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

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 \( h^p \) finite elements is presented. The different nodal elements are evaluated by computing the Lebesgue constants of the corresponding Vandermonde matrix. 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. In the second part, we apply these nodal elements as the basis for discontinuous Galerkin schemes. We shall discuss both the immediate nodal formulation as well as the widely used modal formulation where in the latter case a nodal based integration technique is introduced to reduce the computational cost. We shall 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.

Key words: discontinuous Galerkin, nodal, modal, polynomial interpolation, \( h^p \) finite elements, Lebesgue constants, quadrature free, unstructured, triangle, quadrilateral, tetrahedron, hexahedron, prism, pentahedron, pyramid.
1 Introduction

While discontinuous Galerkin methods were first proposed in the early 1970’s in [31] it was not until the more recent development, initiated in the work of Cockburn and Shu [9,10,8,7,11], that these methods matured into a powerful computational tool for the solution of systems of conservation laws and the equations of gas dynamics [12,4]. The extension to problems of viscous gas dynamics was initiated in [3,5] with the rewriting of higher-order operators into systems of first-order equations and this again has lead to several related formulations along similar lines. Many examples and further details along these lines can be found in [26] for fluid dynamics applications and in the recent more general text [23].

In spite of these significant advances over the last decade, discontinuous Galerkin method still suffer from being perceived as being too expensive when compared to more traditional methods such as finite volume methods. This is particularly true for viscous problems where the approach introduced in [3] leads to a significant increase in required computational effort compared to traditional methods. A recent development to address this particular concern is discussed by several authors [14,34] for the second order diffusion operator and in [6] for more general operators. Here, we shall indeed use the approach in [14] to discretize the second order operators.

Apart from this, however, a major computational cost is found in the traditional use of full order integration in the basic implementation, leading to excessive computational cost for nonlinear problems. Deriving inspiration from the classic spectral methods [15] 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 [16–18] and extended to non-tensorial elements in [19,20].

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 overall 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. As is usually the case, all good things come with a price and in this case the loss of exact integration opens the possibility for instabilities driven by aliasing. This is, however, a well known phenomenon and is well understood within the community of spectral methods [15]. We shall return to this concern briefly later but otherwise refer to the thorough discussion of discontinuous Galerkin methods based on nodal elements which can be found in [23].
One of the limitations of past nodal based formulations and schemes has been 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 construct such nodal general elements, using a recursive construction, and optimize these for maximum accuracy by minimizing the Lebesque 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 general elements in a discontinuous Galerkin scheme and return to the issues of aliasing and 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 introduce the space-time formulation and combine all the pieces of the formulation to arrive at the fully discrete 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 4 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 $\{\pi_i\}_{i=1,...,N}$ for the space of polynomials with degree less than or equal than $p$, where every basis function $\pi_i$ could be written as

$$\pi_i(\vec{x}) = x_1^{\alpha_i^1} \cdot ... \cdot x_d^{\alpha_i^d} \quad \text{with} \quad 0 \leq \alpha_i^1 + ... + \alpha_i^d \leq p. \quad (1)$$

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

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

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

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

3
which holds for arbitrary grid cell shapes. With this modal basis we are now able to define a set of nodal basis functions. Given a set of interpolation points \( \{ \xi_j \}_{j=1,...,M_I} \subset Q \), we can construct the nodal Lagrange basis \( \{ \psi_j \}_{j=1,...,M_I} \) defined by the conditions
\[
\psi_j(\xi_i) = \delta_{ij}, \\
u(\vec{x}) := \sum_{j=1}^{N} \tilde{u}_j \varphi_j(\vec{x}) = \sum_{i=1}^{M_I} \tilde{u}_i \psi_i(\vec{x}).
\]
Combining these conditions yields the following transformations
\[
V \tilde{u} = \tilde{\nu} \quad \text{and} \quad V^T \psi = \phi,
\]
where we introduce the generalized Vandermonde matrix \( V \) with the entries
\[
V_{ij} = \varphi_j(\xi_i), \quad i = 1,...,M_I; \quad j = 1,...,N.
\]
In the general case the inverse of the Vandermonde matrix is not uniquely defined as \( M_I \neq N \) and we use the singular value decomposition framework to define the pseudoinverse transformations
\[
\tilde{u} = V^{-1} \tilde{\nu} \quad \text{and} \quad \psi = V^{-T} \phi,
\]
which in general satisfy the conditions (4) in the least squares sense. If one is interested in avoiding the least squares definition of the inverse transformation, one has to extend the modal basis from dimension \( N \) to dimension \( M_I \). This extension is, however, not straightforward as the non-singularity of the Vandermonde matrix is difficult to achieve. We refer to Lörcher and Munz [29] for a strategy to find such non-singular basis extensions.

With the nodal basis set \( \{ \psi_i \}_{i=1,...,M_I} \), we define the following polynomial approximation of a function
\[
f(\vec{x}) \approx f_I(\vec{x}) := \sum_{j=1}^{M_I} f(\xi_j) \psi_j(\vec{x}) =: \tilde{\psi}^T \vec{f}.
\]
A good measure of the quality of such an approximation is given by the Lebesgue constant \( \Lambda \), defined as
\[
\Lambda := \max_{\vec{x} \in Q} \sum_{j=1}^{M_I} |\psi_j(\vec{x})|.
\]
With this definition one easily realizes that
\[
\| f - f_I \|_{\infty} \leq (1 + \Lambda) \| f - f^* \|_{\infty},
\]
where \( \| . \|_{\infty} \) is the usual maximum norm and \( f^* \) is the best approximating polynomial of \( f \). As the nodal basis \( \{ \psi_j \}_{j=1,...,M_I} \) depends only on the interpolation points
\{\xi_i\}_{i=1,...,M_I}
, we next focus on the construction of nodal sets for different grid cell shapes which minimize the growth of the Lebesgue constant with the order \( p \). We restrict the attention to sets of interpolation points \( \Omega_I := \{\xi_i\}_{i=1,...,M_I} \) 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 \( M_I \geq N \) 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 distributions of the \( p + 1 \) points with the restriction that the endpoints are included. For instance, one can choose equidistant 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 for a good Lebesgue constant \( \Lambda \). An extended discussion of the one-dimensional case can be found in [21].

### 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 \( \Omega^S_I(p) \). The set \( \Omega^S_I(p) \) 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 nonsingular for \( p \) up to a value \( p^* \), which depends on the shape of the grid cell. The value for \( p^* \) is 3 and 4 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

\[
\Omega_I(p) := \begin{cases} 
\emptyset & \text{for } p < 0, \\
\{\bar{x}_{\text{barycenter}}\} & \text{for } p = 0, \\
\mathcal{M}_r(\Omega^S_I(p)) \cup (\Omega_I(p - (p^* + 1) + \pi_{2D})) & \text{for } p > 0.
\end{cases}
\]

We notice that the interior nodes consist of nested and 'shrunk' surface points. The mapping \( \mathcal{M}_r \) determines how the point sets are nested and shrunk for every

5
recursion step \( r \), e.g., the mapping for the first recursion \( r = 1 \) is the identity, as the first points of the set \( \Omega_1^S(p) \) are lying on the real surface of the grid cell and thus will not be shrunk. A simple approach for the mappings \( M_r \) for \( r > 1 \) 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. Starting from this node distributions, it is also possible to further optimize the nodal sets with electrostatic considerations, as proposed by Hesthaven [21].

To illustrate these different strategies, we plot the corresponding node distributions of the \( p = 9 \) \((\pi_{2D} = 0)\) quadrilateral in figure 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 electrostatic 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 easily and straightforward to implement.

An important parameter in the recursion formula (11) is the integer \( \pi_{2D} \) which can be used to tune the relation between the interpolation quality and number of points. For \( \pi_{2D} = 0 \), as considered up to now, algorithm (11) yields the smallest possible number of points and thus, the most efficient scheme according to 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 \( \pi_{2D} \) with \( 0 \leq \pi_{2D} \leq p^* \) can be used to control the number of recursions in (11). Figure 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 \( \pi_{2D} \). 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 \( p \).
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 $\pi_{2D}$, left for triangles and right for quadrilaterals. The limits for $p \to \infty$ are indicated with a dashed line.

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 $\Lambda$ are listed in table 1.

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

Table 1
Lebesgue constants $\Lambda$ and number of interpolation points $M_I$ for the two-dimensional interpolation points.
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 $\Omega^S_I(p)$ denotes the set of points on the surface. The recursion algorithm reads as follows

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

(12)

In this work the 3D standard shapes, namely tetrahedra, hexahedra, pentahedra (prisms) and pyramids are considered. The surfaces of this standard grid cells consist of triangles and quadrilaterals. Thus, for the definition of the surface point set $\Omega^S_I(p, \pi_{2D})$ 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 $0 < p \leq p^*$. 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 $\pi_{2D}$. Although the number of surface points increases with greater $\pi_{2D}$, the rank of the volume interpolation does not. We thus use the recursive nesting strategy (12) and introduce an additional parameter $\pi_{3D}$, 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. In figure 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 1.0 for $p \to \infty$. Compared to the 2D case the convergence for the 3D case is slower, however the magnitudes of the ratios are still reasonable. The corresponding Lebesgue constants are listed in tables 2 and 3.

Before proceeding to the application of these nodal sets, we note that the recursion based strategy of defining interpolation points can be extended to other elements shapes, such as polyhedra and general custom designed elements.

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.
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.

### 3.1 The semi-discrete formulation

To keep matters simple we restrict the discussion to a scalar conservation law of the form

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

with appropriate initial and boundary conditions in a domain $\Omega \times [0, T] \subset \mathbb{R}^d \times \mathbb{R}^+$. 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 (13) by a testfunction $\phi = \phi(\vec{x})$ and integrating over $Q$

$$\int_Q \left( u_t + \nabla \cdot \vec{f}(u) \right) \phi \, d\vec{x} = 0.$$
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>tetrahedron/p</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_I$</td>
<td>4</td>
<td>10</td>
<td>20</td>
<td>35</td>
<td>56</td>
<td>84</td>
<td>120</td>
<td>165</td>
<td>220</td>
<td>286</td>
<td>364</td>
</tr>
<tr>
<td>$\pi = (0, 0)/\Lambda$</td>
<td>1.0</td>
<td>2.0</td>
<td>2.9</td>
<td>4.0</td>
<td>6.4</td>
<td>7.9</td>
<td>10.8</td>
<td>17.6</td>
<td>22.0</td>
<td>34.8</td>
<td>36.5</td>
</tr>
<tr>
<td>hexahedron/p</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>32</td>
<td>50</td>
<td>80</td>
<td>117</td>
<td>160</td>
<td>214</td>
<td>280</td>
<td>358</td>
<td>448</td>
</tr>
<tr>
<td>$\pi = (0, 0)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>6.4</td>
<td>8.8</td>
<td>17.0</td>
<td>20.3</td>
<td>41.5</td>
<td>47.6</td>
<td>103.6</td>
<td>201.3</td>
<td>454.2</td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>32</td>
<td>50</td>
<td>81</td>
<td>124</td>
<td>172</td>
<td>226</td>
<td>298</td>
<td>389</td>
<td>492</td>
</tr>
<tr>
<td>$\pi = (1, 0)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>6.4</td>
<td>8.8</td>
<td>11.6</td>
<td>35.6</td>
<td>37.1</td>
<td>46.6</td>
<td>103.2</td>
<td>113.5</td>
<td>148.2</td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>38</td>
<td>68</td>
<td>104</td>
<td>147</td>
<td>208</td>
<td>280</td>
<td>364</td>
<td>472</td>
<td>592</td>
</tr>
<tr>
<td>$\pi = (0, 1)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>4.8</td>
<td>15.6</td>
<td>11.2</td>
<td>13.0</td>
<td>30.4</td>
<td>32.7</td>
<td>52.0</td>
<td>111.6</td>
<td>323.5</td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>38</td>
<td>68</td>
<td>105</td>
<td>154</td>
<td>220</td>
<td>298</td>
<td>394</td>
<td>509</td>
<td>642</td>
</tr>
<tr>
<td>$\pi = (1, 1)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>4.8</td>
<td>15.6</td>
<td>8.9</td>
<td>18.1</td>
<td>31.3</td>
<td>31.0</td>
<td>49.4</td>
<td>58.0</td>
<td>78.0</td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>32</td>
<td>51</td>
<td>88</td>
<td>136</td>
<td>184</td>
<td>245</td>
<td>336</td>
<td>444</td>
<td>552</td>
</tr>
<tr>
<td>$\pi = (2, 0)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>6.4</td>
<td>7.8</td>
<td>9.1</td>
<td>14.0</td>
<td>30.2</td>
<td>28.3</td>
<td>40.2</td>
<td>56.9</td>
<td>124.2</td>
</tr>
<tr>
<td>$M_I$</td>
<td>8</td>
<td>20</td>
<td>38</td>
<td>69</td>
<td>112</td>
<td>166</td>
<td>238</td>
<td>329</td>
<td>438</td>
<td>570</td>
<td>726</td>
</tr>
<tr>
<td>$\pi = (2, 1)/\Lambda$</td>
<td>1.5</td>
<td>5.0</td>
<td>4.8</td>
<td>5.9</td>
<td>8.9</td>
<td>11.1</td>
<td>12.5</td>
<td>20.3</td>
<td>21.2</td>
<td>31.5</td>
<td>61.2</td>
</tr>
</tbody>
</table>

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

The usual weak formulation results after spatial integration by parts

$$\int_{Q} u_t \phi \, d\vec{x} + \int_{\partial Q} \left( \vec{f}(u) \cdot \vec{n} \right) \phi \, ds - \int_{Q} \vec{f}(u) \cdot \nabla \phi \, d\vec{x} = 0. \tag{15}$$

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 $u_h$’. Based on the nodal elements from Section 2 we shortly review the nodal DG variant in the following subsection.
<table>
<thead>
<tr>
<th></th>
<th>pentahedron/ρ</th>
<th>pyramid/ρ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$M_I$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\pi = (0,0)/\Lambda$</td>
<td>1.7</td>
<td>3.7</td>
</tr>
<tr>
<td>$M_I$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\pi = (1,0)/\Lambda$</td>
<td>1.7</td>
<td>3.7</td>
</tr>
<tr>
<td>$M_I$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\pi = (0,1)/\Lambda$</td>
<td>1.7</td>
<td>3.7</td>
</tr>
<tr>
<td>$M_I$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\pi = (1,1)/\Lambda$</td>
<td>1.7</td>
<td>3.7</td>
</tr>
</tbody>
</table>

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

### 3.1.1 The nodal DG scheme

Recently, Hesthaven and Warburton [22] introduced the nodal DG scheme with a more recent and general discussion given in [23]. In this formulation, the approximation $u_h$ is represented using the nodal basis functions $\{\psi_j\}_{j=1,\ldots,M_I}$

$$u_h(\bar{x}, t) = \sum_{j=1}^{M_I} \bar{u}_j(t) \psi_j(\bar{x}) \text{ for } \bar{x} \in Q,$$

which are also used as testfunctions. Concentrating on the first volume integral, we get

$$\int_Q f_1(u_h) \frac{\partial \psi_j}{\partial x_1}(\bar{x}) \, d\bar{x} \approx K_{1,N}^{1,1},$$

11
where \((\tilde{f}_1)_j := f_1(u_h(\xi_j))\) and the nodal stiffness matrix is given by

\[
K^{1,N} := \int_Q \frac{\partial \psi}{\partial x_1}(\tilde{x}) \psi^T(\tilde{x}) \, d\tilde{x} = V^{-T} \int_Q \frac{\partial \phi}{\partial x_1}(\tilde{x}) \phi^T(\tilde{x}) \, d\tilde{x} V^{-1} =: V^{-T} K^{1,M} V^{-1}. \tag{18}
\]

The modal stiffness matrix \(K^{1,M}\) is computed exactly with Gauss integration and stored in an initial phase of the simulation. The evaluation of the surface integrals can be done in a similar manner, we refer to [23] for a complete description of the nodal DG scheme. In the 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

\[
\int_Q f_1(u_h) \frac{\partial \phi_j}{\partial x_1}(\tilde{x}) \, d\tilde{x} \approx \sum_{j=1}^{(p+1)d} f_1(u_h(\tilde{\chi}_j)) \frac{\partial \phi_j}{\partial x_1}(\tilde{\chi}_j) \omega_j, \tag{19}
\]

where \(\omega_j\) and \(\tilde{\chi}_j\) are the Gauss weights and Gauss positions, respectively. If we consider a hexahedron with a \(p = 5\) approximation, we get \((p+1)d = 216\) evaluation points with this strategy for the first volume integral, and, following section 2.3 with \(\pi = (0,0)\), \(M_I = 80\) evaluation points for the nodal DG scheme.

Reducing the accuracy of quadrature and relying nodal products when computing the nonlinear fluxes naturally introduces an error, known in spectral methods as aliasing [15]. 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 filter (see [23]). In the present work, however, we have not found any need for this additional stabilization for any of the examples presented later.

3.1.2 The modal DG scheme with nodal integration

Using the above presented ideas from the nodal framework we now briefly discuss how to enhance the efficiency of modal DG implementations. We again focus on the simplicity of the approximation of the first volume integral of the modal formulation (19). But instead of using Gauss integration, we use the nodal elements of Section 2 to build a high order interpolation of the flux \(f_1\)

\[
f_1(u_h) \approx \psi^T \tilde{\phi}_1, \tag{20}
\]

where again \((\tilde{f}_1)_j := f_1(u_h(\tilde{\xi}_j)).\) Inserting this into the volume integral yields

\[
\int_Q f_1(u_h) \frac{\partial \phi_j}{\partial x_1}(\tilde{x}) \, d\tilde{x} \approx K^{1} \tilde{f}_1, \tag{21}
\]
where the general stiffness matrix is given by

$$K^1 := \int_Q \frac{\partial \psi}{\partial x_1}(\vec{x}) \psi^T(\vec{x}) d\vec{x} = K^{1,M} V^{-1}. \quad (22)$$

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 with the 'traditional' modal implementation, we now have an approach with approximately the computational complexity of the nodal DG scheme, but with only \( N \) modal degrees of freedom per grid cell, compared to \( M_I \) nodal degrees of freedom.

### 3.1.3 Nodal DG VS modal DG

Let us briefly consider a comparison of the pure nodal DG approach and the modal DG with nodal integration. We start with a reformulation of the modal DG scheme

$$\dot{\hat{\mathbf{u}}} = - \int_{\partial Q} g_n(u, u^+) \phi ds + \int_{Q} \vec{f}(u) \cdot \vec{\nabla} \phi d\vec{x} =: \mathbf{r}(u, \varphi), \quad (23)$$

where we used the fact, that the modal mass matrix is the identity. For the nodal DG scheme we get analogously

$$M \dot{\tilde{\mathbf{u}}} = \mathbf{r}(u, \psi) = V^{-T} \mathbf{r}(u, \varphi), \quad (24)$$

where we used the linearity of the residual with respect to the testfunctions and we introduced the nodal mass matrix

$$M := \int_Q \psi \psi^T d\vec{x} = V^{-T} \int_Q \phi \phi^T d\vec{x} V^{-1} = V^{-T} V^{-1}. \quad (25)$$

If we use the fact that product \( V^T V^{-T} \) is the \( N \times N \) identity, equation (24) can be simplified to

$$V^{-T} V^{-1} \dot{\tilde{\mathbf{u}}} = V^{-T} \mathbf{r}(u, \varphi), \quad \text{and} \quad \dot{\tilde{\mathbf{u}}} = V^{-1} \mathbf{r}(u, \varphi). \quad (26)$$

According to the inverse transformation (7) the term \( V^{-1} \tilde{\mathbf{u}} \) simplifies to \( \hat{\mathbf{u}}^N \) and thus

$$\hat{\mathbf{u}}^N = \mathbf{r}(u, \varphi). \quad (27)$$

If we make now the assumption that we use the same discrete initial condition at \( t = 0 \) for the modal and the nodal scheme and if we remind that the evaluations of the integrals are the same because of the nodal integration, we get that

$$\hat{\mathbf{u}} = \hat{\mathbf{u}}^N \quad (28)$$

and hence the equality of both approximate solutions for all \( t \).
3.2 The fully discrete form

The base of the space-time expansion discontinuous Galerkin (STE-DG) scheme is the semi-discrete discontinuous Galerkin formulation (15). For the fully discrete scheme we simply integrate from $t_n$ to $t_{n+1}$ in time

\[
\int_Q u_{n+1} \varphi_j \, d\vec{x} = \int_Q u_n \varphi_j \, d\vec{x} - \int_{t_n}^{t_{n+1}} \int_Q g_\mu(u^\pm, u) \varphi_j \, d\vec{s} + \int_Q \vec{f}(u) \cdot \nabla \varphi_j \, d\vec{x} \, dt,
\]

for $j = 1, ..., N$. The integrals in this formulation are approximated with the above presented nodal technique in space and a standard Gaussian quadrature rule in time. To get a scheme with $O(\Delta t^{p+1})$ in time, $\frac{p+2}{2}$ time Gauss points are needed.

To evaluate the right-hand side we thus need a high order accurate approximation of the solution $u$ at all the evaluation points in $(\vec{x}, t) \in Q \times [t_n; t_{n+1})$. In the space-time expansion approach, this is done by an explicit predictor which is based on the Cauchy problem

Find $u^{CK} = u^{CK}(\vec{x}, t)$, with

\[
\begin{align*}
    u^{CK}_t + \vec{\nabla} \cdot \vec{f}(u^{CK}) &= 0 \quad \forall (\vec{x}, t) \in \mathbb{R}^d \times \mathbb{R}^+, \\
    u^{CK}(\vec{x}, t = t_n) &= u_n(\vec{x}) \quad \forall \vec{x} \in \mathbb{R}^d,
\end{align*}
\]

where $u_n(\vec{x})$ is the DG polynomial at time $t = t_n$ extended into $\mathbb{R}^d$. As for $u^{CK}$ no boundary effects from the neighboring grid cells are included, the approximation is stable up to $t_{n+1} = t_n + \Delta t$, yielding a Runge-Kutta DG type time step restriction for this explicit scheme, see [28] for more details. For the general non-linear case, the exact solution of problem (30) is quite cumbersome and in many cases impractical. However, we notice that we only need a high order accurate approximation of the Cauchy problem, as this still guarantees high order accuracy of our DG scheme.

An efficient way to get such a high order approximation in space and time is to use a space-time Taylor series about the bary-center $\vec{x}_B$ of the grid cell $Q$ at the ‘old’ time level $t_n$

\[
u^{CK}(\vec{x}, t) = \sum_{j=0}^{p} \frac{1}{j!} \left( (\vec{x} - \vec{x}_B) \cdot \vec{\nabla} + (t - t_n) \frac{\partial}{\partial t} \right)^j u|_{(\vec{x}_B, t_n)}
\]

and to apply the Cauchy-Kovalevskaya (CK) procedure to replace time derivatives of the solution by pure space derivatives using the governing equation. For instance, we immediately get the first time derivative of $u$, according to (13), as

\[
u_t(\vec{x}_B, t_n) = -\vec{\nabla} \cdot \vec{f}(u_n(\vec{x}_B)).
\]

Thus to compute the right-hand side of (29) we simply evaluate the space-time expansion (31) at the spatial interpolation points and the time Gauss points. Beside
the nodal integration presented in section 3.1.2, the explicit space-time approach gives the possibility to introduce some additional reformulations which additionally enhance the efficiency of the computations. These are described in the following.

- **factorization in space-time**: we focus on the first volume integral and get

\[
\int_{t_n}^{t_{n+1}} \int_Q f_1 \partial_{x_1} \varphi_j d\vec{x} dt \approx \sum_{j=1}^{p+2} K_1 f_1(\tau_j) \omega_j,
\]

where \( \tau_j \) and \( \omega_j \) denote the Gauss points and Gauss weights in time, respectively. This means, that we can first integrate the nonlinear nodal flux values in time and multiply with the stiffness matrix afterwards. Thus independent of the order in time, we only have one matrix-vector multiplication. In a Runge Kutta time integration the number of matrix-vector multiplications is at least \( p+1 \) for time order \( p+1 \).

- **time accurate local time stepping**: the locality of the DG semi-discretization and the possibility to evaluate the space-time expansion at arbitrary times \( t \) gives the STE-DG scheme the property of time consistent high order accurate local time-stepping. This means that every grid cell runs with its own time step, adopted to the local stability restriction. For more detailed information we refer to Lörcher et al. [28],[27] and Gassner et al. [13].

- **divergence based volume integral**: the base of this modification is to use the strong form of the semi discrete DG formulation, which results when back-integrating formulation (15) by parts

\[
\int_Q u_i \varphi_j d\vec{x} + \int_Q \vec{\nabla} \cdot \vec{f}(u) \varphi_j d\vec{x} = \int_{\partial Q} \left( \vec{f}(u) \cdot \vec{n} - g_{\vec{n}}(u^+, u) \right) \varphi_j ds,
\]

\[j = 1, ..., N.\] (34)

A straightforward way to use the nodal based integration for the volume integral is to interpolate the flux vector \( \vec{f} \) and multiply it with the adjoint general stiffness matrices, resulting in \( d \) flux interpolations and \( d \) matrix vector multiplications to compute the volume integrals. For the proposed modification, we do not interpolate the flux vector \( \vec{f} \) but the divergence of the flux vector \( \vec{\nabla} \cdot \vec{f} \). Thus the volume integral of the flux is treated as a source, where the interpolation vector of the flux divergence is multiplied with

\[
\int_Q \varphi(\vec{x}) \varphi^T(\vec{x}) d\vec{x} = \int_Q \varphi(\vec{x}) \varphi^T(\vec{x}) d\vec{x} V^{-1} = V^{-1}.
\]

Using now the informations of the Cauchy-Kovalevskaya procedure, we simply get

\[
\vec{\nabla} \cdot \vec{f}(\vec{\xi}_i, \tau_j) \approx -u_i^{CK}(\vec{\xi}_i, \tau_j).
\] (36)
Summing up, we only have to use the time derivative of our space-time expansion \( \tilde{u} \) to evaluate the divergence of the flux and then multiply this interpolation vector with the inverse Vandermonde matrix. To get a better understanding of the STE idea, we consider the pure Nodal DG case. Starting from (34)+(36) and inserting nodal trial and test functions, we get

\[
M \tilde{u}_t(t) = M \tilde{u}^{CK}(t) + \tilde{g}(t),
\]

\[
\tilde{u}_t(t) = \tilde{u}^{CK}(t) + M^{-1} \tilde{g}(t),
\]

(37)

where \( \tilde{g} \) denote the surface terms with the numerical fluxes and the boundary conditions. Integrating in time and using the consistency assumption (30) at \( t = t_n \), yields

\[
\tilde{u}(t_{n+1}) - \tilde{u}(t_n) = \tilde{u}^{CK}(t_{n+1}) - \tilde{u}^{CK}(t_n) + M^{-1} \int_{t_n}^{t_{n+1}} \tilde{g}(t) dt,
\]

\[
\tilde{u}(t_{n+1}) = \tilde{u}^{CK}(t_{n+1}) + M^{-1} \int_{t_n}^{t_{n+1}} \tilde{g}(t) dt,
\]

(38)

emphasizing the predictor nature of the Cauchy-Kovalevskaya solution and the corrector function of the surface terms. We mention, that for linear problems the two different ways of approximating the volume integrals are numerically equal, but comparing computational effort (time and memory), the modification is more efficient. For non-linear problems the two approaches are numerically not equivalent. Due to the nonlinearity, it is not guaranteed that for the constant testfunction \( \varphi_1 \), the surface terms with the flux evaluated with values from inside the element cancel exactly with the divergence volume integral. Thus this formulation is not conservative in the general case. However the trick to get a conservative formulation is very easy, as we just force conservativity by simple using the weak formulation (23) for the first equation \( j = 1 \) and the strong formulation (34) with the proposed volume integral modifications for the remaining equations

\[
\int_Q u_i \varphi_1 \, dx = - \int_{\partial Q} g_\tilde{n}(u^+, u) \varphi_1 \, ds,
\]

\[
\int_Q u_i \varphi_j \, dx + \int_{\partial Q} \nabla \cdot \tilde{f}(u) \varphi_j \, ds = \int_{\partial Q} \left( \tilde{f}(u) \cdot \tilde{n} - g_\tilde{n}(u^+, u) \right) \varphi_j \, ds,
\]

(39)

\( j = 2, \ldots, N. \)

4 Computational examples and validations

In the following we shall present a number of examples of increasing complexity to thoroughly validate the developed scheme.
4.1 Linear Wave Propagation

In this subsection the divergence based volume integral and the temporal accuracy of the STE-DG scheme with local time stepping [28] is investigated. We use the linearized Euler equations (LEE) as a model problem for linear wave propagation

\[ U_t + \nabla \cdot \tilde{F}(U) = 0, \quad (40) \]

with the vector of the conservative variables \( U = (\rho', u', v', p')^T \) and the LEE fluxes \( \tilde{F} := (F_1, F_2, F_3)^T := (A_1 U, A_2 U, A_3 U)^T \) with the jacobi matrices

\[
A_1 = \begin{pmatrix}
    u_0 & \rho_0 & 0 & 0 \\
    0 & u_0 & 0 & 0 \\
    0 & 0 & u_0 & 0 \\
    0 & \kappa p_0 & 0 & u_0 \\
\end{pmatrix},
A_2 = \begin{pmatrix}
    v_0 & \rho_0 & 0 & 0 \\
    0 & v_0 & 0 & 0 \\
    0 & 0 & v_0 & 0 \\
    0 & 0 & \kappa p_0 & v_0 \\
\end{pmatrix},
A_3 = \begin{pmatrix}
    w_0 & 0 & 0 & \rho_0 \\
    0 & w_0 & 0 & 0 \\
    0 & 0 & w_0 & 0 \\
    0 & 0 & \kappa p_0 & w_0 \\
\end{pmatrix},
\]

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

\[ U = RW, \quad (42) \]

with \( W = \hat{W} \sin(k \cdot \bar{x}) \) and the Eigenvector matrix

\[
R = \begin{pmatrix}
    n_1 & n_2 & n_3 & \frac{\rho_1}{2c_0} & \frac{\rho_0}{2c_0} \\
    0 & -n_3 & n_2 & \frac{n_1}{2} & -\frac{n_1}{2} \\
    n_3 & 0 & -n_1 & \frac{n_2}{2} & -\frac{n_2}{2} \\
    -n_2 & n_1 & 0 & \frac{n_1}{2} & -\frac{n_1}{2} \\
    0 & 0 & 0 & \frac{\rho_1}{2c_0} & \frac{\rho_0}{2c_0} \\
\end{pmatrix}, \quad (43) \]

with \( c_0 = \sqrt{\frac{\rho_0}{\kappa \rho_0}} \). We choose the pertubation of the characteristic variable vector \( \hat{W} = (0, 0, 0, 0, 0, 0, 0.001, 0.0)^T \), the normal vector of the wave \( \vec{n} = (1.0, 0.0, 0.0)^T \), the wave number vector \( \vec{k} = (\pi, 0.0, 0.0)^T \) and the background flow \( U_0 = (1.0, 0.0, 0.0, 0.0, \frac{1}{c_0})^T \) with \( \kappa = 1.4 \), resulting in \( c_0 = 1.0 \). The computational domain \( \Omega := [0.0; 2.0]^3 \) is split into 8 regular subdomains \( \Omega_i = \bar{x}_i + [0.0; 1.0]^3 \),
\( i = 1, ..., 8 \) with
\[
\vec{x}_1 := (0.0, 0.0, 0.0)^T, \quad \vec{x}_2 := (1.0, 0.0, 0.0)^T, \quad \vec{x}_3 := (0.0, 1.0, 0.0)^T, \\
\vec{x}_4 := (0.0, 0.0, 1.0)^T, \quad \vec{x}_5 := (1.0, 1.0, 0.0)^T, \quad \vec{x}_6 := (0.0, 1.0, 1.0)^T, \\
\vec{x}_7 := (1.0, 0.0, 1.0)^T, \quad \vec{x}_8 := (1.0, 1.0, 1.0)^T. \tag{44}
\]

For our \( h \)-refinement tests we introduce the parameter \( n \geq 1 \). For a given \( n \), we first split every subdomain \( \Omega_i \) 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 figure 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.

![Fig. 4. Visualization of the different hybrid meshes.](image) 

In table 4 the experimental order of convergence for this test case is plotted 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 \( \tilde{p} \), when we changed the grid points via the parameters \( \pi \) or when we used the modification of the volume integral based on the divergence of the flux. 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 figure 5 the $L_2$ error norm of the pressure $p'$ for polynomial order $p = 1$ up to $p = 8$ with $t_{end} = 20.0$. For the next configurations the hexahedral base grid was further split into tetrahedra, prisms or pyramids, according to figure 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. Please note that for the first four configuration the timesteps 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 due to the different grid cell types and their different inspheres, the scheme runs in local time stepping modus. It is interesting to compare

![Graph](image)

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

for this test case the performances of the different grid cells. First of all comparing the number of grid cells in the different configurations and thus the number of DOF, figure 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. Furthermore if we compare the CPU time for the whole calculation, the hexahedral discretization succeeds again, as they allow 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 that this trends even hold true 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 is investigated. Based on the results from the linear test case, we consider in this subsection the non-linear Euler equations

\[
U_t + \vec{\nabla} \cdot \vec{F}(U) = 0,
\]

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 \( \vec{F} := (F_1, F_2, F_3)^T \):

\[
F_l(U) = \begin{pmatrix}
\rho v_l \\
\rho v_1 v_l + \delta_u p \\
\rho v_2 v_l + \delta_2 p \\
\rho v_3 v_l + \delta_3 p \\
\rho e v_l + \rho v_l
\end{pmatrix}, \quad l = 1, 2, 3.
\]

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

\[
p = \rho RT = (\kappa - 1) \rho (e - \frac{1}{2} \vec{v} \cdot \vec{v}), \quad \text{and} \quad e = \frac{1}{2} \vec{v} \cdot \vec{v} + c_v T.
\]

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 [24]

\[
\vec{r}(\vec{x}, t) = \vec{r}_{\text{vortex}} \times (\vec{x} - \vec{x}_0 - \vec{v}_0 \cdot t),
\]

\[
\delta \vec{v} = \frac{v_{\text{max}}}{2\pi} \exp \left( 1 - \frac{\left| \vec{r} \right|^2}{2} \right),
\]

\[
\vec{v}(\vec{x}, t) = \vec{v}_0 + \delta \vec{v} \cdot \vec{r},
\]

\[
\frac{T}{T_0} = 1 - \frac{\kappa - 1}{2} \left( \frac{\delta \vec{v}}{c_0} \right)^2,
\]

\[
\rho(\vec{x}, t) = \rho_0 \left( \frac{T}{T_0} \right)^{\frac{1}{\kappa}},
\]

\[
p(\vec{x}, t) = p_0 \left( \frac{T}{T_0} \right)^{\frac{\kappa}{\kappa - 1}}.
\]
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 $(\rho_0, \vec{v}_0^T, p_0) = (1, 1, 1, 1, \frac{1}{4})$, $\kappa = 1.4$, 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 $v_{\text{max}} = 0.1$, the halfwidth of the vortex $r_0 = 1.0$ and the endtime of the simulation $t_{\text{end}} = 4.0$. The computational domain $\Omega := [0, 5.0]^3$ with exact boundary conditions prescribed. The solution to this problem at time $t = 2.0$ with $6^3 \, p = 5$ hexahedra is shown in figure 6. The results of tests with $p = 6$ trial functions with different parameters $\pi$ and/or different interpolation orders $\tilde{p}$ are listed in tables 5-8.

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

4.3 Compressible Navier-Stokes equations

The three dimensional unsteady compressible Navier-Stokes equations with a source term reads as

$$U_t + \nabla \cdot \vec{F}(U) - \nabla \cdot \vec{F}^0(U, \nabla U) = S,$$  \hspace{1cm} (49)
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 $\vec{F}^v := (F_1^v, F_2^v, F_3^v)^T$:

$$F_l^v(U, \nabla U) = \begin{pmatrix} 0 \\ \tau_{1l} \\ \tau_{2l} \\ \tau_{3l} \\ \tau_{lj} v_j - q_j \end{pmatrix}, \quad l = 1, 2, 3. \quad (50)$$

<table>
<thead>
<tr>
<th>Interpolation order ($\tilde{p}$) and $\pi$</th>
<th>Nb Int points</th>
<th>$L_2(\rho)$</th>
<th>CPU time/EU</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tilde{p} = 6$, $\pi = (0, 0)$</td>
<td>117</td>
<td>$1,9654E - 05$</td>
<td>100%</td>
</tr>
<tr>
<td>$\tilde{p} = 6$, $\pi = (1, 0)$</td>
<td>124</td>
<td>$1,7455E - 05$</td>
<td>107%</td>
</tr>
<tr>
<td>$\tilde{p} = 6$, $\pi = (0, 1)$</td>
<td>147</td>
<td>$1,8112E - 05$</td>
<td>120%</td>
</tr>
<tr>
<td>$\tilde{p} = 6$, $\pi = (1, 1)$</td>
<td>154</td>
<td>$1,6055E - 05$</td>
<td>121%</td>
</tr>
<tr>
<td>$\tilde{p} = 6$, $\pi = (2, 0)$</td>
<td>136</td>
<td>$1,7399E - 05$</td>
<td>110%</td>
</tr>
<tr>
<td>$\tilde{p} = 6$, $\pi = (2, 1)$</td>
<td>166</td>
<td>$1,5832E - 05$</td>
<td>125%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (0, 0)$</td>
<td>160</td>
<td>$1,7586E - 05$</td>
<td>127%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (0, 0)$</td>
<td>214</td>
<td>$1,6336E - 05$</td>
<td>154%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (1, 0)$</td>
<td>160</td>
<td>$1,6055E - 05$</td>
<td>121%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (1, 0)$</td>
<td>214</td>
<td>$1,6336E - 05$</td>
<td>154%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (0, 1)$</td>
<td>103</td>
<td>$1,7078E - 04$</td>
<td>107%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (0, 1)$</td>
<td>187</td>
<td>$1,5537E - 04$</td>
<td>181%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (1, 1)$</td>
<td>103</td>
<td>$1,7078E - 04$</td>
<td>107%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (1, 0)$</td>
<td>187</td>
<td>$1,5537E - 04$</td>
<td>181%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (1, 1)$</td>
<td>111</td>
<td>$1,6332E - 04$</td>
<td>110%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (1, 1)$</td>
<td>111</td>
<td>$1,6332E - 04$</td>
<td>110%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (2, 0)$</td>
<td>136</td>
<td>$1,7399E - 05$</td>
<td>110%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (2, 0)$</td>
<td>136</td>
<td>$1,7399E - 05$</td>
<td>110%</td>
</tr>
<tr>
<td>$\tilde{p} = 7$, $\pi = (2, 1)$</td>
<td>166</td>
<td>$1,5832E - 05$</td>
<td>125%</td>
</tr>
<tr>
<td>$\tilde{p} = 8$, $\pi = (2, 1)$</td>
<td>166</td>
<td>$1,5832E - 05$</td>
<td>125%</td>
</tr>
<tr>
<td>Gauss Legendre points</td>
<td>637</td>
<td>$1,4665E - 05$</td>
<td>403%</td>
</tr>
</tbody>
</table>

Table 5
Results for different types of integration points for $p = 6$ hexahedra. The domain $\Omega$ is subdivided into 8 hexahedra.

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

Table 6
Results for different types of integration points for $p = 6$ pyramids. The domain $\Omega$ is subdivided into 6 pyramids.
Interpolation order ($\tilde{p}$) and $\pi$ | 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 7
Results for different types of integration points for $p = 6$ prisms. The domain $\Omega$ is subdivided into 8 hexahedra which are further subdivided into 2 prisms, yielding 16 grid cells.

Interpolation order ($\tilde{p}$) and $\pi$ | 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%

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

The viscous stress tensor is given by

$$\tau := \mu (\vec{\nabla} \vec{v} + (\vec{\nabla} \vec{v})^T - \frac{2}{3} (\vec{\nabla} \cdot \vec{v}) I),$$  \hspace{1cm} (51)

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

$$\vec{q} := -k \vec{\nabla} T, \quad \text{with} \quad k = \frac{c_p \mu}{Pr}.$$  \hspace{1cm} (52)

Here, the viscosity coefficient $\mu$ and the Prandtl number $Pr$ depend on the fluid, and are supposed to be constant for this test. If we choose

$$S = \alpha \begin{pmatrix} 
\cos(\beta) (d k - \omega) \\
\cos(\beta) A + \sin(2\beta) \alpha k (\kappa - 1) \\
\cos(\beta) A + \sin(2\beta) \alpha k (\kappa - 1) \\
\cos(\beta) A + \sin(2\beta) \alpha k (\kappa - 1) \\
\cos(\beta) B + \sin(2\beta) \alpha (d k \kappa - \omega) + \sin(\beta) \left( \frac{d k^2 \mu}{Pr} \right) 
\end{pmatrix},$$  \hspace{1cm} (53)
with \( \beta := k(x_1 + x_2 + x_3) - \omega t, A = -\omega + \frac{k}{d-1} \left( (-1)^{d-1} + \kappa (2d - 1) \right) \) and \( B = \frac{1}{2} \left( (d^2 + \kappa (6 + 3d)) k - 8 \omega \right) \), the analytical solution to (49)+(53) is given by

\[
U = \left( \sin(\beta) \alpha + 2, \sin(\beta) \alpha + 2, \sin(\beta) \alpha + 2, \sin(\beta) \alpha + 2, (\sin(\beta) \alpha + 2)^2 \right)^T.
\]

For our test we choose the coefficients \( \kappa = 1.4, Pr = 0.72, \mu = 0.0001, R = 287.14 \) and \( \alpha = 0.5, \omega = 10.0, k = \pi \) with the dimension of the problem \( d = 3 \).

We solve this problem with the recently developed modal STE-DG scheme for compressible Navier-Stokes equations [13], with the above presented nodal modifications. 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 circumventing 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 [33] and for the approximation of the viscous fluxes the recently developed dGRP flux [14], [13], [27], which can be interpreted as a natural extension of the classic IP flux [30] 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 \( t_{end} = 1.0 \) and periodic boundary conditions. The results indicate that the optimal order of convergence \( EOC = p + 1 \), for \( p \) odd and even, is achieved.

<table>
<thead>
<tr>
<th>n</th>
<th>Nb cells</th>
<th>Nb DOF</th>
<th>( L_2(\rho e) )</th>
<th>EOC</th>
<th>Nb DOF</th>
<th>( L_2(\rho e) )</th>
<th>EOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>168</td>
<td>5.880</td>
<td>6,13E-3</td>
<td>-</td>
<td>9.408</td>
<td>3,80E-3</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>1344</td>
<td>47.040</td>
<td>1,91E-4</td>
<td>5,0</td>
<td>75.264</td>
<td>9,36E-5</td>
<td>5,3</td>
</tr>
<tr>
<td>8</td>
<td>10752</td>
<td>376.320</td>
<td>4,32E-6</td>
<td>5,5</td>
<td>602.112</td>
<td>1,54E-6</td>
<td>5,9</td>
</tr>
<tr>
<td>16</td>
<td>86016</td>
<td>3.010.560</td>
<td>1,22E-7</td>
<td>5,1</td>
<td>4.816.896</td>
<td>2,38E-8</td>
<td>6,0</td>
</tr>
</tbody>
</table>

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

We list the average CPU time per element update (CPU/EU) and per degree of freedom (CPU/EU/DOF) for the 3D compressible Navier-Stokes equations with \( p = 6 \) Ansatz functions (84 polynom coefficients \( \times \) 5 variables = 420 DOF/Element) in table 10. Based on the investigations in subsection 4.2, we chose for every grid cell type the most efficient combination (in terms of accuracy versus cpu time) of the parameters \( \pi \) and the interpolation order \( \tilde{p} \). All CPU times were measured on one processor of a Intel Xeon Dual Core CPU with 2.66GHz. An equivalent measurement for a 6th order compact finite difference scheme with 4th order Runge-Kutta time integration, [2], on the same CPU yields \( \sim 11,2 \mu s. \)
4.3.1 Polygonal meshes

In this section first results for a DG discretization with polygonal meshes are shown. Starting from a triangle mesh the corresponding polygonal dual mesh is constructed, figure 7a. The primal triangle mesh is no longer needed as it is only used to construct the dual mesh. We then use the algorithms presented above to construct the different nodal elements for the polygonal grid cells. Numerical investigations indicate that for a general grid cell the shape dependend 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 of the grid cell types discussed in this work. Figure 7b shows a detailed view of the interpolation points distribution for the $h = 0.025$ mesh and a $p = 3$ interpolation, where we chose the recursion parameter $\pi_{2D} = 3$. The pre-computation of the surface and volume integral matrices is done on sub triangles with standard Gaussian integration. In future works we will investigate the effect of the interpolation point distribution for hybrid meshes as a whole and not only for isolated elements. It seems that in this extrem case where quadrillaterals (4 sides, 3 recursive defined interior point layers) and heptagons (7 sides, 0 recursive defined interior point layers) arise, it would be more natural to fix the number of recursions for the whole mesh and thus generating a globally uniform interpolation point distribution. To validate this discretization the two dimensional

![Fig. 7. Primal and dual mesh ($h = 0.1$) and detailed view of the interpolation grid ($h = 0.025$) with $p = 3$ ($\pi_{2D} = 3$) interpolation.](image-url)
compressible Navier-Stokes equations with a source term are considered, where we used the reduced two dimensional version of the previous example with the same parameters, excepting the parameter $k$ which we changed from $\pi$ to $2\pi$ 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. In table 11 the results of this investigations with $t_{end} = 0.1$ and exact boundary conditions are shown. Allthough with such a definition of the mesh sequence the refinement is not uniformly, the experimental convergence order indicates optimal order of accuracy for this discretization.

<table>
<thead>
<tr>
<th>$h$</th>
<th>Nb cells</th>
<th>Nb DOF</th>
<th>$L_2(\rho e)$</th>
<th>EOC</th>
<th>Nb DOF</th>
<th>$L_2(\rho e)$</th>
<th>EOC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>$p = 3$</td>
<td></td>
<td></td>
<td>$p = 4$</td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td>42</td>
<td>420</td>
<td>$1.28E - 2$</td>
<td>-</td>
<td>630</td>
<td>$1.47E - 3$</td>
<td>-</td>
</tr>
<tr>
<td>0.1</td>
<td>134</td>
<td>1.340</td>
<td>$1.04E - 3$</td>
<td>3.6</td>
<td>2.010</td>
<td>$1.00E - 4$</td>
<td>3.9</td>
</tr>
<tr>
<td>0.05</td>
<td>489</td>
<td>4.890</td>
<td>$4.16E - 5$</td>
<td>4.5</td>
<td>7.335</td>
<td>$1.72E - 6$</td>
<td>5.9</td>
</tr>
<tr>
<td>0.025</td>
<td>1878</td>
<td>18.780</td>
<td>$2.16E - 6$</td>
<td>4.3</td>
<td>28.170</td>
<td>$4.70E - 8$</td>
<td>5.2</td>
</tr>
</tbody>
</table>

Table 11

Experimental order of convergence for $p = 3$ and $p = 4$ with $\pi_{2D} = 3$ and $t_{end} = 0.1$.

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 $\Omega$ extends from $x_1 = 337.0$ to $x_1 = 890.0$ and $x_2 = 0.0$ to $x_2 = 22.35$. We choose subsonic inflow and outflow boundary conditions and at $x_2 = 0.0$ isothermal wall conditions with $T_w = 296.0 \, K$. The initial solution of the computation is obtained from a similarity solution with Mach number $M_\infty = 0.8$ and $T_\infty = 280.0 \, K$. The Reynoldsnumber $Re := u_\infty \delta_1 / \mu(\infty) = 1000$, based on the displacement thickness at the inflow $\delta_1$. Using $\delta_1$ 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 viscosity $\mu$ is modelled using Sutherland’s law

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

with $\mu(\infty) = 1.735 \times 10^{-5} \, \text{kg} \, \text{m} / \text{s}$ and $T_s = 110.4 \, K$.

The inflow at $x_1 = 337.0$ 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 was subdivided in $48 \times 22$ regular quadrilaterals and dicretized with $p = 6$ ($\pi_{2D} = 1$) STE-DG scheme, resulting in 29568 DOF. The endtime of the simulation was set to $t_{end} / T_0 = 37$, where
$T_0 = \frac{2\pi}{\omega_0} \approx 92$, to ensure a periodic solution. To analyse 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 figure 8. For comparison, corresponding results obtained with a 6th order compact finite difference code with $330 \times 150$ grid points and 4th order Runge Kutta time integration [1] are included, showing good agreement. We furthermore plot the amplification rate $\alpha_i$ of the velocity $v_1$ based on the maximal amplitude in figure 8. Again, the result is in good accordance to the reference result [1] and the predictions of linear stability theory.

![Fig. 8. Maximum amplitudes of $v_1$ (left). Amplification rate $\alpha_i$ of $u_1$ based on maximum amplitude (right).](image)

### 4.3.3 Flow past a Sphere at $Re = 300$

We consider in this example a sphere with radius $r = 1$ centered at $\vec{x}_0 = (0, 0, 0)^T$. We 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 $x_1 = -20.0$ to $x_1 = 100.0$ and $x_2, x_3 = \pm 30.0$. The grid consists of $\approx 160,000$ 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 of the grid on a cut plane with $\vec{n}_{\text{plane}} = (0, 1, 0)^T$ in figure 9. For the calculation the $p = 3$ STE-DG scheme was used, resulting in $\approx 3,000,000$ DOF. A contour plot of the velocity magnitude, figure 10, shows that the boundary layer is resolved within 1-2 tetrahedral elements. In figure 11 the structure of the vortices are shown using the $\lambda_2$ vortex detection criterium. We list in table 12 the resulting force coefficients, the corresponding oscillating amplitudes and the Strouhal number $Str$. For comparison results from Tomboulides [32] and Johnson&Patel [25], obtained within an incompressible simulation, are listed as well. In figure 12 we plot the drag coefficient $C_d$ and the lateral force coefficient $C_l$ versus time $t$.  

27
Fig. 9. Visualization of the grid for the sphere example.

Fig. 10. Contour plot of the instantaneous velocity magnitude $|\vec{v}| = 0.0...0.3478$ and pressure $p = 0.688...0.762$.

Fig. 11. Isometric view of $\lambda_2$ Isosurface.

5 Conclusion

Part one of this paper deals with a framework for efficient polynomial interpolation on generally grid cells, i.e. the definition of a nodal interpolation basis. In our framework, for non simplices grid cells the number of nodal basisfunctions
<table>
<thead>
<tr>
<th></th>
<th>$C_d$</th>
<th>$\Delta C_d$</th>
<th>$C_I$</th>
<th>$\Delta C_I$</th>
<th>$Str$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tomboulides [32]</td>
<td>0.673</td>
<td>0.0031</td>
<td>-0.065</td>
<td>0.015</td>
<td>0.135</td>
</tr>
<tr>
<td>Johnson &amp; Patel [25]</td>
<td>0.656</td>
<td>0.0035</td>
<td>-0.069</td>
<td>0.016</td>
<td>0.137</td>
</tr>
</tbody>
</table>

Table 12

Force coefficients and Strouhal number.

![Drag and lateral force coefficient](image)

Fig. 12. Drag and lateral force coefficient.

is higher than the number of modal basis functions. We showed that one way to get a reasonable Van der Monde matrix is to use the singular value decomposition framework to build a least squares inverse. The properties of these Van der Monde matrices (and the corresponding interpolation) solely depend 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. We therefor introduced 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 succesive 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 the Nodal DG method a mixed modal DG scheme with the computational complexity of a Nodal DG scheme was constructed. It was also shown, that the Nodal DG scheme and the modal DG scheme with nodal based integration produce numerically the same results. As an example the nodal based integration was combined with the recently developed space-time expansion discontinuous Galerkin scheme yielding an efficient high order discretization on arbitrary unstructured grids for unsteady flow problems.
References


