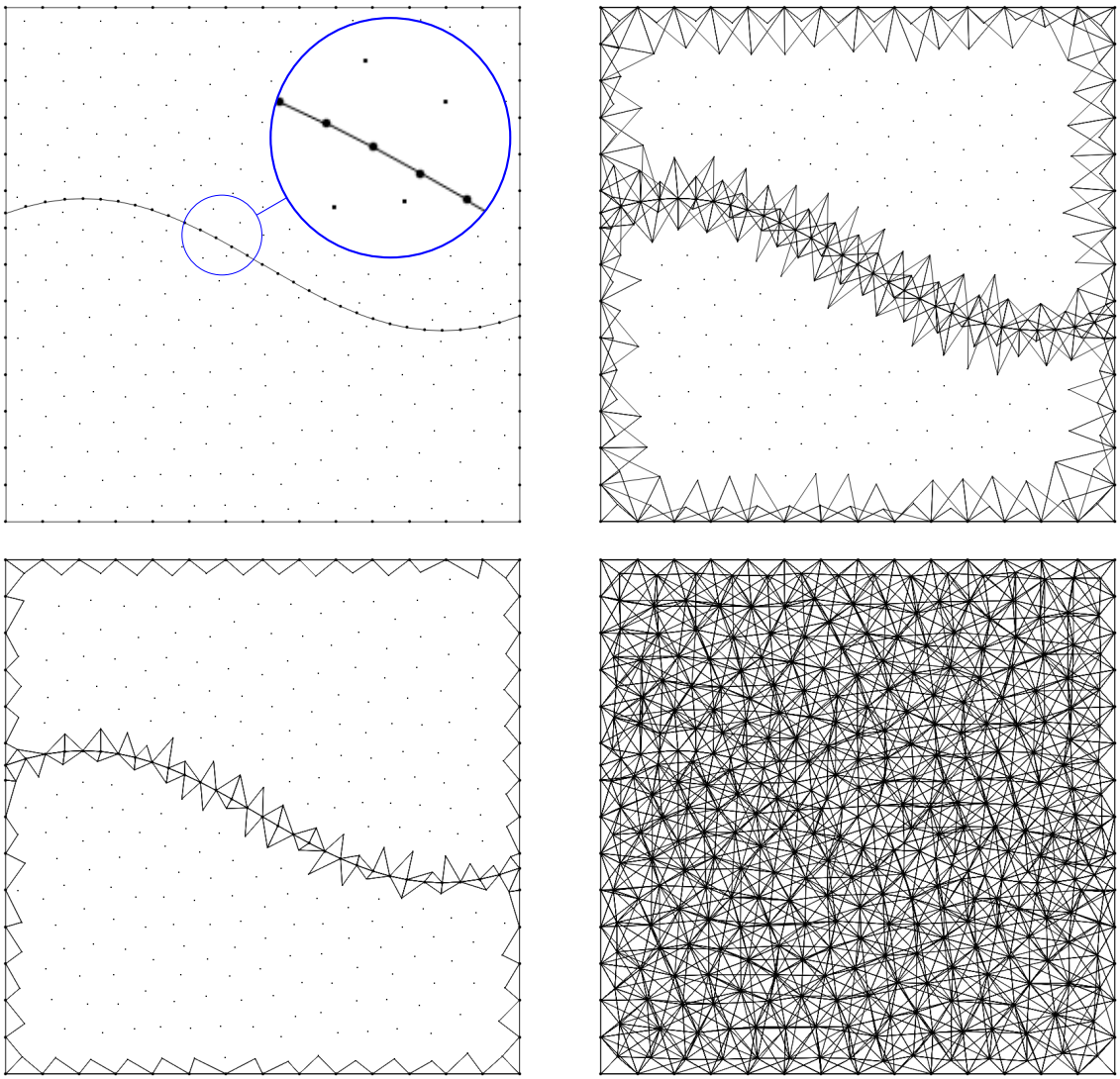

Figure 6: (Top left) a point configuration A of points in $\mathbb{R}^2$, with two domains. We use A to build a spline space of degree $k = 2$. The Gabriel facets are shown in the picture. Points on these facets are repeated $2 + 1$ times. (Top right) the simplices associated to the splines of Proposition 4.2, that protect the boundaries and allow the decomposition of the spline space. (Bottom left) the simplices associated to the splines of Proposition 4.5, corresponding to splines that have a nonzero trace on constrained facets. (Bottom right) the simplices associated to all splines of degree ≤ 2. Their intersection determines the quadrature decomposition.

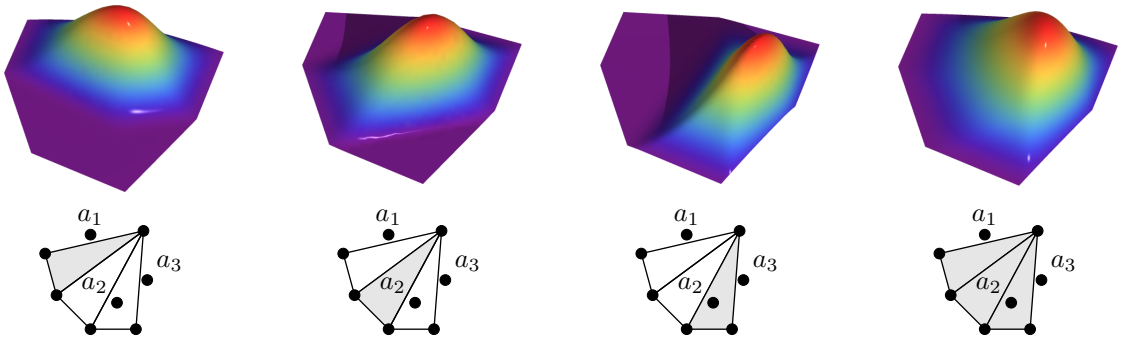

Figure 7: (Top) multivariate B-spline function of degree $k = 3$ (last picture), obtained as a linear combination of the three simplex splines on its left. The splines share the same set of interior knot indices $I = \{1, 2, 3\}$. (Bottom) the corresponding link region $\mathcal{R}_I$ computed by Algorithm 1 and its triangulation, yielding the three simplex splines in the sum.

polynomials of degree k over each simplex of the triangulation $\mathcal{T}(h)$. Consider now two splines $M(x \mid I \sqcup B_1)$ and $M(x \mid I \sqcup B_2)$ sharing the same set of interior knot indices $I \neq \varnothing$. Since these splines are supported on the simplices $\sigma_{B_1} \coloneqq \mathrm{conv}((a_b)_{b \in B_1})$ and $\sigma_{B_2} \coloneqq \mathrm{conv}((a_b)_{b \in B_2})$, respectively, the knots $(a_i)_{i \in I}$ must be vertices of the shared simplex $\sigma_I \coloneqq \sigma_{B_1} \cap \sigma_{B_2}$. Furthermore, these knots must have the same multiplicity in both splines.

Let now $d' < d$ be the spatial dimension of σ_I, and consider the first spline $M(\,\cdot \mid I \sqcup B_1)$. Denote by $C_I \subset B_1$ the indices of unique points lying on σ_I, and by $C_0 \subset B_1$ the indices of points not lying on σ_I. Notice that the points indexed by C_0 must all have multiplicity 1 in $I \sqcup B_1$. One can then write the superposition integral of $M(\,\cdot \mid I \sqcup B_1)$ with any continuous function f, restricted to σ_I, simply by rewriting the last expression of (4.4), setting $(r_c)_{c \in C_0} = 1$ and $(\lambda_c)_{c \in C_0} = 0$, and multiplying by the Jacobian $\mathrm{vol}^{d'}(\sigma_I)/\,\mathrm{vol}^{d}(\sigma_{B_1})$,

$$\int_{\sigma_I} f(x) M(x \mid I \sqcup B_1)\,\mathrm{d}x = \frac{\mathrm{vol}^{d'}(\sigma_I)}{\mathrm{vol}^{d}(\sigma_{B_1})} \frac{\prod_{c \in C_I}(r_c - 1)!}{(k+d)!} \int_{\sigma_I} f\left(\sum_{c \in C_I} \lambda_c a_c\right) (\lambda_c^{r_c - 1})_{c \in C_I} (\,\mathrm{d}\lambda_C)_{c \in C_I}. \quad (4.8)$$

Here, $(a_c)_{c \in C_I}$ is the set of distinct vertices of σ_I and the integers $(r_c)_{c \in C_I}$ represent their multiplicity in both $I \sqcup B_1$ and $I \sqcup B_2$. The same expression is obtained for the other spline function $M(\,\cdot \mid I \sqcup B_2)$, with $\mathrm{vol}^d(\sigma_{B_2})$ replacing $\mathrm{vol}^d(\sigma_{B_1})$ in (4.8). By examining (4.8), one concludes that multiplying the spline $M(\,\cdot \mid I \sqcup B_1)$ by the factor $\mathrm{vol}^d(\sigma_{B_1})$ makes its trace on the interface σ_I independent of the originating simplex σ_{B_1}. Thus, all the splines appearing in the the sum (4.7) have compatible traces on the mesh skeleton, and the corresponding multivariate B-spline function is continuous across the interfaces between its supporting simplices. □

The mechanism that leads to the construction of the Bernstein-Bézier polynomials in Corollary 4.4 and Corollary 4.6 in the case of repeated points can be better understood by looking at Figure 2, panels (a), (d) and (e). One can see that the triangulation of the region $\mathcal{R}_0$ corresponding to a repeated point a always reproduces the set of simplices of the initial Delaunay triangulation of A that have a as a vertex. Thus, all the corresponding spline functions are simply polynomials defined over a simplex, and the corresponding coefficients in the recursive development of (2.1) are simply powers of the barycentric coordinates.

5 Multi-patch DG-IGA scheme for acoustic wave propagation

We showcase the usefulness of our proposed unstructured multi-patch spline spaces by formulating a fully-unstructured multi-patch DG-IGA numerical scheme for the acoustic wave equation,

$$\frac{1}{\rho c^2}\frac{\partial^2 p}{\partial t^2} - \nabla\cdot\left(\frac{1}{\rho}\nabla p\right) = s, \tag{5.1}$$

where $p = p(x,t)$ is the pressure, $\rho = \rho(x)$ is the medium density, $c = c(x)$ is the wave velocity, and $s = s(x,t)$ is a source term. As standard practise for geophysics applications, we solve (5.1) in the time interval $[0,T]$ by setting zero initial conditions,

$$p(x,0) = \frac{\partial p}{\partial t}(x,0) = 0, \tag{5.2}$$

and imposing a set of boundary conditions on $\partial\Omega$ applied throughout the simulation time. More precisely, we subdivide $\partial\Omega$ into three disjoint subsets $\partial\Omega_F$, $\partial\Omega_D$ and $\partial\Omega_A$, over which we impose the Neumann, Dirichlet and first-order absorbing [32, 33] boundary conditions, respectively,

$$p\Big|_{\partial\Omega_D} = 0, \qquad \frac{\partial p}{\partial n}\Big|_{\partial\Omega_F} = 0, \qquad \frac{\partial p}{\partial t}\Big|_{\partial\Omega_A} + c\,\frac{\partial p}{\partial n}\Big|_{\partial\Omega_A} = 0,$$

where $\partial/\partial n$ denotes the outward normal derivative on $\partial\Omega$.

Since unstructured spline spaces are capable of reproducing the usual discontinuous Galerkin polynomial bases as a special case, our approach nicely bridges the multi-patch DG-IGA formulation of [5], based on structured (tensor-product) IGA patches tied by DG fluxes, with more standard mesh-based DG approaches, such as that of [34]. We briefly discuss, later in this section, the implications of this fact for time-explicit schemes.

5.1 Overview of the numerical scheme

The starting point for our formulation is the usual symmetric interior-penalty discontinuous Galerkin (IPDG) scheme presented in [34]. Its weak form can be written as follows,

$$\sum_{i=1}^{n_d}\int_{\Omega_i}\frac{1}{\rho c^2}\varphi\frac{\partial^2 p}{\partial t^2}\,\mathrm{d}\Omega + \sum_{i=1}^{n_d}\int_{\Omega_i}\frac{1}{\rho}\nabla\varphi\cdot\nabla p\,\mathrm{d}\Omega - \sum_{F\in\mathcal{F}_{\mathrm{DG}}}\int_F [\![\varphi]\!]\cdot\{\{\frac{1}{\rho}\nabla p\}\}\,\mathrm{d}F \tag{5.3}$$
$$- \sum_{F\in\mathcal{F}_{\mathrm{DG}}}\int_F \{\{\frac{1}{\rho}\nabla\varphi\}\}\cdot[\![p]\!]\,\mathrm{d}F + \sum_{F\in\mathcal{F}_{\mathrm{DG}}}\alpha_F\int_F [\![\varphi]\!]\cdot[\![p]\!]\,\mathrm{d}F + \sum_{F\in\mathcal{F}_{\mathrm{A}}}\int_F\frac{1}{\rho c}\varphi\frac{\partial p}{\partial t}\,\mathrm{d}S = \sum_{i=1}^{n_d}\int_{\Omega_i}\varphi s\,\mathrm{d}\Omega,$$

where Ω is the domain of interest, subdivided as before into the n_d polytopal (e.g., polygonal for $d=2$ and polyhedral for $d=3$) subdomains $(\Omega_i)_{i=1}^{n_d}$, and φ is a piecewise-polynomial test function, of class at least C^1 in each subdomain. The physical parameters ρ and c are assumed to be positive, bounded, piecewise-smooth functions, and the source $s(x,t)$ is assumed to be in $L^2([0,T];L^2(\Omega))$. The third and fourth terms in (5.3) together form the symmetric flux term, while the fifth term is a penalty term that ensures numerical stability and, weakly, the continuity of the solution.

The set of all subdomain facets $\mathcal{F}$ has been subdivided into three disjoint sets of facets, $\mathcal{F}_D \coloneqq \mathcal{F}\cap\partial\Omega_D$, $\mathcal{F}_N \coloneqq \mathcal{F}\cap\partial\Omega_N$, and $\mathcal{F}_A \coloneqq \mathcal{F}\cap\partial\Omega_A$, as well as a set of interior (or DG) facets, $\mathcal{F}_{\mathrm{DG}} \coloneqq \mathcal{F}\setminus\partial\Omega$. As customary in discontinuous Galerkin methods, given a scalar function u and a vector function v defined over each domain Ω_i, we have denoted by $[\![u]\!]$ and $\{\{v\}\}$, respectively, the *jump* of u and the *average* of v, defined over a facet $F\in\mathcal{F}_{\mathrm{DG}}$ with unit normal vector N_F, as

$$[\![u]\!]_F \coloneqq (u^+ - u^-)N_F,\ \{\{v\}\}_F \coloneqq \frac{1}{2}(v^+ + v^-),$$

where u^+, v^+ (respectively u^-, v^-) are the traces of u and v taken in the domain which sees N_F as an outward normal (resp., inward normal). The definition is extended to facets $F\in\partial\Omega$ by taking N_F to be the unit outward normal, and defining the jump and average simply as $[\![u]\!]_F \coloneqq u^+N_F$, $\{\{v\}\} \coloneqq v^+$.

Notice that free-surface and Dirichlet boundary conditions do not appear directly in (5.3). In fact, the contribution of facets $F \in \mathcal{F}_N$ simply vanishes in (5.3), while the Dirichlet condition is imposed implicitly by discarding from the space all the spline functions with a non-vanishing trace on $\mathcal{F}_D$. Notice also that, compared to the usual IPDG approach, the polytopal subdomains $\Omega_i \subseteq \Omega$ appearing in the sums of (5.3) are in general neither simplices, nor even necessarily convex or simply connected.

5.2 Discrete form

After ensuring that all the constrained facets are Gabriel, we can build the spline space $\mathcal{S}_h$ as follows. First, we construct a spline space $\mathcal{S}^{(k)}$ of degree $k \geq 2$ over $\mathrm{conv}(A)$, using Algorithm 1, which we decompose into a set of n_d spline spaces $(\mathcal{S}_i^{(k)})_{i=1}^{n_d}$, defined over each subdomain, according to the criterion (4.3). We then simply use the following discretization space,

$$\mathcal{S}_h \coloneqq \bigoplus_{i=1}^{n_d} \mathcal{S}_i^{(k)}.$$

After building the discretization space, we select a basis $(\varphi_i)_{i=1}^{n_B}$ of $\mathcal{S}_h$, composed of n_B multivariate B-splines defined via (4.7), and we solve the initial problem (5.2) using a second-order leapfrog (LF2) explicit time scheme, see, e.g., [35]. The discrete form of our problem can be expressed as follows,

$$p^{(t+1)} = \left(M + \frac{\Delta t}{2} B\right)^{-1} \left(2Mp^{(t)} - Mp^{(t-1)} - \Delta t^2 (K + F + P)p^{(t)} + \frac{\Delta t}{2} Bp^{(t-1)}\right) + \Delta t^2 S(t),$$

where $p^{(t-1)}$, $p^{(t)}$ and $p^{(t+1)}$ represent the coefficient vectors of the pressure at three successive time steps over the given basis, $S_i(t) \coloneqq \int_\Omega \varphi_i(x) s(x,t)\,\mathrm{d}x$, and M, K and B are the usual mass, stiffness and damping matrices,

$$M_{ij} \coloneqq \int_\Omega \frac{1}{\rho c^2} \varphi_i \varphi_j \,\mathrm{d}\Omega, \quad K_{ij} \coloneqq \int_\Omega \frac{1}{\rho} \nabla\varphi_i \cdot \nabla\varphi_j \,\mathrm{d}\Omega, \quad B_{ij} \coloneqq \int_{\partial\Omega_A} \frac{1}{\rho c} \varphi_i \varphi_j \,\mathrm{d}S, \tag{5.4}$$

while the flux and penalty terms of (5.3) translate into the corresponding matrices,

$$F_{ij} \coloneqq -\sum_{F \in \mathcal{F}_{\mathrm{DG}}} \int_F \left([\![\varphi_i]\!] \cdot \{\{\frac{1}{\rho}\nabla\varphi_j\}\} + \{\{\frac{1}{\rho}\nabla\varphi_i\}\} \cdot [\![\varphi_j]\!] \right) \mathrm{d}F, \quad P_{ij} \coloneqq \sum_{F \in \mathcal{F}_{\mathrm{DG}}} \int_F \alpha_F [\![\varphi_i]\!] \cdot [\![\varphi_j]\!] \,\mathrm{d}F. \tag{5.5}$$

All vectors have size n_B and all matrices have size $n_B \times n_B$.

The maximum stable timestep of the simulation can be determined using the usual CFL condition, which for the LF2 scheme reads

$$\Delta t_{\mathrm{CFL}} \leq \frac{2}{\sqrt{\lambda_{\max}}}, \tag{5.6}$$

where $\lambda_{\max}$ represents the maximum eigenvalue of the matrix $M^{-1}K'$ and $K' \coloneqq K + F + P$. The penalty coefficient α_F associated to each facet $F \in \mathcal{F}_{\mathrm{DG}}$ needs to be chosen large enough to guarantee the positivity of the bilinear form associated to (5), but not too large, in order to limit its damping effect. We discuss in Section 5.3 how as suitable value can be given to this parameter.

5.2.1 Basis construction, evaluation and quadratures

Once the discrete space $\mathcal{S}_h$ has been build, one can perform the efficient pointwise evaluation of all spline functions by relying on the auxiliary functions constructed in Algorithm 2, using the evaluation process of Algorithm 3. From this point onward, in order to simplify our presentation, we will consider physical parameters that are constant on each subdomain.

In order to be able to assemble the system matrices (5.4) and (5.5), one must be able to compute the integral between two spline functions in $\mathcal{S}_h$. Since simplex splines are piecewise-polynomial functions, this

Algorithm 4 constructs the quadrature cells for the assembly of system matrices.

Input:
A generic point configuration, as in Algorithm 1;
$\mathcal{S}^{(r)}$ the set of spline functions built by Algorithm 1, for $r = 0, \dots, k$.
Output: The set $\mathcal{W}$ of cells over which each spline function is a pure polynomial.
Auxilary procedures: PUSH and POP defined as in Algorithm 2.

procedure BUILDQUADRATURECELLS(A, $\mathcal{S}^{(r)}$ for $r = 0, \dots, k$)
  $\mathcal{T} \leftarrow R^\star$-tree constructed on the simplices $\{\mathrm{conv}((a_b)_{b\in B}) : M(\cdot \mid I \sqcup B) \in \mathcal{S}^{(r)}, 0 \le r \le k\}$
  $x \leftarrow$ a point in the interior of Ω
  $Q = \varnothing$ queue of seed points
  $\mathcal{C} = \varnothing$ set of cells
  PUSH(Q, x) insert x into Q
  while Q is not empty **do**
    $x \leftarrow$POP(Q)
    $\{\sigma_1, \dots, \sigma_m\} \leftarrow$QUERY$(\mathcal{T}, x)$ simplices supported at x ▷ *There are exacty* $\binom{k+d+1}{d}$ *such simplices*
    $W \leftarrow \bigcap_{i=1}^m \sigma_m$
    if W not empty and $\mathcal{W}$ does not contain W **then**
      insert C into $\mathcal{W}$
      for facet F of W **do**
        $f \leftarrow$ centroid of F
        $N_F \leftarrow$ outward normal of F, $|N_F| = 1$
        $x_f \leftarrow f + \varepsilon N_F$ for ε much smaller than any distance between distinct points in A
        PUSH(Q, x_f)
  return $\mathcal{W}$

integral can be evaluated by decomposing the integration domain Ω over a set of disjoint cells $(W_i)_{i=1}^{n_W}$, such that every spline in the space is a pure polynomial over each cell W_i,

$$\int_\Omega M(x \mid I \sqcup B) M(x \mid I' \sqcup B') \,\mathrm{d}\Omega = \sum_{i=1}^{n_W} \int_{W_i} M(x \mid I \sqcup B) M(x \mid I' \sqcup B') \,\mathrm{d}\Omega.$$

We call this decomposition the *quadrature subdivision*. Using the recurrence formula (2.1b) and the validity of the evaluation scheme of Algorithm 3, it is clear that the cells $(W_i)_{i=1}^{n_W}$ can be obtained as intersections of simplices $\sigma_B \coloneqq \mathrm{conv}((a_b)_{b\in B})$ associated to splines $M(x \mid I \sqcup B) \in \mathcal{S}_h$ of degree k or less. This can be achieved via a simple process that we make explicit in Algorithm 4. Notice that, thanks to [8, Proposition 2.12], each quadrature cell is obtained as the intersection of exactly $\sum_{r=0}^{k} \binom{r+d}{d} = \binom{k+d+1}{d}$ such simplices. In turn, these can be computed efficiently, since the halfspace intersection problem is dual to the usual convex hull problem [36], for which many efficient implementations exist. Once the cells are computed, the integration of the product of two splines becomes simply the integral of a polynomial over each cell, which can be computed using standard algorithms.

One drawback of this approach is that the number of cells in the quadrature subdivision can be very large, much larger than the number of cells in a triangulation of A (see Figure 6), and increases with k. We discuss this issue, and some possible solutions, in the concluding section.

5.3 Positivity of the bilinear form and *a priori* error analysis

The IPDG method relies on a penalty term to make its associated bilinear form positive. This is necessary to ensure the stability of the method, but can adversely impact the performance of the numerical scheme, since the condition number of the associated bilinear form scales linearly in the penalty constant (see, e.g., [37]). Thus, the determination of a general criterion for choosing a reasonably small but effective penalty coefficient is crucial.

In [38], Shahbazi introduces a sufficient penalty term α_F on each facet F, computed from the inradius (i.e., the radius of the inscribed sphere) of the two adjacent simplices σ^+ and σ^-. The derivation of this value relies on the existence of inverse inequalities bounding the integral of any polynomial $q(x)$ of degree k over F with integrals over σ^+ and σ^- [39],

$$\int_F q(x)^2 \,\mathrm{d}F \leq \frac{(k+1)(k+d)}{d} \frac{\mathrm{vol}^{d-1}(F)}{\mathrm{vol}^d(\sigma^\pm)} \int_{\sigma^\pm} q(x)^2 \,\mathrm{d}\sigma^\pm. \tag{5.7}$$

Thanks to Proposition 4.5, the same inverse inequalities can be applied to our scheme. In fact, the only spline functions contributing to (5.5) are those that have a non-vanishing trace over the facets $F \in \mathcal{F}_{\mathrm{DG}}$. Proposition 4.5 then guarantees that these functions are all polynomials supported over two fixed simplices, σ_{B^+} and σ_{B^-}, adjacent to F. Consequently, the choice of α_F given in [38, Expression (7)] can be used in our scheme, unchanged, for each face $F \in \mathcal{F}_{\mathrm{DG}}$ between two subdomains. We therefore make the following choice,

$$\alpha_F = \alpha_0 \frac{\max(\frac{1}{\rho^+}, \frac{1}{\rho^-})}{\min(r(\sigma_{B^+}), r(\sigma_{B^-}))}, \tag{5.8}$$

where the subscripts $+$ and $-$ identify the two sides of F, ρ^+ and ρ^- are the respective density values, $B^\pm$ are the two affinely independent sets such that the splines on either side of F are supported on $\sigma_{B^\pm} \coloneqq \mathrm{conv}((a_b)_{b \in B^\pm})$, and $r(\sigma_{B^\pm})$ is the inradius of $\sigma_{B^\pm}$. In [38], the constant factor $\alpha_0 \coloneqq (k+1)(k+d)/2$ is chosen for (5.8), but a larger constant can be used to ensure the good numerical conditioning of the system matrices. In our work, we use the following value,

$$\alpha_0 = \binom{k+d}{k}. \tag{5.9}$$

Finally, and importantly, the *a priori* error estimate of the standard IPDG method (see, e.g., [34]) only relies on the polynomial-reproducing property of the basis, and on the inverse inequality (5.7). Thus, without any additional effort, the same *a priori* error estimate can be carried over without modification to our numerical scheme.

6 Some numerical results

We present in this section a few numerical results of the simulation of the propagation of acoustic waves using our scheme.

6.1 Block-diagonal mass matrix

In Figure 8 we show the effect of domain boundaries and point multiplicities on the sparsity pattern of the mass matrix. In this example, we consider a point configuration A, containing around $1.4 \cdot 10^4$ distinct points, and we show the impact of three different choices for the constrained facets (and thus the repeated knots). We compare a mesh-based approach, where each simplex is a subdomain and all points are repeated $k+1$ times, a multi-patch DG-IGA approach with only 6 subdomains, and a pure IGA approach with a single domain. Notice that the pure DG case is obtained as in Corollary 4.4, and the pure IGA case is obtained by treating the whole Ω as the only subdomain. Consequently, all three numerical schemes are obtained through our construction, with pure DG and pure IGA obtained as the limiting cases. This is reflected in the sparsity pattern of the mass matrix (Figure 8, right), which is always block-diagonal, but where one can arbitrarily vary the size and number of diagonal blocks by choosing a suitable domain decomposition, from one per mesh element up to a single block for the whole domain. Notice also that the blocks are sparse (except near the DG limit) and have a limited bandwidth, comparable with the usual DG case. The pure IGA and DG-IGA cases have a very similar number of nonzero entries, $\sim 3.5 \cdot 10^5$ for $k = 2$, despite their rather different appearance.

In practical implementations, this flexibility can be exploited for efficient load balancing in parallel codes. Ideally, the blocks can be made small enough so that a single computational node is capable of storing the factorization of the mass matrix corresponding to a single subdomain, so as to accelerate the iterations of time-explicit schemes.

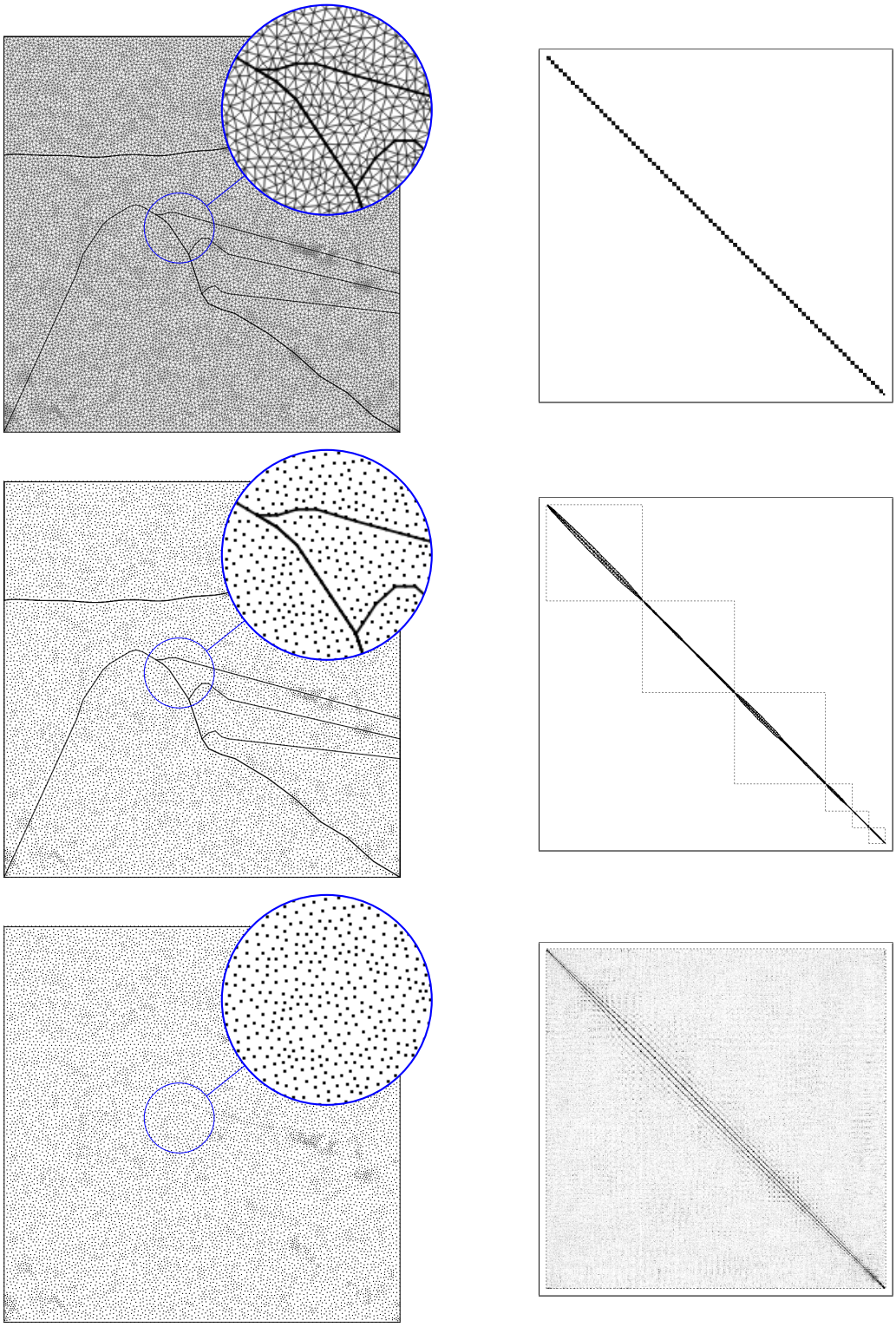

Figure 8: Three choices of constraints for a point configuration A with around $1.4 \cdot 10^4$ distinct points (left), and the resulting sparsity pattern of the mass matrix for $k = 2$ (right). The subdomain shapes are borrowed from the synthetic model discussed in Section 6.3 and Figure 12. (Top) DG approach, with around $2.8 \cdot 10^4$ simplices. (Middle) multi-patch DG-IGA with 6 subdomains. (Bottom) pure IGA with a single subdomain. The subdomain boundaries are not shown in the latter case, since the model now consists of a single domain.

6.2 Validation

We have validated the method on a simple two-dimensional homogeneous model Ω of size $9.2\text{km} \times 3.0\text{km}$ with $\rho = 1000\text{kg}\,\text{m}^{-3}$, $c = 1500\text{m}\,\text{s}^{-1}$, absorbing boundary conditions on all sides, a single source point and an array of receivers. We have used, as in all other results of this section, a pointlike source, whose time dependence is given by the standard Ricker wavelet [40]. We have compared the pressure values at a set of receivers with the analytical result computed using the Gar6more software [41], which are based on the well-known Cagniard–De Hoop method. We have evaluated the L^2 error of the simulation at the location $x_r \in \mathbb{R}^d$, $d = 2, 3$ of each receiver r as

$$e_r^2 \coloneqq \frac{\int_0^T (p(x_r,t) - p_\text{A}(x_r,t))^2 \,\text{d}t}{\int_0^T p_\text{A}^2(x_r,t)\,\text{d}t}, \tag{6.1}$$

where p_A is the analytical solution computed with Gar6more. We have first performed a convergence study on a subdomain $\Omega' \subset \Omega$ of size $2\text{km} \times 2\text{km}$ around the source point, measuring the error as a function of the density of the point set A, i.e., the average distance h between points. Results are in Figure 9 for $k = 2$, 3. We have compared the convergence rate obtained with our method to the corresponding FE and DG simulations, by introducing denser point sets with decreasing average spacing. In each case, we have performed an IGA simulation, defining a single subdomain and increasing the multiplicity of points on $\partial\Omega$, a finite element (FE) simulation obtained by repeating $k+1$ times all the points in A (cf. Corollary 4.6), and a DG simulation obtained from this by additionally defining a subdomain for each simplex in a triangulation of Ω (Corollary 4.4). As can be seen, our method yields the same order of convergence $O(1/h^{k+1})$ in the L^2 norm as the DG and FE reference methods, see, e.g., [34].

After selecting an appropriate point set spacing h, corresponding to a point configuration A with around $3.4 \cdot 10^4$ points, we have repeated the simulation by varying the polynomial order k, ranging from 1 to 4. In Figure 10, we show a snapshot from the simulation for $k = 3$, and we compare the CFL timestep for the LF2 time integration scheme (5.6), the number of degrees of freedom (i.e., the number of multivariate B-splines in the basis), the relative error (6.1) and the error times the number of degrees of freedom, which represents the inverse of the precision per degree of freedom. Again, and for all polynomial degrees, the corresponding bases for the FE and DG methods are supported on a triangulation over the same point set A, and can simply be obtained with Algorithm 1 by setting appropriate point repetitions and domain boundaries, as discussed in the previous section.

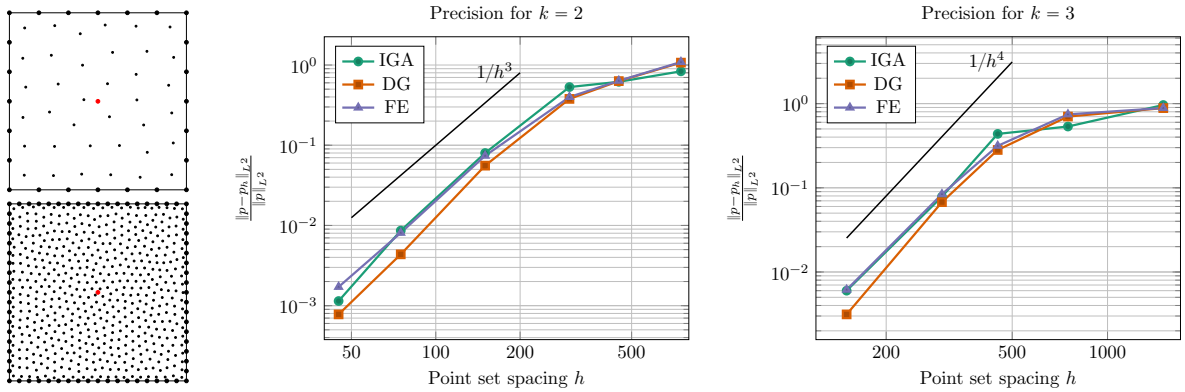

Figure 9: A simple homogeneous domain with two different point set spacings h, corresponding to the different average distance between points (left). The source is located at the center of the model (red dot). The convergence rate of our method (IGA) is compared with that of the DG and FE methods, defined over a triangulation on the same point set, for $k = 2$ (center) and $k = 3$ (right).

As can be seen, multivariate spline spaces share many properties with their more usual tensor-product counterparts. Specifically, the number of degrees of freedom increases only linearly with the order k, due to the fact that no new nodes are inserted. Our numerical simulation suggests that, for higher degrees,

the maximum timestep given by the CFL condition decreases less rapidly than in the case of the FE and DG methods, and possibly only decreases as h/k, instead of h/k^2. However, this conclusion cannot be drawn with certainty from our relatively limited simulation, and would warrant further testing with an extended range of polynomial degrees. The precision per number of degrees of freedom is comparable for FE and IGA, and has the same behavior as a function of k in all three methods.

Notice that, for $k = 1$, FE and IGA coincide, and that at all orders the DG simulation is penalized by the excessive number of degrees of freedom with respect to both FE and IGA, which is a well-known drawback of the method. Also notice that the penalization term α_0 (5.9) used to produce the DG calculation results of Figure 10 is higher than the minimum value necessary for positivity, which further penalizes the maximum allowable timestep of the DG calculation, although it makes the simulation very stable.

In order to investigate whether the introduction of subdomain boundaries impacts the behavior of the CFL condition, we have performed a similar calculation on a simple two-dimensional bi-layered model, with the same dimensions as the two-dimensional homogeneous model above, but split horizontally into halves. The finite element simulation in this case is to be interpreted as a multi-patch FE calculation, where one FE basis is introduced in each of the two subdomains, and DG fluxes and penalty terms are used to couple subdomains.

The two media have the same density $\rho = 1000\mathrm{kg\,m^{-3}}$, but the medium containing the source has a velocity of $c = 1500\mathrm{m\,s^{-1}}$, while the second medium has a velocity of $c = 2500\mathrm{m\,s^{-1}}$. The point configuration contains around $2.3 \cdot 10^4$ points, of which around $1.7 \cdot 10^4$ are in the region of lower velocity, since the point density was adapted to the local wavelength. In this simulation, we have computed the same quantities as in the homogeneous case. The corresponding results, as well as a simulation snapshot, are shown in Figure 11. As one can see, the presence of an interface does in fact penalize the maximum allowable timestep of all the methods. The multi-patch DG-IGA simulation still achieves, however, the best timestep.

In general, as noticed in [5], the superior CFL condition timestep seen in multi-patch DG-IGA methods is due to the increased support of the basis functions. The same behavior is experienced when the usual C^0 finite element bases are modified to have a larger support [42]. In particular, we expect this behavior to be present as long as the number of functions of degree k in each subdomain is much larger than around $\binom{k+d}{k}$.

6.3 Multi-patch simulation and blending with DG

We have tested the multi-patch DG-IGA approach on a simple two-dimensional 3km $\times$ 3km synthetic seismic model, the same used in [43]. This model, shown in Figure 12, is composed of $1.4 \cdot 10^4$ points divided in 6 layers, including water on the top and a salt body in the interior. Although schematic, this model is representative of heterogeneous media found in geophysical problems, with high velocity contrasts typical of subsoil structures containing both sediments and salt bodies.

We have performed a multi-patch DG-IGA simulation with 6 domains, shown in Figure 13, and a hybrid simulation whereby four domains use unstructured spline functions (IGA), and two domains are meshed and use the standard Bernstein-Bézier DG basis obtained via Corollary 4.4, shown in Figure 14. No apparent numerical artifacts were detected at the interfaces between the two numerical schemes, suggesting that, even numerically, the coupling between the two schemes is very natural and seamless.

6.4 Non-simply-connected domains

The usual tensor-product spline spaces used to define multivariate spline functions are limited to a simple topology, namely, that of a topological sphere. Obtaining a non-simply-connected domain then requires gluing together multiple patches. This is a tedious and sometimes very difficult step that often results in reduced regularity along seam lines. Instead, the approach proposed in this work allows to perform full IGA simulations on a non-simply connected domain, simply by placing suitable boundary conditions (or DG fluxes) on the internal boundaries, and excluding the subdomains representing holes from the simulation.

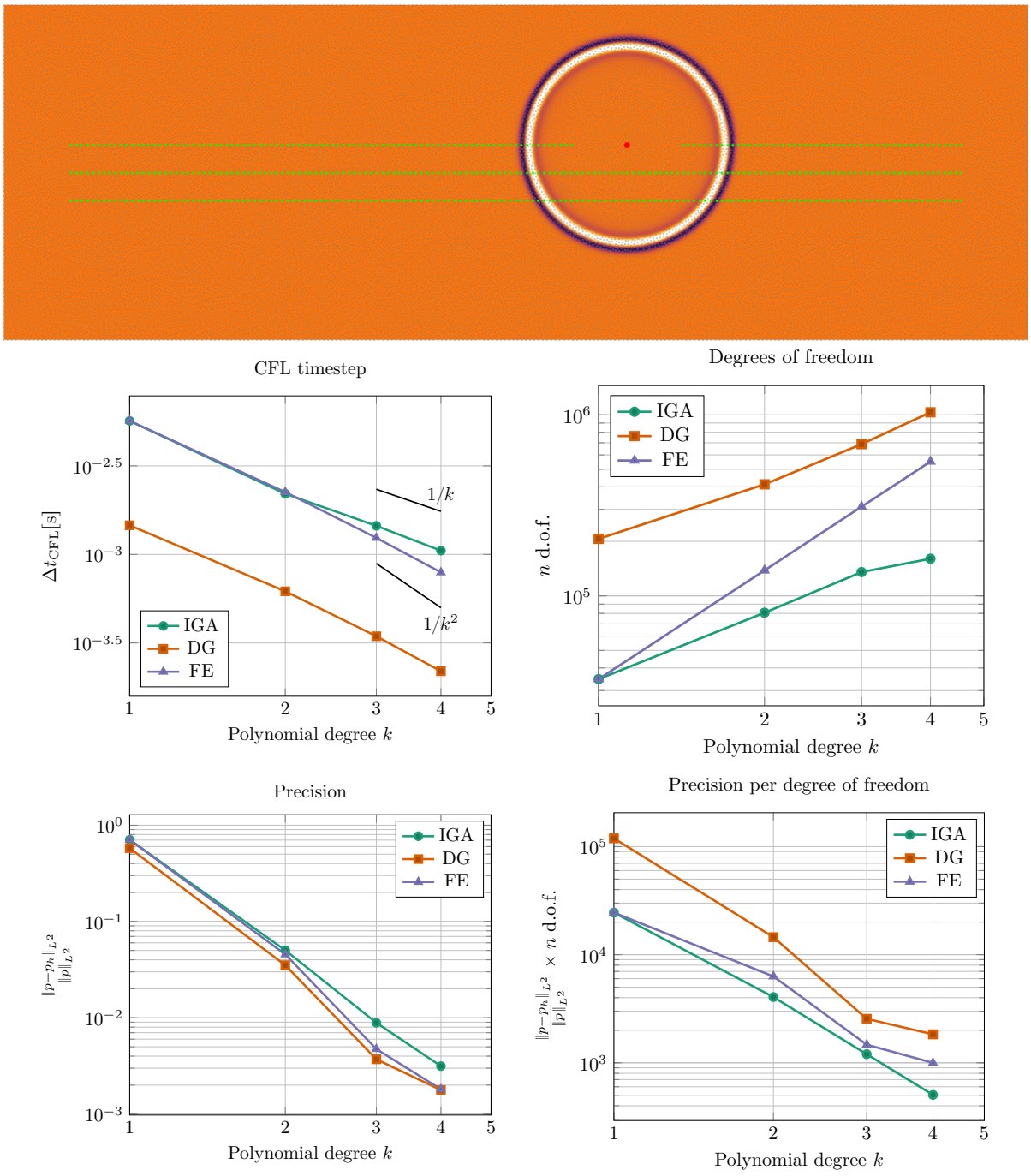

Figure 10: Wave propagation in a two-dimensional homogeneous model (top), in which we compare, for the IGA, FE and DG schemes and for $k = 1, \ldots, 4$, the CFL timestep condition (middle left), the number of degrees of freedom (middle right), the relative error (bottom left) and the relative error times the number of degrees of freedom (bottom right).

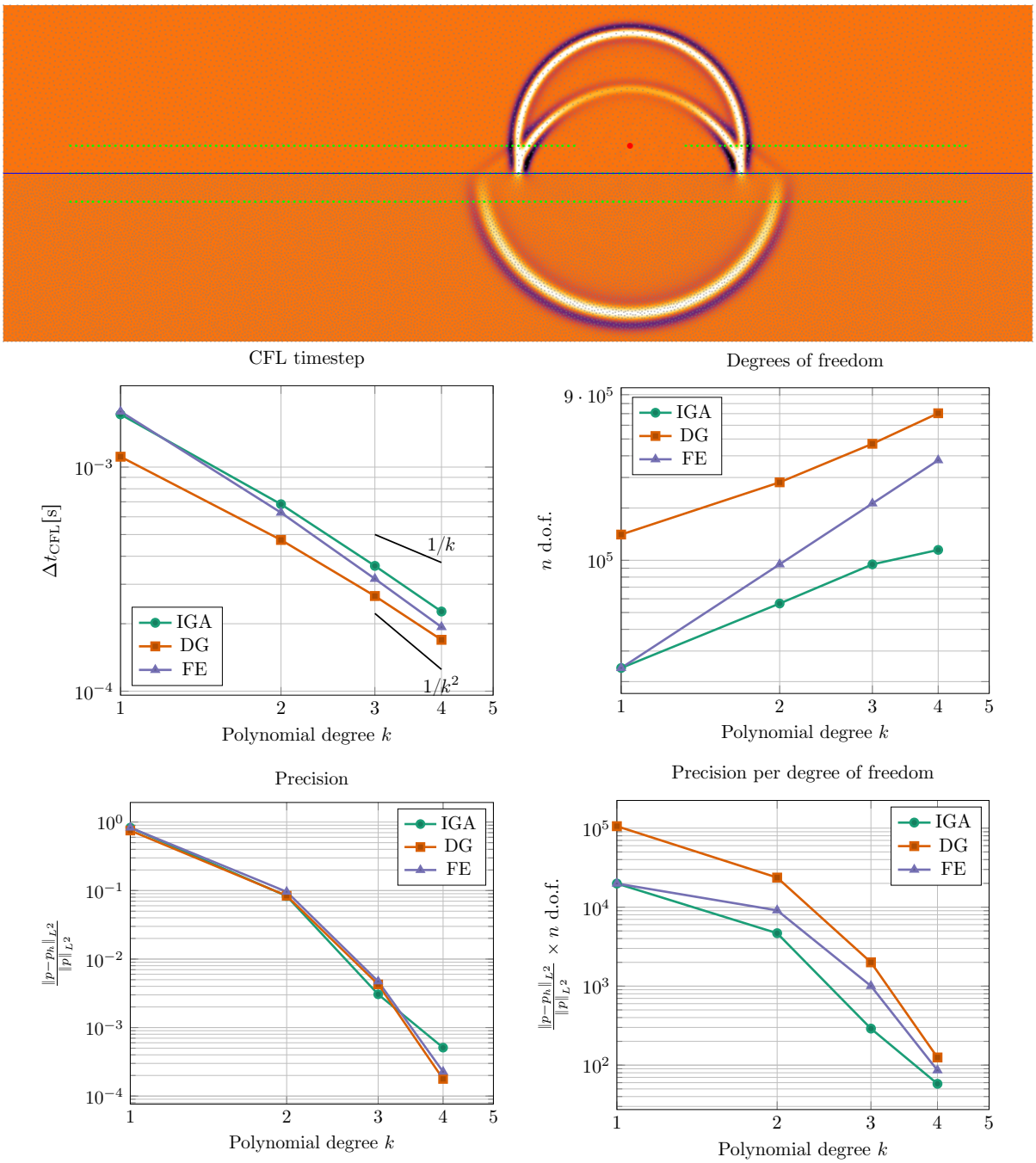

Figure 11: Wave propagation in a two-dimensional bi-layered model (top), in which we compare, for multi-patch IGA, multi-patch FE and DG simulations and for $k = 1, \ldots, 4$, the same quantities as in Figure 10.

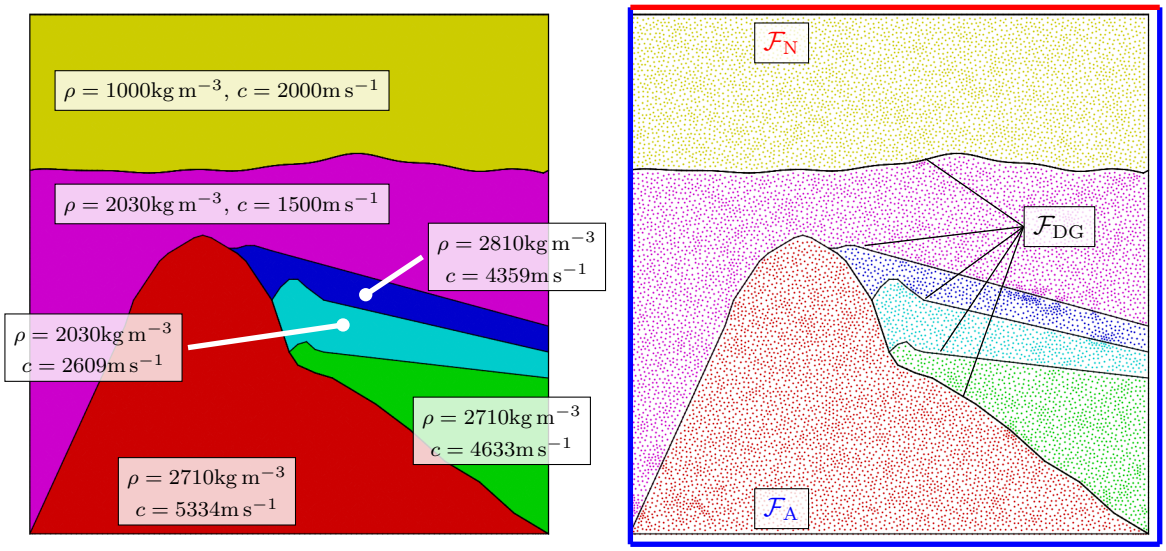

Figure 12: The synthetic model used for testing our multi-patch DG-IGA approach, with its associated physical parameters (left) and the point configuration used to construct our splines and DG basis (right). We also show the location of the absorbing ($\mathcal{F}_\text{A}$), free-surface ($\mathcal{F}_\text{N}$) and DG ($\mathcal{F}_\text{DG}$) boundaries.

We show here two examples, both two-dimensional, of this feature. In Figure 15, we show a simple model inspired by helioseismology applications, with three domains, one with genus zero and two with genus one. The model was adapted from [44], and the spline space was built on a point configuration containing around $6.4 \cdot 10^3$ points. Notice that the density of points has been adapted to the local wavelength. The simulation has degree $k = 3$. Notice that, for this specific case, the exact circular geometry and the availability of a preferred coordinate system (namely, polar coordinates) would allow to treat the same model using standard isogeometric analysis and known trimming techniques. However, an approach based on an unstructured point set provides a simpler and more streamlined workflow, applicable to any shape without relying on finding clever ways of parameterizing the domain. In addition, unstructured spline functions would allow to easily mix and compare standard mesh-based approaches on the same underlying physical simulation domain.

In Figure 16 we show a simple application to the propagation of acoustic waves in a domain of complex topology, by simulating a two-dimensional model of the interior of a church, namely, the Santa Croce basilica in Florence, Italy. The large amount of columns and other obstacles increases the genus of the simulation domain to 99, which would be extremely difficult to obtain by gluing together tensor-product B-spline patches. Using unstructured spline spaces, the regularity of the space is kept maximal (i.e., $k-1$ at degree k) inside the domain. The model comprises around $1.1 \cdot 10^4$ distinct points. The simulation has degree $k = 2$. Contrary to the previous example, devising a set of tensor-product IGA patches covering accurately such a high-genus and complex model would be far from trivial. In practice, this model would be much easier to simulate using a fully unstructured FE or DG method. Notice that, for $k = 2$, our approach uses only about $2.7 \cdot 10^4$ degrees of freedom, while a the FE and DG methods over a triangulation of the same point set would require $4.0 \cdot 10^4$ and $1.2 \cdot 10^5$ degrees of freedom, respectively.

6.5 Three-dimensional domains

Finally, we have performed a few numerical simulations using some simple three-dimensional models, with the goal of testing the capabilities of the method in three-dimensions and proving the feasibility of the simple quadrature cell computation process delineated in Algorithm 4. In practical cases, the algorithm is capable of correctly computing the quadrature cells, which has been determined by inspection of the wave equation solution. However, for $k > 3$, in three dimensions, it seems that the computational burden of determining the quadrature cells and subsequent matrix assembly operations tends to become the

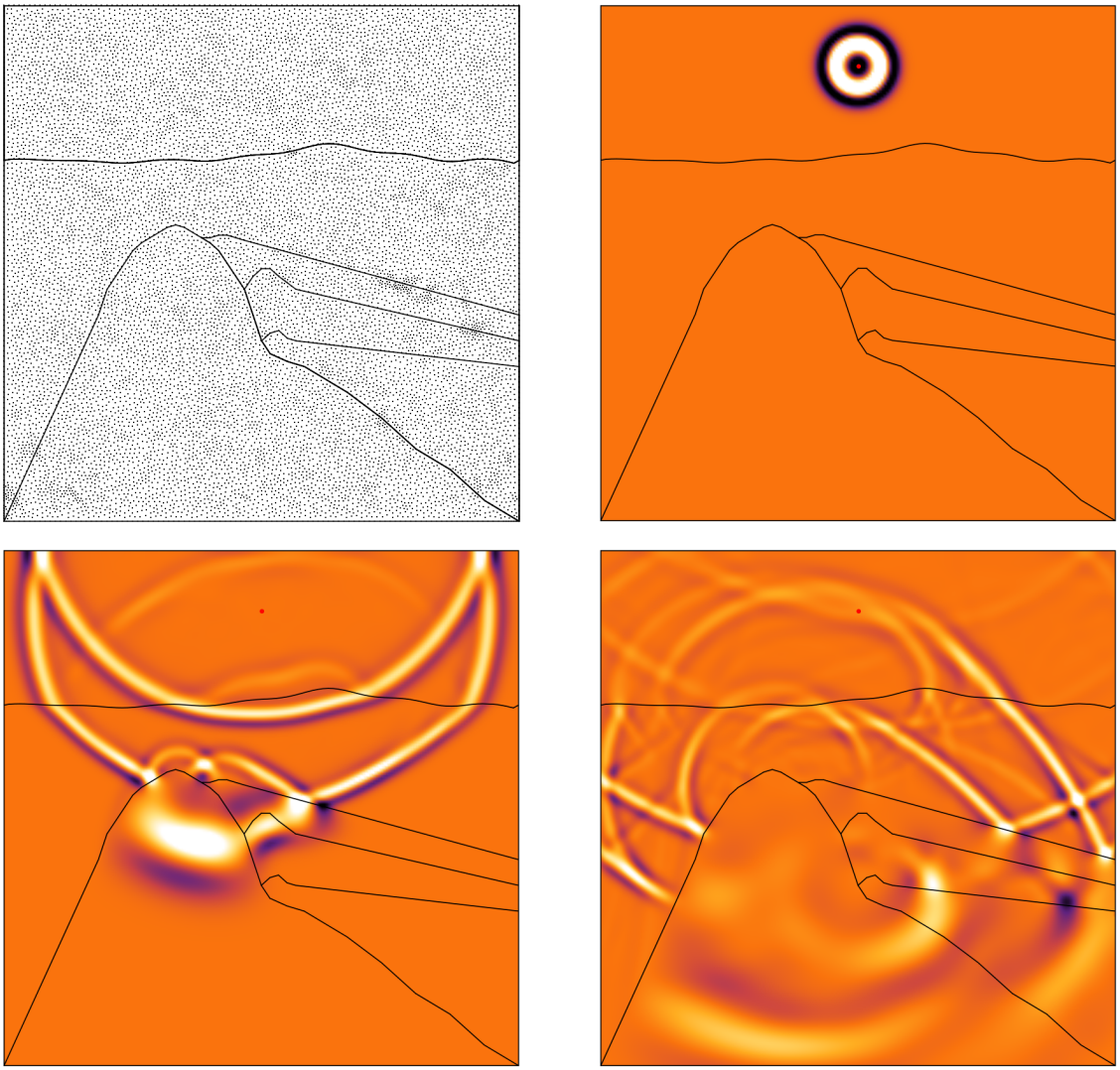

Figure 13: Simulation of the model of Figure 12 using a multi-patch IGA approach based on 6 domains. The spline space used has degree $k = 3$.

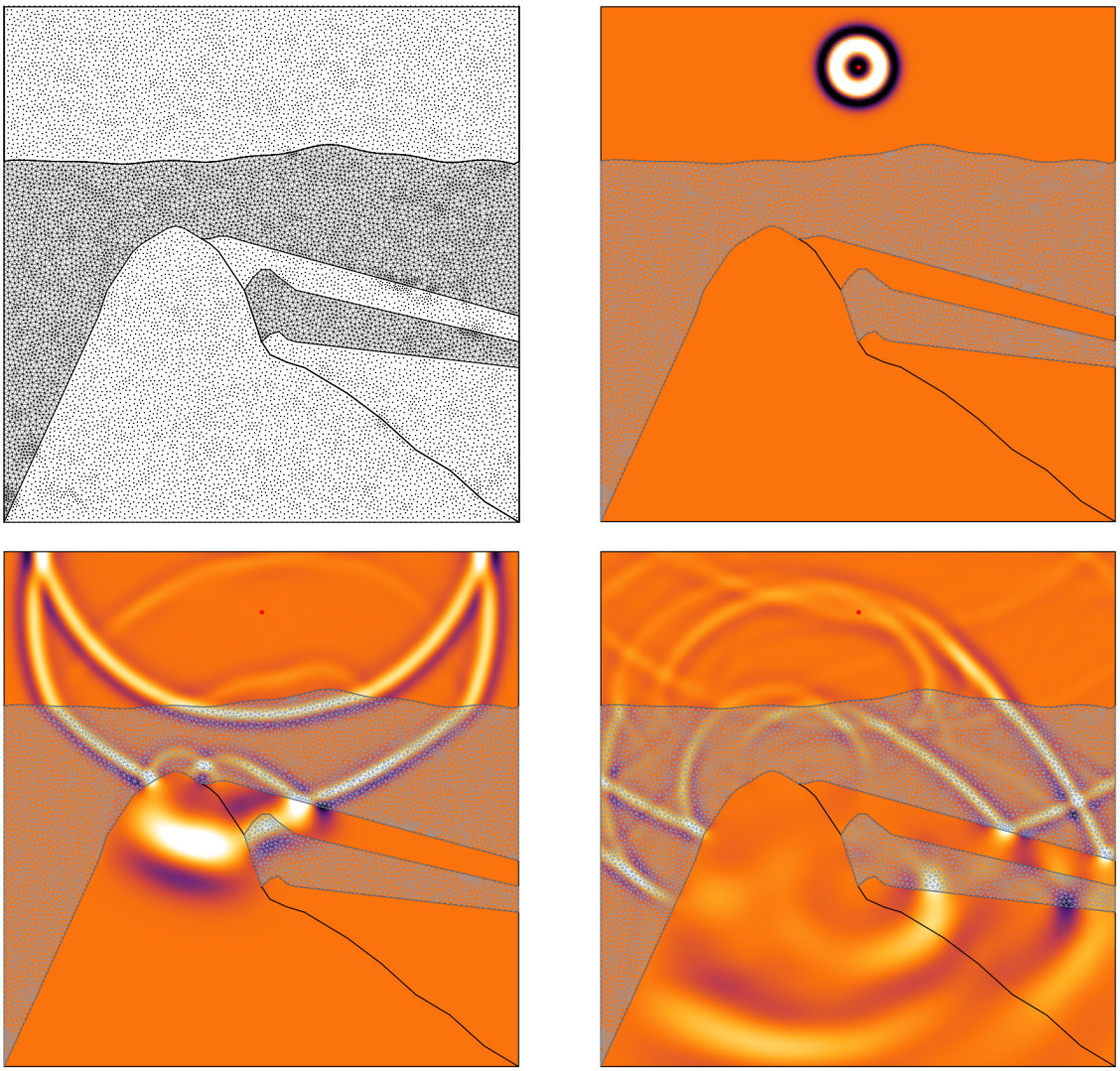

Figure 14: Simulation of the model of Figure 12 using a hybrid multi-patch DG-IGA approach based on 2 meshed (DG) domains and 4 mesh-free (IGA) domains. The spline space used has degree $k = 3$ throughout.

dominant cost of the whole simulation. For this reason, we believe that a different approach for the computation of quadratures is required. We defer this investigation to a future work, and discuss briefly some possible solutions in the concluding section.

We show in Figure 17 the results of the simulation of a simple three-dimensional 1km $\times$ 1km $\times$ 1km bi-layered domain with density $\rho = 1000\mathrm{kg\,m^{-3}}$ everywhere and velocities $c_1 = 2000\mathrm{m\,s^{-1}}$ in the upper half and $c_2 = 3000\mathrm{m\,s^{-1}}$ in the lower half of the model. The simulation was performed at $k = 2$ on the point configuration shown in the picture. The model comprises around $2.6 \cdot 10^4$ points. No significant numerical noise was detected.

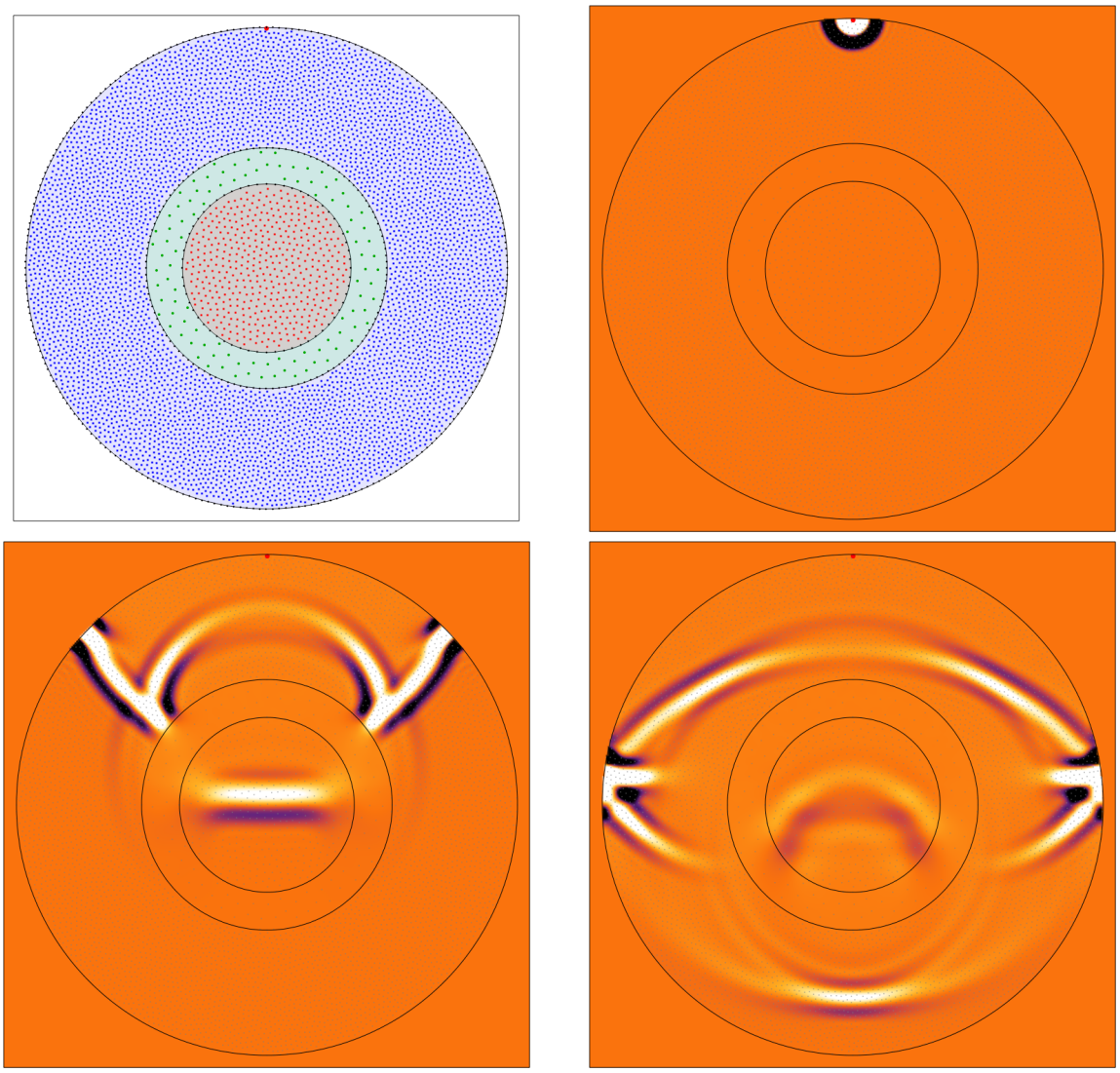

Figure 15: Simulation of a helioseismology-inspired model comprising three domains, two of which are not simply-connected.

7 Conclusions

We have presented a set of practical algorithms and theoretical results that allow to construct a space of polynomial-reproducing simplex spline functions over a multi-patch domain with arbitrary shape and topology. The proposed algorithms allow to build the spline space and evaluate efficiently all the spline

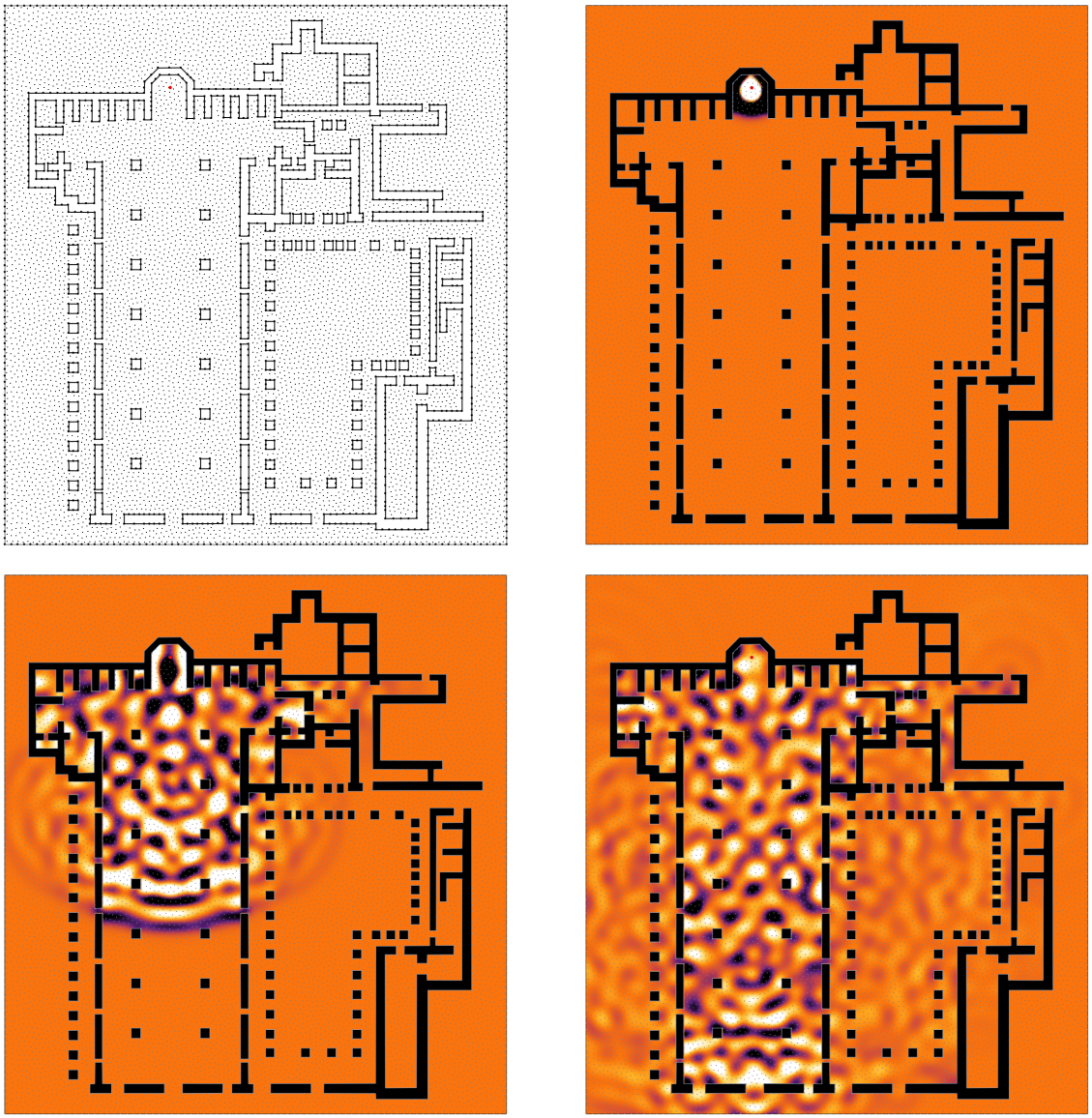

Figure 16: Simulation of wave propagation in the 2D model of a church. The simulation comprises a single domain, with a high genus (99) due to the large number of columns and other obstacles. The regularity of the spline space is maximal inside the simulation domain, and only reduced next to the domain boundaries.

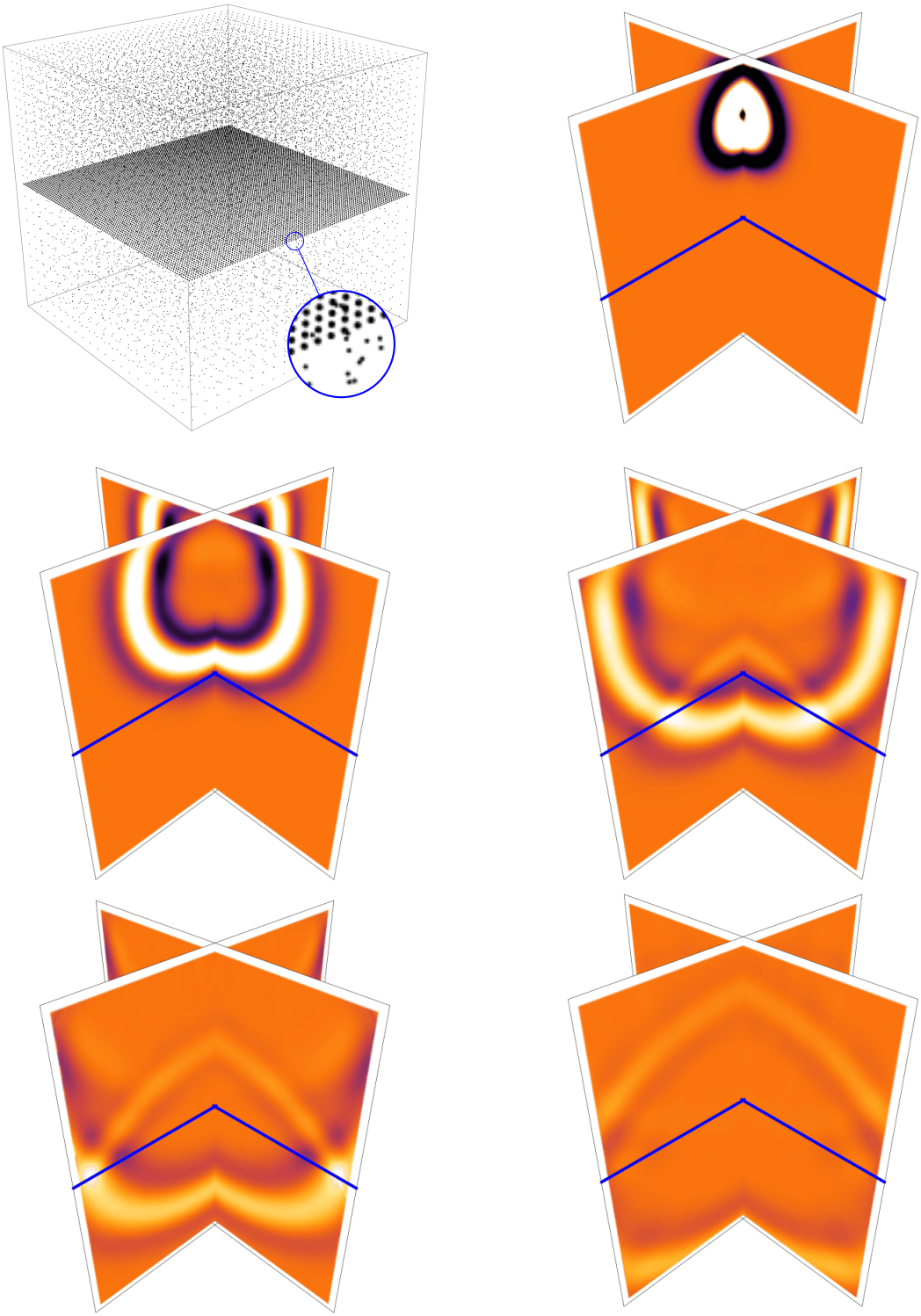

Figure 17: Wave propagation in a simple three-dimensional bi-layered model, composed of two IGA domains. We show the point configuration (interface points are thicker in the image), and five simulation snapshots. The interface between the layers is shown in blue.

functions supported at a given location. We have shown how these spaces can be used to formulate a simple, completely unstructured multi-patch DG-IGA numerical scheme for the solution of PDEs, which we have applied to the propagation of acoustic waves, discussing some of its numerical advantages for explicit time integration schemes.

One of the most interesting features of the proposed method, in our view, is that it provides a natural bridge between the DG and IGA formulations, which can both be recovered as special cases of our construction. Numerically, this translates into the possibility of retaining the block-diagonal structure of the mass matrix of DG approaches (cf. Figure 8) while improving the CFL condition thanks to the smoother, more regular shape of spline functions (cf. Figure 10). Furthermore, the regularity of the basis can be tuned locally at the level of each subdomain, providing a natural way to couple the DG and (unstructured) IGA formulations without introducing any additional numerical noise (cf. Figure 14).

Compared to the usual tensor-product B-spline bases, unstructured simplex spline spaces allow to construct subdomains with arbitrary shape and topology, making it easier to align the zones of reduced regularity with the discontinuities of the physical model. Moreover, a larger number of artificial subdomains may be introduced for purely numerical reasons, for example in order to reduce the size of the diagonal blocks of the system matrices, if required. This flexibility can help fine-tune the size of the subdomains to facilitate load balancing in parallel machines for HPC applications.

One of the main drawbacks of this method is the cost of computing the cells over which all spline functions are pure polynomials and quadrature rules can be used, a prerequisite to the assembly of system matrices. Even though each quadrature cells can be efficiently computed, the number of such cells in realistic cases can be very high, and can increase combinatorially with the degree k. The computation of quadrature cells, and the subsequent matrix assembly operations, can be performed in an embarrassingly parallel way, making full use of high-performance computational resources when available. However, we consider this issue to be an important point to be investigated in future work.

One might be able to avoid a sizable amount of computation by using techniques such as that proposed in [45, 46]. In this work, the inversion of the mass matrix is avoided, while retaining a favorable timestep CFL condition, by using a modified timestepping scheme based on defect correction (DeC) techniques (see, e.g., [47]). Within this paradigm, one recovers the accurate solution of the problem by performing a small and convergent set of iterations over approximate, cheaper problems with appropriate residuals on the right hand side. It is conceivable that one might define a similar approximate solution for unstructured splines, one that not only avoids the inversion of the mass matrix but possibly much of the computational complexity of the definition of exact quadrature cells. The approximate problem would be similar to that obtained through mass lumping techniques, but the accuracy would be recovered via the deferred correction method. We plan on investigating this possibility in a future work.

Alternatively, an improvement in the computational cost associated to matrix assembly might also come from the use of additional combinatorial structures in the computation of superposition integrals (see, for example, the technique based on triangulations of simploids [48]). Given the combinatorial origin of polynomial-reproducing simplex spline spaces, these techniques may also be successful in improving our method. We defer the exploration of these techniques to a future work.

A possible extension of our simplex-spline approach could come from its extension to rational functions, similarly to how usual Non-Uniform Rational B-Splines (NURBS, see, e.g., [49]) extend tensor-product B-splines. In fact, the simplex spline bases introduced in this work form a partition of unity on the whole domain. One could therefore simply introduce control points with $(d+1)$ coordinates and interpret them projectively by dividing the whole expression by the last coordinate, completely analogously to the standard approach of tensor-product bases. This might have interesting applications for Computer-Aided Design (CAD) and engineering modeling, since it would allow to seamlessly include exact-geometry representations of smooth features (e.g., quadrics) into an unstructured, mesh-based model (e.g., a triangulated surface) using the same functional basis.

Finally, we have shown in Figure 15, albeit briefly, that the point set density can be adapted locally to match the expected wavelength of the solution. One might imagine adopting *a posteriori* error estimates to drive an adaptive refinement scheme (see, e.g., [50] for a recent application to hyperbolic problems). The fact that simplex splines are build atop a simple unstructured point set, and not a full-blown mesh, could then simplify the refinement step, as there would be no need to preserve any topological consistency

of the discretization.

Acknowledgments

This work is supported by the Inria - TotalEnergies S.E. joint research team Makutu (`https://www.inria.fr/makutu`).

References

[1] H. Barucq, R. Djellouli, and E. Estecahandy. "Efficient DG-like formulation equipped with curved boundary edges for solving elasto-acoustic scattering problems". In: *International Journal for Numerical Methods in Engineering* 98.10 (2014), pp. 747–780.

[2] T. Warburton and T. Hagstrom. "Taming the CFL number for discontinuous Galerkin methods on structured meshes". In: *SIAM Journal on Numerical Analysis* 46.6 (2008), pp. 3151–3180.

[3] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement". In: *Computer methods in applied mechanics and engineering* 194.39-41 (2005), pp. 4135–4195.

[4] J. A. Cottrell, T. J. Hughes, and Y. Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

[5] J. Chan and J. A. Evans. "Multi-patch discontinuous Galerkin isogeometric analysis for wave propagation: Explicit time-stepping and efficient mass matrix inversion". In: *Computer Methods in Applied Mechanics and Engineering* 333 (2018), pp. 22–54.

[6] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. "Parameterization of computational domain in isogeometric analysis: methods and comparison". In: *Computer Methods in Applied Mechanics and Engineering* 200.23-24 (2011), pp. 2021–2031.

[7] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. "Analysis-suitable volume parameterization of multi-block computational domain in isogeometric applications". In: *Computer-Aided Design* 45.2 (2013), pp. 395–404.

[8] H. Barucq, H. Calandra, J. Diaz, and S. Frambati. "Polynomial-reproducing spline spaces from fine zonotopal tilings". In: *Journal of Computational and Applied Mathematics* 402 (2022), p. 113812.

[9] H. B. Curry and I. J. Schoenberg. "On Pólya frequency functions IV: the fundamental spline functions and their limits". In: *Journal d'analyse mathématique* 17.1 (1966), pp. 71–107.

[10] C. A. Micchelli. "A constructive approach to Kergin interpolation in $\mathbb{R}^k$: multivariate B-splines and Lagrange interpolation". In: *The Rocky Mountain Journal of Mathematics* (1980), pp. 485–497.

[11] B. C. Carlson. "B-splines, hypergeometric functions, and Dirichlet averages". In: *Journal of approximation theory* 67.3 (1991), pp. 311–325.

[12] W. Zu Castell. "Dirichlet splines as fractional integrals of B-splines". In: *The Rocky Mountain Journal of Mathematics* (2002), pp. 545–559.

[13] M. Neamtu. "Delaunay configurations and multivariate splines: a generalization of a result of BN Delaunay". In: *Transactions of the American Mathematical Society* 359.7 (2007), pp. 2993–3004.

[14] Y. Liu and J. Snoeyink. "Quadratic and cubic B-splines by generalizing higher-order Voronoi diagrams". In: *Proceedings of the twenty-third annual symposium on Computational geometry*. ACM. 2007, pp. 150–157.

[15] Y. Liu. "Computations of Delaunay and higher order triangulations, with applications to splines". PhD thesis. University of North Carolina, Chapel Hill, 2008.

[16] L. Guibas and J. Stolfi. "Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams". In: *ACM transactions on graphics (TOG)* 4.2 (1985), pp. 74–123.

[17] H. Edelsbrunner and N. R. Shah. "Incremental topological flipping works for regular triangulations". In: *Algorithmica* 15.3 (1996), pp. 223–241.

[18] D. Schmitt. "Bivariate B-Splines from convex pseudo-circle configurations". In: *International Symposium on Fundamentals of Computation Theory*. Springer. 2019, pp. 335–349.

[19] E. Schönhardt. "Über die Zerlegung von Dreieckspolyedern in Tetraeder". In: *Mathematische Annalen* 98.1 (1928), pp. 309–312.

[20] C. De Boor. *A practical guide to splines*. Vol. 27. Springer-Verlag New York, 1978.

[21] A. Guttman. "R-trees: a dynamic index structure for spatial searching". In: *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*. 1984, pp. 47–57.

[22] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. "The $R^\star$-tree: an efficient and robust access method for points and rectangles". In: *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*. 1990, pp. 322–331.

[23] A. B. Kahn. "Topological sorting of large networks". In: *Communications of the ACM* 5.11 (1962), pp. 558–562.

[24] J. R. Shewchuk. "General-dimensional constrained Delaunay and constrained regular triangulations, I: Combinatorial properties". In: *Twentieth Anniversary Volume:* Springer, 2009, pp. 1–58.

[25] J. Ruppert and R. Seidel. "On the difficulty of triangulating three-dimensional nonconvex polyhedra". In: *Discrete & Computational Geometry* 7.3 (1992), pp. 227–253.

[26] K. R. Gabriel and R. R. Sokal. "A new statistical approach to geographic variation analysis". In: *Systematic zoology* 18.3 (1969), pp. 259–278.

[27] M. Alexa. "Conforming weighted Delaunay triangulations". In: *ACM Transactions on Graphics (TOG)* 39.6 (2020), pp. 1–16.

[28] J. R. Shewchuk. "Mesh generation for domains with small angles". In: *Proceedings of the sixteenth annual Symposium on Computational Geometry*. 2000, pp. 1–10.

[29] D. Cohen-Steiner, E. C. De Verdière, and M. Yvinec. "Conforming Delaunay triangulations in 3D". In: *Computational Geometry* 28.2-3 (2004), pp. 217–233.

[30] B. C. Carlson. *Special Functions of Applied Mathematics*. Academic Press, 1977. ISBN: 9780121601508.

[31] G. Farin. "Triangular Bernstein-Bézier patches". In: *Computer Aided Geometric Design* 3.2 (1986), pp. 83–127.

[32] R. Clayton and B. Engquist. "Absorbing boundary conditions for acoustic and elastic wave equations". In: *Bulletin of the seismological society of America* 67.6 (1977), pp. 1529–1540.

[33] F. Nataf. "Absorbing boundary conditions and perfectly matched layers in wave propagation problems". In: *Direct and inverse problems in wave propagation and applications*. de Gruyter, 2013, pp. 219–232.

[34] M. J. Grote, A. Schneebeli, and D. Schötzau. "Discontinuous Galerkin finite element method for the wave equation". In: *SIAM Journal on Numerical Analysis* 44.6 (2006), pp. 2408–2431.

[35] C. Agut and J. Diaz. "Stability analysis of the Interior Penalty Discontinuous Galerkin method for the wave equation". In: *ESAIM: Mathematical Modelling and Numerical Analysis* 47.3 (2013), pp. 903–932.

[36] F. P. Preparata and M. I. Shamos. *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

[37] P. Castillo. "Performance of discontinuous Galerkin methods for elliptic PDEs". In: *SIAM Journal on Scientific Computing* 24.2 (2002), pp. 524–547.

[38] K. Shahbazi. "An explicit expression for the penalty parameter of the interior penalty method". In: *Journal of Computational Physics* 205.2 (2005), pp. 401–407.

[39] T. Warburton and J. S. Hesthaven. "On the constants in *hp*-finite element trace inverse inequalities". In: *Computer methods in applied mechanics and engineering* 192.25 (2003), pp. 2765–2773.

[40] C. S. Clay. *Elementary exploration seismology*. Prentice Hall, 1990.

[41] J. Diaz and A. Ezziani. *Gar6more 2d*. 2008. URL: `http://gar6more2d.gforge.inria.fr`.

[42] J. W. Banks and T. Hagstrom. "On Galerkin difference methods". In: *Journal of Computational Physics* 313 (2016), pp. 310–327.

[43] A. Citrain. "Hybrid finite element methods for seismic wave simulation: coupling of discontinuous Galerkin and spectral element discretizations". PhD thesis. Normandie Université, 2019.

[44] L. Gizon, H. Barucq, M. Duruflé, C. S. Hanson, M. Leguèbe, A. C. Birch, J. Chabassier, D. Fournier, T. Hohage, and E. Papini. "Computational helioseismology in the frequency domain: acoustic waves in axisymmetric solar models with flows". In: *Astronomy & Astrophysics* 600 (2017), A35.

[45] R. Abgrall, P. Bacigaluppi, and S. Tokareva. "How to avoid mass matrix for linear hyperbolic problems". In: *Numerical Mathematics and Advanced Applications ENUMATH 2015*. Springer, 2016, pp. 75–86.

[46] R. Abgrall. "High order schemes for hyperbolic problems using globally continuous approximation and avoiding mass matrices". In: *Journal of Scientific Computing* 73.2 (2017), pp. 461–494.

[47] M. L. Minion. "Semi-implicit spectral deferred correction methods for ordinary differential equations". In: *Communications in Mathematical Sciences* 1.3 (2003), pp. 471–500.

[48] T. A. Grandine. "The evaluation of inner products of multivariate simplex splines". In: *SIAM Journal on Numerical Analysis* 24.4 (1987), pp. 882–886.

[49] L. Piegl and W. Tiller. *The NURBS book*. Springer Science & Business Media, 2012.

[50] M. Semplice and R. Loubère. "Adaptive-Mesh-Refinement for hyperbolic systems of conservation laws based on a posteriori stabilized high order polynomial reconstructions". In: *Journal of Computational Physics* 354 (2018), pp. 86–110.