SIAM J. Sci. Comput.
Vol. 21, No. 6, pp. 2352–2380

STABLE SPECTRAL METHODS ON TETRAHEDRAL ELEMENTS*

J. S. HESTHAVEN[†] AND C. H. TENG[†]

Abstract. A framework for the construction of stable spectral methods on arbitrary domains with unstructured grids is presented. Although most of the developments are of a general nature, an emphasis is placed on schemes for the solution of partial differential equations defined on the tetrahedron.

In the first part the question of well-behaved multivariate polynomial interpolation on the tetrahedron is addressed, and it is shown how to extend the electrostatic analogy of the Jacobi polynomials to problems beyond the line. This allows for the identification of nodal sets suitable for polynomial interpolation within the tetrahedron and, subsequently, for the formulation of stable spectral schemes on such unstructured nodal sets. The second part of this work is devoted to a discussion of weakly imposed boundary conditions, and energy-stable schemes are formulated for a wide class of problems, exemplified by advection problems, advection-diffusion problems, and linear symmetric hyperbolic systems.

Finally, in the third part, issues related to computational efficiency and implementation of the schemes are discussed. The spectral accuracy of the approximation is confirmed through an example, and factorization methods for the efficient computation of derivatives on the general nodal sets within the d-simplex are developed, ensuring that the proposed schemes are competitive with tensor-product-based methods. In this last part we also show that the advective operator results in an $\mathcal{O}(n^{-2})$ restriction on the time-step, similar to that of spectral collocation methods employing a tensor-product-based approximation. The performance of the proposed scheme is illustrated by solving a wave problem on a triangulated domain, confirming the expected accuracy and stability.

Key words. spectral methods, asymptotic stability, penalty methods, tetrahedral elements

AMS subject classifications. 65L20, 65A70, 65A60, 41A10, 41A63

PII. S1064827598343723

1. Introduction.

The formulation of pseudospectral methods for solving partial differential equations has traditionally been confined to problems defined on very simple domains that can be smoothly mapped onto the d-cube, with d signifying the dimensionality of the cube. Choosing the one-dimensional Gauss quadrature points as the nodal sets, with the multidimensional generalization appearing through a tensor product, yields a highly efficient procedure for the construction of the multidimensional interpolating polynomials and the introduction of differentiation matrices in the spirit of classical pseudospectral methods. The need for geometric flexibility has traditionally been addressed by introducing a multidomain formulation, as exemplified by the spectral element method [1] for the solution of the incompressible Navier–Stokes equations and spectral multidomain methods for the solution of the compressible Navier–Stokes equations; see, e.g., [2]. However, these methods all suffer from the requirement of a conforming, or almost conforming, semistructured discretization of the computational domain. Not only does that complicate the implementation of such schemes, it also poses significant problems for the application of automated grid-generation.

[*] Received by the editors August 17, 1998; accepted for publication (in revised form) September 9, 1999; published electronically May 23, 2000.

http://www.siam.org/journals/sisc/21-6/34372.html

[†] Division of Applied Mathematics, Brown University, Box F, Providence, RI 02912 (Jan_Hesthaven@brown.edu, ct@cfm.brown.edu). The work of the first author was supported by NSF grant ASC-9504002, by DoE grant DE-FG02-95ER25239, and by DARPA/AFOSR grant F49620-96-1-0426.

Finite element or finite volume methods, on the other hand, traditionally employ a low-order approximation on the d-simplex, hence allowing for highly efficient grid discretization and automated unstructured mesh generation. However, the need for high-order approximation methods has, in recent years, become increasingly apparent and has lead to the development of high-order finite element methods, most notably the $h-p$ method [3] and the new spectral element methods [4, 5], which all ensure the geometric flexibility by using triangles or tetrahedra as the basic building blocks while maintaining the high spatial accuracy of traditional spectral methods.

The development of such methods hinges upon the application of more complex basis functions, specifically tailored to the accurate approximation of smooth functions defined on the d-simplex. Tensor-product-like methods based on Jacobi polynomials, as devised in [6] by employing the polynomials of [7, 8], supply the basis utilized in [4], while $h-p$ methods classically employ multivariate Lagrange interpolation. The former approach utilizes a special tensor-like combination of Jacobi polynomials to construct the basis in such a way that a time-step restriction scaling as $\mathcal{O}(n^{-2})$ results. Here n represents the order of the polynomial. Moreover, the existence of a tensor-product structure of the approximation is advantageous as it can be exploited to compute derivatives in $\mathcal{O}(n^4)$ operations. A disadvantage of this approach, however, is the lack of a corresponding nodal sets and, consequently, the need to perform all computations in modal space.

On the other hand, utilizing multivariate Lagrange polynomials as the basis tool of approximation, as is classically done in $h-p$ methods, requires the identification of collocation points on the d-simplex that ensures good behavior of the interpolating polynomials. While the identification of such nodal sets remains open in the general case, approximate optimal points for polynomial interpolation have recently been obtained [9, 10, 11, 12], thereby allowing for the construction of well-behaved high-order approximations of smooth functions defined on the d-simplex. Unfortunately, there is no tensor-product structure associated with these nodal sets, and the computation of derivatives scales asymptotically as $\mathcal{O}(n^{2d})$, which becomes prohibitive for large n.

It is the purpose of this paper to show that by combining a number of recent results and techniques we can develop high-order, efficient nodal-based collocation methods for the stable solution of partial differential equations on general domains with an emphasis on the tetrahedron. The central parts of this construction comprise the identification of nodal sets well suited for high-order interpolation and the construction of stable high-order methods on such unstructured grids. To address the first problem we shall further develop the ideas put forward in [11] relating the nodal sets to the solution of a problem of electrostatics. The resolution of the second concern relies on vast generalizations of ideas put forward in [13, 14] in which it is shown how to impose boundary conditions in a way that ensures that the scheme maintains stability irrespective of the choice of grid.

The theoretical developments of the scheme provides little information about the actual performance of the complete framework, a concern to which we shall devote considerable attention. In particular, we shall show how to exploit the delicate structure of the multivariate differentiation matrices to realize a sparse factorization that results in a very considerable speed-up of the complete setting, hence making it very competitive with schemes utilizing special polynomials on the d-simplex [4] and even traditional purely tensor-product-based approximations on the d-cube.

The remaining part of the paper is organized as follows. In section 2 we introduce the necessary notation and recall a few results on the treatment of curvilinear tetrahedra and multidimensional Lagrangian interpolation. These results are central to the global multivariate approximations developed in section 3 by using an electrostatic analogy and setting the stage for the developments in section 4, where we formulate spectral schemes for the approximation of scalar conservation laws defined on the tetrahedron. The schemes are shown to be L^2-stable and generalizations to hyperbolic systems and higher-order problems are briefly discussed. Section 5 is devoted to a detailed discussion of computational issues related to the proposed methods. In particular we show how to exploit the symmetries of the nodal sets to effectively factorize the differentiation matrices, hence resulting in a very competitive computationally framework. In this section we also discuss the discrete stability of the schemes and the time-step restriction associated with explicit temporal integration. The performance of the complete framework is illustrated through the solution of a linear wave problem. Section 6 provides a brief summary and suggestions for future work along the line of thinking introduced in this paper.

2. General concepts and notation. We shall consider the curvilinear tetrahedron, $\mathsf{D} \subset \mathsf{R}^3$, illustrated in Figure 1. While it is not required that the faces of the tetrahedron are planar, such an assumption will, as we shall see shortly, simplify the scheme as well as the analysis considerably. Moreover, in a general computational scenario the vast majority of the tetrahedral elements will be straight sided, thus representing the most important special case.

We assume that there exists a diffeomorphism, $\Psi : \mathsf{D} \to \mathsf{I}$, where $\mathsf{I} \subset \mathsf{R}^3$ represents the standard tetrahedron with the vertices

$$\boldsymbol{v}_{\mathrm{I}} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{v}_{\mathrm{II}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \boldsymbol{v}_{\mathrm{III}} = \begin{bmatrix} 0 \\ \sqrt{3} \\ 0 \end{bmatrix}, \quad \boldsymbol{v}_{\mathrm{IV}} = \frac{1}{\sqrt{3}} \begin{bmatrix} 0 \\ 1 \\ 2\sqrt{2} \end{bmatrix},$$

forming four equilateral triangular faces as illustrated in Figure 1. The corresponding vertices in D are named $\boldsymbol{v}_1 - \boldsymbol{v}_4$ with $\boldsymbol{v}_1$ being related to $\boldsymbol{v}_{\mathrm{I}}$ and so on. Moreover, we shall refer to the four faces in I as well as in D as "a–d", where no confusion is possible. The faces are named such that face "a" has base-vertex $\boldsymbol{v}_{\mathrm{I}}$ in I and $\boldsymbol{v}_1$ in D and correspondingly for the other three faces.

Finally, we shall term the coordinates, $\boldsymbol{x} \in \mathsf{D}$, as (x, y, z) while the coordinates, $\boldsymbol{\xi} \in \mathsf{I}$, are named (ξ, η, ζ).

2.1. Curvilinear geometry and the tetrahedron. Let us initially assume that the global map, $\Psi : \mathsf{D} \to \mathsf{I}$, has been established in some form. Spatial derivatives in D of a function, $f(\boldsymbol{x})$, are then computed utilizing the curvilinear space with the metric of the transformation, $\nabla\xi$, $\nabla\eta$, and $\nabla\zeta$ given as

$$(1) \qquad \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{x}} = \begin{bmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{bmatrix} \begin{bmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Within this new metric, the divergence of a vector field, $\boldsymbol{F} = (F_x, F_y, F_z)$, becomes

$$\nabla \cdot \boldsymbol{F} = \frac{1}{J} \left[\frac{\partial}{\partial \xi} (J\boldsymbol{F} \cdot \nabla\xi) + \frac{\partial}{\partial \eta} (J\boldsymbol{F} \cdot \nabla\eta) + \frac{\partial}{\partial \zeta} (J\boldsymbol{F} \cdot \nabla\zeta) \right],$$

with the transformation Jacobian on the form

$$J = \left| \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}} \right| = \frac{1}{\nabla\xi \cdot (\nabla\eta \times \nabla\zeta)} \quad .$$

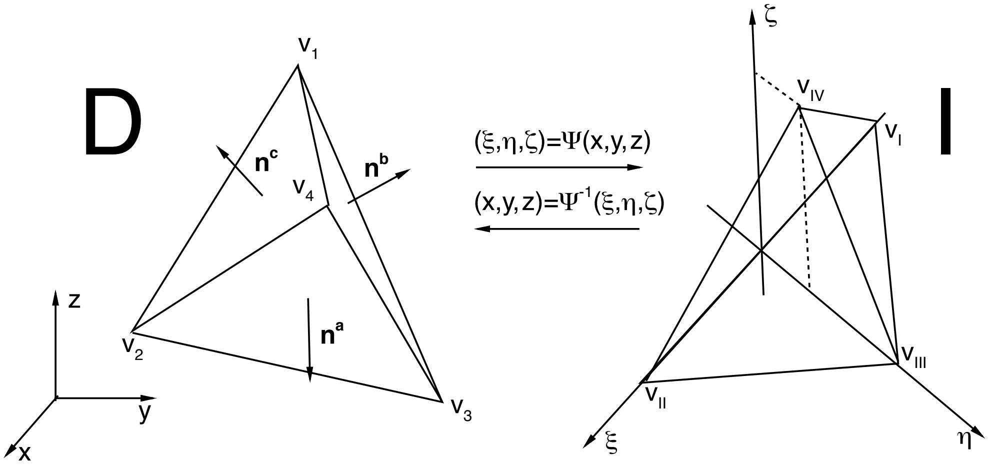

FIG. 1. *Mapping between the curvilinear tetrahedral,* D*, and the standard tetrahedral,* I*, including the numbering and notation introduced in the text.*

Let us now return to the actual construction of the smooth and invertible mapping function, $\Psi : \mathsf{D} \rightarrow \mathsf{I}$, and consider the situation illustrated in Figure 1. We are primarily interested in the important case where D is a general straight-sided tetrahedron defined through the vertices, and we recall that any $\boldsymbol{x} \in \mathsf{D}$ can be expressed as a convex combination of the vertices

$$\boldsymbol{x} = \sum_{i=1}^{4} \boldsymbol{v}_i b_i(\boldsymbol{\xi}), \quad \sum_{i=1}^{4} b_i(\boldsymbol{\xi}) = 1, \tag{2}$$

where $0 \leq b_i(\boldsymbol{\xi}) \leq 1$ for convexity. Within I, the barycentric coordinates, $b_i(\boldsymbol{\xi})$, are recovered as

$$b_1(\boldsymbol{\xi}) = \frac{1-\xi}{2} - \frac{\sqrt{2}\eta + \zeta}{\sqrt{24}}, \quad b_2(\boldsymbol{\xi}) = \frac{1+\xi}{2} - \frac{\sqrt{2}\eta + \zeta}{\sqrt{24}}, \tag{3}$$

$$b_3(\boldsymbol{\xi}) = \frac{\eta - \sqrt{3}\zeta}{\sqrt{24}}, \quad b_4(\boldsymbol{\xi}) = \sqrt{\frac{3}{8}}\zeta.$$

A close inspection of (2) reveals that it is indeed exactly the mapping we are in search of. Moreover, $b_i(\boldsymbol{\xi})$ is linear in $\boldsymbol{\xi}$, hence the metric of the transformation between any two straight-faced tetrahedra is constant. It is therefore sufficient to discuss the development of schemes for some specific tetrahedron with plane faces as exemplified by I, and the results for any D follow immediately.

The metric of the transformation, $\Psi : \mathsf{D} \rightarrow \mathsf{I}$, also directly yields the outward pointing vectors normal to the faces of D on the form

$$\boldsymbol{n}^{\mathrm{a}} = \nabla\xi + \nabla\eta + \frac{1}{2}\nabla\zeta, \quad \boldsymbol{n}^{\mathrm{b}} = -\nabla\xi + \nabla\eta + \frac{1}{2}\nabla\zeta, \tag{4}$$

$$\boldsymbol{n}^{\mathrm{c}} = -\nabla\eta + \frac{1}{\sqrt{8}}\nabla\zeta, \quad \boldsymbol{n}^{\mathrm{d}} = -\nabla\zeta.$$

We shall finally define the four surfaces, enclosing D and I, respectively, as

$$\boldsymbol{x}^{\mathrm{a}} = \boldsymbol{x}(b_1(\boldsymbol{\xi}) = 0), \;\; \boldsymbol{\xi}^{\mathrm{a}} = \boldsymbol{\xi}(b_1(\boldsymbol{\xi}) = 0),$$

where we utilize that $b_1 = 0$ at face "a." In a similar fashion we define $\boldsymbol{x}^{\mathrm{b}}$ as the face with $b_2 = 0$ as for $\boldsymbol{\xi}^{\mathrm{b}}$ and likewise for the remaining faces.

In the more general case where the faces of D are curved, the Jacobian as well as the metric are functions of $\boldsymbol{\xi}$ and Ψ can be constructed using blending functions. We shall not give the details here but refer to [15] for an account of this procedure.

2.2. Multivariate Lagrange interpolation. The general curvilinear coordinates allow us to focus the attention on the problem of interpolation in the standard tetrahedron, I.

We define the space of n-degree polynomials in three variables, P^3_n, with the dimension of the approximation space being

$$\dim \mathsf{P}^3_n \equiv N^3_n = \left(\begin{array}{c} 3+n \\ 3 \end{array} \right) = \frac{(n+1)(n+2)(n+3)}{6},$$

which is the minimum dimension of the space that allows P^3_n to be complete. In other words, $\dim \mathsf{P}^3_n \equiv N^3_n$ is exactly the number of elements in the nth-order generalized Pascal triangle [11]. Let us also introduce the nodal set, $\Pi^3_n = (\boldsymbol{\xi}_0, \ldots, \boldsymbol{\xi}_N)$, where the nodal points, or collocation points, are termed $\boldsymbol{\xi}_i \in \mathsf{I}$ or, interchangeably, $\boldsymbol{x}_i \in \mathsf{D}$. Following the notation of the previous section we refer to $\boldsymbol{\xi}^{\mathrm{a}}_i$ as those nodal points in Π^3_n that are placed on face "a" in I and, likewise, to $\boldsymbol{x}^{\mathrm{a}}_i$ as the corresponding nodal points on face "a" in D. A similar notation will be used for the subsets of Π^3_n being placed entirely on the remaining faces of either D or I, and we refer to these subsets as $\pi^{\mathrm{a}}_n = (\boldsymbol{\xi}^{\mathrm{a}}_0, \ldots, \boldsymbol{\xi}^{\mathrm{a}}_{N^{\mathrm{a}}})$ and likewise for π^{b}_n, π^{c}_n, and π^{d}_n. We shall use $N = N^3_n - 1$ to simplify the notation unless clarification is deemed necessary.

We moreover assume, although strictly speaking unnecessarily, that each of the four faces have exactly

$$N^{\mathrm{a}} = N^{\mathrm{b}} = N^{\mathrm{c}} = N^{\mathrm{d}} = \frac{(n+1)(n+2)}{2}$$

nodal points and that $3(n+1)$ of these are distributed along the edges with the vertices being included. This allows for the construction of uniquely defined one- and two-dimensional polynomials along the edges and on the faces as discussed in [11] from which we may also recover such nodal sets. Alternatives can be found in [9, 12]. However, we still have a total of

$$\frac{(n-1)(n-2)(n-3)}{6}$$

nodal points to specify in the interior of the tetrahedron when the order of the polynomial exceeds 3.

In this setting the natural polynomial space for the approximation of functions defined on the tetrahedron is

$$\mathsf{P}^3_n = \mathrm{span}\{\xi^i \eta^j \zeta^k; i, j, k \geq 0; i + j + k \leq n\},$$

where n signifies the maximum polynomial order of the multivariate approximation.

Interpolation of a smooth function, $f(\boldsymbol{\xi}) : \mathsf{I} \to \mathsf{R}$, where $f[\mathsf{I}] \in C[\mathsf{I}]$ and $f \in \mathsf{H}$ is square integrable and belongs to the Hilbert space, H, can now be viewed as for a given nodal set, Π^3_n. Find the polynomial, $\mathcal{I}^3_n f \in \mathsf{P}^3_n$, such that the interpolation operator, $\mathcal{I}^3_n[\mathsf{I}] : \mathsf{H} \to \mathsf{P}^3_n$, can be defined by the property

$$\forall \boldsymbol{\xi}_i \in \mathsf{I} \ : \ \mathcal{I}^3_n f(\boldsymbol{\xi}_i) = f(\boldsymbol{\xi}_i);$$

i.e., we seek a polynomial approximation with the Lagrange interpolation property.

The actual construction of the interpolating polynomial relies on the introduction of the complete polynomial basis, $p_i(\boldsymbol{\xi}) \in \mathsf{P}^3_n$, where $\mathsf{P}^3_n = \mathrm{span}\{p_i(\boldsymbol{\xi})\}_{i=0}^N$. The interpolation property implies that

$$\forall i \in [0, \dots, N] : f(\boldsymbol{\xi}_i) = \sum_{j=0}^{N} a_j p_j(\boldsymbol{\xi}_i),$$

leaving us with the problem of recovering the expansion coefficients, a_j, as is accomplished through the solution of the linear problem

$$\begin{bmatrix} p_0(\boldsymbol{\xi}_0) & \dots & p_N(\boldsymbol{\xi}_0) \\ \vdots & & \vdots \\ p_0(\boldsymbol{\xi}_N) & \dots & p_N(\boldsymbol{\xi}_N) \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_N \end{bmatrix} = \begin{bmatrix} f(\boldsymbol{\xi}_0) \\ \vdots \\ f(\boldsymbol{\xi}_N) \end{bmatrix},$$

or in short VDM$\boldsymbol{a} = \boldsymbol{f}$. Here the matrix VDM $=$ VDM$(\boldsymbol{\xi}_0, \dots, \boldsymbol{\xi}_N)$ is recognized as the multidimensional version of the Vandermonde matrix if the polynomial basis is taken as the basis of monomials $p_i(\boldsymbol{\xi}) = \xi^i \eta^j \zeta^k$. An alternative and computationally more appropriate choice of basis is the generalized version of the polynomials introduced in [7], and more recently in [8, 6], as it dramatically improves the conditioning of the VDM-matrix [12].

Existence and uniqueness of the interpolating polynomial is ensured if and only if the Vandermonde determinant, |VDM|, is different from zero. For interpolation along the line it is well known that we can guarantee that $|\mathrm{VDM}| \neq 0$ provided only that the nodal points are distinct. Unfortunately, no such simple results are known for interpolation in I. Sufficient conditions for uniqueness of the interpolating polynomial was obtained in [16] through a geometric characterization of the admissible nodal sets. However the requirements are rather strict and by no means necessary.

In light of this theoretical shortcoming we shall subsequently assume that the nodal set, Π^3_n, is constructed to ensure that $|\mathrm{VDM}| \neq 0$. This done, we may express the unique polynomial approximation, $\mathcal{I}^3_n f(\boldsymbol{\xi}) \in \mathsf{P}^3_n$, using Lagrange polynomials, $L_i(\boldsymbol{\xi}) \in \mathsf{P}^3_n$, as

$$\mathcal{I}^3_n f(\boldsymbol{\xi}) = \sum_{i=0}^{N} f_i L_i(\boldsymbol{\xi}), \tag{5}$$

where we recall that $L_i(\boldsymbol{\xi}_j) = \delta_{ij}$, and we have introduced $f_i = f(\boldsymbol{\xi}_i)$ and the associated grid vector $\boldsymbol{f} = (f_0, \dots, f_N)^T$.

Equation (5) must hold for any smooth $f(\boldsymbol{\xi}) \in C[\mathsf{I}]$, and in particular for the polynomial basis, $p_i(\boldsymbol{\xi})$, itself. This observation allows us to recover the interpolating Lagrange polynomials by solving the dual interpolation problem

$$\begin{bmatrix} p_0(\boldsymbol{\xi}_0) & \dots & p_0(\boldsymbol{\xi}_N) \\ \vdots & & \vdots \\ p_N(\boldsymbol{\xi}_0) & \dots & p_N(\boldsymbol{\xi}_N) \end{bmatrix} \begin{bmatrix} L_0(\boldsymbol{\xi}) \\ \vdots \\ L_N(\boldsymbol{\xi}) \end{bmatrix} = \begin{bmatrix} p_0(\boldsymbol{\xi}) \\ \vdots \\ p_N(\boldsymbol{\xi}) \end{bmatrix} \tag{6}$$

through which, using Cramers rule, the expression for the interpolating Lagrange polynomials appears directly as

$$L_i(\boldsymbol{\xi}) = \frac{|\mathrm{VDM}(\boldsymbol{\xi}_0, \dots, \boldsymbol{\xi}_{i-1}, \boldsymbol{\xi}, \boldsymbol{\xi}_{i+1}, \dots, \boldsymbol{\xi}_N)|}{|\mathrm{VDM}(\boldsymbol{\xi}_0, \dots, \boldsymbol{\xi}_N)|}. \tag{7}$$

The explicit expressions for the Lagrange polynomials are known only for very specific choices of nodal sets and the Lagrange polynomials must, in general, be generated numerically, hence the concern of the conditioning of VDM discussed above.

In seeking approximate solutions to partial differential equations, we need to evaluate spatial derivatives at the nodal points, $\boldsymbol{\xi}_i$. If we assume that the function, f, is approximated using (5), approximations to spatial derivatives are obtained directly through matrix-vector products as

$$\mathcal{I}_n^3 \frac{\partial f}{\partial \xi} \simeq \frac{\partial \mathcal{I}_n^3 f}{\partial \xi} = \mathrm{D}^\xi \boldsymbol{f}, \quad \mathcal{I}_n^3 \frac{\partial f}{\partial \eta} \simeq \frac{\partial \mathcal{I}_n^3 f}{\partial \eta} = \mathrm{D}^\eta \boldsymbol{f}, \quad \mathcal{I}_n^3 \frac{\partial f}{\partial \zeta} \simeq \frac{\partial \mathcal{I}_n^3 f}{\partial \zeta} = \mathrm{D}^\zeta \boldsymbol{f},$$

where the entries of the $(N+1)$ square differentiation matrices are given as

$$\mathrm{D}^\xi_{ij} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \xi}, \ \ \mathrm{D}^\eta_{ij} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \eta}, \ \ \mathrm{D}^\zeta_{ij} = \frac{\partial L_j(\boldsymbol{\xi}_i)}{\partial \zeta}. \tag{8}$$

These can be computed directly through differentiation of the expression for the Lagrange polynomials given in (7). Indeed, if we choose $p_i(\boldsymbol{\xi})$ as the monomials in (ξ, η, ζ), differentiation of $L_j(\boldsymbol{\xi})$ at $\boldsymbol{\xi}_i$ is implemented directly through the derivative of column j in the Vandermonde matrix prior to solving. Using the more complicated polynomial basis of [7, 8, 6] involves the differentiation of Jacobi-polynomials but results in a numerically much more stable procedure.

It shall prove central for the construction of the stable schemes to recall that the Lagrange polynomials, $L_i^{\mathrm{a}} = L_i^{\mathrm{a}}(\boldsymbol{\xi}^{\mathrm{a}})$, based solely on the nodal point-set, π_n^{a}, have the property

$$\sum_{i=0}^{N^{\mathrm{a}}} f(\boldsymbol{\xi}_i^{\mathrm{a}}) L_i^{\mathrm{a}}(\boldsymbol{\xi}^{\mathrm{a}}) = \sum_{i=0}^{N} f(\boldsymbol{\xi}_i) L_i(\boldsymbol{\xi}^{\mathrm{a}}), \tag{9}$$

as a consequence of the uniqueness of the Lagrange polynomials in the tetrahedron as well as on the face, as also L_i^{a} is complete due to the previously imposed conditions on the structure of the nodal sets. An alternative way of interpreting the restrictions put on the nodal set is that we require that the Lagrange polynomials attain their maximum order along the faces and edges of the tetrahedral. In fact, provided only this latter statement is true, (9) remains valid also for an incomplete polynomial basis or even an overspecified basis. As we shall find, the construction of the stable schemes also remains valid for such more exotic cases, provided the nodal set ensures uniqueness of the interpolating polynomials. Nevertheless, we consider only the case of a complete and unique polynomial basis as this is the most frequently appearing situation. In such a situation, statements similar to that of (9) are valid also for the approximations along the three other faces and we shall refer to the corresponding Lagrange polynomials as L_i^{b}, L_i^{c}, and L_i^{d} corresponding to the nodal sets π_n^{b}, π_n^{c}, and π_n^{d}, respectively.

Let us finally define a number of different inner products. Consider the two smooth functions $f[\mathsf{D}] \in C[\mathsf{D}]$ and $g[\mathsf{D}] \in C[\mathsf{D}]$ for which $f(\boldsymbol{x}) : \mathsf{D} \to \mathsf{R}$ and $g(\boldsymbol{x})$:

$\mathsf{D} \to \mathsf{R}$. The global inner product, the associated L^2-norm, and the inner product over the surface of D are defined as

$$(f,g)_{\mathsf{D}} = \int_{\mathsf{D}} f(\boldsymbol{x})g(\boldsymbol{x})\,d\boldsymbol{x},\ \ (f,f)_{\mathsf{D}} = \|f\|^2_{\mathsf{D}},\ \ (f,g)_{\delta\mathsf{D}} = \oint_{\delta\mathsf{D}} f(\boldsymbol{x})g(\boldsymbol{x})\,d\boldsymbol{x}.$$

Along the faces of the tetrahedron, we define the face-based norm as

$$[f,g]_{\mathrm{a}} = \int_{\mathrm{a}} f(\boldsymbol{x}^{\mathrm{a}})g(\boldsymbol{x}^{\mathrm{a}})\,d\boldsymbol{x}^{\mathrm{a}},\ \ [f,f]_{\mathrm{a}} = \|f\|^2_{\mathrm{a}}.$$

A similar notation will be used for the inner products of functions defined on the three remaining faces as named in Figure 1.

3. Multivariate nodal sets through electrostatics. While we have put a number of restrictions on the nodal sets on the faces of the tetrahedron we have yet to specify the positions of the nodes on the faces as well as in the interior of the tetrahedron.

Before we continue with the actual specification of the nodal sets, let us introduce a measure that shall allow us to quantitatively judge between several alternative nodal sets. If we term the best approximating polynomial, $p^*(\boldsymbol{\xi})$, we have

$$\left|f(\boldsymbol{\xi}) - \mathcal{I}^3_n f(\boldsymbol{\xi})\right| \le (1 + \Lambda_n(\Pi_n))\left|f(\boldsymbol{\xi}) - p^*(\boldsymbol{\xi})\right|, \tag{10}$$

provided only that $f(\boldsymbol{\xi})$ is continuous. Here we have introduced the constant

$$\Lambda_n(\Pi_n) = \max_{\xi\in\mathsf{I}} \sum_{i=0}^{N} |L_i(\boldsymbol{\xi})|,$$

known as the Lebesgue constant. Hence, the interpolation is bounded by the best approximation up to the size of the Lebesgue constant. It is noteworthy that the value of $\Lambda_n(\Pi_n)$ depends solely on the specification of the nodal set. Moreover, from (10) it is clear that we shall strive to devise nodal sets with as small a Lebesgue constant as possible or, rather, as slow a growth with n as possible.

Having identified the Lebesgue constant as a convenient measure, let us return to the actual specification of the nodal sets. Utilizing the barycentric coordinates, a natural first attempt to specify the nodal sets is the equidistant nodal being defined as

$$\Pi^{\mathrm{Eq}}_n = \left\{\boldsymbol{x}_i|(b_1,b_2,b_3,b_4) = \frac{(i,j,k,l)}{n}\ ,\ \ 0 \le i,j,k,l\ ,\ \ i+j+k+l = n\right\}.$$

However, following [17], the Lebesgue constant can be bounded as

$$\Lambda_n(\Pi^{\mathrm{Eq}}_n) \le \left(\begin{array}{c} 2n-1 \\ n \end{array}\right),$$

where n represents the order of the polynomial approximation. Although this estimate is conservative it clearly illustrates that the equidistant nodal set is inappropriate for high-order interpolation, essentially reflecting the appearance of a multidimensional Runge-phenomenon.

As is well known, one way of overcoming the Runge-phenomenon for interpolation on the finite interval involves the use of some family of quadrature points associated with the classical orthogonal polynomials, e.g., Legendre or Chebyshev polynomials. It is not clear, however, how to extend this approach to the multidimensional case, as no equivalent of the Gauss quadrature is known. One could, among other things, seek inspiration in the expression of the Lagrange polynomials, (7), and choose to use the nodal sets that maximize the Vandermonde determinant as is done in [9, 12] for the triangle and in [10] for the tetrahedron. While this choice results in the Legendre–Gauss–Lobatto quadrature points in the one-dimensional case, no such general results are known for the multidimensional simplex, although the bound

$$\Lambda_n(\Pi_n^{\mathrm{VDM}}) \leq \left(\begin{array}{c} n+3 \\ n \end{array} \right)$$

has been derived in [17], where solutions for $n \leq 4$ are also given.

In [10] a different approach was considered to derive nodal sets with a slowly growing Lebesgue constant. The ultimate approach would seem to involve seeking through the whole state-space of nodal sets with the aim of identifying the global minimum of the Lebesgue constant, if it exists. For high-order polynomials this is not feasible and the approach taken in [10] is to assume that the Lebesgue constant behaves in a way similar to the L^2-norm of the Lagrange polynomials. The advantage of this approach lies in the ease by which the L^2-norm can be computed using high-order cubature. As found in [10] this procedure does indeed result in nodal sets, computed to degree $n = 9$, that are as good as any that were previously published, although, as we shall see shortly, they do not represent the optimal nodal sets in the sense of a minimum Lebesgue constant.

We shall rely on [11] in which the remarkable connection between problems of electrostatics and zeros of the Jacobi polynomials [18], some of which are known to provide excellent nodal sets, is extended to include the computation of nodal sets in the triangle. Inspired by the one-dimensional analogy, the triangle nodal sets are computed by interpreting the edges of the triangle as line charges, providing an exterior force-field within the triangle in which we seek steady state solutions to the N-body problem of a number of freely moving charges. If only the electrostatic potential was properly defined it was shown that the steady state solution to the two-dimensional electrostatic problem provides nodal sets with the corresponding Lebesgue constant being as good and, in many cases, better than all previously known sets. Moreover, the nodal sets were designed specifically such that the nodal distribution along the edges takes the form of either the Legendre– or the Chebyshev–Gauss–Lobatto quadrature points. These specific choices were made to facilitate the use of the triangular elements in a hybrid element setting, where the efficient interfacing with elements employing a tensor-product-based approximation becomes crucial.

To extend the electrostatic procedure to the tetrahedron we begin by assuming that we may use the nodal sets computed in [11] on the four faces of the tetrahedron. These nodal sets do indeed fulfill all of the requirements put forward in section 2.2; e.g., they include the vertices and allow for completeness of the polynomial basis on the faces alone. Hence, we are left with the problem of specifying the nodal sets in the interior of the tetrahedron. To address that problem, we turn to the electrostatic interpretation and assume that a number, N_p, of unit mass charges, ρ_p, are allowed to move freely inside the tetrahedron, I, mutually interacting according to the potential

$$\phi(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j) = \frac{\rho_p^2}{|\boldsymbol{\xi}_i - \boldsymbol{\xi}_j|^2}. \tag{11}$$

These freely moving charges are held trapped in an exterior potential field. In the spirit of [11] we choose to use

$$\phi^{\mathrm{Ext}}(\boldsymbol{\xi}_i) = \sum_{j=1}^{4} \int_{\mathrm{Face}\ j} \frac{\rho_f \rho_p}{|\boldsymbol{\xi} - \boldsymbol{\xi}_i|^2}\, d\boldsymbol{\xi}, \tag{12}$$

where ρ_f represents the continuous charge density of the face which we shall later use to optimize the quality of the nodal sets. The expression for the exterior potential field can be expressed on closed, albeit quite complex, form.

While the general electrostatic analogy is the key inspiration, there is certainly a great deal of freedom in choosing the potential laws. Supported by the one-dimensional analysis [18], the two-dimensional computational results [11], and numerous three-dimensional studies, it is clear that to arrive at good nodal sets for interpolation, the exterior potential should be of a logarithmic character. This has guided us in the formulation of (11) as those results in (12) having a significant logarithmic component. It remains unknown, however, whether this is the best generalization of the one-dimensional electrostatic analogy. Here, as in [11], we choose to compose the exterior field from that of finite line/face segments. However, preliminary results indicate that by constructing the exterior fields as the sum over infinitely large lines/faces, in which case the exterior potential simply depends logarithmicly on the normal distance to the particle at $\boldsymbol{\xi}_i$, does one obtain nodal sets at least as good as those reported here. The advantage of this approach is its simplicity and hence its potential for analytic investigation. We hope to report on such developments in the future.

The actual computation of the nodal sets now involves seeking steady state solutions to an N-body problem evolving according to Newton's second law as

$$\ddot{\boldsymbol{\xi}}_i = -\left(\nabla\phi^{\mathrm{Ext}}(\boldsymbol{\xi}_i) + \sum_{\substack{j=1 \\ j\neq i}}^{N_p} \nabla\phi(\boldsymbol{\xi}_i, \boldsymbol{\xi}_j) \right) - \varepsilon\dot{\boldsymbol{\xi}}_i,$$

where N_p signifies the number of freely moving charges and the last term, $\dot{\boldsymbol{\xi}}_i$, represents a small friction added to make the problem slightly dissipative. This is done to ensure that a steady state be found and, due to the enclosed computational domain, $\dot{\boldsymbol{\xi}}_i$ must vanish in this state, hence recovering the solution to the original problem. The equation is advanced in time using a 7(6) embedded Nyström–Runge–Kutta scheme with error control [19]. In all cases the charges are assumed to be unit charges.

The most important remaining issue is the specification of the initial conditions. As we are interested in solutions suitable for polynomial interpolation, we restrict the attention to steady state solutions possessing a high degree of symmetry. Furthermore, in [11] we found, by considering a number of different symmetry patterns, that a specific family appears as the most favorable both in terms of approximation and also as the energy minimizing configuration in accordance with the electrostatic approach. This specific family was characterized by the interior distribution of the nodal set for an order n approximation being very similar in terms of symmetries to the full nodal distribution for the $n-3$ solution. This is very fortunate as the latter are easily computed and hence provide a very good set of initial conditions for the computation at the higher n. In the tetrahedron we utilize the same approach to construct the initial conditions, keeping in mind that the initial conditions for the nth-order problem use the solution for order $n-4$.

Table 1

Lebesgue constants for various choices of nodal sets in the tetrahedron. Listed are the Lebesgue constants, Λ_n, for the nodal set computed using the electrostatic approach. For comparison, we also give the Lebesgue constants, Λ_n^{Opt}, for the nodal sets presented in [10]*, and the Lebesgue constants, Λ_n^{Eq}, for the equidistant nodal set, Π_n^{Eq}.*

n	N_n^3	Λ_n	Λ_n^{Opt}	Λ_n^{Eq}
1	4	1.0000	1.0000	1.0000
2	10	2.0000	2.0000	2.0000
3	20	2.9328	2.9339	3.0000
4	35	4.0774	4.1120	4.8801
5	56	5.3470	5.6158	8.0937
6	84	7.3391	7.3632	13.659
7	120	9.7588	9.3659	23.379
8	165	13.626	12.311	40.546
9	220	18.901	15.659	71.152
10	286	27.190	-	-

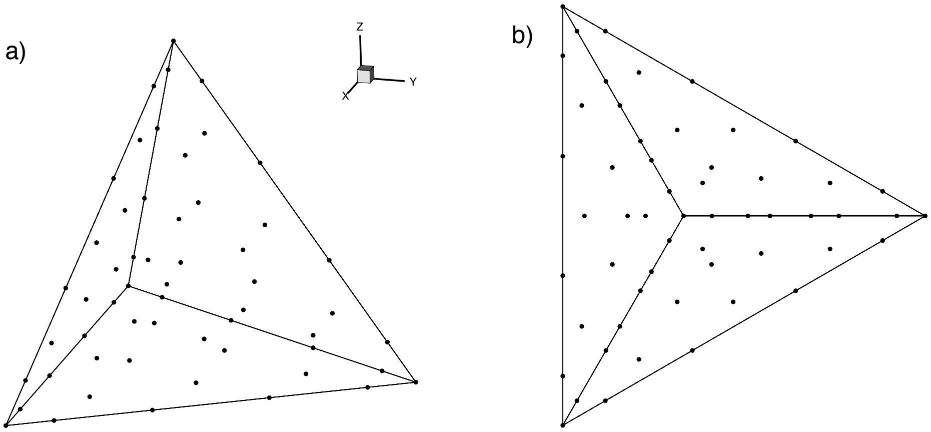

Fig. 2. *Example of nodal set for an order* 5 *interpolation as computed using the electrostatic analogy. In* (a) *we show a three-dimensional view of the nodes within the tetrahedron, while* (b) *gives a top view hence emphasizing the high degree of symmetry associated with the nodal sets.*

Following the approach outlined in the above, and utilizing the nodal distributions computed in [11] along the four faces and Legendre–Gauss–Lobatto nodes along the edges, we have computed nodal sets within the tetrahedron. The charge density, ρ_f, of the faces has been used to optimize the nodal sets in terms of the Lebesgue constants which are given in Table 1 for polynomial order up to 10. Numerical evidence suggests that $\rho_f \sim 0.75n$ results in very good nodal sets. In Figure 2, we illustrate the computed nodal set for $n = 5$ corresponding to 56 nodes. In the appendix we give the barycentric coordinates for the computed nodal sets for $n \in [2, 10]$.

From Table 1 we see that the nodal sets computed using the generalized electrostatic analogy results in interpolations that are at least as good as those based on the nodal sets in [10], obtained by minimizing the L^2-norm of the Lagrange polynomials rather than the actual L^1-based Lebesgue constant, and significantly better than an interpolation based on the equidistant nodes.

We emphasize that the nodal sets in the tetrahedron contain a significant degree of symmetry (see the appendix) as do those presented in [9, 11] for the 2-simplex. As we shall see in section 5, this is important not only for the quality of the interpolation but also for the construction of efficient differentiation schemes which actively exploit the delicate structure of the associated differentiation matrices; see (8).

4. Stable spectral methods on a tetrahedron.

The identification of multidimensional nodal sets suitable for the construction of high-order interpolants in the tetrahedron provides the first step in the development of stable high-order methods for problems on such domains. However, approximation of functions is very different from solving partial differential equations in that the latter requires attention to the way in which the equation is satisfied and the introduction of boundary conditions. It is well known that imposing boundary conditions is one of the many difficulties associated with traditional high-order/spectral methods, and we shall focus our attention on this particular concern.

4.1. A stable scheme for the scalar problem.

Let us begin by considering the three-dimensional scalar conservation law

$$\frac{\partial u}{\partial t} + \nabla \cdot \boldsymbol{F} = 0, \tag{13}$$

where we have $u(\boldsymbol{x}, t) : \mathsf{D} \times \mathsf{R}_+ \to \mathsf{R}$ and $\boldsymbol{F} = \boldsymbol{F}(u) = (F_x, F_y, F_z)$ in general. In what follows, we assume that $u(\boldsymbol{x}, t) \in C[\mathsf{D}]$ at all times and restrict the attention to the case of constant coefficients $\boldsymbol{F} = [a_x u, a_y u, a_z u]^T$, with $\boldsymbol{V} = (a_x, a_y, a_z)$ being the constant advective velocity.

The boundary operator is given in the form [2]

$$\forall \boldsymbol{x} \in \delta\mathsf{D} \ : \quad \alpha(\boldsymbol{x}) u(\boldsymbol{x}, t) = 0,$$

with

$$\forall \boldsymbol{x} \in \delta\mathsf{D} \ : \ \alpha(\boldsymbol{x}) = \begin{cases} |\boldsymbol{V} \cdot \boldsymbol{n}| & \text{if } \boldsymbol{V} \cdot \boldsymbol{n} \leq 0, \\ 0 & \text{if } \boldsymbol{V} \cdot \boldsymbol{n} > 0, \end{cases} \tag{14}$$

where $\boldsymbol{n} = |\boldsymbol{n}|\hat{\boldsymbol{n}}$ is the outward pointing normal vector associated with every point of the boundary. In other words, boundary conditions are specified at all point of inflow on the surface of the tetrahedron.

In our quest to solve (13) let us now seek solutions of the form

$$\mathcal{I}_n^3 u(\boldsymbol{x}, t) = \sum_{i=0}^{N} u_i(t) L_i(\boldsymbol{x}), \tag{15}$$

where $u_i(t) = u(\boldsymbol{x}_i, t)$ represents the grid-function at the nodal points, $\boldsymbol{x}_i = \boldsymbol{x}(\boldsymbol{\xi}_i)$. The grid-vector is named as $\boldsymbol{u} = (u_0, \ldots, u_N)^T$. The interpolating Lagrange polynomials, $L_i(\boldsymbol{x}(\boldsymbol{\xi}))$, is based on the nodal set, Π_n^3, with $\boldsymbol{\xi}_i \in \mathsf{I}$ and, as discussed previously, we assume that Π_n^3 ensures existence and uniqueness of $L_i(\boldsymbol{x})$.

To arrive at a stable scheme, we choose to enforce the boundary conditions in a weak form by requiring that the equation be satisfied in a Galerkin-like way as

$$\forall j \in [0, \ldots, N] : \left(\frac{\partial u}{\partial t} + \nabla \cdot \boldsymbol{F}(u), L_j(\boldsymbol{x}) \right)_{\mathsf{D}} = -\mathrm{T}_{jj} \mathrm{A}_{jj} (\boldsymbol{u}_j - g(\boldsymbol{x}_j, t)).$$

We have introduced the two diagonal $(N+1)^2$-matrices, T and A, which are responsible for enforcing the boundary conditions and have the entries

$$\mathrm{T}_{ii} = \begin{cases} \mathrm{T}_{ii} & \boldsymbol{x}_i \in \delta\mathsf{D}, \\ 0 & \boldsymbol{x}_i \in \mathsf{D}/\delta\mathsf{D}, \end{cases} \qquad \mathrm{A}_{ii} = \begin{cases} \alpha(\boldsymbol{x}_i) & \boldsymbol{x}_i \in \delta\mathsf{D}, \\ 0 & \boldsymbol{x}_i \in \mathsf{D}/\delta\mathsf{D}; \end{cases}$$

i.e., while the nonzero entries of T will be specified later, the entries of A are simply given such as to enforce the boundary operator; see (14).

Introducing the mass-matrix, M, and the stiffness-matrix, S, as

$$\mathrm{M}_{ij} = (L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\mathsf{D}}\,, \quad \mathrm{S}_{ij} = (\boldsymbol{V}\cdot\nabla L_i(\boldsymbol{x}), L_j(\boldsymbol{x}))_{\mathsf{D}}\,, \tag{16}$$

we recover the short form of the scheme

$$\mathrm{M}\dot{\boldsymbol{u}} + \mathrm{S}\boldsymbol{u} = -\mathrm{TA}(\boldsymbol{u} - g(\boldsymbol{x}, t)). \tag{17}$$

Let us now analyze the properties of the operators with the aim of specifying the entries of T_{ii} to ensure L^2-stability of the scheme.

We first of all note that the mass-matrix, M, is symmetric and positive definite, and consequently its inverse, M^{-1}, exists. Moreover, the action of M is given exactly as

$$\boldsymbol{u}^T\mathrm{M}\boldsymbol{u} = \int_{\mathsf{D}} u^2(\boldsymbol{x})\,d\boldsymbol{x} = \|u\|^2_{\mathsf{D}};$$

i.e., the vector space norm, $\boldsymbol{u}^T\mathrm{M}\boldsymbol{u}$, plays the role of the usual unweighted L_2-norm.

The properties of the stiffness matrix, S, are a little more subtle, but understanding these is at the heart of the construction. Let us first of all introduce the four face-based subsets of the grid vector $\boldsymbol{u}$ as

$$\boldsymbol{u}^{(\mathrm{a,b,c,d})} = \{\boldsymbol{u}(\boldsymbol{x}_i)|\boldsymbol{x}_i \in \boldsymbol{x}^{(\mathrm{a,b,c,d})}\};$$

i.e., they consist of the nodal values of $u(\boldsymbol{x}, t)$ at the four faces of D. Moreover, let us define the four mass-matrices based on the nodal sets on the faces only as

$$\mathrm{M}_{ij}^{(\mathrm{a,b,c,d})} = \left[L_i^{(\mathrm{a,b,c,d})}(\boldsymbol{x}), L_j^{(\mathrm{a,b,c,d})}(\boldsymbol{x})\right]_{(\mathrm{a,b,c,d})},$$

where the face-based polynomials, $L_i^{(\mathrm{a,b,c,d})}$, are defined in (9). To proceed, we need to recall the assumption of uniqueness of the polynomial basis combined with the restriction that the polynomials take their maximum order along the faces of the tetrahedron as discussed in section 2.2. The important consequence of this restriction is the validity of

$$\begin{aligned} \mathrm{S}_{ij} &= (\boldsymbol{V}\cdot\nabla L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\mathsf{D}} \\ &= (\boldsymbol{V}\cdot\boldsymbol{n}L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\delta\mathsf{D}} - (L_j(\boldsymbol{x}), \boldsymbol{V}\cdot\nabla L_i(\boldsymbol{x}))_{\mathsf{D}}\,, \end{aligned} \tag{18}$$

provided $\boldsymbol{V}$, as assumed, is constant. In other words

$$\mathrm{S}_{ij} + \mathrm{S}_{ji} = (\boldsymbol{V}\cdot\boldsymbol{n}L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\delta\mathsf{D}}\,;$$

i.e., the stiffness-matrix is almost skew-symmetric. It is in the telescoping to the boundary of D that the uniqueness of $L_i(\boldsymbol{x})$ is required. For the present case of the tetrahedron, as illustrated in Figure 1, we hence have

$$\begin{aligned} \boldsymbol{u}^T\mathrm{S}\boldsymbol{u} = \frac{1}{2}\big[&\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{a}}(\boldsymbol{u}^{\mathrm{a}})^T\mathrm{M}^{\mathrm{a}}\boldsymbol{u}^{\mathrm{a}} + \boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{b}}(\boldsymbol{u}^{\mathrm{b}})^T\mathrm{M}^{\mathrm{b}}\boldsymbol{u}^{\mathrm{b}} \\ &+ \boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{c}}(\boldsymbol{u}^{\mathrm{c}})^T\mathrm{M}^{\mathrm{c}}\boldsymbol{u}^{\mathrm{c}} + \boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{d}}(\boldsymbol{u}^{\mathrm{d}})^T\mathrm{M}^{\mathrm{d}}\boldsymbol{u}^{\mathrm{d}}\big]\,, \end{aligned} \tag{19}$$

where the normals, $\boldsymbol{n}$, are given in (4).

We are now in a position to state the following basic result on the stability of the scheme, (17), for D being a general straight-sided tetrahedron.

THEOREM 4.1. *Assume that $u(\boldsymbol{x}, t)$ represents a solution to the scalar conservation law defined on the tetrahedral domain, D, and that there exists a diffeomorphism, $\Psi : \mathsf{D} \to \mathsf{I}$, that maps D onto the standard tetrahedron, I. Assume also that a nodal set, Π_n^3, for interpolation is given in I and that this nodal set ensures unique Lagrange interpolation.*

Then (17) *can always be made stable by properly choosing the elements of the matrix,* T *, and the growth of the solution is bounded as*

$$\frac{1}{2}\frac{d}{dt}\boldsymbol{u}^T \mathrm{M}\boldsymbol{u} \leq 0,$$

where the vector, $\boldsymbol{u}$, has the entries $u_i(t) = u(\boldsymbol{x}_i, t)$.

Proof. Multiplying (17) from the left with $\boldsymbol{u}^T$ and utilizing (19) we obtain, assuming homogeneous boundary conditions, that

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\boldsymbol{u}^T \mathrm{M}\boldsymbol{u} =& -\frac{1}{2}\left[\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{a}}(\boldsymbol{u}^{\mathrm{a}})^T\mathrm{M}^{\mathrm{a}}\boldsymbol{u}^{\mathrm{a}} + \boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{b}}(\boldsymbol{u}^{\mathrm{b}})^T\mathrm{M}^{\mathrm{b}}\boldsymbol{u}^{\mathrm{b}}\right.\\ &\left.+\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{c}}(\boldsymbol{u}^{\mathrm{c}})^T\mathrm{M}^{\mathrm{c}}\boldsymbol{u}^{\mathrm{c}} + \boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{d}}(\boldsymbol{u}^{\mathrm{d}})^T\mathrm{M}^{\mathrm{d}}\boldsymbol{u}^{\mathrm{d}}\right] - \boldsymbol{u}^T\mathrm{T}\mathrm{A}\boldsymbol{u},\end{aligned}$$

where we have used the usual notation for the face vectors of $\boldsymbol{u}$. We recall that T as well as A are diagonal matrices with nonzero elements in those rows corresponding to boundary nodes only. Hence, we may express the product $\boldsymbol{u}^T\mathrm{T}\mathrm{A}\boldsymbol{u}$ as

$$\boldsymbol{u}^T\mathrm{T}\mathrm{A}\boldsymbol{u} = (\boldsymbol{u}^{\mathrm{a}})^T\mathrm{T}^{\mathrm{a}}\mathrm{A}^{\mathrm{a}}\boldsymbol{u}^{\mathrm{a}} + (\boldsymbol{u}^{\mathrm{b}})^T\mathrm{T}^{\mathrm{b}}\mathrm{A}^{\mathrm{b}}\boldsymbol{u}^{\mathrm{b}} + (\boldsymbol{u}^{\mathrm{c}})^T\mathrm{T}^{\mathrm{c}}\mathrm{A}^{\mathrm{c}}\boldsymbol{u}^{\mathrm{c}} + (\boldsymbol{u}^{\mathrm{d}})^T\mathrm{T}^{\mathrm{d}}\mathrm{A}^{\mathrm{d}}\boldsymbol{u}^{\mathrm{d}},$$

where

$$\mathrm{T}_{ii}^{(\mathrm{a,b,c,d})} = \{\mathrm{T}_{ii}(\boldsymbol{x}_i) | \boldsymbol{x}_i \in \boldsymbol{x}^{(\mathrm{a,b,c,d})}\}$$

represents those parts of the T matrix that relate to each of the four faces. In a similar fashion, the entries of $\mathrm{A}^{(\mathrm{a,b,c,d})}$ are simply those parts of the boundary operator that relate to each of the faces. A minor detail is that the elements in either $\mathrm{A}^{(\mathrm{a,b,c,d})}$ or $\mathrm{T}^{(\mathrm{a,b,c,d})}$ related to the edges must be divided by a factor of two while those corresponding to the vertices must be divided by four to account for the nodes shared among the faces and edges. Following the interpretation of [2] this corresponds to defining normal vectors along the edges and vertices as being composed of the simple sums of the two or four normal face-vectors.

Stability is consequently established by requiring that

$$\begin{aligned}\frac{1}{2}\frac{d}{dt}\boldsymbol{u}^T \mathrm{M}\boldsymbol{u} =& -(\boldsymbol{u}^{\mathrm{a}})^T\left[\frac{\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{a}}}{2}\mathrm{M}^{\mathrm{a}} + \mathrm{T}^{\mathrm{a}}\mathrm{A}^{\mathrm{a}}\right]\boldsymbol{u}^{\mathrm{a}} - (\boldsymbol{u}^{\mathrm{b}})^T\left[\frac{\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{b}}}{2}\mathrm{M}^{\mathrm{b}} + \mathrm{T}^{\mathrm{b}}\mathrm{A}^{\mathrm{b}}\right]\boldsymbol{u}^{\mathrm{b}}\\ &-(\boldsymbol{u}^{\mathrm{c}})^T\left[\frac{\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{c}}}{2}\mathrm{M}^{\mathrm{c}} + \mathrm{T}^{\mathrm{c}}\mathrm{A}^{\mathrm{c}}\right]\boldsymbol{u}^{\mathrm{c}} - (\boldsymbol{u}^{\mathrm{d}})^T\left[\frac{\boldsymbol{V}\cdot\boldsymbol{n}^{\mathrm{d}}}{2}\mathrm{M}^{\mathrm{d}} + \mathrm{T}^{\mathrm{d}}\mathrm{A}^{\mathrm{d}}\right]\boldsymbol{u}^{\mathrm{d}}.\end{aligned}$$

We observe that all four terms in the stability condition are quadratic forms. Let us consider the condition along edge "d" for which we have $\boldsymbol{n}^{\mathrm{d}} = -\nabla\zeta$ (cf. (4)). In this case stability is achieved provided the matrix

$$\frac{\boldsymbol{V}\cdot\nabla\zeta}{2}\mathrm{M}^{\mathrm{d}} - \mathrm{T}^{\mathrm{d}}\mathrm{A}^{\mathrm{d}} = \boldsymbol{V}\cdot\nabla\zeta\left(\frac{1}{2}\mathrm{M}^{\mathrm{d}} - \alpha^{\mathrm{d}}\mathrm{T}^{\mathrm{d}}\right)$$

is negative semidefinite. Here we have introduced the definition of A such that $\alpha^{\mathrm{d}} = 1$ if the face is an inflow and zero otherwise. If we first consider the outflow case, we have $\alpha^{\mathrm{d}} = 0$ and $\boldsymbol{V} \cdot \nabla\zeta < 0$. Since M^{d} is positive definite, stability in the outflow case follows. For the inflow case, we have $\alpha^{\mathrm{d}} = 1$ and $\boldsymbol{V} \cdot \nabla\zeta \geq 0$, leaving only the requirement that

$$\frac{1}{2}\mathrm{M}^{\mathrm{d}} - \mathrm{T}^{\mathrm{d}}$$

be negative definite by properly choosing the entries of T^{d}. However, since T^{d} is diagonal and M^{d} is symmetric positive definite, this is clearly always possible. This establishes stability of the scheme along face "d." The exact same procedure can be applied to prove stability along the remaining three faces. □

It should be noted that the form of the penalty-term in (17) is nonunique, and alternatives are discussed in [14]. One should also note that the stability proof, irrespective of the actual formulation of the penalty-term, is strictly valid only in the case of a straight-sided tetrahedron due to loss of uniqueness of the polynomials for more general situations. This is, however, not a severe restriction. When using tetrahedra as building blocks for unstructured multidomain methods, the majority of the tetrahedra are straight sided. Moreover, more conventional multidomain spectral methods, based on a tensor-product formulation in a d-cube, can likewise be proven stable only in the cases where the Jacobian is constant [2]. Nevertheless, it is generally found that the schemes maintain their properties also for nonconstant Jacobians, although at present we are not aware of any proofs to substantiate such claims.

4.2. Generalizations. Let us now briefly consider extensions of the proposed approach to allow for the construction of stable schemes for problems beyond the scalar wave-equation.

We shall begin by considering the formulation of schemes for the solution of symmetric hyperbolic systems defined on a tetrahedral domain. We consider the hyperbolic system

$$\frac{\partial u}{\partial t} + \mathrm{A}_x \frac{\partial u}{\partial x} + \mathrm{A}_y \frac{\partial u}{\partial y} + \mathrm{A}_z \frac{\partial u}{\partial z} = 0,$$

defined on D, where $u(\boldsymbol{x}, t)$ now represents a vector field with k elements and A_x, A_y, and A_z are constant k-by-k matrices, which may not be symmetric but can be symmetrized simultaneously through a similarity transform. Examples of such problems include the homogeneous Maxwell's equations and the equations of acoustics in a uniform flow.

Expressing the system in general coordinates we obtain

$$\frac{\partial u}{\partial t} + \mathrm{A}_{\nabla\xi} \frac{\partial u}{\partial \xi} + \mathrm{A}_{\nabla\eta} \frac{\partial u}{\partial \eta} + \mathrm{A}_{\nabla\zeta} \frac{\partial u}{\partial \zeta} = 0,$$

where we have the transformed matrices

$$\mathrm{A}_n = n_1 \mathrm{A}_x + n_2 \mathrm{A}_y + n_3 \mathrm{A}_z,$$

with $\boldsymbol{n} = (n_1, n_2, n_3)$ being a vector. If we consider the similarity transform of A_n as $\Lambda_n = (\mathrm{V}_n)^{-1}\mathrm{A}_n\mathrm{V}_n$, the elements of the diagonal matrix, Λ_n, contains the wave-speed of the characteristic variables, $(\mathrm{V}_n)^{-1}u$, along $\boldsymbol{n}$; i.e., they take the exact same role as $\boldsymbol{V} \cdot \boldsymbol{n}$ in the scalar problem.

The construction of stable schemes for systems using the penalty method is discussed in great detail in [2], and we follow the approach developed in that work. Combining this with the results arrived at for the scalar case we propose to consider a scheme on the form

$$\mathrm{M}\dot{\boldsymbol{u}} + \mathrm{S}\boldsymbol{u} = -\mathrm{T}\mathrm{V}_n\mathrm{A}(\mathrm{V}_n)^{-1}(\boldsymbol{u} - g(\boldsymbol{x}, t)),$$

with the only difference from the scalar case being that M, S, and T now are block forms of the mass- and stiffness-matrix, respectively, due to the k variables, while T remains purely diagonal. The effect of the specific construction of the penalty-term is to enforce the boundary conditions in characteristic form. In the system case, the matrix A has k block entries of a form similar to that for the scalar problem such that for Λ_n being negative, corresponding to an inflow situation, the entry is nonzero while it is zero in case of an outflow face. The stability of this scheme follows immediately by transforming the problem into characteristic variables and establishing stability of the characteristic form. We refer to [2] for a detailed discussion of this approach within a general curvilinear setting.

Let us also briefly consider the formulation of schemes for the solution of linear advection-diffusion problems

$$\frac{\partial u}{\partial t} + \nabla \cdot \boldsymbol{F} = \varepsilon \nabla \cdot \boldsymbol{F}_\varepsilon,$$

where $\boldsymbol{F} = \boldsymbol{V}u$, while $\boldsymbol{F}_\varepsilon = \nabla u$ represents the gradient of the velocity field. Following [2] the boundary operator takes the form

$$\forall \boldsymbol{x} \in \delta\mathsf{D} \ : \quad \alpha(\boldsymbol{x})u(\boldsymbol{x}, t) + \varepsilon \boldsymbol{F}_\varepsilon \cdot \boldsymbol{n} = 0,$$

where $\alpha(\boldsymbol{x})$ is defined as in (14) and $\boldsymbol{n}$ signifies the outward pointing normal vector at $\boldsymbol{x} \in \delta\mathsf{D}$.

A stable scheme takes the form

$$\frac{\partial u}{\partial t} + \nabla \cdot \boldsymbol{F} = \varepsilon \nabla \cdot \boldsymbol{F}_\varepsilon - \mathrm{T}\left[\mathrm{A}\left(u - g(t)\right) + \varepsilon \boldsymbol{n} \cdot (\boldsymbol{F}_\varepsilon - \boldsymbol{h}(t))\right],$$

where $g(t)$ and $\boldsymbol{h}(t)$ represents the Dirichlet and Neumann-type boundary conditions, respectively. With this formulation, stability follows from the realization that

$$\begin{aligned}
\mathrm{S}_{ij} + \mathrm{S}_{ji} &= (\boldsymbol{V} \cdot \nabla L_j(\boldsymbol{x}) - \varepsilon \nabla \cdot \nabla L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\mathsf{D}} \\
&= (\boldsymbol{V} \cdot \boldsymbol{n} L_j(\boldsymbol{x}), L_i(\boldsymbol{x}))_{\delta\mathsf{D}} \\
&\quad -\varepsilon \left[(\nabla L_j(\boldsymbol{x}) \cdot \boldsymbol{n}, L_i(\boldsymbol{x}))_{\delta\mathsf{D}} + (L_j(\boldsymbol{x}), \nabla L_i(\boldsymbol{x}) \cdot \boldsymbol{n})_{\delta\mathsf{D}}\right] \\
&\quad +2\varepsilon \left(\nabla L_j(\boldsymbol{x}), \nabla L_i(\boldsymbol{x})\right)_{\mathsf{D}}.
\end{aligned}$$

As was the case for (18), the validity of the integration by parts argument hinges entirely on the assumption of uniqueness of the polynomials. However, once this form of S is realized, stability is straightforwardly established in a manner similar to the above.

5. Computational issues. The stable schemes developed in the previous section are general in nature, and, while the emphasis has been on tetrahedral domains, very general domains can be considered provided only that well-behaved nodal sets for polynomial interpolation are known in such domains. Nevertheless, to make these schemes interesting from a computational point of view we must address a number of additional issues related to computational accuracy and efficiency.

We shall again, for simplicity, restrict the attention to the linear wave equation

$$\frac{d\boldsymbol{u}}{dt} + \mathrm{M}^{-1}\mathrm{S}\boldsymbol{u} = -\mathrm{M}^{-1}\mathrm{TA}\left(\boldsymbol{u} - g(\boldsymbol{x}, t)\right)$$

solved on a tetrahedral domain using the scheme of (17).

Let us first claim that

$$\mathrm{M}^{-1}\mathrm{S} = \boldsymbol{V} \cdot \left[\nabla\xi \mathrm{D}^{\xi} + \nabla\eta \mathrm{D}^{\eta} + \nabla\zeta \mathrm{D}^{\zeta}\right], \tag{20}$$

where the differentiation matrices are given in (8). Hence, the computation is reduced to matrix-vector multiplications similar to the approach in conventional collocation methods. To recover (20) we introduce the expansion

$$\nabla L_j(\boldsymbol{x}) = \sum_{k=0}^{N} \nabla L_j(\boldsymbol{x}_k) L_k(\boldsymbol{x}),$$

which is valid under the assumption of uniqueness of the Lagrange polynomials, and continue by considering

$$\begin{aligned}\mathrm{S}_{ij} = \left(\boldsymbol{V} \cdot \nabla L_j(\boldsymbol{x}), L_i(\boldsymbol{x})\right)_{\mathsf{D}} &= \int_{\mathsf{D}} L_i(\boldsymbol{x}) \sum_{k=0}^{N} \boldsymbol{V} \cdot \nabla L_j(\boldsymbol{x}_k) L_k(\boldsymbol{x})\, d\boldsymbol{x} \\ = \sum_{k=0}^{N} \left(L_i(\boldsymbol{x}), L_k(\boldsymbol{x})\right)_{\mathsf{D}} \boldsymbol{V} \cdot \nabla L_j(\boldsymbol{x}_k) &= \sum_{k=0}^{N} \mathrm{M}_{ik} \boldsymbol{V} \cdot \nabla L_j(\boldsymbol{x}_k),\end{aligned}$$

from which (20) immediately follows by introducing the differentiation matrices in (8). The same approach can be exploited to identify high-order differentiation matrices when considering, e.g., an advection-diffusion problem.

This now leaves us with issues related to the formulation

$$\frac{d\boldsymbol{u}}{dt} + \boldsymbol{V} \cdot \left[\nabla\xi \mathrm{D}^{\xi} + \nabla\eta \mathrm{D}^{\eta} + \nabla\zeta \mathrm{D}^{\zeta}\right] \boldsymbol{u} = -\mathrm{M}^{-1}\mathrm{TA}\left(\boldsymbol{u} - g(\boldsymbol{x}, t)\right), \tag{21}$$

for which a number of questions begs the attention. However, before we attend to these matters, let us first assess the accuracy of the scheme through a simple computation of the derivative of the function

$$f(\boldsymbol{x}) = \exp\left(-(x + y + z)2^{-\alpha}\right), \quad \boldsymbol{x} \in \mathsf{I}, \tag{22}$$

using the differentiation matrices of (8) based on the nodal sets given in the appendix. The parameter, α, controls the smoothness of the test-function and models, at least to some extent, the effect of refining the grid by introducing more tetrahedra in the approximation.

In Figure 3 we plot the maximum pointwise error of the x-derivative for different values of α and n. The computation of the interpolating Lagrange polynomials and the differentiation matrices are all done in extended precision (32 digits) to avoid effects of ill-conditioning for increasing n. Alternatively, the more complicated basis of [7, 8, 6] could be exploited to avoid the need for extended precision in the preprocessing stage. The results, however, would be identical for all practical purposes

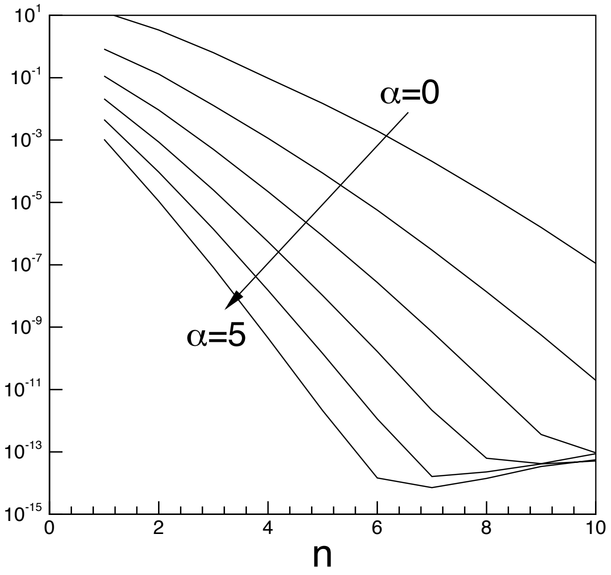

FIG. 3. *Maximum error in the x-derivative as computed using the differentiation matrices of order n defined on the tetrahedron. Increasing α implies differentiation of an increasingly smooth function.*

and, as shown in Figure 3, confirm the expected spectral accuracy on the unstructured grids for increasing order, n, of the polynomial.

Having confirmed that the evaluation of derivatives can be accomplished with the expected accuracy, we can now turn the attention toward issues related to computational cost and discrete stability of the proposed scheme. Understanding these issues is central to the application of the proposed framework, and we shall devote the remainder of this section to the development of efficient factorization methods for the computation of derivatives and an analysis of the behavior of the eigenvalue spectra of the advective operators. The latter plays a central role in regards to temporal integration using explicit schemes.

5.1. Efficient computation of derivatives. Returning to the scheme given in (21), it is clear that the most time- as well as memory-consuming element lies in the computation of the derivatives through matrix-vector products and the need to store these large matrices. Compared to schemes utilizing tensor-based approximations, resulting in an $\mathcal{O}(n^4)$ complexity, the computation of derivatives through the matrix-vector operations yields an $\mathcal{O}(n^6)$ complexity which clearly is prohibitive for n being large.

From a practical point of view, however, it is less interesting to discuss the behavior of the algorithm for asymptotically large values n. Nodal sets are, on one hand, only known for moderate values of n, i.e., presently, only to the order of 16 in the triangle and to the order of 10 in the tetrahedron. Moreover, the use of high-order methods implies that there is a trade-off in computational efficiency between the order of the scheme and the size of the element to achieve a given formal accuracy in an optimal, i.e., minimal, time. While the work associated with increasing the order of the scheme grows nonlinearly in n, the work grows only linearly when increasing the number of elements. Not only can such ideas be brought out in a

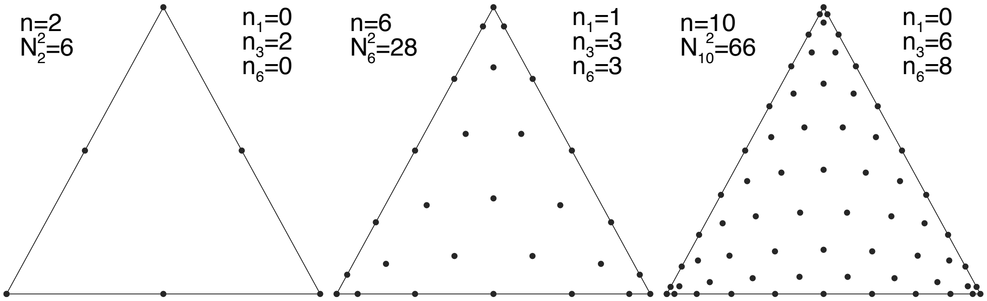

FIG. 4. *Example of nodal sets for polynomial interpolation of order n within the triangle. The symmetries of points is emphasized by $n_1 + 3n_3 + 6n_6 = N_n^2$, where n_1, n_3, and n_6 signify the number of one, three, and six symmetries, respectively.*

more quantitative manner (see [20, 2] and references therein) but it is noteworthy that it remains true for practical large scale applications. Hence, for the vast majority of problems it will be computationally inefficient to increase the order of the scheme beyond 10–12 for schemes employing triangles and even lower in the case of the tetrahedron.

It is consequently worthwhile to seek and identify properties that can be exploited to make the unstructured grid methods competitive with tensor-product based methods for low to moderate order of the approximation. As we shall see shortly, actively exploiting a few central observations can lead to a very significant decrease in computer-time as compared to simple matrix-vector multiplication and render the computation of derivatives on the d-simplex faster than on the d-cube for all practical orders of the polynomial approximation.

To understand the basic idea of the fast scheme for multiplication with the differentiation matrices, let us briefly recall the properties underlying a few known fast matrix-vector algorithms. In the case of the celebrated FFT, requiring an $N = 2^p$ equidistant grid, it is the existence of exactly p folding symmetries that results in the binary-tree structure of the $\mathcal{O}(N \log N)$-scheme rather than the straightforward $\mathcal{O}(N^2)$ matrix-vector algorithm. Moreover, in [21] it was observed that the differentiation matrices resulting from approximations based on the symmetric Jacobi polynomials all share an even-odd symmetry, hence allowing for the formulation of an $\mathcal{O}(N^2/2)$ algorithm for the computation of derivatives. The appearance of the even-odd symmetry can be attributed to the single folding symmetry of the symmetric Jacobi–Gauss–Lobatto grid points. Hence, it is apparent that symmetries in the nodal sets implies a certain amount of structure in the differentiation matrices and, by exploiting this structure, one can compute the action of the differentiation matrix at a reduced computational cost.

Let us first exploit the applicability of these ideas to the computation of derivatives on triangular domains. In Figure 4 we illustrate the distribution of the nodes as obtained from [11]. We first of all note that the nodal sets possess a significant degree of symmetry as given through the numbers n_3 and n_6 signifying 3- and 6-symmetries, respectively, within the nodal sets. The question remains how to exploit these symmetries.

Let us assume that we wish to compute

$$\frac{du}{dx} = \left(\xi_x \mathrm{D}^\xi + \eta_x \mathrm{D}^\eta\right) \boldsymbol{u}.$$

Inspecting the nodes in Figure 4 it is clear that the symmetries are of a rotational character. Hence, it seems natural to consider the computation of the derivatives of the form

$$\frac{du}{dx} = \left(r_x \mathrm{D}^r + \frac{\theta_x}{r} \mathrm{D}^\theta \right) \boldsymbol{u},$$

where the polar coordinate system (r, θ) has its origin at the center of the triangle and we can form the new differentiation matrices directly as

$$\mathrm{D}^r = \xi_r \mathrm{D}^\xi + \eta_r \mathrm{D}^\eta, \quad \mathrm{D}^\theta = \xi_\theta \mathrm{D}^\xi + \eta_\theta \mathrm{D}^\eta, \tag{23}$$

where the metric of the transformation $(\xi, \eta) \to (r, \theta)$ is nothing else than the usual polar transformation. However, to realize the structure in these new differentiation matrices we need to carefully order the nodes, a problem special to the multidimensional algorithm. It seems, at least as a first step, only natural to order the nodes in the groups of three nodes that have a constant r and are equidistant in θ. Upon doing so, one observes that D^r as well as D^θ take on a block form consisting of $(N/3)^2$ 3-by-3 circulant blocks as a consequence of the circular symmetries inherent in the nodal sets. The realization of the block-circulant form has a number of consequences. First of all, we need only store one column of each block, hence decreasing the memory requirement to 1/3 of the original approach. Moreover, since multiplication by a circulant matrix is nothing more that the discrete equivalent of convolution it is clear that the right evaluation of the 3-by-3 matrix-vector products is in Fourier space where the circulant matrix is diagonal. In other words, the differentiation matrices on polar form can be factorized as $(n = 2)$

$$\mathrm{D}^{r,\theta} = \begin{bmatrix} \mathrm{F}^T & 0 \\ 0 & \mathrm{F}^T \end{bmatrix} \begin{bmatrix} \tilde{\mathrm{D}}^{r,\theta}_{11} & \tilde{\mathrm{D}}^{r,\theta}_{12} \\ \tilde{\mathrm{D}}^{r,\theta}_{21} & \tilde{\mathrm{D}}^{r,\theta}_{22} \end{bmatrix} \begin{bmatrix} \mathrm{F} & 0 \\ 0 & \mathrm{F} \end{bmatrix},$$

where F represent the 3-by-3 Fourier matrix and $\mathrm{D}^{r,\theta}_{ij}$ are all 3-by-3 diagonal, complex matrices. This latter complication can be avoided by assuming that the entries of $\mathrm{D}^{r,\theta}$ are all real.

Hence, provided only that the nodal sets are appropriately ordered, the resulting block circulant forms of D^r and D^θ can be precomputed and stored in a compact form in Fourier-space. The computation of an x-derivatives then involves $N/3$ length 3 Fourier transforms of $\boldsymbol{u}$, followed by multiplication with the two Fourier-space block-diagonal forms of D^r and D^θ, and finally 2 $N/3$ length 3 Fourier transforms to recover the r and θ derivatives of $\boldsymbol{u}$ at the grid points. The x-derivative is recovered upon multiplication with the metric.

It is easy to see that the factorization of the differentiation matrices results in a dramatic reduction in the memory usage, requiring only 1/3 of the memory of the straightforward approach. Moreover, a 5/9-reduction in execution time can be expected as a result of a reduced number of arithmetic operations. These simple estimates are confirmed in Figure 5(a), where we show the measured speed-up compared to an optimized matrix-vector computation. Indeed, for increasing values of n we find more than a factor of 2 speed-up and closer to a factor of 3 for n approaching 16. This latter improvement cannot be attributed solely to a lower number of operations but rather to the lower memory requirement and, hence, a more efficient usage of the cache—a property that becomes increasing important for contemporary machines with deep memory hierarchies. We note that for $n = 3, 6, 9, 12, 15$ we have a single

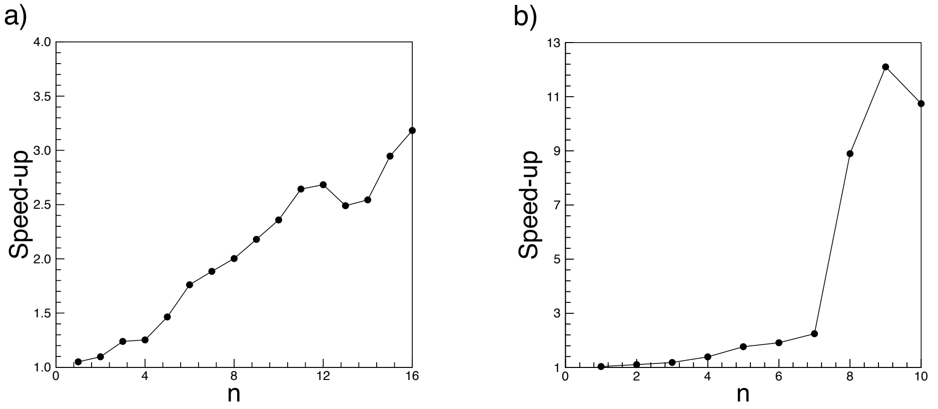

FIG. 5. *Speed-up achieved by exploiting the* 3*-symmetry of the nodal sets. The execution times for the applications of the differentiation matrix is compared to straightforward matrix-multiplication. In* (a) *we show the results for the nodal sets within a triangle, while* (b) *illustrates the corresponding results for the tetrahedron. For large values of* n *the more efficient memory utilization is seen to play a very significant role.*

node at $r = 0$, the n_1-symmetry in Figure 4, which has to be dealt with separately as a vector-vector product. This is a situation very similar to that in [21] for the even-odd decomposition with an even order approximation for which the $x = 0$, i.e., the point of symmetry, is contained in the nodal set. However, as observed in Figure 5, this special procedure has no effect on the performance of the scheme for increasing n.

It should be noted that while we have exploited the 3-symmetry of the nodal set, a similar technique could be implemented to take advantage of the 6-symmetry inherent in subsets of the nodal sets. This would result in even further reductions in the operations and memory requirements, in particular, for increasing n as the number of 6-symmetries increases for increasing n [11].

Let us now consider the question of how to generalize this approach to the fast computation of derivatives

$$\frac{du}{dx} = \left(\xi_x \mathrm{D}^\xi + \eta_x \mathrm{D}^\eta + \zeta_x \mathrm{D}^\zeta\right) \boldsymbol{u}$$

within the tetrahedron. In the spirit of the approach developed for the triangle, we shall look for symmetries in the nodal sets and rewrite the process of differentiation accordingly. As discussed in the appendix the nodal sets within the tetrahedron possess a number of symmetries, the nature of which suggests considering the problem of differentiation either on cylindrical or spherical form. While the latter may seem the most natural generalization of the polar formulation within triangle, the cylindrical formulation

$$\frac{du}{dx} = \left(r_x \mathrm{D}^r + \frac{\theta_x}{r} \mathrm{D}^\theta + z_x \mathrm{D}^z\right) \boldsymbol{u}$$

has a number of advantages. Here the center of the (r, θ, z)-system is at the center of the tetrahedron with the z-axis passing through $\boldsymbol{v}_{\mathrm{IV}}$ and being aligned with the ζ-axis (see Figure 1 for notation). The new differentiation matrices can be recovered in a way similar to (23), where the metric of the transformation $(\xi, \eta, \zeta) \rightarrow (r, \theta, z)$ is the usual cylindrical transformation.

Returning to Figure 2, we realize that for a constant value of z, the problem in the plane is equivalent to that of the triangle. Moreover, for constant z, entries in D^z form circulant blocks in a fashion similar to that for constant r. Hence, the immediate advantage of using the cylindrical formulation is the knowledge that all matrices recover the block-circulant form from the triangle, provided only that the nodal sets are sorted in groups of three for constant (r, z) and equidistant in θ. At first it may seem tempting to exploit the 4-symmetry of the nodal sets. These, however, are associated with the vertices and center of the faces, one of which requires special treatment as $r = 0$ along the z axis. A similar problem occurs in the spherical formulation for $\theta = 0$, hence making the 3-symmetry the natural lowest order symmetry to exploit. Similar to the scheme on the triangle, all nodes along the z-axis must be dealt with separately through a few vector-vector products.

Nevertheless, the basic algorithm for the 3-symmetry in planes of z remains the same as that developed for the triangle, resulting in a similar theoretical reduction in memory usage and execution time. The resulting block circulant forms of D^r, D^θ, and D^z can be precomputed and stored in a compact form in Fourier-space. The computation of derivatives then consists of $N/3$ length 3 Fourier transforms of $\boldsymbol{u}$, followed by multiplication with the three Fourier-space block-diagonal forms of the differentiation matrices and finally 3 $N/3$ length 3 Fourier transforms to recover the r, θ, and z derivatives of $\boldsymbol{u}$ at the grid points.

In Figure 5(b) we show the average speed-up achieved by applying the fast technique for the computation of 1000 derivatives on the tetrahedron as compared to the time for a similar computation using a matrix-multiplication technique. As for the triangle, we find a speed-up between 2 and 3 in accordance with the reduction of operations. However, the effect of the reduced memory requirement is very dramatic for $n > 8$ for the tetrahedron, yielding a close to 10-fold speed-up, which can only be attributed to the difference between in-cache and out-of-cache memory references.

While the algorithm on the triangle has some room for improvement through the use of the 6-symmetry, there is very significant potential for further improvements for the tetrahedron by exploiting not only the 6-symmetry but also the 12-symmetry which plays an increasingly important role for increasing n as shown in the appendix. The basic idea, however, remains the same as the one utilized here and we have left it for the future to further improve on the scheme through the active use of these symmetries.

Let us finally return to the issue discussed in the beginning of this section and compare the performance of the efficient factorization methods with the computation of derivatives on domains that allow for tensor-product-based approximations. Recall that the question is not what happens for asymptotically large n but rather how the two different approaches compare for low to moderate values of n. In Figure 6 we show a direct comparison in execution times (averaged over 10000 computations) for the evaluation of a spatial derivative on the quadrilateral/triangle and the hexahedron/tetrahedron, respectively, for increasing order of the approximation.

With a single exception, $n = 10$ in the tetrahedron, the fast schemes on the d-simplex are consistently faster than employing the tensor-product approximation. On the triangle we find an average of a 2/3 reduction in execution time compared to the quadrilateral, while the computation of derivatives on the tetrahedron tends to be twice as fast as on the hexahedron. Keeping in mind, however, that it takes two triangles to cover the area of a quadrilateral and six tetrahedra to fill the volume of a hexahedron, there is still a need for some improvement. These improvements, on the

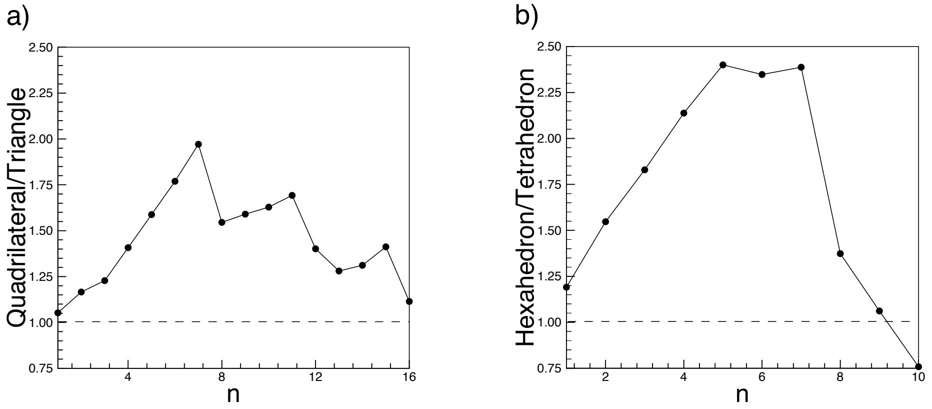

FIG. 6. *Comparison between the average execution time for the evaluation of a spatial derivatives using the unstructured fast methods based on the* 3*-symmetry on the n-simplex and a tensor-product approximation on the n-cube. In* (a) *we show the comparison between the quadrilateral and the triangle for increasing order of approximation n, while* (b) *shows a similar comparison for the hexahedron and the tetrahedron.*

other hand, are readily available by exploiting the higher degrees of symmetry inherent in the nodal sets as discussed above. Such an implementation could conceivably provide the computational kernel for the computation of derivatives at a speed equal to that achieved for tensor-product-based approximations, hence supplying the building blocks for a balanced scheme using simplices and cubes in a polymorphic high-order element setting.

5.2. Spectrum of the advective operator. Solving unsteady advection problems one often employs an explicit method for advancing the solution in time; i.e., a restriction, being connected to the eigenvalue spectrum of the advection operator, on the time-step is introduced. Understanding the eigenvalue spectrum of the total advection operator, including the boundary terms, is therefore important, as it provides if not a sufficient condition as the operator is strongly nonnormal, then a necessary condition for stability of the fully discrete scheme. Moreover, using the notion of generalized stability, as introduced in [22], for the advection problem and applying a locally stable method for advancing in time, e.g., a third- or fourth-order Runge–Kutta method, sufficient conditions for stability are also intimately related to the eigenvalue spectrum of the discrete operator as it needs to be bounded by a semicircle inscribed inside the stability region rather than the stability region of the time integration scheme as in the von Neumann stability.

Let us recall the semidiscrete form of the advection problem

$$\frac{d\boldsymbol{u}}{dt} + \boldsymbol{V}\cdot\left[\nabla\xi \mathrm{D}^{\xi} + \nabla\eta \mathrm{D}^{\eta} + \nabla\zeta \mathrm{D}^{\zeta}\right]\boldsymbol{u} = -\mathrm{M}^{-1}\mathrm{TA}\boldsymbol{u}, \tag{24}$$

where we assume homogeneous boundary conditions and, as usual, restrict the attention to the constant coefficient case on a straight-faced tetrahedron. Here TA represents a diagonal $N+1$ matrix with only the entries on the inflow parts of the faces being nonzero. The entries of T must be chosen to ensure stability following the result of Theorem 4.1, i.e., a sufficient condition for stability is that the entries of $\mathrm{T}_{ii} \geq \lambda_{\max}(\mathrm{M})/2$. Here $\lambda_{\max}(\mathrm{M})$ represents the maximum eigenvalue of any of the four face-based mass-matrices, i.e., a constant that depends solely on the nodal sets

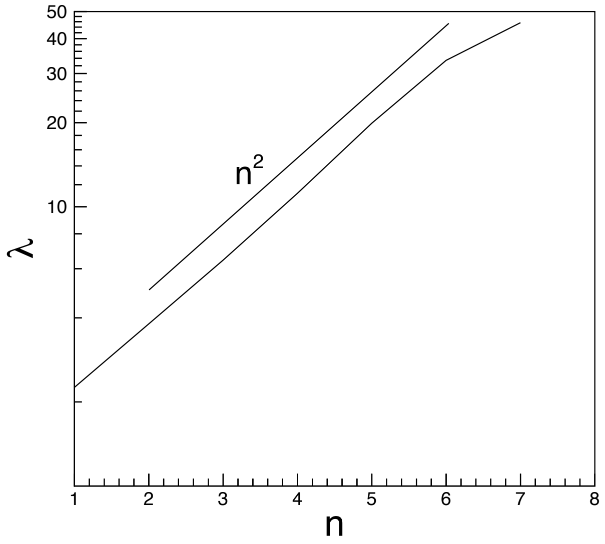

FIG. 7. *Magnitude of the maximum eigenvalue of the advective operator approximated on the tetrahedron.*

and hence can be precomputed. It should be noted that this procedure yields sufficient but often conservative conditions, resulting in unnecessarily large entries in T which again has a negative effect on the eigenvalue spectrum as discussed in [23, 14]. We have found that using the three parameters

$$\tau^{\text{Face}} = \frac{16}{5n^2} \ , \ \tau^{\text{Edge}} = \frac{2}{5n^2} \ , \ \tau^{\text{Vertex}} = \frac{1}{15n^2}$$

in T for entries corresponding to nodes on the faces, τ^{Face}, the edges, τ^{Edge}, and the vertices, τ^{Vertex}, results in well-behaved eigenvalue spectra. To make this more quantitative, we consider the situation where the advective velocity is given as

$$\boldsymbol{V} = [\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta]^T$$

in the spherical coordinate system with the origin at the center of the tetrahedron. The maximum eigenvalue appears for four combinations of (θ, ϕ) corresponding to the flow being perpendicular to one of the faces. However, more importantly, as shown in Figure 7, we have that $\lambda_{\max} \sim \mathcal{O}(n^2)$, similar to the optimal behavior of a tensor-product-based approximation.

We recall that the specification of the penalty term as well as the entries used in the above do not necessarily reflect an optimal combination but rather serve as an example of the discrete stability and behavior that can be expected from the types of schemes proposed here. In particular, the $\mathcal{O}(n^{-2})$ restriction on the time-step is critical and compares well with the bounds reported in [4] for a similar problem but using the basis proposed in [7, 8, 6] for the approximation of the operators. A more quantitative comparison is difficult due to significant differences in the schemes being considered.

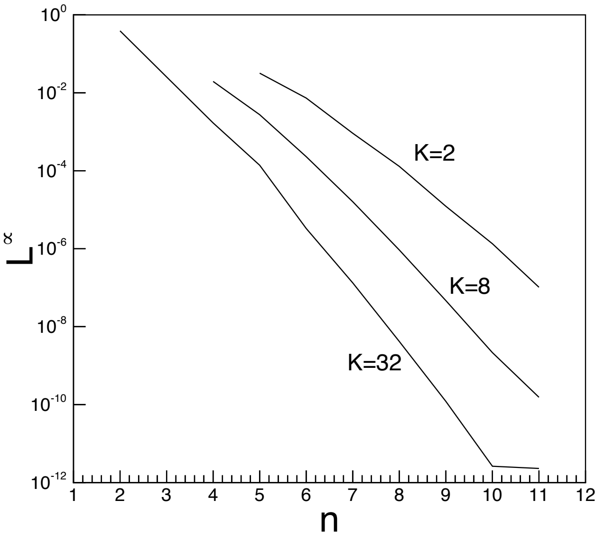

FIG. 8. *Maximum pointwise error at $t = 1.0$ when solving the linear wave-equation using K triangles, each employing an nth-order multivariate polynomial approximation.*

5.3. A final example. To illustrate the performance of the proposed framework, let us, for simplicity, consider the solution of the two-dimensional wave-equation

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = 0,$$

where $\boldsymbol{x} \in [-1,1]^2$ and the initial and boundary conditions are obtained from the exact solution

$$u(x,y,t) = \sin(\pi(x+y-t)).$$

While this problem is two-dimensional rather than three-dimensional, as has been the emphasis for most of the discussions in this work, all the elements of the scheme are contained in this simpler problem which illustrates well the performance of the overall framework.

To solve (22), we triangulate the computational domain using K equal-sized triangles. In Figure 8 we show the L^∞ error at $t = 1.0$ for solving the linear wave equation using K triangles, each employing an nth-order multivariate polynomial approximation. We observe in particular that the spectral convergence is achieved and that the scheme exhibits convergence through h- as well as p-refinement.

6. Concluding remarks. The development of spectral methods for problems defined on domains different from the d-cube remains important for several reasons. First, it expands the applicability of spectral methods to general domains. Second, when considering the d-simplex, as we have done here, it allows for the development of high-order spectral element methods which can effectively utilize existing algorithms for automated unstructured mesh generation. Third, it facilitates the development and implementation of spectral methods employing general polymorphic elements thereby allowing for a very efficient tiling of complex computational domains as they often appear in industrial settings.

In this paper we have addressed a number of issues central to the formulation of such schemes. One must, on one hand, ensure that the polynomial approximation/interpolation possesses qualities that allow for the formulation of high-order schemes. We have shown how to extend the electrostatic analogy to problems within the d-simplex but a similar approach could certainly be attempted to compute nodal sets in any convex volume although the general quality of such nodal sets remains unknown. However, the approach developed in section 4 is not specific to the nodal sets given here.

With the theoretical foundation for the schemes in place we continued by considering a number of issues central to the computational efficiency of the schemes. In particular we showed how to factorize the differentiation matrices in a way that significantly reduce the computational cost as well as the memory usage. Computational evidence suggests that the cost associated with the efficient evaluation of a derivative is 50%–66% of that for the pure tensor-product-based approximation on the d-cube, which is a performance very similar to that of schemes employing a tensor-product-based polynomial expansion on the d-simplex [24]. However, one should recall that this performance is achieved by exploiting only the most basic factorization with a number of additional symmetries in the d-simplex that have the potential to further reduce the computational cost. It is noteworthy that the identification of the factorization methods relies entirely on the symmetries inherent in the nodal sets and a careful ordering of the nodes. Indeed, the exact same approach is applicable to other nodal sets given in [9, 10, 12]. We finally considered the issue of discrete stability, making the important observation that the penalty parameters, ensuring semidiscrete stability, can be chosen such that the time-step is restricted only as $\mathcal{O}(n^{-2})$. The performance of the scheme was illustrated by solving a linear wave equations, confirming the expected accuracy and stability.

With the formulation of high-order methods for the solution of conservation laws and hyperbolic systems on tetrahedral domains we have developed a scheme, geometrically flexible and computationally efficient enough to handle problems of industrial complexity, yet highly accurate and long-time stable, hence providing the ideal setting for accurate and reliable modeling of large transient problems. Examples of suitable problems can be found in electromagnetics and aeroacoustics, among others, and we hope to report on such advanced applications in the near future.

Appendix A. Below is a table with the barycentric coordinates (b_1, b_2, b_3, b_4) for the nodal sets within a tetrahedron as computed using the electrostatic approach discussed in section 3.

The physical coordinates, $(x, y, z) \in \mathsf{D}$, can be recovered through (3). Moreover, to emphasize the significant symmetry of the nodal sets we have introduced the notation

$$n_1 + 4n_4 + 6n_6 + 12n_{12} + 24n_{24} = N_n^3 = \frac{(n+1)(n+2)(n+3)}{6},$$

representing the number of $1(n_1)$, $4(n_4)$, $6(n_6)$, $12(n_{12})$, and $24(n_{24})$ symmetries within the nodal set for each order of the polynomial. As discussed in section 5, the symmetries play an integral role in the formulation of the factorization methods for the efficient computation of derivatives. Finally, we recall that the nodal sets on the faces are taken from [11] and the nodes along each edge are Legendre–Gauss–Lobatto distributed.

n	n_1	n_4	n_6	n_{12}	n_{24}	b_1	b_2	b_3	b_4
1		1				1.0000000000	0.0000000000	0.0000000000	0.0000000000
2		1				1.0000000000	0.0000000000	0.0000000000	0.0000000000
			1			0.5000000000	0.5000000000	0.0000000000	0.0000000000
3		2				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.3333333333	0.3333333333	0.3333333333	0.0000000000
				1		0.7236067977	0.2763932023	0.0000000000	0.0000000000
4	1					0.2500000000	0.2500000000	0.2500000000	0.2500000000
		1				1.0000000000	0.0000000000	0.0000000000	0.0000000000
			1			0.5000000000	0.5000000000	0.0000000000	0.0000000000
				2		0.8273268354	0.1726731646	0.0000000000	0.0000000000
						0.2371200168	0.2371200168	0.5257599664	0.0000000000
5		2				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.1834903473	0.1834903473	0.1834903473	0.4495289581
				4		0.8825276620	0.1174723380	0.0000000000	0.0000000000
						0.6426157582	0.3573842418	0.0000000000	0.0000000000
						0.1575181512	0.1575181512	0.6849636976	0.0000000000
						0.4105151510	0.4105151510	0.1789696980	0.0000000000
6		3				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.3333333333	0.3333333333	0.3333333333	0.0000000000
						0.1402705801	0.1402705801	0.1402705801	0.5791882597
			2			0.5000000000	0.5000000000	0.0000000000	0.0000000000
						0.3542052583	0.3542052583	0.1457947417	0.1457947417
				3		0.9151119481	0.0848880519	0.0000000000	0.0000000000
						0.7344243967	0.2655756033	0.0000000000	0.0000000000
						0.1061169285	0.1061169285	0.7877661430	0.0000000000
					1	0.3097982151	0.5569099204	0.1332918645	0.0000000000
7		3				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.1144606542	0.1144606542	0.1144606542	0.6566180374
						0.2917002822	0.2917002822	0.2917002822	0.1248991534
				7		0.9358700743	0.0641299257	0.0000000000	0.0000000000
						0.7958500907	0.2041499093	0.0000000000	0.0000000000
						0.6046496090	0.3953503910	0.0000000000	0.0000000000
						0.0660520784	0.0660520784	0.8678958432	0.0000000000
						0.4477725053	0.4477725053	0.1044549894	0.0000000000
						0.2604038024	0.2604038024	0.4791923952	0.0000000000
						0.1208429970	0.1208429970	0.4770203357	0.0000000000
					1	0.2325524777	0.6759625951	0.0914849272	0.0000000000
8	1					0.2500000000	0.2500000000	0.2500000000	0.2500000000
		2				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.0991203900	0.0991203900	0.0991203900	0.7026388300
			2			0.5000000000	0.5000000000	0.0000000000	0.0000000000
						0.3920531037	0.3920531037	0.1079468963	0.1079468963
				8		0.9498789977	0.0501210023	0.0000000000	0.0000000000
						0.8385931398	0.1614068602	0.0000000000	0.0000000000
						0.6815587319	0.3184412681	0.0000000000	0.0000000000
						0.0660520784	0.0660520784	0.8678958432	0.0000000000
						0.2033467796	0.2033467796	0.5933064408	0.0000000000
						0.3905496216	0.3905496216	0.2189007568	0.0000000000
						0.1047451941	0.1047451941	0.5581946462	0.2323149656
						0.2419418605	0.2419418605	0.4062097450	0.1099065340
					2	0.3617970895	0.5541643672	0.0840385433	0.0000000000
						0.1801396087	0.7519065566	0.0679538347	0.0000000000

n	n_1	n_4	n_6	n_{12}	n_{24}	b_1	b_2	b_3	b_4
9		4				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.3333333333	0.3333333333	0.3333333333	0.0000000000
						0.0823287303	0.0823287303	0.0823287303	0.7530138091
						0.2123055477	0.2123055477	0.2123055477	0.3630833569
				11		0.9597669541	0.0402330459	0.0000000000	0.0000000000
						0.8693869326	0.1306130674	0.0000000000	0.0000000000
						0.7389624749	0.2610375251	0.0000000000	0.0000000000
						0.5826394788	0.4173605212	0.0000000000	0.0000000000
						0.0355775717	0.0355775717	0.9288448566	0.0000000000
						0.4640303025	0.4640303025	0.0719393950	0.0000000000
						0.1633923069	0.1633923069	0.6732153862	0.0000000000
						0.0873980781	0.0873980781	0.6297057875	0.1954980564
						0.0916714679	0.0916714679	0.4819523024	0.3347047619
						0.2040338880	0.2040338880	0.4996292993	0.0923029247
						0.3483881173	0.3483881173	0.2075502723	0.0956734931
					3	0.2966333890	0.6349633653	0.0684032457	0.0000000000
						0.1439089974	0.8031490682	0.0529419344	0.0000000000
						0.3225890045	0.4968009397	0.1806100558	0.0000000000
10		4				1.0000000000	0.0000000000	0.0000000000	0.0000000000
						0.0678316144	0.0678316144	0.0678316144	0.7965051568
						0.1805746957	0.1805746957	0.1805746957	0.4582759129
						0.3051527124	0.3051527124	0.3051527124	0.0845418628
			3			0.5000000000	0.5000000000	0.0000000000	0.0000000000
						0.3164336236	0.3164336236	0.1835663764	0.1835663764
						0.4219543801	0.4219543801	0.0780456199	0.0780456199
				11		0.9670007152	0.0329992848	0.0000000000	0.0000000000
						0.8922417368	0.1077582632	0.0000000000	0.0000000000
						0.7826176635	0.2173823365	0.0000000000	0.0000000000
						0.6478790678	0.3521209322	0.0000000000	0.0000000000
						0.0265250690	0.0265250690	0.9469498620	0.0000000000
						0.1330857076	0.1330857076	0.7338285848	0.0000000000
						0.4232062312	0.4232062312	0.1535875376	0.0000000000
						0.2833924371	0.2833924371	0.4332151258	0.0000000000
						0.1734555313	0.1734555313	0.5762731177	0.0768158196
						0.0724033935	0.0724033935	0.6893564961	0.1658367169
						0.0768451848	0.0768451848	0.5573732958	0.2889363346
					5	0.3934913008	0.5472380443	0.0592706549	0.0000000000
						0.2462883939	0.6991456238	0.0545659823	0.0000000000
						0.1163195334	0.8427538829	0.0409265838	0.0000000000
						0.2707097521	0.5811217960	0.1481684519	0.0000000000
						0.3019928872	0.4393774966	0.1776946096	0.0809350066

Acknowledgments. Many useful comments by T. Warburton, Oxford University, greatly improved the present work, as did the very constructive reports from the anonymous reviewers.

REFERENCES

[1] A. Patera, *A spectral element method for fluid dynamics: Laminar flow in a channel expansion*, J. Comput. Phys., 54 (1984), pp. 468–488.

[2] J. S. Hesthaven, *A stable penalty method for the compressible Navier–Stokes equations.* III. *Multidimensional domain decomposition schemes*, SIAM J. Sci. Comput., 20 (1998), pp. 62–93.

[3] I. Babuška and M. Suri, *The p- and $h-p$ versions of the finite element method, an overview*, Comput. Methods Appl. Mech. Engrg., 80 (1990), pp. 5–26.

[4] S. J. Sherwin and G. E. Karniadakis, *Tetrahedral hp finite elements: Algorithms and flow simulations*, J. Comput. Phys., 124 (1996), pp .14–45.

[5] B. A. Wingate and M. A. Taylor, *The Natural Function Space for Triangular and Tetrahedral Spectral Elements*, manuscript.

[6] M. Dubiner, *Spectral methods on triangles and other domains*, J. Sci. Comput., 6 (1991), pp. 345–390.

[7] J. Proriol, *Sur une famille de polynomes à deux variables orthogonaux dans un triangle*, C. R. Acad. Sci. Paris, 245 (1957), pp. 2459–2461.

[8] T. Koornwinder, *Two-variable analogues of the classical orthogonal polynomials*, in Theory and Application of Special Functions, R.A. Askey, ed., Academic Press, New York, 1975. pp. 435–495.

[9] Q. Chen and I. Babuška, *Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle*, Comput. Methods Appl. Mech. Engrg., 128 (1995), pp. 405–417.

[10] Q. Chen and I. Babuška, *The optimal symmetrical points for polynomial interpolation of real functions in a tetrahedron*, Comput. Methods Appl. Mech. Engrg., 137 (1996), pp. 89–94.

[11] J. S. Hesthaven, *From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex*, SIAM J. Numer. Anal., 35 (1998), pp. 655–676.

[12] M. A. Taylor and B. A. Wingate, *Fekete collocation points for triangular spectral elements*, SIAM J. Numer. Anal., submitted.

[13] M. H. Carpenter and D. Gottlieb, *Spectral methods on arbitrary grids*, J. Comput. Phys., 129 (1996), pp. 74–86.

[14] J. S. Hesthaven and D. Gottlieb, *Stable spectral methods for conservation laws on triangles with unstructured grids*, Comput. Methods Appl. Mech. Engrg., 175 (1999), pp. 361–381.

[15] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method. Volume 1. Basic Formulation and Linear Problems*, 4th ed., McGraw-Hill, London, 1989.

[16] K. C. Chung and T. H. Yao, *On lattices admitting unique Lagrange interpolations*, SIAM J. Numer. Anal., 14 (1977), pp. 735–743.

[17] L. P. Bos, *Bounding the Lebesgue Function for Interpolation in a Simplex*, J. Approx. Theory, 38 (1983), pp. 43–59.

[18] G. Szegö, *Orthogonal Polynomials*, American Mathematical Society Colloquium Publications 23, AMS, New York, 1939.

[19] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations* I, Springer Ser. Comput. Math. 8, Springer-Verlag, Berlin, 1989.

[20] J. S. Hesthaven, *A stable penalty method for the compressible Navier–Stokes equations.* II. *One-dimensional domain decomposition schemes*, SIAM J. Sci. Comput., 18 (1997), pp. 658–685.

[21] A. Solomonoff, *A fast algorithm for spectral differentiation*, J. Comput. Phys., 98 (1992), pp. 174–177.

[22] H. O. Kreiss and L. Wu, *On the stability definition of difference approximations for the initial boundary value problem*, Appl. Numer. Math., 12 (1993), pp. 213–227.

[23] J. S. Hesthaven and D. Gottlieb, *A stable penalty method for the compressible Navier–Stokes equations.* I. *Open boundary conditions*, SIAM J. Sci. Comput., 17 (1996), pp. 579–612.

[24] G. E. Karniadakis, *private communication*, 1998.