Discontinuous Galerkin for High Performance Computational Fluid Dynamics

Andrea Beck, Thomas Bolemann, Hannes Frank, Florian Hindenlang, Marc Staudenmaier, Gregor Gassner, and Claus-Dieter Munz

Abstract In this report we present selected simulations performed on the HLRS clusters. Our simulation framework is based on the Discontinuous Galerkin method and consists of four different codes, each of which is developed with a distinct focus. All of those codes are written with a special emphasis on (MPI) based high performance computing. In this report, we show results of our newest code, FLEXI, which is tailored solely towards high performance computing of three dimensional advection dominated problems. Currently, we are interested in extending our direct numerical simulation framework to large eddy simulations and present computations and comparisons of our framework for the flow past a cylinder and the flow past a wing. Furthermore, first results on a MPI based multi-physics extension of our framework is presented. All simulations are typically performed on hundreds and thousands of CPU cores.

1 Introduction

The central goal of our research is the development of high order discretization schemes for a wide range of continuum mechanic problems with a special emphasis on fluid dynamics. Therein, the main research focus lies on the class of Discontinuous Galerkin (DG) schemes. The in-house simulation framework consists of four different Discontinuous Galerkin based codes with different features, such as structured/unstructured grids, non-conforming grids (h-adaptation),

A. Beck (✉) · T. Bolemann · H. Frank · F. Hindenlang · M. Staudenmaier · G. Gassner · Claus-Dieter Munz
Institute of Aerodynamics and Gasdynamics, Universität Stuttgart, Pfaffenwaldring 21, 70569 Stuttgart, Germany
e-mail: beck@iag.uni-stuttgart.de; bolemann@iag.uni-stuttgart.de; frank@iag.uni-stuttgart.de; hindenlang@iag.uni-stuttgart.de; staudenmaier@iag.uni-stuttgart.de; gassner@iag.uni-stuttgart.de; munz@iag.uni-stuttgart.de

W.E. Nagel et al. (eds.), *High Performance Computing in Science and Engineering '13*, DOI 10.1007/978-3-319-02165-2_20,

non-conforming approximations spaces (p-adaptation), high order grids (curved) for approximation of complex geometries, modal and nodal hybrid finite elements and spectral elements with either Legendre-Gauss or Legendre-Gauss-Lobatto nodes. The time discretization is an important aspect in our research and plays a major role in the computing performance of the resulting method. The simulation framework includes standard explicit integrators such as Runge-Kutta, a time accurate local time stepping scheme developed in-house and an implicit time discretization based on implicit Runge-Kutta methods. The general layout of the framework outsources all aspects of a specific physical problem to be solved (e.g. fluid dynamics) in an encapsulated module separated from the main code by clearly defined interfaces. Thus by exchanging this physical problem definition module, the framework is able to solve various partial differential equations such as the compressible Navier-Stokes equations (fluid dynamics), linearized Euler equations (aeroacoustics), Maxwells equations (electrodynamics) and Magnetohydrodynamics equations (Plasma simulation). One of the major foci in the group is the simulation of unsteady compressible turbulence in the context of Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS). Due to the occurrence of multiple spatial and temporal scales in such problems and the resulting high demand in resolution for both, space and time, a high performance computing framework is mandatory. Each new code iteration is based and improved on findings and insights learned during the development of its predecessor. Our newest code iteration called FLEXI (unstructured successor of the structured code STRUKTI) is build from ground up with high performance computing in mind and is currently our main tool to simulate three dimensional PDE based problems.

2 Description of Methods and Algorithms

Discontinuous Galerkin (DG) schemes may be considered a combination of finite volume (FV) and finite element (FE) schemes. While the approximate solution is a continuous polynomial in every grid cell, discontinuities at the grid cell interfaces are allowed which enables the resolution of strong gradients. The jumps on the cell interfaces are resolved by Riemann solver techniques, already well-known from the finite volume community. Due to their interior grid cell resolution with high order polynomials, the DG schemes can use coarser grids. The main advantage of DG schemes compared to other high order schemes (Finite Differences, Reconstructed FV) is that the high order accuracy is preserved even on distorted and irregular grids. The formal order of the scheme can be set by a parameter, namely the polynomial degree of freedom N. Additionally to the formal order of convergence, the polynomial degree N has a strong influence on the dispersion and dissipation behavior. The artificial dissipation of the method is generated via the numerical flux function which is typically based on Riemann solver technology borrowed from the FV community. It can be shown that the overall dissipation in the resolved spectrum decreases with increasing polynomial degree N. Furthermore,

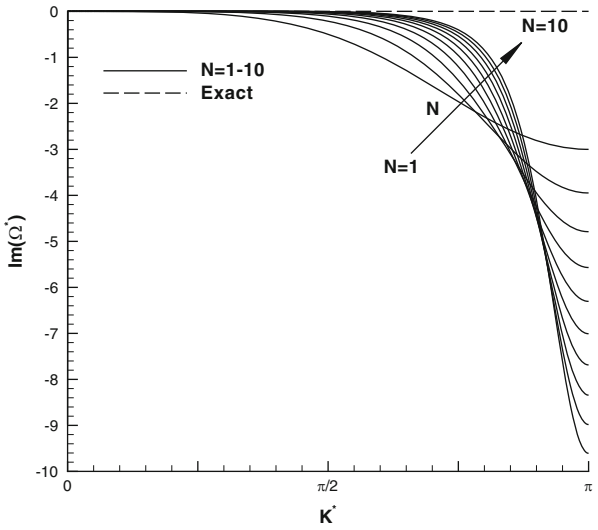

Fig. 1 Dissipation behavior of the DG-derivative operator for different polynomial degrees N

the dissipation behavior for large N acts like a cut-off filter at a certain wavelength, as can be seen in Fig. 1. We have low dissipation in the resolved spectrum and high dissipation in the underresolved part. This is the ideal situation for implicit large eddy simulation (iLES) of turbulent flow.

The low dissipation of high order DG methods also comes with a price: we have a higher sensitivity with respect to aliasing errors which may even lead to aliasing instabilities. A cure for those instabilities lies in the proper evaluation of all spatial integrals [6]. By choosing a sufficient accurate quadrature rule, which also accounts for the non-linearity of the flux function, stability is re-established, also in case of severe underresolution which is typical for large eddy simulations. In conclusion, we use a high order accurate Discontinuous Galerkin discretization with sufficient number of integration points to get a stable and accurate iLES type approach. We use the computing power provided by HLRS to investigate the performance and accuracy of this approach.

In the following subsections, our simulation codes are detailed. All of our codes are developed in FOTRAN 90 with MPI based parallelization.

2.1 High Order DGSEM Solver STRUKTI

A very efficient variant of a Discontinuous Galerkin formulation is the Discontinuous Galerkin spectral element method (DGSEM). This special variation of the DG-method is based on a nodal tensor-product basis with collocated

integration and interpolation points on hexahedral elements, allowing for very efficient dimension-by-dimension element-wise operations.

An easy-to-use structured code (STRUKTI) was set up to test the performance of this method, especially for large scale calculations. As a time integration method, a five stage fourth order accurate explicit Runge-Kutta method is implemented and used for all test cases.

2.2 High Order DGSEM Solver FLEXI

To enable the efficient simulation of complex geometries, a second DGSEM based solver was developed. Sharing the same numerical discretization as STRUKTI, FLEXI is tailored to handle unstructured and even non-conforming hexahedra meshes. A base tool for grid pre-processing was developed. This program allows us to process grid files from different commercial grid generators and translate them into readable FLEXI meshes. Furthermore, a module for curved grid generation and for non-conforming grid connection is included in this tool [5]. As FLEXI shares the same efficient discretization as STRUKTI, the performance of both codes is comparable as in a high order method, the effort of managing the grid is negligible. The difference lies in the parallelization of both codes. FLEXI uses domain partition based on space filling curves, whereas STRUKTI is optimized for structured meshes. Benchmarking of STRUKTI and FLEXI and improvements of FLEXI's parallelization is an ongoing important task in the group.

However, the performance results we already have for FLEXI on the HLRS CRAY-XE6 cluster are at least promising and show great strong scalability of the framework. Figure 2 shows the strong parallel scaling of FLEXI up to 1,024 processors for different polynomial degrees N and a fixed mesh of 1,024 grid cells. This means that we are able to perform strong scaling up to the limit of domain decomposition based MPI, namely with only one grid cell on a processor left.

In DG, the load on the processor is not only determined by the number of grid cells, but also likewise by the polynomial degree N. In the left part of Fig. 2 the influence of the polynomial degree on the strong scaling capabilities is investigated. By choosing a polynomial degree of $N = 8$, we get a super-linear strong scaling of over 100 %. With only one or two grid cells on a processor left, the load but also the memory resources are so small, that strong caching effects can be observed. The right part of Fig. 2 shows investigations of the influence of the load per processor for a fixed number of processors (64 cores in this case) on the PID. The performance index (PID) is a very good measure to compare different simulation setups: it is the computational time needed to update one degree of freedom for one time step, and is computed from the total core-h, the total number of timesteps and degrees of freedom

$$\mathrm{PID} = \frac{\text{Wall-clock-time \#cores}}{\text{\#DOF \#Timesteps}}.$$

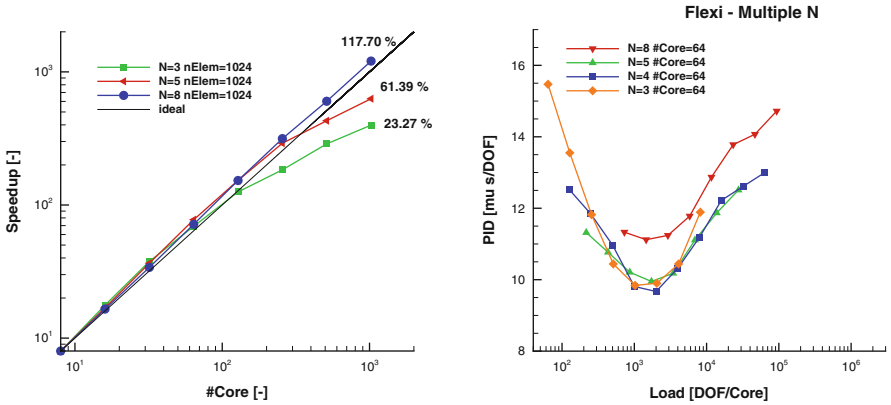

Fig. 2 Strong scaling results for discretizations with constant mesh size and different polynomial degree N. *Left figure* shows the total runtime speedup and right shows the behavior of the specific CPU time index PID

It helps to compare the performance of the different runs. For example, a perfect strong scaling would result in a constant performance index. However, the right part of Fig. 2 shows that there is a strong influence of the local processor load on the PID and that the PID drastically decreases when reducing the load on a processor although the number of processors is constant. This holds until a certain minimum load per processor (2000 DOF/processor) is reached. For even lower values communication overhead causes an increase of the PID. Our experience shows that the minimum value of load/processor depends on the HPC architecture, with 2000 DOF/processor (the lower this value, the better the architecture) being a very good value. When performing a strong scaling investigation where we reach the limits of MPI (one grid cell on a processor), the load on a processor is in the range of hundreds of DOF, thus the caching effects augment the increasing communication overhead resulting in a very good strong scaling capability. If choosing high order polynomials, i.e. $N = 7$, the load on a processor is in the range of the limit of positive caching effects resulting in a real super-linear speed up of the computation as can be seen in Fig. 2.

3 Flow Past a Cylinder at Reynolds number $Re = 3{,}900$

In this section we use a standard LES benchmark example, namely the flow past a cylinder with Reynolds number $Re = 3{,}900$, as a test case to validate our framework. There are many results in literature available, e.g. Kravchenko and Moin used a B-spline based spectral element method with an explicit Smagorinsky model [7] with a resolution varying in-between 1–2 mill DOF, Blackburn and Schmidt

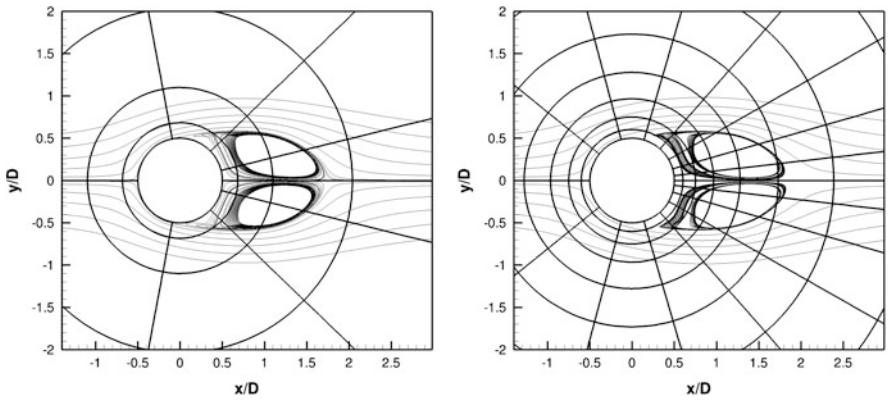

Fig. 3 Zoomed view of the computational grid with mean streamlines of the resulting simulation

used a continuous spectral element method with a dynamic Smagorinsky model [1] with about 1.5 mill DOF; Meyer, Hickel and Adams a finite volume based implicit LES method called ALDM with 6 mill DOF resolution [8]. To compare our results achieved with a compressible solver to the literature results which are all based on simulating the incompressible Navier-Stokes equations, we choose a low Mach number of $M = 0.1$.

For our investigations, we choose two different setups: a setup with a coarse grid of $8 \times 6 \times 6$ grid cells and polynomial degree $N = 11$ resulting in about 500k DOF and a setup with a medium grid of $16 \times 12 \times 12$ grid cells and polynomial degree $N = 7$ resulting in about 1.2 mill DOF. A zoomed view on the computational grid is plotted in Fig. 3. All computations are dealiased with approximately $2\,N$ quadrature points.

After the initial transient phase from uniform flow conditions to the development of the steady vortex shedding, turbulent statistics where gathered over 144 shedding cycles. The $N = 7$ computations were run on 2,304 processors on the Cray XE6, which means that only one element was located on a processors (limit of MPI scaling). The wall-clock time for this run was about 3 h. For the coarse grid calculation with $N = 11$, only 288 processors could be used due to the MPI limit of one grid cell on a processor. The wall-clock time for this run was about 17 h.

3.1 Results

In the first part of the results section, integral quantities are compared to results available in literature. As listed in Table 1, there is a certain spread of the values and no clear reference value can be determined. Both our computations, the computation

Table 1 Comparison of current results with simulations from literature

Author	C_{pBase}	Str	C_D	L_r/D
Kravchenko and Moin	−0.94	0.21	1.04	1.35
Blackburn and Schmidt	−0.93	0.218	1.01	1.63
Meyer and Hickel	−0.92	0.21	1.05	1.38
Current: $N = 11$	−1.0	0.212	1.09	1.26
Current: $N = 7$	−0.93	0.208	1.04	1.37

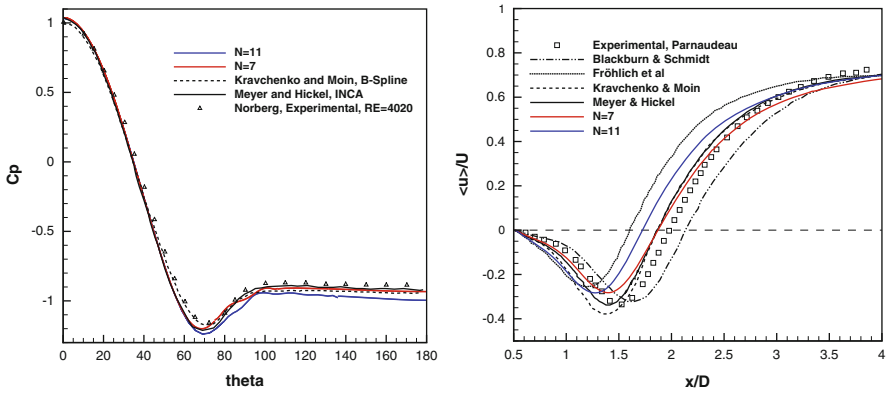

Fig. 4 Distribution of pressure and mean centerline velocity and comparison to results from literature

with coarse mesh and $N = 11$ and the medium mesh computation with $N = 7$, fit well to the range of literature results while using less degrees of freedom than the other approaches. This is also confirmed by the distribution of the pressure and the mean centerline velocity, which compares again well to the available literature results, see Fig. 4.

In a second step, the influence of our artificial dissipation mechanism on the numerical result is investigated. As stated above, artificial dissipation is introduced via the numerical flux functions which are typically based on approximate Riemann solvers. Two very common variants used in the DG community is either the very simple local Lax-Friedrichs flux (LLF) function which is generally known to be highly dissipative and the approximate Riemann solver of Roe which is used because of its low inherent dissipation. Averaged velocity profiles at different distances x/d are compared in Fig. 5 and reveal that the simulation based on the Roe dissipation mechanism is more accurate than the simulation based on the local Lax-Friedrichs flux.

This is an important result for us and will be the base for further investigations in future projects.

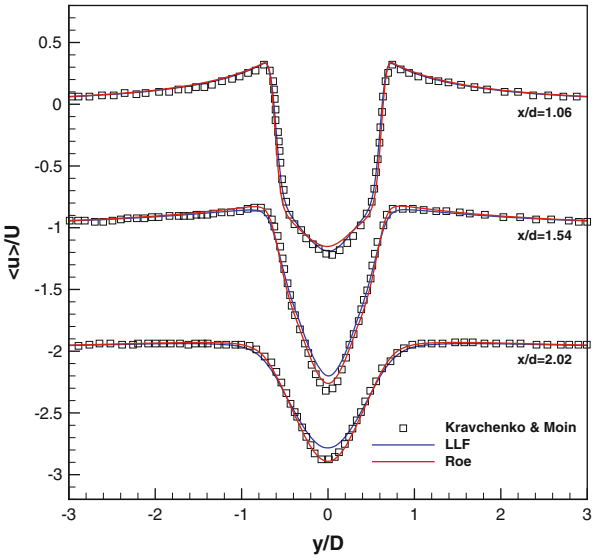

Fig. 5 Comparison of averaged velocity profiles at fixed distances x/d for different numerical flux functions

4 Flow Past an SD7003 Wing at Reynolds number $Re = 60{,}000$

Following the test case C3.3 from the High Order CFD Workshop 2013 (Köln), we perform simulations of the low Reynolds number flow around an SD7003 airfoil at high angle of attack ($Re = 60{,}000, \alpha = 8$). The Mach number is fixed at $Ma = 0.1$. The aim of this test case is to investigate the capability of implicit LES approaches for mixed laminar, transitional and fully turbulent flows. The described setup results in a short separation bubble at the leading edge of the suction side and subsequent turbulent reattachment.

4.1 Simulation Setup

For the discretization, we choose the polynomial degree $N = 3$ and use a fourth order accurate explicit Runge-Kutta time integration method. In this case, only $1.5\,N$ quadrature points are chosen for dealiasing, as the rather low polynomial degree $N = 3$ offers already a certain amount of artificial dissipation.

For the presented case, we use an unstructured mesh with $21,660$ cells, corresponding to $1.39\,10^6$ degrees of freedom at a polynomial degree of 3. Isothermal walls are applied at the wing surface, the geometry curvature is maintained through

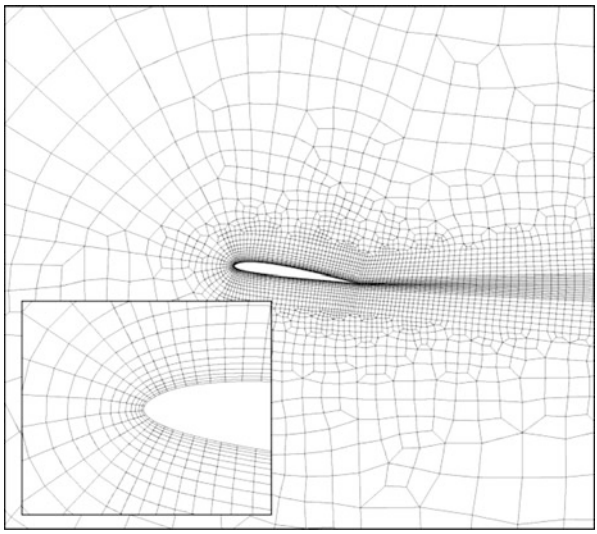

Fig. 6 2D cut of the computational mesh

curved boundaries. In the spanwise direction, we resolve 0.2 chords and employ periodic boundary conditions. The farfield boundary with free-stream conditions is located five to seven chords away from the wing. Since the influence of the farfield boundaries cannot be guaranteed to vanish with this domain size, further work will use larger domains and address this issue. The grid used in this simulation is shown in Fig. 6.

4.2 Results

In order to collect a sufficient amount of statistical data, the simulation is run for 20 convective times (C/U_∞ with the chord C and the free-stream velocity U_∞). The simulation was performed on the Cray XE6 Hermit cluster at HLRS with 2,272 processors and took 8.35 h.

As can be seen in the lift and drag coefficients evolution in Fig. 7, an unsteady transition phase is followed by a statistical steady state.

We plot iso-surfaces of the Q criterion at $t = 10$ in Fig. 8 as a qualitative visualization of the instantaneous near-wall shear flow. Vortical structures emerge at the leading edge of the suction side due to the high angle of attack, the remainder of the suction side's boundary layer remains turbulent.

More quantitative features of this flow are discussed using time averaged data. We extract the data in the time interval $t \in [8, 20]$. In Table 2, the mean aerodynamic loads (lift and drag coefficient) are compared to results from literature. Our results compare favorably to the published data.

Figure 9 (left) shows the computed negative pressure coefficient in comparison with LES results from literature. While the distribution on the pressure coefficients

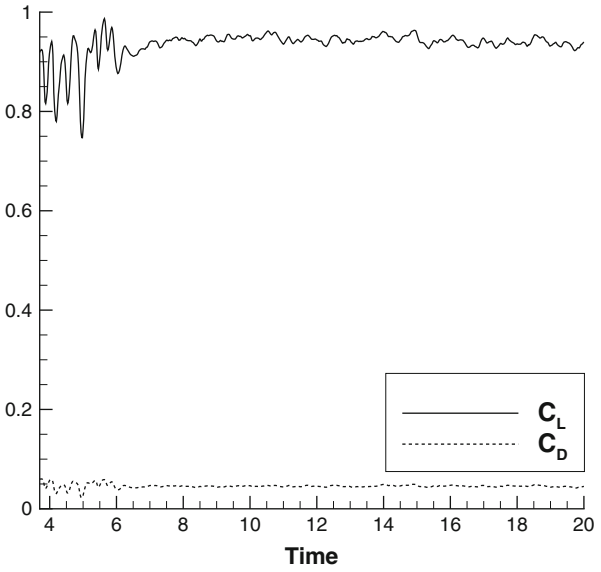

Fig. 7 Evolution of the drag and lift coefficient over time

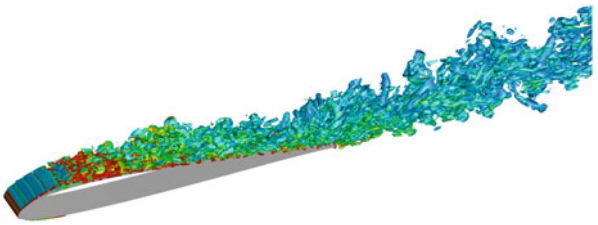

Fig. 8 Iso-surfaces of the Q criterion ($Q = 50$), colored with vorticity magnitude, after 10 convection times

Table 2 Mean aerodynamic loads and comparison to literature

	C_L	C_D
DGSEM, $N = 3$	0.943	0.0455
Garmann and Visbal [4]	0.9696	0.0391
Galbraith and Visbal [3]	≈0.91	≈0.043
Catalano and Tognaccini [2]	≈0.94	≈0.044

on the pressure side is in good agreement between all references and our solution, some differences are visible in the separation region on the first half of the suction side.

The friction coefficient shown in the left part of Fig. 9 reveals an even wider spread between the different results for this case. The negative peak in C_f is in good quantitative agreement with the results of Garmann and Visbal as well as Catalano

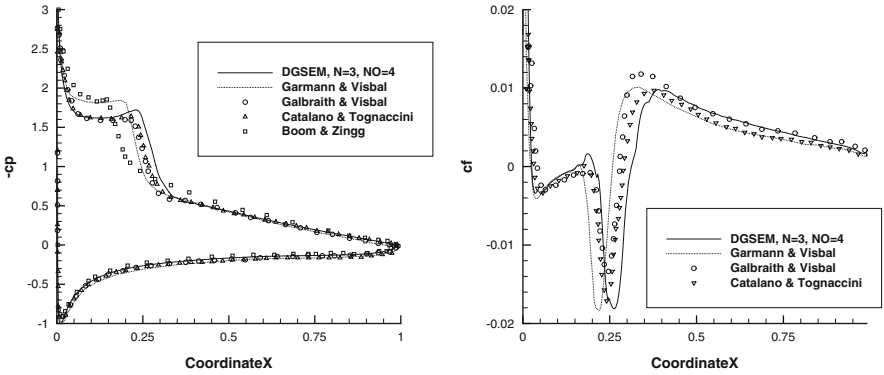

Fig. 9 Time-averaged pressure coefficient (*left*) and friction coefficient (*right*)

Table 3 Separation (x_1) and reattachment (x_2) locations and comparison to literature

	x_1	x_2
DGSEM, $N = 3, NO = 4$	0.025	0.312
Garmann and Visbal [4]	0.023	0.259
Galbraith and Visbal [3]	0.040	0.280
Catalano and Tognaccini [2]	0.030	0.290

and Tognaccini, while the location of this peak is slightly shifted. The computed separation and reattachment points on the suction side are listed in Table 3.

Evaluation of this test case is very difficult, since no DNS data is available for reference. We will continue our work on this case with mesh convergence investigations and higher polynomial degrees as well as larger computational domains.

5 Fluid-Structure Coupling

In the last example, we are presenting the first result obtained with an extension of our framework to a complex multi-physics application. We are considering a fluid-structure interaction, where the flow solver FLEXI is coupled using MPI to a structural solver. The problem is a membrane water pump. The membrane moves up and down and induces the flow to move through the pump, without the need of valves. A sketch of the pump is shown in Fig. 10. In Fig. 11, the unstructured hexahedral mesh, consisting of 1,162 elements, and the flow solution with a deformed membrane is shown. A polynomial degree of $N = 2$ was used, which leads to 31,374 degrees of freedom. The simulation was run with up to 1,162 cores, using only one element per core.

With this test case, the interaction of fluid and structure is investigated. Because of the unsteady nature of the problem and large deformations, a strong coupling

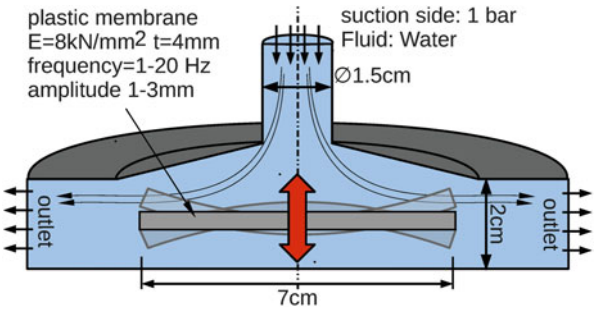

Fig. 10 Setup of the membrane pump fluid-structure interaction

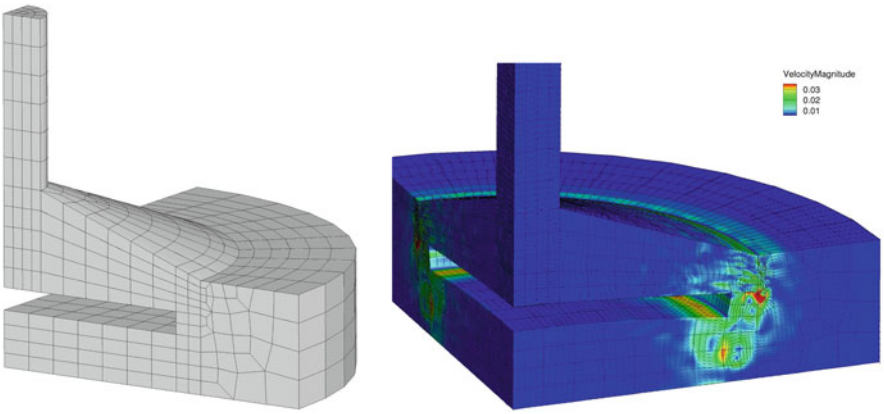

Fig. 11 Mesh of the quarter part of the pump, and flow velocity with a deformed membrane

is envisaged, explicitly in time. We need to solve not only for the flow dynamics, but also the structure dynamics, and as well realize a coupling between the two physics. We will follow a separated approach, where the flow solver works on a moving mesh, and is coupled via MPI to a structural solver. The flow solver FLEXI introduced in Sect. 2.2 was modified, to be able to treat moving meshes. Here, we implemented the approach presented in [9], which leads to a time-consistent DG method on moving meshes for the simulation of unsteady flows.

Up to now, the structure dynamics of the circular membrane is modeled via a one-dimensional fourth order equation, making use of the axial symmetry of the membrane. It is solved on one core with a fast Chebyshev-Collocation method and integrated implicitly in time. The pressure forces are integrated over the membrane surface and communicated from the flow solver to the structure solver in each time step. Then the one-dimensional deformation is passed via MPI messages to the flow solver, which is transformed to a three-dimensional mesh movement. A sketch of the coupling methodology is shown in Fig. 12. Both codes run separately, but inside the

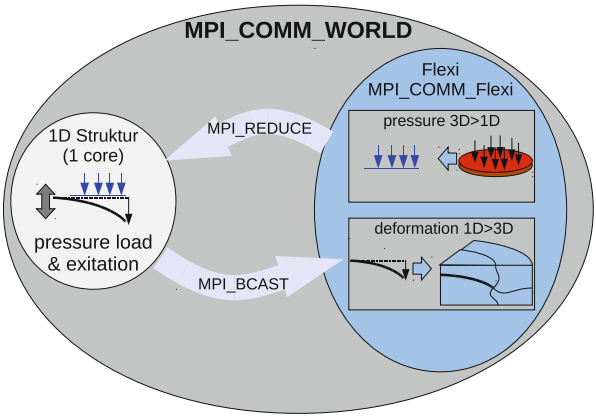

Fig. 12 MPI methodology for the coupling between flow and structure solver

Table 4 Pump simulations (FLEXI $N = 2$, coupled and not coupled) on the HLRS CRAY XE6 system

	#cores	DOF per core	Sim. time	Wall-clock time (h)	Core-h	PID (µs)
Not coupled	581	54	0.4	3.38	1,940	27
Not coupled	1,162	27	0.15	0.78	914	32
Coupled	581	54	0.3	0.13	76.2	28.5

global communicator `MPI_COMM_WORLD`. This is a multiple program multiple data approach (MPMD), and both executables are executed in the same MPI universe using the command

```
mpiexec -n 1  ./struct structure.ini : -n 512 ./flexi flow.ini
```

where the flow solver runs on 512 cores and the structural solver on a single core. This gives a lot of flexibility in both codes, since only the coupling data is interchanged via MPI communication, and the data structures of each program are not affected. In addition, since both codes run parallel to each other, and a timestep for the flow solution takes more time than the structure solution, the coupling data is always ready for communication, leading to a very low overhead, see the difference in the performance index (PID) in Table 4. The uncoupled reference PID is 27 µs whereas the coupled computation has a PID of 28 µs.

Table 4 summarizes the computational time spend on different runs of the problem. Factoring in the different physical runtimes of the simulation (sim. time in the table), a strong scaling from two (PID = 27 µs) to one element (PID = 32 µs) per core has still a scaling efficiency of 84 %.

6 Summary and Outlook

In this project, we have extended our framework to simulate compressible high Reynolds number flows by means of a large eddy type approach and successfully computed the turbulent flow past a cylinder and a wing with typical runs on $> 1{,}000$ processors. The available CPU resources are used to demonstrate the high accuracy of this novel approach in comparison to standard methods from the literature. Furthermore, in the last test case, an extension of this code to a multi-physics application where the MPI interface is used as a connector of the different field models is shown.

Our typical reliable "production" runs use $O(1{,}000)$ processors. In the future, we plan to extend and improve our framework to support reliable simulation runs on $O(10{,}000)$ and even on $O(100{,}000)$ processors to fully unleash the available (and projected) processing power of the HLRS CRAY-XE6 cluster.

Acknowledgements The research presented in this paper was supported in parts by Deutsche Forschungsgemeinschaft (DFG), amongst others within the Schwerpunktprogramm 1276: MetStroem and the Graduiertenkolleg 1095: Aerothermodynamische Auslegung eines Scramjet-Antriebssystems für zukünftige Raumtransportsysteme and the research projects IDIHOM within the European Research Framework Programme.

References

1. H.M. Blackburn, S. Schmidt, Large eddy simulation of flow past a circular cylinder, in *Proceedings of 14th Australasian Fluid Mechanics Conference*, Adelaide (2001)
2. P. Catalano, R. Tognaccini, Large eddy simulations of the flow around the SD7003 airfoil, in *Proceedings of AIMETA Conference*, Bologna (2011)
3. M.C. Galbraith, M.R. Visbal, Implicit large-eddy simulation of low Reynolds number flow past the SD 7003 airfoil, in *Proceedings of 46th AIAA Aerospace Sciences Meeting and Exhibit*, Reno (2008)
4. D.J. Garmann, M.R. Visbal, High-order solutions of transitional flow over the SD7003 airfoil using compact finite-differencing and filtering. Presented at the 1st International Workshop on High-Order CFD Methods, 2012, 50th AIAA Aerospace Sciences Meeting, Nashville
5. F. Hindenlang, G. Gassner, T. Bolemann, C.-D. Munz, Unstructured high order grids and their application in Discontinuous Galerkin methods, in *Conference Proceedings, V European Conference on Computational Fluid Dynamics (ECCOMAS CFD 2010)*, Lisbon, 2010
6. R.M. Kirby, G.E. Karniadakis, De-aliasing on non-uniform grids: algorithms and applications. J. Comput. Phys. **191** (2003)
7. A.G. Kravchenko, P. Moin, Numerical studies of flow over a circular cylinder at ReD = 3900. Phys. Fluids **12**, 403 (2000)
8. M. Meyer, S. Hickel, N.A. Adams, Assessment of implicit large-eddy simulation with a conservative immersed interface method for turbulent cylinder flow. Int. J. Heat Fluid Flow **31**(3) (2010)
9. C.A.A. Minoli, D.A. Kopriva, Discontinuous Galerkin spectral element approximations on moving meshes. J. Comput. Phys. **230**(5), 1876 (2011)