Bernstein-Bézier Finite Elements on Tetrahedral-Hexahedral-Pyramidal Partitions

Mark Ainsworth[1], Oleg Davydov and Larry L. Schumaker

Abstract. A construction for high order continuous finite elements on partitions consisting of tetrahedra, hexahedra and pyramids based on polynomial Bernstein-Bézier shape functions is presented along with algorithms that allow the computation of the system matrices in optimal complexity $O(1)$ per entry.

§1. Introduction

In theory at least, finite element approximation on a three dimensional domain differs little from approximation over planar domains. However, the reality is that one is beset by various niggling issues not least of which is the choice of a partitioning for the domain. On the one hand, hexahedral elements are often preferred for reasons of efficiency stemming from the tensor-product nature of the elements and the underlying approximation space, but the practical issues of meshing a complicated three dimensional geometry using only hexahedra should not be underestimated. On the other hand, while there are many efficient algorithms available for meshing using tetrahedral elements, the numbers of degrees of freedom in the resulting finite element space generally significantly exceeds what would be needed should a hexahedral partitioning be available, without a commensurate improvement in accuracy.

An obvious solution would seem to consist of using a partition comprised of a mix of both tetrahedra and hexahedra with hexahedra used as much as possible, with the tetrahedra used to mesh the remaining more complicated geometrical features. This would be fine were it not for the fact that tetrahedra and hexahedra fail to tessellate. Again, a possible solution readily suggests itself whereby pyramidal elements are used as a means of interfacing between the tetrahedra with the hexahedra, see e.g. [6,25].

The question then becomes one of how to define shape functions on the pyramids in such a way as to obtain globally C^0 (conformal) test functions. The problem now is that it is *impossible* to do this using polynomial basis functions. Various approaches have been taken to solving this problem, see [8,9,23] and Remark 3. The upshot is that non-polynomial basis functions are required which, in turn, gives rise to further difficulties associated with numerical integration and the analysis [8,24].

[1] This author gratefully acknowledges the partial support of this work under AFOSR contract FA9550-12-1-0399.

At this point, one might be left with a growing feeling that hybrid partitions may be more trouble than they are worth. Nevertheless, the current work presents a fresh approach to dealing with the pyramidal transition elements which: avoids the need for non-polynomial functions in favour of using piecewise polynomials, and as such is entirely in keeping with the spirit of the finite element method; neatly side steps technical questions associated with the error of the quadrature and approximation power of the spaces; and, opens up the possibility for adopting fast (optimal complexity) algorithms developed in [3] for the system matrix assembly and matrix-free implementations.

The paper is organized as follows. Sect. 2 contains a description of the hybrid partitions of interest in this paper. While this might seem to be straightforward, in practice, it is of considerable interest to allow partitions in which the faces of the hexahedra are allowed to be bilinear quadrilaterals. In turn, this means that one is obliged to consider pyramids whose quadrilateral face is bilinear, which is generally disallowed in existing work on pyramidal elements. We therefore devote attention to carefully defining pyramidal elements of this type and characterizing conditions under which associated maps to the reference element are invertible. The bilinear face of the pyramid means that the splitting of the pyramid into interface tetrahedra is not straightforward. The section concludes with a discussion of an appropriate way to split pyramids into a pair of interface tetrahedra which, of necessity, must be curvilinear. In Sect. 3 we introduce the approximation space to be used in the finite element method, while Sect. 4 is concerned with the construction of the shape functions associated with each of our elements. In Sect. 5 we discuss domain points and minimal determining sets, while in Sect. 6 we use these concepts to find a formula for the dimension of our spline space and to construct a locally supported basis for it. In Sect. 7 we collect some useful computational algorithms for dealing with Bernstein basis polynomials from our previous work, which are then used in Sect. 8 to develop algorithms for efficiently computing all of the various quantities needed for a finite element analysis and evaluation of the resulting approximation. Sect. 9 describes a nodal basis for the approximation space. Finally, the approximation power of our spline space is treated in Sect. 10.

§2. Partitions

We begin by specifying precisely what is meant by a hexahedron, a pyramid and a tetrahedron in the context of hybrid finite element partitions. While the cases of tetrahedra and hexahedra contain no surprises for the seasoned finite element analyst, the situation for pyramids is less straightforward.

2.1 Tetrahedra

A tetrahedron with vertices $v_1, \ldots, v_4$ is a polyhedron with four triangular faces. We write $T = \langle v_1, v_2, v_3, v_4 \rangle$.

2.2 Hexahedra

An ordinary hexahedron is a polyhedron with eight vertices and six *flat* quadrilateral faces. We follow the usual practice of allowing hexahedra whose faces are not necessarily flat, see e.g. [14]. The precise definition makes use of the linear Bernstein polynomials $B_0(\lambda) = 1 - \lambda$ and $B_1(\lambda) = \lambda$ defined on $[0, 1]$, with H_0 being the unit cube $[0, 1]^3$.

Definition 2.1. *Given eight distinct points* $\{v_{ijk}\}_{0\le i,j,k\le 1}$ *in* $\mathbb{R}^3$, *let*

$$\phi_H(\lambda_1, \lambda_2, \lambda_3) := \sum_{0\le i,j,k\le 1} v_{ijk} B_i(\lambda_1) B_j(\lambda_2) B_k(\lambda_3), \tag{2.1}$$

and let $H = \phi_H(H_0)$. *Suppose that* ϕ_H *is a one-to-one mapping from* H_0 *to* H, *i.e., it is invertible on* H. *Then we call* H *a* hexahedron.

We note that a hexahedron as defined here is not necessarily convex. Mappings of the form (2.1) are called trilinear maps and are not necessarily invertible for every location of the points $\{v_{ijk}\}_{0\le i,j,k\le 1}$. We shall not dwell on this point, but refer the curious reader to [20] and references therein for conditions guaranteeing invertibility. Clearly, the vertices of H_0 map to the vertices of H, and the edges of H_0 map to the edges of H, which are easily seen to be straight lines. The faces of H_0, which correspond to setting one of the variables in ϕ_H to either zero or one, map into bilinear surface patches with four straight line edges. We call these the faces of H. Thus, for example, the face corresponding to $\lambda_3 = 0$ is

$$F := \Big\{ \sum_{0\le i,j\le 1} v_{ij0} B_i(\lambda_1) B_j(\lambda_2) : (\lambda_1, \lambda_2) \in [0, 1]^2 \Big\}. \tag{2.2}$$

In general, bilinear surface patches are curved. A face of H is flat if and only if its four vertices lie in a plane. If all faces of H are flat, then H is an ordinary hexahedron.

2.3 Pyramids

An ordinary pyramid is a polyhedron with four triangular faces and one convex quadrilateral (planar) face. Existing work on pyramidal elements is generally confined to the case of ordinary pyramids which, in turn, can only be attached to ordinary hexahedra with flat faces. However, it is of considerable practical importance to allow for pyramids to be attached to hexahedra of the type defined above. Consequently, we are obliged to consider a more general class of pyramids whose bases are allowed to be bilinear Bézier patches.

Definition 2.2. *Suppose*

$$G_P := \{(1-\lambda_1)(1-\lambda_2)v_2 + \lambda_1(1-\lambda_2)v_1 + (1-\lambda_1)\lambda_2 v_3 + \lambda_1\lambda_2 v_4 : (\lambda_1, \lambda_2) \in [0, 1]^2\}, \tag{2.3}$$

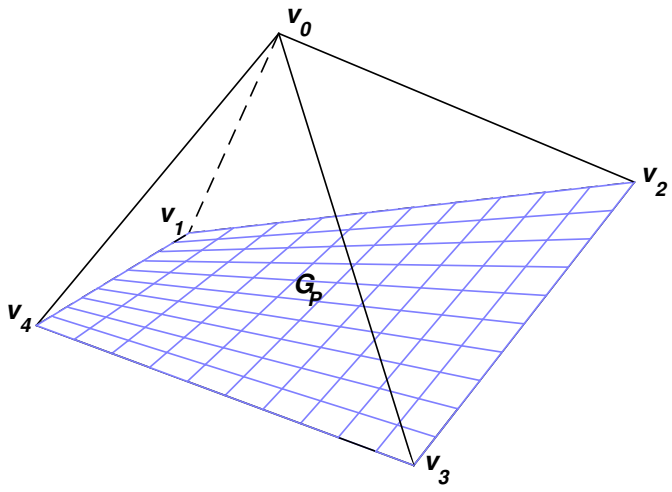

Fig. 1. A typical pyramid: see Definition 2.2.

is a nondegenerate bilinear Bézier patch in $\mathbb{R}^3$ *without self-intersections. Given a point* $v_0 \in \mathbb{R}^3$*, let* P^c *be the polyhedron obtained by taking the convex hull of the points* $v_0, \ldots, v_4$*. Suppose* v_0 *is not in the convex hull* G *of the points* $v_1, \ldots, v_4$*, and is such that* $\langle v_0, v_1, v_2\rangle$*,* $\langle v_0, v_2, v_3\rangle$*,* $\langle v_0, v_3, v_4\rangle$*,* $\langle v_0, v_4, v_1\rangle$*, are nondegenerate triangular faces of* P^c*. We define a* pyramid P *associated with the vertices* $v_0, \ldots, v_4$ *to be the unique bounded, simply connected domain in* $\mathbb{R}^3$ *whose boundary is given by the above triangles and the patch* G_P*.*

Fig. 1 shows a typical pyramid. Note that v_0 cannot lie on the bilinear patch G_P because G_P is contained in G. The points $v_0, \ldots, v_4$ form the vertices of the pyramid P which has: eight edges, each of which is a straight line segment; four triangular faces, each of which is planar; and a base, comprising of the bilinear patch G_P. The base G_P need not necessarily be planar, which means that although the pyramid P may not be convex, it is contained in the convex set P^c. If G_P is a convex planar quadrilateral, then P is an ordinary pyramid.

2.4 THP partitions

We can now define the class of hybrid partitions that will be considered.

Definition 2.3. *Suppose we are given a collection* $\triangle$ *of elements (tetrahedra, pyramids, or hexahedra) such that if any pair of elements have a nonempty intersection* Γ*, then* Γ *is either a single common vertex, a single common edge, or a single common face of both elements. We assume that no faces of a pyramid are shared with another pyramid or with the boundary of* Ω*. Suppose also that the union* Ω *of the elements is a simply connected closed set in* $\mathbb{R}^3$*. Then we say that* $\triangle$ *is a* Tetrahedral-Hexahedral-Pyramidal (THP) *partition.*

Since a nondegenerate bilinear patch cannot coincide with a triangle, the interior faces of $\triangle$ are either triangles shared by two tetrahedra or a tetrahedron and a pyramid, or are bilinear patches shared by two hexahedra or a hexahedron and a pyramid.

Note that the restriction for the pyramids to share no faces with other pyramids or with the boundary is consistent with the typical usage whereby pyramids are used as an interface element between sub-regions meshed using purely tetrahedral and hexahedral elements.

The next step is to define an approximation space on a THP partition $\triangle$ consisting of C^0 functions on Ω whose restrictions to the elements of $\triangle$ are either polynomials or simple maps of polynomials. These restrictions are often called shape functions in the finite element literature. However, it turns out to be impossible to require the shape functions associated with pyramids to be polynomials, see Remark 3. To overcome this problem, we shall not work directly with the THP partition $\triangle$, but shall instead introduce a special refinement obtained by splitting pyramids into so-called interface tetrahedra.

2.5 Splitting a pyramid

Suppose P is a pyramid with vertices $v_0, \ldots, v_4$, where $v_1, \ldots, v_4$ are the vertices of the base G_P defined in (2.3), enumerated in clockwise sense along the boundary of G_P as viewed from the apex v_0. Set $v_P = v_2 + v_4 - v_1 - v_3$, and define interface tetrahedra as follows:

$$P_1 = \phi_{P_1}(T_0), \quad P_2 = \phi_{P_2}(T_0),$$

where

$$\begin{aligned} \phi_{P_1}(\lambda_1, \lambda_2, \lambda_3) &:= \lambda_1 v_1 + \lambda_2 v_3 + \lambda_3 v_2 + \lambda_4 v_0 + \lambda_1\lambda_2 v_P, \\ \phi_{P_2}(\lambda_1, \lambda_2, \lambda_3) &:= \lambda_1 v_1 + \lambda_2 v_3 + \lambda_3 v_4 + \lambda_4 v_0 + \lambda_1\lambda_2 v_P, \end{aligned} \tag{2.4}$$

with

$$\lambda_4 = 1 - \lambda_1 - \lambda_2 - \lambda_3,$$

and

$$T_0 := \{(\lambda_1, \lambda_2, \lambda_3) \in \mathbb{R}^3 : \lambda_1, \lambda_2, \lambda_3 \geq 0 \text{ and } \lambda_1 + \lambda_2 + \lambda_3 \leq 1\}. \tag{2.5}$$

The invertibility of the mappings, as in the case of hexahedra, is a non-trivial issue. Nevertheless, it is of practical importance to establish conditions for the invertibility of the mappings in (2.4) for a given pyramid, and it is to this issue that we now turn.

Lemma 2.4. *Suppose that the points $v_0, \ldots, v_3 \in \mathbb{R}^3$ do not all lie on one plane, and let $v_4 = v_0 + \alpha_1(v_1 - v_0) + \alpha_2(v_2 - v_0) + \alpha_3(v_3 - v_0)$. The mapping $\phi_{P_1} : T_0 \to P_1$ defined in (2.4) is invertible if and only if $\alpha_1 > 0$ and $\alpha_3 > 0$.*

Proof: Although the lemma could be derived after a close examination of the proof of Theorem 2.1 in [19], we give a short proof. We can rewrite

$$\phi_{P_1}(\lambda_1, \lambda_2, \lambda_3) = v_0 + (\lambda_1 + \beta_1\lambda_1\lambda_2)\sigma_1 + (\lambda_2 + \beta_3\lambda_1\lambda_2)\sigma_3 + (\lambda_3 + \beta_2\lambda_1\lambda_2)\sigma_2,$$

where $\sigma_i := v_i - v_0$, $\beta_i := \alpha_i - 1$, and $\lambda_1, \lambda_2, \lambda_3 \geq 0$, $\lambda_1 + \lambda_2 + \lambda_3 \leq 1$. Since the vectors $\sigma_1, \sigma_2, \sigma_3$ are linearly independent, it suffices to consider the invertibility of the mapping

$$\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{pmatrix} \longmapsto \begin{pmatrix} \lambda_1 + \beta_1 \lambda_1 \lambda_2 \\ \lambda_2 + \beta_3 \lambda_1 \lambda_2 \\ \lambda_3 + \beta_2 \lambda_1 \lambda_2 \end{pmatrix}. \tag{2.6}$$

The Jacobian of the mapping is given by $1 + \beta_1 \lambda_2 + \beta_3 \lambda_1$, which is positive for all admissible λ_1, λ_2 if and only if $\alpha_1, \alpha_3 > 0$. The statement now follows from the fact that the invertibility of a multilinear mapping on a set with constant second order derivatives on the boundary is guaranteed if its Jacobian is positive, see the proof of Theorem 37.2 in [12] and page 665 of [19]. □

The next result gives equivalent conditions for the invertibility of the mappings (2.4) in a more readily verifiable form.

Theorem 2.5. *Let $v_0, \ldots, v_4$ be any points in $\mathbb{R}^3$. Assume that the bilinear patch G_P defined by (2.3) is nondegenerate and has no self-intersections. Then the following conditions are equivalent:*

A) *The patch G_P and the vertex v_0 form a pyramid.*

B) *Both mappings ϕ_{P_1} and ϕ_{P_2} defined in (2.4) are invertible.*

C) *Let $\sigma_4 = \alpha_1 \sigma_1 + \alpha_2 \sigma_2 + \alpha_3 \sigma_3$, where $\sigma_i := v_i - v_0$, $i = 1, \ldots, 4$. Then $\alpha_1 > 0$, $\alpha_3 > 0$ and $\alpha_2 < 0$.*

D) *a) v_0 can be separated from $v_1, \ldots, v_4$ by a plane, b) the plane through v_0, v_1, v_3 separates v_2 and v_4, and c) the plane through v_0, v_2, v_4 separates v_1 and v_3.*

Proof: $A \Leftrightarrow C$. If A holds, then the four vertices v_0, v_1, v_2, v_3 cannot be coplanar (since the triangles $\langle v_0, v_1, v_2 \rangle$ and $\langle v_0, v_2, v_3 \rangle$ are the faces of a polyhedron), and thus $\sigma_1, \sigma_2, \sigma_3$ are linearly independent and σ_4 can be written as their linear combination. The triangle $\langle v_0, v_2, v_3 \rangle$ is a face of the polyhedron P^c formed by the convex hull of $v_0, \ldots, v_4$ if and only if v_1 and v_4 lie in the same half-space defined by the plane through v_0, v_2, v_3, which is equivalent to the condition that $\alpha_1 > 0$. Similarly, $\langle v_0, v_1, v_2 \rangle$ is a face of P^c if and only if $\alpha_3 > 0$, $\langle v_0, v_3, v_4 \rangle$ is a face of P^c if and only if $\alpha_1 \alpha_2 < 0$, and $\langle v_0, v_4, v_1 \rangle$ is a face of P^c if and only if $\alpha_2 \alpha_3 < 0$.

$B \Leftrightarrow C$. By Lemma 2.4 the pair of inequalities $\alpha_1 > 0$, $\alpha_3 > 0$ is equivalent to the invertibility of ϕ_{P_1}. By writing $\sigma_2 = -\frac{\alpha_1}{\alpha_2}\sigma_1 + \frac{1}{\alpha_2}\sigma_4 - \frac{\alpha_3}{\alpha_2}\sigma_3$, and applying Lemma 2.4 again, we see that the invertibility of ϕ_{P_2} is characterized by $-\frac{\alpha_1}{\alpha_2} > 0$, $-\frac{\alpha_3}{\alpha_2} > 0$, and thus both ϕ_{P_1} and ϕ_{P_2} are invertible if and only if $\alpha_1 > 0$, $\alpha_3 > 0$, $\alpha_2 < 0$. Note that $B \Leftrightarrow C$ also follows from Theorem 2.1 in [19] whose formulation uses the coefficients of the vector $v_4 - v_2$ in terms of $v_i - v_2$, $i = 0, 1, 3$, in our notation.

$C \Leftrightarrow D$. We first note that b) is equivalent to $\alpha_2 < 0$ since $(\sigma_4, \sigma_1, \sigma_3) = \alpha_2(\sigma_2, \sigma_1, \sigma_3)$, where $(a, b, c) = a \cdot (b \times c)$ denotes the scalar triple product of three vectors. Moreover, from $(\sigma_1, \sigma_2, \sigma_4) = \alpha_3(\sigma_1, \sigma_2, \sigma_3)$ and $(\sigma_3, \sigma_2, \sigma_4) = \alpha_1(\sigma_3, \sigma_2, \sigma_1) = -\alpha_1(\sigma_1, \sigma_2, \sigma_3)$ it follows that c) is equivalent to $\alpha_1 \alpha_3 > 0$. Now,

a) is equivalent to $v_0 \notin \langle v_1, \ldots, v_4 \rangle$, and, on the other hand, to the existence of a vector ν such that the inner products (ν, σ_i), $i = 1, \ldots, 4$, are all positive. Since $(\nu, \sigma_4) = \alpha_1(\nu, \sigma_1) + \alpha_2(\nu, \sigma_2) + \alpha_3(\nu, \sigma_3)$, we conclude that at least one of $\alpha_1, \alpha_2, \alpha_3$ must be positive. □

The interface tetrahedra P_1 and P_2 are each images of the reference tetrahedron T_0 under an invertible quadratic mapping. As such, they are quadratic tetrahedral Bézier volumes, see e.g. [17], with control points located at the vertices and at the midpoints of the straight line segments between vertices, with the exception of $\langle v_1, v_3 \rangle$ whose midpoint is replaced by $\frac{1}{2}(v_2 + v_4)$.

We refer to P_1 and P_2 as interface tetrahedra, although strictly speaking they are not tetrahedra since they have two curved faces. More precisely, P_1 has the two triangular faces $\langle v_0, v_1, v_2 \rangle$ and $\langle v_0, v_2, v_3 \rangle$, while P_2 has the two triangular faces $\langle v_0, v_3, v_4 \rangle$ and $\langle v_0, v_4, v_1 \rangle$. Moreover, P_1 and P_2 share the common face

$$F_P := \Big\{ \lambda_1 v_1 + \lambda_2 v_3 + \lambda_4 v_0 + \lambda_1 \lambda_2 v_P : \lambda_1 + \lambda_2 + \lambda_4 = 1,\ \lambda_1, \lambda_2, \lambda_4 \geq 0 \Big\}.$$

This is a quadratic triangular Bézier patch. Clearly,

$$\phi_{P_1}|_{\{\lambda_3 = 0\}} = \phi_{P_2}|_{\{\lambda_3 = 0\}}. \tag{2.7}$$

The fourth face of P_1 (corresponding to $\lambda_4 = 0$ in (2.4)) is given by

$$F_1 := \{(1-\lambda_1)(1-\lambda_2)v_2 + \lambda_1(1-\lambda_2)v_1 + (1-\lambda_1)\lambda_2 v_3 + \lambda_1\lambda_2 v_4\}, \tag{2.8}$$

while the corresponding face of P_2 is

$$F_2 := \{(1-\lambda_1)(1-\lambda_2)v_4 + \lambda_1(1-\lambda_2)v_1 + (1-\lambda_1)\lambda_2 v_3 + \lambda_1\lambda_2 v_2\}. \tag{2.9}$$

These faces are both quadratic triangular Bézier patches. The union of F_1 and F_2 is the base G_P of the pyramid P. Clearly, P_1 and P_2 define a *refinement* of P, i.e., $P = P_1 \cup P_2$ with $P_1 \cap P_2 = F_P$. Fig. 2 shows a typical split of a pyramid into two interface tetrahedra.

The inverse of the mapping (2.6) can be easily evaluated. Indeed, let $\mu_1 = \lambda_1 + \beta_1\lambda_1\lambda_2$, $\mu_2 = \lambda_2 + \beta_3\lambda_1\lambda_2$, $\mu_3 = \lambda_3 + \beta_2\lambda_1\lambda_2$. Then λ_1 is a solution of the quadratic equation

$$\beta_3\lambda_1^2 + (\beta_1 + \beta_1\mu_2 - \beta_3\mu_1)\lambda_1 - \beta_1\mu_1 = 0,$$

whereas $\lambda_2 = \mu_2 - \frac{\beta_3}{\beta_1}(\mu_1 - \lambda_1)$, $\lambda_3 = \mu_3 - \beta_2\lambda_1\lambda_2$. The coefficients β_1, β_3 can be obtained directly from the coordinates of the vertices by using the formulae $\beta_1 = (\sigma_4, \sigma_2, \sigma_3)/(\sigma_1, \sigma_2, \sigma_3) - 1$, $\beta_3 = (\sigma_1, \sigma_2, \sigma_4)/(\sigma_1, \sigma_2, \sigma_3) - 1$. This inverse mapping can be used to determine in which of the two interface tetrahedra a given point $x \in P$ lies.

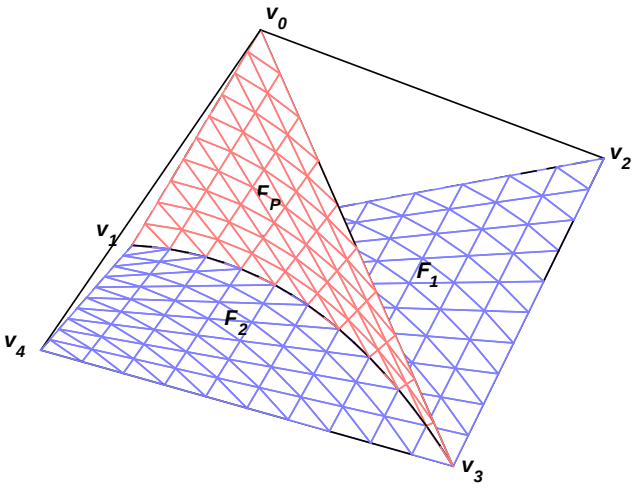

Fig. 2. A typical split of a pyramid into two interface tetrahedra.

2.6 Attaching a pyramid to a hexahedron

Consider a face F of the hexahedron $H = \phi_H(H_0)$, and suppose without loss of generality that F corresponds to $\lambda_3 = 1$, and thus has vertices $v_1 = v_{101}$, $v_2 = v_{001}$, $v_3 = v_{011}$, and $v_4 = v_{111}$. Then

$$F = \{(1-\lambda_1)(1-\lambda_2)v_2 + \lambda_1(1-\lambda_2)v_1 + (1-\lambda_1)\lambda_2 v_3 + \lambda_1\lambda_2 v_4 : (\lambda_1, \lambda_2) \in [0,1]^2\}.$$

Let P be a pyramid whose base is defined by these four vertices, i.e., attached to the top of H. Suppose P_1 and P_2 are the two associated interface tetrahedra. Then

$$\phi_H|_{\{(\lambda_1,\lambda_2,1):\,\lambda_1+\lambda_2\le 1\}} = \phi_{P_1}|_{\{(\lambda_1,\lambda_2,1-\lambda_1-\lambda_2,0):\,\lambda_1+\lambda_2\le 1\}}. \tag{2.10}$$

i.e., the face F_1 of P_1 corresponding to $\lambda_4 = 0$, see (2.8), lies on the part of the face F of H corresponding to $\lambda_1, \lambda_2 \ge 0$ with $\lambda_1 + \lambda_2 \le 1$. This is a quadratic triangular Bézier patch with the two straight edges $\langle v_2, v_1\rangle$ and $\langle v_2, v_3\rangle$, and a third edge which is the quadratic Bézier curve

$$\delta(\lambda) := \lambda^2 v_1 + \lambda(1-\lambda)(v_2 + v_4) + (1-\lambda)^2 v_3, \qquad 0 \le \lambda \le 1,$$

with control points $v_1, (v_2 + v_4)/2, v_3$.

We call the curve δ a hanging edge of the refined partition since it lies in the interior of a face of a hexahedron and as such does not match with any edge of the neighbouring element. The face F_2 also lies on F, and is a triangular Bézier patch with two straight edges $\langle v_4, v_1\rangle$ and $\langle v_4, v_3\rangle$, and the third edge δ, such that $F = F_1 \cup F_2$. In addition, similar to (2.10), the mappings ϕ_H and ϕ_{P_2} can be identified on the other half of F. Note that a pyramid could equally well be split with δ running from v_2 to v_4 instead of from v_1 to v_3.

2.7 Refining a THP partition

Given a THP-partition $\triangle$, let $\triangle_R$ be a refinement obtained by splitting each pyramid into two interface tetrahedra. This eliminates the pyramids, and the elements of this partition are now either tetrahedra or hexahedra, where a tetrahedron can be either an ordinary tetrahedron or an interface tetrahedron.

Effectively eliminating the pyramids comes at the price that the partition now contains hanging edges, and some of the edges of $\triangle_R$ may be curved, namely, the hanging edges which arise when we split pyramids.

There are three kinds of faces of $\triangle_R$.

1) The planar triangles which originally were faces of the initial THP partition $\triangle$. All of the faces of ordinary tetrahedra are triangles, as are two of the faces of each interface tetrahedra. They can be on the boundary of Ω, or can be shared by two tetrahedra.
2) The quadratic triangular Bézier patches which arise in splitting pyramids. Each interface tetrahedron has two of these. They can be the common face of two interface tetrahedra, or can lie in the bilinear face of a hexahedron.
3) The bilinear patches which were faces of the hexahedra in the initial THP partition $\triangle$.

Note that faces of type 2 can lie in faces of type 3.

§3. The approximation space

Given an integer $d > 0$, let $\mathcal{P}_d$ be the usual space of trivariate polynomials of total degree d, and let $\mathcal{Q}_d$ be the space of trivariate tensor-product polynomials of degree d in each coordinate. For each hexahedron H of $\triangle_R$, let

$$\mathcal{P}_H := \{p \circ \phi_H^{-1} : p \in \mathcal{Q}_d\},$$

and for each interface tetrahedron T of $\triangle_R$, let

$$\mathcal{P}_T := \{p \circ \phi_T^{-1} : p \in \mathcal{P}_{2d}\}.$$

Then we define

$$\begin{aligned} \mathcal{S}_d^0(\triangle) := \{s \in C^0(\Omega) : \; & s|_H \in \mathcal{P}_H \text{ for each hexahedron } H \text{ of } \triangle_R, \\ & s|_T \in \mathcal{P}_d \text{ for each ordinary tetrahedron } T \text{ of } \triangle_R, \\ & s|_T \in \mathcal{P}_T \text{ for each interface tetrahedron } T \text{ of } \triangle_R \; \}. \end{aligned} \tag{3.1}$$

Clearly, $\mathcal{S}_d^0(\triangle)$ is a finite dimensional linear space. We give a formula for its dimension in Sect. 6. It may be regarded as a kind of spline space in the sense that it consists of piecewise functions defined on the partition $\triangle$ which are joined together with C^0 continuity.

§4. Bernstein-Bézier shape functions

In this section we describe the shape functions for the space $\mathcal{S}_d^0(\triangle)$. First we introduce some notation.

4.1. Notation

We use standard multi-index notation. Thus if $\nu = (\nu_1, \ldots, \nu_m) \in \mathbb{Z}_+^m$ is a set of nonnegative integers, we write $|\nu| = \sum_{i=1}^m \nu_i$, and define

$$\nu! := \nu_1! \cdots \nu_m!$$

and

$$\binom{\nu + \mu}{\mu} := \frac{(\nu + \mu)!}{\nu!\, \mu!}.$$

The inequality $\nu \leq \mu$ means that $\nu_i \leq \mu_i$, $i = 1, \ldots, m$. We need the following index sets

$$\mathcal{I}_m^n := \{\nu \in \mathbb{Z}_+^{m+1} : |\nu| = n\}$$

and

$$\mathcal{J}_3^{(n,m,\ell)} := \{(i,j,k) \in \mathbb{Z}_+^3 : i \leq n,\ j \leq m,\ k \leq \ell\}.$$

Given $d > 0$, we describe some convenient basis functions for the spaces $\mathcal{P}_d^1$, $\mathcal{P}_d^2$, and $\mathcal{P}_d^3$ of polynomials of total degree at most d in one, two, and three variables. First, for an interval $I := [0,1]$, we define the classical Bernstein basis polynomials of degree d as

$$B_{ij}^{d,I}(\lambda) := \frac{d!}{i!\, j!}(1-\lambda)^i \lambda^j, \qquad (i,j) \in \mathcal{I}_1^d. \tag{4.1}$$

Now suppose $F = \langle v_1, v_2, v_3 \rangle$ is a triangle with vertices v_1, v_2, v_3, and let d be a positive integer. Given any point $v \in \mathbb{R}^3$, let $(\lambda_1, \lambda_2, \lambda_3)$ be its barycentric coordinates relative to F, i.e., $v = \lambda_1 v_1 + \cdots + \lambda_3 v_3$. Then the associated bivariate Bernstein basis polynomials are defined to be

$$B_{ijk}^{d,F}(\lambda_1, \lambda_2, \lambda_3) := \frac{d!}{i!\, j!\, k!} \lambda_1^i \lambda_2^j \lambda_3^k, \qquad (i,j,k) \in \mathcal{I}_2^d. \tag{4.2}$$

These functions form a basis for $\mathcal{P}_d^2$, and and have many attractive properties that can be exploited for stable and efficient computation, see [21]. In particular, on F they are nonnegative and form a partition of unity.

We can define trivariate Bernstein basis polynomials in a similar way on a tetrahedron $T = \langle v_1, v_2, v_3, v_4 \rangle$. Given any point $v \in \mathbb{R}^3$, let $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ be its barycentric coordinates relative to T, i.e., $v = \lambda_1 v_1 + \cdots + \lambda_4 v_4$. Then the associated trivariate Bernstein basis polynomials are defined to be

$$B_{ijkl}^{d,T}(\lambda_1, \lambda_2, \lambda_3, \lambda_4) := \frac{d!}{i!\, j!\, k!\, l!} \lambda_1^i \lambda_2^j \lambda_3^k \lambda_4^l, \qquad (i,j,k,l) \in \mathcal{I}_3^d. \tag{4.3}$$

These functions form a basis for $\mathcal{P}_d^3$, and also have many nice properties, see [21]. In particular, on T they are nonnegative and form a partition of unity.

We shall also make use of tensor-product Bernstein basis polynomials. In two variables we define them by

$$B_{ij}^{n,m}(\lambda_1,\lambda_2) := B_i^n(\lambda_1)B_j^m(\lambda_2), \quad (\lambda_1,\lambda_2)\in[0,1]^2, \tag{4.4}$$

for $0\le i\le n$ and $0\le j\le m$. In three variables we define them as

$$B_{ijk}^{n,m,\ell}(\lambda_1,\lambda_2,\lambda_3) := B_i^n(\lambda_1)B_j^m(\lambda_2)B_k^\ell(\lambda_3), \quad (\lambda_1,\lambda_2,\lambda_3)\in[0,1]^3, \tag{4.5}$$

for $0\le i\le n$, $0\le j\le m$, and $0\le k\le \ell$.

We are now ready to define shape functions for the three kinds of elements contained in $\triangle_R$.

4.2. Shape functions for an ordinary tetrahedron

Let $T := \langle v_1, v_2, v_3, v_4\rangle$ be a nondegenerate ordinary tetrahedron. For each point $v\in\mathbb{R}^3$, let $(\lambda_1,\lambda_2,\lambda_3,\lambda_4)$ denote its barycentric coordinates relative to T, and set

$$\psi_{ijkl}^{d,T}(v) := B_{ijkl}^{d,T}(\lambda_1,\lambda_2,\lambda_3,\lambda_4), \qquad (i,j,k,l)\in\mathcal{I}_3^d, \tag{4.6}$$

where $B_{ijkl}^{d,T}$ are the Bernstein basis polynomials defined in (4.3).

Instead of indexing the shape functions as in (4.6), for later use it is convenient to introduce an alternative indexing scheme used frequently in the spline literature, see [21]. Given T and d as above, the set of associated domain points is defined as

$$\mathcal{D}_{d,T} := \left\{\xi_{ijkl}^{d,T} := \frac{iv_1+jv_2+kv_3+lv_4}{d}\right\}_{i+j+k=d}.$$

For each $\xi=\xi_{ijkl}^T\in\mathcal{D}_{d,T}$, we now set $\psi_\xi^{d,T} := \psi_{ijkl}^{d,T}$. Then for any $s\in\mathcal{S}_d^0(\triangle)$ we can write

$$s|_T = \sum_{\xi\in\mathcal{D}_{d,T}} c_\xi^T\,\psi_\xi^{d,T}. \tag{4.7}$$

We call the coefficients c_ξ^T the B-coefficients of $s|_T$.

4.3. Shape functions for a hexahedron

Let $H=\phi_H(H_0)$ be a general hexahedral element as defined in Sect. 2. If H is a rectangular box with edges parallel to the axes, we can use tensor-product polynomials for the shape functions associated with H. For general H, we map back to H_0 using the inverse of the map ϕ_H. In particular, we define the shape functions associated with H to be

$$\psi_{ijk}^{d,H}(v) := B_{ijk}^{d,d,d}(\phi_H^{-1}(v)), \qquad 0\le i,j,k\le d, \tag{4.8}$$

where $B^{d,d,d}_{ijk}$ are the tensor-product Bernstein polynomials defined in (4.5) on $H_0 = [0,1]^3$.

Just as was done for tetrahedra, we would like to index these shape functions using certain domain points. Let

$$\mathcal{D}_{d,H_0} := \{\xi^{d,H_0}_{ijk} := (i/d, j/d, k/d) : 0 \le i,j,k \le d\}.$$

Then we define the domain points associated with H as $\mathcal{D}_{d,H} = \phi_H(\mathcal{D}_{d,H_0})$. Note that these points are in H, not in H_0. In this case we write $\psi^{d,H}_{\xi} = \psi^{d,H}_{ijk}$ whenever $\xi = \xi^{d,H}_{ijk}$. For any $s \in \mathcal{S}^0_d(\triangle)$, we can now write

$$s|_H = \sum_{\xi \in \mathcal{D}_{d,H}} c^H_\xi \, \psi^{d,H}_\xi. \tag{4.9}$$

We call the coefficients c^H_ξ the B-coefficients of $s|_H$.

4.4. Shape functions for an interface tetrahedron

Suppose P_1 and P_2 are the interface tetrahedra associated with a pyramid P. Now for each $v \in P_1$, we define

$$\psi^{2d,P_1}_{ijkl}(v) := B^{2d}_{ijkl}(\phi^{-1}_{P_1}(v)), \qquad (i,j,k,l) \in \mathcal{I}^{2d}_3, \tag{4.10}$$

where B^{2d}_{ijkl} are the trivariate Bernstein basis polynomials of degree $2d$ associated with the reference tetrahedron T_0 defined in (2.5). Instead of indexing these shape functions with integer subscripts, we can follow the idea of Sect. 4.2 and index them with the set of domain points $\mathcal{D}_{2d,P_1} := \phi_{P_1}(\mathcal{D}_{2d,T_0})$. Then for any spline $s \in \mathcal{S}^0_d(\triangle)$, we can write

$$s|_{P_1} = \sum_{\xi \in \mathcal{D}_{2d,P_1}} c^{P_1}_\xi \, \psi^{2d,P_1}_\xi. \tag{4.11}$$

We call the $c^{P_1}_\xi$ the B-coefficients of $s|_{P_1}$. Shape functions and domain points for the interface tetrahedron P_2 can be defined in exactly the same way.

§5. Domain points and a minimal determining set for $\mathcal{S}^0_d(\triangle)$

The concept of domain points and minimal determining sets is frequently used in the analysis of splines, and play a similar role to the notion of degrees of freedom in the finite element community. Let $\triangle$ be a THP partition, and suppose $\triangle_R$ is its refinement as defined in Sect. 2.7. We define the set of domain points $\mathcal{D}_{d,\triangle_R}$ associated with $\triangle_R$ to be the union of the sets of domain points associated with the elements of $\triangle_R$, but with no repetitions. To describe $\mathcal{D}_{d,\triangle_R}$ more precisely, we need some additional notation:

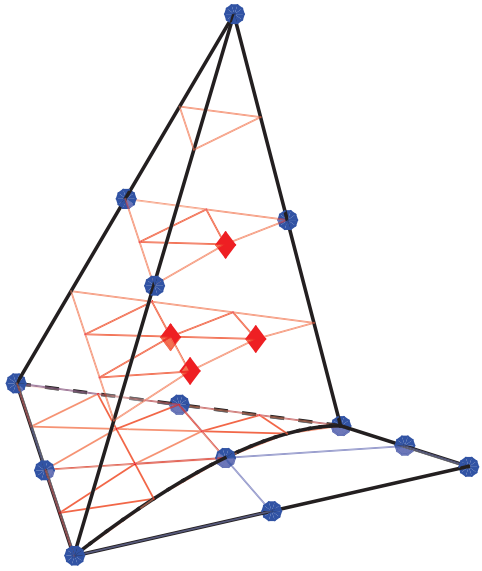

Fig. 3. Domain points associated with an interface tetrahedron attached to the top of a hexahedron.

1) For each vertex v, let $\mathcal{D}_v = \{v\}$.
2) For each edge e, let $\mathcal{D}^\circ_{d,e}$ be the set of $d-1$ equally spaced points in the interior of e.
3) For each triangular face F of an ordinary tetrahedron T, let $\mathcal{D}^\circ_{d,F}$ be the set of $\binom{d-1}{2}$ domain points in $\mathcal{D}_{d,T}$ lying in the interior of F.
4) For each face F shared by an interface tetrahedron $\widetilde{T}$ with another interface tetrahedron, let $\mathcal{D}^\circ_{2d,F}$ be the set of $\binom{2d-1}{2}$ domain points in $\mathcal{D}_{2d,\widetilde{T}}$ lying in the interior of F.
5) For each bilinear face R of a hexahedron, let $\mathcal{D}^\circ_{d,R}$ be the set of $(d-1)^2$ domain points in $\mathcal{D}_{d,H}$ lying in the interior of R.
6) For each ordinary tetrahedron T, let $\mathcal{D}^\circ_{d,T}$ be the set of domain points in $\mathcal{D}_{d,T}$ that lie in the interior of T.
7) For each interface tetrahedron $\widetilde{T}$, let $\mathcal{D}^\circ_{2d,\widetilde{T}}$ be the set of $\binom{2d-1}{3}$ domain points in $\mathcal{D}_{2d,\widetilde{T}}$ that lie in the interior of $\widetilde{T}$.
8) For each hexahedron H, let $\mathcal{D}^\circ_{d,H}$ be the set of $(d-1)^3$ domain points of $\mathcal{D}_{d,H}$ that lie in the interior of H.

The superscript circles in this notation are to remind us that the points in the corresponding sets are in the interiors of the indicated domains.

To help understand where these points lie, in Fig. 3 we show an interface tetrahedron $\widetilde{T}$ joined to the top of a hexahedron H. For $d = 2$, there are nine domain

points lying on the top face of the hexahedron: these are shown as blue dots, and belong to $\mathcal{D}_{d,H}$. Assuming this configuration is connected to an ordinary tetrahedron on the left and on the back, we can think of the six blue dots on the left and back triangular faces as belonging to sets of the form $\mathcal{D}_{d,T}$. For the curved face F of this interface tetrahedron, there are three points in the set $\mathcal{D}^\circ_{2d,F}$ – these are shown as red diamonds. Finally, there is one point in the set $\mathcal{D}^\circ_{2d,\widetilde{T}}$ – it is also shown as a red diamond. Four red networks in the interface tetrahedron indicate the positions of the domain points in $\mathcal{D}_{2d,\widetilde{T}}$.

Let $\mathcal{V}$ and $\mathcal{E}$ be the sets of vertices and edges of $\triangle$. In addition, let $\mathcal{F}_O$ be the set of faces of ordinary tetrahedra, and let $\mathcal{F}_I$ be the set of faces shared by two interface tetrahedra. Let $\mathcal{F}_R$ be the set of all bilinear faces of $\triangle$. Finally, let $\mathcal{H}$, $\mathcal{T}_O$, and $\mathcal{T}_I$ be the sets of all hexahedra, ordinary tetrahedra, and interface tetrahedra of $\triangle_R$, respectively. We now define

$$\begin{aligned}\mathcal{D}_{d,\triangle} := \bigcup_{v\in\mathcal{V}} \mathcal{D}_v \cup \bigcup_{e\in\mathcal{E}} \mathcal{D}^\circ_{d,e} \cup \bigcup_{F\in\mathcal{F}_O} \mathcal{D}^\circ_{d,F} \cup \bigcup_{F\in\mathcal{F}_I} \mathcal{D}^\circ_{2d,F} \cup \bigcup_{R\in\mathcal{F}_R} \mathcal{D}^\circ_{d,R} \\ \cup \bigcup_{T\in\mathcal{T}_O} \mathcal{D}^\circ_{d,T} \cup \bigcup_{\widetilde{T}\in\mathcal{T}_I} \mathcal{D}^\circ_{2d,\widetilde{T}} \cup \bigcup_{H\in\mathcal{H}} \mathcal{D}^\circ_{d,H}.\end{aligned} \tag{5.1}$$

The sets in (5.1) are pairwise disjoint. We can also write

$$\mathcal{D}_{d,\triangle} = \bigcup_{F\in\mathcal{F}_I} \mathcal{D}^\circ_{2d,F} \cup \bigcup_{T\in\mathcal{T}_O} \mathcal{D}_{d,T} \cup \bigcup_{\widetilde{T}\in\mathcal{T}_I} \mathcal{D}^\circ_{2d,\widetilde{T}} \cup \bigcup_{H\in\mathcal{H}} \mathcal{D}_{d,H}, \tag{5.2}$$

with the proviso that repeated points should be identified as a single point in the set.

We now describe how to parametrize a spline $s \in \mathcal{S}^0_d(\triangle)$ in terms of the coefficients of the expansions of Sect 4. The following definitions follow standard spline usage, see [21].

Definition 5.1. *For each point $\xi \in \mathcal{D}_{d,\triangle}$, pick some element $E \in \triangle_R$ containing ξ, and let c_ξ be the coefficient associated with the domain point ξ in the shape-function expansion of s on E. Then we call $\{c_\xi\}_{\xi\in\mathcal{D}_{d,\triangle}}$ the set of* B-coefficients *of s. A set $\mathcal{M} \subseteq \mathcal{D}_{d,\triangle}$ is called a* determining set *for $\mathcal{S}^0_d(\triangle)$ provided that if we set $c_\xi = 0$ for all $\xi \in \mathcal{M}$, then $s \equiv 0$. Moreover, if $\mathcal{M}$ is a smallest such set, it is called a* minimal determining set *for $\mathcal{S}^0_d(\triangle)$.*

The minimal determining set is important in practical finite element analysis because it may be identified with the global degrees of freedom in the finite element space.

Theorem 5.2. *The set $\mathcal{M} := \mathcal{D}_{d,\triangle_R}$ is a determining set for $\mathcal{S}^0_d(\triangle)$.*

Proof: Let $s \in \mathcal{S}^0_d(\triangle)$, and suppose we assign a value of zero to all of the B-coefficients associated with domain points in $\mathcal{M}$. Let H be a hexahedron. Then

since $\mathcal{M}$ contains $\mathcal{D}_{d,H}$, it is clear that $s|_H \equiv 0$. Similarly, if T is an ordinary tetrahedron, then since $\mathcal{M}$ contains $\mathcal{D}_{d,T}$, $s|_T \equiv 0$. Now consider an interface tetrahedron $\widetilde{T}$. We already know that s vanishes on the triangular faces of $\widetilde{T}$ where $\widetilde{T}$ joins an ordinary tetrahedron. It also vanishes on the face that lies in a neighboring hexahedron. Now if F is the Bézier patch forming the boundary between $\widetilde{T}$ and another interface tetrahedron, then $\mathcal{M}$ contains $\mathcal{D}_{2d,F}$, and so s also vanishes on this patch. We can now conclude that s also vanishes on all of $\widetilde{T}$ since $\mathcal{D}_{d,\widetilde{T}} \subset \mathcal{M}$. □

Theorem 5.3. *The set $\mathcal{M}$ is a minimal determining set for $\mathcal{S}_d^0(\triangle)$.*

Proof: We show that the set $\mathcal{M}$ is consistent in the sense of Definition 5.14 of [21], i.e., if we fix the coefficients $\{c_\xi\}_{\xi\in\mathcal{M}}$, then all pieces of s are uniquely determined and join together with C^0 continuity. Suppose H is a hexahedron. Then since we have set the coefficients associated with all of the domain points at the vertices, edges, and faces of H, as well as those in the set $\mathcal{D}^\circ_{d,H}$, it follows that s is uniquely defined on H by the expansion in (4.9). Similarly, if T is an ordinary tetrahedron, then since we have set the coefficients corresponding to all of the domain points in T, we see that s is uniquely defined on T by the expansion (4.7).

Suppose now that $\widetilde{T}$ is an interface tetrahedron, and let F be a triangular face of $\widetilde{T}$ shared with an ordinary tetrahedron T. Then we have already set the coefficients of s corresponding to domain points in $\mathcal{D}_{d,T} \cap F$. But using degree raising, this uniquely determines the coefficients of s corresponding to the domain points in $\mathcal{D}_{2d,\widetilde{T}} \cap F$, and by C^0 continuity these values are consistent. Now suppose F is the face of $\widetilde{T}$ which lies on the face R of a hexahedron H. We already know the coefficients of the bivariate $d \times d$ tensor-product polynomial $s|_R \circ \phi_H^{-1}$. Using Lemma 7.2 below, these can be uniquely expressed as coefficients of a polynomial of total degree $2d$, and so the coefficients of $s_{\widetilde{T}}$ associated with F are determined by C^0 continuity. We have shown that all B-coefficients of $s|_{\widetilde{T}}$ are consistently determined. □

§6. A dimension formula and a basis for $\mathcal{S}_d^0(\triangle)$

Our next result gives a formula for the dimension of the finite element space $\mathcal{S}_d^0(\triangle)$ in terms of the following combinatorial quantities associated with the partition:

$$
\begin{aligned}
n_V &:= \text{number of vertices,}\\
n_E &:= \text{number of edges,}\\
n_T &:= \text{number of ordinary tetrahedra,}\\
n_P &:= \text{number of pyramids,}\\
n_H &:= \text{number of hexahedra.}
\end{aligned}
$$

Theorem 6.1. *For any $d \geq 1$,*

$$\begin{aligned} n_d := \dim \mathcal{S}_d^0(\triangle) = n_V + (d-1)n_E + \Big[4\binom{d-1}{2} + \binom{d-1}{3}\Big]n_T \\ + \Big[\binom{2d-1}{2} + 2\binom{2d-1}{3}\Big]n_P + \Big[6(d-1)^2 + (d-1)^3\Big]n_H. \end{aligned} \tag{6.1}$$

Proof: The fact that $\mathcal{M}$ is a minimal determining set for $\mathcal{S}_d^0(\triangle)$ implies that $\dim \mathcal{S}_d^0(\triangle) = \#\mathcal{M}$, which is easily seen to be given by the formula in (6.1). □

We now describe a basis for $\mathcal{S}_d^0(\triangle)$ that is suitable for the finite element implementation. For any domain point $\xi \in \mathcal{M}$ and any spline $s \in \mathcal{S}_d^0(\triangle)$, let $\lambda_\xi s = c_\xi$, where c_ξ is the B-coefficient of s associated with ξ. Then it follows from the fact that $\mathcal{M}$ is a minimal determining set that for each $\xi \in \mathcal{M}$, there exists a unique spline $\psi_\xi \in \mathcal{S}_d^0(\triangle)$ such that

$$\lambda_\eta \psi_\xi = \delta_{\xi,\eta}, \qquad \text{all } \eta \in \mathcal{M}. \tag{6.2}$$

This expression is reminiscent of the Lagrange bases commonly used in low order finite element approximations. Indeed, one will not go far wrong in thinking of the Bernstein-Bézier basis in this fashion, but it is important to realize that the B-coefficients *do not* correspond to the values of the function at the corresponding domain points.

Theorem 6.2. *The set $\Psi := \{\psi_\xi\}_{\xi \in \mathcal{M}}$ is a basis for $\mathcal{S}_d^0(\triangle)$. Moreover, for each $\xi \in \mathcal{M}$, the support of ψ_ξ is given by*

$$\sigma(\psi_\xi) = \textit{the union of all of the elements of } \triangle \textit{ that contain the point } \xi.$$

Proof: The splines in Ψ are clearly linearly independent in view of the dual property (6.2). Now if Ω_k is a hexahedron or ordinary tetrahedron that does not contain ξ, then all of its coefficients must be zero, and so $\psi_\xi|_{\Omega_k} \equiv 0$. If Ω_k is an interface tetrahedron which is not part of a pyramid containing ξ, then $\psi_\xi|_{\Omega_k} \equiv 0$. This establishes the result on the supports. □

The basis functions in this theorem all have small supports, which ensures that the standard finite element sub-assembly procedures remain efficient. For example if ξ is at a vertex of $\triangle$, the support of ψ_ξ is just the collection of elements sharing that vertex. Similarly, if ξ lies in the interior of an edge e of $\triangle$, we get the collection of elements sharing that edge. If ξ lies in the interior of a face shared by a pair of elements, the support is just the set obtained by taking the union of those two elements. Finally, if ξ is in the interior of a hexahedron, an ordinary tetrahedron or a pyramid, the support is just that single element of $\triangle$.

Note that the domain points in $\mathcal{D}_{d,T}$ for an ordinary tetrahedron T, $\mathcal{D}_{2d,T}$ for an interface tetrahedron T, or $\mathcal{D}_{d,H}$ for a hexahedron H can be identified with the local degrees of freedom of the corresponding element. The set $\mathcal{M}$ can be identified with the global degrees of freedom of $\mathcal{S}_d^0(\triangle)$.

§7. Computing with Bernstein basis polynomials

We now turn to the issue of how the basis developed in the previous section lends itself to effective algorithms for the computation of the various entities needed for the finite element analysis. We shall be particularly interested in the dependence of the complexity estimates on the polynomial degree d.

7.1. Degree raising

One attractive property of the Bernstein-Bézier basis is a simple formula for expressing a polynomial of degree d as a combination of Bernstein polynomials of degree $d+1$, a process known as *degree raising* in the spline literature.

Lemma 7.1. *Let F be a triangle, and suppose p is a bivariate polynomial of degree d with B-coefficients $\{c_{ijk}\}_{i+j+k=d}$. Then p can be rewritten as*

$$p = \sum_{(i,j,k)\in I_2^{d+1}} \tilde{c}_{ijk} B_{ijk}^{d+1,F},$$

where

$$\tilde{c}_{ijk} = \frac{ic_{i-1,j,k} + jc_{i,j-1,k} + kc_{i,j,k-1}}{d+1}, \qquad (i,j,k) \in I_2^{d+1}. \tag{7.1}$$

For a proof, see Sect. 2.15 of [21]. This result can be interpreted in terms of matrix multiplication. Suppose we insert the B-coefficients c_{ijk} and $\tilde{c}_{ijk}$ in the above lemma into vectors c of length n and $\tilde{c}$ of length $\tilde{n}$, using the lexicographical order. Then there exists a $\binom{d+3}{2} \times \binom{d+2}{2}$ matrix $R^{d,d+1}$ such that

$$\tilde{c} = R^{d,d+1} c.$$

This matrix is sparse since each row contains just three entries corresponding to the factors appearing in (7.1). For example, in the lowest order case,

$$R^{1,2} := \begin{pmatrix} 1 & 0 & 0 \\ .5 & .5 & 0 \\ .5 & 0 & .5 \\ 0 & 1 & 0 \\ 0 & .5 & .5 \\ 0 & 0 & 1 \end{pmatrix}.$$

More generally, one can apply degree raising to a polynomial of degree n to any higher degree m through a succession of single degree raises, corresponding to multiplication by the matrix

$$R^{n,m} := R^{m-1,m} \cdots R^{n,n+1}. \tag{7.2}$$

It is straightforward to precompute and store $R^{n,m}$, see Remark 12, so that $\tilde{c} = R^{n,m}c$ can be computed from c by matrix multiplication. However, a simple count shows that for large $m - n$ it is more efficient to compute $\tilde{c}$ by repeated use of the formula (7.1), or equivalently by repeated multiplications by the sparse matrices $R^{i,i+1}$ appearing in (7.2). Indeed, we note that the number of nonzeros in any row of $R^{n,n+\ell}$ is equal to $\#\mathcal{I}_2^\ell$, which is three in the case $\ell = 1$ corresponding to a single step of degree raising by one. The sparse matrix-vector multiplication $R^{n,n+\ell}c$ costs

$$\#\mathcal{I}_2^n \cdot \#\mathcal{I}_2^\ell = \binom{n+2}{2}\binom{\ell+2}{2} = \frac{1}{4}n^2\ell^2 + \frac{3}{4}(n^2\ell + n\ell^2) + \mathcal{O}(n^2 + n\ell + \ell^2)$$

floating point operations (counting only multiplications) if done directly. However, it is performed more efficiently for large ℓ if replaced by ℓ multiplications by the sparse matrices $R^{i,i+1}$, $i = n, \ldots, n+\ell-1$, that is by the step-by-step degree raising, with total cost

$$\begin{aligned} 3\sum_{i=n+1}^{n+\ell}\binom{i+2}{2} &= 3\binom{n+\ell+3}{3} - 3\binom{n+3}{3} \\ &= \frac{1}{2}\ell[(n+\ell+2)(n+\ell+3) + (n+1)(2n+\ell+5)] \\ &= \frac{1}{2}\ell(3n^2 + 3n\ell + \ell^2) + \mathcal{O}(n^2 + n\ell + \ell^2). \end{aligned}$$

The same count applies to $(R^{n,n+\ell})^t c$ computed as $(R^{n,n+1})^t \cdots (R^{n+\ell-1,n+\ell})^t c$.

For present purposes, we need to degree raise a polynomial written as a combination of Bernstein polynomials of degree d relative to a face F of an ordinary tetrahedron, to obtain its coefficients as a combination of Bernstein polynomials $B_{ijk}^{2d,F}$ of degree $2d$ relative to F considered as a face of a neighboring interface tetrahedron $\widetilde{T}$:

Lemma 7.2. *Let T be an ordinary tetrahedron that shares a triangular face F with an interface tetrahedron $\widetilde{T}$. Suppose we insert the B-coefficients of s corresponding to the domain points in $\mathcal{D}_{d,F} = \mathcal{D}_{d,T} \cap F$ into a vector $c := (c_1, \ldots, c_n)$, where $n := \binom{d+2}{2}$. Let $\tilde{c} := (\tilde{c}_1, \ldots, \tilde{c}_{\tilde{n}})$ with $\tilde{n} := \binom{2d+2}{2}$ be the coefficients of $s|_{\widetilde{T}} \in \mathcal{P}_{2d}$ corresponding to domain points in $\mathcal{D}_{2d,F} = \mathcal{D}_{2d,\widetilde{T}} \cap F$. Then there exists an $\tilde{n} \times n$ matrix U_d such that*

$$\tilde{c} = U_d\, c. \tag{7.3}$$

Proof: *The matrix U_d can be obtained from $R^{d,2d}$ after permuting its rows and columns to take account of the ordering of the coefficients.* □

It follows from the above that if we multiply the matrix U_d or $(U_d)^t$ directly with a vector, then the cost is $\frac{1}{4}d^4 + \frac{3}{2}d^3 + \mathcal{O}(d^2)$ operations. If we instead perform step-by-step degree raising, then the cost is $\frac{7}{2}d^3 + \mathcal{O}(d^2)$. A more precise operation count shows that the break even point occurs when $d = 10$.

7.2. Conversion from tensor-product to total degree form

We also need to express a tensor-product polynomial of degree d associated with a bilinear face R of a hexagon as a combination of the Bernstein polynomials of degree $2d$ associated with the bottom face F of an interface tetrahedron. The following lemma can be used for this conversion, compare [16]. Suppose $B_{ij}^{k,s}$ are the tensor-product Bernstein basis polynomials on $[0,1]^2$, see (4.4), and let $B_\mu^m := B_\mu^{m,F_0}$, $\mu = (\mu_1, \mu_2, \mu_3) \in \mathcal{I}_2^m$ be the bivariate Bernstein basis polynomials defined by (4.2) on the triangle F_0 with vertices at $(1,0), (0,1), (0,0)$.

Lemma 7.3. *For any positive integers k, s and any set of B-coefficients $\{c_{ij}\}$,*

$$\sum_{i=0}^{k}\sum_{j=0}^{s} c_{ij} B_{ij}^{k,s} = \sum_{\mu\in\mathcal{I}_2^{k+s}} \tilde{c}_\mu B_\mu^{k+s,F_0},$$

where

$$\tilde{c}_\mu = \sum_{\substack{\kappa\in\mathcal{I}_2^k \\ \kappa\le\mu}} \frac{\binom{\mu}{\kappa}}{\binom{k+s}{k}} c_{\kappa_1,\mu_2-\kappa_2}, \quad \mu\in\mathcal{I}_2^{k+s}. \tag{7.4}$$

Proof: We have

$$B_{ij}^{k,s}(\lambda) = B_{i,k-i}^{k,I}(\lambda_1) B_{j,s-j}^{s,I}(\lambda_2) = \sum_{\substack{\kappa\in\mathcal{I}_2^k \\ \kappa_1=i}} B_\kappa^{k,F_0}(\lambda) \sum_{\substack{\sigma\in\mathcal{I}_2^s \\ \sigma_2=j}} B_\sigma^{s,F_0}(\lambda).$$

Using the product formula

$$B_\nu^{n,F} B_\mu^{m,F} = \frac{\binom{\nu+\mu}{\nu}}{\binom{n+m}{n}} B_{\nu+\mu}^{n+m,F}, \qquad \nu\in\mathcal{I}_2^n, \quad \mu\in\mathcal{I}_2^m, \quad F \text{ any triangle}, \tag{7.5}$$

we get

$$B_{ij}^{k,s} = \sum_{\substack{\kappa\in\mathcal{I}_2^k,\,\sigma\in\mathcal{I}_2^s \\ \kappa_1=i,\,\sigma_2=j}} \frac{\binom{\kappa+\sigma}{\kappa}}{\binom{k+s}{k}} B_{\kappa+\sigma}^{k+s,F_0},$$

compare [16]. This leads to

$$\sum_{i=0}^{k}\sum_{j=0}^{s} c_{ij} B_{ij}^{k,s} = \sum_{\kappa\in\mathcal{I}_2^k,\,\sigma\in\mathcal{I}_2^s} c_{\kappa_1,\sigma_2} \frac{\binom{\kappa+\sigma}{\kappa}}{\binom{k+s}{k}} B_{\kappa+\sigma}^{k+s,F_0},$$

which implies (7.4). □

The conversion process for computing $\tilde{c} = (\tilde{c}_\mu)_{\mu\in\mathcal{I}_2^{k+s}}$ from $c = (c_{ij})_{i,j=0}^{k,s}$ in this lemma can be written in terms of matrix multiplication as

$$\tilde{c} = T^{k,s} c,$$

where

$$T^{k,s}_{\mu,(ij)} = \begin{cases} M^{k,s}_{\kappa,\mu-\kappa}, & \text{if } \kappa \le \mu, \text{ where } \kappa = (i, \mu_2 - j, k - i - \mu_2 + j), \\ 0, & \text{otherwise,} \end{cases}$$

and $M^{k,s}$ is the trinomial matrix whose elements are

$$M^{k,s}_{\alpha,\beta} := \frac{\binom{\alpha+\beta}{\alpha}}{\binom{k+s}{k}} \qquad \alpha \in \mathcal{I}^k_2, \quad \beta \in \mathcal{I}^s_2, \tag{7.6}$$

see Algorithm 5 of [3]. Note that the condition $(i, \mu_2 - j, k - i - \mu_2 + j) \le \mu$ is equivalent to

$$i \le \mu_1 \le i + (s - j),$$
$$j \le \mu_2 \le j + (k - i).$$

By the proof of Theorem 3 of [3], the cost of Algorithm 5 of that paper for finding $M^{k,s}$ is

$$(k+1)(s+1) + 2\binom{k+2}{2}\binom{s+2}{2} = \mathcal{O}(k^2 s^2).$$

Hence, by computing and storing $M^{k,s}_{\kappa,\sigma}$ as the $(\kappa + \sigma, (\kappa_1, \sigma_2))$-th entry of $T^{k,s}$, for all $\kappa \in \mathcal{I}^k_2$, $\sigma \in \mathcal{I}^s_2$, we get an $\mathcal{O}(k^2 s^2)$ algorithm for computing $T^{k,s}$ without storing $M^{k,s}$. Since the number of nonzero entries in the column of $T^{k,s}$ corresponding to the pair (i, j) is $(s - j + 1)(k - i + 1)$, the total number of nonzero entries in $T^{k,s}$ is $\binom{k+2}{2}\binom{s+2}{2} = \mathcal{O}(k^2 s^2)$.

As an example, we give the matrix $T^{1,1}$ for converting from a bilinear tensor-product polynomial on the square $[0, 1]^2$ to a bivariate polynomial of degree two on the triangle F_0 with vertices $(1, 0), (0, 1), (0, 0)$:

$$T^{1,1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ .5 & 0 & 0 & .5 \\ .5 & 0 & .5 & 0 \\ 0 & 1 & 0 & 0 \\ .5 & .5 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix},$$

where the rows correspond to B^{2,F_0}_{200}, B^{2,F_0}_{110}, B^{2,F_0}_{101}, B^{2,F_0}_{020}, B^{2,F_0}_{011}, B^{2,F_0}_{002}, and the columns to $B^{1,1}_{00}$, $B^{1,1}_{01}$, $B^{1,1}_{10}$, $B^{1,1}_{11}$. This can be easily checked either directly or by the above formulae using

$$M^{1,1} = \begin{pmatrix} 1 & .5 & .5 \\ .5 & 1 & .5 \\ .5 & .5 & 1 \end{pmatrix}.$$

Lemma 7.4. *Let H be a hexahedron that shares a face F with an interface tetrahedron $\widetilde{T}$. Suppose we insert the B-coefficients of the tensor-product polynomial*

$s|_H \circ \phi_H \in \mathcal{Q}_d$ corresponding to the domain points in $\mathcal{D}_{d,H}$ lying in the face R of H containing F into a vector $c := (c_1, \ldots, c_n)$, where $n := (d+1)^2$. Similarly, let $\tilde{c} := (\tilde{c}_1, \ldots, \tilde{c}_{\tilde{n}})$ with $\tilde{n} := \binom{2d+2}{2}$ be the coefficients of the polynomial $s|_{\widetilde{T}} \circ \phi_{\widetilde{T}} \in \mathcal{P}_{2d}$ corresponding to domain points in $\mathcal{D}_{d,F} = \mathcal{D}_{d,\widetilde{T}} \cap F$. Then there exists a $\tilde{n} \times n$ matrix W_d such that

$$\tilde{c} = W_d\, c. \tag{7.7}$$

Proof: We can take W_d to be a permuted version of $T^{d,d}$, where the permutation depends on the ordering of the components of c and $\tilde{c}$. □

This lemma involves converting a tensor-product polynomial of degree (d, d) to a total degree polynomial of degree $2d$. Since the sparse matrix W_d in (7.7) has at most $\mathcal{O}(d^4)$ nonzero entries, this requires $\mathcal{O}(d^4)$ operations. The same operation count obviously applies to the multiplication of $(W_d)^t$ with a vector.

7.3. Bernstein-Bézier moments

As with any projection method, the finite element method requires the computation of moments of the data against the basis functions. Given a function f, its Bernstein-Bézier (BB) moments [3] are defined as

$$\mu_\nu^d(f, T) = \int_T B_\nu^{d,T}(x) f(x)\, dx, \qquad \nu \in \mathcal{I}_3^d,$$

for any tetrahedron T, and as

$$\mu_\nu^{(n,m,\ell)}(f, H_0) = \int_{H_0} B_\nu^{n,m,\ell}(x) f(x)\, dx, \qquad \nu \in \mathcal{J}_3^{(n,m,\ell)},$$

for the unit cube $H_0 = [0, 1]^3$.

Algorithms proposed in [3] (see Corollary 2 there) allow the computation of the arrays $\{\mu_\nu^d(f, T)\}_{\nu \in \mathcal{I}_3^d}$ with $\mathcal{O}(d^4)$ floating point operations, that is with $\mathcal{O}(d)$ cost per entry. According to Theorem 6 of [3], the cost of computing the array $\{\mu_\nu^{(n,m,\ell)}(f, H_0)\}_{\nu \in \mathcal{J}_3^{(n,m,\ell)}}$ is $\mathcal{O}(n^2 m\ell + nm^2\ell + nm\ell^2)$.

§8. Use of $\mathcal{S}_d^0(\triangle)$ in the FEM

The space $\mathcal{S}_d^0(\triangle)$ can be used for finite element analysis in the standard way, which entails that the mass, stiffness matrices etc. of the basis functions ψ_ξ of Sect. 6 have to be computed, along with the corresponding load vector. The local support property of the Bernstein polynomial shape functions means that the finite element sub-assembly procedure can be applied, see e.g. [26,27]. The foregoing properties might be regarded as a basic minimum for a basis. However, a crucial advantage of the Bernstein shape functions, in comparison to the shape functions of Lagrange or orthogonal polynomial type (see e.g. [18]), is the availability of optimal algorithms for the assembly of the element level system matrices [3]. The

resulting approximating spline s is represented in the computer by simply storing its B-coefficients on all elements of $\triangle_R$. This means that the evaluation of s, or its partial derivatives, can be efficiently performed using the well-known de Casteljau algorithm, see e.g. [21]. In particular, we can efficiently compute the gradient of s by using sparse formulae for the gradients of Bernstein polynomials, see (8.9) and (8.10) below.

8.1. Global-Local transformation

An important tool for the finite element implementation is that if c is the global vector of all B-coefficients of a spline $s \in \mathcal{S}_d^0(\triangle)$ corresponding to the domain points in $\mathcal{M}$, and $\hat{c}$ is the vector of coefficients of the shape functions associated with the individual elements of $\triangle_R$, then there exists an $\hat{n} \times n_d$ matrix A such that

$$\hat{c} = A\, c,$$

where $n_d = \dim \mathcal{S}_d^0(\triangle)$ and $\hat{n}$ is the sum of the number of local degrees of freedom in each element ignoring the continuity constraints between neighbouring elements:

$$\hat{n} := \binom{d+3}{3} n_T + 2\binom{2d+3}{3} n_P + (d+1)^3 n_H.$$

The matrix A is called the global-local transformation matrix. The columns of A consist of the coefficients of the restrictions of the basis functions $\{\psi_\xi\}_{\xi \in \mathcal{M}}$ to the elements of $\triangle_R$, expanded into linear combinations of the shape functions.

In practice, the full matrix A is never constructed explicitly thanks to the block structure

$$A = \begin{pmatrix} A^{(1)} \\ \vdots \\ A^{(N)} \end{pmatrix}.$$

where N is the number of elements and each block $A^{(k)}$ corresponds to the degrees of freedom in an individual element Ω_k. For instance, each block corresponding to a hexahedron H will have $\#\mathcal{D}_{d,H} = (d+1)^3$ rows; blocks corresponding to an ordinary tetrahedron will have $\#\mathcal{D}_{d,T} = \binom{d+3}{3}$ rows; and blocks corresponding to an interface tetrahedron $\widetilde{T}$ will have $\#\mathcal{D}_{2d,\widetilde{T}} = \binom{2d+3}{3}$ rows.

For a partition comprising of purely tetrahedra or hexahedra, the blocks $A^{(k)}$ are Boolean matrices (composed of 0's and 1's). However, the presence of pyramidal elements, and the attendant splitting into interface tetrahedra, means that the corresponding blocks are no longer Boolean. Nevertheless, it is not difficult to assemble the matrix A (and hence the individual blocks) using the following algorithm.

Algorithm 8.1. *Assembly of the transformation matrix A.*

1) Initialize A as an $\hat{n} \times n_d$ zero matrix.

2) *For each column corresponding to a point $\xi \in \mathcal{M}$, insert 1 in that column in all rows corresponding to ξ.*

3) *For each face F shared by an ordinary tetrahedron $T \in \mathcal{T}_O$ and an interface tetrahedron $\widetilde{T} \in \mathcal{T}_I$, write the entries of the matrix U_d appearing in (7.3) into the rows of A corresponding to $\mathcal{D}_{2d,\widetilde{T}} \cap F$ and columns corresponding to $\mathcal{M} \cap F = \mathcal{D}_{d,T} \cap F$.*

4) *For each face F shared by a hexahedron $H \in \mathcal{H}$ and an interface tetrahedron $\widetilde{T} \in \mathcal{T}_I$, write the entries of the matrix W_d appearing in (7.7) into the rows of A corresponding to $\mathcal{D}_{2d,\widetilde{T}} \cap F$ and columns corresponding to $\mathcal{M} \cap R = \mathcal{D}_{d,H} \cap R$, where R is the face of H containing F.*

We reiterate that the *explicit* evaluation and storage of A is generally avoided, see Algorithms 8.2 and 8.3 below. However, for expository purposes, we shall discuss the full matrix A with understanding that, in the actual code, the matrix is treated blockwise.

A basic operation used in the finite element analysis using our spline space is the the computation of matrix vector products of the A and its transpose: Ac and $A^t\hat{c}$. These can be accomplished directly without assembling the full matrix A as follows. We split the vector $\hat{c}$ into blocks $\hat{c}^{(k)} = \{\hat{c}_\xi^{(k)}\}_{\xi \in \mathcal{D}_{\Omega_k}}$, $k = 1, \ldots, N$, corresponding to the elements Ω_k of $\triangle_R$, with a local indexing within these blocks by the domain points with

$$\mathcal{D}_{\Omega_k} := \begin{cases} \mathcal{D}_{d,\Omega_k} & \text{if } \Omega_k \text{ is either a hexahedron or an ordinary tetrahedron,} \\ \mathcal{D}_{2d,\Omega_k} & \text{if } \Omega_k \text{ is an interface tetrahedron.} \end{cases}$$

For any $K \subseteq \Omega$, let $\hat{c}_K^{(k)} := \{\hat{c}_\xi^{(k)}\}_{\xi \in \mathcal{D}_{\Omega_k} \cap K}$ and $c_K := \{c_\xi\}_{\xi \in \mathcal{M} \cap K}$.

Algorithm 8.2. *(Computation of $\hat{c} = Ac$.) Input: The vector $c = \{c_\xi\}_{\xi \in \mathcal{M}}$. Output: The vector $\hat{c}$ of length $\hat{n}$ split into blocks $\hat{c}^{(k)} = \{\hat{c}_\xi^{(k)}\}_{\xi \in \mathcal{D}_{\Omega_k}}$, $k = 1, \ldots, N$.*

For all $k = 1, \ldots, N$:

1) *If Ω_k is a hexahedron or an ordinary tetrahedron, set $\hat{c}^{(k)} = c_{\Omega_k}$.*

2) *If Ω_k is an interface tetrahedron:*
   a) *set $\hat{c}_K^{(k)} = c_K$, where K is the interior of Ω_k;*
   b) *set $\hat{c}_K^{(k)} = c_K$, where K is the interior of the face of Ω_k shared with another interface tetrahedron;*
   c) *for the face F of Ω_k shared with a hexahedron H, set $\hat{c}_F^{(k)} = W_d\, c_R$, where R is the face of H containing F, and W_d is the matrix in (7.7);*
   d) *for each face F of Ω_k shared with an ordinary tetrahedron, set $\hat{c}_F^{(k)} = U_d\, c_F$, where U_d is the matrix in (7.3).*

The corresponding algorithm for the transpose reads as follows:

Algorithm 8.3. *(Computation of $c = A^t\hat{c}$.) Input: The vector $\hat{c}$ of length $\hat{n}$ split into blocks $\hat{c}^{(k)} = \{\hat{c}_\xi^{(k)}\}_{\xi\in\mathcal{D}_{\Omega_k}}$, $k = 1,\dots,N$. Output: The vector $c = \{c_\xi\}_{\xi\in\mathcal{M}}$. Initialize c as a zero vector of length n_d.*

For all $k = 1,\dots,N$ such that $\hat{c}^{(k)} \neq 0$:

1) If Ω_k is a hexahedron or an ordinary tetrahedron, set $c_{\Omega_k} = c_{\Omega_k} + \hat{c}^{(k)}$.

2) If Ω_k is an interface tetrahedron:

a) set $c_K = c_K + \hat{c}_K^{(k)}$, where K is the interior of Ω_k;

b) set $c_K = c_K + \hat{c}_K^{(k)}$, where K is the interior of the face of Ω_k shared with another interface tetrahedron;

c) for the face F of Ω_k shared with a hexahedron H, set $c_R = c_R + W_d^t\,\hat{c}_F^{(k)}$, where R is the face of H containing F and W_d is the matrix in (7.7);

d) Let F_1 and F_2 be the two faces of Ω_k shared with ordinary tetrahedra. Let e_1 and e_2 be the edges they share with H, and let e be the common edge of F_1, F_2. Then set

d1) $c_{F_1\setminus e_1} = c_{F_1\setminus e_1} + \tilde{c}_1$, where $\tilde{c}_1$ is the vector obtained from $U_d^t\hat{c}_{F_1}^{(k)}$ by removing the components corresponding to points in e_1, where U_d is the matrix in (7.3);

d2) $c_{F_2\setminus(e\cup e_2)} = c_{F_2\setminus(e\cup e_2)} + \tilde{c}_2$, where $\tilde{c}_2$ is the vector obtained from $U_d^t\hat{c}_{F_2}^{(k)}$ by removing the components corresponding to points in $e\cup e_2$;

It is easy to see that Algorithm 8.2 involves at most $\mathcal{O}(Nd^3)$ copying operations and $\mathcal{O}(n_P d^4)$ floating point operations. The computational cost of Algorithm 8.3 is $\mathcal{O}(d^3(m+m_P d))$ operations (of which only $\mathcal{O}(m_P d^4)$ are multiplications), where $m \leq N$ is the number of hexahedra and ordinary tetrahedra for which $\hat{c}^{(k)} \neq 0$, and $m_P \leq n_P$ is the number of pyramids with $\hat{c}^{(k)} \neq 0$.

8.2. Computing the load vector

The load vector L generated by the basis functions $\{\psi_\xi\}_{\xi\in\mathcal{M}}$, has the components

$$L_\xi = \int_\Omega \psi_\xi(x) f(x)\,dx, \quad \xi \in \mathcal{M}.$$

To compute it, we observe that

$$\psi_\xi|_{\Omega_k} = \sum_{\zeta\in\mathcal{D}_{\Omega_k}} A_{\zeta\xi}^{(k)}\psi_\zeta^{\Omega_k}, \qquad k = 1,\dots,N, \tag{8.1}$$

where

$$\psi_\zeta^{\Omega_k} := \begin{cases} \psi_\zeta^{d,T} & \text{if } \Omega_k \text{ is an ordinary tetrahedron } T \in \mathcal{T}_O, \\ \psi_\zeta^{2d,T} & \text{if } \Omega_k \text{ is an interface tetrahedron } T \in \mathcal{T}_I, \\ \psi_\zeta^{d,H} & \text{if } \Omega_k \text{ is a hexahedron } H \in \mathcal{H}. \end{cases}$$

This implies

$$L = A^t \hat{L}, \tag{8.2}$$

where the vector $\hat{L}$ consists of the blocks $\hat{L}^{(k)}$, $k = 1, \ldots, N$, (element level load vectors) with entries

$$\hat{L}^{(k)}_\zeta := \int_{\Omega_k} \psi^{\Omega_k}_\zeta(x) f(x)\, dx, \qquad \zeta \in \mathcal{D}_{\Omega_k}.$$

Recall that the domain points $\zeta \in \mathcal{D}_{\Omega_k}$ are indexed by the elements ν of either $\mathcal{I}^d_3$ or $\mathcal{J}^{\mathbf{d}}_3$, where $\mathbf{d} := (d, d, d)$. Therefore we also write $\hat{L}^{(k)}_\nu := \hat{L}^{(k)}_{\zeta_\nu}$ for the sake of brevity. Using the mappings ϕ_T or ϕ_H (if needed), we obtain the following representations for the entries of $\hat{L}^{(k)}$, where in general for any mapping ϕ, we write J_ϕ for its corresponding Jacobian matrix.

Lemma 8.4. *(Element level load vectors.)*

1) If Ω_k is an ordinary tetrahedron $T \in \mathcal{T}_O$, then

$$\hat{L}^{(k)}_\nu = \mu^d_\nu(f, T), \qquad \nu \in \mathcal{I}^d_3.$$

2) If Ω_k is an interface tetrahedron $T \in \mathcal{T}_I$, then

$$\hat{L}^{(k)}_\nu = \mu^{2d}_\nu(f(\phi_T) \det J_{\phi_T}, T_0), \qquad \nu \in \mathcal{I}^{2d}_3.$$

3) If Ω_k is a hexahedron $H \in \mathcal{H}$, then

$$\hat{L}^{(k)}_\nu = \mu^{\mathbf{d}}_\nu(f(\phi_H) \det J_{\phi_H}, H_0), \qquad \nu \in \mathcal{J}^{\mathbf{d}}_3.$$

Proof: Consider the case where Ω_k is an interface tetrahedron $T \in \mathcal{T}_I$. Then

$$\begin{aligned}\int_T \psi^{2d,T}_\nu(x) f(x)\, dx &= \int_{T_0} B^{2d,T}_\nu(\lambda) f(\phi_T(\lambda)) \det J_{\phi_T}(\lambda)\, d\lambda \\ &= \mu^{2d}_\nu(f(\phi_T) \det J_{\phi_T}, T_0).\end{aligned}$$

The other cases are similar. □

Using this lemma, we can now formulate an algorithm for assembling the load vector.

Algorithm 8.5. *Assembly of the load vector L.*

1) Use Lemma 8.4 to compute element level load vectors $\hat{L}^{(k)}$, $k = 1, \ldots, N$, where the BB-moments are obtained using Algorithm 3 of [3] *for both ordinary and interface tetrahedra, and Theorem 6 of* [3] *for hexahedra.*

2) Compute L according to (8.2), using Algorithm 8.3.

Clearly, the computational cost of the first step of this algorithm is $\mathcal{O}(Nd^4)$, and that of the second step is $\mathcal{O}(d^3(N+n_P d))$, which gives the total cost as $\mathcal{O}(Nd^4)$, that is $\mathcal{O}(d)$ per component of L.

8.3. Computing the mass matrix

The entries of the mass matrix M are given by

$$M_{\xi\zeta} = \int_\Omega \psi_\xi(x)\psi_\zeta(x)f(x)\,dx, \quad \xi,\zeta \in \mathcal{M}.$$

In view of (8.1),

$$M = A^t \hat{M} A, \tag{8.3}$$

where $\hat{M}$ is the block-diagonal $\hat{n} \times \hat{n}$ matrix

$$\hat{M} = \mathrm{diag}(\hat{M}^{(1)}, \ldots, \hat{M}^{(N)}),$$

whose blocks $\hat{M}^{(k)}$ are the element level mass matrices with entries

$$\hat{M}^{(k)}_{\xi\zeta} := \int_{\Omega_k} \psi^{\Omega_k}_\xi(x)\psi^{\Omega_k}_\zeta(x)f(x)\,dx, \qquad \xi,\zeta \in \mathcal{D}_{\Omega_k}.$$

Lemma 8.6. *(Element level mass matrices.)*
1) If Ω_k is an ordinary tetrahedron $T \in \mathcal{T}_O$, then

$$\hat{M}^{(k)}_{\nu\kappa} = \frac{\binom{\nu+\kappa}{\nu}}{\binom{2d}{d}}\mu^{2d}_{\nu+\kappa}(f,T), \qquad \nu,\kappa \in \mathcal{I}^d_3.$$

2) If Ω_k is an interface tetrahedron $T \in \mathcal{T}_I$, then

$$\hat{M}^{(k)}_{\nu\kappa} = \frac{\binom{\nu+\kappa}{\nu}}{\binom{4d}{2d}}\mu^{4d}_{\nu+\kappa}(f(\phi_T)\det J_{\phi_T}, T_0), \qquad \nu,\kappa \in \mathcal{I}^{2d}_3.$$

3) If Ω_k is a hexahedron $H \in \mathcal{H}$, then

$$\hat{M}^{(k)}_{\nu\kappa} = \frac{\binom{\nu+\kappa}{\nu}\binom{2\boldsymbol{d}-\nu-\kappa}{\boldsymbol{d}-\nu}}{\binom{2d}{d}^3}\mu^{2\boldsymbol{d}}_{\nu+\kappa}(f(\phi_H)\det J_{\phi_H}, H_0), \qquad \nu,\kappa \in \mathcal{J}^{\boldsymbol{d}}_3.$$

where we recall that $\boldsymbol{d} = (d,d,d)$.

Proof: If $\Omega_k = T \in \mathcal{T}_O$ or $\mathcal{T}_I$, then

$$\hat{M}^{(k)}_{\nu\kappa} = \int_T B^d_\nu(x)B^d_\kappa(x)f(x)\,dx, \quad \nu,\kappa \in \mathcal{I}^d_3,$$

or, respectively,

$$\hat{M}^{(k)}_{\nu\kappa} = \int_{T_0} B^{2d}_\nu(\lambda)B^{2d}_\kappa(\lambda)f(\phi_T(\lambda))\det J_{\phi_T}(\lambda)\,d\lambda, \quad \nu,\kappa \in \mathcal{I}^{2d}_3,$$

and the statements follow by the product formula

$$B_\nu^n B_\kappa^m = \frac{\binom{\nu+\kappa}{\nu}}{\binom{n+m}{n}} B_{\nu+\kappa}^{n+m}, \qquad \nu \in \mathcal{I}_3^n, \quad \kappa \in \mathcal{I}_3^m. \tag{8.4}$$

If $\Omega_k = H \in \mathcal{H}$, then

$$\hat{M}_{\nu\kappa}^{(k)} = \int_{H_0} B_\nu^{\mathbf{d}}(\lambda) B_\kappa^{\mathbf{d}}(\lambda) f(\phi_H(\lambda)) \det J_{\phi_H}(\lambda)\, d\lambda, \quad \nu, \kappa \in \mathcal{J}_3^{\mathbf{d}},$$

and we use

$$B_\nu^{\boldsymbol{n}} B_\kappa^{\boldsymbol{m}} = \frac{\binom{\nu+\kappa}{\nu}\binom{\boldsymbol{n+m}-\nu-\kappa}{\boldsymbol{n}-\nu}}{\binom{\boldsymbol{n+m}}{\boldsymbol{n}}} B_{\nu+\kappa}^{\boldsymbol{n+m}}, \qquad \nu \in \mathcal{J}_3^{\boldsymbol{n}}, \kappa \in \mathcal{J}_3^{\boldsymbol{m}}. \quad \square \tag{8.5}$$

For the purpose of efficient implementation we make the following observation.

Lemma 8.7. *The formula in part 3 of Lemma 8.6 can be written in the form*

$$\hat{M}_{\nu\kappa}^{(k)} = \sigma_{\nu_1,\kappa_1}\sigma_{\nu_2,\kappa_2}\sigma_{\nu_3,\kappa_3}\mu_{\nu+\kappa}^{2\boldsymbol{d}}(f(\phi_H)\det J_{\phi_H}, H_0), \qquad \nu, \kappa \in \mathcal{J}_3^{\boldsymbol{d}}, \tag{8.6}$$

where $\nu = (\nu_1, \nu_2, \nu_3)$, $\kappa = (\kappa_1, \kappa_2, \kappa_3)$ *and*

$$\sigma_{nm} := \frac{\binom{n+m}{n}\binom{2d-n-m}{d-n}}{\binom{2d}{d}}, \qquad 0 \le n, m \le d. \tag{8.7}$$

The matrix σ can be easily precomputed with $\mathcal{O}(d^2)$ cost. Its d^2 storage requirement is negligible in comparison to the $\mathcal{O}(d^6)$ storage of $\hat{M}^{(k)}$ or even the $\mathcal{O}(d^3)$ storage requirement of the moments $\mu_\nu^{2\boldsymbol{d}}(f(\phi_H)\det J_{\phi_H}, H_0)$. Thus, assuming that the moment vector is known, $\hat{M}^{(k)}$ can be computed according to (8.6) with just three multiplications per entry.

Algorithm 8.8. *Assembly of the mass matrix* M.

1) *For each* $k = 1, \ldots, N$:
   a) *If* Ω_k *is an ordinary tetrahedron* $T \in \mathcal{T}_O$, *then use Algorithm 3 of* [3] *to compute the BB moments* $\mu_\nu^{2d}(f, T)$, $\nu \in \mathcal{I}_3^{2d}$, *and assemble the element level mass matrix* $\hat{M}^{(k)}$ *of Lemma 8.6 using Algorithm 6 of [3].*
   b) *If* Ω_k *is an interface tetrahedron* $T \in \mathcal{T}_I$, *then use Algorithm 3 of* [3] *to compute the BB moments* $\mu_\nu^{4d}(f(\phi_T)\det J_{\phi_T}, T_0)$, $\nu \in \mathcal{I}_3^{4d}$, *and assemble the element level mass matrix* $\hat{M}^{(k)}$ *of Lemma 8.6 using Algorithm 6 of [3].*
   c) *If* Ω_k *is a hexahedron* $H \in \mathcal{H}$, *then use Theorem 6 of* [3] *to compute the BB moments* $\mu_\nu^{2\boldsymbol{d}}(f(\phi_H)\det J_{\phi_H}, H_0)$, $\nu \in \mathcal{J}_3^{\boldsymbol{2d}}$, *and assemble the element level mass matrix* $\hat{M}^{(k)}$ *of Lemma 8.6 using (8.6).*

2) *Compute* $\tilde{M} := A^t \hat{M}$ *by applying Algorithm 8.3 to the columns of* $\hat{M}$*. Note that* $\hat{c}^{(k)}$ *in the notation of that algorithm can only be nonzero if the current column of* $\hat{M}$ *contains some entry of* $\hat{M}^{(k)}$*.*

3) *Compute* $M = (A^t \tilde{M}^t)^t$*, of (8.3) by applying Algorithm 8.3 to the rows of* $\tilde{M}$*. The vector* $\hat{c}^{(k)}$ *can be nonzero only if the current row of* $\tilde{M}$ *corresponds to a domain point* ξ *in* Ω_k *if* Ω_k *is either a hexahedron or an ordinary tetrahedron, or in* $\Omega_k \cup R$ *if* Ω_k *is an interface tetrahedron and* R *is the face of the hexahedron containing a face of* Ω_k*.*

The computational cost of the first step is $\mathcal{O}(Nd^4)$ for the computation of the moment vectors, and $\mathcal{O}(Nd^6)$ for the element level mass matrices. The second and third steps involve $\mathcal{O}(d^7 n_P)$ multiplications and $\mathcal{O}(d^6(N + n_P d))$ additions. If $n_P d = \mathcal{O}(N)$, then the overall cost is optimal because the sparse matrix M has $\mathcal{O}(Nd^6)$ entries. The storage requirement is $\mathcal{O}(Nd^6)$ for all element level mass matrices, and $\mathcal{O}(Nd^6)$ for the sparse matrices $\tilde{M}$ and M. The moments need to be stored temporarily in Step 1, but this requires only $\mathcal{O}(d^3)$ storage space.

8.4. Computing the stiffness matrix

The entries of the stiffness matrix S are given by

$$S_{\xi\zeta} = \int_\Omega \nabla^t \psi_\xi(x) F(x) \nabla \psi_\zeta(x)\, dx, \quad \xi, \zeta \in \mathcal{M},$$

where $F : \Omega \to \mathbb{R}^{3\times 3}$ is a given matrix function and ∇ (∇^t) is the column (row) gradient operator. Hence

$$S = A^t \hat{S} A, \tag{8.8}$$

where $\hat{S}$ is the block-diagonal $\hat{n} \times \hat{n}$ matrix $\hat{S} = \mathrm{diag}(\hat{S}^{(1)}, \ldots, \hat{S}^{(N)})$, whose blocks $\hat{S}^{(k)}$ are the element level stiffness matrices with entries

$$\hat{S}^{(k)}_{\xi\zeta} = \int_{\Omega_k} \nabla^t \psi^{\Omega_k}_\xi(x) F(x) \nabla \psi^{\Omega_k}_\zeta(x)\, dx, \quad \xi, \zeta \in \mathcal{D}_{\Omega_k}.$$

Lemma 8.9. *(Element level stiffness matrices.)*

1) *If* Ω_k *is an ordinary tetrahedron* $T \in \mathcal{T}_O$*, then*

$$\hat{S}^{(k)}_{\nu\kappa} = d^2 \sum_{i,j=1}^{4} \frac{\binom{\nu - e_i + \kappa - e_j}{\nu}}{\binom{2d-2}{d-1}} \nabla^t \lambda_i^T \mu^{2d-2}_{\nu - e_i + \kappa - e_j}(F, T) \nabla \lambda_j^T, \quad \nu, \kappa \in \mathcal{I}_3^d,$$

where e_i *is the* i*-th unit vector in* $\mathbb{R}^4$*, and* λ_i^T *is the* i*-th barycentric coordinate of* x *w.r.t.* T*. Here we adopt the usual convention whereby the terms with* μ^n_ξ *for which* $\xi \notin \mathcal{I}_3^n$ *are ignored in the sum.*

2) *If* Ω_k *is an interface tetrahedron* $T \in \mathcal{T}_I$*, then*

$$\hat{S}^{(k)}_{\nu\kappa} = 4d^2 \sum_{i,j=1}^{4} \frac{\binom{\nu - e_i + \kappa - e_j}{\nu}}{\binom{4d-2}{d-1}} \nabla^t \lambda_i^{T_0} \mu^{4d-2}_{\nu - e_i + \kappa - e_j}(\tilde{F}^T, T_0) \nabla \lambda_j^{T_0}, \quad \nu, \kappa \in \mathcal{I}_3^{2d},$$

with

$$\tilde{F}^T := J_{\phi_T}^{-1} F(\phi_T) J_{\phi_T}^{-t} \det J_{\phi_T},$$

where J^{-t} *denotes the transpose of the inverse matrix of* J, *and* $\nabla^t \lambda_1^{T_0} = (1,0,0)$, $\nabla^t \lambda_2^{T_0} = (0,1,0)$, $\nabla^t \lambda_3^{T_0} = (0,0,1)$, $\nabla^t \lambda_4^{T_0} = (-1,-1,-1)$.

3) If Ω_k *is a hexahedron* $H \in \mathcal{H}$, *then for* $\nu, \kappa \in \mathcal{J}_3^{\boldsymbol{d}}$,

$$\begin{aligned}\hat{S}_{\nu\kappa}^{(k)} = d^2 \sum_{i,j=1}^{3} \Big[& \frac{\binom{\nu-e_i+\kappa-e_j}{\nu-e_i}\binom{2\boldsymbol{d}-\nu-\kappa}{\boldsymbol{d}-\nu}}{\binom{2\boldsymbol{d}-e_i-e_j}{\boldsymbol{d}-e_i}} \mu_{\nu-e_i+\kappa-e_j}^{2\boldsymbol{d}-e_i-e_j}(\tilde{F}_{ij}^H, H_0) \\ & - \frac{\binom{\nu-e_i+\kappa}{\nu-e_i}\binom{2\boldsymbol{d}-\nu-\kappa-e_j}{\boldsymbol{d}-\nu}}{\binom{2\boldsymbol{d}-e_i-e_j}{\boldsymbol{d}-e_i}} \mu_{\nu-e_i+\kappa}^{2\boldsymbol{d}-e_i-e_j}(\tilde{F}_{ij}^H, H_0) \\ & - \frac{\binom{\nu+\kappa-e_j}{\nu}\binom{2\boldsymbol{d}-\nu-e_i-\kappa}{\boldsymbol{d}-\nu-e_i}}{\binom{2\boldsymbol{d}-e_i-e_j}{\boldsymbol{d}-e_i}} \mu_{\nu+\kappa-e_j}^{2\boldsymbol{d}-e_i-e_j}(\tilde{F}_{ij}^H, H_0) \\ & + \frac{\binom{\nu+\kappa}{\nu}\binom{2\boldsymbol{d}-\nu-e_i-\kappa-e_j}{\boldsymbol{d}-\nu-e_i}}{\binom{2\boldsymbol{d}-e_i-e_j}{\boldsymbol{d}-e_i}} \mu_{\nu+\kappa}^{2\boldsymbol{d}-e_i-e_j}(\tilde{F}_{ij}^H, H_0) \Big],\end{aligned}$$

where

$$\tilde{F}^H := J_{\phi_H}^{-1} F(\phi_H) J_{\phi_H}^{-t} \det J_{\phi_H},$$

and e_i *is the* i*-th unit vector in* $\mathbb{R}^3$.

Proof: If $\Omega_k = T \in \mathcal{T}_O$, then

$$\hat{S}_{\nu\kappa}^{(k)} = \int_T \nabla^t B_\nu^d(x) F(x) \nabla B_\kappa^d(x)\, dx, \quad \nu, \kappa \in \mathcal{I}_3^d,$$

and if $T \in \mathcal{T}_I$, then

$$\hat{S}_{\nu\kappa}^{(k)} = \int_{T_0} \nabla^t B_\nu^{2d}(\lambda) \tilde{F}^T(\lambda) \nabla B_\kappa^{2d}(\lambda)\, d\lambda, \quad \nu, \kappa \in \mathcal{I}_3^{2d}.$$

The statements of the theorem follow by (8.4) in view of the gradient formula

$$\nabla B_\nu^{d,T}(x) = d \sum_{i=1}^{4} B_{\nu-e_i}^{d-1,T}(x) \nabla \lambda_i^T, \tag{8.9}$$

compare equation (44) of [3].

If $\Omega_k = H \in \mathcal{H}$, then

$$\hat{S}_{\nu\kappa}^{(k)} = \int_{H_0} \nabla^t B_\nu^{\mathbf{d}}(\lambda) \tilde{F}^H(\lambda) \nabla B_\kappa^{\mathbf{d}}(\lambda)\, d\lambda, \quad \nu, \kappa \in \mathcal{J}_3^d,$$

and we use (8.5) and

$$\nabla B_\nu^{\mathbf{d}}(\lambda) = d \sum_{i=1}^{3} \big(B_{\nu-e_i}^{\mathbf{d}-e_i}(\lambda) - B_\nu^{\mathbf{d}-e_i}(\lambda) \big) e_i. \quad \square \tag{8.10}$$

For the purposes of an efficient implementation, we make the following observation.

Lemma 8.10. *The formula in part 3 of Lemma 8.9 can be rewritten as*

$$\hat{S}^{(k)}_{\nu\kappa} = 2d\sigma_{\nu_1,\kappa_1}\sigma_{\nu_2,\kappa_2}\sigma_{\nu_3,\kappa_3}\sum_{i,j=1}^{3}\Big[\alpha_{ij}\mu^{2\boldsymbol{d}-e_i-e_j}_{\nu-e_i+\kappa-e_j}(\tilde{F}^H_{ij},H_0) - \beta_{ij}\mu^{2\boldsymbol{d}-e_i-e_j}_{\nu-e_i+\kappa}(\tilde{F}^H_{ij},H_0) - \gamma_{ij}\mu^{2\boldsymbol{d}-e_i-e_j}_{\nu+\kappa-e_j}(\tilde{F}^H_{ij},H_0) + \delta_{ij}\mu^{2\boldsymbol{d}-e_i-e_j}_{\nu+\kappa}(\tilde{F}^H_{ij},H_0)\Big],$$

where $\nu=(\nu_1,\nu_2,\nu_3)$, $\kappa=(\kappa_1,\kappa_2,\kappa_3)$, σ *is given by (8.7),*

$$\alpha_{ij} = \tfrac{2d\nu_{\mathsf{i}}\kappa_{\mathsf{j}}}{(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}})(\nu_{\mathsf{j}}+\kappa_{\mathsf{j}})},\quad \beta_{ij} = \tfrac{2d\nu_{\mathsf{i}}(d-\kappa_{\mathsf{j}})}{(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}})(2d-\nu_{\mathsf{j}}-\kappa_{\mathsf{j}})},$$
$$\gamma_{ij} = \tfrac{2d(d-\nu_{\mathsf{i}})\kappa_{\mathsf{j}}}{(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}})(\nu_{\mathsf{j}}+\kappa_{\mathsf{j}})},\quad \delta_{ij} = \tfrac{2d(d-\nu_{\mathsf{i}})(d-\kappa_{\mathsf{j}})}{(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}})(2d-\nu_{\mathsf{j}}-\kappa_{\mathsf{j}})},$$

if $i\neq j$, *and*

$$\alpha_{ii} = \tfrac{(2d-1)\nu_{\mathsf{i}}\kappa_{\mathsf{i}}}{(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}})(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}}-1)},\quad \beta_{ii} = \tfrac{(2d-1)\nu_{\mathsf{i}}(d-\kappa_{\mathsf{i}})}{(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}})(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}})},$$
$$\gamma_{ii} = \tfrac{(2d-1)(d-\nu_{\mathsf{i}})\kappa_{\mathsf{i}}}{(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}})(\nu_{\mathsf{i}}+\kappa_{\mathsf{i}})},\quad \delta_{ii} = \tfrac{(2d-1)(d-\nu_{\mathsf{i}})(d-\kappa_{\mathsf{i}})}{(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}})(2d-\nu_{\mathsf{i}}-\kappa_{\mathsf{i}}-1)}.$$

Assuming that the moment vectors are known, $\hat{S}^{(k)}$ can be computed in the above form with $\mathcal{O}(1)$ floating point operations per entry.

Algorithm 8.11. *Assembly of the stiffness matrix* S.

1) *For each* $k=1,\dots,N$:
   a) *If* Ω_k *is an ordinary tetrahedron* $T\in\mathcal{T}_O$, *then use Algorithm 3 of* [3] *to compute the BB moments* $\mu^{2d-2}_\nu(F,T)$, $\nu\in\mathcal{I}^{2d-2}_3$, *and assemble the element level stiffness matrix* $\hat{S}^{(k)}$ *of Lemma 8.9 using Algorithm 6 of* [3].
   b) *If* Ω_k *is an interface tetrahedron* $T\in\mathcal{T}_I$, *then use Algorithm 3 of* [3] *to compute the BB moments* $\mu^{4d-2}_\nu(\tilde{F}^T,T_0)$, $\nu\in\mathcal{I}^{4d-2}_3$, *and assemble the element level stiffness matrix* $\hat{S}^{(k)}$ *of Lemma 8.9 using Algorithm 6 of* [3].
   c) *If* Ω_k *is a hexahedron* $H\in\mathcal{H}$, *then for all* $i,j\in\{1,2,3\}$ *use Theorem 6 of* [3] *to compute the BB moments*

$$\mu^{2\boldsymbol{d}-e_i-e_j}_\nu(\tilde{F}^H_{ij},H_0),\qquad \nu\in\mathcal{J}_3^{2\boldsymbol{d}-e_i-e_j},$$

   and assemble the element level stiffness matrix $\hat{S}^{(k)}$ *of Lemma 8.9 using Lemma 8.10.*
2) *Compute* $\tilde{S}:=A^t\hat{S}$ *by applying Algorithm 8.3 to the columns of* $\hat{S}$. *The vector* $\hat{c}^{(k)}$ *in the notation of that algorithm can only be nonzero if the current column of* $\hat{S}$ *contains any entries of* $\hat{S}^{(k)}$.
3) *Compute* $S=(A^t\tilde{S}^t)^t$, *according to (8.8), by applying Algorithm 8.3 to the rows of* $\tilde{S}$. *The vector* $\hat{c}^{(k)}$ *can be nonzero only if the current row of* $\tilde{S}$ *corresponds to a domain point* ξ *in* Ω_k *when* Ω_k *is either a hexahedron or an*

ordinary tetrahedron, or in $\Omega_k \cup R$ when it is an interface tetrahedron and R is the face of the hexahedron containing a face of Ω_k.

The computational and storage cost of the assembling the stiffness matrix is $\mathcal{O}(d^6(N + n_P d))$, that is $\mathcal{O}(1)$ per nonzero entry if $n_P d = \mathcal{O}(N)$. This can be seen in the same way as for the mass matrix in Section 8.4. For a discussion of assembling stiffness matrices one element at a time, see [26,27].

8.5. Matrix free finite element implementation via BB moments

Iterative methods, such as the conjugate gradient method do not require computation and storage of the system matrices if the matrix-vector multiplication can be implemented without doing so. As pointed out in Sect. 4.4 of [3], this also amounts to the computation of BB moments. Indeed, let the current iterate u be expressed as a linear combination of the basis functions ψ_ξ of $\mathcal{S}_d^0(\triangle)$,

$$u = \sum_{\xi \in \mathcal{M}} c_\xi \psi_\xi.$$

Then, for example, the product Sc of the stiffness matrix S with the coefficient vector c is given by

$$Sc = A^t \hat{S} A c.$$

Splitting the vector $\hat{c} = Ac$ of length $\hat{n}$ into blocks $\hat{c}^{(k)} = \{\hat{c}_\xi^{(k)}\}_{\xi \in \mathcal{D}_{\Omega_k}}$, $k = 1, \ldots, N$, we obtain

$$\tilde{c} := \hat{S}\hat{c},$$

where $\tilde{c}$ consists of the blocks $\tilde{c}^{(k)} = \hat{S}^{(k)}\hat{c}^{(k)}$, $k = 1, \ldots, N$. With

$$u^{(k)} := \sum_{\xi \in \mathcal{D}_{\Omega_k}} \hat{c}_\xi^{(k)} \psi_\xi^{\Omega_k}, \qquad k = 1, \ldots, N,$$

we get for all $\xi \in \mathcal{D}_{\Omega_k}$

$$(\hat{S}^{(k)}\hat{c}^{(k)})_\xi = \int_{\Omega_k} \nabla^t \psi_\xi^{\Omega_k}(x) F(x) \nabla u^{(k)}(x)\, dx,$$

which, with the help of the gradient formulae (8.9) and (8.10), can be easily expressed in terms of several moments of the following types:

$$\mu_\nu^{d-1}(F\nabla u^{(k)}, T), \quad \nu \in \mathcal{I}_3^{d-1}, \quad \text{if } \Omega_k = T \in \mathcal{T}_O,$$
$$\mu_\nu^{2d-1}(\tilde{F}^T \nabla\{u^{(k)}(\phi_T)\}, T_0), \quad \nu \in \mathcal{I}_3^{2d-1}, \quad \text{if } \Omega_k = T \in \mathcal{T}_I,$$
$$\mu_\nu^{\boldsymbol{d}-e_i}(\tilde{F}^H \nabla\{u^{(k)}(\phi_H)\}, H_0), \quad \nu \in \mathcal{J}_3^{\boldsymbol{d}-e_i},\ i = 1,2,3, \quad \text{if } \Omega_k = H \in \mathcal{H}.$$

Note that the values of the vector-polynomials $\nabla u^{(k)}$ or $\nabla\{u^{(k)}(\phi_T)\}$ at the Stroud points in T or T_0, respectively, as needed for the above moment computations, can be efficiently computed with the help of Algorithm 1 of [3] in $\mathcal{O}(d^4)$ operations. The same count applies to the evaluation of $\nabla\{u^{(k)}(\phi_H)\}$ at the tensor-product Gauss points in H_0 performed by the analogous (and simpler) method. Since the moments themselves require $\mathcal{O}(d^4)$ operations, we arrive at a total cost of $\mathcal{O}(Nd^4)$ for the computation of the product $\hat{S}\hat{c}$. Finally, Sc is computed as the product $A^t\tilde{c}$. We summarize the computation of Sc in the following algorithm.

Algorithm 8.12. *Matrix-vector product Sc.*

1) *Use Algorithm 8.2 to compute* $\hat{c} = Ac$.
2) *Compute* $\tilde{c} = \hat{S}\hat{c}$ *as described above.*
3) *Compute* $Sc = A^t\tilde{c}$ *using Algorithm 8.3.*

By taking into account the discussion of the cost of Algorithms 8.2 and 8.3 at the end of Section 8.1, we see that the total cost of Algorithm 8.12 is $\mathcal{O}(Nd^4)$, and its storage requirement is $\mathcal{O}(Nd^3)$.

We refer the reader to Section 3.4 of [3] for information on how Bernstein-Bézier moments can be used for solving non-linear equations.

§9. A nodal minimal determining set for $\mathcal{S}_d^0(\triangle)$

Let

$$\mathcal{N} := \{\gamma_\xi\}_{\xi\in\mathcal{D}_{d,\triangle}},$$

where $\mathcal{D}_{d,\triangle}$ is the set of domain points described in (5.1), and where γ_t is the linear functional that produces the value of a function at the point t, that is $\gamma_t f = f(t)$.

Theorem 9.1. *The set* $\mathcal{N}$ *is a nodal minimal determining set for* $\mathcal{S}_d^0(\triangle)$*, and in particular, for any function* $f \in C(\Omega)$*, there exists a unique* $s \in \mathcal{S}_d^0(\triangle)$ *such that*

$$\gamma_\xi s = \gamma_\xi f, \qquad \text{all } \xi \in \mathcal{D}_{d,\triangle}.$$

Proof: If H is a hexahedron, then we can find a tensor-product polynomial $p \in \mathcal{Q}_d$ that interpolates the values $f(\xi)$, $\xi \in \mathcal{D}_{d,H}$, at the $(d+1)^3$ domain points $\phi_H^{-1}(\xi) \in \mathcal{D}_{d,H_0}$. This determines the B-coefficients of $s|_H = p \circ \phi_H^{-1}$. Similarly, for each ordinary tetrahedron T, we can find a polynomial $p \in \mathcal{P}_d$ that interpolates f at the $\binom{d+3}{3}$ domain points in T. This determines the B-coefficients of $s|_T$, see Theorem 5.38 in [21]. Now suppose T is an interface tetrahedron. If F_i, $i = 1, 2$, is a face of T shared with an ordinary tetrahedron T_i, then we get all coefficients of a bivariate polynomial p_i of total degree $2d$ by Lemma 7.2, and evaluate it at all domain points in $\mathcal{D}_{2d,F_i}$. Similarly, for the face F_3 shared with a hexahedron H, we obtain the coefficients of a bivariate polynomial p_3 of total degree $2d$ by Lemma 7.3, and evaluate it at all domain points in $\mathcal{D}_{2d,\phi_T^{-1}(F_3)}$. Now, a polynomial $p \in \mathcal{P}_{2d}$ is determined by the interpolation at the domain points in $\mathcal{D}_{2d,T_0}$ as follows: $p(\phi_T^{-1}(\xi)) = p_i(\xi)$ for all $\xi \in F_i \cap \mathcal{D}_{2d,T}$, $i = 1, 2$, $p(\phi_T^{-1}(\xi)) = p_3(\phi_H^{-1}(\xi))$ for all $\xi \in F_3 \cap \mathcal{D}_{2d,T}$, and $p(\phi_T^{-1}(\xi)) = f(\xi)$ for all remaining $\xi \in \mathcal{D}_{2d,T}$. Finally, we set $s|_T = p \circ \phi_T^{-1}$. Theorem 5.3 ensures that the function s defined in this way belongs to $\mathcal{S}_d^0(\triangle)$, and that the interpolation problem has a unique solution. □

There is a natural dual basis associated with the nodal minimal determining set $\mathcal{N}$. In particular, for each $\xi \in \mathcal{M}$, we define ℓ_ξ to be the unique spline in $\mathcal{S}_d^0(\triangle)$ such that

$$\ell_\xi(\eta) = \delta_{\xi,\eta}, \qquad \text{all } \eta \in \mathcal{D}_{d,\triangle}. \tag{9.1}$$

The basis $\{\ell_\xi\}_{\xi\in\mathcal{M}}$ is different from the one constructed in Theorem 6.2. We refer to it as a Lagrange basis. These basis functions have the same support as the basis functions in the set Ψ of Theorem 6.2, and could also be used to compute with $\mathcal{S}_d^0(\triangle)$. However, due to the advantages of the Bernstein–Bézier approach for computations, we recommend using the basis Ψ.

§10. Approximation power of $\mathcal{S}_d^0(\triangle)$

We conclude the paper with a result showing how well smooth functions can be approximated by $\mathcal{S}_d^0(\triangle)$. Given $f\in C(\Omega)$, we define

$$\mathcal{I}f := \sum_{\xi\in\mathcal{D}_{d,\triangle}} f(\xi)\,\ell_\xi,$$

where the ℓ_ξ are the Lagrange basis functions of the previous section. Then by (9.1), $\mathcal{I}f$ interpolates f at all domain points in $\mathcal{D}_{d,\triangle}$. Moreover, if $f|_R = 0$ for a boundary edge or face R of $\triangle$, then also $\mathcal{I}f|_R = 0$.

Now given an integer m and a real number $1\le q\le\infty$, let $W_q^m(\Omega)$ be the usual Sobolev space with associated seminorm

$$|f|_{m,q,\Omega} := \begin{cases} \Big[\sum_{|\alpha|=m}\|D^\alpha f\|_{q,\Omega}^q\Big]^{1/q}, & \text{if } 1\le q<\infty,\\ \max_{|\alpha|=m}\|D^\alpha f\|_\Omega, & \text{if } q=\infty,\end{cases}$$

and norm

$$\|f\|_{m,q,\Omega} := \max_{0\le k\le m}|f|_{k,q,\Omega}.$$

Here $\alpha=(\alpha_1,\alpha_2,\alpha_3)$ is understood as a multi-index, and $D^\alpha := D_x^{\alpha_1}D_y^{\alpha_2}D_z^{\alpha_3}$.

We also need the shape parameter associated with ordinary tetrahedra in $\triangle$ defined by

$$\kappa_\triangle^O := \max_{T\in\mathcal{T}_O}\frac{|T|}{\rho_T},$$

where $|\Omega_k|$ denotes the diameter of Ω_k, and ρ_T is the radius of the largest ball that can be inscribed in T. We set

$$|\triangle| = \max_{1\le k\le N}|\Omega_k|.$$

Recall that a bijective mapping $\phi:\overline{G_0}\to\overline{G}$ with G_0, G domains in $\mathbb{R}^n$ is said to be m-smooth if all components of ϕ and ϕ^{-1} are m times continuously differentiable [1]. Let

$$\|D\phi\|_{k,\infty,G_0} := \max_{1\le i\le n}\max_{|\alpha|=1}\|D^\alpha\phi_i\|_{k,\infty,G_0},\quad 0\le k\le m-1.$$

For any mapping ϕ, we write J_ϕ for its Jacobian. We will need the following extension of Theorem 3.41 of [1].

Lemma 10.1. *Let $f \in W_q^m(G)$, $1 \le q \le \infty$, $m \ge 1$, and let $\phi : \overline{G_0} \to \overline{G}$ be m-smooth. Then for any $\alpha \in \mathbb{Z}_+^n$ with $|\alpha| = m$,*

$$\|D^\alpha(f \circ \phi)\|_{q,G_0} \le K\|J_\phi\|_{\infty,G_0}^{-1/q} \sum_{\substack{\beta \le \alpha \\ |\beta| \ne 0}} (1 + \|D\phi\|_{m-|\beta|,\infty,G_0}^{|\beta|})\|D^\beta f\|_{q,G},$$

where K depends only on m and n.

Proof: By the chain rule

$$D^\alpha(f \circ \phi) = \sum_{\substack{\beta \le \alpha \\ |\beta| \ne 0}} M_{\alpha\beta}(\phi) \cdot (D^\beta f) \circ \phi,$$

where $M_{\alpha\beta}(\phi)$ is a polynomial of degree not exceeding $|\beta|$ in derivatives of orders between 1 and $m - |\beta| + 1$ of the functions ϕ_i, see the proof of Theorem 3.41 of [1]. Since the coefficients of this polynomial only depend on α, β, it follows that

$$\|M_{\alpha\beta}(\phi)\|_{\infty,G_0} \le K(1 + \|D\phi\|_{m-|\beta|,\infty,G_0}^{|\beta|}),$$

where K depends only on m and n. In addition, for $q < \infty$ a substitution delivers the Jacobian factor,

$$\int_{G_0} |D^\beta f(\phi(y))|^q \, dy = \int_G |D^\beta f(x)|^q |J_{\phi^{-1}}(x)| \, dx \le \|J_\phi\|_{\infty,G_0}^{-1} \int_G |D^\beta f(x)|^q \, dx. \quad \square$$

To factor out the influence of the different sizes of the domains G_0 and G, we introduce the mappings

$$\begin{aligned} &\bar\phi : \bar G_0 \to G, \quad \bar G_0 := |G| G_0, \quad \bar\phi(y) := \phi(|G|^{-1} y), \\ &\hat\phi : \hat G \to G_0, \quad \hat G := |G|^{-1} G, \quad \hat\phi(y) := \phi^{-1}(|G| y), \end{aligned}$$

derived from a mapping $\phi : G_0 \to G$.

Lemma 10.2. *Under the hypotheses of Lemma 10.1,*

$$\|D^\alpha(f \circ \phi)\|_{q,G_0} \le K|G|^{m-n/q} |\hat\phi|_{1,\infty,\hat G}^{n/q} (1 + \|D\bar\phi\|_{m-1,\infty,\bar G_0}^m)\|f\|_{m,q,G}, \qquad (10.1)$$

and

$$\|D^\alpha f\|_{q,G} \le K|G|^{-m+n/q} |\bar\phi|_{1,\infty,\bar G_0}^{n/q} (1 + \|D\hat\phi\|_{m-1,\infty,\hat G}^m)\|f \circ \phi\|_{m,q,G_0}, \qquad (10.2)$$

where K depends only on m and n.

Proof: To show (10.1), we use the identity

$$\|D^\alpha(f \circ \phi)\|_{q,G_0} = |G|^{m-n/q} \|D^\alpha(f \circ \bar\phi)\|_{q,\bar G_0},$$

apply Lemma 10.1 with ϕ replaced by $\bar{\phi}$, and take into account that

$$\|D\bar{\phi}\|^{|\beta|}_{m-|\beta|,\infty,\bar{G}_0} \le \|D\bar{\phi}\|^{m}_{m-1,\infty,\bar{G}_0}, \quad 0 < |\beta| \le m,$$

and $\bar{\phi}^{-1}(x) = |G|\hat{\phi}(|G|^{-1}x)$, which implies

$$\|J_{\bar{\phi}}\|^{-1}_{\infty,\bar{G}_0} = \|J_{\bar{\phi}^{-1}}\|_{\infty,G} = \|J_{\hat{\phi}}\|_{\infty,\hat{G}} \le |\hat{\phi}|^{n}_{1,\infty,\hat{G}}.$$

Similarly, (10.2) follows by an application of Lemma 10.1 with ϕ replaced by $\hat{\phi}$, and taking into account the identities

$$\|D^\alpha f\|_{q,G} = |G|^{-m+n/q}\|D^\alpha(f\circ\phi\circ\hat{\phi})\|_{q,\hat{G}}, \quad \hat{\phi}^{-1}(x) = |G|^{-1}\bar{\phi}(|G|x). \quad \square$$

We define the shape parameters associated with hexahedra in $\mathcal{H}$ by

$$\kappa_{H,m} := \max\{\|D\bar{\phi}_H\|_{m-1,\infty,\bar{H}_0}, \|D\hat{\phi}_H\|_{m-1,\infty,\hat{H}}\}, \qquad H \in \mathcal{H},$$

and set

$$\kappa^{\mathcal{H}}_{\triangle,m} := \max_{H\in\mathcal{H}} \kappa_{H,m}.$$

Similarly, we define the shape parameters associated with interface tetrahedra in $\mathcal{T}_I$ by

$$\kappa_{T,m} := \max\{\|D\bar{\phi}_T\|_{m-1,\infty,\bar{T}_0}, \|D\hat{\phi}_T\|_{m-1,\infty,\hat{T}}\}, \qquad T \in \mathcal{T}_I,$$

and set

$$\kappa^{I}_{\triangle,m} := \max_{T\in\mathcal{T}_I} \kappa_{T,m}.$$

We note that this definition is consistent with the notion of the shape parameter of an ordinary tetrahedra since in the case when ϕ_T is an affine mapping,

$$\kappa_{T,m} \le K\frac{|T|}{\rho_T},$$

where K is an absolute constant. Indeed, let $\phi_T(x) = Ax + b$ for some $A = [a_{i,j}]^3_{i,j=1} \in \mathbb{R}^{3\times 3}$ and $b \in \mathbb{R}^3$. Then

$$\kappa_{T,m} = \max\{|T|^{-1}\max_{i,j}|a_{i,j}|,\ |T|\max_{i,j}|b_{i,j}|\},$$

where $B = [b_{i,j}]^3_{i,j=1} = A^{-1}$, and the bound follows in view of Theorem 3.1.3 of [13], which says that $\|A\|_2 \le |T|/\rho_{T_0}$ and $\|B\|_2 \le |T_0|/\rho_T$.

We recall that $\mathcal{H}$, $\mathcal{T}_O$, and $\mathcal{T}_I$ are the sets of hexahedra, ordinary tetrahedra, and interface tetrahedra in the partition $\triangle_R$.

Theorem 10.3. *Suppose $f \in W_q^m(\Omega)$ with $3/q < m \le d+1$ for some $q \ge 1$, and all mappings ϕ_H, $H \in \mathcal{H}$ and ϕ_T, $T \in \mathcal{T}_I$, are bijective and m-smooth. Given an element Ω_k of $\triangle_R$, we define $\tilde{\Omega}_k = \Omega_k$ if $\Omega_k \in \mathcal{H} \cup \mathcal{T}_O$. If Ω_k is an interface tetrahedron, we define $\tilde{\Omega}_k = \Omega_k \cup H \cup T_1 \cup T_2$, where $H \in \mathcal{H}$ is the hexahedron and $T_1, T_2 \in \mathcal{T}_O$ are the two ordinary tetrahedra that share a face with Ω_k. Then*

$$\|D^\alpha(f - \mathcal{I}f)\|_{q,\Omega_k} \le K|\tilde{\Omega}_k|^{m-|\alpha|}\|f\|_{m,q,\tilde{\Omega}_k}, \qquad 1 \le k \le N, \tag{10.3}$$

for all α with $|\alpha| < m$. The constant K depends only on d and the shape parameters $\kappa_\triangle^O$, $\kappa_{\triangle,m}^{\mathcal{H}}$ and $\kappa_{\triangle,m}^I$.

Proof: Clearly, $\mathcal{I}f$ is well defined since $W_q^m(\Omega) \subset C(\Omega)$ by Sobolev embedding as soon as $m > 3/q$. If $\Omega_k = T \in \mathcal{T}_O$, then (10.3) holds with a constant depending only on d and $|T|/\rho_T$, and even with the seminorm $|f|_{m,q,T}$ rather than the norm $\|f\|_{m,q,T}$ on the right hand side. This follows by standard estimates of the error of polynomial interpolation, see e.g. Theorem 4.4.4 in [10].

Let Ω_k be a hexagon H. Then the estimates of the error of tensor-product polynomial interpolation, see e.g. Theorem 4.6.11 in [10], imply

$$\|D^\alpha[(f - \mathcal{I}f) \circ \phi_H]\|_{q,H_0} \le K_1|f \circ \phi_H|_{m,q,H_0}, \qquad |\alpha| < m,$$

with a constant K_1 depending only on d (since $m \le d+1$). Since ϕ_H is m-smooth, it follows by (10.1) that

$$|f \circ \phi_H|_{m,q,H_0} \le K_2|H|^{m-3/q}\|f\|_{m,q,H}, \tag{10.4}$$

where K_2 depends only on d and $\kappa_{H,m}$. Similarly, by (10.2)

$$\|D^\alpha(f - \mathcal{I}f)\|_{q,H} \le K_3|H|^{-|\alpha|+3/q}\|(f - \mathcal{I}f) \circ \phi_H\|_{|\alpha|,q,H_0}$$

where K_3 depends only on d and $\kappa_{H,m}$. Combining the last three inequalities leads to (10.3).

It remains to consider the case when Ω_k is an interface tetrahedron T in $\mathcal{T}_I$. For $i = 1, 2$, let F_i be the face of T shared with the ordinary tetrahedron T_i, and let F_3 be the face shared with the hexahedron H. Furthermore, let $p_i = \mathcal{I}f|_{T_i} \in \mathcal{P}_d$, $i = 1, 2$, and $p_3 = \mathcal{I}f \circ \phi_H \in \mathcal{Q}_d$. By Corollary 4.4.7 in [10],

$$\|f - p_i\|_{\infty,T_i} \le K_4|T_i|^{m-3/q}|f|_{m,q,T_i}, \qquad i = 1, 2,$$

where K_4 depends only on d and $|T_i|/\rho_{T_i}$. In particular

$$|f(\xi) - p_i(\xi)| \le K_4|T_i|^{m-3/q}|f|_{m,q,T_i}, \quad \text{for all } \xi \in F_i \cap \mathcal{D}_{2d,T}, \quad i = 1, 2.$$

By Theorem 4.6.11 in [10],

$$\|f \circ \phi_H - p_3\|_{\infty,H_0} \le K_5|f \circ \phi_H|_{m,q,H_0},$$

where K_5 depends only on d. Hence

$$|f(\xi) - p_3(\phi_H^{-1}(\xi))| \le K_5 |f \circ \phi_H|_{m,q,H_0}, \quad \text{for all } \xi \in F_3 \cap \mathcal{D}_{2d,T}.$$

Let $\mathcal{I}_0 : C(T_0) \to \mathcal{P}_{2d}$ denote the polynomial interpolation operator with interpolation nodes in $\mathcal{D}_{2d,T_0}$, and let $p = \mathcal{I}f \circ \phi_T$ and $\tilde{p} = \mathcal{I}_0(f \circ \phi_T)$. Then $p, \tilde{p} \in \mathcal{P}_{2d}$ and, by the definition of $\mathcal{I}f$,

$$(\tilde{p} - p)(\phi_T^{-1}(\xi)) = \begin{cases} f(\xi) - p_i(\xi), & \text{if } \xi \in F_i \cap \mathcal{D}_{2d,T},\ i = 1, 2, \\ f(\xi) - p_3(\phi_H^{-1}(\xi)), & \text{if } \xi \in F_3 \cap \mathcal{D}_{2d,T}, \\ 0, & \text{if } \xi \in \mathcal{D}_{2d,T} \setminus (F_1 \cup F_2 \cup F_3). \end{cases}$$

Now

$$\tilde{p} - p = \mathcal{I}_0(\tilde{p} - p),$$

and in view of the boundedness of the operator $\mathcal{I}_0 : C(T_0) \to C(T_0)$ and the Markov inequality, there exists a constant K_6 depending only on d such that

$$\|D^\alpha(\tilde{p} - p)\|_{q,T_0} \le K_6 \max_{\xi \in \mathcal{D}_{2d,T}} |(\tilde{p} - p)(\phi_T^{-1}(\xi))|.$$

By Theorem 4.4.4 in [10],

$$\|D^\alpha(f \circ \phi_T - \tilde{p})\|_{q,T_0} \le K_7 |f \circ \phi_T|_{m,q,T_0}, \qquad |\alpha| < m,$$

where K_7 depends only on d. Hence

$$\begin{aligned} \|D^\alpha[(f - \mathcal{I}f) \circ \phi_T]\|_{q,T_0} &\le \|D^\alpha(f \circ \phi_T - \tilde{p})\|_{q,T_0} + \|D^\alpha(\tilde{p} - p)\|_{q,T_0} \\ &\le K_7 |f \circ \phi_T|_{m,q,T_0} + K_6\big(K_4 \sum_{i=1}^{2} |T_i|^{m-3/q} |f|_{m,q,T_i} + K_5 |f \circ \phi_H|_{m,q,H_0}\big). \end{aligned}$$

Taking into account (10.4) and Lemma 10.2 which implies

$$|f \circ \phi_T|_{m,q,T_0} \le K_8 |T|^{m-3/q} \|f\|_{m,q,T},$$

and the following inequality

$$\|D^\alpha(f - \mathcal{I}f)\|_{q,T} \le K_9 |T|^{-|\alpha|+3/q} \|(f - \mathcal{I}f) \circ \phi_T\|_{|\alpha|,q,T_0},$$

with constants K_8, K_9 depending only on d and $\kappa_{T,m}$, we arrive at (10.3). □

§11. Remarks

Remark 1. In addition to pyramids, prismatic elements are often helpful in creating interfaces between hexahedra and tetrahedra, see e.g. [19]. Bernstein-Bézier shape functions for them can be easily constructed as described in Sect. 5 of [3], using tensor products of triangle and univariate shape functions.

Remark 2. Given the B-coefficients of a spline $s \in \mathcal{S}_d^0(\triangle)$, Algorithm 8.2 shows how to convert it to a collection of tensor-product and tetrahedral Bézier volumes, which are standard objects in Computer Aided Design. This can be helpful in visualizing the spline.

Remark 3. The problem of defining suitable shape functions for pyramids is complex as it turns out to be impossible to use polynomials, or even any C^1 functions on a reference pyramid with a square base, see [29]. To deal with a pyramid P, several authors proposed the use of rational polynomials [7,8,15,23], while others [9,11,19,22,29] prefer to split the pyramid into two or more pieces, each of which is an image of the reference tetrahedron T_0. We follow the latter approach.

Remark 4. The THP partitions introduced in Sect. 2 are examples of 3D mixed meshes consisting of hexahedra and quadratic tetrahedral Bernstein–Bézier volumes, where two or more quadratic tetrahedra share faces with subsets of a face of a hexahedron, and/or have a hanging edge inside the face of a hexahedron. 3D mixed meshes are natural 3D analogs of the TR-meshes discussed in [28] which consist of a mixture of triangles and rectangles, where hanging vertices are allowed. In principle, the interface between hexahedral and tetrahedral meshes can be filled with quadratic tetrahedra in a more flexible way without the restriction that their pairs form pyramids. This may be beneficial for quality mesh generation. To be practical, such 3D mixed meshes should not be completely arbitrary, and in particular should not allow cycles which greatly complicate the 2D case of mixed meshes discussed in [28].

Remark 5. Our methods can be easily extended to the case where the pyramid is subdivided into four interface tetrahedra by splitting its base using two crossing hanging edges as suggested in [22]. Indeed, as in the case of one hanging edge, we can transform the tensor-product B-form of degree d into triangle B-forms of degree $2d$ on each of the four pieces of the base of the pyramid, and we can deal with the tetrahedra sharing faces with the pyramid by degree raising while defining the shape functions with the help of the B-forms of degree $2d$ in the four interface tetrahedra.

Remark 6. We can reduce the dimension of the space $\mathcal{S}_d^0(\triangle)$ without reducing the approximation order of the method by setting $\mathcal{P}_T := \{p \circ \phi_T^{-1} : p \in \mathcal{P}\}$ for any interface tetrahedron T, where $\mathcal{P}$ is any subspace of $\mathcal{P}_{2d}$ such that a) $\mathcal{P}_d \subset \mathcal{P}$ and b) the restriction of $\mathcal{P}$ to the face F of the interface tetrahedron T shared with a hexahedron includes all bivariate tensor-product polynomials of degree d. It is easy to see that the smallest possible dimension for $\mathcal{P}$ is $\binom{d+2}{3} + (d+1)^2$ if we for

example extend the Bernstein basis for $\mathcal{P}_d$ with respect to T by $\binom{d+1}{2}$ Bernstein polynomials of degree up to $2d$ associated with the face F that are needed to ensure b). A further reduction of the dimension of $\mathcal{S}_d^0(\triangle)$ can be achieved by imposing higher order smoothness conditions across the common face of the pair of interface tetrahedra that form a pyramid. However, any construction of this type would be significantly more difficult to use than the straightforward choice $\mathcal{P} = \mathcal{P}_{2d}$ suggested in this paper. In particular, Algorithms 8.2 and 8.3 needed for the efficient implementation relying on BB-moments would be much more complex. If the number of pyramids in a THP partition is very small comparing to the total number of elements (which is expected due to their role as interface elements between hexahedra and tetrahedra), then the savings from using a smaller space $\mathcal{P}_T$ do not reduce the overall cost of the method by a significant amount. Moreover, all of the extra basis functions allowed by our construction are supported on a pyramid, and therefore can be dealt efficiently with by the standard techniques of static condensation of internal degrees of freedom, see e.g. Sect. 4.2.3 of [18].

Remark 7. As required in the *hp*-method, different polynomial degrees can be used on different tetrahedra and hexahedra. In this case the global degrees of freedom can be handled with the help of degree raising, similar to Lemma 7.2, or following the scheme suggested in [2]. Moreover, anisotropic tensor-product polynomial spaces

$$\mathcal{Q}_{(n,m,\ell)} = \operatorname{span}\{B_{ijk}^{(n,m,\ell)} : 0 \le i \le n,\, 0 \le j \le m,\, 0 \le k \le \ell\}$$

can be used on hexahedra if $\mathcal{P}_H$ is replaced by $\{p \circ \phi_H^{-1} : p \in \mathcal{Q}_{(n,m,\ell)}\}$. In fact, in Sect. 7 we have provided the formulae for the general degree raising matrices $R^{n,m}$ and tensor-product/total-degree polynomial transformation matrices $T^{k,s}$ needed for the implementation of the generalized transformation matrices arising in the methods based on varying degrees on tetrahedra and anisotropic hexahedra.

Remark 8. It is also possible to use serendipity elements (see [5]) on hexahedra. To this end, the expansion of the degree d serendipity polynomials in terms of the tensor-product Bernstein polynomials would need to be implemented.

Remark 9. In the case $d = 1$, the restriction of $\mathcal{S}_d^0(\triangle)$ to a pyramid coincides with the space suggested in [9,19,29]. For $d = 2$ the dimension of this restriction is 19, whereas the quadratic space in [9] has 14 degrees of freedom. Note that two out of five additional basis functions are supported on one interface tetrahedron each, and three are associated with the common face of the interface tetrahedra and supported on the pyramid. The quadratic pyramidal element of [29] has 13 degrees of freedom because it uses a serendipity space on the base of the pyramid. The higher order pyramidal elements of [9] provide an example of the element with reduced dimension discussed in Remark 6.

Remark 10. We have avoided the 'reference pyramid' approach of [9,19,29] because this eliminates the need for any separate 'pyramidal BB moments' by resorting directly to the standard tetrahedral moments.

Remark 11. The development of finite elements of $H(div)$ and $H(curl)$ type for THP partitions that take full advantage of Bernstein–Bézier techniques would be of considerable practical interest. However, while suitable bases have been developed for the 2D case (see [4]), the development of Bernstein–Bézier bases for $H(div)$ and $H(curl)$ conforming spaces on pure tetrahedral triangulations in 3D is still in a state of flux.

Remark 12. Using the partition of unity property of Bernstein polynomials and the product formula (7.5), we get

$$p := \sum_{\nu \in \mathcal{I}_2^n} c_\nu B_\nu^{n,F} = \sum_{\nu \in \mathcal{I}_2^n,\, \lambda \in \mathcal{I}_2^{m-n}} c_\nu B_\nu^{n,F} B_\lambda^{m-n,F} = \sum_{\nu \in \mathcal{I}_2^n,\, \lambda \in \mathcal{I}_2^{m-n}} c_\nu \frac{\binom{\nu+\lambda}{\nu}}{\binom{m}{n}} B_{\nu+\lambda}^{m,F},$$

which implies that the B-coefficients $\tilde{c}$ of p as a polynomial of degree m are given by

$$\tilde{c}_\mu = \sum_{\substack{\nu \in \mathcal{I}_2^n \\ \nu \le \mu}} \frac{\binom{\mu}{\nu}}{\binom{m}{n}} c_\nu, \quad \mu \in \mathcal{I}_2^m.$$

This shows that the matrix $R^{n,m}$ in (7.2) is given by

$$R_{\mu,\nu}^{n,m} = \begin{cases} M_{\nu,\mu-\nu}^{n,m-n}, & \text{if } \nu \le \mu, \\ 0, & \text{otherwise,} \end{cases}$$

where $M^{n,m-n}$ is the trinomial matrix described in (7.6).

Remark 13. The convective matrix V with entries

$$V_{\xi\zeta} = \int_\Omega \psi_\xi(x) f(x) \nabla \psi_\zeta(x)\, dx, \quad \xi, \zeta \in \mathcal{M},$$

where $f : \Omega \to \mathbb{R}^3$ is a given vector function, can be treated in a similar way to the mass and stiffness matrices, compare Sect. 4.3 of [3].

Remark 14. The transformation matrix A defined in Sect. 8.1 maps the B-coefficients of a spline in $\mathcal{S}_d^0(\triangle)$ to the vector of all coefficients of the individual shape functions. Thus, it is essentially a mapping from $\mathcal{S}_d^0(\triangle)$ to a space of piecewise functions with no continuity between the pieces. The transformation matrices used in [26,27] are a little different since they map coefficients of a spline in a smooth subspace of C^0 splines to coefficients of a C^0 spline.

References

1. Adams, R. A. and J. F. Fournier, *Sobolev Spaces*, Academic Press, Amsterdam, 2003.

2. Ainsworth, M., Pyramid algorithms for Bernstein-Bézier finite elements of high, nonuniform order in any dimension, SIAM J. Scient. Computing **36** (2014), A543–A569.
3. Ainsworth, M., G. Andriamaro, and O. Davydov, Bernstein-Bézier finite elements of arbitrary order and optimal assembly procedures, SIAM J. Scient. Computing **33** (2011), 3087–3109.
4. Ainsworth, M., G. Andriamaro, and O. Davydov, A Bernstein-Bézier basis for arbitrary order Raviart-Thomas finite elements, Constr. Approx. **41** (2015), 1–22.
5. Arnold, Douglas N. and Gerard Awanou, The serendipity family of finite elements, Foundations of Computational Mathematics **11** (2011), 337–344.
6. Baudouin, Tristan Carrier, Jean-Francois Remacle, Emilie Marchandise, Francois Henrotte, and Christophe Geuzaine, A frontal approach to hex-dominant mesh generation, Advanced Modeling and Simulation in Engineering Sciences **1:8** (2014), 1–30.
7. Bedrosian, G., Shape functions and integration formulas for three-dimensional finite element analysis, International Journal for Numerical Methods in Engineering **35** (1992), 95–108.
8. Bergot, Morgane, Gary Cohen, and Marc Duruflé, Higher-order finite elements for hybrid meshes using new nodal pyramidal elements, Journal of Scientific Computing **42** (2010), 345–381.
9. Bluck, M. J. and S. P. Walker, Polynomial basis functions on pyramidal elements, Communications in Numerical Methods in Engineering **24** (2008), 1827–1837.
10. Brenner, S. C. and L. R. Scott, *The Mathematical Theory of Finite Element Methods,* Springer-Verlag, New York, 2002.
11. Chan, J. and T. Warburton, A comparison of high order interpolation nodes for the pyramid, SIAM J. Sci. Comput. **37** (2015), A2151–A2170.
12. Ciarlet, P.G., Basic error estimates for elliptic problems, in: P.G. Ciarlet and J.L. Lions, eds., Handbook of numerical analysis, Vol. II: Finite Element Methods (Part 1), North-Holland, Amsterdam, 1991, pp.17–351.
13. Ciarlet, P.G., *The Finite Element Method for Elliptic Problems, 2nd edition,* Society for Industrial and Applied Mathematics, Philadelphia, 2002.
14. Erickson, Jeff, Efficiently hex-meshing things with topology, Discrete & Computational Geometry **52** (2014), 427–449.
15. Fuentes, F., B. Keith, L. Demkowicz, and S. Nagaraj, Orientation embedded high order shape functions for the exact sequence elements of all shapes, Comput. Math. Appl. **70** (2015), 353–458.
16. Goldman, R. N. and D. J. Filip, Conversion from Bézier rectangles to Bézier triangles, Computer-Aided Design **19** (1987), 25–27.

17. Hoschek, J. and D. Lasser, *Fundamentals of Computer Aided Geometric Design,* AK Peters, Wellesley, MA, 1993.
18. Karniadakis, George Em and Spencer J. Sherwin, *Spectral/hp Element Methods for Computational Fluid Dynamics. 2nd edition,* Oxford University Press, Oxford, 2005.
19. Knabner, P. and G. Summ, The invertibility of the isoparametric mapping for pyramidal and prismatic finite elements, Numer. Math. **88** (2001), 661-681.
20. Knabner, P., S Korotov, and G. Summ, Conditions for the invertibility of the isoparametric mapping for hexahedral finite elements, Finite Elements in Analysis and Design **40** (2003), 159–172.
21. Lai, M. J. and L. L. Schumaker, *Spline Functions on Triangulations,* Cambridge University Press, Cambridge, 2007.
22. Liu, Liping, Kevin B. Davies, Michal Krezek, and Li Guan, On higher order pyramidal finite elements, Adv. Appl. Math. Mech. **3** (2011), 131–140.
23. Nigam, Nilima and Joel Phillips, High-order conforming finite elements on pyramids, IMA J. Numer. Anal. **32** (2012), 448–483.
24. Nigam, Nilima and Joel Phillips, Numerical integration for high order pyramidal finite elements, ESAIM Math. Model. Numer. Anal. **46** (2012), 239–263.
25. Owen, Steve and Sunil Saigal, Formation of pyramid elements for hexahedra to tetrahedra transitions, Computer Methods in Applied Mechanics and Engineering **190** (2001), 4505–4518.
26. Schumaker, L. L., Computing bivariate splines in scattered data fitting and the finite element method, Numerical Algorithms **48** (2008), 237–260.
27. Schumaker, L. L., *Spline Functions: Computational Methods,* SIAM, Philadelphia, 2015.
28. Schumaker, L. L. and Lujun Wang, Spline Spaces on TR-meshes with hanging vertices, Numer. Math. **118** (2011), 531–548.
29. Wieners, Christian, Conforming discretizations on tetrahedrons, pyramids, prisms and hexahedrons, manuscript, 1997.