Journal of Computational Physics 228 (2009) 1573–1590

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Polymorphic nodal elements and their application in discontinuous Galerkin methods

Gregor J. Gassner [a,*], Frieder Lörcher [a], Claus-Dieter Munz [a], Jan S. Hesthaven [b]

[a] Institute for Aerodynamics and Gasdynamics, University of Stuttgart, Pfaffenwaldring 21, 70550 Stuttgart, Germany
[b] Division of Applied Mathematics, Brown University, Box F, Providence, RI 02912, USA

ARTICLE INFO

Article history:
Received 29 January 2008
Received in revised form 17 July 2008
Accepted 2 November 2008
Available online 25 November 2008

Keywords:
Discontinuous Galerkin
Nodal
Modal
Polynomial interpolation
hp Finite elements
Lebesgue constants
Quadrature free
Unstructured
Triangle
Quadrilateral
Polygonal
Tetrahedron
Hexahedron
Prism
Pentahedron
Pyramid

ABSTRACT

In this work, we discuss two different but related aspects of the development of efficient discontinuous Galerkin methods on hybrid element grids for the computational modeling of gas dynamics in complex geometries or with adapted grids. In the first part, a recursive construction of different nodal sets for *hp* finite elements is presented. They share the property that the nodes along the sides of the two-dimensional elements and along the edges of the three-dimensional elements are the Legendre–Gauss–Lobatto points. The different nodal elements are evaluated by computing the Lebesgue constants of the corresponding Vandermonde matrix. In the second part, these nodal elements are applied within the modal discontinuous Galerkin framework. We still use a modal based formulation, but introduce a nodal based integration technique to reduce computational cost in the spirit of pseudospectral methods. We illustrate the performance of the scheme on several large scale applications and discuss its use in a recently developed space-time expansion discontinuous Galerkin scheme.

1. Introduction

While discontinuous Galerkin (DG) methods were first proposed in the early 1970s in [36] it was not until the more recent development, initiated in the work of Cockburn and Shu [9,7,10–12], that these methods have matured into a powerful computational tool for the solution of systems of conservation laws and the equations of gas dynamics [4,13]. The extension to problems of viscous gas dynamics was initiated in [3,5] and this again has led to several related formulations [17,29,35] for the compressible Navier–Stokes equations. Many examples and further details along these lines can be found in [8,26,30].

In spite of these significant advances over the last decade, discontinuous Galerkin methods still suffer from being too expensive when compared to more traditional methods such as finite volume methods. This is particularly true for viscous

* Corresponding author. Tel.: +49 7116853454.
E-mail addresses: gassner@iag.uni-stuttgart.de (G.J. Gassner), loercher@iag.uni-stuttgart.de (F. Lörcher), munz@iag.uni-stuttgart.de (C.-D. Munz), Jan.Hesthaven@Brown.edu (J.S. Hesthaven).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.11.012

problems, where the common solution approach is based on a *mixed* finite element formulation, which was introduced in [3] and extended to higher order problems in [40,41]. In recent developments for the DG discretization of second order terms [15,16,32], the introduction of auxiliary variables is circumvented by the use of *two* partial integrations, or by *multiple* partial integrations for higher order operators [6].

Apart from this, however, a major computational cost is found in the traditional use of full order integration in the basic implementation, leading to an excessive computational cost for nonlinear problems. Deriving inspiration from the classic spectral methods [22] it is natural to consider the use of a nodal basis, leading to a formulation which in spirit shares much with a spectral collocation formulation in which the boundary conditions are imposed weakly. Such methods, often known as spectral penalty methods, have been developed for the compressible Navier–Stokes equations in [18–20] and extended to non-tensorial elements in [21,23].

The main advantages of such a formulation are found in the exact reduction to the standard discontinuous Galerkin formulation for linear problems, hence ensuring the accuracy for smooth problems, and the quadrature free approach for nonlinear problems, leading to a dramatic reduction in the computational cost. Furthermore, the use of a nodal basis with the correct structure of the points along the edges and faces leads to a natural separation of the basis into boundary and internal degrees of freedom. This becomes particularly beneficial for schemes using a high-order basis. Although the loss of exact integration opens up for the possibility of instabilities driven by aliasing, this is a well known and well understood phenomenon within the community of spectral methods [22]. We shall return to this concern briefly later.

One of the limitations of past nodal based formulations and schemes is the reliance on either cubic or tetrahedral element shapes. While these suffice in many cases, for problems with significant geometric flexibility one is tempted to also use more general types of elements such as prisms and pyramids.

In this work, we explore how one constructs such general nodal elements, using a nodal recursive construction, and optimize these for maximum accuracy by minimizing the Lebesgue constant of the associated multivariate Lagrange polynomial. This is discussed in Section 2 and sets the stage for Section 3 where we discuss in detail the use of these elements in a discontinuous Galerkin scheme and return to the issues of aliasing and the potential for instabilities caused by this. We shall also discuss how nodal elements can be used with advantage in an already existing scheme based on a modal expansion and finally we use a recently developed explicit space-time discretization to arrive at the fully discrete explicit scheme. In Section 4 we demonstrate how this general scheme, employing polymorphic elements and local time-stepping, can be used with benefit for both linear and nonlinear wave problems and, finally, the full three-dimensional compressible Navier–Stokes equations. Most of the tests illustrate the potential for a four fold reduction in computational time without impacting the accuracy by using the nodal based approach for large scale simulations. Section 5 concludes with a few general remarks and outlook toward future work.

2. The nodal elements

We will first focus on defining different sets of high-order basis functions for a given grid cell $Q \subset \mathbb{R}^d$. We introduce the *monomial* basis $\{\pi_i\}_{i=1,\dots,N}$ for the space of polynomials with degree less than or equal than p, where every basis function π_i is written as

$$\pi_i(\vec{x}) = x_1^{\alpha_1^i} \cdots x_d^{\alpha_d^i} \quad \text{with} \quad 0 \leqslant \alpha_1^i + \cdots + \alpha_d^i \leqslant p. \tag{1}$$

The dimension N of this space depends on the order p and on the spatial dimension d of the grid cell Q and is given by

$$N = N(p,d) = \frac{(p+d)!}{d!p!}. \tag{2}$$

Based on a monomial basis $\{\pi_i\}_{i=1,\dots,N}$ expanded in the barycenter of the grid cell Q and its geometry the construction of an orthonormal basis $\{\varphi_i\}_{i=1,\dots,N}$ using Gram–Schmidt orthogonalization is straight forward. This basis set is characterized by the property

$$\int_Q \varphi_i(\vec{x})\varphi_j(\vec{x})d\vec{x} = \delta_{ij}, \tag{3}$$

which holds for *arbitrary* grid cell shapes. With this *modal* basis we are now able to proceed to define a set of *nodal* basis functions. Given a set of interpolation points $\{\vec{\xi}_j\}_{j=1,\dots,M_I} \subset Q$, we can construct the nodal Lagrange basis $\{\psi_j\}_{j=1,\dots,M_I}$ and the nodal degrees of freedom $\underline{\tilde{u}}$ defined by the conditions

$$\begin{aligned} &\psi_j(\vec{\xi}_i) = \delta_{ij}, \\ &u(\vec{x}) := \sum_{j=1}^{N} \hat{u}_j \varphi_j(\vec{x}) \overset{!}{=} \sum_{i=1}^{M_I} \tilde{u}_i \psi_i(\vec{x}). \end{aligned} \tag{4}$$

Combining these conditions yields the transformations

$$\underline{\underline{V}}\,\underline{\hat{u}} = \underline{\tilde{u}} \quad \text{and} \quad \underline{\underline{V}}^T \underline{\psi} = \underline{\phi}, \tag{5}$$

where we introduce the generalized Vandermonde matrix $\underline{\underline{V}}$ with entries

$$V_{ij} = \varphi_j(\vec{\xi}_i), \quad i = 1, \dots, M_I; \quad j = 1, \dots, N. \tag{6}$$

The inverse of the Vandermonde matrix is not uniquely defined as $M_I \neq N$. If one is interested in avoiding this problem, one has to extend the modal basis from dimension N to dimension M_I. We refer to Lörcher and Munz [33] for a strategy to find a basis extensions for non-tensor product interpolation on a cartesian grid. However, the extension of this approach to the general case is not straightforward, as the non-singularity of the Vandermonde matrix is not guaranteed. To overcome this issue a singular value decomposition based strategy is used to define the following pseudo-inverse transformations

$$\underline{\hat{u}} = \underline{\underline{V}}^{-1}\underline{\tilde{u}} \quad \text{and} \quad \underline{\psi} = \underline{\underline{V}}^{-T}\underline{\phi}. \tag{7}$$

Using the pseudo-inverse Vandermonde matrix, condition (4) is only satisfied in the *least squares* sense. Thus, if we define the polynomial approximation of a function f as

$$f(\vec{x}) \approx f_I(\vec{x}) := \sum_{j=1}^{M_I} f(\vec{\xi}_j)\psi_j(\vec{x}) := \underline{\psi}^T\underline{\tilde{f}}, \tag{8}$$

the nodal degree of freedom $\tilde{f}_j = f(\vec{\xi}_j)$ is not the value of the interpolation $f_I(\vec{x})$ at the node $\vec{x} = \vec{\xi}_j$, as $\psi_j(\vec{\xi}_i) \neq \delta_{ij}$. Furthermore, the modal approximation

$$f(\vec{x}) \approx f_M(\vec{x}) := \sum_{j=1}^{N} \hat{f}_j\phi_j(\vec{x}) \quad \text{with} \quad \underline{\hat{f}} = \underline{\underline{V}}^{-1}\underline{\tilde{f}}, \tag{9}$$

is in the general case not equal to the nodal approximation

$$f_I(\vec{x}) \neq f_M(\vec{x}). \tag{10}$$

A good measure of the quality of such a polynomial approximation is given by the Lebesgue constant Λ, defined as

$$\Lambda := \max_{\vec{x} \in Q} \sum_{j=1}^{M_I} |\psi_j(\vec{x})|. \tag{11}$$

With this definition one easily realizes that

$$\|f - f_I\|_\infty \leqslant (1 + \Lambda)\|f - f^*\|_\infty, \tag{12}$$

where $\|.\|_\infty$ is the usual maximum norm and f^* is the best approximating polynomial of f. As the nodal basis $\{\psi_j\}_{j=1,\dots,M_I}$ depends only on the interpolation points $\{\vec{\xi}_i\}_{i=1,\dots,M_I}$, we next focus on the construction of nodal sets which minimize the growth of the Lebesgue constant with order p for different grid cell shapes. We restrict our attention to sets of interpolation points $\Omega_I := \{\vec{\xi}_i\}_{i=1,\dots,M_I}$ with the following characteristics:

- The interpolation based on these points is of order p for functions defined in the volume and for functions defined on the grid cell surfaces. This guarantees that the basis separates into boundary and interior components.
- The distribution of the points reflects the possible symmetries of the grid cell.
- The size of the nodal set $M_I \geqslant N$ depends on the order p, the dimension d and the *shape* of the grid cell.

2.1. One-dimensional node distributions

For an interval, $p+1$ points have to be chosen. There may be a number of different distributions of the $p+1$ points with the restriction that the endpoints are included. For instance, one can choose equidistant (E) points, Chebychev–Gauss–Lobatto points or Legendre–Gauss–Lobatto (LGL) points. We choose for every side in 2D and edge in 3D the LGL node distribution, as these are known to have a good Lebesgue constant Λ. An extended discussion of the one-dimensional case can be found in [24]. Based on the LGL node distribution we define the following warp function for $x \in [0;1]$

$$w_p(x) = \sum_{j=1}^{p+1} \xi_j^{LGL}\psi_j^E(x), \tag{13}$$

where $\{\xi_j^{LGL}\}_{j=1,\dots,p+1}$ are the Legendre–Gauss–Lobatto points and $\{\psi_j^E\}_{j=1,\dots,p+1}$ the Lagrange polynomials based on equidistant points. According to [39], $w_p(x)$ is a $(p+1)$*th* order approximation to the function which maps the 'bad' points (E) to the 'good' points (LGL).

2.2. Two-dimensional node distributions

In two space dimensions we split the set of interpolation points $\Omega_I(p)$ into two parts: The set of points that live in the interior of the cell and the set of points that live on the surface, named $\Omega_I^S(p)$. The set $\Omega_I^S(p)$ is defined such that it contains

$p+1$ LGL points for each side of the grid cell surface. This guarantees that the nodal approximation on the whole surface is of order $p+1$ and a separated basis by polynomial uniqueness. We note that using only these surface points for the approximation within the volume, the corresponding Vandermonde matrix is non-singular for p up to a value p^*, which depends on the shape of the grid cell. The value for p^* is 2 and 3 for triangles and quadrilaterals, respectively. Hence, for an interpolation with $p > p^*$, additional points in the interior of the grid cell are needed. The definition of these interpolation points can be done in the following recursive way

$$\Omega_I(p) := \begin{cases} \emptyset & \text{for} \quad p < 0, \\ \{\vec{x}_{bary}\} & \text{for} \quad p = 0, \\ \mathcal{M}_r(\Omega_I^S(p)) \cup \Omega_I(p - p^* - 1 + \pi_{2D}) & \text{for} \quad p > 0. \end{cases} \tag{14}$$

We notice that the interior nodes consist of nested and shrunk surface points. The mapping $\mathcal{M}_r$ determines how the surface point sets $\Omega_I^S(p)$ are nested and shrunk for every recursion step r, e.g. the mapping for the first recursion $r = 0$ is the identity, as the first points of the set $\Omega_I^S(p)$ are lying on the real surface of the grid cell and thus will not be shrunk. A simple approach for the mapping $\mathcal{M}_r$ for $r > 0$ would be one which yields an equidistant nesting. However, it is well known that the Lebesgue constant of the corresponding nodal basis is improved when the node distribution is more dense close to the boundary of the grid cell. Thus, to improve the nodal set, we propose to use a mapping which yields LGL-type nesting. In this work the warp function (13) is used to define the following mapping:

$$\begin{aligned} &\mathcal{M}_r(\vec{\xi}) = (\vec{\xi} - \vec{x}_{bary})\alpha(r) + \vec{x}_{bary}, \\ &\alpha(r) = 1 - 2w_{p_{(r=0)}}\left(\frac{r}{max(1, \tilde{r} - 1)}\right). \end{aligned} \tag{15}$$

where $\vec{\xi} \in \Omega_I^S(p), \vec{x}_{bary}$ denotes the barycenter of the grid cell, r the recursion level and $\tilde{r}$ is given as

$$\tilde{r} = \begin{cases} 2r_{max} - 1, & \text{if} \quad p_{(r=r_{max})} = 0, \\ 2r_{max}, & \text{otherwise}, \end{cases} \tag{16}$$

with the maximum number of recursions r_{max} and the polynomial degree at the last recursion level $p_{(r=r_{max})}$. Another approach is to start with the pure equidistant point distribution and optimize the nodal set with electrostatic considerations, as pro-

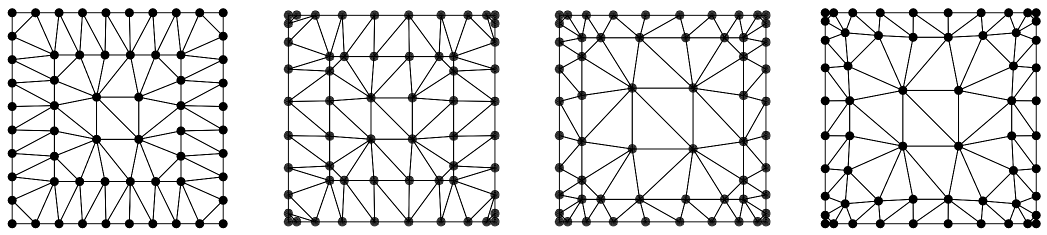


Fig. 1. Quadrilateral with $p = 9(\pi_{2D} = 0)$. From left to right: pure equidistant distribution, LGL points with equidistant nesting, LGL points with LGL-type nesting and optimized points.

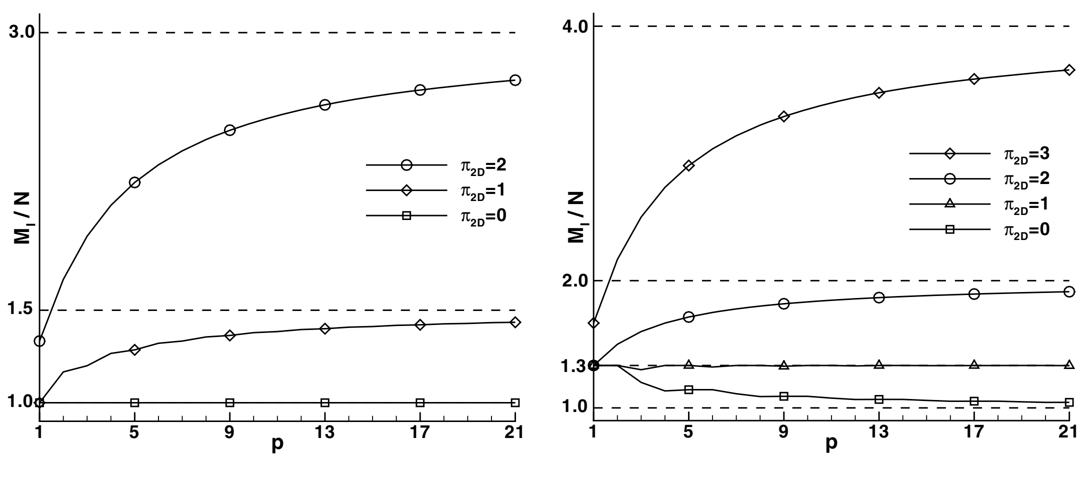


Fig. 2. Ratio of the number $M_I(p)$ of interpolation points and the dimension $N(p)$ of the polynomial space as a function of the polynomial degree p for different parameters π_{2D}, left for triangles and right for quadrilaterals. The limits for $p \to \infty$ are indicated with a dashed line.

posed by Hesthaven [24]. To illustrate these different strategies, we plot the corresponding node distributions of the $p = 9(\pi_{2D} = 0)$ quadrilateral in Fig. 1. The set with a purely equidistant distribution yields a Lebesgue constant $\Lambda = 97$, whereas the LGL points with equidistant nesting yields $\Lambda = 44$. Using LGL points and LGL-type nesting yields a Lebesgue constant of 21, which is slightly greater than $\Lambda = 17$ for the electro-static optimized points. Although the electro-static optimized interpolation points yield the best Lebesgue constant, we use the LGL points with LGL-type nesting in the computations shown below, as these point sets are easy and straight forward to implement. An important parameter in the recursion formula (14) is the integer π_{2D} which can be used to tune the relation between the interpolation quality and the number of points. For $\pi_{2D} = 0$, as considered up to now, algorithm (14) yields the smallest possible number of points and thus, the most efficient scheme in terms of the computational effort. However, we observed that in some cases, especially for quadrilaterals, the use of a few more points pays off in terms of a dramatically improved accuracy. The parameter π_{2D} with $0 \leqslant \pi_{2D} \leqslant p^*$ can be used to control the number of recursions in (14). Fig. 2 shows the ratio of the overall interpolation points $M_I(p)$ and the number $N(p)$ of the basis functions as a function of the polynomial degree p for different values of π_{2D}. The plot indicates that for triangles and $\pi_{2D} = 0$ the number is always optimal. For quadrilaterals and $\pi_{2D} = 0$ the number of interpolation points converges to the optimum with increasing p. In all the calculations presented in the following we use $\pi_{2D} = 0$ for triangles and $\pi_{2D} \in \{0, 1\}$ for quadrilaterals. For this type of interpolation points the corresponding Lebesgue constants Λ are listed in Table 1.

2.3. Three-dimensional node distributions

The definition of the three-dimensional set of interpolation points is done analogously to that of the two-dimensional case. Again, the set $\Omega_I(p)$ is split into two parts, where $\Omega_I^S(p, \pi_{2D})$ denotes the set of points on the surface. The recursion algorithm reads as follows:

$$\Omega_I(p) := \begin{cases} \emptyset & \text{for} \quad p < 0, \\ \{\vec{x}_{bary}\} & \text{for} \quad p = 0, \\ \mathcal{M}_r(\Omega_I^S(p, \pi_{2D})) \cup \Omega_I(p - p^* - 1 + \pi_{3D}) & \text{for} \quad p > 0. \end{cases} \tag{17}$$

In this work, the 3*D* *standard* shapes, namely tetrahedra, hexahedra, pentahedra (prisms) and pyramids are considered. The surfaces of these *standard* grid cells consist of triangles and quadrilaterals. Thus, for the definition of the surface point set $\Omega_I^S(p, \pi_{2D})$ we can use the two-dimensional nodal points from the previous subsection. Again, using surface points only yields non-singular interpolation up to a polynomial degree $0 < p \leqslant p^*$. The value of p^* is 3 for the tetrahedron, 5 for the hexahedron and 4 for the pentahedron and pyramid, respectively. We note that these values are independent of the choice of the parameter π_{2D}. Although the number of surface points increases with greater π_{2D}, the rank of the volume interpolation does not. We thus use the recursive nesting strategy (17) and introduce an additional parameter π_{3D}, which controls the number of recursions. The mapping $\mathcal{M}_r$ is again used to shrink the new nested surface points in a LGL-type manner (15). In Fig. 3 the ratios of the interpolation points $M_I(p)$ between the optimal number $N(p)$ for different parameters $\pi := (\pi_{3D}, \pi_{2D})$ are plotted. Again, for tetrahedra and $\pi = (0, 0)$ the number of interpolation points are always optimal, whereas for other grid cell shapes the ratio converges to 1 for $p \to \infty$. Compared to the 2*D* case the convergence for the 3*D* case is slower, however the magnitudes of the ratios are still reasonable. The corresponding Lebesgue constants are listed in Tables 2 and 3.

3. Application in discontinuous Galerkin methods

In the following we will discuss in detail how to construct a discontinuous Galerkin (DG) scheme using the nodal elements developed above. To keep matters simple we restrict the discussion to a scalar conservation law of the form

Table 1
Lebesgue constants Λ and number of interpolation points M_I for the two-dimensional interpolation points.

p	M_I	Λ	M_I	Λ	M_I	Λ
	Tri ($\pi_{2D} = 0$)		Quad ($\pi_{2D} = 0$)		Quad ($\pi_{2D} = 1$)	
1	3	1.0	4	1.5	4	1.5
2	6	1.7	8	3.0	8	3.0
3	10	2.1	12	4.0	13	3.2
4	15	3.8	17	4.2	20	5.3
5	21	3.2	24	5.8	28	4.6
6	28	4.6	32	7.5	37	4.5
7	36	6.8	40	15.3	48	5.1
8	45	7.5	49	14.5	60	7.5
9	55	8.6	60	21.0	73	8.0
10	66	11.2	72	28.6	88	10.8
11	78	18.8	84	61.8	104	14.8
12	91	20.2	97	62.7	121	15.4

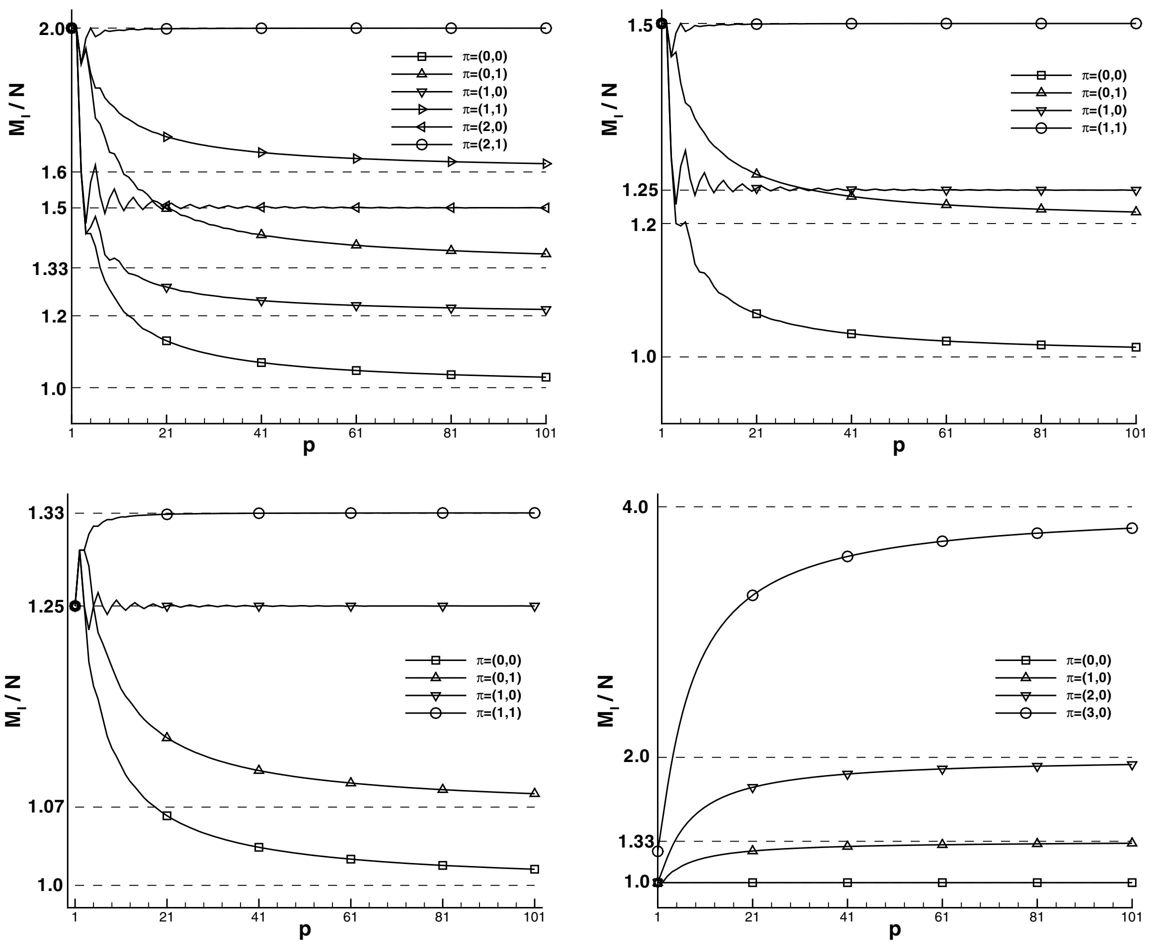


Fig. 3. Ratio of the number $M_I(p)$ of interpolation points and the dimension $N(p)$ of the polynomial space as a function of the polynomial degree p for hexahedron (top left), pentahedron (top right), pyramid (bottom left) and tetrahedron (bottom right) and different parameters $\pi = (\pi_{3D}, \pi_{2D})$. The limits for $p \to \infty$ are indicated with dashed lines.

$$u_t + \vec{\nabla} \cdot \vec{f}(u) = 0, \tag{18}$$

with appropriate initial and boundary conditions in a domain $\Omega \times [0, T] \subset \mathbb{R}^d \times \mathbb{R}_0^+$. The base of the semi-discrete DG formulation is a local weak formulation, which is obtained for a grid cell $Q \subset \Omega$ by multiplying (18) by a test function $\phi = \phi(\vec{x})$ and integrating over Q

Table 2
Lebesgue constants Λ and number of interpolation points M_I for the 3D interpolation sets with different parameters $\pi = (\pi_{3D}, \pi_{2D})$.

Tetrahedron/p	1	2	3	4	5	6	7	8	9	10	11
M_I	4	10	20	35	56	84	120	165	220	286	364
$\pi = (0,0)/\Lambda$	1.0	2.0	2.9	4.0	6.4	7.9	10.8	17.6	22.0	34.8	36.5
Hexahedron/p	1	2	3	4	5	6	7	8	9	10	11
M_I	8	20	32	50	80	117	160	214	280	358	448
$\pi = (0,0)/\Lambda$	1.5	5.0	6.4	8.8	17.0	20.3	41.5	47.6	103.6	201.3	454.2
M_I	8	20	32	50	81	124	172	226	298	389	492
$\pi = (1,0)/\Lambda$	1.5	5.0	6.4	8.8	11.6	35.6	37.1	46.6	103.2	113.5	148.2
M_I	8	20	38	68	104	147	208	280	364	472	592
$\pi = (0,1)/\Lambda$	1.5	5.0	4.8	15.6	11.2	13.0	30.4	32.7	52.0	111.6	323.5
M_I	8	20	38	68	105	154	220	298	394	509	642
$\pi = (1,1)/\Lambda$	1.5	5.0	4.8	15.6	8.9	18.1	13.3	31.0	49.4	58.0	78.0
M_I	8	20	32	51	88	136	184	245	336	444	552
$\pi = (2,0)/\Lambda$	1.5	5.0	6.4	7.8	9.1	14.0	30.2	28.3	40.2	56.9	124.2
M_I	8	20	38	69	112	166	238	329	438	570	726
$\pi = (2,1)/\Lambda$	1.5	5.0	4.8	5.9	8.9	11.1	12.5	20.3	21.2	31.5	61.2

Table 3
Lebesgue constants Λ and number of interpolation points M_I for the 3D interpolation sets with different parameters $\pi = (\pi_{3D}, \pi_{2D})$.

Pentahedron/p	1	2	3	4	5	6	7	8	9	10	11
M_I	6	15	26	42	67	101	141	188	248	322	407
$\pi = (0,0)/\Lambda$	1.7	3.7	4.4	6.0	8.1	21.4	22.7	42.3	96.7	112.1	175.2
M_I	6	15	26	43	72	110	152	205	278	365	458
$\pi = (1,0)/\Lambda$	1.7	3.7	4.4	5.9	10.0	11.2	23.4	24.2	61.6	74.2	167.8
M_I	6	15	29	51	79	116	165	224	296	382	482
$\pi = (0,1)/\Lambda$	1.7	3.7	4.1	9.4	7.2	15.6	15.0	34.2	70.8	86.8	117.6
M_I	6	15	29	52	84	125	179	247	329	428	545
$\pi = (1,1)/\Lambda$	1.7	3.7	4.1	5.7	10.0	8.7	13.0	17.2	33.3	34.5	60.2
Pyramid/p	1	2	3	4	5	6	7	8	9	10	11
M_I	5	13	25	42	66	98	138	187	247	319	403
$\pi = (0,0)/\Lambda$	1.5	3.0	4.2	6.8	9.7	15.6	24.5	39.7	71.4	146.9	366.2
M_I	5	13	25	43	70	106	150	205	275	359	455
$\pi = (1,0)/\Lambda$	1.5	3.0	4.2	5.3	7.2	11.4	20.0	20.8	54.8	38.6	83.6
M_I	5	13	26	45	70	103	146	199	263	339	428
$\pi = (0,1)/\Lambda$	1.5	3.0	3.8	8.4	9.0	13.1	20.2	32.8	65.5	137.7	360.6
M_I	5	13	26	46	74	111	159	219	292	380	484
$\pi = (1,1)/\Lambda$	1.5	3.0	3.8	6.0	7.0	9.5	12.9	18.0	27.4	27.4	42.1

$$\int_Q (u_t + \vec{\nabla}\cdot\vec{f}(u))\phi d\vec{x} = 0. \tag{19}$$

The usual weak formulation results after spatial integration by parts

$$\int_Q u_t\phi d\vec{x} + \int_{\partial Q}\left(\vec{f}(u)\cdot\vec{n}\right)\phi ds - \int_Q \vec{f}(u)\cdot\vec{\nabla}\phi d\vec{x} = 0. \tag{20}$$

For the DG discretization the exact solution u is next replaced by a piecewise polynomial approximation u_h. As this approximation is in general discontinuous across grid cell interfaces, the surface flux integrals are not well defined. To get an unique solution and a stable discretization, the normal flux $\vec{f}\cdot\vec{n}$ in the surface integral is replaced with a numerical flux function $g_{\vec{n}}$, which depends on the values from both sides of the grid cell interface. Independent of the choice of the numerical flux $g_{\vec{n}}$, there are a lot of different ways of how to implement the semi-discrete DG scheme. The implementations differ in terms of 'evaluation of the integrals and representation of the approximation u_h'. Hesthaven and Warburton introduced the quadrature free nodal DG scheme [25], where they used nodal basis functions $\{\psi_j\}_{j=1,\dots,M_I}$ for the approximation u_h and for the test functions. Recently, Dumbser et al. introduced an interesting novel quadrature free approach, where they constructed a local flux expansion in space and time [14]. In this work, we choose a more 'classic' approach, where we use the modal basis functions $\{\phi_j\}_{j=1,\dots,N}$ to define the test functions and the DG polynomial

$$u_h(\vec{x},t) := \sum_{j=1}^{N}\hat{u}_j(t)\varphi_j(\vec{x}) \quad \text{for} \quad \vec{x}\in Q, \tag{21}$$

with the time dependent modal DOF $\{\hat{u}_j(t)\}_{j=1,\dots,N}$. In standard modal DG implementations, the evaluation of the integrals is usually done with Gauss integration. For instance we get the following approximation for the first volume integral

$$\begin{aligned}\int_Q f_1(u_h)\frac{\partial\varphi}{\partial x_1}(\vec{x})d\vec{x} &\approx \sum_{j=1}^{(p+1)^d} f_1(u_h(\vec{\chi}_j))\frac{\partial\varphi}{\partial x_1}(\vec{\chi}_j)\omega_j,\\ &:= \underline{\underline{K}}^{1,GP}\underline{f}_1,\end{aligned} \tag{22}$$

where ω_j are the Gauss weights, $\vec{\chi}_j$ the Gauss positions, $\underline{f}_1$ the vector of flux evaluations and $\underline{\underline{K}}^{1,GP}$ the integration matrix with

$$(\underline{\underline{K}}^{1,GP})_{ij} := \frac{\partial\varphi_i}{\partial x_1}(\vec{\chi}_j)\omega_j, \quad i = 1,\dots,N; \quad j = 1,\dots,(p+1)^d. \tag{23}$$

We note that $u_h(\vec{\chi}_j)$ is evaluated using (21). If we consider a hexahedron with a $p = 5$ approximation, we get $(p+1)^d = 216$ evaluations with this strategy for the approximation of the volume integrals. We will show in the next subsection how to make use of the nodal elements to reduce the computational complexity of modal implementations.

3.1. The modal DG scheme with nodal integration

We first introduce the nodal interpolation of the nonlinear flux function according to (8)

$$f_1(u_h(\vec{x})) \approx f_{1,I}(\vec{x}) := \sum_{i=1}^{M_I}\tilde{f}_{1,i}\psi_i(\vec{x}), \tag{24}$$

where the nodal DOF is calculated as $\tilde{f}_{1,i} = f_1(u_h(\vec{\xi}_i))$. The evaluation of the DG polynomial (21) at the nodal points can be done using the Vandermonde matrix (5)

$$\underline{\tilde{u}} = \underline{\underline{V}}\underline{\hat{u}}, \tag{25}$$

yielding the nodal DOF of the flux as $\tilde{f}_{1,i} = f_1(\tilde{u}_i)$. As a next step, the interpolation of the flux function is inserted into the volume integral and integrated exactly

$$\begin{aligned}\int_Q f_1(u_h)\frac{\partial\varphi_j}{\partial x_1}(\vec{x})d\vec{x} &\approx \int_Q f_{1,I}(\vec{x})\frac{\partial\varphi_j}{\partial x_1}(\vec{x})d\vec{x}, \\ &:= \underline{\underline{K}}^1\underline{\tilde{f}}_1,\end{aligned} \tag{26}$$

where we introduced the general stiffness matrix

$$\underline{\underline{K}}^1 := \int_Q \frac{\partial\underline{\varphi}}{\partial x_1}(\vec{x})\underline{\psi}^T(\vec{x})d\vec{x} = \int_Q \frac{\partial\underline{\varphi}}{\partial x_1}(\vec{x})\underline{\varphi}^T(\vec{x})d\vec{x}\underline{\underline{V}}^{-1} =: \underline{\underline{K}}^{1,M}\underline{\underline{V}}^{-1}. \tag{27}$$

The evaluation of the stiffness matrix can be done with Gauss integration in an initial phase of the simulation, yielding a quadrature free approach. The surface integrals are treated in a similar manner. Comparing computational complexity we only need M_I evaluations to calculate the volume integrals. Considering for instance the $p = 5$ $(\pi = (1,1))$ hexahedron we get $M_I = 105$. Furthermore as the nodal elements support an interpolation in the volume and on the boundary at the same time no additional evaluations of the polynomial are needed to calculate the surface integrals. We note that the modal DG with nodal integration and the nodal DG [26] are strongly related. In fact the modal DG scheme with nodal integration can be interpreted as a nodal DG scheme using modal DOF and the Vandermonde matrix for the calculation of the nodal DOF (25). Reducing the accuracy of quadrature and relying on nodal products when computing nonlinear fluxes naturally introduces an error, known in spectral methods as aliasing [22]. However, the scheme maintains its full linear accuracy and the potential for aliasing driven instabilities is well understood and can, if needed, be controlled by the use of a weak *modal* filter (see [26]). In the present work, however, we have not found any need for this additional stabilization for any of the examples presented later.

4. Computational examples and validations

In the following we shall present a number of examples of increasing complexity to thoroughly validate the developed scheme. The spatial discontinuous Galerkin scheme is integrated in time using the recently developed space-time expansion (STE) approach [31,16], which allows an arbitrary high-order accurate local time-stepping.

4.1. Linear wave propagation

In this subsection the spatial accuracy of the nodal integration approach for a linear problem is investigated. We use the linearized Euler equations (LEE) as a model problem for linear wave propagation

$$U_t + \vec{\nabla}\cdot\vec{F}(U) = 0, \tag{28}$$

with the vector of the conservative variables $U = (\rho', u', v', w', p')^T$ and the LEE fluxes $\vec{F} := (F_1, F_2, F_3)^T := (\underline{A}_1U, \underline{A}_2U, \underline{A}_3U)^T$ with the Jacobi matrices

$$\underline{A}_1 = \begin{pmatrix} u_0 & \rho_0 & 0 & 0 & 0 \\ 0 & u_0 & 0 & 0 & \frac{1}{\rho_0} \\ 0 & 0 & u_0 & 0 & 0 \\ 0 & 0 & 0 & u_0 & 0 \\ 0 & \kappa p_0 & 0 & 0 & u_0 \end{pmatrix}, \quad \underline{A}_2 = \begin{pmatrix} v_0 & 0 & \rho_0 & 0 & 0 \\ 0 & v_0 & 0 & 0 & 0 \\ 0 & 0 & v_0 & 0 & \frac{1}{\rho_0} \\ 0 & 0 & 0 & v_0 & 0 \\ 0 & 0 & \kappa p_0 & 0 & v_0 \end{pmatrix}, \quad \underline{A}_3 = \begin{pmatrix} w_0 & 0 & 0 & \rho_0 & 0 \\ 0 & w_0 & 0 & 0 & 0 \\ 0 & 0 & w_0 & 0 & 0 \\ 0 & 0 & 0 & w_0 & \frac{1}{\rho_0} \\ 0 & 0 & 0 & \kappa p_0 & w_0 \end{pmatrix}, \tag{29}$$

where $U_0 := (\rho_0, u_0, v_0, w_0, p_0)^T$ is the background flow. As an example, a plane wave is initialized such that it contains only fluctuations in the right moving characteristic wave with the Eigenvalue $u_0 + c_0$

$$U = RW, \tag{30}$$

with $W = \widehat{W}\sin(\vec{k}\cdot\vec{x})$ and the eigenvector matrix

$$R = \begin{pmatrix} n_1 & n_2 & n_3 & \frac{\rho_0}{2c_0} & \frac{\rho_0}{2c_0} \\ 0 & -n_3 & n_2 & \frac{n_1}{2} & -\frac{n_1}{2} \\ n_3 & 0 & -n_1 & \frac{n_2}{2} & -\frac{n_2}{2} \\ -n_2 & n_1 & 0 & \frac{n_3}{2} & -\frac{n_3}{2} \\ 0 & 0 & 0 & \frac{\rho_0}{2c_0} & \frac{\rho_0}{2c_0} \end{pmatrix}, \tag{31}$$

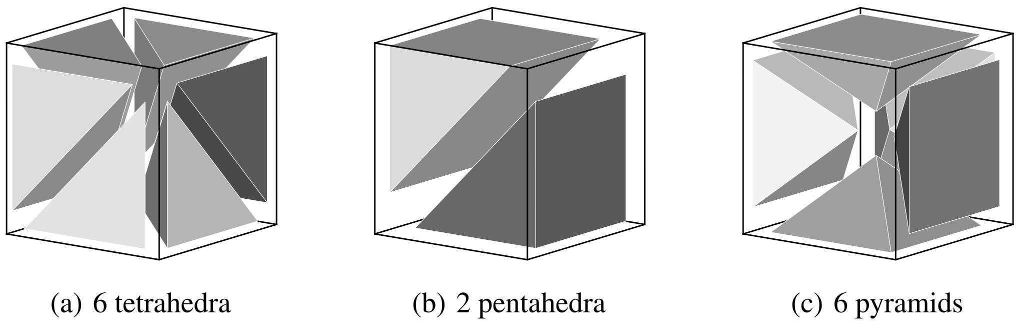


Fig. 4. Visualization of the different hybrid meshes.

with $c_0 = \sqrt{\kappa \frac{p_0}{\rho_0}}$. We choose the perturbation of the characteristic variable vector $\widehat{W} = (0, 0, 0, 10^{-3}, 0)^T$, the normal vector of the wave $\vec{n} = (1, 0, 0)^T$, the wave number vector $\vec{k} = (\pi, 0, 0)^T$ and the background flow

$U_0 = (1, 0, 0, 0, \kappa^{-1})^T$ with $\kappa = 1.4$, resulting in $c_0 = 1$. The computational domain $\Omega := [0; 2]^3$ is split into eight regular subdomains $\Omega_i = \vec{x}_i + [0; 1]^3, i = 1, \dots, 8$ with

$$\begin{aligned}
&\vec{x}_1 := (0,0,0)^T, \quad \vec{x}_2 := (1,0,0)^T, \quad \vec{x}_3 := (0,1,0)^T, \\
&\vec{x}_4 := (0,0,1)^T, \quad \vec{x}_5 := (1,1,0)^T, \quad \vec{x}_6 := (0,1,1)^T, \\
&\vec{x}_7 := (1,0,1)^T, \quad \vec{x}_8 := (1,1,1)^T. \qquad (32)
\end{aligned}$$

For our h-refinement tests we introduce the parameter $n \geqslant 1$. For a given n, we first split every sub-domain Ω_i into n^3 regular hexahedral elements. To generate the hybrid mesh, we furthermore split the hexahedra in the domain $i = 1$ into tetrahedra, in the domains $i = 2, 3, 4$ into prisms and in the domain $i = 8$ into pyramids. We illustrate the different hexahedra splittings in Fig. 4 (please note that the front pyramid is blanked for better visualization purpose). For $n = 1$ the hybrid prototype mesh consists of 21 grid cells. In Table 4 the experimental order of convergence (EOC) for this test case is shown for $p = 3$ and $p = 4$. These results suggest that the order of the STE-DG discretization is $p + 1$ in space *and* time. As expected, for the linear problem the results did not change when we increased the interpolation order $\tilde{p}$ or when we changed the grid points via the parameters π. To further investigate the behavior of the discretization for different polynomial approximations, five configurations were tested. In the first configuration, a fixed grid with 2^3 hexahedral grid cells was used. We plot in Fig. 5 the L_2 error norm of the pressure p' for polynomial order $p = 1$ up to $p = 8$ with $t_{end} = 20$. For the next configurations, the hexahedral base grid was further split into tetrahedra, prisms or pyramids, according to Fig. 4, resulting in 48, 16 and 48 grid cells, respectively. In the last configuration, the hybrid grid with $n = 1$ was used, resulting in 21 grid cells. Note that for the first four configurations the time steps do not differ over the computational domain, thus the local time-stepping STE-DG scheme reduces to a global time-stepping scheme. But for configuration five, the scheme runs in local time-stepping mode due to the different grid cell types and their different in-spheres. It is interesting to compare the performances of the different grid cells for this test case. Comparing the number of grid cells in the different configurations and the number of DOF, Fig. 5 shows that the error norms do not differ much, thus uncovering a superior approximation behavior of the hexahedral grid cells compared to the other types for this test case. Furthermore, if we compare the CPU time for the whole calculation, the hexahedral discretization remains superior again, as it allows larger time steps, resulting in the following ranking of this performance test: hexahedra (rel. CPU time $t = 1$), prisms (rel. CPU time $t \approx 4$), tetrahedra (rel. CPU time $t \approx 10$) and pyramids (rel. CPU time $t \approx 20$). Several investigations indicate the same trends for non-linear problems, especially for the Navier–Stokes equations.

4.2. *The Euler equations*

In the following test, the influence of the recursion parameter $\pi = (\pi_{3D}, \pi_{2D})$ and the influence of different interpolation orders $\tilde{p}$ is investigated. We consider in this subsection the non-linear Euler equations

Table 4
Experimental order of convergence for $p = 3$ and $p = 4$.

n	Nb cells	Nb DOF	$L_2(p')$	EOC	Nb DOF	$L_2(\rho e)$	EOC
		$p = 3$			p=4		
1	21	420	5,03E−5	–	9.408	3,51E−6	–
2	168	3360	2,21E−6	4.5	75.264	1,22E−7	4.8
3	567	11.340	4,22E−7	4.1	19.845	1,68E−8	4.9
4	1344	26.880	1,22E−7	4.1	47.040	4,06E−9	4.9

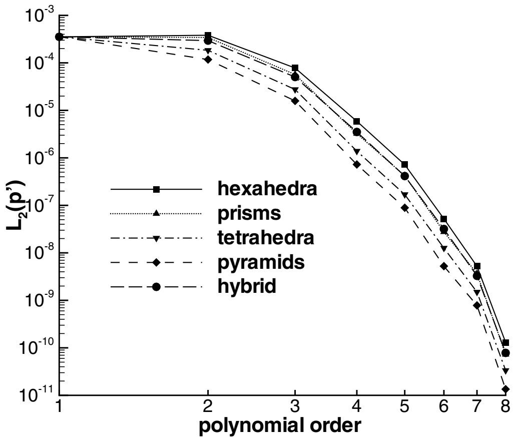


Fig. 5. Double logarithmic plot of L_2 error versus the polynomial order for different element types and grids.

$$U_t + \vec{\nabla} \cdot \vec{F}(U) = 0, \tag{33}$$

with the vector of the conservative variables $U = (\rho, \rho v_1, \rho v_2, \rho v_3, \rho e)^T$ and the Euler fluxes $\vec{F} := (F_1, F_2, F_3)^T$:

$$F_l(U) = \begin{pmatrix} \rho v_l \\ \rho v_1 v_l + \delta_{1l} p \\ \rho v_2 v_l + \delta_{2l} p \\ \rho v_3 v_l + \delta_{3l} p \\ \rho e v_l + p v_l \end{pmatrix}, \quad l = 1, 2, 3. \tag{34}$$

Here, we use the usual notation of the physical quantities: ρ, $\vec{v} = (v_1, v_2, v_3)^T$, p, and e denote the density, the velocity vector, the pressure, and the specific total energy, respectively. The adiabatic exponent is $\kappa = \frac{c_p}{c_v}$ with the specific heats c_p, c_v depending on the fluid, and assumed to be constant for this test. The system is closed with the perfect gas equation of state:

$$p = \rho R T = (\kappa - 1)\rho\left(e - \frac{1}{2}\vec{v} \cdot \vec{v}\right), \quad \text{and} \quad e = \frac{1}{2}\vec{v} \cdot \vec{v} + c_v T. \tag{35}$$

with the specific gas constant $R = c_p - c_v$. The considered test case is a three-dimensional variation of the isentropic vortex convection problem of Hu and Shu [27]

$$\begin{aligned} \vec{r}(\vec{x}, t) &= \vec{r}_{vortex} \times (\vec{x} - \vec{x}_0 - \vec{v}_0 \cdot t), \\ \delta v &= \frac{v_{max}}{2\pi} \exp\left(\frac{1 - \left(\frac{|\vec{r}|}{r_0}\right)^2}{2}\right), \\ \vec{v}(\vec{x}, t) &= \vec{v}_0 + \delta v \cdot \vec{r}, \\ \frac{T}{T_0} &= 1 - \frac{\kappa - 1}{2}\left(\frac{\delta v}{c_o}\right)^2, \\ \rho(\vec{x}, t) &= \rho_0 \left(\frac{T}{T_0}\right)^{\frac{1}{\kappa - 1}}, \\ p(\vec{x}, t) &= p_0 \left(\frac{T}{T_0}\right)^{\frac{\kappa}{\kappa - 1}}. \end{aligned} \tag{36}$$

If we choose, the rotational axis of the vortex $\vec{r}_{vortex} = (0, 0, 1)^T$ and $\rho_0 = p_0 = R = 1$, then the standard two-dimensional problem is recovered. For our test problem we chose the background flow $(\rho_0, \vec{v}_0^T, p_0) = (1, 1, 1, 1, \kappa^{-1})$, $\kappa = 1.4$, the rotational axis of the vortex $\vec{r}_{vortex} = (1, -0.5, 1)^T$, the initial center of the vortex $\vec{x}_0 = (0.5, 0.5, 0.5)^T$, the amplitude of the vortex $v_{max} = 0.1$, the halfwidth of the vortex $r_0 = 1$ and the endtime of the simulation $t_{end} = 4$. The computational domain $\Omega := [0, 5]^3$ with exact boundary conditions prescribed. The solution to this problem at time $t = 2$ with 6^3 $p = 5$ hexahedra is shown in Fig. 6. The results of tests with $p = 6$ trial functions with different parameters π and/or different interpolation orders $\tilde{p}$ are listed in Tables 5–8.

The general observation is that by increasing the number of interpolation points, the error norm decreases and the CPU time increases. We also compared the nodal integration to the standard Gaussian integration, where we chose $7^3 = 343$ ten-

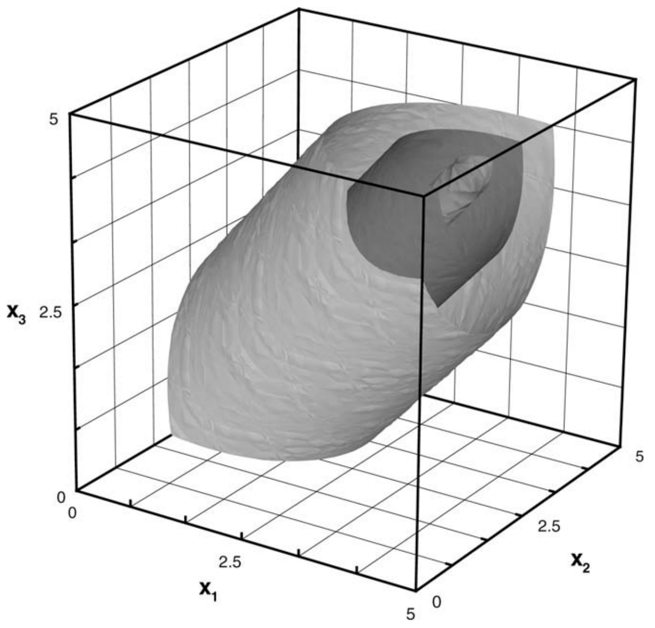


Fig. 6. 3D isentropic vortex. Isosurfaces of density ($\rho = 0.99977, 99989, 99998$).

Table 5
Results for different types of integration points for $p = 6$ hexahedra. The domain Ω is subdivided into eight hexahedra.

Interpolation order ($\tilde{p}$) and π	Nb Int points	$L_2(\rho)$	CPU time/EU (%)
$\tilde{p} = 6, \pi = (0, 0)$	117	1,9654E−05	100
$\tilde{p} = 6, \pi = (1, 0)$	124	1,7455E−05	107
$\tilde{p} = 6, \pi = (0, 1)$	147	1,8112E−05	120
$\tilde{p} = 6, \pi = (1, 1)$	154	1,6055E−05	121
$\tilde{p} = 6, \pi = (2, 0)$	136	1,7399E−05	110
$\tilde{p} = 6, \pi = (2, 1)$	166	1,5832E−05	125
$\tilde{p} = 7, \pi = (0, 0)$	160	1,7586E−05	127
$\tilde{p} = 8, \pi = (0, 0)$	214	1,6336E−05	154
$\tilde{p} = 7, \pi = (4, 2)$	512	1,4770E−05	255
Gauss Legendre points	637	1,4665E−05	403

Table 6
Results for different types of integration points for $p = 6$ pyramids. The domain Ω is subdivided into six pyramids.

Interpolation order ($\tilde{p}$) and π	Nb Int points	$L_2(\rho)$	CPU time/EU (%)
$\tilde{p} = 6, \pi = (0, 0)$	98	2,8744E−04	100
$\tilde{p} = 6, \pi = (1, 0)$	106	2,8256E−04	109
$\tilde{p} = 6, \pi = (0, 1)$	103	1,7078E−04	107
$\tilde{p} = 6, \pi = (1, 1)$	111	1,6332E−04	110
$\tilde{p} = 7, \pi = (0, 0)$	138	2,7298E−04	127
$\tilde{p} = 8, \pi = (0, 0)$	187	1,5537E−04	181
$\tilde{p} = 7, \pi = (1, 1)$	159	1,0978E−04	136
Gauss Jacobi points	588	9,8771E−05	425

sor product Jacobi Gauss points for the volume integrals and $7^2 = 49$ tensor product Jacobi Gauss points for each of the surface integrals. Although the results with standard Gauss cubature are slightly more accurate, comparing CPU times clearly confirms that the nodal type integration is more efficient.

Table 7
Results for different types of integration points for $p = 6$ prisms. The domain Ω is subdivided into eight hexahedra which are further subdivided into two prisms, yielding 16 grid cells.

Interpolation order ($\tilde{p}$) and π	Nb Int points	$L_2(\rho)$	CPU time/EU (%)
$\tilde{p} = 6, \pi = (0, 0)$	101	1,4853E−05	100
$\tilde{p} = 6, \pi = (1, 0)$	110	1,4235E−05	109
$\tilde{p} = 6, \pi = (0, 1)$	116	1,2260E−05	114
$\tilde{p} = 6, \pi = (1, 1)$	125	1,2250E−05	118
$\tilde{p} = 7, \pi = (0, 0)$	141	1,4210E−05	127
$\tilde{p} = 8, \pi = (0, 0)$	188	1,2925E−05	154
$\tilde{p} = 7, \pi = (0, 1)$	165	1,1562E−05	141
Gauss Jacobi points	588	1, 1006E-05	424

Table 8
Results for different types of integration points for $p = 6$ tetrahedra. The domain Ω is subdivided into six tetrahedra.

Interpolation order ($\tilde{p}$) and π	Nb Int points	$L_2(\rho)$	CPU time/EU (%)
$\tilde{p} = 6, \pi = (0, 0)$	84	1,414E−04	100
$\tilde{p} = 7, \pi = (0, 0)$	120	1,4386E−04	113
$\tilde{p} = 8, \pi = (0, 0)$	165	1,3945E−04	135
Gauss Jacobi points	539	1,3790E−04	399

4.3. Compressible Navier–Stokes equations

The three-dimensional unsteady compressible Navier–Stokes equations with a source term is given as

$$U_t + \vec{\nabla} \cdot \vec{F}(U) - \vec{\nabla} \cdot \vec{F}^v(U, \vec{\nabla}U) = S, \tag{37}$$

with the vector of the conservative variables U, the non-linear Euler fluxes $\vec{F} := (F_1, F_2, F_3)^T$ and the diffusion fluxes $\vec{F}^v := (F_1^v, F_2^v, F_3^v)^T$:

$$F_l^v(U, \vec{\nabla}U) = \begin{pmatrix} 0 \\ \tau_{1l} \\ \tau_{2l} \\ \tau_{3l} \\ \tau_{lj}v_j - q_l \end{pmatrix}, \quad l = 1, 2, 3. \tag{38}$$

The viscous stress tensor is given by

$$\underline{\tau} := \mu(\vec{\nabla}\vec{v} + (\vec{\nabla}\vec{v})^T - \frac{2}{3}(\vec{\nabla} \cdot \vec{v})\underline{I}), \tag{39}$$

and the heat flux as $\vec{q} = (q_1, q_2, q_3)^T$ with

$$\vec{q} := -k\vec{\nabla}T, \quad \text{with} \quad k = \frac{c_p\mu}{Pr}, \tag{40}$$

Here, the viscosity coefficient μ and the Prandtl number Pr depend on the fluid, and they are assumed to be constant. If we choose

$$S = \alpha \begin{pmatrix} \cos(\beta)(dk - \omega) \\ \cos(\beta)A + \sin(2\beta)\alpha k(\kappa - 1) \\ \cos(\beta)A + \sin(2\beta)\alpha k(\kappa - 1) \\ \cos(\beta)A + \sin(2\beta)\alpha k(\kappa - 1) \\ \cos(\beta)B + \sin(2\beta)\alpha(dk\kappa - \omega) + \sin(\beta)\left(\frac{dk^2\mu\kappa}{Pr}\right) \end{pmatrix}, \tag{41}$$

with $\beta := k(x_1 + x_2 + x_3) - \omega t, A = -\omega + \frac{k}{d-1}((-1)^{d-1} + \kappa(2d - 1))$ and $B = \frac{1}{2}((d^2 + \kappa(6 + 3d))k - 8\omega)$, the analytical solution to (37) and (41) is given by

$$U = (\sin(\beta)\alpha + 2, \sin(\beta)\alpha + 2, \sin(\beta)\alpha + 2, \sin(\beta)\alpha + 2, (\sin(\beta)\alpha + 2)^2)^T. \tag{42}$$

For our test we choose the coefficients $\kappa = 1.4, Pr = 0.72, \mu = 0.0001, R = 287.14$ and $\alpha = 0.5, \omega = 10, k = \pi$ with the dimension of the problem $d = 3$. We solve this problem with the recently developed modal STE-DG scheme for compressible Na-

Table 9
Experimental order of convergence for $p = 4$ and $p = 5$ with $\pi = (0,0)$ and $t_{end} = 1.0$.

n	Nb cells	Nb DOF	$L_2(\rho e)$	EOC	Nb DOF	$L_2(\rho e)$	EOC
		$p = 4$			$p = 5$		
2	168	5.880	6,13E−3	–	9.408	3,80E−3	–
4	1344	47.040	1,91E−4	5,0	75.264	9,36E−5	5,3
8	10752	376.320	4,32E−6	5,5	602.112	1,54E−6	5,9
16	86016	3.010.560	1,22E−7	5,1	4.816.896	2,38E−8	6,0

vier–Stokes equations [16], with the nodal modifications discussed above. The main building block of this discretization is a new weak formulation, where integration by parts is used twice, circumventing the need for resorting to a mixed first order system and thus avoiding the need for additional auxiliary variables. For the numerical fluxes we choose approximate Riemann solvers for *both*, the hyperbolic part and the parabolic part. For the approximation of the Euler flux we choose the HLLC flux [38] and for the approximation of the viscous fluxes the recently developed dGRP flux [15,16,32], which can be interpreted as a natural extension of the classic *interior penalty* flux [34] for the Laplace equation to the viscous terms of the compressible Navier–Stokes equations. The results of a convergence test with the hybrid grids from example 4.1 are listed in Table 9 for $p = 4$ and $p = 5$ with $\pi = (0,0)$, where we used $t_{end} = 1$ and periodic boundary conditions. The results show that the optimal order of convergence $EOC = p + 1$ is achieved.

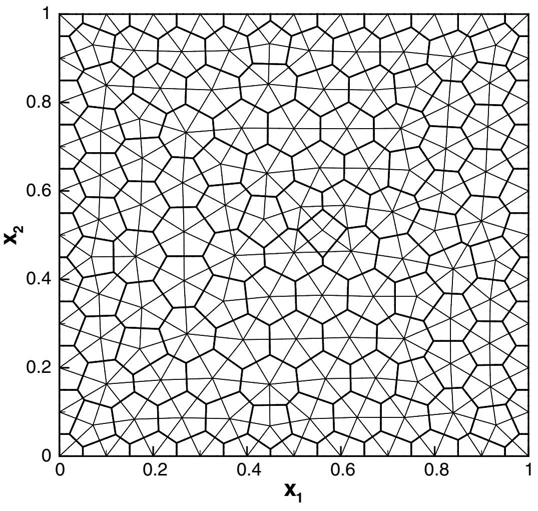
We list the average CPU time per element update (EU) and per degree of freedom for the 3D compressible Navier–Stokes equations with $p = 6$ (84 DOF/Element) in Table 10. Based on the investigations in Section 4.2, we chose for every grid cell type the most efficient combination (in terms of accuracy versus cpu time) of the parameters π and the interpolation order $\tilde{p}$. All CPU times were measured on one processor of a Intel Xeon Dual Core CPU with 2.66 GHz. An equivalent measurement for a 6th order compact finite difference scheme with 4th order Runge–Kutta time integration, [2], on the same CPU yields $\sim$56 µs.

4.3.1. Polygonal meshes

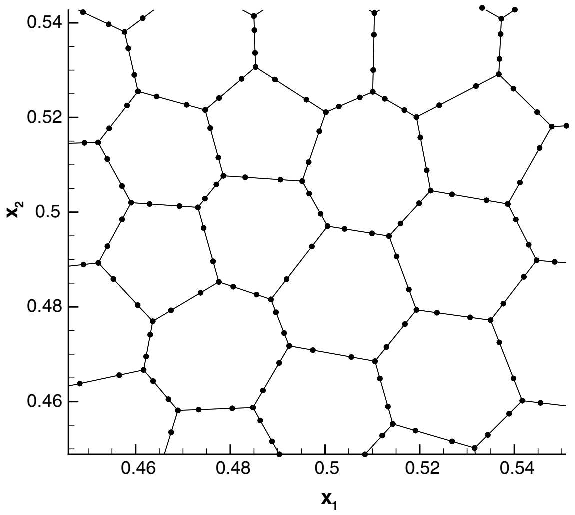
In this section preliminary results for a DG discretization with polygonal meshes are shown. We propose to apply the recursion based algorithm to define efficient sets of interpolation points for polygonal grid cells. Numerical investigations indicate that for a general grid cell the shape dependent parameter p^*, which is the maximal possible interpolation order with surface points only, has the value 'number of sides minus one', which we choose for all grid cell types (2D and 3D) discussed in this work. Starting from a triangle mesh the corresponding dual mesh is constructed and used as polygonal mesh, Fig. 7(a). The primal triangle mesh is no longer needed as it is only used to construct the dual mesh. The resulting polygonal mesh contains elements with 4–7 sides. For the distribution of the interpolation points two different strategies are used for an approximation with $p = 3$. For the first strategy we directly use the recursion algorithm (14) with a fixed recursion parameter π_{2D} for all elements. If we choose $\pi_{2D} = 0$, test configuration A shown in Fig. 7(b), the resulting interpolation grid is only distributed at grid cell boundaries, as all grid cells have at least four sides. As in the discussion above it is favorable for non-linear problems to use more interpolation points, i.e. increasing the recursion parameter π_{2D}. In Fig. 8(a) the recursion parameter is set to $\pi_{2D} = 3$, test configuration B, for all grid cells. In this extreme case where quadrilaterals (4 sides, 3 recursively defined interior point layers) and heptagons (7 sides, 0 recursively defined interior point layers) arise, the resulting point distribution is non-uniform and seems to be less well suited. To avoid this, our second strategy is to *fix* the number of recursions for every grid cell type, thus introducing the recursion parameter π_{2D} independently for every grid cell type. In Fig. 8(b) the interpolation grid for a fixed recursion number $r_{max} = 1$, test configuration C, with a second order inner point distribution is shown, corresponding to the parameter $\pi_{2D} = 2$ for quadrilaterals up to $\pi_{2D} = 5$ for heptagons. To validate this discretizations the compressible Navier–Stokes equations with a source term are considered, and we use the reduced two-dimensional version of the previous example with the same parameters, except for the parameter k which we change from π to 2π and the dimension d from 3 to 2. For the grid refinement, four different regular triangle grids with typical mesh size h are constructed and then converted to polygonal meshes, similar to what is shown in Fig. 7(a). The pre-computation of the surface and volume integral matrices is done on sub triangles with standard Gaussian integration. In Table 11 the results for configuration A and the results for the reference computation on the primal triangular grid with $t_{end} = 0.5$ and exact boundary conditions are shown. We notice first that the expected order of convergence is achieved. Considering efficiency, the results on the primal mesh are more accurate, whereas the CPU time for configuration A is $t_{CPU} = 378s$ and the CPU time for the

Table 10
CPU times for the 3D compressible Navier–Stokes equations with $(p = 6)$ STE-DG discretization (7th order in space and time).

Interpolation order $(\tilde{p})$ and π	Element type	CPU time/EU/DOF (µs)
$\tilde{p} = 6, \pi = (1,1)$	Hexahedron	39,9
$\tilde{p} = 7, \pi = (1,1)$	Pyramid	43,1
$\tilde{p} = 6, \pi = (0,1)$	Prism	31,5
$\tilde{p} = 6, \pi = (0,0)$	Tetrahedron	27,7

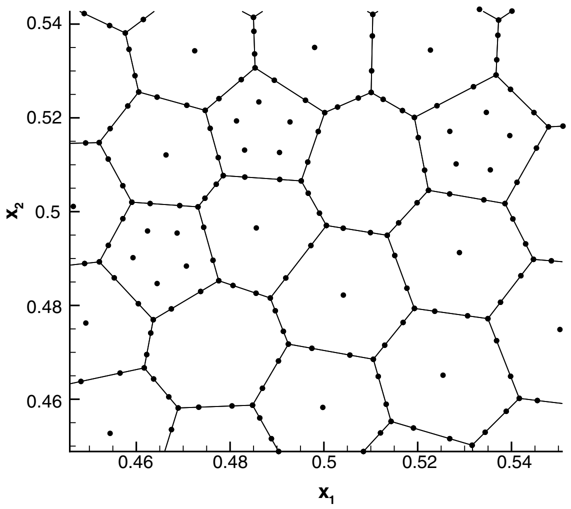


(a) primal and dual mesh

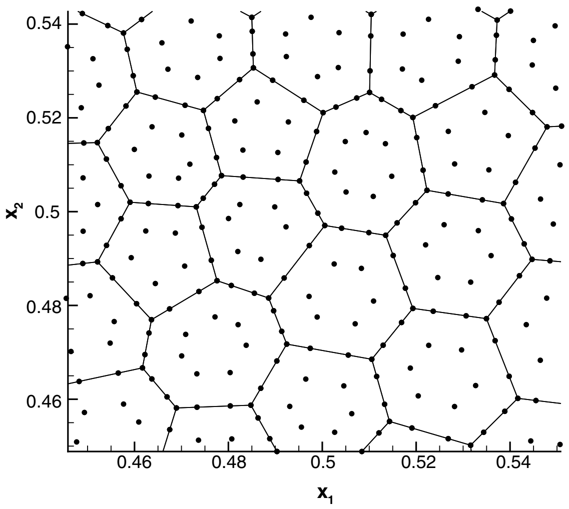


(b) configuration A

Fig. 7. Primal and dual mesh ($h = 0.1$) and detailed view of the interpolation grid ($h = 0.025$) with $p = 3(\pi_{2D} = 0)$ interpolation.



(a) configuration B



(b) configuration C

Fig. 8. Detailed view of the interpolation grid ($h = 0.025$) for $p = 3$ approximation with $\pi_{2D} = 3$ or $r_{max} = 1$.

primal configuration is $t_{CPU} = 594$ s. The reasons for the CPU time advantage is that the resulting polygonal configuration has only about half the DOF and allows larger (explicit) time steps. To account for the non-linearity of the Navier–Stokes fluxes, computations with configuration B and C are performed and corresponding results are listed in Table 12. We notice that the accuracy of the solution is improved, approaching the quality of the primal configuration solution. As expected, the results of configuration C are more accurate when compared to the results of configuration B. Considering the efficiency of the computations, the CPU time for test B is $t_{CPU} = 380$ s and for test C $t_{CPU} = 398$ s showing a large potential for DG discretizations on polygonal meshes compared to traditional triangular meshes. In future works we will investigate the influence of different node distribution strategies and recursion parameters for polygonal meshes and furthermore investigate the applicability of the recursion algorithm (17) with 'p^* = number of sides minus 1' for polyhedral meshes in three-dimensions.

4.3.2. Boundary layer instability

We consider in this example the evolution of a Tolmien–Schlichting wave in a subsonic compressible boundary layer. The computational domain Ω extends from $x_1 = 337$ to $x_1 = 890$ and $x_2 = 0$ to $x_2 = 22.35$. We choose subsonic inflow and outflow boundary conditions and at $x_2 = 0$ isothermal wall conditions with $T_w = 296$ K. The initial solution of the computation

Table 11
Experimental order of convergence for $p = 3$ (10 DOF per grid cell) for reference test on primal triangular mesh and for test configuration A.

h	Nb cells	$L_2(\rho e)$	EOC	Nb cells	$L_2(\rho e)$	EOC
	Triangular configuration			Configuration A		
0,2	62	2,44E−3	–	42	1,28E−2	–
0,1	226	1,92E−4	3,7	134	1,31E−3	3,3
0,05	896	1,07E−5	4,3	489	7,16E−5	4,2
0,025	3595	6,42E−7	4,1	1878	4,77E−6	3,9

Table 12
Experimental order of convergence for $p = 3$ (10 DOF per grid cell) for test configuration B and C.

h	Nb cells	$L_2(\rho e)$	EOC	$L_2(\rho e)$	EOC
		Configuration B		Configuration C	
0,2	42	9,55E−3	–	5,17E−3	–
0,1	134	7,22E−4	3,7	4,25E−4	3,6
0,05	489	3,38E−5	4,4	2,64E−5	4,0
0,025	1878	1,84E−6	4,2	1,64E−6	4,0

is obtained from a similarity solution with Mach number $M_\infty = 0.8$ and $T_\infty = 280$ K. The Reynolds number $Re := \frac{\rho_\infty v_1 \delta_1}{\mu(T_\infty)} = 1000$, based on the displacement thickness at the inflow δ_1. Using δ_1 as the reference length, we get $\delta_1 = 1.0$ at the inflow and the boundary layer thickness $\delta_{99} = 2.95$ and $\delta_{99} = 4.8$ at the inflow and outflow, respectively. The temperature dependence of the viscosity μ is modeled using Sutherland's law

$$\mu(T) = \mu(T_\infty)T^{3/2}\frac{1+T_s}{T+T_s}, \tag{43}$$

with $\mu(T_\infty) = 1.73510^{-5}$ kg ms and $T_s = 110.4$ K.

The inflow at $x_1 = 337$ is superimposed with a forcing term, composed of the eigenfunction of the Tolmien–Schlichting wave with the fundamental frequency $\omega_0 = 0.0688$. For a detailed description of the similarity solution and the eigenfunction we refer to Babucke et al. [1]. The computational domain is subdivided into 48×22 regular quadrilaterals and discretized with a $p = 6(\pi_{2D} = 1)$ STE-DG scheme, resulting in 29568 DOF. The endtime of the simulation was set to $\frac{t_{end}}{T_0} = 37$, where $T_0 = \frac{2\pi}{\omega_0} \approx 92$, to ensure a periodic solution. To analyze our results we apply a discrete Fourier analysis using one period of the forcing frequency T_0 from $\frac{t}{T_0} = 36$ to $\frac{t}{T_0} = 37$. We plot the maximal amplitude of v_1 with respect to x_2 as a function of x_1 in Fig. 9. For comparison, corresponding results obtained with a 6th order compact finite difference code with 330×150 grid points and 4th order Runge–Kutta time integration [1] are included, showing good agreement. We furthermore plot the amplification rate α_i of the velocity v_1 based on the maximal amplitude in Fig. 9. Again, the result is in good agreement with the reference result [1] and the predictions of linear stability theory.

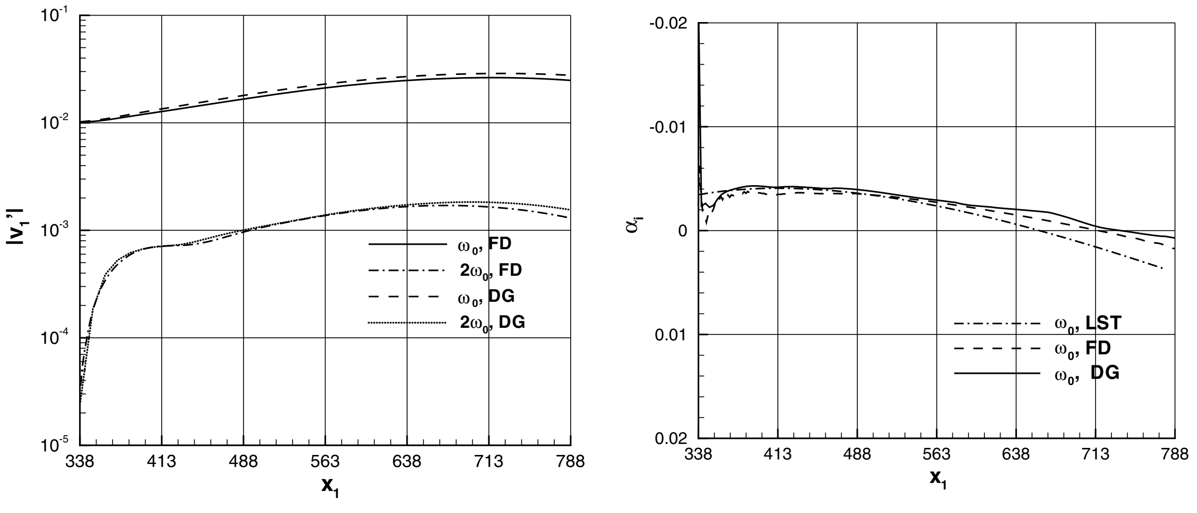


Fig. 9. Maximum amplitudes of v_1 (left). Amplification rate α_i of u_1 based on maximum amplitude (right).

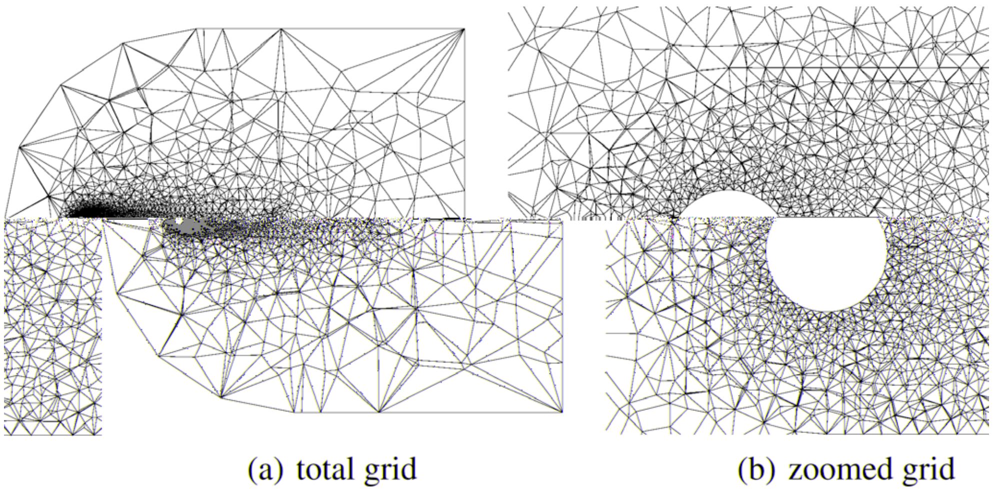


Fig. 10. Visualization of the grid for the sphere example.

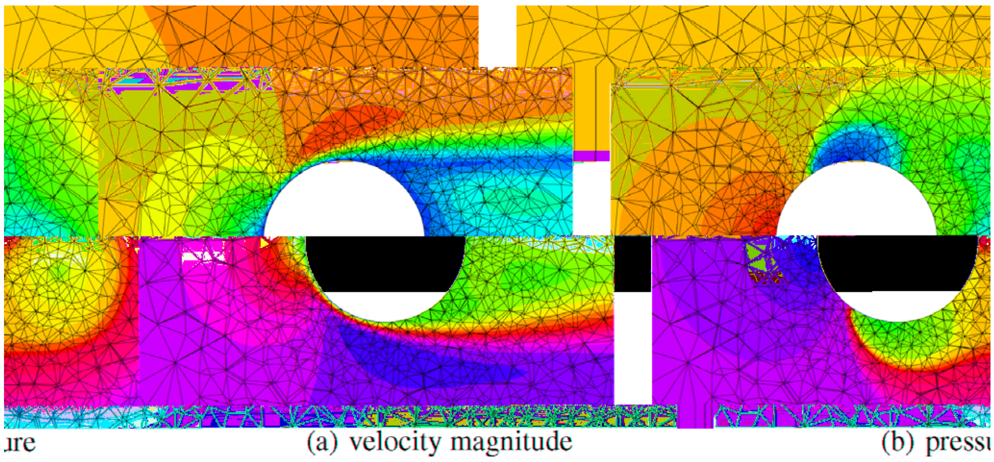


Fig. 11. Contour plot of the instantaneous velocity magnitude $|\vec{v}| = 0.0, \ldots, 0.3478$ and pressure $p = 0.688, \ldots, 0.762$.

4.3.3. *Flow past a Sphere at $Re = 300$*

We consider in this example a sphere with radius $r = 1$ centered at $\vec{x}_0 = (0, 0, 0)^T$ and solve the 3D unsteady compressible Navier–Stokes equations with Mach number $M = 0.3$ and Reynolds number $Re = 300$ based on the diameter of the sphere. The computational domain extends from $x_1 = -20$ to $x_1 = 100$ and $x_2, x_3 = \pm 30$. The grid consists of ≈160,000 tetrahedra, where the wake of the sphere is resolved with $h \approx 0.4$. The surface of the sphere is discretized using triangles with $h \approx 0.1$. To capture the right geometry of the sphere, tetrahedra with curved boundary surfaces are used. We plot the cut

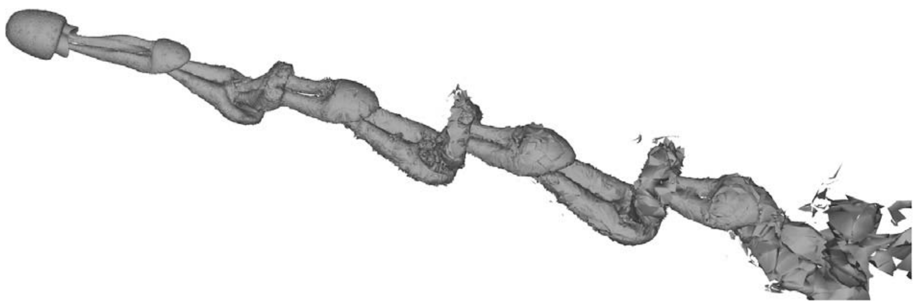


Fig. 12. Isometric view of λ_2 isosurface.

Table 13
Force coefficients and Strouhal number.

	C_d	ΔC_d	C_l	ΔC_l	Str
	0.673	0.0031	−0.065	0.015	0.135
Tomboulides [37]	0.671	0.0028	–	–	0.136
Johnson and Patel [28]	0.656	0.0035	−0.069	0.016	0.137

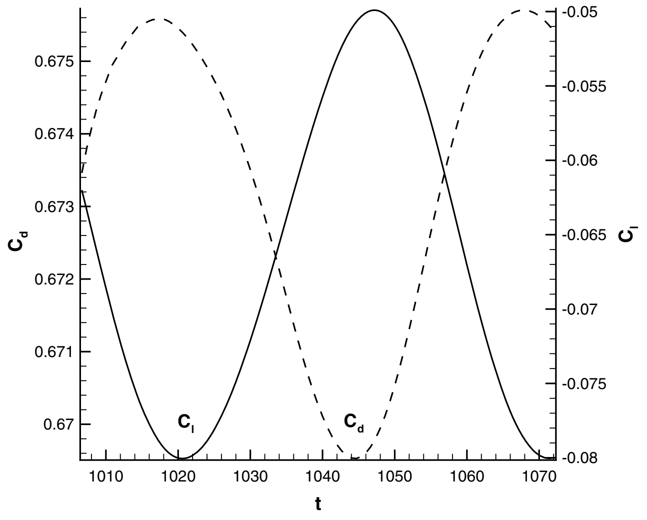


Fig. 13. Drag and lateral force coefficient.

of the grid on a plane with $\vec{n}_{plane} = (0, 1, 0)^T$ in Fig. 10. For the calculation a $p = 3$ STE-DG scheme is used, resulting in ≈3,000,000 DOF. A contour plot of the velocity magnitude, Fig. 11, shows that the boundary layer is resolved within 1–2 tetrahedral elements. In Fig. 12 the structure of the vortices are shown using the λ_2 vortex detection criterium. We list in Table 13 the resulting force coefficients, the corresponding oscillating amplitudes and the Strouhal number *Str*. For comparison, results from Tomboulides [37] and Johnson and Patel [28], obtained using an incompressible simulation, are listed as well, showing food agreement. In Fig. 13 we plot the drag coefficient C_d and the lateral force coefficient C_l versus time t.

5. Conclusion

Part one of this paper deals with a framework for efficient polynomial interpolation on polymorphic grid cells, i.e. the definition of a nodal interpolation basis and the associated operators. In our framework, for non simplex grid cells the number of nodal basis functions is higher than the number of modal basis functions. We showed that one way to get a reasonable Vandermonde matrix is to use the singular value decomposition framework to build a least squares inverse. The properties of these Vandermonde matrices (and the corresponding interpolation) depend solely on the position of the interpolation points. We consider in this paper only interpolation points with a symmetric distribution, points which support an interpolation of order p in the volume of the grid cell and simultaneously an interpolation of the same order on each of the faces of the grid cell. This is used to introduce a simple construction guideline, which is based on a recursive algorithm starting from a given surface points distribution. Using a set of 1D points, we can successively define points for 2D faces and consequently define points for 3D volumes.

In the second part of the paper we introduced a novel integration framework for modal discontinuous Galerkin schemes. Borrowing from nodal methods a mixed quadrature free modal–nodal DG scheme is constructed. If we consider an existing modal DG code with Gauss type integration, it is easy to implement the nodal type integration, as essentially only the integration matrices have to be changed. As an example the nodal based integration was combined with the recently developed modal space-time expansion discontinuous Galerkin scheme. Numerical investigations indicate a four fold reduction in computational time without impacting the accuracy.

Acknowledgments

This project was supported by the Deutsche Forschungsgemeinschaft (Schwerpunktprogramm 1276: MetStroem and Graduiertenkolleg 1095/1: Aerothermodynamische Auslegung eines Scramjet – Antriebssystems für zukünftige Raumtransportsysteme). The authors would further like to thank Andreas Babucke for providing the initial condition and the eigensolutions for the Tolmien–Schlichting wave example. The work of the last author (JSH) was partially supported by NSF Career Award DMS-0132967.

References

[1] A. Babucke, M.J. Kloker, U. Rist, Accuracy analysis of the fundamental finite-difference methods on non-uniform grids, SIAM J. Num. Anal., submitted for publication.
[2] A. Babucke, M.J. Kloker, U. Rist, Dns of a plane mixing layer for the investigation of sound generation mechanisms, Comput. Fluid 37 (2008) 360–368.
[3] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, J. Comput. Phys. 131 (1997) 267–279.
[4] F. Bassi, S. Rebay, A high-order discontinuous Galerkin finite element method solution of the 2D euler equations, J. Comput. Phys. 138 (1997) 251–285.
[5] F. Bassi, S. Rebay, Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier–Stokes equations, Int. J. Numer. Meth. Fluid 40 (2002) 197–207.
[6] Y. Cheng, C.-W. Shu, A discontinuous Galerkin finite element method for time dependent partial differential equations with higher order derivatives, Math. Comput. 77 (2008) 699–730.
[7] B. Cockburn, S. Hou, C.W. Shu, The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case, Math. Comput. 54 (1990) 545–581.
[8] B. Cockburn, G.E. Karniadakis, C.-W. Shu, Discontinuous Galerkin Methods, Lecture Notes in Computational Science and Engineering, Springer, 2000.
[9] B. Cockburn, S.Y. Lin, C. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems, J. Comput. Phys. 84 (1989) 90–113.
[10] B. Cockburn, C. Shu, The Runge–Kutta local projection p^1-discontinuous Galerkin method for scalar conservation laws, M^2AN 25 (1991) 337–361.
[11] B. Cockburn, C.W. Shu, TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: General framework, Math. Comput. 52 (1989) 411–435.
[12] B. Cockburn, C.W. Shu, The Runge–Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems, J. Comput. Phys. 141 (1998) 199–224.
[13] R.B.K. Devine, J. Flaherty, Parallel, adaptive finite element methods for conservation laws, Appl. Numer. Math. 14 (1994) 255–283.
[14] M. Dumbser, D. Balsara, E. Toro, C.-D. Munz, A unified framework for the construction of one-step finite-volume and discontinuous Galerkin schemes on unstructured meshes, J. Comput. Phys. (2008), doi:10.1016/j.jcp.2008.05.025.
[15] G. Gassner, F. Lörcher, C.-D. Munz, A contribution to the construction of diffusion fluxes for finite volume and discontinuous Galerkin schemes, J. Comput. Phys. 224 (2) (2007) 1049–1063.
[16] G. Gassner, F. Lörcher, C.-D. Munz, A discontinuous Galerkin scheme based on a space-time expansion. II. Viscous flow equations in multi dimensions, J. Sci. Comput. 34 (3) (2008) 260–286.
[17] R. Hartmann, P. Houston, Symmetric interior penalty DG methods for the compressible Navier–Stokes equations I: Method formulation, Int. J. Numer. Anal. Model. 3 (1) (2006) 1–20.
[18] J. Hesthaven, A stable penalty method for the compressible Navier–Stokes equations. II. One-dimensional domain decomposition schemes, SIAM J. Sci. Comput. 18 (1998) 658–685.
[19] J. Hesthaven, A stable penalty method for the compressible Navier–Stokes equations. III. Multi dimensional domain decomposition schemes, SIAM J. Sci. Comput. 20 (1999) 62–93.
[20] J. Hesthaven, D. Gottlieb, A stable penalty method for the compressible Navier–Stokes equations. I. Open boundary conditions, SIAM J. Sci. Comput. 17 (1996) 579–612.
[21] J. Hesthaven, D. Gottlieb, Stable spectral methods for conservation laws on triangles with unstructured grids, Comput. Meth. Appl. Mech. Eng. 175 (1999) 361–381.
[22] J. Hesthaven, S. Gottlieb, D. Gottlieb, Spectral Methods for Time-Dependent Problems, Cambridge University Press, Cambridge, 2006.
[23] J. Hesthaven, C.H. Teng, Stable spectral methods on tetrahedral elements, SIAM J. Sci. Comput. 21 (2000) 2352–2380.
[24] J.S. Hesthaven, From electrostatics to almost optimal nodal sets for polynomial interpolation in simplex, SIAM J. Numer. Anal. 35 (2) (1998) 655–676.
[25] J.S. Hesthaven, T. Warburton, Nodal high-order methods on unstructured grids I: Time-domain solution of Maxwell's equations, J. Comput. Phys. 181 (1) (2002) 186–221.
[26] J.S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, Springer-Verlag, New York, 2008.
[27] C. Hu, C.-W. Shu, Weighted essentially non-oscillatory schemes on triangular meshes, J. Comput. Phys. 1505 (1999) 97–127.
[28] T. Johnson, V. Patel, Flow past a sphere up to a Reynolds number of 300, J. Fluid Mech. 378 (1999) 19–70.
[29] C. Klaij, J.J.W. Van der vegt, H. Van der ven, Spacetime discontinuous Galerkin method for the compressible Navier–Stokes equations, J. Comput. Phys. 217 (2) (2006) 589–611.
[30] B. Li, Discontinuous Finite Elements in Fluid Dynamics and Heat Transfer, Springer-Verlag, Berlin, 2006.
[31] F. Lörcher, G. Gassner, C.-D. Munz, A discontinuous Galerkin scheme based on a space-time expansion. I. Inviscid compressible flow in one space dimension, J. Sci. Comput. 32 (2) (2007) 175–199.
[32] F. Lörcher, G. Gassner, C.-D. Munz, An explicit discontinuous Galerkin scheme with local time-stepping for general unsteady diffusion equations, J. Comput. Phys. 227 (11) (2008) 5649–5670.
[33] F. Lörcher, C.-D. Munz, Lax–Wendroff-type schemes of arbitrary order in several space dimensions, IMA J. Numer. Anal. 27 (2007) 593–615.
[34] J.A. Nitsche, Über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind, Abh. Math. Sem. Univ. Hamburg 36 (1971) 9–15.
[35] J. Peraire, P. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput. 30 (4) (2008) 1806–1824.
[36] W. Reed, T. Hill, Triangular mesh methods for the neutron transport equation, Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.
[37] A. Tomboulides, Direct and large-eddy simulation of wake flows: flow past a sphere, Ph.D. thesis, Princeton University, 1993.
[38] E. Toro, Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer, 1999. June.
[39] T. Warburton, An explicit construction of interpolation nodes on the simplex, J. Eng. Math. 56 (2006) 247–262.
[40] J. Yan, C.-W. Shu, A local discontinuous Galerkin method for KdV type equations, SIAM J. Numer. Anal. 40 (2) (2002) 769–791.
[41] J. Yan, C.-W. Shu, Local discontinuous Galerkin methods for partial differential equations with higher order derivatives, J. Sci. Comput. 17 (1–4) (2002) 32.