V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal,14-17 June 2010

AN EXPLICIT SPACE-TIME ADAPTIVE DISCONTINUOUS GALERKIN SCHEME

C. Altmann, G. Gassner and C.-D. Munz

University of Stuttgart,
Pfaffenwaldring 21, 70569 Stuttgart, Germany, www.iag.uni-stuttgart.de
e-mail: {altmann,gassner,munz}@iag.uni-stuttgart.de

Key words: discontinuous Galerkin, time accurate local time-stepping, arbitrary order, space-time expansion, nodal, modal

Abstract. *The explicit discontinuous Galerkin scheme presented in this paper may be considered as a predictor corrector approach. The predictor is based on a solution in the small which takes into account the time evolution within the grid cell only. The locality of this approach allows the introduction of a time-consistent local time stepping in a natural way. This functionality can drastically improve the scheme's efficiency when calculating highly time dependent problems. The necessary solution in the small can be obtained e.g. by a Taylor series expansion as well as special Runge-Kutta schemes. This numerical discretization is applied to several test cases, such as a two dimensional simulation of a mixing layer and a three dimensional simulation of the flow past a sphere. In addition, results of the parallel performance of the proposed scheme are given.*

1 INTRODUCTION

For the simulation of unsteady flow problems on large scale parallel hardware, an explicit discontinuous Galerkin (DG) scheme is a good choice. A major disadvantage of an explicit DG scheme may be the global time step restriction to establish stability. This restriction depends on the space increment, on the degree of the polynomial approximation and on wave speeds for advection terms or on diffusion coefficients for diffusion terms and may cause inefficiency. One reason can be the size of the grid cells. A small grid cell will cause small time steps. If we need the small grid cells for the spatial resolution, then a small time step is necessary to capture the time development. But, the small grid cell may also be imposed by the grid generator due to a complex geometry, e.g., at corners. Another class of problems occurs, if the time evolution of the process is only weakly connected with the fastest wave speed. In this case the time step is kept small due to the fast wave speed, while the time evolution is connected with a smaller speed. A typical example is low Mach number flow, where the flow transport may fully determine the time evolution of the process, while the fast acoustic waves determine the time step in an explicit approximation. In the following, we will describe a method to overcome these problems and focus ourselves to unsteady problems and explicit DG schemes.

2 THE DISCONTINUOUS GALERKIN SCHEME

2.1 Weak formulation

Without the loss of generality, let us consider the scalar advection-diffusion equation

$$u(\vec{x})_t + \vec{\nabla} \cdot \left(\vec{f^a}(u(\vec{x})) - \mu \nabla u(\vec{x}) \right) \Big) = 0, \tag{1}$$

where $\vec{f^a}$ denotes the advection flux. We will only show the key ingredients for deriving the weak formulation. The reader will find a complete step-by-step explanation for equation systems and handling of viscosity in Gassner et al.[5] and Lörcher et al.[7].

We will next introduce the DG framework by subdividing our domain Ω into non-overlapping spatial grid cells Q_i. Our numerical DG solution is then defined as

$$u\Big|_{Q_i} \approx u_i(\vec{x},t) := \sum_{j=1}^{N} \hat{u}_i^j(t)\phi_j(\vec{x}) \text{ for all } \vec{x} \in Q_i. \tag{2}$$

The $\hat{u}_i^j$ are the time dependent degrees of freedom (DOF) and the basis functions $\{\phi_j(x)\}_{j=1,\dots,N}$ span the space of polynomials with degree $\leq P$ over the spatial grid cell Q_i. To keep things simple, we will suppress in the following the index i in the notation of our numerical solution u^i and ϕ^i.

To derive the weak formulation, we first multiply the equation (1) by a test function $\phi = \phi(\vec{x})$, integrate over an arbitrary space-time cell $Q_i^n := Q_i \times [t^n, t^{n+1}]$ and perform a

spatial integration by parts:

$$\int\limits_{Q_i^n} u_t \phi \, d\vec{x}dt + \int\limits_{\partial Q_i^n} \left(\vec{f}^a - \mu \vec{\nabla} u \right) \cdot \vec{n} \, \phi \, dsdt - \int\limits_{Q_i^n} \left(\vec{f}^a - \mu \vec{\nabla} u \right) \cdot \vec{\nabla} \phi \, d\vec{x}dt = 0. \tag{3}$$

For the diffusive part of our equation which contains second order derivatives, we proceed with a second integration by parts of the viscous volume integral part in (3):

$$\int\limits_{Q_i^n} \mu \vec{\nabla} u \cdot \vec{\nabla} \phi \, d\vec{x}dt = \int\limits_{Q_i^n} \vec{\nabla} u \cdot \mu \vec{\nabla} \phi \, d\vec{x}dt = \int\limits_{\partial Q_i^n} u \mu \vec{\nabla} \phi \cdot \vec{n} dsdt - \int\limits_{Q_i^n} u \nabla \cdot \left(\mu \vec{\nabla} \phi \right) \, d\vec{x}dt. \tag{4}$$

The integration by parts is used three times, forth and back, under the assumption that the volume integral is calculated from data inside Ω_i^n and only the surface integral covers the interaction between the grid cells. The objective is to lift a jump at the boundary between the functional values from the interior and the interface state into the discrete variational formulation, see Gassner et al.[5] in addition.

We then end up with the weak formulation of the advection-diffusion equation

$$\begin{aligned} &\int\limits_{Q_i^n} u_t \phi \, d\vec{x}dt - \int\limits_{Q_i^n} \vec{f}^a \cdot \vec{\nabla} \phi \, d\vec{x}dt + \int\limits_{Q_i^n} \mu \vec{\nabla} u \cdot \vec{\nabla} \phi \, d\vec{x}dt + \\ &\int\limits_{\partial Q_i^n} \vec{g^a} \cdot \vec{n} \phi \, dsdt - \int\limits_{\partial Q_i^n} \vec{g^d} \cdot \vec{n} \phi \, dsdt + \int\limits_{\partial Q_i^n} g^s \left[\vec{\nabla} \phi \cdot \vec{n} \right]^- dsdt = 0 \, , \end{aligned} \tag{5}$$

where the test function ϕ runs through all the basis functions. Here, $[.]^-$ denotes the internal evaluation at the boundary of Q_i^n, the term $\vec{g^a}$ denotes the numerical advection flux, the term $\vec{g^d}$ denotes the numerical diffusion flux and $g^s := \mu u - [\mu u]^-$ denotes the additional scalar diffusion flux. u denotes the numerical solution in grid cell Q_i without any restrictions on the equation system type. Because our approximation u is discontinuous across element interfaces, we have to introduce numerical flux functions to guarantee both the stability and consistency of the discretization. We use in this work the HLLC flux, see Toro[8].

For nonlinear flux functions, the space-time integrals in (5) have to be computed in an approximate way. While this could be done using Gaussian quadrature formulae in space and time, we need to find a way to get approximate values at the space-time Gauss points. This has to be done in an explicit way, since we are interested in an explicit scheme.

2.2 Space-time prediction

To get a high order scheme in space and time, we need a high order accurate approximation of the values at the time Gauss points. We therefore construct a local space-time approximation $v = v(\vec{x}, t)$ for $\vec{x} \in Q_i$ and $t \in [t^n, t^{n+1}]$. We search the space-time solution

for the following Cauchy problem

$$\begin{aligned} &v_t + \vec{\nabla} \cdot \left(f^a(v) - f^d(v, \vec{\nabla} v) \right) = 0, \\ &v(\vec{x}, t = 0) = u_*^n(\vec{x}), \forall \vec{x} \in \Re^d, \end{aligned} \tag{6}$$

where u_*^n is the DG polynomial in the grid cell Q_i at time level t^n extended in $\Re^d$. We note that the calculation of the exact solution for this problem is due to the non-linearity impractically costly. However, considering the 'limited' accuracy of the resulting DG discretization, only a high order accurate approximation of this problem is needed. An efficient way to construct such an approximation is to use a Taylor expansion in space and time with appropriate order, see Lörcher et al.[6] and Gassner et al.[5]. The derivatives of the space-time expansion are approximated using the so-called Cauchy-Kowalevsky procedure, see Dumbser and Munz[3] for an efficient way of implementing a Cauchy-Kowalevsky procedure for non-linear problems. In practice, every technique for the solution of an initial value problem could be chosen, as long as the result is an analytical space-time solution. We can also construct the approximative solution using a continuous extension Runge-Kutta Galerkin (CERKG) method, see Gassner[4] for more information.

2.3 Local time stepping

This space-time setting allows to perform a high order time accurate and fully conservative explicit local time stepping, where each cell within the computational domain runs with its own optimal time step. Via Taylor expansion or the CERK method, necessary values at the cell boundaries can be reconstructed in time. It is therefore ensured that no information that has to be exchanged with the neighbors will be lost during the time updates. In the following, the methodology of the local time stepping is briefly outlined.

A sequence of four time steps with three adjacent grid cells in Figure 1 starting from a common time level $t^0 = 0$ holds as a starting point:

After the determination of the local time steps, which are assumed to be different in our example due to the local stability restriction, a predictive approximate solution in the space-time cells $Q_i \times [t_i^0, t_i^1]$, in our example for $i = 1, 2, 3$ is calculated. These space-time polynomials are stored. We note that after this step the DOF $\vec{\hat{u}}_i^0$ at the time level t_i^0 are not needed any longer and may be overwritten. First, the volume integrals are calculated for each element Q_i. They rely only on the local space-time polynomials. The contribution of these terms are added to the DOF of the old time level. We call these values $\vec{\hat{u}}_i^*$. Next, the surface flux contributions involving neighboring grid cells have to be considered.

The local time stepping algorithm relies on the following evolve condition. The update of the DOF can only be completed, if

$$t_i^{n+1} \leq \min \left\{ t_j^{n+1} \right\}, \forall j : Q_j \cap Q_i \neq \emptyset \tag{7}$$

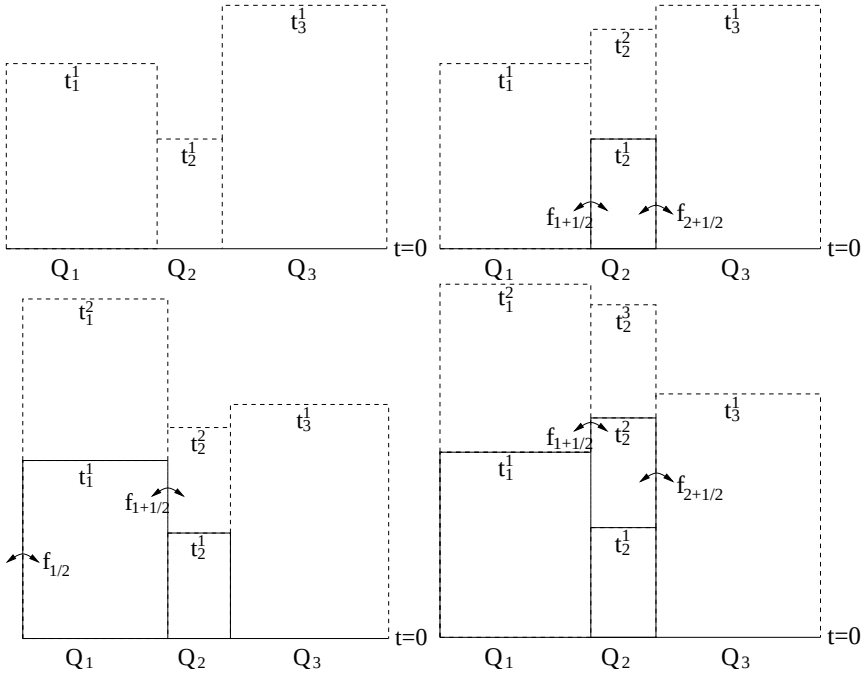

Figure 1: Sequence of steps 1-4 of a computation with 3 different elements and local time stepping

is satisfied. This condition guarantees that all the data for the interface fluxes are available. In our example, the first grid cell satisfying this condition is Q_2. So Q_2 can now be evolved to t_2^1. To do so, the flux contributions at the right and left cell interface have to be computed and its contribution is then added to the local $\hat{\vec{u}}_2^*$. The flux integrals are calculated from $t = t_2^0$ to $t = t_2^1$ at the right interface $\partial Q_{2+\frac{1}{2}}$ and the left interface $\partial Q_{2-\frac{1}{2}}$. The arguments for the numerical flux functions at the time Gaussian points are obtained from the left and right space-time polynomials.

In order to keep this calculation exactly conservative as well as efficient, the flux contributions computed for the evolution of Q_2 are added simultaneously to the corresponding neighbors $\hat{\vec{u}}_1^*$ and $\hat{\vec{u}}_3^*$ with the minus sign. Then the update for Q_2 is completed and the DOF at the new time level t_2^1 are known. We can now start the procedure again: a new space-time polynomial is constructed in $Q_2 \times [t_2^1, t_2^2]$ and the volume integral contribution is added to the local DOF $\hat{\vec{u}}_2^1$, now named by $\hat{\vec{u}}_2^*$.

If boundary values at the left are given, now Q_1 satisfies the evolve condition and can be advanced to t_1^1. As before, the volume integral contribution is already added. But in this case, also a part of the flux contributions has already been added to the $\hat{\vec{u}}_1^*$ during the previous evolution of Q_2. Thus, only the missing flux contributions, which are sketched in the lower left corner of Figure 1, have to be added to the $\hat{\vec{u}}_1^*$ in order to get the $\hat{\vec{u}}_1^1$. Namely, on the interface $\partial Q_{1+\frac{1}{2}}$, the flux integral has to be computed with a quadrature formula from t_2^1 to t_1^1. As before, the flux integral computed on this shared interface is

not only added to $\vec{\hat{u}}_1^*$, but also to $\vec{\hat{u}}_2^*$. The time interval, for which the flux contribution at the interface shared by an element Q_i and an adjacent element Q_j has to be computed when evolving $\vec{\hat{u}}_i^*$ to $\vec{\hat{u}}_i^{n+1}$, is generally

$$[t_{ij}^\star, t_i^{n+1}] = [max(t_i^n, t_j^n), t_i^{n+1}]. \tag{8}$$

In this manner, the algorithm continues by searching for elements satisfying the evolve condition (7). At each time, the interface fluxes are defined uniquely for both adjacent elements, making the scheme exactly conservative. The presented local time stepping algorithm minimizes the total number of time steps for a computation with fixed end time. However, it is not difficult to introduce some common global time levels as needed for example at the end of the computation. This procedure has absolutely no influence on the accuracy of the underlying numerical scheme.

3 NUMERICAL EXAMPLES

We will show applicability and performance of the numerical scheme for different test cases in the following.

3.1 Parallel performance

For testing the scaling capabilities of our proposed method, we chose an example where a perfect load balance was to achieve. We were using the so-called manufactured solution technique for the three-dimensional compressible instationary Navier-Stokes equations: When forcing an exact solution, this results in an inhomogeneous source term on the right hand side of the Navier-Stokes equations that is put into the code. The problem was set up with periodic boundaries so that the boundary communication will not differ from the inter-processor communications. The size of the computational problem was increased exactly the same way as the number of processors for calculation was increased. This way, we kept a constant load in computation as well as in communication. Table 3.1 shows the good scale-up efficiency for up to 4080 processors with a constant load per processor. The efficiency when calculating on N processors is calculated as the calculation time on one processor divided by the time needed for a calculation on N processors.

Number of processors	1	1000	2197	4080
Efficiency [%]	-	99.1	97.8	98.8

Table 1: Scale-up efficiency of the HALO code.

3.2 Two-dimensional mixing layer

For this two-dimensional example, we adopted the situation simulated by Colonius, Lele and Moin[2] and Babucke, Kloker and Rist[1]. The initial condition for the mixing

layer is the laminar solution to the steady compressible two-dimensional boundary layer equations. The Mach number of the upper and the lower stream are $Ma_1 = 0.5$ and $Ma_2 = 0.25$, respectively. The Reynolds number $Re = \rho_1 u_1 \delta / \mu = 500$ is based on the vorticity thickness on the inflow

$$\delta(x_0) := \left(\frac{\Delta u}{|\partial u / \partial y|_{max}} \right)_{x_0}, \tag{9}$$

which is also used to normalize the length scales. The computational domain starts at $x_0 = 30$, which yields $\delta(x_0) = 1$, and extends to $x = 900$ and $y = \pm 200$. Velocities are normalized with u_1 and all other quantities by their corresponding values in the upper stream. The flow is forced at the inflow with eigenfunctions found from the spatial viscous linear stability theory, see Babucke et al.[1] and references therein for more details. The disturbances are composed of the fundamental frequency $\omega_0 = 0.6293$ and the three subharmonics. The amplitudes of the eigenfunctions are normalized by their maximum value of u_1 and then scaled by the amplitude factor 0.002. In accordance to the simulation of Colonius et al.[2], the phase shift is $\Delta\Theta = -0.028$ for the first, $\Delta\Theta = 0.141$ for the second and $\Delta\Theta = 0.391$ radians for the third disturbance. The shown $N = 6$ (7^{th} order) calculation had a grid resolution in the region $[30; 320] \times [-12; 12]$ of 27 grid cells in y-direction, corresponding to $\Delta y \in [0.65; 2.75]$ and 230 grid cells in x-direction corresponding to $\Delta x \in [0.75; 4]$, resulting in ≈ 180000 DOF.

The Reynolds number is given by

$$Re = \frac{\max(\rho_0)\, \max(u_0)\, \delta}{\mu} = \frac{1 \cdot 0.5 \cdot \delta}{\mu} = 500 \tag{10}$$

with

$$\delta := \frac{\max(u_0) - \min(u_0)}{\max(\frac{\partial u_0}{\partial y})} = 1, \tag{11}$$

while the viscosity coefficient μ has the value 0.001. We plot in Figure 2 the vorticity contour levels at $\frac{t}{T} = 68$ and included the corresponding results of Colonius et al.[2] for comparison. The numerical results show good agreement with the reference data.

In the left plot of Figure 3, the maximum amplitude of u_2 with respect to y as a function of x is shown. In the first part of the domain, where the magnitude of the amplitudes are small, the growth is nearly exponentially, showing good agreement with linear stability theory. We furthermore plotted the amplification rate α_i of the velocity u_2 for the ω_0 mode, based on the maximum amplitude shown in the left picture. The result is in good accordance with linear stability theory.

3.3 Three-dimensional freestream injector

This calculation especially targets shock capturing and p-adaption capabilities of the proposed numerical scheme, as we simulate a $Ma = 1.4$, $Re = 30000$ injection. The

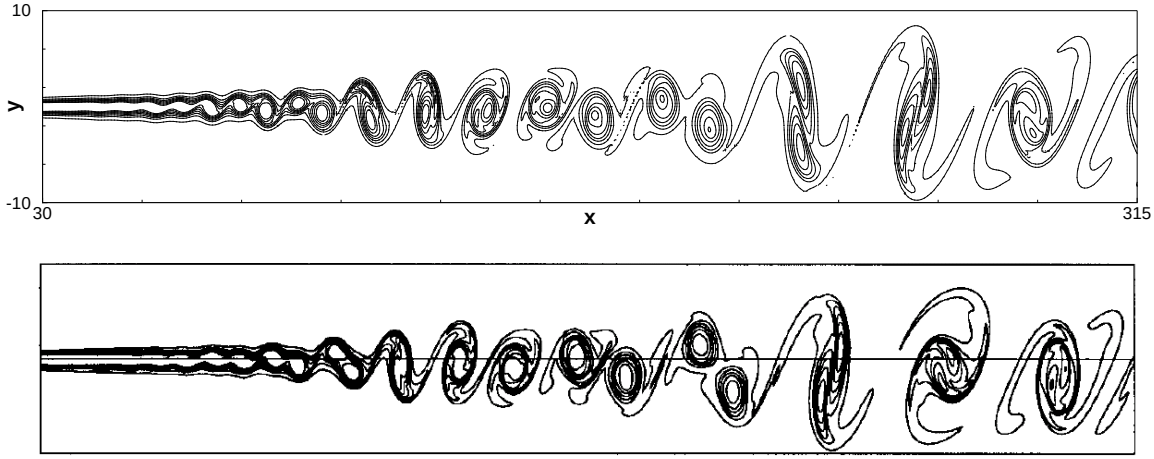

Figure 2: Mixing Layer. Vorticity contours in near-field mixing region at $\frac{t}{T} = 68$ from contour level -0.26 to level 0.02 with increment 0.04 are shown. The results from Colonius et al.[2] are shown below.

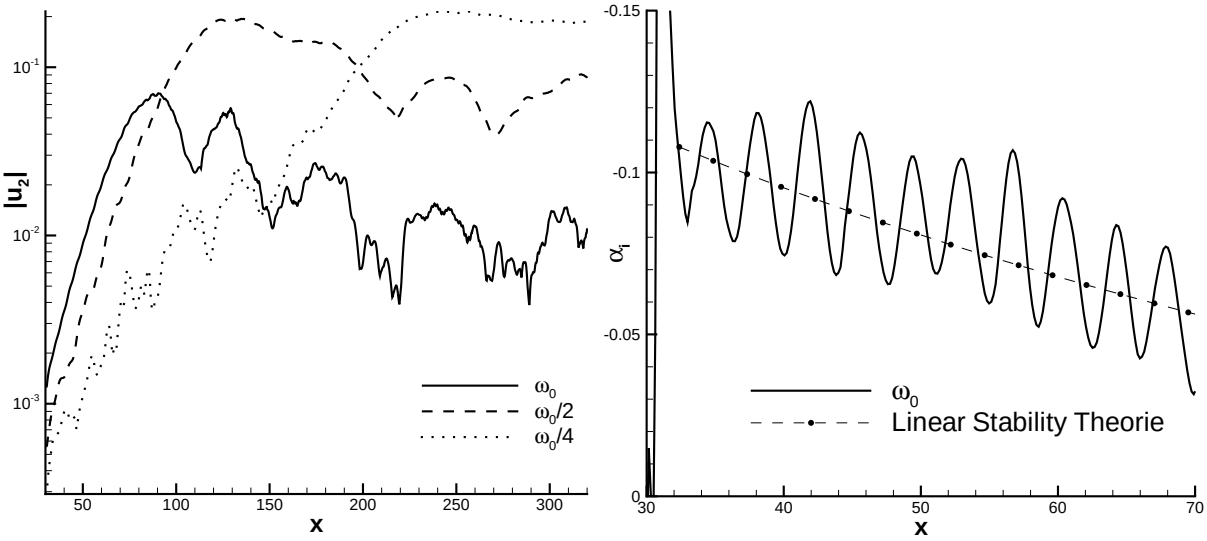

Figure 3: Maximum amplitudes of u_2 (left). Amplification rate α_i of u_2 based on maximum amplitude (right).

injection nozzle is designed according to devices used for gas injection devices of the automotive industry. This problem also contains complex curved geometries that are challenging for high order schemes. The calculation aims at an aeroacoustic simulation of the instationary injection process, including the start up of the process. Results can be validated with calculations performed with other codes, in-house as well as in industry. Preliminary two-dimensional calculations already provided an insight regarding the necessary grid resolutions and shock capturing strategies as well as a comparison with a

commercial CFD. Figure 4 shows density contours from $\rho \in [0.9; 1.3]\, kg/m^3$ at $t = 1ms$ in comparison with a commercial CFD tool. Our proposed DG scheme was 14 times faster, producing even better results. The geometry of the three-dimensional injector itself is

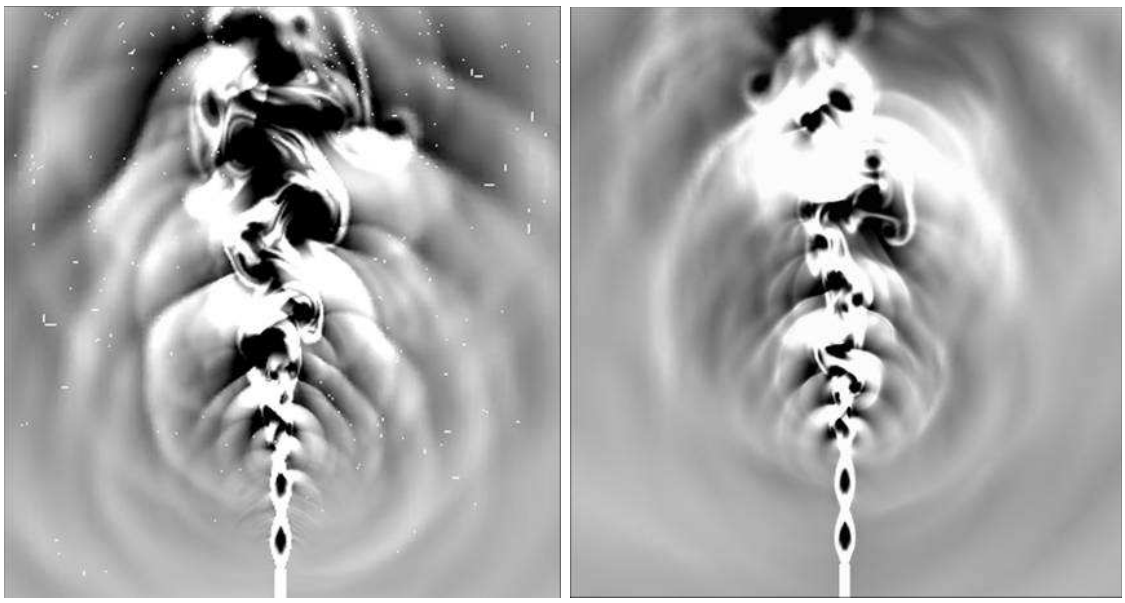

Figure 4: Density distribution, proposed DG scheme 2^{nd} order (left) and commercial CFD tool (right).

rather complex, see Figure 5, consisting of four kidney-shaped injection nozzles within the cylindric injector and is assembled with unstructured hexahedra, allowing hanging nodes and polygons at connection surfaces. This problem was calculated on a grid with

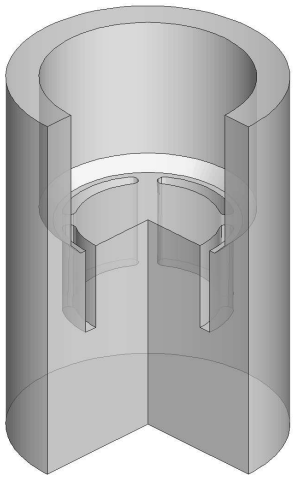

Figure 5: Geometry of the three-dimensional freestream injector.

over 16 million DG DOF (4 million of hexahedral elements) on 500 up to 1000 processors. Figure 6 shows a two-dimensional slice plane of the calculation (density distribution and velocity streamlines) together with an isosurface plot of the density that brings out the development of the flow. To illustrate the flow development, several streamlines have been added. The picture presents an early phase of simulation where the injection process was just started.

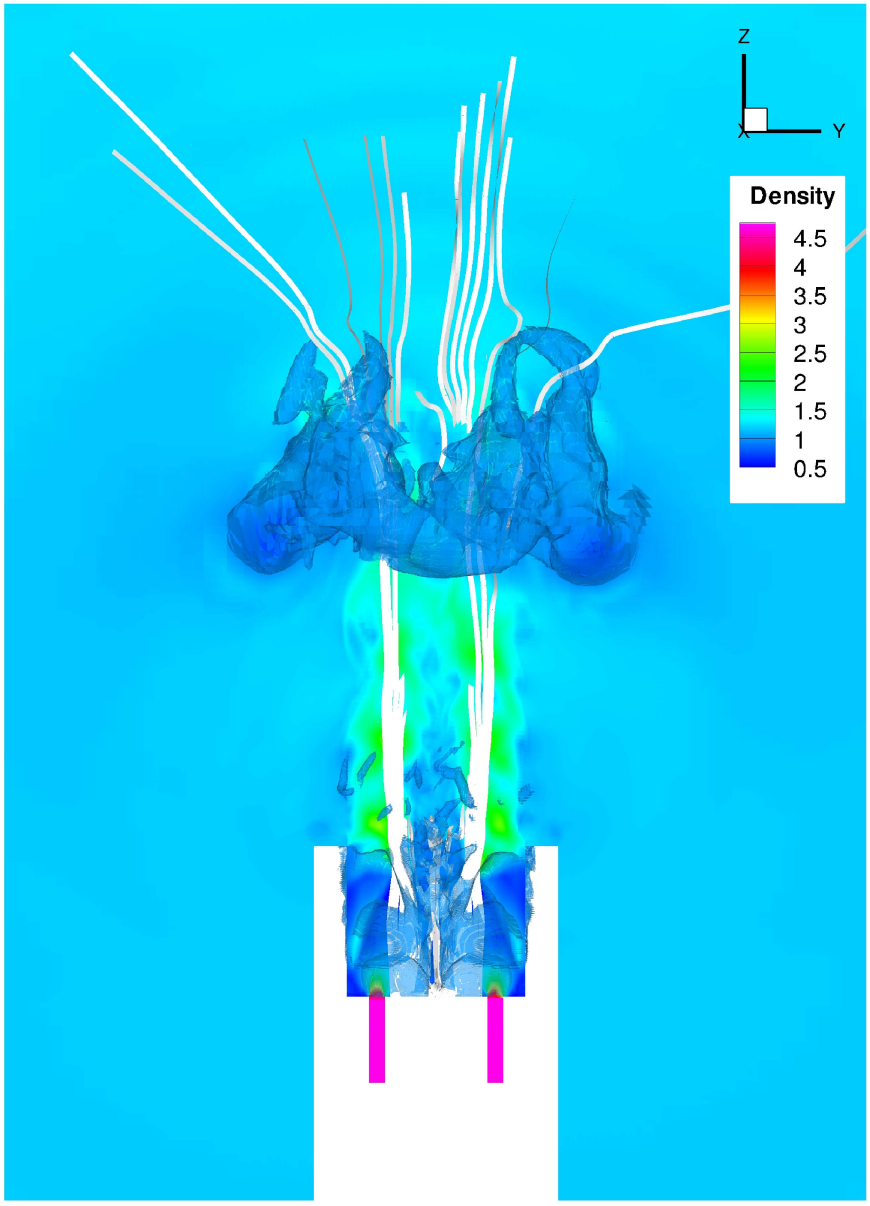

Figure 6: Density distribution and velocity streamlines of the three-dimensional freestream injector.

3.4 Three-dimensional laminar flow past a sphere

The laminar time periodic flow around a sphere was set up to test the scheme's ability to handle unstructured hybrid grids and a p-adaption mechanism in three space dimensions. We were solving the unsteady compressible Navier-Stokes equations with a free-stream Mach number of $Ma_\infty = 0.3$ and a Reynolds number, based on the injector diameter, of $Re = 300$. The problem was discretized with a block-unstructured grid consisting of prisms for the boundary layer, tetrahedra and hexahedra. Figure 7 shows the different grid blocks and dimensions of the computational domain. P-adaption was arranged to vary the polynomial degree p between 1 and 5 where each grid cell was allowed to adapt every 500 time steps.

To demonstrate the calculation results, Figure 9 shows a three-dimensional view of the instantaneous vortex measure λ_2 for this calculation. The color indicates the velocity magnitude. Here, one can easily see that the very large cells at the end of the wake cannot provide the necessary resolution and are therefore producing large inter cell jumps of the solution. Finally, Figure 8 on the next page shows the distribution of the local polynomial degree p at end time $t_{end} = 1000$.

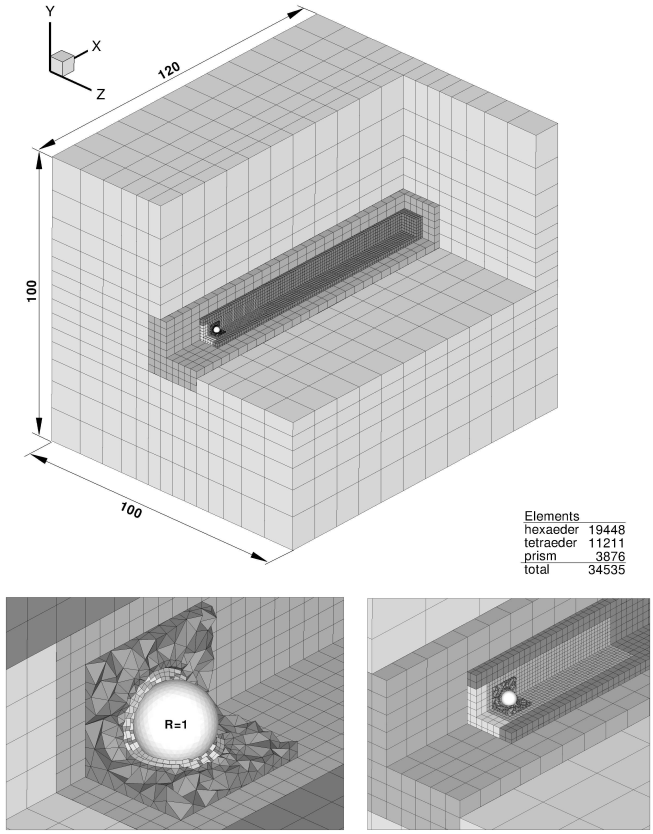

Figure 7: Hybrid block-unstructured mesh for the sphere calculation.

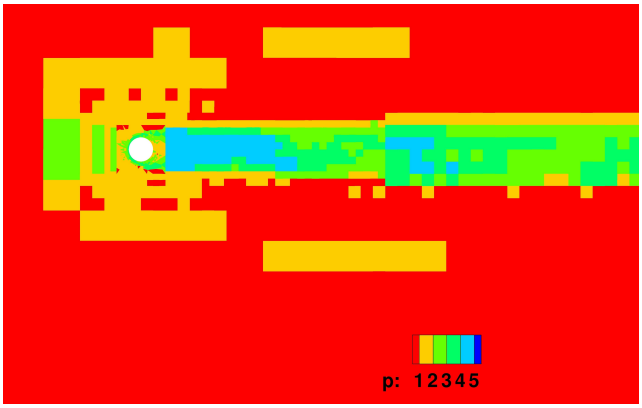

Figure 8: Distribution of the local polynomial degree at end time of the sphere calculation.

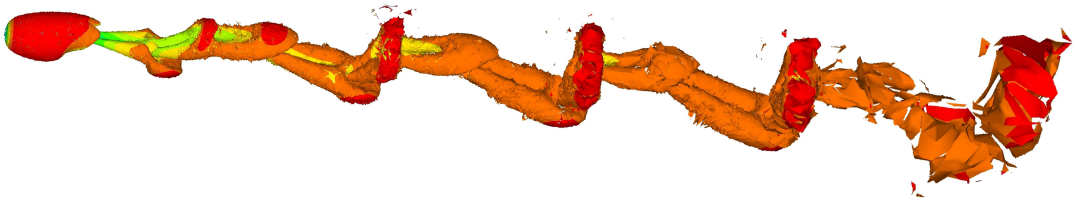

Figure 9: Instantaneous λ_2 isosurface of the sphere.

4 CONCLUSION

In this paper we have shown an explicit discontinuous Galerkin discretization in two and three space dimensions. The possibility of using local time steps will strongly increase the efficiency for all problems which need different resolution in different regions of the computational domain and in which the time step restriction for stability strongly varies as a local function of element size, wave speed, viscosity and polynomial degree. Two- and three-dimensional calculations were performed. They showed up that the scheme is so far working as expected. With the integration of the VMS-LES technique, the scheme is ready to aim at massively parallel large scale LES calculations.

1. A. Babucke, M. J. Kloker, and U. Rist. DNS of a plane mixing layer for the investigation of sound generation mechanisms. *Computers and Fluids*, 37(4):360 – 368, 2008. Turbulent Flow and Noise Generation.

2. T. Colonius, S. K. Lele, and P. Moin. Sound generation in a mixing layer. *J. Fluid Mech.*, 330:375–409, 1997.

3. M. Dumbser and C.-D. Munz. Building blocks for arbitrary high order discontinuous Galerkin schemes. *Journal of Scientific Computing*, 27(1-3):215–230, June 2006.

4. G. Gassner. *Discontinuous Galerkin Methods for the Unsteady Compressible Navier–Stokes Equations.* Dissertation, University of Stuttgart, 2009.

5. G. Gassner, F. Lörcher, and 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):260–286, 2008.

6. F. Lörcher, G. Gassner, and 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):175–199, 2007.

7. F. Lörcher, G. Gassner, and C.-D. Munz. An explicit discontinuous Galerkin scheme with local time-stepping for general unsteady diffusion equations. *J. Comput. Phys.*, 227(11):5649–5670, 2008.

8. E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics.* Springer, second edition, 1999.