A Domain Decomposition Method for the Efficient Direct Simulation of Aeroacoustic Problems

A thesis accepted by the
Faculty of Aerospace Engineering and Geodesy
of the Universität Stuttgart
in partial fulfilment of the requirements for the degree of
Doctor of Engineering Sciences (Dr.-Ing.)

by

Jens Utzmann

born in Bamberg

Main-referee	:	Prof. Dr. Claus-Dieter Munz
Co-referee	:	Prof. Dr. Eric Sonnendrücker
Date of defence	:	01.12.2008

Institut für Aerodynamik und Gasdynamik
Universität Stuttgart

2008

"In den Wissenschaften ist viel Gewisses, sobald man sich von den Ausnahmen nicht irre machen läßt und die Probleme zu ehren weiß."
Johann Wolfgang von Goethe

"D'OH!"
Homer J. Simpson

Preface

I would like to thank my doctoral supervisor Prof. Dr. Claus-Dieter Munz, who managed to spark my interest in the field of numerical methods already in my years of study. He supported and arranged my stay in Ann Arbor, Michigan, USA, where I wrote my diploma thesis under the inspiring supervision of Prof. Philip L. Roe. After that, I received a lot of support and thought-provoking impulses from him during my time as a Ph.D. candidate at the IAG. There, especially the informal and creative working environment left much space for the realization of own ideas.

Also many thanks to my colleagues at the IAG, they were always available for discussions, offered a lot of input and made my time at the institute a very enjoyable one. Special thanks to Harald Klimach from the HLRS: His expertise and humor helped me out more than once during debugging sessions.

This work was financed by the Deutsche Forschungsgemeinschaft (DFG) in the framework of the project C8 in the SFB 404 "Multifield Problems in Solid and Fluid Mechanics" and the German-French DFG-CNRS Research Group FOR 508, *Noise Generation in Turbulent Flows*. Thank you for supporting me in these projects, which opened the doors to a very interesting community of researchers.

Stuttgart, 1^{st} of October 2008

Jens Utzmann

Contents

Symbols

$\mathcal{O}$	Order of the numerical scheme in space and time
a	Maximum convection speed
c	Speed of sound
c_p, c_v	Specific heats
d	Dimension
f	Frequency / function
$\vec{f}$	Flux in x-, y- and z-direction of a scalar variable
g	Numerical flux of a variable in normal direction
i_{GP}	Spatial Gauss point index in a ghost element
i_S	Stencil element index
k, $\vec{k}$	Wave number and wave number vector
l_{min}	Minimum inner circle or sphere radius
n	Refraction index
n_{DOF}	Number of degrees of freedom
n_{GP}	Number of spatial Gauss points in a ghost element
n_λ	Number of spatial GPs in a domain element
n_ξ	Number of temporal GPs in a domain element
n_{Poly}	Polynomial degree
n_S	Number of stencil cells
n_{Var}	Number of state vector variables
n_W	Number of different interpolation stencils
$\vec{n}$	Normal vector
p	Pressure
r	Radius
s	Sponge power
t	Time
u, v, w	Velocity component in x-, y- and z-direction
u	General scalar state
v_G	Group velocity
$\vec{v}$	Velocity vector
w	Characteristic variable

w_{WENO}	Reconstructed WENO polynomial
A	Amplitude
$\underline{\underline{A}}, \underline{\underline{B}}, \underline{\underline{C}}$	Jacobians in x-, y- and z-direction
C_L, C_D	Lift and Drag coefficient
CK_k	CK procedure for the kth time derivative
D	Diameter
$\vec{F}$, $\vec{G}$, $\vec{H}$	Flux vectors in x-, y- and z-direction
I_I, I_R, I_T	Intensity of impinging, reflected, transmitted wave
$\underline{\underline{J}}$	Jacobian of the transformation matrix
L	Lagrangian polynomial / sponge thickness
L_1, L_2, L_∞	Error norms
M	Number of Gauss integration points in 1D
Ma	Mach number
N	Maximum degree of the basis functions
$N_{\Delta h}$	Number of mesh elements in x-direction
Pr	Prandtl number
R	Specific gas constant or reflection index
Re	Reynolds number
S_F	Smoothing factor
$\vec{S}$	Right-hand side source term
T	Temperature / period length / transmission index
$\vec{U}$	State vector
V	Volume
$\vec{X}$	Vector of spatial coordinates
$\vec{X}_{i_{GP}}$	Coordinate of Gauss point i_{GP}
$\vec{X}^T$	Target coordinate
$\underline{\mathrm{X}}_T$	Transposed vector of monomials
γ	Ratio of specific heats
λ	Wave length
λ_C, ϵ, r	WENO parameters
$\vec{\lambda}$	Vector of Eigenvalues
μ	Dynamic viscosity
ν	Kinematic viscosity
ω	Angular velocity

$\omega_{i_{GP}}$	Weight of Gauss point i_{GP}
ω_i, $\tilde{\omega}_i$	Normalized and non-norm. nonlin. WENO weight
ω_λ	Weight of spatial Gauss point λ
ω_ξ	Weight of temporal Gauss point ξ
ρ	Density
ρE	Total energy per mass unit
σ	Sponge parameter
σ_i	WENO oscillation indicator
τ	Relative time with respect to time level t^n
$\underline{\underline{\tau}}$	Viscous stress tensor
ξ, η, ζ	Coordinates in the reference coordinate system
Δh	Characteristic element size
Δt	Time step
Δx, Δy, Δz	Size of a rectangular element in x-, y-, z-direction
Γ	Coupling boundary
Ω	Domain volume / control volume
$\partial\Omega$	Domain boundary / control volume boundary
Φ	Basis and test function
$(\cdot)_\infty$	Ambient flow variable
$(\cdot)_0$	Mean flow variable
$(\cdot)'$	Perturbation variable
$(\cdot)_t$, $(\cdot)_x$, $(\cdot)_y$, $(\cdot)_z$	Time and space derivatives of a variable
$(\cdot)^\pm$	Value at the left (-) / right (+) side of an interface
$(\cdot)_{i-\frac{1}{2}}$, $(\cdot)_{i+\frac{1}{2}}$	Value at the left / right cell interface (x-index i)
$(\cdot)_{ijk}$	Value for an element with the struct. indices i,j,k
$(\cdot)^n$, $(\cdot)^c$	Value at time level n and at current time level
$\hat{(\cdot)}$	Degree of freedom (DG) / amplitude
$(\cdot)^T$	Transposed
$(\cdot)_T$	Target
$\vec{(\cdot)}$, $\underline{(\cdot)}$	Vector
$\underline{\underline{(\cdot)}}$	Matrix

Abbreviations

ADER	Arbitrary high order using derivatives
CAA	Computational aeroacoustics
CFD	Computational fluid dynamics
CFL	Courant-Friedrichs-Levy number
CK	Cauchy-Kovalevskaja
CPU	Central processing unit
DMR	Double Mach reflection
DNC	Direct noise computation
DNS	Direct numerical simulation
DG	Discontinuous Galerkin
DOF	Degree of freedom
DRP	Dispersion relation preserving
EE	Euler equations
ENO	Essentially non-oscillatory
FD	Finite difference
Flop	Floating point operations
FV	Finite volume
GP	Gauss integration point
GRP	Generalized Riemann problem
HLLE	Harten, Lax, Van Leer, Einfeldt
LEE	Linearized Euler equations
LTS	Local time stepping
N.-S.	Navier-Stokes
PDE	Partial differential equation
PPW	Points per wavelength
QF	Quadrature-free
Rec-FV	Reconstructed finite volume
RK	Runge-Kutta
RMS	Root mean square
RP	Riemann problem
STE	Space-time expansion
TVD	Total variation diminishing
WENO	Weighted essentially non-oscillatory

Kurzfassung

In dieser Arbeit wird ein Gebietszerlegungsverfahren entwickelt, welches die direkte Simulation von aeroakustischen Problemen erheblich beschleunigt. Alle relevanten Skalen müssen hier sehr genau aufgelöst werden, von den kleinen, Schall produzierenden Strömungsphänomenen, die relativ viel Energie enthalten (z.B. Wirbel), bis hin zum langwelligen Schall mit niedrigen Druckamplituden, der über eine große Distanz und ohne Dissipations- und Dispersionsfehler transportiert werden muss. Damit der Rechenaufwand nicht die Möglichkeiten heutiger Computer übersteigt, bzw. um lange Rechenzeiten auf ein wirtschaftlich vertretbares Maß zu verkürzen, wird das Rechengebiet gemäß der auftretenden physikalischen Phänomene aufgeteilt. In diesen Untergebieten wird dann ein möglichst optimales und auf das Teilproblem zugeschnittenes Verfahren verwendet. Für das vorgestellte Verfahren wird die Gebietszerlegungsidee von Schwartzkopff aufgegriffen und daraus ein neuer, verallgemeinerter Ansatz entwickelt. Dabei unterscheidet sich die Methode von bisher üblicherweise eingesetzten Techniken. So beschränkt sich die Gitterkopplung z.B. nicht auf Chimera-Verfahren, sondern zeigt einen konsistenten Ansatz für die Kopplung in Raum und Zeit von Verfahren hoher Ordnung auf.
Verschiedene Optionen der Gebietszerlegung werden untersucht und in einem übergreifenden Programmgerüst vereint. In den Untergebieten werden die Navier-Stokes-, Euler- und linearisierten Eulergleichungen gelöst, wofür Discontinuous Galerkin (DG), Finite Volumen (FV) und Finite Differenzen (FD) Methoden jeweils mit ihren speziellen Eigenschaften zur Verfügung stehen. So eignen sich z.B. DG Verfahren aufgrund ihrer Lokalität für sehr genaue Lösungen auf unstrukturierten Gittern, während FD Verfahren lineare Schallausbreitung besonders effizient auf kartesischen Gittern simulieren. FV Verfahren wiederum sind sehr robust in der Gegenwart von starken Gradienten, z.B. Stößen. Allen implementierten Methoden ist gemein, dass sie als explizite Einschrittverfahren in der Zeit besonders für instationäre Probleme geeignet sind und sich ihre Genauigkeitsordnung in Raum und Zeit frei wählen lässt. Ein im Rahmen dieser Arbeit neu entwickeltes Verfahren, die STE-FV Methode auf kartesischen Gittern, schließt die Lücken in der Auswahl der vorhandenen Löser und stellt ein schnelles Verfahren hoher Ordnung dar, welches durch einen WENO Algorithmus auch bei Nichtlinearitäten robust bleibt. Zur Validierung der STE-FV Methode werden Konvergenztests durchgeführt und Beispiele mit bis zu sechster Ordnung in Raum und Zeit und darüber hinaus gerechnet, wie

z.B. die bekannte "double Mach reflection" in 2D und eine Explosion in 3D. Die Gitterkopplung selbst beruht auf Interpolationen hoher Ordnung und dem Datenaustausch über die Geisterelemente der Rechengebiete. Hier werden die Stützstellen für die Gauß-Integration in den Zellen benutzt, um geeignete Quellgebiete für die Interpolation zu finden und danach die Kopplungsrandbedingungen hoher Ordnung zu setzen. Die Gitter brauchen sich dabei nicht zu überlappen und müssen nicht konform sein, außerdem sind beliebige Konstellationen unstrukturierter und strukturierter Gitter möglich. In den Teilgebieten sind jeweils optimale und voneinander verschiedene Zeitschritte erlaubt, was durch den Einsatz der Cauchy-Kovalevskaja Prozedur ermöglicht wird. Diese stellt eine Taylorreihe in der Zeit zur Verfügung, welche Randdaten für die Zwischenzeitpunkte der Gebiete mit kleinerem Zeitschritt liefert. Die Implementierung im Programmgerüst ist weitgehend modular aufgebaut, so dass die einzelnen Strömungs- und Akustiklöser auch separat verwendet und neue hinzugefügt werden können. Weiterhin besteht die Möglichkeit, externe Programme, die zudem auf einem getrennten Computersystem laufen können, anzukoppeln. Die Verteilung auf verschiedene Rechnerarchitekturen ist ebenso für die internen Rechengebiete möglich, so dass die jeweils unterschiedlichen Eigenschaften hinsichtlich Vektorisierung und Parallelisierung optimal ausgenutzt werden können.
Für das Gebietszerlegungsverfahren wird in Konvergenzstudien für verschiedene Gitter-, Gleichungs- und Verfahrenskonstellationen bewiesen, dass die hohe Ordnung der eingesetzten Methoden global erhalten bleibt. Untersuchungen bezüglich hochfrequenter Störungen zeigen, dass selbst ohne das Anwenden eines räumlichen Filteroperators eine Art natürliche Filterung stattfindet, wenn die Störungen auf einem groben Gitter nicht mehr aufgelöst werden können. Eine weitere Studie zeigt, dass die Größe der auftretenden Reflektionen an den Gebietsgrenzen gut mit theoretischen Abschätzungen übereinstimmt. Außer dem Wechsel von nichtlinearen zu linearisierten Gleichungen spielt hier auch der Sprung im Auflösungsvermögen eines Verfahrens eine Rolle. Generell sind die Reflektionen so klein, dass sie vernachlässigt werden können.
Die Effizienz und Genauigkeit der vorgestellten Gebietszerlegungsmethode wird anhand von Benchmark-Beispielen wie der akustischen Streuung an einer Kugel und mehreren Zylindern, sowie der Von Karmanschen Wirbelstraße aufgezeigt. Hier wird besonders das Potential des Verfahrens für effiziente Fernfeldrechnungen, aber auch die Einsetzbarkeit beim Vorhandensein komplexer Geometrien deutlich. Die Simulation einer Düse mit supersonischem Freistrahl und des damit verbundenen Lärms unterstreicht schließlich die praktische Anwendbarkeit des Gebietszerlegungsverfahrens.

Abstract

A novel domain decomposition approach is developed in this thesis, which significantly accelerates the direct simulation of aeroacoustic problems. All relevant scales must be resolved with high accuracy, from the small, noise generating flow features (e.g. vortices) to the sound with small pressure amplitudes and large wavelengths. Furthermore, the acoustic waves must be propagated over great distances and without dissipation and dispersion errors. In order to keep the computational effort within reasonable and feasible limits, the calculation domain is divided into subregions with respect to the local physical requirements. In these domains, the numerical method which is most suitable and optimized for the considered subproblem is employed. Following the decomposition philosophy by Schwartzkopff, a new and generalized method is developed in this work. The scheme differs from established approaches, e.g. the grid coupling is not limited to Chimera techniques but presents a consistent way for the space-time coupling of high order methods. Various domain decomposition options are examined and implemented in a common code framework. In the subdomains, the Navier-Stokes, Euler and linearized Euler equations are solved, for which methods from the discontinuous Galerkin (DG), finite volume (FV) and finite difference (FD) class are available with their respective special properties. For example, DG methods are very suitable for highly accurate solutions on unstructured grids due to their locality, while FD methods are very efficient on Cartesian grids for the simulation of linear wave propagation. In turn, FV methods are very robust in the presence of strong gradients, e.g. shocks. All implemented methods have in common, that they are explicit one-step time integration schemes and thus are especially applicable for unsteady calculations. Furthermore, their order of accuracy in space and time may be chosen arbitrarily. A newly developed numerical solver, the STE-FV method on Cartesian grids, closes the gaps in the repertoire of numerical schemes in the coupling framework. It forms a fast high order method that features great robustness also at nonlinearities by employing a WENO algorithm. For validation purposes, convergence studies and benchmark tests, e.g. the popular double Mach reflection in 2D and an explosion in 3D, are performed for the STE-FV method with orders in space and time up to six and beyond.
The coupling of different grids is based on high order interpolations and the data exchange over the ghost elements of the calculation domains. The Gauss integration points in the cells are used here in order to find a source domain for the interpolation and for providing high order boundary conditions afterwards. The grids are not required to be matching or overlapping. Furthermore, ar-

bitrary constellations of structured and unstructured grids are possible. The optimal time steps, which can be different of each other, are allowed in the subregions. This is made possible by employing the Cauchy-Kovalevskaja procedure, which delivers a Taylor series that provides boundary information for the intermediate points of time for domains with a smaller time step. The implementation structure inside the code framework is largely modular. The fluid and acoustics solvers can be used as stand-alone codes, and also new ones can be easily added. Furthermore, external programs, which may run on separate computer systems, can be linked to the framework. The distribution to different system architectures is also possible for the internal solvers. Hence, the respective properties of the numerical methods regarding vectorization and parallelization can be exploited in an optimal way.

It is shown on the basis of convergence studies for different constellations of grids, equations and methods, that the domain decomposition approach is capable of maintaining high order of accuracy globally. An examination regarding high-frequency perturbations reveals a natural filtering process if perturbations cannot be resolved on a coarse mesh anymore. Hence, a spatial filtering operator is not a necessity. Another study shows, that the magnitude of reflections occurring at the domain boundaries are in good accordance with theoretical estimations. Besides the change from nonlinear to linear equations, also the jump in resolution matters in this context. However, the reflections are negligible in general.

The accuracy and efficiency of the proposed domain decomposition method is illustrated for benchmark examples like the acoustic scattering at a sphere or at multiple cylinders and for the Von Karman vortex street. Here, especially the method's potential for efficient far field calculations becomes clear, but also the advantages in the presence of complex geometries are emphasized. Finally, the simulation of a nozzle flow with a supersonic free jet and the associated noise underlines the practical applicability of the domain decomposition approach.

1 Introduction

Noise in flows is usually produced by the generation or interaction of vortices. Due to the so-called multi-scale problem of aeroacoustics, the numerical simulation of both generation and propagation of acoustic waves in one calculation is difficult: While the noise producing flow features are mainly very small but have large amplitudes, sound waves contain significantly less energy and possess a comparatively large wavelength. In order to resolve all scales accurately in an unsteady simulation, a vast number of grid elements would be necessary with the standard second order schemes that are used nowadays. Even then those methods are often too dissipative in terms of numerical damping and are not capable of handling long-distance propagation of sound waves. Furthermore, in flows with small Mach numbers, the difference between the flow velocity and the speed of sound leads to a disparity of the local CFL numbers of the calculation, as the largest eigenvalue is used for the time step calculation. Consequently, the global time step of the computation becomes very small for *all* elements in the domain. All of this results in a prohibitive computational effort for direct simulations and has lead in the past to hybrid approaches in order to overcome some of these difficulties. The most classical approach is the volume coupling over source terms. Here, the flow simulation is performed first, followed by the computation of acoustic source terms from this flow by an acoustic analogy. These source terms are then used in a second calculation that solves acoustic equations, for example Lighthill's equation [64, 65] or the Ffowcs Williams-Hawkins equation [30]. More recent developments are the perturbed compressible equations (PCE) by Seo and Moon [94, 95] and the acoustic perturbation equations (APE) by Ewert and Schröder [28]. Such approaches are often referred to as "hybrid coupling" or "hydrodynamic/acoustic splitting" (the latter if the flow calculation is based on incompressible equations). Another traditional approach for the simulation of acoustic problems is the Kirchhoff method (e.g., Farassat and Myers [29]). The coupling of flow and sound is treated over an integration surface around the acoustically active region, which must include every relevant source of noise. Note that the sound waves need to be resolved here already in the flow simulation. Hence, the solver must be accurate enough to transport these waves to the integration surface

without significant numerical damping. Then the source terms are handed over to the respective acoustic tool, for example a wave equation solver. All of the mentioned methods have in common, that the acoustic field is calculated independent of the flow field, the latter serves merely as a generator for the acoustic sources. Thus, there is no feedback from the acoustics to the flow, which can however be of great importance as some phenomena are directly related to such a feedback (e.g., jet screech, Tam et al. [96, 107]). Many of the acoustic models are restricted in a way, for example some of the models do not permit solid walls in the flow, some are only suitable for flows with uniform or no background velocity. Furthermore, the extraction of the acoustic sources always involves some modeling in one way or the other. Hence, along with the uncertainties in the original flow calculation, the final noise prediction underlies a lot of assumptions. This is not very satisfying. The direct computation of both flow and noise in one calculation, also called direct noise computation (DNC), is the remaining alternative. It includes the feedback of the sound to the flow and requires the least modeling, but as mentioned in the beginning, the computational effort can grow easily beyond the the capabilities of modern computer systems due to the multi-scale problem. In order to circumvent this problem and to facilitate simulations, the idea of heterogeneous domain decomposition has been developed. The most important methods in this regard are given a review in the following section.

1.1 Domain Decomposition Methods: State of the Art

In order to reduce the overall complexity of an aeroacoustic simulation, different parts of the calculation domain are treated differently, depending on their local physical requirements. By using the most suitable and efficient scheme in every subregion, the calculation is accelerated or even made feasible in the first place. This local adaptation will be referred to as domain decomposition or domain coupling and is generally implemented by

1. decomposing the computational domain into several partitions with different properties;

2. connecting these domains by exchanging data amongst them or by coupling the governing equations;

The example of a nozzle flow with a free jet (Fig. 1.1) illustrates the various ways, how a computational domain can be decomposed.

First of all, a nozzle can be a rather complex geometry (much more compli-

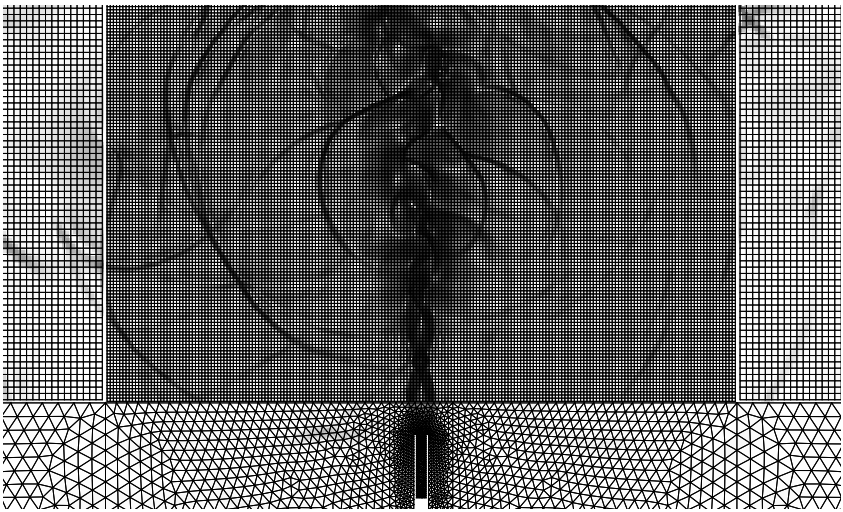


Figure 1.1: Nozzle and supersonic free jet with acoustic waves in a simulation with domain decomposition.

cated than in Fig. 1.1). The generation of a suitable structured mesh is often a matter of days or weeks. It can consume as much or even more time as the actual numerical simulation. A very convenient alternative are unstructured meshes (e.g., based on triangles or tetrahedrons), which are considerably easier to create with modern grid generators. Even the most complicated objects can be meshed relatively fast. Farther away from the nozzle and in the absence of other objects, there is no need for unstructured grids anymore. Structured or even Cartesian grids can be used and the methods for those grids are usually more efficient in terms of solution quality and computational time. This is the first way to decompose the problem, dividing the domain into an unstructured and a structured part. Back to the nozzle, the grid in its vicinity most likely needs to be very fine in order to resolve boundary layers or other regions with strong gradients, such as shocks. Delicate geometry details might require small element sizes, too. Other small-scale flow features (e.g., vortices) ask for relatively fine grids also in structured domains. When unsteady calculations are performed with explicit solvers for aeroacoustics, very small cells are directly

reflected in a very small global time step. On the other hand, in regions where only acoustic propagation is significant, the grid can be quite coarse for the considerably larger wavelengths. Hence, those cells suffer from an unnecessary small time step. Thus, the coupling of different grid sizes offers two advantages at the same time: It reduces the overall number of elements which must be computed, and every subdomain can use its own ideal time step if there is a suitable algorithm for this. Hence, two more ways of decomposing the domain are the coupling of different grid sizes and local time stepping. Note that in regions which require a very fine grid resolution, for example due to the mesh geometry or a shock (and not necessarily for the resolution of other flow phenomena), also a lower order solver could be used to reduce the numerical effort. In subdomains with smooth, large-scale features, a high order solver on a coarse grid can be applied on the other hand, which is usually more efficient. Thus, also an adaptivity regarding the order of the scheme is possible. Returning again to the nozzle, the numerical method of choice near complicated (e.g., cambered) wall geometries would be a finite volume (FV) or a discontinuous Galerkin (DG) method, for which a high order treatment of curved boundaries is possible. Moreover, high order of accuracy, which is needed in order to resolve both flow and acoustics, can be naturally obtained with the very local DG methods (and recently also with FV methods) on unstructured grids. Then again, FV methods have the great advantage of being very robust in the presence of discontinuities. On the other hand, optimized high order finite difference methods exist for large-scale wave propagation on structured grids. Therefore, another way of domain decomposition opens up: The coupling of different numerical methods. Section 2.1 provides a detailed review of the strengths and drawbacks of the single numerical methods. Last but not least, also the governing equations may vary locally. The Navier-Stokes equations must be solved where viscous effects are crucial, for example in the boundary layer. Saving the effort of handling the diffusion terms, the Euler equations could be employed in regions where viscous effects have decayed. Finally, if only acoustic perturbations remain in the subdomain, the problem can be described with linearized equations (e.g., the linearized Euler equations, the wave equation), for which very efficient methods exist.

Several of these aspects of domain decomposition have been explored in the past. Block-structured grids with matching grid points are the easiest way to couple meshes, followed by grids with hanging nodes and linear interpolation. Overset-grid (Chimera) methods (introduced by Steger et al. [103]) couple structured, body fitted grid components with Cartesian background grids. They allow moving grid interfaces and are used in a wide range of applications.

An overview is given by Meakin [71]. In order to maintain a high order of accuracy globally, the interpolations between the grids need to be high order as well (Delfs [17]) and have been subject to several studies regarding optimizations (Tam et al. [105], Sherer et al. [99]). The idea of multi-size overset grids with different time steps has been examined by Tam et al. [106] for dispersion relation preserving (DRP) schemes. For a particular jump in the grid spacing (exactly by a factor of two), an equivalent jump in the time step could be realized. The interpolation and data organization between overlapping grid partitions is rather complex and differs from application to application. For this purpose, so-called grid assembly tools are available. The *Overture* library [43] by the Lawrence Livermore Laboratory connects structured, overlapping grids in 2D and 3D with arbitrary order of interpolation. It also handles load balancing and adaptive mesh refinement. Another tool worth mentioning is *MpCCI* (Mesh-based parallel Code Coupling Interface [32]) by the Fraunhofer-Institue SCAI, which aims at the coupling of entirely different codes and has even standardized interfaces to the most common commercial simulation tools (like *FLUENT*, *ANSYS CFX*, etc.). It organizes the interpolation and communication of physical quantities between different grids and supports MPI parallelization. Although the grid types involve both structured and unstructured meshes, the interpolation is only low-order (linear) so far. However, it has been successfully used for CAA in volume coupled hybrid calculations (Ali et al. [1], Escobar et al. [27], Kaltenbacher et al. [52], Krey [59]).
For aeroacoustics, the coupling of equations has been examined by Freund, Lele and Moin [33,34,63]. They solved the compressible Navier-Stokes equations in the source region, while the isentropic linearized Euler equations were used for the radiation of acoustic waves. Also different Cartesian grid sizes were employed in the subregions. Using energy estimates, Nordström [77] combined for hyperbolic problems an unstructured finite volume method with a high order finite difference scheme. He also proved strict stability for this coupling method.
An entirely different branch of decomposition approaches remove the interface coupling and introduce an iterative process between subdomains instead. Quarteroni [84] gives an overview of those "monolithic" methods. Coclici [15, 16] examined the coupling of the Navier-Stokes equations in the near field with the linearized Euler equations in the far field for *steady* problems, using such an iterative approach. It is emphasized, that the calculation of steady solutions is not the aim of the presented decomposition method.
Last but not least, the class of mortar methods for domain decomposition shall be mentioned. These surface coupling schemes are able to handle non-matching unstructured grids and employ a so-called Lagrange multiplier space

in order to create a finite element-like constraint at the interface. The Lagrange multipliers can be interpreted as the normal flux at the coupling boundary. Although the mortar method is categorized as an iterative scheme, it has been also successfully applied to acoustic-acoustic and elasto-acoustic coupled problems (Flemisch et al. [31]) with time-accurate solvers in 2D and 3D subdomains.
Many other approaches for a huge variety of scenarios and equations (fluid-structure coupling, contact problems in elasticity, etc.) have been examined in the past and it would go beyond the scope of this overview to elaborate on all of them. Despite of that, a related domain coupling method which preceded this work is given more regard in the following.

1.2 Heterogeneous Domain Decomposition by Schwartzkopff

A flexible heterogeneous domain coupling method was developed by Schwartzkopff [88, 89, 119]: Different domains are coupled at their common boundary over the data in their ghost elements. These ghost cells provide boundary conditions for the numerical methods. In the so-called restriction case, the projection of the ghost cell on the partner domain covers more than one cell. In this case, an integral average of all cells which lie partially in the projection is used. The weight of this averaging is the ratio of the respective area of the cell which is in the projection, and the total area of the ghost cell. In the interpolation case, the projection of the ghost cell lies completely within one cell on the partner domain. Here, a conservative interpolation from the coarse grid to the Gauss quadrature points of the target cell is used. Dumbser [18] added techniques for the coupling of ADER-DG domains with other ADER-DG or ADER-FV domains to the framework. Schwartzkopff's method worked well on both structured and unstructured grids in two dimensions and included features such as the subcycling of time steps and the coupling of different equations. However, although the method had been successfully applied to several benchmark examples, it possessed several disadvantages. First of all, determining the cut-out polygons for the averaging case leads to complex algorithms. Furthermore, the treatment of "multi-domain ghost elements", cells which overlap different neighboring domains (Fig. 2.13), is very cumbersome. Those already in 2D unattractive properties cause great difficulties for an extension to 3D, yet the coupling method is still limited to some selected decomposition scenarios. Moreover, it is not very convenient to make a difference between "restriction" and "interpolation" cells: Such a distinction is not quite clear if meshes of similar grid size or different type (element shape) are

coupled. The distinction between restrictions and interpolations is based on the neighbor element's volume, while the very important subcycling feature is based on the time step ratio between domains. This prohibits configurations like a high order DG method on a coarse grid with a small time step (restricted by the DG scheme's stability limit) coupling with a somewhat finer FD grid with a larger time step: The DG ghost cells would be filed under "restriction" (thus averaging) and would not be provided with time derivatives for a CK procedure. The latter is required for subcycling.

1.3 Aims of this Work

The final objective of this work is to create a simulation framework based on domain decomposition, that offers as much flexibility as possible and that can be applied to complex aeroacoustic problems. The optimal numerical method in terms of efficiency and quality shall be used exactly there – and only there – where it is needed in order to reduce the overall computational effort of a simulation. The "ideal" decomposition framework should possess the following properties:

- The computational domain can be divided into arbitrary, non-overlapping domains with different properties.
- The spatial decomposition and the choice of numerical methods depend only on the problem one likes to solve.
- The single methods can be combined and coupled with others, so they can be utilized in a common framework. This means in particular the coupling of different
  - equations (e.g., N.-S., EE, LEE)
  - methods (e.g., FV, FD, DG)
  - grids (e.g., structured hexahedrons, unstructured tetrahedrons)
  - time steps (e.g., different time step ratios between domains)
- The data exchange routine between the domains is able to handle the conversion between the different local topologies (e.g., FV-/FD-/DG-data on structured/unstructured grids).
- The coupling procedure takes advantage of the particular solver properties and conserves them:

  - High order of accuracy during the data exchange.
  - Conservation of globally high order of accuracy if all employed methods are high order. This ensures, that the method error remains globally as small as possible.
  - Time accurate coupling of explicit solvers for unsteady solutions.

- Subdomains are distributed to and calculated on different and optimal system architectures, for example structured FD methods on vector computers, unstructured DG schemes on PC clusters.
- The entire decomposition framework can be parallelized effectively for a large number of processors.
- The coupling algorithm connects 2D domains as well as 3D domains.
- It creates a minimum of computational overhead, therefore keeps the overall number of exchanged data and element updates low.

Of course, many other features are thinkable and could be added to this list. The established decomposition methods or commercial packages listed in section 1.1 are capable of handling only few of the above mentioned features at once.
In order to add more flexibility, a new approach based on high order polynomial interpolations is investigated. A straightforward way opens up if the coupling and data exchange is organized over the boundary conditions of the numerical methods in the subdomains, that have to be treated anyway for all calculations. Many numerical methods employ so-called ghost elements for this purpose, which have to be provided with information prior to every time step. This is the strategy, with which Schwartzkopff (see section 1.2) successfully introduced a method that implemented the above mentioned properties (e.g., global conservation of high order) to some extent.
The aim of the proposed coupling approach is now to use Schwartzkopff's basic principle, the coupling over the physical state in the ghost cells, but generalize the procedure to a great extent. The simplification is achieved by creating a coupling mechanism completely based on the exchange of data in the Gauss integration points of the ghost elements. Then the difference between restriction and prolongation vanishes. Geometric considerations regarding the cell sizes become mostly obsolete and are only used for optimizations of the scheme. Because the interpolation points can be exactly targeted, arbitrarily complex domain constellations with "multi-domain ghost elements" are possible and

the extension to 3D is straightforward. The different aspects of the domain decomposition are implemented in a "building block" manner. This modularity supports the integration of new coupling features and numerical methods. A variety of novel and established high accuracy solvers is integrated and a thorough investigation of the single building blocks is performed to ensure a coherent framework for the domain decomposition.

1.4 Outline

The single components of the domain decomposition framework are explained in the following chapter. First of all, the implemented high order flow and acoustics solvers are explained in their basic principle. Amongst these methods, the WENO STE-FV method for Cartesian grids is given special attention. It will be described in more detail, as it is a special development for closing the gaps in the repertoire of numerical methods in the framework. The STE-FV method is examined and validated with a convergence study and several test cases. An overview of the governing equations in the subdomains is given, before the grid coupling mechanism is illustrated extensively. After that, issues such as the the subcycling method and implementation details are addressed. Finally, the behavior towards convergence, reflections and high frequency perturbations is examined thoroughly in the validation section. In the last chapter, the domain decomposition framework is applied to selected benchmark problems and "real life" simulations. The first example is the so-called *Multiple Cylinder Scattering* from the *Fourth CAA Workshop on Benchmark Problems*. It illustrates, how domains with relatively complicated arrangements of geometries can be easily split into subdomains. Afterwards, the flow around a cylinder and the resulting *Von Karman vortex street* shows the efficiency of the domain decomposition method especially for far field acoustics. In another benchmark example, the *Sphere Scattering* from the *Second CAA Workshop on Benchmark Problems*, the 3D capabilities of the coupling approach are demonstrated. Last but not least, the challenging problem of sound generation and propagation in a supersonic free jet is simulated.

2 Domain Decomposition

2.1 Numerical Methods

Within the coupling framework, a variety of numerical methods are available for use in the subdomains. They all have in common that they are explicit time integration schemes of arbitrary high order in space and time.
Sound generation and propagation are inherently unsteady problems, hence explicit methods seem to be the natural choice for a direct simulation. Steady solutions are then included as a special case. Nevertheless, also semi-implicit solvers with a physical "outer" time step and "inner" iterations (so-called "dual

2D methods	Grid	Globally LEE	Locally LEE	EE	N.-S.
ADER-DG	unstr.	•	•	•	
Rec-FV	unstr.	•	•	•	•
ADER-FV	str.	•		•	○
STE-FV	str.			•	•
ADER-FD	str.	•			
Taylor-FD	str.	•	•		
3D methods	**str.**	**Globally LEE**	**Locally LEE**	**EE**	**N.-S.**
ADER-DG	unstr.	•	•		
ADER-FV	unstr.	•	•	•	•
ADER-FV	str.	•			
STE-FV	str.			•	•
ADER-FD	str.	•			
Taylor-FD	str.	•	•		

Table 2.1: Equations and methods that are implemented in the domain decomposition framework. The viscous fluxes in the structured 2D ADER-FV Navier-Stokes solver are only implemented up to $\mathcal{O}2$.

time stepping", used e.g., in the *Tau* solver by the DLR) could be integrated into the coupling framework.
In order to give accurate sound level predictions for acoustic waves traveling through nonlinear and linear domains over large distances and long periods of time, a high resolution of the waves must be ensured. Numerical dissipation and dispersion must be kept low. Also, the entire range of scales from sound generation to sound propagation into the far field should be captured. Methods with a high order discretization in space and time have the potential to fulfill these requirements. Provided that the numerical error of a calculation is required to drop below a certain threshold, high order methods are more efficient in terms of CPU time and memory and thus cheaper than than low-order solvers (Dumbser [18]). This also implies, that low-order solvers may even fail for especially challenging calculations for which only a very small error can be accepted: Even if the mesh is greatly refined, the desired accuracy would never be obtained in a reasonable time as the time step decreases and thus the number or iterations increases dramatically.
Although the following methods originate from the so-called ADER class, it shall be emphasized, that this is no requirement in order to fit into the proposed coupling methodology. Any other explicit solver for hyperbolic problems could be integrated without much effort. The order of accuracy in space and time of the implemented methods can be chosen arbitrarily. This includes a great flexibility: The scheme can also be switched to lower order in domains, where the elements must be very fine due to the local geometry or due to a boundary layer.
It is important for the coupling strategy, that the most suitable methods are available where they are needed. Because of this, a whole zoo of numerical schemes (DG, FV, FD) is implemented for nonlinear and linear equations on structured and unstructured grids. While the implemented methods and equations are listed in Table 2.1, the advantages and drawbacks are given for each method separately (Tables 2.2, 2.3 and 2.4). The tables include properties of the particular methods but also the general characteristics of their class.
In the following, the ADER class methods are presented briefly. After that, particular attention is devoted to the so-called STE-FV method, which will be explained and validated in detail.
One remark regarding the use of the term "order" and its symbol $\mathcal{O}$: Throughout this work, an *nth* order method or $\mathcal{O}n$ method denotes a scheme, which shows an experimental order of convergence of n for smooth problems. However, the same term will be used for the respective method applied to a problem containing a discontinuity! Of course in this case, *nth* order of convergence

would *not* be obtained in an experiment. Nevertheless, this expression is used in order to characterize and distinguish calculations. This is also done for DG methods, for which the polynomial degree of the elements is normally used (a scheme with $P3$ elements for example denotes an $\mathcal{O}4$ method here).

2.1.1 ADER Discontinuous Galerkin Schemes

To give an overview of DG methods, the scalar conservation law

$$u_t + \vec{\nabla} \cdot \vec{f}(u) = 0 \tag{2.1}$$

is considered. A weak formulation is obtained locally for a control volume Ω by multiplication with a test function Φ_k and integration over Ω:

$$\int_\Omega \Phi_k u_t \; dV + \int_\Omega \Phi_k \; \vec{\nabla} \cdot \vec{f}(u) \; dV = 0. \tag{2.2}$$

Furthermore, an integration by parts yields

$$\int_\Omega \Phi_k u_t \; dV + \oint_{\partial\Omega} \Phi_k \left(\vec{f}(u) \cdot \vec{n} \right) \; dS - \int_\Omega \vec{f}(u) \cdot \vec{\nabla} \Phi_k \; dV = 0. \tag{2.3}$$

The data inside each cell are represented in form of piece-wise polynomials with

$$u_h(\vec{X}, t) = \sum_{l=1}^{N} \hat{u}_l(t) \Phi_l(\vec{X}), \;\; for \;\; \vec{X} \in \Omega \tag{2.4}$$

being the approximated solution inside Ω. The space of polynomials is spanned by the set of basis functions $\{\Phi_l\}_{l=1,\ldots,N}$, where N is given by the polynomial degree p and the spatial dimension d:

$$N := N(p, d) = \prod_{j=1}^{d} \frac{p + j}{j}. \tag{2.5}$$

The DG solution may jump at the cell interfaces, therefore a numerical flux function

$$g_{\vec{n}} \approx \vec{f}(u) \cdot \vec{n} \tag{2.6}$$

replaces the exact flux in the boundary integral of equation 2.3. Inserting u_h and $g_{\vec{n}}$, equation 2.3 yields

$$\int_\Omega \sum_{l=1}^{N} \Phi_k \Phi_l (\hat{u}_l)_t \; dV + \oint_{\partial\Omega} \Phi_k \; g_{\vec{n}}(u_h^+, u_h^-) \; dS - \int_\Omega \vec{f}(u_h) \cdot \vec{\nabla} \Phi_k \; dV = 0, \tag{2.7}$$

where u_h^- denotes the state at the interior edge of the considered cell and u_h^+ the state at the edge of the adjacent element. Hence, a set of N equations result from (2.7), for $\{\Phi_k\}_{l=1,\dots,N}$.
An introduction to DG methods can be found in Hesthaven et al. [44] and Becker [5]. The differences in the particular DG methods lie in the evaluation of the single volume and boundary integrals. One of the inherent difficulties of DG class methods is to find efficient integration procedures in order to keep the computational costs down.
A very common approach for explicit time integration is the Runge-Kutta scheme. For an overview of RK time integration in DG methods, see Cockburn and Shu [9–14] and Qiu et al. [82]. However, the efficiency of a RK time discretization decreases strongly when the order of accuracy is greater than four and the so-called Butcher barrier [8] kicks in. In order to circumvent this problem, Dumbser et al. [18, 21–23] applied the ADER approach of Toro et al. [50, 90–93, 109, 110, 113, 116, 117] to the DG method for linear and nonlinear hyperbolic systems. The ADER idea is explained in more detail in section 2.1.2. Other time discretizations can be based for example on Lax-Wendroff type methods (Qiu et al. [80, 81]).
Generally, the space and time integrals can be approximated by numerical integration rules, such as Gauss quadrature and hence require integration points in time and space. This becomes quite computationally expensive for high order elements in 3D. Gauss integration yields $(p+1)^d$ Gauss integration points for each volume integral. For a sixth order scheme ($p = 5$), this means 216 integration points per element only for the volume integral!
Dumbser proposed a method which is quadrature-free in space and time for linear equations (linear ADER-DG-QF). For nonlinear hyperbolic systems, he constructed several versions: The ADER-DG-SX method (**s**tate **e**xpansion, numerical quadrature in space and time), the ADER-DG-FX method (numerical quadrature in space, **f**lux **e**xpansion and analytical integration in time) and the ADER-DG-QF method (**q**uadrature-**f**ree in space and time). All of those ADER-DG methods and some more variants (e.g., the local time stepping version LTS-DG and reconstructed DG schemes) are integrated in the coupling framework. However, for the sake of simplicity, only the term "ADER-DG" will be used in the following and refers in case of the linearized Euler equations always to the ADER-DG-QF method, while in the nonlinear case, ADER-DG refers to the ADER-DG-SX scheme.
Table 2.2 gives a review of the more and the less favorable properties of the ADER-DG methods. As DG schemes are strongly in the focus of the ongoing research, this list cannot be considered as ultimate. Recent developments

	Advantages
+	Polynomial representation of the solution inside one cell.
+	Highly suitable for unstructured grids and thus for problems involving complex geometries: + High order ($\gg \mathcal{O}3$) without reconstruction. + High convergence rates maintained even on highly irregular grids.
+	A single high order cell can resolve substructures, hence the total number of elements can be reduced in the computational domain.
+	A discontinuous solution is allowed over element interfaces: Therefore, advection dominated problems and nonlinear effects such as shocks can be treated robustly.
+	Very local, enabling the efficient implementation of + local time stepping, + hp-adaptivity, + parallelization.
+	Easy treatment of boundary conditions.
+	Reduces to first order FV scheme for $\Phi_l = 1$.
+	One-step method: Order of accuracy not limited by Butcher barrier.
	Drawbacks
-	In principal more degrees of freedom than for example FD methods.
-	Great practical and theoretical effort required regarding the implementation and the techniques for numerical integration in order to keep the computational costs down for high order of accuracy.
-	Less robust regarding under-resolved flow features.
-	High order shock capturing yet to be explored.

Table 2.2: Advantages and drawbacks of ADER-DG methods.

for high order DG class methods consider space-time elements in an explicit (STE-DG methods with local time stepping and hp-adaptivity, see Gassner et al. [36, 37]) or an implicit framework (dual time stepping with space-time elements, van der Vegt et al. [54, 121]). Furthermore, the construction of diffusion fluxes for the DG schemes has been investigated (Gassner et al. [35], Van Leer [123], Houston [47]).

2.1.2 ADER Finite Volume Schemes

Finite volume (FV) methods are based on the weak solution of conservation laws in integral form. To give an example, the scalar conservation equation (2.1) is considered and integrated:

$$\int_{t^n}^{t^{n+1}} \int_\Omega u_t \; dV dt + \int_{t^n}^{t^{n+1}} \int_\Omega \vec{\nabla} \cdot \vec{f}(u) \; dV dt = 0. \tag{2.8}$$

By introducing the mean-value

$$\bar{u} = \frac{1}{|\Omega|} \int_\Omega u \; dV \tag{2.9}$$

of a control volume Ω (grid cell), equation (2.8) yields

$$\bar{u}^{n+1} = \bar{u}^n - \frac{1}{|\Omega|} \int_{t^n}^{t^{n+1}} \int_{\partial\Omega} \left(\vec{f}(u(\vec{X}, t)) \cdot \vec{n} \right) \; dS dt, \tag{2.10}$$

the so-called "evolution equation for integral mean-values", which is still the exact weak solution of (2.1). In an analogy to (2.6), the unknown exact flux in normal direction of the cell boundary can be replaced by a numerical flux function $g_{\vec{n}}$ (the normal vector $\vec{n}$ must not be confused with the time level index n) depending on the states to the left and to the right of the cell interface:

$$\bar{u}^{n+1} = \bar{u}^n - \frac{1}{|\Omega|} \int_{t^n}^{t^{n+1}} \int_{\partial\Omega} g_{\vec{n}}(u_h^+, u_h^-) \; dS dt \tag{2.11}$$

The most simple first order scheme in space and time is obtained if the mean-values of the own ($\bar{u}_h^-$) and the neighbor cell ($\bar{u}_h^+$) are chosen as the values for the flux calculation at the cell interfaces. To give an example in 1D, equation (2.11) then yields

$$\bar{u}_i^{n+1} = \bar{u}_i^n - \frac{\Delta t}{\Delta x}(g_{\vec{n}}(u^{n,+}_{i+\frac{1}{2}}, u^{n,-}_{i+\frac{1}{2}}) - g_{\vec{n}}(u^{n,+}_{i-\frac{1}{2}}, u^{n,-}_{i-\frac{1}{2}})), \tag{2.12}$$

$$= \bar{u}_i^n - \frac{\Delta t}{\Delta x}(g_{\vec{n}}(\bar{u}^n_{i+1}, \bar{u}^n_i) - g_{\vec{n}}(\bar{u}^n_i, \bar{u}^n_{i-1})), \tag{2.13}$$

for a cell i with the left and right interfaces at $i - \frac{1}{2}$ and $i + \frac{1}{2}$ and the length Δx.

In order to compute the fluxes $g_{\vec{n}}$ at the interfaces, local Riemann problems are solved by using either exact (Godunov [39]) or approximate solvers (e.g., Roe [85], **H**arten, **L**ax and Van **L**eer [42]). Toro [112] gives a detailed overview of these so-called Godunov type methods and their basic building block, the Riemann solvers. An introduction to these and other significant issues (discretization, boundary conditions, etc.) of standard FV schemes is given also by Blazek [7].
However, first order FV schemes suffer from high numerical damping, which implies a large number of domain elements and a very small time step in order to achieve a more or less acceptable time accurate solution. Second and higher order schemes usually need slope limiters, for example from the TVD (total variation diminishing) type in order to avoid oscillations at discontinuities. A well-known representative of the TVD methods is the second order MUSCL scheme by Van Leer [122], which employs a second order time discretization and a (slope-limited) linear reconstruction at the cell interfaces. A higher order polynomial reconstruction (ENO: Harten et al. [41], WENO: Liu [66], Shu et al. [4, 48, 51]) can lead to very high order of spatial accuracy, but proved to be cumbersome on unstructured grids. Furthermore, the Butcher barrier [8] puts a limit on the order of the time integration (≤ 4) if it is done by TVD or classical Runge-Kutta methods (Shu et al. [40]).
The ADER (**A**rbitrary high order using **DER**ivatives) approach of Toro et al. [50, 88, 90–93, 109, 110, 113, 116, 117] is able to circumvent this barrier: In order to obtain a highly accurate approximation of the flux in equation (2.11), the so-called Generalized Riemann Problem (GRP) is solved. Instead of feeding constant data on both sides of the interface into a conventional Riemann problem, the GRP considers the distribution of the solution in form of a polynomial that has been obtained by a previous reconstruction for each element. The local $GRP\left(P^+(x), P^-(x)\right)$

$$\begin{aligned} u_t + f(u)_x &= 0 \\ u(x,0) &= \begin{cases} P^-(x), & x < 0 \\ P^+(x), & x > 0. \end{cases} \end{aligned} \tag{2.14}$$

is then split into a conventional nonlinear Riemann Problem for the reconstructed states directly at the interface plus linearized Riemann Problems for the spatial derivatives (Toro et al. [111, 114]). Knowing the state u_{GRP} and its derivatives, the Riemann solution on the interface can be expanded into a

Taylor series in time:

$$u_{GRP}(t^n + \tau) = u_{GRP}(t^n) + \sum_{k=1}^{\mathcal{O}-1} \frac{\tau^k}{k!} \left(\frac{\partial u^k(t^n)}{\partial^k t} \right)_{GRP}, \quad (2.15)$$

$$= u_{GRP}(t^n) + \sum_{k=1}^{\mathcal{O}-1} CK_k \left(\left(\frac{\partial^p u}{\partial x^p} \right)_{GRP} \right), \quad (2.16)$$

where the Cauchy-Kovalevskaja Procedure CK_k creates the kth time derivative by using the known spatial derivatives (see also section 2.4 for details) and $\mathcal{O}$ denotes the desired order of the method in space and time.
Returning to equation (2.11), an arbitrary order single time step method is obtained by using Gauss quadrature for both the space and the time integral and evaluating the numerical flux at these discrete integration points. To continue the 1D example of (2.12), the arbitrary high order version would therefore read

$$\bar{u}_i^{n+1} = \bar{u}_i^n - \frac{\Delta t}{\Delta x} \sum_{\xi=1}^{n_\xi} \omega_\xi \left(g_{\vec{n}}(u_{GRP}(t_\xi, x_{i+\frac{1}{2}})) - g_{\vec{n}}(u_{GRP}(t_\xi, x_{i-\frac{1}{2}}) \right), \quad (2.17)$$

with n_ξ and ω_ξ denoting the number and the weights of the Gauss integration points ξ on the time interval $\Delta t = t^{n+1} + t^n$ for a given order of accuracy $\mathcal{O}$. For 2D and 3D elements, also Gauss integration points in space need to be considered.
For structured grids, Schwartzkopff [88] combined this so-called ADER-SX (**s**tate **e**xpansion, numerical quadrature in space and time) method with a dimension-by-dimension ENO reconstruction for the nonlinear Euler equations. He also constructed a version where the time integration is done analytically by performing the Taylor expansion with the flux itself (ADER-FX), instead with the Riemann state. A very efficient *fast*ADER formulation was implemented by him for the linearized Euler equations, where the ADER-FV framework is condensed to a mere matrix-vector multiplication form. The coefficient matrix contains the reconstruction step, the GRP solution and the Jacobians. This scheme requires constant Jacobians, grid spacings, time steps and a linear reconstruction in space. In the following, the term "ADER-FV on structured grids" refers to both the nonlinear ADER-SX and the *fast*ADER implementation for linear equations.
On unstructured grids, Dumbser et al. [19,20] found a way to cope with both the reconstruction problem for a high order spatial discretization and the Butcher barrier for the time discretization. The one-step ADER-FV method of arbitrary order in space and time differs greatly from the structured version: The

	Advantages
+	High order schemes are easy to implement for structured grids.
+	High order reconstruction ($\gg \mathcal{O}3$) on regular, structured grids as well as on unstructured grids.
+	Fast algorithms and reconstructions for structured grids.
+	Very robust regarding discontinuities due to integral conservation and fluxes based on the physics of nonlinear wave propagation.
+	Well understood limiting and shock treatment techniques.
+	No additional volume integrals as for DG, limited integration effort.
+	One-step method: Order of accuracy not limited by Butcher barrier.
	Drawbacks
-	Only piecewise constant data in one cell. High order requires high order reconstruction.
-	Reconstruction stencil reduces locality.

Table 2.3: Advantages and drawbacks of ADER-FV methods.

reconstruction operator, developed originally in a DG framework, uses hierarchical orthogonal basis functions and delivers entire high order polynomials for a cell instead of point values. Furthermore, the reconstruction is performed in a reference system and not in physical coordinates, which avoids ill-conditioned scaling matrices. The WENO type reconstruction is done in characteristic variables for nonlinear problems. A space-time Taylor series for the evolution of the state and the physical fluxes along with a special numerical flux treatment at the cell interfaces results in a less costly quadrature-free formulation in space and time. In the following, the term "Rec-FV" (reconstructed FV) is used for the nonlinear and the linear version of the ADER-FV method on unstructured grids. Note that the implementation framework for this scheme resembles very much the one of the ADER-DG method, hence all coupling techniques for ADER-DG domains discussed in section 2.3 will also apply to Rec-FV elements.
The properties of the ADER-FV methods on structured and unstructured grids are shown in Table 2.3.

2.1.3 ADER and Taylor Finite Difference Schemes

The arbitrary high order ADER-FD (or generalized Lax-Wendroff-type scheme, LW-FD) method by Lörcher et al. [69] is based on a Taylor expansion in time:

$$\vec{U}^{n+1} = \vec{U}^n + \sum_{k=1}^{\infty} \frac{(\Delta t)^k}{k!} \frac{\partial^k \vec{U}^n}{\partial t^k}. \tag{2.18}$$

If it is truncated after the first three terms and the the time derivatives are replaced by the spatial derivatives from central differences, the second order Lax-Wendroff scheme [61] is obtained. However, if it is truncated at at a higher order, the missing time derivatives can be replaced also with space derivatives by the before mentioned CK procedure (see also section 2.4). To give an example, the k*th* time derivative for the linearized Euler equations is

$$\frac{\partial^k \vec{U}}{\partial t^k} = (-1)^k \left(\underline{\underline{A}} \frac{\partial}{\partial x} + \underline{\underline{B}} \frac{\partial}{\partial y} + \underline{\underline{C}} \frac{\partial}{\partial z} \right)^k \vec{U}, \tag{2.19}$$

with $\vec{U}$ being the perturbation state vector and $\underline{\underline{A}}$, $\underline{\underline{B}}$ and $\underline{\underline{C}}$ being the linearized Jacobians. The space derivatives themselves are obtained by arbitrary high order polynomial interpolations, which is especially easy on structured grids. If the order of the method is even ($\mathcal{O} = 2, 4, 6, ...$), the interpolation is symmetric and results in a central scheme, independent of the direction of wave propagation. Having turned out to be the most efficient approach for linear problems, only even order in space and time is actually implemented. However, by using upwind-methodology and solving Riemann problems, the extension to odd schemes is straight-forward.
Nonlinear regions with strong gradients are always problematic for FD methods and have to be treated specially with sophisticated ENO and WENO techniques [4, 41, 51, 66, 83]. Because it is the intention of the coupling framework to exploit the positive properties of each class of numerical methods, nonlinear effects such as shocks shall be not considered here. Instead, the ADER-FD approach is implemented and used for the linearized Euler equations only, concentrating on an efficient scheme for linear wave propagation over long distances. A detailed discussion of general FD schemes is given by Hirsch [45, 46]. In contrast to methods like the DRP (Dispersion Relation Preserving) scheme by Tam et al. [108], which uses fourth order Runge-Kutta time discretization and a sixth order difference stencil in space, the one-step ADER-FD method does not underlie the Butcher barrier. Furthermore and very similar to the linear ADER-FV method, it can be formulated in a very compressed Matrix-Vector

multiplication form. The 3D version of the ADER-FD method for linearized Euler equations reads then

$$\vec{U}_{ijk}^{n+1} = \vec{U}_{ijk}^{n} - \sum_{ii=-\frac{\mathcal{O}}{2}}^{\frac{\mathcal{O}}{2}} \sum_{jj=-\frac{\mathcal{O}}{2}}^{\frac{\mathcal{O}}{2}} \sum_{kk=-\frac{\mathcal{O}}{2}}^{\frac{\mathcal{O}}{2}} \underline{\underline{C}}^{\star}_{ii,jj,kk} \vec{U}_{i+ii,j+jj,k+kk}^{n}, \tag{2.20}$$

where the coefficient matrix $\underline{\underline{C}}^{\star}$ is constant and the same for every grid point on a Cartesian grid. At the same time, equation (2.20) shows the drawback of this method: For high orders and especially in 3D, the number of points required for the interpolation stencil is quite high.
The Taylor-FD method by Lörcher [67] starts again from the Taylor expansion (2.18), but replaces the time derivatives by applying a discrete space operator $\underline{\underline{\Phi}}$ to the discrete solution. For the linearized Euler equations, the time evolution of the state vector $\vec{U}$ is then given by the one-step scheme

$$\vec{U}^{n+1} = \vec{U}^{n} + \sum_{k=1}^{\mathcal{O}} \frac{\Delta t^k}{k!} \underline{\underline{\Phi}}^{k} \vec{U}^{n}, \tag{2.21}$$

where the differential operator can be defined by differentiating the original PDE with respect to the time:

$$\begin{aligned} \frac{\partial^k}{\partial t^k}\vec{U} &= -\underline{\underline{A}}\left(\frac{\partial^{k-1}}{\partial t^{k-1}}\vec{U}\right)_x - \underline{\underline{B}}\left(\frac{\partial^{k-1}}{\partial t^{k-1}}\vec{U}\right)_y - \underline{\underline{C}}\left(\frac{\partial^{k-1}}{\partial t^{k-1}}\vec{U}\right)_z \\ &=: \underline{\underline{\Phi}}^{k}\vec{U}. \end{aligned} \tag{2.22}$$

Hence, by calculating the first order derivatives $\frac{\partial \vec{U}}{\partial x}$, $\frac{\partial \vec{U}}{\partial y}$ and $\frac{\partial \vec{U}}{\partial z}$ in a dimension-by-dimension manner, the k*th* time derivative can be created recursively. The computation of arbitrary high order space derivatives can be done in different ways, for example with the DRP finite differences by Tam et al. [108], which are optimized for wave propagation. Note that now only difference stars are required, not the entire volume stencil which is needed for ADER-FD. In the following, the term "Taylor-FD" refers to the implementation of the method with DRP differences in space.
Table 2.4 summarizes the properties of the ADER-FD and Taylor-FD methods.

	Advantages
+	High order schemes ($\gg \mathcal{O}3$) are easy to implement for structured grids.
+	Very fast algorithms and excellent vectorization.
+	Very efficient especially for linear problems (e.g., wave propagation).
+	Simple treatment of ADER-FD boundary conditions.
+	Treatment of high order source terms easy for Taylor-FD.
+	Optimized interpolation coefficients for Taylor-FD.
+	One-step methods: Order of accuracy not limited by Butcher barrier.
	Drawbacks
-	Highly structured, regular grids are required. Difficult mesh treatment and cumbersome grid design near complex geometries.
-	Severe problems in the presence of discontinuities (e.g., shocks).
-	Large interpolation stencils required for ADER-FD methods.
-	Decrease in efficiency for odd orders of accuracy.
-	Taylor-FD unstable for schemes of time integration order 2, 6, 10 ... (but stable for order 4, 8, 12, ...).
-	Boundary conditions require time derivatives for Taylor-FD.

Table 2.4: Advantages and drawbacks of ADER- and Taylor-FD methods.

2.1.4 The Space-Time Expansion Finite Volume Method

In order to obtain an efficient and robust high order scheme for nonlinear equations, building blocks from the unstructured Rec-FV method by Dumbser, from the STE-DG method by Gassner et al. [36, 37] and from the structured ADER-FV method by Schwartzkopff have been combined to a new scheme, the space-time expansion finite volume method (STE-FV) on Cartesian grids.
The methods so far cover a large spectrum of desired properties in the domain decomposition framework. However, an efficient solver for nonlinear equations on Cartesian grids had still been missing. Although the structured ADER-FV method by Schwartzkopff in section 2.1.2 had been constructed for those kinds of grids, it possesses some severe drawbacks:
First of all, the method depends on the solution of a GRP at the edge of each cell. Only exact (e.g., Godunov) or approximative Riemann solvers (e.g., MUSTA, see Toro et al. [118]) can be used, which determine the Riemann state itself. Hence, this excludes robust Riemann solvers such as HLLE, which compute directly the flux and are especially favorable at strong shocks and rarefaction waves. Furthermore, the reconstruction has to be performed for every spatial Gauss integration point for the state expansion version, which becomes computationally expensive for higher orders. Last but not least, the implemented ENO reconstruction involves complicated if-statements and thus decreases performance, especially on vector computers.
The proposed STE-FV method has been designed to overcome these problems and aims at

- high order of accuracy for the Navier-Stokes and the Euler equations,
- an uncomplicated 2D and 3D implementation,
- robust and fast WENO reconstruction,
- efficient performance on Cartesian grids,
- good vectorization properties.

It is stressed, that the formulation of the STE-FV method on structured grids differs in several aspects from the Rec-FV method by Dumbser: The dimension-by-dimension WENO reconstruction in characteristic variables delivers point-values at the barycenter of each cell rather than an entire polynomial. No least-squares ansatz is necessary here, as the system of equations for the reconstruction is exactly determined on a Cartesian grid. Also, a transformation

into a non-dimensional reference space is not required on structured grids. Furthermore, the scheme involves explicit numerical quadrature for space and time integration instead of a quadrature-free approach: For simple finite volume rectangles and hexahedrons, an optimal low number of quadrature points can be chosen, when Gauss integration is used. Also because an accurate and flexible treatment of the fluxes (usage of well-known Riemann solvers, e.g., Godunov, Roe, HLLE) and a most simple implementation are desirable, Gauss quadrature is favored. A regular Riemann problem is solved at every space-time Gauss integration point. The values at these points are obtained by a space-time Taylor expansion in the barycenter of each cell. Harten et al. [41] was one of the first to propose such a space-time Taylor expansion in the barycenter within an ENO finite volume framework. The STE-FV method can be structured into single, consecutive building blocks:

1. High order central or WENO reconstruction.
2. Cauchy-Kovalevskaja procedure.
3. Space-time Taylor expansion.
4. Solving the flux at the integration points.
5. Space-time integration of the fluxes.
6. Updating the mean-values.

In the following, the fully-discrete STE-FV scheme is presented and the most important building blocks and algorithms are explained. In order to validate the method and its shock-capturing features, numerical convergence studies are performed and the method is applied to various test cases in 2D and 3D, e.g the Double mach reflection problem.

Evolution equation and time update Applying numerical Gauss integration in space and time to a Cartesian hexahedron in 3D, equation (2.11) yields

$$
\begin{aligned}
\bar{u}_{ijk}^{n+1} &= \bar{u}_{ijk}^{n} - \frac{1}{|\Omega_{ijk}|} \int_{t^n}^{t^{n+1}} \int_{\partial\Omega} \left(\vec{f}(u(\vec{X},t)) \cdot \vec{n} \right) \, dS dt \\
&= \bar{u}_{ijk}^{n} - \frac{1}{|\Omega_{ijk}|} \sum_{\xi=1}^{n_\xi} \omega_\xi \int_{\partial\Omega} \left(\vec{f}(u(\vec{X},t_\xi)) \cdot \vec{n} \right) \, dS \\
&= \bar{u}_{ijk}^{n} - \frac{1}{|\Omega_{ijk}|} \sum_{\xi=1}^{n_\xi} \omega_\xi \qquad (2.23) \\
&\cdot \quad [\sum_{\lambda=1}^{n_\lambda} \omega_{1_\lambda} f_1(u(\vec{X}_{\lambda_{i+\frac{1}{2},j,k}}, t_\xi)) - \sum_{\lambda=1}^{n_\lambda} \omega_{1_\lambda} f_1(u(\vec{X}_{\lambda_{i-\frac{1}{2},j,k}}, t_\xi)) \\
&+ \quad \sum_{\lambda=1}^{n_\lambda} \omega_{2_\lambda} f_2(u(\vec{X}_{\lambda_{i,j+\frac{1}{2},k}}, t_\xi)) - \sum_{\lambda=1}^{n_\lambda} \omega_{2_\lambda} f_2(u(\vec{X}_{\lambda_{i,j-\frac{1}{2},k}}, t_\xi)) \\
&+ \quad \sum_{\lambda=1}^{n_\lambda} \omega_{3_\lambda} f_3(u(\vec{X}_{\lambda_{i,j,k+\frac{1}{2}}}, t_\xi)) - \sum_{\lambda=1}^{n_\lambda} \omega_{3_\lambda} f_3(u(\vec{X}_{\lambda_{i,j,k-\frac{1}{2}}}, t_\xi))].
\end{aligned}
$$

Here, $|\Omega_{ijk}| = \Delta x \cdot \Delta y \cdot \Delta z$ denotes the volume of the cell, ω_{1_λ}, ω_{2_λ}, ω_{3_λ} are the Gauss integration weights of the spatial integration point λ in x-, y- and z-direction, ω_ξ is the integration weight of time integration point ξ. The exact fluxes in x-, y- and z-direction (f_1, f_2, f_3) are functions of the unknown Riemann state u at the position of the space-time Gauss integration points on the cell surface. $\vec{X}_{\lambda_{i+\frac{1}{2},j,k}}$ denotes the spatial position on the interface between cells (i,j,k) and $(i+1,j,k)$ and t_ξ is literally the Gauss "point in time". Using a conventional Riemann solver (e.g., Roe, HLLE), the exact fluxes are replaced by numerical fluxes $g_{\vec{n}}$ (see equation (2.6)). They depend on the left (-) and right (+) states at the interface, which must be given by a preceding reconstruction.

Inserted in (2.23), one gets

$$
\begin{aligned}
\bar{u}_{ijk}^{n+1} &= \bar{u}_{ijk}^{n} - \frac{1}{|\Omega_{ijk}|} \sum_{\xi=1}^{n_\xi} \omega_\xi \\
&\cdot \Big[\sum_{\lambda=1}^{n_\lambda} \omega_{1_\lambda} g_{n_1}(u^+(\vec{X}_{\lambda_{i+\frac{1}{2},j,k}}, t_\xi), u^-(\vec{X}_{\lambda_{i+\frac{1}{2},j,k}}, t_\xi)) \\
&- \sum_{\lambda=1}^{n_\lambda} \omega_{1_\lambda} g_{n_1}(u^+(\vec{X}_{\lambda_{i-\frac{1}{2},j,k}}, t_\xi), u^-(\vec{X}_{\lambda_{i-\frac{1}{2},j,k}}, t_\xi)) \\
&+ \sum_{\lambda=1}^{n_\lambda} \omega_{2_\lambda} g_{n_2}(u^+(\vec{X}_{\lambda_{i,j+\frac{1}{2},k}}, t_\xi), u^-(\vec{X}_{\lambda_{i,j+\frac{1}{2},k}}, t_\xi)) \\
&- \sum_{\lambda=1}^{n_\lambda} \omega_{2_\lambda} g_{n_2}(u^+(\vec{X}_{\lambda_{i,j-\frac{1}{2},k}}, t_\xi), u^-(\vec{X}_{\lambda_{i,j-\frac{1}{2},k}}, t_\xi)) \\
&+ \sum_{\lambda=1}^{n_\lambda} \omega_{3_\lambda} g_{n_3}(u^+(\vec{X}_{\lambda_{i,j,k+\frac{1}{2}}}, t_\xi), u^-(\vec{X}_{\lambda_{i,j,k+\frac{1}{2}}}, t_\xi)) \\
&- \sum_{\lambda=1}^{n_\lambda} \omega_{3_\lambda} g_{n_3}(u^+(\vec{X}_{\lambda_{i,j,k-\frac{1}{2}}}, t_\xi), u^-(\vec{X}_{\lambda_{i,j,k-\frac{1}{2}}}, t_\xi)) \Big].
\end{aligned}
\tag{2.24}
$$

So far, due to simplicity reasons, only the scalar conservation law (2.1) has been used for the derivation of the scheme. When a system of equations is considered, the fluxes are vectors. In case of the Navier-Stokes equations, f_1, f_2 and f_3 correspond with $\vec{F}(\vec{U}) - \vec{F}^v(\vec{U}, \vec{\nabla}\vec{U})$, $\vec{G}(\vec{U}) - \vec{G}^v(\vec{U}, \vec{\nabla}\vec{U})$ and $\vec{H}(\vec{U}) - \vec{H}^v(\vec{U}, \vec{\nabla}\vec{U})$, see section 2.2.3. Note that also the gradient of $\vec{U}$ is required for the viscous fluxes $\vec{F}^v$, $\vec{G}^v$ and $\vec{H}^v$, which are computed by the dGRP Riemann solver for diffusion fluxes by Gassner et al. [35]. This implies, that – in addition to the states – also left (-) and right (+) gradients needs to be reconstructed at the integration points.

High order reconstruction at the space-time integration points The time update (2.24) will be high order accurate, when on the one hand the Gauss integrations are performed with the desired order and when on the other hand the arguments of the numerical fluxes are provided with high accuracy. The

latter must be reconstructed from the mean-values of the cells inside the computational domain. As the number of integration points at the cell interfaces increases for rising order of accuracy, it is cumbersome to perform a reconstruction and a CK procedure for every single one of them. Therefore, a different strategy is pursued: Point values for state *and* spatial derivatives are interpolated from a stencil of mean-values $\vec{\bar{U}}_{ijk}$ only at the position $\vec{X}_B$ of the cell *barycenters*. To give a simple example, one can use the central interpolation operator from section 2.3.3

$$\frac{\partial^{p+q+r}\vec{U}(\vec{X}_B,t^n)}{\partial x^p\partial y^q\partial z^r} = \sum_{i=1}^{n_S}\sum_{j=1}^{n_S}\sum_{k=1}^{n_S}\frac{\partial^{p+q+r}L_{ijk}(\vec{X}_B)}{\partial x^p\partial y^q\partial z^r}\cdot\vec{\bar{U}}_{ijk} \tag{2.25}$$

for smooth problems. The actual implementation and general high order interpolation with WENO reconstruction is described in detail in the subsequent section.
The CK procedure (see section 2.4) is used afterwards to provide also time derivatives for the barycenter:

$$\frac{\partial^{p+q+r+k}\vec{U}(\vec{X}_B,t^n)}{\partial x^p\partial y^q\partial z^r\partial t^k} = CK\left(\frac{\partial^{p+q+r}\vec{U}(\vec{X}_B,t^n)}{\partial x^p\partial y^q\partial z^r}\right),\ \forall\ 0\le p+q+r+k\le\mathcal{O}-1. \tag{2.26}$$

These derivatives are used in a general space-time Taylor expansion around the barycenter:

$$\vec{U}(\vec{X},t) = \vec{U}(\vec{X}_B,t^n) + \sum_{k=1}^{\mathcal{O}-1}\frac{1}{k!}\left((t-t^n)\frac{\partial}{\partial t} + (\vec{X}-\vec{X}_B)\cdot\vec{\nabla}\right)^k\vec{U}(\vec{X}_B,t^n). \tag{2.27}$$

The expansion is made also for the gradients $\vec{\nabla}\vec{U} = grad(\vec{U})$,

$$\vec{\nabla}\vec{U}(\vec{X},t) = \vec{\nabla}\vec{U}(\vec{X}_B,t^n) + \sum_{k=1}^{\mathcal{O}-2}\frac{1}{k!}\left((t-t^n)\frac{\partial}{\partial t} + (\vec{X}-\vec{X}_B)\cdot\vec{\nabla}\right)^k\vec{\nabla}\vec{U}(\vec{X}_B,t^n), \tag{2.28}$$

but with one order of accuracy less than the initial interpolation of the states, as the gradient is already the first derivative of the reconstruction polynomial (which can only be derived $\mathcal{O}-1$ times).
By evaluating these Taylor expansions at the space-time positions $(\vec{X}_\lambda, t_\xi)$ of the Gauss integration points, one finally obtains the reconstructed states and gradients at the cell interfaces (Fig. 2.1). These can now be directly used as input for the Riemann solvers in order to perform the flux computations.

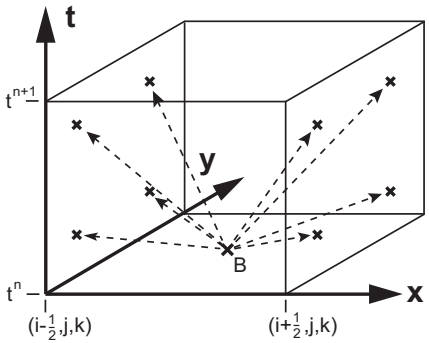


Figure 2.1: An $\mathcal{O}4$ space-time element in 2D. Only the Gauss integration points for the faces $i - \frac{1}{2}$ and $i + \frac{1}{2}$ are depicted.

High order central reconstruction Central reconstructions with symmetric interpolation stencils are favorable for smooth solutions. Because mixed space derivatives are needed for the CK procedure, cheap "finite difference star"-like reconstructions cannot be applied. The reconstruction must be based on full two or three-dimensional volume stencils. They produce an exactly determined system of equations for the multi-dimensional interpolation polynomial. Lörcher et al. [69] showed, that it is possible to reduce the stencil size by omitting the monomials of higher order than the desired accuracy. However, this option has not been considered here in favor of better stability and robustness. Due to the Cartesian grid, the interpolation procedure can be performed in a dimension-by-dimension manner. This means a greatly reduced number of operations in 2D and 3D compared to a full volume stencil reconstruction for each element:

$$n_{Ops,FullStencil} = n_{Elem} \cdot n_{Ops,1D}^{d}, \tag{2.29}$$

$$n_{Ops,DimByDim} = n_{Elem} \cdot n_{Ops,1D} \cdot d, \tag{2.30}$$

with n_{Elem} being the number of reconstructed elements, d being the spatial dimension and $n_{Ops,1D}$ being the number of computational operations necessary for reconstructing one 1D element. To give an example, for a $\mathcal{O}5$ reconstruction, usually five 1D stencil points are needed. In 3D, the effort for a full-stencil reconstruction compared to a dimension-by-dimension reconstruction is

$\frac{n_{Ops,FullStencil}}{n_{Ops,DimByDim}} = \frac{5^3}{5 \cdot 3} = \frac{125}{15} \approx 8.33!$
The polynomial interpolation from mean-values onto point values at the barycenter conserves the mean-values of the stencil cells and is explained in the interpolation section 2.3.3. It is emphasized, that the full-stencil and the dimension-by-dimension reconstructions are equivalent and deliver an identical polynomial. In 1D, the central interpolation for the point value of the state and the spatial derivatives can be written as:

$$\frac{\partial^p \vec{U}(x_B)}{\partial x^p} = \sum_{i=1}^{n_S} \frac{\partial^p L_i(x_B)}{\partial x^p} \cdot \vec{\bar{U}}_i, \tag{2.31}$$

where the $\vec{\bar{U}}_i$'s are the mean-values of the n_S stencil cells necessary for the desired interpolation order and i denotes the local index of a stencil element in the computational domain. The cell-weight coefficients $\frac{\partial^p L_i(x_B)}{\partial x^p}$ are constant and can be computed in advance. For odd spatial orders, the interpolation stencil is always symmetrical towards the barycenter ($n_s = \mathcal{O}$). For even order schemes however, the number of necessary stencil cells produces two possible stencils with a slight up- or downwind bias:

$$\frac{\partial^p \vec{U}(x_B)}{\partial x^p}_{leftsided} = \sum_{i=1}^{n_S} \frac{\partial^p L_i(x_B)}{\partial x^p}_{leftsided} \cdot \vec{\bar{U}}_i, \tag{2.32}$$

$$\frac{\partial^p \vec{U}(x_B)}{\partial x^p}_{rightsided} = \sum_{i=2}^{n_S+1} \frac{\partial^p L_i(x_B)}{\partial x^p}_{rightsided} \cdot \vec{\bar{U}}_i. \tag{2.33}$$

In order to avoid instabilities, those two stencils are superposed to a single stencil with new cell-weights

$$\frac{\partial^p L_i^\star(x_B)}{\partial x^p} = \frac{1}{2} \cdot \left(\frac{\partial^p L_i(x_B)}{\partial x^p}_{leftsided} + \frac{\partial^p L_i(x_B)}{\partial x^p}_{rightsided} \right) \tag{2.34}$$

and an increased number of stencil cells $n_s^\star = \mathcal{O} + 1$, so the final stencil has an odd number of elements again:

$$\frac{\partial^p \vec{U}(x_B)}{\partial x^p} = \sum_{i=1}^{n_S^\star} \frac{\partial^p L_i^\star(x_B)}{\partial x^p} \cdot \vec{\bar{U}}_i. \tag{2.35}$$

In the following and for the purpose of uniformity, $n_s^\star$ and $L^\star$ will be denoted n_S and L also for the even order schemes.

The reconstruction for the entire computational domain is given in the following algorithm. The global indices of a domain element are denoted by (i, j, k), the fields marked with $\star$ and $\star\star$ are temporary fields that can be deleted after each reconstruction and $IS = \frac{n_s-1}{2}$. Note that also the first row of ghost elements is reconstructed in order to provide accurate boundary data.

Algorithm 2.1.1. ***Central dimension-by-dimension reconstruction***

FOR ALL(i,j,k)
$\left(\frac{\partial^p \vec{U}}{\partial x^p}\right)^{\star\star}_{ijk} = 0.0,\ \left(\frac{\partial^{p+q} \vec{U}}{\partial x^p \partial y^q}\right)^{\star}_{ijk} = 0.0,\ \left(\frac{\partial^{p+q+r} \vec{U}(\vec{X}_B)}{\partial x^p \partial y^q \partial z^r}\right)_{ijk} = 0.0$
END FOR ALL

DO r = 0,$\mathcal{O}$-1 ! z-derivatives
DO q = 0,$\mathcal{O}$-1-r ! y-derivatives
DO p = 0,$\mathcal{O}$-1-r-q ! x-derivatives

! Reconstruction in x-direction
DO k=-nGhost+1,kmax+nGhost
DO j=-nGhost+1,jmax+nGhost
DO i=0,imax+1
DO II=1,n_s
$\left(\frac{\partial^p \vec{U}}{\partial x^p}\right)^{\star\star}_{ijk} = \left(\frac{\partial^p \vec{U}}{\partial x^p}\right)^{\star\star}_{ijk} + \frac{\partial^p L_{II}(x_B)}{\partial x^p} \cdot \vec{U}_{i-IS+II-1,j,k}$
END DO
END DO
END DO
END DO

! Reconstruction in y-direction
DO k=-nGhost+1,kmax+nGhost
DO j=0,jmax+1
DO i=0,imax+1
DO II=1,n_s
$\left(\frac{\partial^{p+q} \vec{U}}{\partial x^p \partial y^q}\right)^{\star}_{ijk} = \left(\frac{\partial^{p+q} \vec{U}}{\partial x^p \partial y^q}\right)^{\star}_{ijk} + \frac{\partial^q L_{II}(y_B)}{\partial y^q} \cdot \left(\frac{\partial^p \vec{U}}{\partial x^p}\right)^{\star\star}_{i,j-IS+II-1,k}$
END DO
END DO
END DO
END DO

! Reconstruction in z-direction
DO k=0,kmax+1
DO j=0,jmax+1
DO i=0,imax+1
DO II=1,n_s
$\left(\frac{\partial^{p+q+r}\vec{U}(\vec{X}_B)}{\partial x^p \partial y^q \partial z^r}\right)_{ijk} = \left(\frac{\partial^{p+q+r}\vec{U}(\vec{X}_B)}{\partial x^p \partial y^q \partial z^r}\right)_{ijk} + \frac{\partial^r L_{II}(z_B)}{\partial z^r} \cdot \left(\frac{\partial^{p+q}\vec{U}}{\partial x^p \partial y^q}\right)^{\star}_{i,j,k-IS+II-1}$
END DO
END DO
END DO
END DO

END DO !p
END DO !q
END DO !r

In order to achieve a better computational performance, the loops and data arrays may be re-arranged and optimized.

High order WENO reconstruction In order to be able to capture strong discontinuities in the computational domain, a high order WENO reconstruction has been implemented. While its oscillation indicator and the nonlinear weighting of the WENO polynomials are based on classical 1D WENO methods (e.g., Shu [101]), the linear weights are taken from Dumbser et al. [19, 20]. Unlike other WENO reconstructions for cell interfaces, this WENO procedure delivers values for state and spatial derivatives at the *barycenter* of the cell, so only one reconstruction has to be performed per element. Known for being very robust and for good quality results, a characteristic decomposition is made. In addition to the characteristic variables, the conservative variables are also tested for oscillations. A reconstruction in primitive variables for higher orders has not been considered because a transformation from primitive to conservative spatial derivatives would be required then. This would add the effort of a CK-like procedure! The multidimensional reconstructions can again be implemented in an efficient dimension-by-dimension manner. Surprisingly, besides being far more computationally effective, this produces also the least oscillations in the diagonal direction of a structured grid. Tests with truly multi-dimensional oscillation indicators which consider the full volume stencil produced significantly larger oscillations. A possible explanation could be the greater number

of "weighting and smoothing" actions in the dimension-by-dimension case, as every single 1D stencil is tested instead of only one large multi-dimensional stencil.
The final reconstructed WENO polynomial w_{WENO} in 1D is a weighted sum of the polynomials $w_i(x)$ from the n_W different possible interpolation stencils:

$$w_{WENO}(x) = \sum_{i=1}^{n_W} \omega_i w_i(x). \tag{2.36}$$

Because *all* possible stencils are considered here, their number is equal to the order of the interpolation, $n_W = \mathcal{O}$, which is also the number of required stencil cells per single interpolation. Note that the stencil is *not* enlarged by one in the even order case as it is done for central reconstruction. The polynomials themselves are built from a monomial basis:

$$w_i(x) = \sum_{j=0}^{\mathcal{O}-1} c_j \Phi_j(x) = \sum_{j=0}^{\mathcal{O}-1} c_j x^j, \tag{2.37}$$

The normalized nonlinear weights ω_i are defined classically (Shu [101], Jiang et al. [51]) as

$$\omega_i = \frac{\tilde{\omega}_i}{\sum_{k=1}^{n_W} \tilde{\omega}_k}, \tag{2.38}$$

with

$$\tilde{\omega}_i = \frac{\lambda_i}{(\epsilon + \sigma_i)^r} \tag{2.39}$$

being dependent on the linear weights λ_i and the oscillation indicators σ_i.
For the parameters ϵ and r, different preferred values can be found in the literature, e.g., $\epsilon = 10^{-5} - 10^{-7}$, $r = 2 - 4$ (Liu et al. [66], Shu [101], Jiang et al. [51], Dumbser et al. [19]). By all authors, ϵ is regarded a safeguard against division by zero only and is chosen very small. However, the choice of ϵ in fact has an influence on the reconstruction: By definition (2.41), the oscillation indicator σ_i ranges from zero (no oscillations at all) to an unknown arbitrary large positive number (let's say 5000 for example, indicating severe oscillations). The denominator will be increased by the positive exponent r if $(\epsilon + \sigma_i) > 1.0$ and will be decreased if $(\epsilon + \sigma_i) < 1.0$. Hence, ϵ works as a threshold: The weight of stencils with small oscillations below a certain level or none at all is amplified, while all other weights are reduced. The ideal value for this threshold is problem dependent, though. For a vanishing ϵ, it would be $\sigma_i = 1.0$, a value that does not have any particular meaning regarding the oscillations. Hence,

it is suggested to chose ϵ in the vicinity of 1.0 for general problems because *all* occurring oscillations are amplified then and are put in relation in a problem independent manner. As a matter of fact, demanding calculations such as the double Mach reflection problem (DMR, see later in this section) proved to be more stable with such an ϵ. Slight variations can increase the solution quality for specific computations (e.g., $\epsilon = 10^{-2}$ for the DMR). Note that also the best error norms in the convergence tests (see later in this section and the appendix) could be achieved with this choice. It is emphasized, that both for the WENO STE-FV convergence studies and in tests with the Shu vortex (a similar setup can be found in 2.7.1.2), the very same or even better results are achieved in comparison with the central reconstruction STE-FV method.
The linear weights λ_i are defined according to Dumbser et al. [19] by

$$\lambda_i = \begin{cases} \lambda_C, & \text{if } i \text{ is the index of the central stencil,} \\ 1.0, & \text{else.} \end{cases} \tag{2.40}$$

λ_C puts a large weight on the central stencil. For the proposed STE-FV method, numerical tests indicated, that λ_C should be chosen between 10^2 (better for discontinuities) and 10^6 (for smooth solutions). While for lower order calculations ($\mathcal{O}3$ and $\mathcal{O}4$), values of $\lambda_C = 10^3 - 10^5$ and $r = 2 - 4$ have turned out to be robust while still resolving smooth solutions very well, values of $\lambda_C = 10^2$ and $r = 1$ seem to be advisable for high order calculations ($\geq \mathcal{O}5$) with strong shocks. In the benchmark examples at the end of this section, a great variety of different parameters (also for very high order) have been successfully tried, which underlines the robustness of the method.
Note that there is exactly one central stencil only for odd interpolation orders! This is actually vice versa to the traditional WENO reconstruction for interface quadrature points, where the even order interpolations produce the symmetric stencils. However, there are two central stencils now in the *even* order case for the WENO STE-FV method, a slightly left- and right-sided one (shifted by one cell relative to each other). These central stencils are both assigned λ_C.
The oscillation indicator σ_i is the classical 1D version by Shu [101]:

$$\sigma_i = \sum_{r=1}^{\mathcal{O}-1} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \Delta x^{2r-1} \left(\frac{\partial^r w_i(x)}{\partial x^r} \right)^2 dx. \tag{2.41}$$

It can be expressed by a quadratic functional

$$\sigma_i = \underline{\mathrm{C}}^T \underline{\underline{\Sigma}} \, \underline{\mathrm{C}}, \tag{2.42}$$

with the polynomial coefficients $\underline{\mathrm{C}}^T = (c_0, ..., c_{\mathcal{O}-1})$ of $w_i(x)$ and the oscillation matrix

$$\Sigma_{lm} = \sum_{r=1}^{\mathcal{O}-1} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \Delta x^{2r-1} \cdot \frac{\partial^r}{\partial x^r}\Phi_l(x) \cdot \frac{\partial^r}{\partial x^r}\Phi_m(x)\, dx. \tag{2.43}$$

$\underline{\mathrm{C}}$ and $\underline{\underline{\Sigma}}$ are grid dependent because they contain dimensioned quantities. In order to get rid of this grid dependency, equation (2.42) is rewritten. With the help of

$$M_{ij} = \frac{1}{\Delta x_i} \int\limits_{\Omega_{i_S}} x^j dx \tag{2.44}$$

and

$$\underline{\mathrm{C}} = \underline{\underline{M}}^{-1}\, \underline{\bar{\mathrm{W}}} \tag{2.45}$$

(see equations (2.108) and (2.109) in section 2.3.3), where $\underline{\bar{\mathrm{W}}}^T = (\bar{w}_1, ..., \bar{w}_{n_S})$ denotes the mean-values of the stencil cells for the interpolation, one obtains

$$\begin{aligned} \sigma_i &= \left(\underline{\underline{M}}^{-1}\, \underline{\bar{\mathrm{W}}}\right)^T \underline{\underline{\Sigma}}\, \underline{\underline{M}}^{-1}\, \underline{\bar{\mathrm{W}}}, & (2.46)\\ &= \underline{\bar{\mathrm{W}}} \left(\underline{\underline{M}}^{-1}\right)^T \underline{\underline{\Sigma}}\, \underline{\underline{M}}^{-1}\, \underline{\bar{\mathrm{W}}}, & (2.47)\\ &= \underline{\bar{\mathrm{W}}}\, \underline{\underline{Q}}\, \underline{\bar{\mathrm{W}}}. & (2.48) \end{aligned}$$

The new $n \times n$ (with $n = \mathcal{O}$) matrix $\underline{\underline{Q}} = \left(\underline{\underline{M}}^{-1}\right)^T \underline{\underline{\Sigma}}\, \underline{\underline{M}}^{-1}$, which is dimensionless and thus grid independent, can be calculated and stored for every desired order of accuracy.

The interpolation of point values at the barycenter for state and spatial derivatives for each tested WENO stencil is done exactly as for a central reconstruction, see equation (2.31). The only difference is, that the interpolation weights of each stencil cell have to be also determined for all shifted stencils, which can be done in the initialization. The dimension-by-dimension algorithm remains largely the same too, although some features need to be added. Note that the nonlinear WENO weights ω_i are based on the values of the *states* and do not need to be computed again for the derivatives, as they are in fact the derivatives of the reconstructed polynomial!

The conservative mean-values $\vec{\bar{U}}_i$ in each stencil cell are transformed into characteristic variables $\vec{\bar{W}}_i$ prior to the WENO reconstruction of a cell. Instead of considering each face's normal vector as in Dumbser et al. [20] for the characteristic decomposition, it is sufficient to treat only the x-, y- and z-direction for

the Cartesian elements as 2 faces are always parallel. Here, two possibilities arise:
One way is to consider *all* characteristic directions in each single reconstruction direction *plus* the conservative variables as an extra "direction". This is surely the most expensive way but promises a maximum of robustness. This will be referred to as *FCC* option ("full characteristic plus conservative"). Another way is to perform the characteristic decomposition only in the respective reconstructed directions. The conservative variables can be also included here as an additional set of polynomials (referred to as *CC* option) or can be just omitted (referred to as "characteristic only", *CO* option). Numerical tests showed for the *CO* case, that this still robust approach is the most efficient method regarding quality and effort. Note that this lean and preferred method had been derived after the others. Therefore, some of the examples later in this section were calculated with the more costly *FCC* option. In the following, the method is formulated for a general implementation, including the possibility of several characteristic decompositions.
Usually, if the reconstruction is performed for a cell interface (traditional WENO), an arithmetic average of the mean-values of two neighboring cells is used for the transformation. However, because the reconstruction is done for the barycenter of the cell, only the state $\vec{\bar{U}}_j$ inside the interpolated cell j is considered for the transformation, which is defined as

$$\vec{\bar{W}}_i^{(\vec{n}_{i_D})} = \left(\underline{\underline{R}}(\vec{\bar{U}}_j, \vec{n}_{i_D})\right)^{-1} \vec{\bar{U}}_i, \tag{2.49}$$

with $i_D = 0, ..., n_D$ and $\vec{n}_1^T = (1,0,0)$, $\vec{n}_2^T = (0,1,0)$, $\vec{n}_3^T = (0,0,1)$ in 3D. Like in Dumbser et al. [20], also the conservative variables themselves can be tested for oscillations, therefore a "fourth" direction $\vec{n}$ may be considered, with $\underline{\underline{R}}(\vec{\bar{U}}_j, \vec{n}_0)$ being the identity matrix.
After each 1D WENO reconstruction, the interpolated point value at the barycenter $\vec{X}_B$ of the cell j must be transformed back into conservative variables

$$\vec{U}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO} = \underline{\underline{R}}(\vec{\bar{U}}_j, \vec{n}_{i_D}) \vec{W}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO}, \tag{2.50}$$

which must be based on the original transformation. It is emphasized, that this is a critical step, as each 1D stencil of the reconstruction in the next dimension requires again stencil cells which have been transformed on the basis of the *same* mean-value. Hence, a back-transformation into conservative variables prior to the next reconstruction direction is essential.

Equation (2.50) contains only the interpolated WENO states, but as the STE-FV method requires also the spatial derivatives, the interpolation procedure is done for them too, and one gets in fact

$$\frac{\partial^p}{\partial x^p}\vec{U}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO} = \underline{\underline{R}}(\vec{\bar{U}}_j, \vec{n}_{i_D})\frac{\partial^p}{\partial x^p}\vec{W}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO}. \tag{2.51}$$

If only one characteristic decomposition per direction was made (*CO* option), the final result for the reconstruction has been obtained with equation (2.51). Otherwise, if several directions (*FCC*- and *CC* option) per reconstruction have been considered for the characteristic decomposition, a decision for one of the $n_D + 1$ reconstruction polynomials must be made, before going on in the next direction. This can be done for example on an ENO basis. For this purpose, the oscillation matrix is exploited again in order to determine the oscillation indicator for each conservative polynomial:

$$\sigma^{(\vec{n}_{i_D})} = \underline{\mathrm{C}}^{T(\vec{n}_{i_D})}\underline{\underline{\Sigma}}\,\underline{\mathrm{C}}^{(\vec{n}_{i_D})}. \tag{2.52}$$

The coefficients $\underline{\mathrm{C}}$ are built from the known state and spatial derivatives at the barycenter. For a conservative variable u_i, the *pth* coefficient of its polynomial

$$u_i(x) = \sum_{j=0}^{\mathcal{O}-1} c_j x^j = c_0 + c_1 x + c_2 x^2 + \ ... \ + c_{\mathcal{O}-1} x^{\mathcal{O}-1} \tag{2.53}$$

can be determined by simply deriving the polynomial p times, for example for $p = 3$:

$$\frac{\partial^3}{\partial x^3}u_i(x) = 3{\cdot}2{\cdot}1{\cdot}c_3 + 4{\cdot}3{\cdot}2{\cdot}c_4 x + \ ... \ + (\mathcal{O}-1){\cdot}(\mathcal{O}-2){\cdot}(\mathcal{O}-3){\cdot}c_{\mathcal{O}-1}x^{\mathcal{O}-4}, \tag{2.54}$$

$$\begin{aligned} c_3 &= \frac{1}{6}\cdot\frac{\partial^3}{\partial x^3}u_i(x) - 4\cdot c_4 x - \ ... \ - \frac{(\mathcal{O}-1)\cdot(\mathcal{O}-2)\cdot(\mathcal{O}-3)}{6}\cdot c_{\mathcal{O}-1}x^{\mathcal{O}-4}, \\ &= \frac{1}{6}\cdot\frac{\partial^3}{\partial x^3}u_i(x_B = 0.0). \end{aligned} \tag{2.55}$$

All space derivatives are known at the barycenter $\vec{X}_B$ and the interpolation weights for each element were determined in a relative coordinate system with $\vec{X}_B = (0,0,0)^T$. Hence, every arbitrary coefficient can be obtained:

$$c_j = \frac{1}{j!}\cdot\frac{\partial^j}{\partial x^j}u_i(x_B). \tag{2.56}$$

In order to avoid problems with grid dependency, $\underline{\underline{\Sigma}}$ is precalculated on a reference element of size $\Delta x = 1.0$ and the coefficients $\underline{C}^T = (\tilde{c}_0, ..., \tilde{c}_{\mathcal{O}-1})$ are normalized by the cell size:

$$\tilde{c}_j = (\Delta x)^j \cdot c_j. \tag{2.57}$$

Having determined the indicators $\sigma^{(\vec{n}_{i_D})}$ from equation (2.52), the least oscillating polynomial is chosen for each component u_i of the conservative state vector $\vec{U}_j = (u_1, ..., u_i, ..., u_{n_{Var}})$ in the interpolated cell j:

$$\frac{\partial^p}{\partial x^p} u_i(\vec{X}_B)_{WENO} = \frac{\partial^p}{\partial x^p} u_i^{(minloc(\sigma^{(\vec{n}_{i_D})}))}(\vec{X}_B)_{WENO}. \tag{2.58}$$

The 1D algorithm is repeated with the WENO values from the previous step as input for the reconstruction in the next dimension, with one exception: After the back-transformation into conservative variables, one ends up with a two- and later with a three-dimensional polynomial, or rather their derivatives $\frac{\partial^{p+q}}{\partial x^p \partial y^q} \vec{U}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO}$ and $\frac{\partial^{p+q+r}}{\partial x^p \partial y^q \partial z^r} \vec{U}_j^{(\vec{n}_{i_D})}(\vec{X}_B)_{WENO}$. In order to make an ENO decision as in equation (2.52) for one of the polynomials, truly multidimensional oscillation matrices are used:

$$\begin{aligned} \Sigma_{lm,2D} \quad &= \quad \sum_{\alpha=0}^{\mathcal{O}-1} \sum_{\beta=0}^{\mathcal{O}-1} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} (\Delta x^{2\alpha-1} \cdot \Delta y^{2\beta-1} \\ &\cdot \quad \frac{\partial^{\alpha+\beta} \Phi_l(x,y)}{\partial x^\alpha \partial y^\beta} \cdot \frac{\partial^{\alpha+\beta} \Phi_l(x,y)}{\partial x^\alpha \partial y^\beta}) \, dxdy, \\ &with \quad 0 < \alpha + \beta, \end{aligned} \tag{2.59}$$

$$\begin{aligned} \Sigma_{lm,3D} \quad &= \quad \sum_{\alpha=0}^{\mathcal{O}-1} \sum_{\beta=0}^{\mathcal{O}-1} \sum_{\gamma=0}^{\mathcal{O}-1} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \int_{y_{j-\frac{1}{2}}}^{y_{j+\frac{1}{2}}} \int_{z_{k-\frac{1}{2}}}^{z_{k+\frac{1}{2}}} (\Delta x^{2\alpha-1} \cdot \Delta y^{2\beta-1} \cdot \Delta z^{2\gamma-1} \\ &\cdot \quad \frac{\partial^{\alpha+\beta+\gamma} \Phi_l(x,y,z)}{\partial x^\alpha \partial y^\beta \partial z^\gamma} \cdot \frac{\partial^{\alpha+\beta+\gamma} \Phi_m(x,y,z)}{\partial x^\alpha \partial y^\beta \partial z^\gamma}) \, dxdydz, \\ &with \quad 0 < \alpha + \beta + \gamma, \end{aligned} \tag{2.60}$$

where Φ denotes the full tensor product monomial basis. Note that always at least one derivative is considered. For $\alpha + \beta + \gamma = 0$, no contribution is made to the matrices.
The following algorithm summarizes the most important steps of the WENO reconstruction.

Algorithm 2.1.2. ***WENO dimension-by-dimension reconstruction***

Reconstruction in x-direction
For all elements (including the ghost cells in y- and z-direction):

1. *Get the right Eigenvector matrix* $\underline{\underline{R}}(\vec{\bar{U}}_j, \vec{n}_{i_D})$ *and its inverse for all desired normal vector directions* $\vec{n}_{i_D}$ *for the characteristic decomposition.*

2. *For all desired normal vector directions:*
   For all x-derivatives:

   a) *For all possible interpolation stencils:*

      i. *Transformation into characteristic variables of all stencil cells, based on the state in the interpolated cell.*
      ii. *Interpolate onto the barycenter* $\vec{X}_B$.
      iii. *Calculate the oscillation indicator* σ_i *for all characteristic variables.*

   b) *For all components of the characteristic state vector:*
      For all possible interpolation stencils:

      i. *Calculate the nonlinear weight* ω_i *and weight the interpolated value of the considered stencil.*
      ii. *Back-transformation into conservative variables.*

3. *If more than one characteristic decomposition (otherwise finished):*
   For all components of the conservative state vector:
   For all normal vector directions:
   Determine the oscillation indicator $\sigma^{(\vec{n}_{i_D})}$ *and choose the least oscillating polynomial.*

$\Rightarrow \left(\frac{\partial^p \vec{U}(\vec{X}_B)}{\partial x^p}\right)^{\star\star}_{ijk,WENO}$

Reconstruction in y-direction
For all elements (including the ghost cells in z-direction):
Repeat the algorithm from the x-Reconstruction with the following modifications:

1. *Take the WENO-reconstructed values from the previous reconstruction as input.*

2. *Consider now also the y-derivatives.*

3. *If more than one characteristic decomposition: Use the 2D oscillation matrix in order to determine the least oscillating 2D polynomial.*

$\Rightarrow \left(\frac{\partial^{p+q}\vec{U}(\vec{X}_B)}{\partial x^p \partial y^q}\right)^{\star}_{ijk,WENO}$

Reconstruction in z-direction
For all domain elements:
Repeat the algorithm from the x-Reconstruction with the following modifications:

1. *Take the WENO-reconstructed values from the previous reconstruction as input.*
2. *Consider now also the z-derivatives.*
3. *If more than one characteristic decomposition: Use the 3D oscillation matrix in order to determine the least oscillating 3D polynomial.*

$\Rightarrow \left(\frac{\partial^{p+q+r}\vec{U}(\vec{X}_B)}{\partial x^p \partial y^q \partial z^r}\right)_{ijk,WENO}$

Boundary conditions The treatment of the boundary conditions for the STE-FV method is equivalent to the implementation for the structured ADER-FV method. So-called ghost cells are set prior to every iteration. The most simple case, periodic boundary conditions, is realized by copying the values from the domain cells opposite to the boundary. This of course requires a domain with sufficient domain cells in every direction, at least $n_{Elem,iDir} = n_{Ghost}$. In order to facilitate the boundary treatment and the loops in the algorithms, also the first row of ghost cells is reconstructed in the STE-FV method, which adds one ghost cell row per direction in comparison with the ADER-FV method with central reconstruction. Hence, $n_{Ghost_{STE-FV,Central}} = int(\frac{\mathcal{O}+2}{2.})$. For the WENO reconstruction, enough ghost cells need to be provided for the outermost stencil, therefore $n_{Ghost_{STE-FV,WENO}} = \mathcal{O}$. Table 2.8 in section 2.3.1 lists the number of ghost cell rows per direction for several orders of accuracy. Note that the STE-FV method is never forced to rely on one-sided stencils at the boundary. Even for wall boundary conditions (slip-walls and adiabatic no-slip-walls), the ghost cells can be set by mirroring the domain values and giving the normal velocities the respective other sign. Both for the central and the WENO reconstruction, the Taylor expansion will yield symmetrical values at the domain boundary and the Riemann solvers will automatically set the normal velocity to zero. Last but not least, inflow, outflow and coupling boundary conditions are realized by prescribing mean-values in the ghost elements.

Numerical convergence studies The experimental order of convergence is tested for the 2D and 3D version of the STE-FV method. In order to examine the implementation for the full Navier-Stokes equations (see section 2.2.3), the following test case from Gassner et al. [38] is considered. The Navier-Stokes equations with the source term

$$\vec{S} = \alpha \begin{pmatrix} \cos(\beta)\,(d\,k - \omega) \\ \cos(\beta)\,A + \sin(2\beta)\alpha\,k\,(\gamma - 1) \\ \cos(\beta)\,A + \sin(2\beta)\alpha\,k\,(\gamma - 1) \\ \cos(\beta)\,A + \sin(2\beta)\alpha\,k\,(\gamma - 1) \\ \cos(\beta)\,B + \sin(2\beta)\alpha\,(d\,k\gamma - \omega) + \sin(\beta)\left(\frac{d\,k^2\mu\gamma}{Pr}\right) \end{pmatrix} \tag{2.61}$$

on the right hand side are solved, where $\beta = k(x_1 + x_2 + x_3) - \omega t$, $A = -\omega + \frac{k}{d-1}\left((-1)^{d-1} + \gamma\,(2\,d - 1)\right)$ and $B = \frac{1}{2}\left((d^2 + \gamma(6 + 3\,d))\,k - 8\,\omega\right)$.
The problem has an analytical solution, which is given by

$$\vec{U} = \begin{pmatrix} \sin(\beta)\alpha + 2 \\ \sin(\beta)\alpha + 2 \\ \sin(\beta)\alpha + 2 \\ \sin(\beta)\alpha + 2 \\ (\sin(\beta)\alpha + 2)^2 \end{pmatrix}. \tag{2.62}$$

For the numerical experiments in the dimensions $d = 2$ and $d = 3$, the parameters are chosen as $\gamma = 1.4$, $Pr = 0.72$, $\mu = 0.0001$, $R = 287.14$ and $\alpha = 0.1$, $\omega = 10$ and $k = \pi$. The exact Godunov Riemann solver (Toro [112]) is used for the numerical fluxes of the hyperbolic part, while the dGRP Riemann solver (Gassner et al. [35], [37], Lörcher et al. [68]) is chosen for the viscous fluxes. For all calculations, the CFL number is set to $CFL = 0.45$. The extents of the computational domain are $[0, 2] \times [0, 2]$ and the number of elements per direction are increased in several stages. Periodic boundary conditions are employed and the error norms are calculated with the help of the exact solution when the simulation ends at $t_{end} = 1$.
Tables A.1-A.6 show the number of grid elements per direction, the error norms for the total energy ρE and the convergence rates for the 2D and 3D STE-FV method. The respective orders of convergence are achieved for both the central reconstruction and the WENO reconstruction (CO option with parameters $\lambda_C = 10^4$, $\epsilon = 1.0$, $r = 3$).

Performance The 3D implementation of the STE-FV method is examined for its computational performance. The setup is simple: After an empty domain has been initialized with a constant flow, the Navier-Stokes equations are solved and 100 time steps are performed. The CPU time is measured on two different platforms, on a NEC-SX8 vector processor and on a scalar Intel Xeon 5150 $2.66GHz$ core. Both the central and the WENO reconstruction (CO option) are employed. Tables 2.5 and 2.6 show the remarkable performance on the vector computer. The method is most efficient for the orders $\mathcal{O}4$-$\mathcal{O}6$, where up to 46% of the machine's peak performance (16 GFlop/s) are achieved. Note that such a performance is considered excellent even for SX8 custom-built finite difference codes! The STE-FV scheme is about one magnitude slower on the scalar Xeon processor, as its vectorization properties cannot be exploited that well anymore. Hence, all loops are executed in a "normal", scalar way.
A comparison of the CPU time per element and iteration with a calculation solving the Euler equations instead of the Navier-Stokes equations shows, that the additional effort for the treatment of the viscous terms is quite low (about 5% for $\mathcal{O}4$, Table 2.5). Furthermore, depending on which processor and order is used, employing WENO reconstruction instead of central reconstruction is about 8-40% more expensive with regard to the total CPU time (Table 2.6).
The jump in the costs from even order to the next odd order is considerably bigger (about a factor of four) than from odd order to the next even order (about a factor of two). In the first case, the number of required integration points increases, while it remains the same in the latter case. The other extra costs result from the respective higher effort for reconstruction, space-time Taylor expansion and CK procedure.
Finally, the time for an element update is compared to the unstructured Rec-FV method on the Xeon machine. After being at first considerably faster for lower orders, it seems that the advantage of the STE-FV scheme decreases with higher orders. This can be largely attributed to the Rec-FV's quadrature-free formulation, resulting in less integration effort. However, it is stressed, that the STE-FV method calculates the fluxes over the six sides of a hexahedron, Rec-FV on the other hand over the four sides of a tetrahedron. Furthermore, hexahedral elements usually allow much larger time steps and provide better accuracy. Last but not least, the STE-FV code is clearly superior on the vector platform, where the Rec-FV method shows only a performance of about 100 MFlop/s.

$\mathcal{O}$	STE-FV, SX8	STE-FV, Xeon	STE-FV, EE, Xeon
2	1.766E-06 (1269 MFlop/s)	8.257E-06	7.702E-06
3	7.416E-06 (3865 MFlop/s)	7.035E-05	6.149E-05
4	1.359E-05 (5065 MFlop/s)	1.511E-04	1.434E-04
5	6.646E-05 (7333 MFlop/s)	6.032E-04	6.114E-04
6	1.081E-04 (7407 MFlop/s)	1.209E-03	1.177E-03
7	7.781E-04 (4234 MFlop/s)	4.363E-03	4.079E-03
8	1.279E-03 (4274 MFlop/s)	8.084E-03	7.945E-03

Table 2.5: Efficiency factors $\frac{t_{CPU}[s]}{Elem \cdot Iter}$, STE-FV, 3D, central reconstruction.

$\mathcal{O}$	STE-FV, SX8	STE-FV, Xeon	Rec-FV, Xeon
2	2.456E-06 (2041 MFlop/s)	1.157E-05	4.057E-05
3	9.854E-06 (4089 MFlop/s)	7.648E-05	1.214E-04
4	1.987E-05 (5161 MFlop/s)	1.731E-04	3.209E-04
5	7.051E-05 (7036 MFlop/s)	6.517E-04	7.089E-04
6	1.358E-04 (7161 MFlop/s)	1.316E-03	1.410E-03
7	7.973E-04 (4595 MFlop/s)	4.081E-03	2.636E-03
8	1.398E-03 (4378 MFlop/s)	9.303E-03	-

Table 2.6: Efficiency factors $\frac{t_{CPU}[s]}{Elem \cdot Iter}$, STE-FV, 3D, WENO reconstruction.

Double Mach reflection The famous and challenging double Mach reflection (DMR) problem by Woodward and Colella [125] is chosen as a test case for the 2D implementation of the WENO STE-FV scheme. It is realized in its original setup for the Euler equations on Cartesian grids and consists of a Mach 10 shock in air ($\gamma = 1.4$), which impinges on a reflecting wall in a 60° angle. This is equivalent to a horizontally moving shock hitting a 30° ramp. The rectangular domain has the extents $[0, 4] \times [0, 1]$, while only the area $[0, 3] \times [0, 1]$ is depicted in the overall views of Figs. 2.2 and 2.3. At $t = 0$, the shock is initialized such that it touches the slip-wall at $x = \frac{1}{6}$. The primitive state in front of the shock is $\vec{U}_1 = (1.4, 0.0, 0.0, 1.0)^T$ while the Rankine-Hugoniot conditions give $\vec{U}_2 = (8.0, 8.25, 0.0, 116.5)^T$ for the post-shock region. For the boundary conditions, the ghost cells at the left and at the upper boundary are set by prescribing the exact solution to the moving shock (which is simply

$\vec{U}_2$ for the left boundary). On the lower boundary, also the ghost cells from $x = 0.0$ to $x = \frac{1}{6}$ are set using the moving shock solution (respectively $\vec{U}_2$, as the shock is initialized at this position), which forces the shock to be attached to the wall. Otherwise, without the prescribed post-shock values at the lower boundary, the reflection would travel upstream. Finally, the ghost cells at the right boundary are set to the undisturbed state $\vec{U}_1$. The HLLE flux is used and the CFL number is set to 0.5 for all calculations, which are run until $t_{end} = 0.2$. Two different meshes are considered, a coarse one with cell size $\Delta h = \Delta x = \Delta y = \frac{1}{120}$ and a fine one with $\Delta h = \frac{1}{480}$. All calculations were performed on one Intel Xeon 5150 $2.66GHz$ core.
The results in Figs. 2.2- 2.4 depict the density (plotted for 30 equidistant contour levels from 1.5 to 21.5) and are in good accordance with other publications (e.g., Dumbser et al. [20], Woodward and Colella [125]). However, other than in the previous calculations, methods up to $\mathcal{O}6$ are used! The WENO STE-FV method (CO option) remains stable even for very high orders. Note that the WENO parameters for the orders $\mathcal{O}3$ and $\mathcal{O}4$ are chosen as $\lambda_C = 10^4$ and $r = 3$, while for the orders $\mathcal{O}5$ and $\mathcal{O}6$ they are adapted to $\lambda_C = 10^2$ and $r = 1$. For all coarse ($\Delta h = \frac{1}{120}$) calculations and the fine ($\Delta h = \frac{1}{480}$) $\mathcal{O}3$ calculation, $\epsilon = 10^{-2}$ was used, while for the fine higher order computations ($\mathcal{O}4 - \mathcal{O}6$), $\epsilon = 1.0$ turned out to deliver better and more stable results.
Both on the coarse and the fine grid (Figs. 2.2 and 2.3), the effect of the increasing order becomes visible especially in the "roll-up" region in front of the contact discontinuity. This (as a matter of fact unphysical) feature with its small-scale structures is considered "a qualitative indicator of the amount of numerical viscosity introduced by the scheme" (Dumbser et al. [20]). The method's capability to capture these structures grows with rising order of accuracy on the coarse grid until $\mathcal{O}5$. For $\mathcal{O}6$, no significant differences are visible: The method has reached its maximum capabilities for this grid size. On the fine grid however, even the contact discontinuity itself begins to roll-up. The detail views for the calculations (Figs. 2.3) show, that the amount of rolling still grows strongly, when the order is increased. Also the shape of the main vortex begins to change and another feature, which could be an additional vortex, begins to emerge for $\mathcal{O}5$ and $\mathcal{O}6$. It is clear that oscillations become more and more difficult to handle with rising order and grid resolution, as the numerical viscosity decreases and the number of potentially oscillating stencils increases. Nevertheless, the calculations remain stable. Note the difference in the shock width for the coarse and the fine grid, which is in both cases and for all orders about three to four cell sizes!

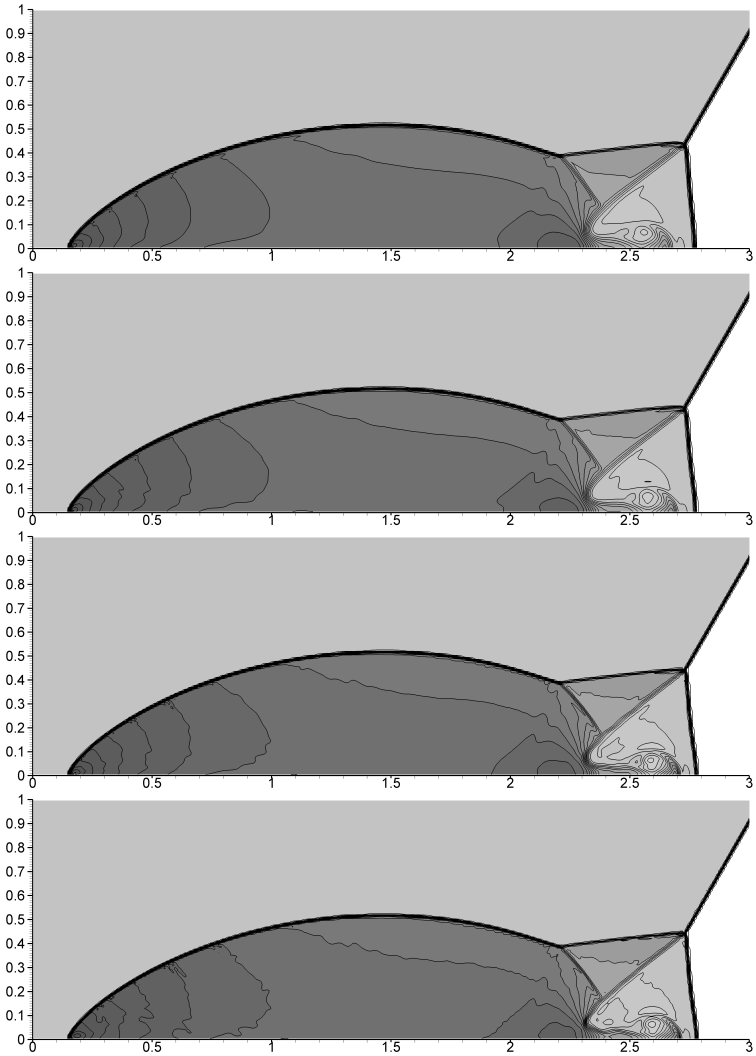


Figure 2.2: DMR, $\Delta h = \frac{1}{120}$. From top to bottom: $\mathcal{O}3$, $\mathcal{O}4$, $\mathcal{O}5$, $\mathcal{O}6$.

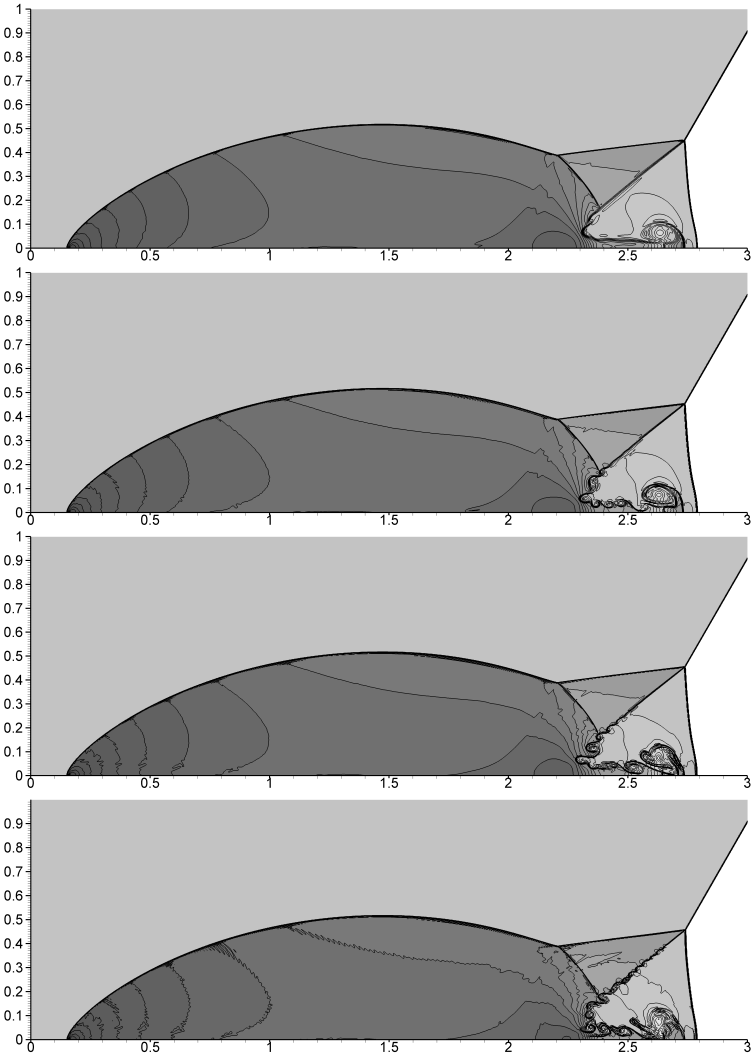


Figure 2.3: DMR, $\Delta h = \frac{1}{480}$. From top to bottom: $\mathcal{O}3$, $\mathcal{O}4$, $\mathcal{O}5$, $\mathcal{O}6$.

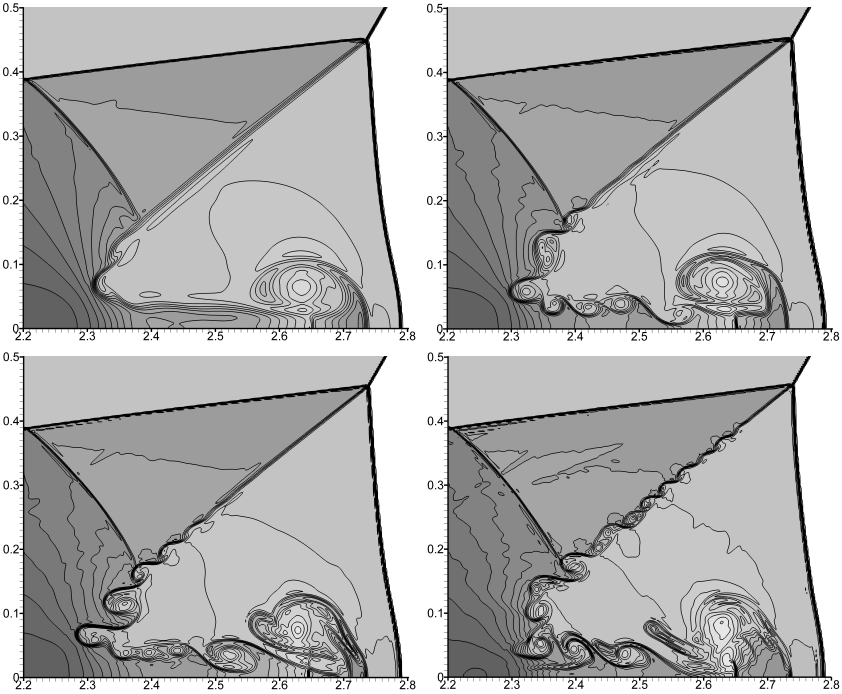


Figure 2.4: DMR, $\Delta h = \frac{1}{480}$. Zooms into the roll-up region: Top left: $\mathcal{O}3$, top right: $\mathcal{O}4$, bottom left: $\mathcal{O}5$, bottom right: $\mathcal{O}6$.

3D explosion problem The 3D WENO algorithm for the STE-FV method is tested on this example for the Euler equations. The problem is based on the 1D Sod test case (Toro [112]), for which a reference solution can be computed by solving a one-dimensional PDE with source term. This was done on a very fine mesh (10000 elements) with a second order TVD method and the data have been kindly provided by Dumbser et al. [20]. In the cubicle calculation domain with the extents $[-1.0,+1.0]\times[-1.0,+1.0]$, the cells inside a radius of $r=0.4$ are initialized with the primitive values $\vec{U}_{in}=(1.0,0.0,0.0,0.0,1.0)^T$ and the cells outside the radius with $\vec{U}_{out}=(0.125,0.0,0.0,0.0,0.1)^T$. The initial conditions are projected by setting the Gauss integration points i_{GP} of respective order, with a subsequent integration to the mean-value. In order to avoid unphysical initial states for cells that overlap the given radius r, which especially occurs in the diagonal directions, the transition from $\vec{U}_{in}$ to $\vec{U}_{out}$ is slightly smoothed with the help of the tanh function:

$$\begin{aligned}\vec{U}(\vec{X}_{i_{GP}}) &= 0.5\cdot(\vec{U}_{in}-\vec{U}_{out})\\ &- 0.5\cdot(\vec{U}_{in}-\vec{U}_{out})\cdot\tanh(S_F\cdot(\sqrt{x_{i_{GP}}^2+y_{i_{GP}}^2+z_{i_{GP}}^2}-r))\\ &+ \vec{U}_{out},\end{aligned} \tag{2.63}$$

where S_F is a smoothing factor. It is set to $S_F=100$ for this example. Note that the greater the smoothing factor, the sharper the edge of the discontinuity turns out. In the boundary ghost cells, $\vec{U}_{out}$ is prescribed throughout the entire calculation. The number of cells inside the domain is 100^3, which is equivalent to a cell size of $\Delta x=\Delta y=\Delta z=0.02$. A CFL number of $CFL=0.5$ was used for the calculation, which stops at $t_{end}=0.25$. As Riemann solver, the HLLE flux is employed and the WENO parameters (FCC option) are set to $\lambda_C=10^4$, $\epsilon=10^{-5}$, $r=4$.
The results along the positive x-axis and the xyz-diagonal are shown in Fig. 2.5 for the orders $O2$, $O3$ and $O4$. The data have been extracted from the mean-values (1D data extraction with 1000 points in *Tecplot*). Especially the close-ups of the contact discontinuity and the shock illustrate, that the flow features are well resolved and that the solution is essentially oscillation-free in both directions. Furthermore, the higher order calculations are clearly superior to the $\mathcal{O}2$ solution at the rarefaction fan and the contact discontinuity. The calculations were performed on a single Intel Xeon 5150 $2.66GHz$ core.

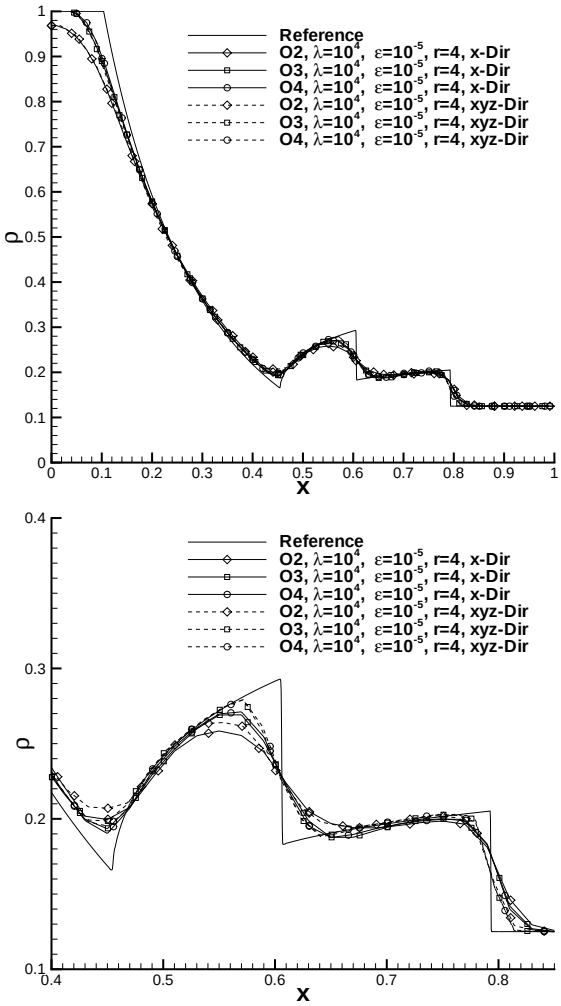


Figure 2.5: 3D explosion problem (top) with zoom into the region of the contact discontinuity and the shock (bottom).

Shock-density interaction This actually one-dimensional test case by Shu and Osher [102] is calculated with the 2D and 3D version of the STE-FV method. It was designed to underline the advantages of high order schemes. The Euler equations are solved in a domain of size $[-5.0, +5.0] \times [-0.075, +0.075]^2$, with $400 \times 6 \times 6$ elements in the respective directions. Thus, the element size is $\Delta x = \Delta y = \Delta z = 0.025$. For the $\mathcal{O}7$ and $\mathcal{O}8$ calculations, eight elements in y- and z-direction are used because periodic boundary conditions are employed at the long walls of the tube. At position $x = -4.0$, a discontinuity is initialized with the primitive left $\vec{U}_l = (3.8570, 2.6294, 0.0, 0.0, 10.333)^T$ and right $\vec{U}_r = (1 + 0.2 \cdot sin(5x), 0.0, 0.0, 0.0, 1.0)^T$ states. A second order TVD method is used in order to compute the reference solution in 1D, using a very fine spacing with 10000 elements. The 3D STE-FV method (*FCC* option) is chosen for the orders $\mathcal{O}2$-$\mathcal{O}5$. After that, the 2D version is used for the orders $\mathcal{O}6$-$\mathcal{O}8$ in order to save time and memory. Note that 2D and 3D results for the same order turned out to be identical. For both the 2D and 3D simulations, the HLLE flux is employed, the time step is determined with $CFL = 0.5$ and the end time of the calculation is $t_{end} = 1.8$. A single Intel Xeon 5150 $2.66GHz$ core was used for the computation. The results along with the WENO parameters for the different calculations are given in Figs. 2.6 and 2.7. Only information from the mean-values is plotted.
When the simulation starts, the initial condition produces a shock that moves into a sinusoidal fluctuation, which causes high frequency entropy waves. These waves are not at all resolved for $\mathcal{O}2$, while the solution approaches the reference with rising order and is almost perfect for $\mathcal{O}5$ (see the close-up in Fig. 2.6). The calculations $\mathcal{O}6$-$\mathcal{O}8$ begin to show minor oscillations at the bottom of the first two sawtooth patterns at the left and at the shock. However, the calculations are in very good agreement with the reference solution and remain stable (close-up in Fig. 2.7). It is emphasized, that the latter is remarkable for orders up to $\mathcal{O}8$.

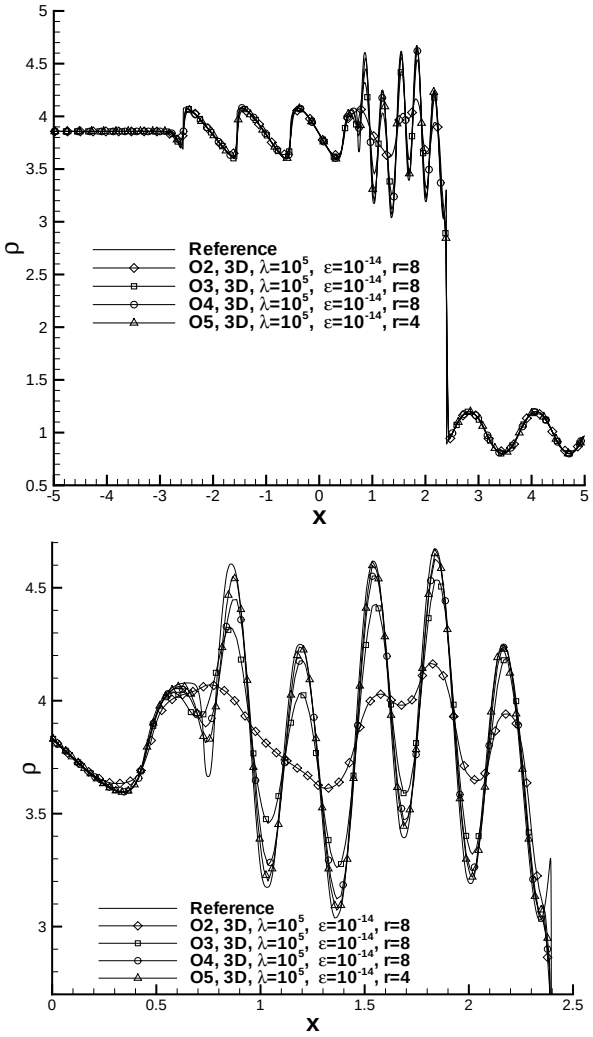


Figure 2.6: Shock density interaction (top) with close-up (bottom) for orders $\mathcal{O}2$-$\mathcal{O}5$.

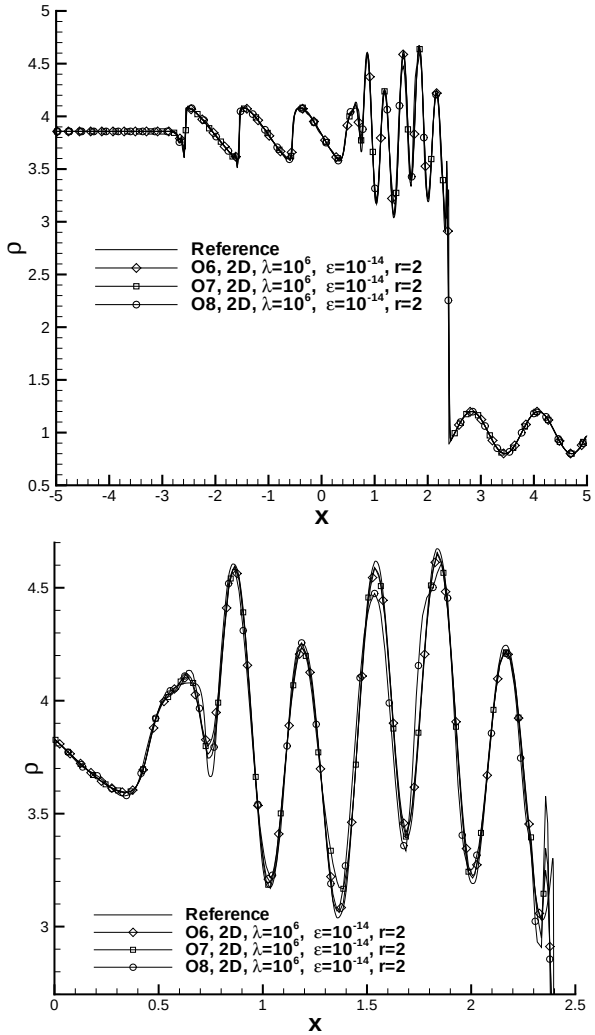


Figure 2.7: Shock density interaction (top) with close-up (bottom) for orders $\mathcal{O}6$-$\mathcal{O}8$.

Shock tube problems Last but not least, a variety of classical 1D shock tube problems (STP) are solved with the 3D STE-FV method. The governing equations are the Euler equations, the computational domain consists of a tube with rectangular cross-section and the dimensions $[-0.5,+0.5]\times[-0.1,+0.1]^2$. The spacing is $\Delta x=\Delta y=\Delta z=0.01$, which equals $100\times 20\times 20$ grid cells. Table 2.7 lists the initial conditions for the four considered problems. The position of the Riemann interface lies always at $x=0.0$. To each of the problems, the reference solution can be determined by an exact Riemann solver (e.g., the Godunov RP solver, Toro [112]). For all computations (single CPU core, Intel Xeon 5150 $2.66GHz$) the HLLE flux and $CFL=0.5$ are used. The WENO parameters (*FCC* option) are given in Figs. 2.8-2.11. Note that only the mean-values are taken for the 1D data extraction with 1000 points in *Tecplot*. It is emphasized, that *all* of the shock tube problems are calculated for the orders $\mathcal{O}2$ to $\mathcal{O}5$. This is quite exceptional, especially for STP 3 and 4, which are considered difficult and are usually not solved with an order higher than $\mathcal{O}3$ (Dumbser et al. [20]).
STP 1 is a variation of the standard Sod test case (Toro [112]) and contains a sonic point which leads usually to the sonic glitch problem (Dumbser et al. [20]). However, the results for any order of accuracy (Fig. 2.8) hold nothing like that for the STE-FV method. The discontinuities are resolved sharply and the close-up shows the superiority of the higher orders, in particular at the contact discontinuity. The rarefaction fan (not depicted) is captured well by all method.
STP 2 by Lax [60] is often used in the literature for testing high order WENO schemes. Good results with the tendency to better performance for higher order are obtained and depicted in Fig. 2.9. Note that the plateau on top of the pressure jump is represented more and more precisely for higher orders. The same is true for the edges of the jump.
STP 3 (Toro [112]) involves two close discontinuities and very high pressure jumps (five orders of magnitude, see initial conditions in Table 2.7). Nevertheless, all calculations remain stable. A slight overshoot is visible for $\mathcal{O}5$ in Fig. 2.9. Again, the flanks of the pressure jump are resolved better for increasing order.
STP 4 (Toro [112]) contains a very slowly moving shock, which can be problematic for numerical methods, leading to spurious oscillations. However, both the slowly moving and the strong right-moving shock wave are represented well by the STE-FV scheme. For $\mathcal{O}4$, small oscillations appear on top of the discontinuities, getting bigger for $\mathcal{O}5$, especially at the strong shock. Yet the calculations for all orders remain stable!

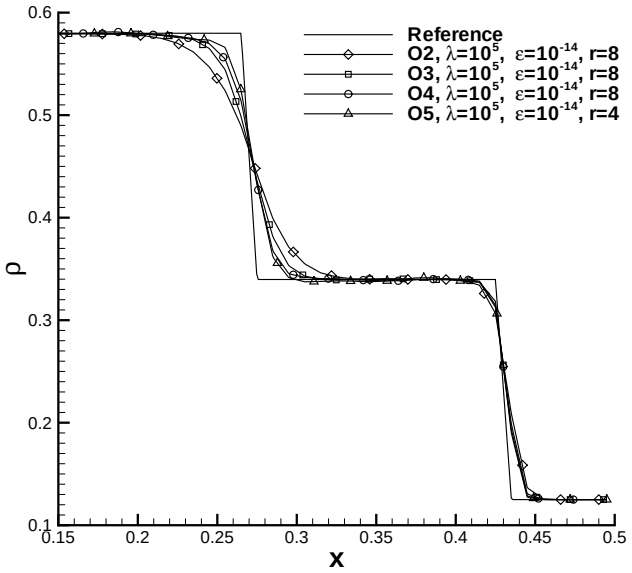


Figure 2.8: STP 1: Close-up of the contact discontinuity and the shock.

STP	ρ_l	u_l	p_l	ρ_r	u_r	p_r	t_{end}
1	1.0	0.75	1.0	0.125	0.0	0.1	0.20
2	0.445	0.698	3.528	0.5	0.0	0.571	0.14
3	1.0	0.0	1000.0	1.0	0.0	0.01	0.012
4	5.99924	19.5975	460.895	5.99242	-6.19633	46.095	0.035

Table 2.7: Initial conditions for the shock tube problems.

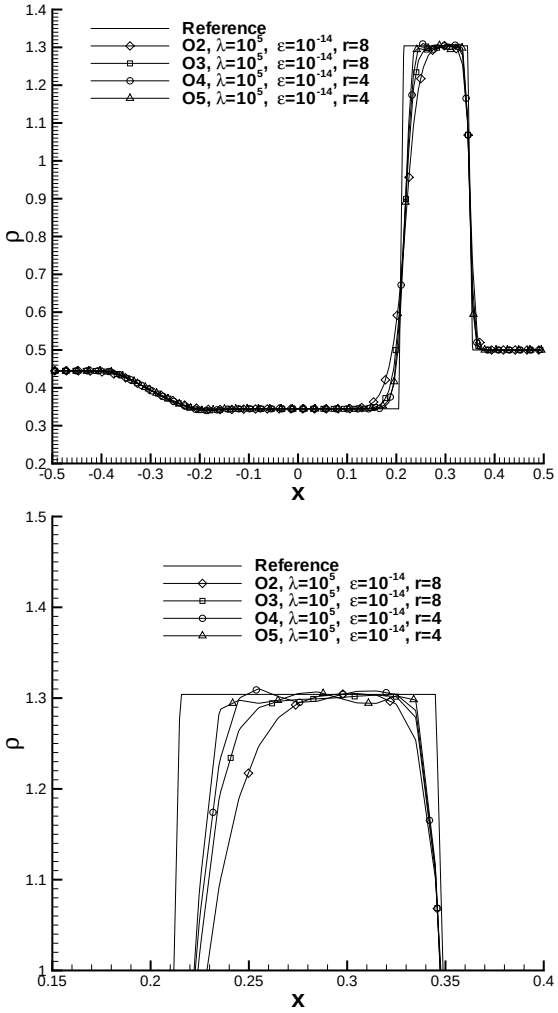


Figure 2.9: STP 2 (top) with close-up (bottom).

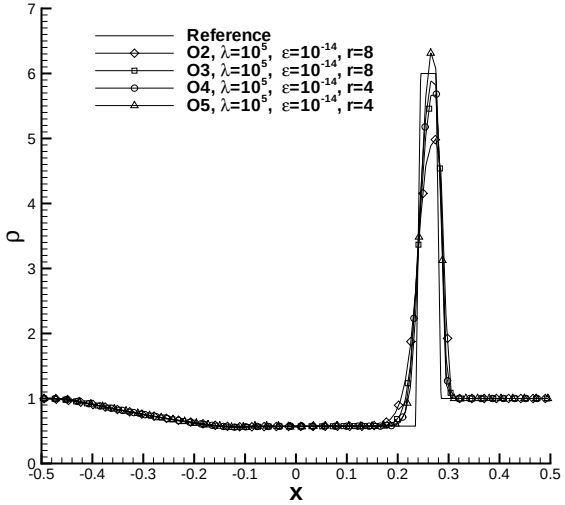


Figure 2.10: STP 3.

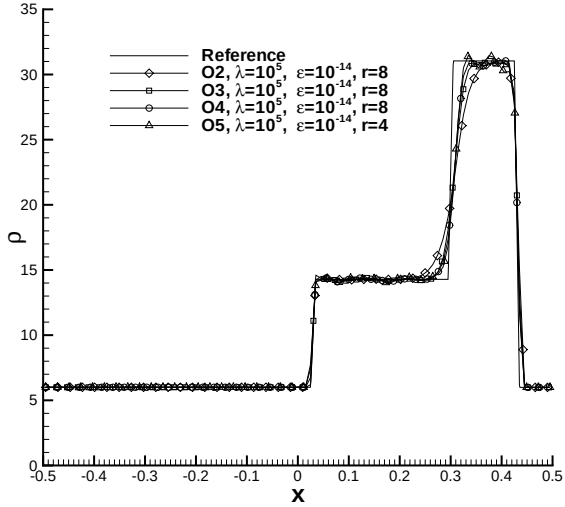


Figure 2.11: STP 4.

2.2 Equations

In the following, the equations which are solved in the coupling framework are described and the issue of data conversion between the conservative and the primitive variables is addressed.

2.2.1 Linearized Euler Equations

The linearized Euler equations (LEE) are used in their primitive form:

$$\vec{U}'_t + \underline{\underline{A}}\vec{U}'_x + \underline{\underline{B}}\vec{U}'_y + \underline{\underline{C}}\vec{U}'_z = 0. \tag{2.64}$$

The state vector of the primitive perturbation variables and the Jacobians are

$$\vec{U}' = \begin{pmatrix} \rho' \\ u' \\ v' \\ w' \\ p' \end{pmatrix}, \ \underline{\underline{A}} = \begin{pmatrix} u_0 & \rho_0 & 0 & 0 & 0 \\ 0 & u_0 & 0 & 0 & \frac{1}{\rho_0} \\ 0 & 0 & u_0 & 0 & 0 \\ 0 & 0 & 0 & u_0 & 0 \\ 0 & \gamma \cdot p_0 & 0 & 0 & u_0 \end{pmatrix}, \tag{2.65}$$

$$\underline{\underline{B}} = \begin{pmatrix} v_0 & 0 & \rho_0 & 0 & 0 \\ 0 & v_0 & 0 & 0 & 0 \\ 0 & 0 & v_0 & 0 & \frac{1}{\rho_0} \\ 0 & 0 & 0 & v_0 & 0 \\ 0 & 0 & \gamma \cdot p_0 & 0 & v_0 \end{pmatrix}, \ \underline{\underline{C}} = \begin{pmatrix} w_0 & 0 & 0 & \rho_0 & 0 \\ 0 & w_0 & 0 & 0 & 0 \\ 0 & 0 & w_0 & 0 & 0 \\ 0 & 0 & 0 & w_0 & \frac{1}{\rho_0} \\ 0 & 0 & 0 & \gamma \cdot p_0 & w_0 \end{pmatrix},$$

where $\gamma = 1.4$ is the ratio of specific heats. $\vec{U}_0 = (\rho_0, u_0, v_0, w_0, p_0)^T$ are the mean-values for the background flow and may be either constant for the whole computational domain (global linearization) or dependent on the location of the element ($\vec{U}_0 = \vec{U}_0(\vec{X})$, local linearization). Note that in case of local linearization, the background value at the element's barycenter is used for the whole element in the implementation.

2.2.2 Euler Equations

The nonlinear Euler equations in their conservation form without source terms read:

$$\vec{U}_t + \vec{F}(\vec{U})_x + \vec{G}(\vec{U})_y + \vec{H}(\vec{U})_z = 0, \tag{2.66}$$

with the state vector of the conservative variables and the nonlinear fluxes

$$\vec{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix}, \ \vec{F} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho E + p) \end{pmatrix},$$

$$\vec{G} = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vw \\ v(\rho E + p) \end{pmatrix}, \ \vec{H} = \begin{pmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + p \\ w(\rho E + p) \end{pmatrix}. \tag{2.67}$$

The equation of state of an ideal gas closes the system:

$$p = (\gamma - 1) \cdot \left(\rho E - \frac{1}{2} \rho (u^2 + v^2 + w^2) \right). \tag{2.68}$$

2.2.3 Navier-Stokes Equations

The Navier-Stokes equations in their conservation form without source terms read:

$$\vec{U}_t + \vec{F}(\vec{U})_x + \vec{G}(\vec{U})_y + \vec{H}(\vec{U})_z = \vec{F}^v(\vec{U}, \vec{\nabla}\vec{U})_x + \vec{G}^v(\vec{U}, \vec{\nabla}\vec{U})_y + \vec{H}^v(\vec{U}, \vec{\nabla}\vec{U})_z, \tag{2.69}$$

with the conservative variables $\vec{U}$, the nonlinear Euler fluxes from (2.67) and the diffusion fluxes $\vec{F}^v(\vec{U}, \vec{\nabla}\vec{U})_x$, $\vec{G}^v(\vec{U}, \vec{\nabla}\vec{U})_y$, $\vec{H}^v(\vec{U}, \vec{\nabla}\vec{U})_z$:

$$\vec{F}^v(\vec{U}, \vec{\nabla}\vec{U})_x = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ (\underline{\underline{\tau}} \cdot \vec{v})_x + k \frac{\partial T}{\partial x} \end{pmatrix}, \ \vec{G}^v(\vec{U}, \vec{\nabla}\vec{U})_y = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ \tau_{yz} \\ (\underline{\underline{\tau}} \cdot \vec{v})_y + k \frac{\partial T}{\partial y} \end{pmatrix},$$

$$\vec{H}^v(\vec{U}, \vec{\nabla}\vec{U})_z = \begin{pmatrix} 0 \\ \tau_{zx} \\ \tau_{zy} \\ \tau_{zz} \\ (\underline{\underline{\tau}} \cdot \vec{v})_z + k \frac{\partial T}{\partial z} \end{pmatrix} \tag{2.70}$$

with $k = \frac{c_p \cdot \mu}{Pr}$ and $\vec{v} = (u, v, w)^T$.
The viscous stress tensor is given by

$$\underline{\underline{\tau}} = \mu \left[\left(\vec{\nabla}\vec{v} + (\vec{\nabla}\vec{v})^T \right) - \frac{2}{3}(\vec{\nabla} \cdot \vec{v})\underline{\underline{I}} \right], \tag{2.71}$$

where $\underline{\underline{I}}$ is the unit tensor. The dynamic viscosity μ, the Prandtl number Pr, the adiabatic exponent $\gamma = \frac{c_p}{c_v}$ with the specific heats c_p and c_v depend on the fluid. The system is closed with the equation of state of an ideal gas:

$$p = \rho RT = (\gamma - 1) \cdot \left(\rho E - \frac{1}{2}\rho(u^2 + v^2 + w^2) \right), \tag{2.72}$$

with

$$E = \frac{1}{2}(u^2 + v^2 + w^2) + c_v T \tag{2.73}$$

and the specific gas constant $R = c_p - c_v$.

2.2.4 State Vector Conversion

The continuity of the physical state variables is ensured by the data exchange (section 2.3). Depending on the set of equations, the state vectors either contain the primitive perturbation variables (linearized case) or conservative variables (nonlinear case). If the data exchange takes place amongst domains of equal type (*nonlinear* $\rightleftharpoons$ *nonlinear* or *linear* $\rightleftharpoons$ *linear*), the state vectors can be simply exchanged. However, if the equation type changes (*nonlinear* $\rightleftharpoons$ *linear*), the state vectors must be converted:

Conversion: Nonlinear → linear

1. Transform the conservative state vector $\vec{U}_{con} = (\rho, \rho u, \rho v, \rho w, \rho E)^T$ into its primitive version $\vec{U}_{prim} = (\rho, u, v, w, p)^T$:

$$\rho = \rho, \tag{2.74}$$

$$u = \frac{(\rho u)}{\rho}, \tag{2.75}$$

$$v = \frac{(\rho v)}{\rho}, \tag{2.76}$$

$$w = \frac{(\rho w)}{\rho}, \tag{2.77}$$

$$p = (\gamma - 1) \cdot \left[(\rho E) - \frac{1}{2\rho} \left((\rho u)^2 + (\rho v)^2 + (\rho w)^2 \right) \right]. \tag{2.78}$$

2. Obtain the perturbation vector by subtracting the background values from the linearized domain:

$$\vec{U}'_{prim} = \vec{U}_{prim} - \vec{U}_{0,prim} = \begin{pmatrix} \rho' \\ u' \\ v' \\ w' \\ p' \end{pmatrix} = \begin{pmatrix} \rho \\ u \\ v \\ w \\ p \end{pmatrix} - \begin{pmatrix} \rho_0 \\ u_0 \\ v_0 \\ w_0 \\ p_0 \end{pmatrix}. \quad (2.79)$$

Conversion: Linear → nonlinear

1. Obtain the full primitive state vector by adding the background values from the linearized domain:

$$\vec{U}_{prim} = \vec{U}'_{prim} + \vec{U}_{0,prim} = \begin{pmatrix} \rho \\ u \\ v \\ w \\ p \end{pmatrix} = \begin{pmatrix} \rho' \\ u' \\ v' \\ w' \\ p' \end{pmatrix} + \begin{pmatrix} \rho_0 \\ u_0 \\ v_0 \\ w_0 \\ p_0 \end{pmatrix}. \quad (2.80)$$

2. Transform the primitive state vector $\vec{U}_{prim} = (\rho, u, v, w, p)^T$ into its conservative version $\vec{U}_{con} = (\rho, \rho u, \rho v, \rho w, \rho E)^T$:

$$\rho = \rho, \quad (2.81)$$
$$\rho u = \rho \cdot u, \quad (2.82)$$
$$\rho v = \rho \cdot v, \quad (2.83)$$
$$\rho w = \rho \cdot w, \quad (2.84)$$
$$\rho E = \frac{1}{2}\rho \cdot (u^2 + v^2 + w^2) + \frac{1}{\gamma - 1} \cdot p\,. \quad (2.85)$$

2.3 The Coupling between Grids

The building blocks for the coupling mechanism between the grids are described in the following. The interpolation procedures are depicted for two-dimensional elements, while the actual implementation is also three-dimensional.

2.3.1 Grid Configurations

Two or more different domains Ω_i are coupled at their common boundary $\partial\Omega = \Gamma$ over the data in the ghost elements. Depending on the discretization method, such an element can be a cell (FV and DG methods) or a point (FD methods). These ghosts are then used by the numerical methods to update the inner elements in each domain. The data are exchanged by interpolating the values from the neighbor grid onto the Gauss integration points (in the following simply called Gauss points or GPs) of the ghost cells for FV and DG methods. With a subsequent integration, the mean-values (FV) or the degrees of freedom (DG) in the elements are obtained. For FD methods, the values are interpolated onto the position of the ghost points themselves. The integration for the setting of the ghost element is independent of the way how the interpolation onto the Gauss point is performed. The conservative variables are used for interpolation in the nonlinear case and the primitive perturbation variables in the linearized case. When the coupling procedure is applied to domains with jumping grid sizes (fine $\rightarrow$ coarse), a form of filtering takes place regarding small-scale structures that cannot be resolved on the coarse grid (see section 2.7.2). The number of ghost elements that are required by the numerical method strongly depends on the method itself. The high order FV and FD schemes on structured grids use reconstruction operators and thus need a larger number of ghost elements. Their number scales with the chosen order of accuracy inside the domain and they are needed for all boundary conditions (Table 2.8). On the other hand, DG methods are very local: Usually, no ghost cells are needed for the setting of the boundary conditions, which can be prescribed on the element's boundary side. However, for a coupling boundary, ghost cells can be easily constructed by mirroring the boundary element. Thus, DG domains *will* have one row of ghost elements at their coupling interface. Table 2.9 provides an overview of the method/grid combinations which are examined in the domain decomposition framework. At the coupling interface Γ, the grid configuration can be arbitrary and non-matching. Although the ghost elements extend into the neighbor domain, the coupling method is non-overlapping as the actual domain boundaries only need to be aligned relative

to each other. Figure 2.12 depicts one of many possible setups: A structured FD domain Ω_1 is connected to an unstructured DG domain Ω_2. The outermost domain points of Ω_1 lie directly on the boundary (which is typical for FD schemes). If a fourth order method is used, there are two rows of ghost points, all located inside Ω_2. Last but not least, the DG domain's single row of ghost cells extends into Ω_1.
The domain decomposition approach also simplifies the treatment of more complex configurations with so-called multi-domain cells (Fig. 2.13): The large ghost cell from domain Ω_1 overlaps two different domains Ω_2 and Ω_3. Each GP of the cell, in this case $2^2 = 4$ GPs for a fourth order method, can be assigned a unique element of a neighboring grid that is the basis for the interpolation stencil.

Method	$\mathcal{O}2$	$\mathcal{O}3$	$\mathcal{O}4$	$\mathcal{O}5$	$\mathcal{O}6$	$\mathcal{O}7$	$\mathcal{O}8$
ADER-DG & Rec-FV	1	1	1	1	1	1	1
ADER-FV & FD	1	2	2	3	3	4	4
STE-FV, Central	2	2	3	3	4	4	5
STE-FV, WENO	2	3	4	5	6	7	8

Table 2.8: Number of ghost element rows for the different methods.

Method	Grid type	2D elements	3D elements
ADER-DG	unstructured	triangles	tetrahedrons
Rec-FV	unstructured	triangles	tetrahedrons
ADER-FV	structured, Cartesian	quadrangles	hexahedrons
ADER-FD	structured, Cartesian	points	points
Taylor-FD	structured, Cartesian	points	points

Table 2.9: Implemented grid types and methods.

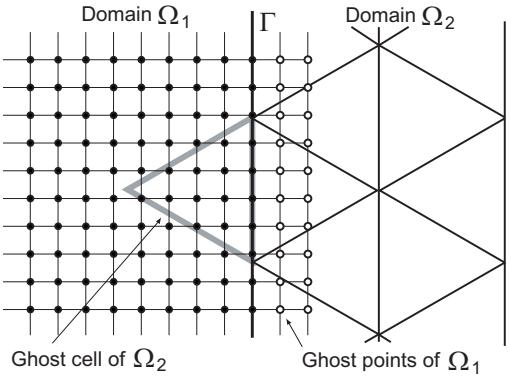


Figure 2.12: A grid configuration example: A FD domain is connected to a DG domain.

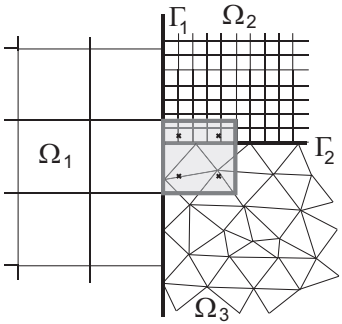
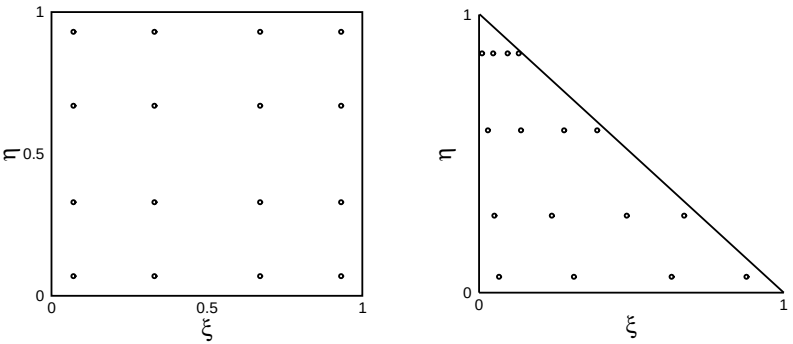


Figure 2.13: A multi-domain ghost cell and its Gauss points.

2.3.2 Interpolation Points

If the coupling ghost element is a finite difference point, the target coordinates $\vec{X}_T = (x, y, z)_T$ for this interpolation are obviously the ghost point coordinates. However, in case of a FV target element, the integral mean-value must be obtained. For a DG element, a projection onto its degrees of freedom must be performed. Both cases require an integration rule for the volume integral of the cell. Gauss quadrature formulas are chosen for this purpose, as high order can be achieved with a low number of integration points, which keeps the additional computational effort for the coupling procedure low. Polynomials of degree $n_{Poly} = 2 \cdot M - 1$ are integrated exactly, where M denotes the number of Gauss integration points in 1D ($M = n_{GP,1D}$). The DG elements demand a greater number of integration points because the polynomial projection involves a multiplication with the base function Φ, which is a polynomial itself (see also Dumbser [18]). The number of Gauss integration points becomes $M = (n_{Poly} + 1)$ in 1D. For 2D ($d = 2$) and 3D ($d = 3$), the total number of Gauss integration points is $n_{GP} = M^d$.
Depending on the element type, its dimension and its location in the physical xyz-space, the coordinates $\vec{X}_{i_{GP}}$ and the weight $\omega_{i_{GP}}$ of every Gauss integra-



FV quadrangle, $\mathcal{O}8$. DG triangle, $\mathcal{O}4$.

Figure 2.14: Number and location of the Gauss quadrature points in the reference elements.

tion point i_{GP} must be determined for each cell. The target coordinates for the interpolation between the grids are then $\vec{X}_T = \vec{X}_{i_{GP}}$. In Fig. 2.14, the number and position of the GPs are shown for a quadrangle and a triangle in the $\xi\eta$-reference system. Note that the total number of GPs is $n_{GP} = 16$ in both cases, although the order is different.

Quadrangles and hexahedrons: Gauss-Legendre quadrature is the natural choice here, as the positions and weights can be determined out of 1D integrations (Stroud [104]) in a dimension-by-dimension manner:

$$\int_0^1 f(\xi)\, d\xi \approx \sum_{i=1}^{M} \omega_i f(\xi_i)\,. \tag{2.86}$$

On the given interval $[0;1]$, the weights A_i and the position ξ_i of the integration points can be calculated up to any degree n_{Poly} by using standard algorithms from the literature [79]. The tensor product of (2.86) delivers the formula for the three-dimensional integration on a reference cube:

$$\int_{\xi=0}^{1}\int_{\eta=0}^{1}\int_{\zeta=0}^{1} f(\xi,\eta,\zeta)\, d\zeta d\eta d\xi \approx \sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{k=1}^{M} \omega_i\omega_j\omega_k f(\xi_i,\eta_j,\zeta_k)\,. \tag{2.87}$$

For two-dimensional integration, the j- and k- indices can simply be skipped. The physical interval sizes Δx_1, Δx_2 and Δx_3 can be fed as integration boundaries to the standard algorithms. In this case, the weights, which then contain the interval sizes, have to be scaled with $\frac{1}{\Delta x_1 \cdot \Delta x_2 \cdot \Delta x_3}$ in order to ensure that the sum of the integration weights equals 1. The position $\vec{X}_{i_{GP}} = (x,y,z)_{i_{GP}}$ of integration point i_{GP} in the physical element is determined by:

$$\begin{aligned} x &= \frac{x_2 - x_1}{2}\xi + \frac{x_2 - x_1}{2}, \\ y &= \frac{y_2 - y_1}{2}\eta + \frac{y_2 - y_1}{2}, \\ z &= \frac{z_2 - z_1}{2}\zeta + \frac{z_2 - z_1}{2}, \end{aligned} \tag{2.88}$$

with $\vec{X}_j = (x,y,z)_j$, $j \in [1,2]$, denoting the integration interval boundaries. The total number of GPs in the element becomes $n_{GP} = n_{GP,1D}^d$. For the sake

of simplicity, every integration point i_{GP} of element Ω is given only one index $i_{GP} \in [1, n_{GP}]$ in the following:

$$\int_{\Omega} f(x,y,z)dxdydz \approx \sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{k=1}^{M} \omega_i\omega_j\omega_k f\left(x_i, y_j, z_k\right) = \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} f\left(\vec{X}_{i_{GP}}\right), \tag{2.89}$$

with

$$\sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} = 1. \tag{2.90}$$

Triangles and tetrahedrons: For triangular and tetrahedral elements, Gauss-Jacobi quadrature is favorable in order to avoid transformations to quadrangles/hexahedrons prior to the integration (Stroud [104]). In one dimension, the Gauss-Jacobi integration reads:

$$\int_{0}^{1} (1-\xi)^{2-k} f\left(\xi_k\right) d\xi_k \approx \sum_{i=1}^{M} \omega_{k,i}\ f\left(\xi_{k,i}\right), \qquad k = 1, 2. \tag{2.91}$$

Again, standard algorithms from the numerical recipes [79] can be used in order to calculate the weights $\omega_{k,i}$ and the position $\xi_{k,i}$ on the interval $[0;1]$ of the Gauss integration points for arbitrary order. Using a conical product of (2.91), the formula for two-dimensional quadrature on a reference triangle can be obtained:

$$\int_{\xi=0}^{1}\int_{\eta=0}^{1-\xi} f\left(\xi,\eta\right) d\eta d\xi \approx \sum_{i=1}^{M}\sum_{j=1}^{M} \omega_{1,i}\omega_{2,j} f\left(\xi_{1,i}, \eta_{2,j}\cdot\left(1-\xi_{1,i}\right)\right). \tag{2.92}$$

In a similar way, the product formula for the three-dimensional integration on a reference tetrahedron can be constructed:

$$\int_{\xi=0}^{1}\int_{\eta=0}^{1-\xi}\int_{\zeta=0}^{1-\xi-\eta} f(\xi,\eta,\zeta)d\zeta d\eta d\xi \tag{2.93}$$

$$\approx \sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{k=1}^{M} \omega_{1,i}\omega_{2,j}\omega_{3,k} f(\xi_{1,i}, \eta_{2,j}\cdot(1-\xi_{1,i}), \zeta_{3,k}\cdot(1-\xi_{1,i})\cdot(1-\eta_{2,j})).$$

If the position of integration point i_{GP} has been determined in the $\xi\eta\zeta$-reference element, its position $\vec{X}_{i_{GP}} = (x,y,z)_{i_{GP}}$ in the physical element is determined

by a transformation, which is defined in 2D by (see also Dumbser [18]):

$$\begin{aligned} x &= x_1 + (x_2 - x_1)\,\xi + (x_3 - x_1)\,\eta, \\ y &= y_1 + (y_2 - y_1)\,\xi + (y_3 - y_1)\,\eta. \end{aligned} \tag{2.94}$$

In 3D, the transformation reads:

$$\begin{aligned} x &= x_1 + (x_2 - x_1)\,\xi + (x_3 - x_1)\,\eta, + (x_4 - x_1)\,\zeta, \\ y &= y_1 + (y_2 - y_1)\,\xi + (y_3 - y_1)\,\eta, + (y_4 - y_1)\,\zeta, \\ z &= z_1 + (z_2 - z_1)\,\xi + (z_3 - z_1)\,\eta, + (z_4 - z_1)\,\zeta, \end{aligned} \tag{2.95}$$

with $\vec{X}_j = (x_j, y_j, z_j)$, $j \in [1, 2, 3, 4]$, being the physical coordinates of the element's vertices. In physical xyz-space, the integration weights finally need to be scaled by a factor 2 (2D, triangles) or by a factor 6 (3D, tetrahedrons). Again, the total number of GPs in the element becomes $n_{GP} = M^d$ and every integration point i_{GP} of element Ω is given only one index $i_{GP} \in [1, n_{GP}]$:

$$\int_{\Omega} f(x, y, z) dx dy dz \approx \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} f\left(\vec{X}_{i_{GP}}\right), \tag{2.96}$$

with property (2.90).

2.3.3 Structured Source Domains

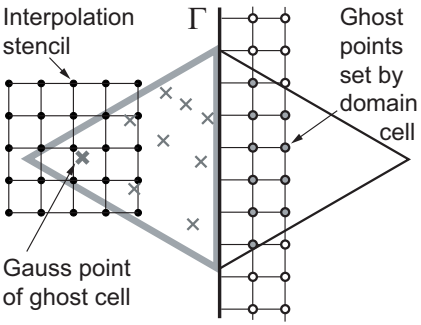


Figure 2.15: A DG domain is connected to a FD domain: Close-up of Gauss integration points, interpolation stencils and ghost points.

The order of interpolation from a structured source domain (the "donor") may be arbitrary. However, because the structured high order methods are already based on reconstructions, it makes sense to use the source domain's order of accuracy also for the interpolation.
Uneven numbers of interpolation stencil elements in each direction provide more symmetrical stencils. Furthermore, the number of ghost elements that can be additionally used for the interpolation at the boundary is limited by the order in the domain. Therefore, symmetrical interpolation stencils are preferred, which are either of the same order $\mathcal{O}$ as the domain's numerical method (if $\mathcal{O}$ is uneven) or $\mathcal{O}+1$ (if $\mathcal{O}$ is even). In this regard, "symmetrical" means that the nearest stencil element for a given interpolation coordinate $\vec{X}_T$ is the central point in the stencil. Figure 2.15 shows the $[5 \times 5]$-stencil for the interpolation from the $\mathcal{O}4$ FD grid onto one of the GPs of a $\mathcal{O}3$ DG ghost cell. The central stencil point is found by a "nearest neighbor to the GP position" search on the structured grid.
If the same interpolation order is used also for very high order methods, the interpolation stencils would become very large, for example $13^3 = 2197$ elements for $\mathcal{O} = 12$ in 3D. This would result in a large computational overhead of the coupling procedure. On the other hand, in the most cases, such a high order interpolation is not necessary for the preservation of a high-quality solution.

Anyway, very high order inside the domains is primarily used for ensuring good wave propagation properties (e.g., low dissipation). Moreover, interpolations of a very high order tend to produce spurious oscillations (Munz [76]). Therefore and if not mentioned otherwise, the order of interpolation is limited to a maximum of $\mathcal{O}5$ in the presented numerical examples. Exceptions are convergence studies where the interpolation order has to match the desired accuracy.
A distinction similar to restrictions and prolongations must be made for the Cauchy-Kovalevskaja procedure (see section 2.4): If the target element's domain has a smaller time step than the neighbor-domain, not only the state vector but also its spatial derivatives have to be determined at the interpolation target coordinate $\vec{X}_T$.
Polynomial interpolation is favorable here because the differentiation of the polynomial directly delivers the state derivative. Furthermore, 2D and 3D interpolations can be easily obtained by a product of 1D interpolations. Last but not least, the interpolation coefficients can be precalculated and stored for all target coordinates, so they only have to be multiplied with the states in the stencil elements in order to obtain the value for the state and derivatives at $\vec{X}_T$. This reduces the computational overhead of the coupling procedure significantly.

Interpolation from an FD domain The state vector $\vec{U}_T$ is interpolated from the regular FD mesh onto the target position $\vec{X}_T$.
In 1D, the function

$$L_i(x) = \frac{(x-x_0)(x-x_1)...(x-x_{i-1})(x-x_{i+1})...(x-x_n)}{(x_i-x_0)(x_i-x_1)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_{n_{Poly}})} \tag{2.97}$$

is a polynomial of degree n_{Poly} with the properties

$$L_i(x_j) = \begin{cases} 1 & \text{for } i = j, \\ 0 & \text{for } i \neq j \end{cases} \tag{2.98}$$

and

$$\sum_{i=0}^{n_{Poly}} L_i(x) = 1. \tag{2.99}$$

It defines a polynomial

$$p_{n_{Poly}}(x) = \sum_{i=0}^{n_{Poly}} u_i L_i(x). \tag{2.100}$$

with the function values u_i at the positions x_i.
For a multi-dimensional interpolation, the Lagrangian polynomials $L_i(x)$, $L_j(y)$ and $L_k(z)$ can be calculated independently. Their product delivers the coefficients for the 3D interpolation:

$$L_{ijk}(\vec{X}) = L_i(x) \cdot L_j(y) \cdot L_k(z), \tag{2.101}$$

with $i, j, k \in [0, n_{Poly}]$. These coefficients can be stored for each interpolation target coordinate $\vec{X}_T$ for computational efficiency.
The state vector $\vec{U}_T$ at position $\vec{X}_T$ is then obtained by:

$$\vec{U}_T = \sum_{i=0}^{n_{Poly}} \sum_{j=0}^{n_{Poly}} \sum_{k=0}^{n_{Poly}} L_{ijk}(\vec{X}_T) \cdot \vec{U}_{ijk}, \tag{2.102}$$

with $\vec{U}_{ijk} = (\rho, u, v, w, p)^T_{ijk}$ being the state vector at the stencil point with the indices i,j,k.
For the sake of simplicity, these indices can be mapped onto one single parameter i_S with $n_S = (n_{Poly} + 1)^d$ elements:

$$\vec{U}_T = \sum_{i_S=1}^{n_S} L_{i_S}(\vec{X}_T) \cdot \vec{U}_{i_S}. \tag{2.103}$$

If spatial derivatives are needed, the Lagrangian polynomial from (2.101) can be easily differentiated with respect to the pth, qth and rth derivatives in x- y- and z-direction:

$$\frac{\partial^{p+q+r}\vec{U}_T}{\partial x^p \partial y^q \partial z^r} = \sum_{i=0}^{n_{Poly}} \sum_{j=0}^{n_{Poly}} \sum_{k=0}^{n_{Poly}} \frac{\partial^{p+q+r} L_{ijk}(\vec{X}_T)}{\partial x^p \partial y^q \partial z^r} \cdot \vec{U}_{ijk}, \tag{2.104}$$

or

$$\frac{\partial^{p+q+r}\vec{U}_T}{\partial x^p \partial y^q \partial z^r} = \sum_{i_S=1}^{n_S} \frac{\partial^{p+q+r} L_{i_S}(\vec{X}_T)}{\partial x^p \partial y^q \partial z^r} \cdot \vec{U}_{i_S}. \tag{2.105}$$

Interpolation from a FV domain In case of a FV source domain, the polynomial interpolation onto a specific coordinate $\vec{X}_T$ is slightly different from the previous approach because the available data in the stencil cells are mean-values instead of point values. Hence, functions $\bar{L}_i(x)$ must be found that build a polynomial

$$u(x) = p_{n_{Poly}}(x) = \sum_{i=0}^{n_{Poly}} \bar{u}_i \bar{L}_i(x) = \sum_{j=0}^{n_{Poly}} c_j x^j, \tag{2.106}$$

for which the integral mean-values $\bar{u}_i$ in the interpolation stencil cells Ω_i are conserved:

$$\bar{u}_i = \frac{1}{\Delta x_i} \int\limits_{\Omega_i} p_{n_{Poly}}(x) dx, \ \forall i \in [0, n_{Poly}]. \tag{2.107}$$

From (2.106) and (2.107) follows:

$$\begin{aligned}
\bar{u}_i &= \frac{1}{\Delta x_i} \int\limits_{\Omega_i} \sum_{j=0}^{n_{Poly}} c_j x^j dx, \ \forall i \in [0, n_{Poly}], \\
\bar{u}_i &= \sum_{j=0}^{n_{Poly}} c_j \cdot \frac{1}{\Delta x_i} \int\limits_{\Omega_i} x^j dx, \ \forall i \in [0, n_{Poly}], \\
\bar{u}_i &= \sum_{j=0}^{n_{Poly}} M_{i,j} \cdot c_j, \ \forall i \in [0, n_{Poly}], \\
\bar{u}_i &= \underline{\mathrm{M}}_i^T \, \underline{\mathrm{C}}, \ \forall i \in [0, n_{Poly}], \\
\underline{\bar{\mathrm{U}}} &= \underline{\underline{M}} \, \underline{\mathrm{C}},
\end{aligned} \tag{2.108}$$

with $M_{ij} = \frac{1}{\Delta x_i} \int\limits_{\Omega_i} x^j dx$, $j \in [0, n_{Poly}]$, $\underline{\bar{\mathrm{U}}} = (\bar{u}_0, ..., \bar{u}_{n_{Poly}})^T$, and $\underline{\mathrm{C}} = (c_0, ..., c_{n_{Poly}})^T$.
The interpolation coefficients $\underline{\mathrm{C}}$ are then

$$\underline{\mathrm{C}} = \underline{\underline{M}}^{-1} \, \underline{\bar{\mathrm{U}}}. \tag{2.109}$$

On the other hand, remembering (2.106), the solution u at a certain position x can be written as:

$$u(x) = \sum_{j=0}^{n_{Poly}} c_j x^j = \underline{\mathrm{X}}^T \, \underline{\mathrm{C}}, \tag{2.110}$$

with $\underline{\mathrm{X}}^T = (1, x, x^2, ..., x^j, ..., x^{n_{Poly}})$. Be careful not to confuse the transposed vector $\underline{\mathrm{X}}^T$ of the monomials with the target coordinate $\vec{X}_T = (x, y, z)_T$ for the interpolation!
Inserting (2.109) into (2.110) yields:

$$\begin{aligned}
u(x) &= \underline{\mathrm{X}}^T \, \underline{\underline{M}}^{-1} \, \underline{\bar{\mathrm{U}}}, \\
&= \underline{\bar{\mathrm{L}}}^T \, \underline{\bar{\mathrm{U}}},
\end{aligned} \tag{2.111}$$

with $\underline{\bar{\mathrm{L}}}^T = \underline{\bar{\mathrm{L}}}^T(x) = (\bar{L}_0(x), \bar{L}_1(x), ..., \bar{L}_{n_{Poly}}(x))$ being the functions sought in (2.106), which can be easily calculated by computing $\underline{\mathrm{X}}^T$ (e.g., for a specific target position x_T) and $\underline{\underline{M}}^{-1}$ first.
If spatial derivatives are needed for a CK procedure (see section 2.4), they can be constructed by differentiating (2.110):

$$\begin{aligned} \frac{\partial^p u(x)}{\partial x^p} &= \frac{\partial^p}{\partial x^p}(\underline{\mathrm{X}}^T)\,\underline{\mathrm{C}}, \\ &= \frac{\partial^p}{\partial x^p}(\underline{\mathrm{X}}^T)\,\underline{\underline{M}}^{-1}\,\underline{\bar{\mathrm{U}}}, \\ &= \frac{\partial^p}{\partial x^p}(\underline{\bar{\mathrm{L}}}^T)\,\underline{\bar{\mathrm{U}}}. \end{aligned} \tag{2.112}$$

Again, 2D and 3D interpolations can be obtained by multiplying the corresponding the entries of $\underline{\bar{\mathrm{L}}}^T(x)$, $\underline{\bar{\mathrm{L}}}^T(y)$ and $\underline{\bar{\mathrm{L}}}^T(z)$:

$$\bar{L}_{ijk}(\vec{X}) = \bar{L}_i(x) \cdot \bar{L}_j(y) \cdot \bar{L}_k(z), \tag{2.113}$$

with $i, j, k \in [0, n_{Poly}]$. Note that by using this tensor product approach, the integral conservation property from (2.107) is indeed preserved for each 2D and 3D element!
The state vector $\vec{U}_T$ at position $\vec{X}_T$ is finally computed by:

$$\vec{U}_T = \sum_{i=0}^{n_{Poly}} \sum_{j=0}^{n_{Poly}} \sum_{k=0}^{n_{Poly}} \bar{L}_{ijk}(\vec{X}_T) \cdot \vec{\bar{U}}_{ijk}, \tag{2.114}$$

with $\vec{\bar{U}}_{ijk} = (\bar{\rho}, \bar{u}, \bar{v}, \bar{w}, \bar{p})^T_{ijk}$ being the mean-value state vector for the stencil cell with the indices i,j,k.
Again for simplicity, the indices i,j,k can be mapped onto one single parameter i_S with $n_S = (n_{Poly} + 1)^d$ elements:

$$\vec{U}_T = \sum_{i_S=1}^{n_S} \bar{L}_{i_S}(\vec{X}_T) \cdot \vec{\bar{U}}_{i_S}. \tag{2.115}$$

The 3D coefficients for the derivatives of the state vector are calculated the same way and are stored for each interpolation target coordinate $\vec{X}_T$ in order to increase computational efficiency.
Note that the coefficients for the interpolation from a FD grid can also be easily computed with this approach, only with the modification that $\bar{u}_i(x_i)$ is now the value at the position x_i. This has to be considered in (2.107) and (2.108).

2.3.4 Unstructured Source Domains

If an interpolation coordinate, for example a ghost point, is connected to an unstructured domain, it can be assigned a unique neighbor cell. This source element S is the cell, in which the target coordinate is located. Again, the interpolation method depends on the numerical scheme that is used on the unstructured grid. However, the interpolation procedure is in the following described for DG and Rec-FV only, as these are mainly used in the coupling framework. The interpolation for traditional FV methods on unstructured grids (e.g., by reconstructing gradients) is rather straight forward and will not be discussed here.
As a DG cell's degrees of freedom (DOF) contain the complete polynomial information, there is no need for an interpolation stencil and the value of the state vector $\vec{U}_T$ at the target position $\vec{X}_T$ can be obtained by using the definition (Dumbser [18]) of the numerical solution inside a DG cell:

$$\vec{U}_T = \vec{U}(\vec{X_T}) = \sum_{i_{DOF}=1}^{n_{DOF}} \phi_{i_{DOF}}(\xi, \eta, \zeta) \cdot \vec{\hat{U}}_{i_{DOF}}(S), \tag{2.116}$$

with the degrees of freedom $\vec{\hat{U}}_{i_{DOF}}(S) = (\hat{\rho}, \hat{u}, \hat{v}, \hat{w}, \hat{p})^T_{i_{DOF}}$ (exemplarily for primitive variables) of DG source cell S and the value of the basis function $\phi_{i_{DOF}}(\xi, \eta, \zeta)$ at position $\vec{X}_T$ of the point. The order of the interpolation equals the order of the DG scheme that is used.
In order to fit the style of (2.103) and (2.115) and to make the implementation in the framework more general, equation (2.116) can be interpreted and rewritten as

$$\vec{U}_T = \vec{U}(\vec{X_T}) = \sum_{i_S=1}^{n_S} L_{i_S} \cdot \vec{U}_{i_S}(S), \tag{2.117}$$

regarding the degrees of freedom as the "stencil cells" ($n_{DOF} = n_S$) and so on. Spatial derivatives of the state vector are obtained in the same way by using the derivatives of the basis function, which are available in the DG framework. Note that the Rec-FV methods for unstructured grids are based on the DG framework, so the interpolation procedure for those source elements is the same as for DG elements. The only difference is that a reconstruction step has to be performed prior to the interpolation in order to obtain the full polynomial information inside the source cell. In this case, there is in fact a reconstruction stencil, but it is hidden in the numerical method and is not "seen" by the coupling procedure. If the stencil becomes large, it may be asymmetrical at

the boundaries.
The transformations from the xyz-coordinates into the $\xi\eta\zeta$-reference system are the inverse transformations of (2.94) and (2.95) (see also Dumbser et al. [18, 19]).
In 2D, the transformation is defined by:

$$\xi = \frac{1}{\left|\underline{\underline{J}}\right|}\left((x_3y_1 - x_1y_3) + x\left(y_3 - y_1\right) + y\left(x_1 - x_3\right)\right), \tag{2.118}$$

$$\eta = \frac{1}{\left|\underline{\underline{J}}\right|}\left((x_1y_2 - x_2y_1) + x\left(y_1 - y_2\right) + y\left(x_2 - x_1\right)\right), \tag{2.119}$$

where $\left|\underline{\underline{J}}\right| = \left(x_2 - x_1\right)\left(y_3 - y_1\right) - \left(x_3 - x_1\right)\left(y_2 - y_1\right)$ is the determinant of the transformation's Jacobian matrix

$$\underline{\underline{J}} = \begin{pmatrix} x_\xi & y_\xi \\ x_\eta & y_\eta \end{pmatrix}. \tag{2.120}$$

$\left|\underline{\underline{J}}\right|$ is equal to the double of the triangle's surface.
In 3D, the transformation reads:

$$\begin{aligned}\xi = \frac{1}{\left|\underline{\underline{J}}\right|}(&x(+z_4y_3 + z_1y_4 - z_3y_4 - z_1y_3 - z_4y_1 + z_3y_1)\\ &+y(-x_1z_3 + z_4x_1 - z_4x_3 + z_1x_3 - z_1x_4 + x_4z_3)\\ &+z(-y_1x_3 + y_3x_1 + y_1x_4 + y_4x_3 - y_3x_4 - y_4x_1)\\ &+(x_1z_3y_4 - z_3y_1x_4 + z_1x_4y_3 - x_1z_4y_3 - z_1x_3y_4 + z_4y_1x_3)),\end{aligned} \tag{2.121}$$

$$\begin{aligned}\eta = \frac{1}{\left|\underline{\underline{J}}\right|}(&-x(z_4y_2 - z_1y_2 + z_1y_4 + z_2y_1 - z_2y_4 - z_4y_1)\\ &-y(z_2x_4 - z_2x_1 - z_1x_4 - z_4x_2 + z_4x_1 + z_1x_2)\\ &-z(-y_2x_4 + y_1x_4 + y_4x_2 - y_1x_2 - y_4x_1 + y_2x_1)\\ &-(-z_2y_1x_4 + x_1z_2y_4 - x_1z_4y_2 - z_1x_2y_4 + z_1x_4y_2 + z_4y_1x_2)),\end{aligned} \tag{2.122}$$

$$\begin{aligned}\zeta = \frac{1}{\left|\underline{\underline{J}}\right|}(&x(z_1y_3 - z_3y_1 - z_1y_2 + z_3y_2 + z_2y_1 - z_2y_3)x\\ &+y(x_1z_3 - z_2x_1 - z_3x_2 + z_1x_2 + z_2x_3 - z_1x_3)y\\ &+z(y_2x_1 - y_3x_1 + x_2y_3 - y_1x_2 + y_1x_3 - x_3y_2)z\\ &+(-x_1z_3y_4 - z_4x_2y_3 + z_1x_3y_4 - z_3x_4y_2\\ &\quad - z_2y_4x_3 - z_4y_1x_3 + z_3y_1x_4 - z_2y_1x_4\\ &\quad + x_1z_2y_4 - x_1z_4y_2 - z_1x_2y_4 + z_1x_4y_2\\ &\quad + z_4y_1x_2 + x_1z_4y_3 + z_3y_4x_2 + z_4x_3y_2\\ &\quad - z_1x_4y_3 + z_2x_4y_3 + \left|\underline{\underline{J}}\right|),\end{aligned} \tag{2.123}$$

where

$$\begin{aligned}\left|\underline{\underline{J}}\right| = &- z_1x_2y_3 + x_1z_3y_4 + x_1z_2y_3 + z_4x_2y_3\\ &+ z_3x_4y_2 + z_2y_4x_3 + z_4y_1x_3 - z_3y_1x_4\\ &+ z_3y_1x_2 - x_1z_2y_4 + x_1z_4y_2 + z_1x_2y_4\\ &+ z_2y_1x_4 - z_4y_1x_2 - x_1z_4y_3 - z_3y_4x_2\\ &- z_1x_3y_4 - z_4x_3y_2 - z_2x_4y_3 + z_1x_3y_2\\ &- z_2y_1x_3 - z_1x_4y_2 - x_1z_3y_2 + z_1x_4y_3\end{aligned} \tag{2.124}$$

is the determinant of the transformation's Jacobian matrix

$$\underline{\underline{J}} = \begin{pmatrix} x_\xi & y_\xi & z_\xi \\ x_\eta & y_\eta & z_\eta \\ x_\zeta & y_\zeta & z_\zeta \end{pmatrix}. \tag{2.125}$$

$\left|\underline{\underline{J}}\right|$ is six times the tetrahedron's volume.

2.3.5 Interpolation Stencil Symmetry

During the data exchange, the interpolation procedures between the domains can demand a certain logical sequence because stencils may interfere with each other.

No special treatment is necessary if two or more unstructured domains are linked together. As it was illustrated in section 2.3.4, no stencil cells are needed for the interpolation (although there is a reconstruction stencil for the Rec-FV method, which is hidden in the method). Hence, every target Gauss point in each coupling ghost cell can be set independently from the ghost elements of the source domain.

For connections between an unstructured and a structured domain, the target points of the structured ghost elements should be set first by the unstructured source domain. Then those ghost elements already contain updated information in case that one of them is required as a stencil element for the interpolation of the unstructured ghost Gauss points. This might occur for symmetrical stencils.

A more complex logic is needed where solely structured grids share a common interface. In this case, the interpolation stencils may extend into the ghost element regions of both domains. The previous strategy would run in circles here because a piece of stencil information would always be missing. In order to be able to perform a completely independent procedure (like for the unstructured - unstructured coupling), asymmetrical stencils which do not contain their own ghost elements would be required. However, this idea is discarded: It seems non-physical to use two opposing one-sided stencils at the same time, when it is not clear in which direction the information travels at the interface. The more favorable approach is to ensure a symmetrical interpolation from the coarse grid onto the target elements of the fine grid, first of all (Fig. 2.18). On the one hand, a coarse one-sided interpolation stencil would introduce much greater interpolation errors than a fine one-sided stencil. On the other hand, the target points of the coarse ghost elements are then the first to be set by a possibly very much finer stencil, which might not even require information from its own ghost elements. If they do, the stencil is simply shifted with regard to the interface (Stencil 1. in Fig. 2.17).

There is, however, another way of making symmetrical stencils possible on both the fine mesh (Stencil 2. and 3. in Fig. 2.17) and the coarse mesh (Fig. 2.18). The routine is illustrated in Fig. 2.16 for the example of a fine FV domain coupling with a coarser FD grid: First of all, the ghost points of the FD domains are interpolated. If the stencil contains a ghost cell (the hatched

element in the left drawing of Fig. 2.16) of its own domain, this cell will not be on the current time level yet. Now a Taylor series in time (known from section 2.3.6 and 2.4) is used to provide an interim value. Note that this preliminary time update is in fact the "last" possible Taylor step inside the stability region. The Taylor series is based on the data of the previous data exchange and will be immediately replaced by an updated version after the on-going data exchange. The stencil may also extend into the ghost cell region of a boundary which is not a coupling interface (Stencil 3. in Fig. 2.17). In this case, these elements can be set by prescribing the usual boundary conditions (e.g., inflow, outflow, wall) first and are then available for the interpolation.
After the ghost points of the FD domain in the example have been treated, they can be used for the actual symmetrical interpolation onto the FV mesh's ghost Gauss points (right drawing in Fig. 2.16). The target points in the ghost cells (of which some of them had been given preliminary values before) receive now their real values for the current time level!
This procedure proved to be very stable in the conducted numerical experiments. Moreover, it is favorable for more complex grid configurations. If domains with different time steps are linked with several others at the same time, it is possible that ghost element information is required by a source domain which is connected to another domain that is not even taking part in the current subcycle and thus in the data exchange!

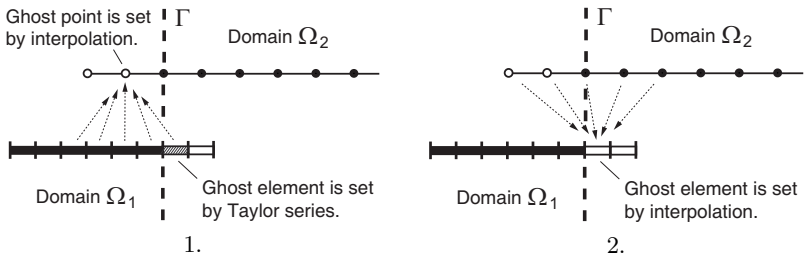


Figure 2.16: Symmetrical interpolation stencil by employing the CK procedure: 1.: Ghost points of the coarser domain are interpolated. Stencil cells are set before by the CK procedure where necessary. 2.: Ghost cells of the finer grid are interpolated.

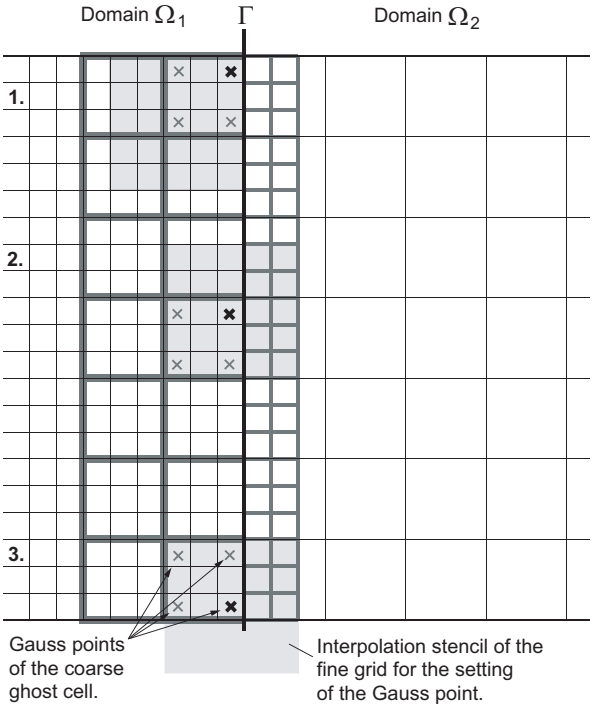


Figure 2.17: Interpolation from a fine grid onto the Gauss points of a coarse grid. Fourth order FV methods are used in both domains. The grey shaded areas contain the necessary stencil for a fifth order interpolation onto the **bold** Gauss point. 1.: Asymmetrical. 2.: Symmetrical. 3.: Symmetrical at the boundaries.

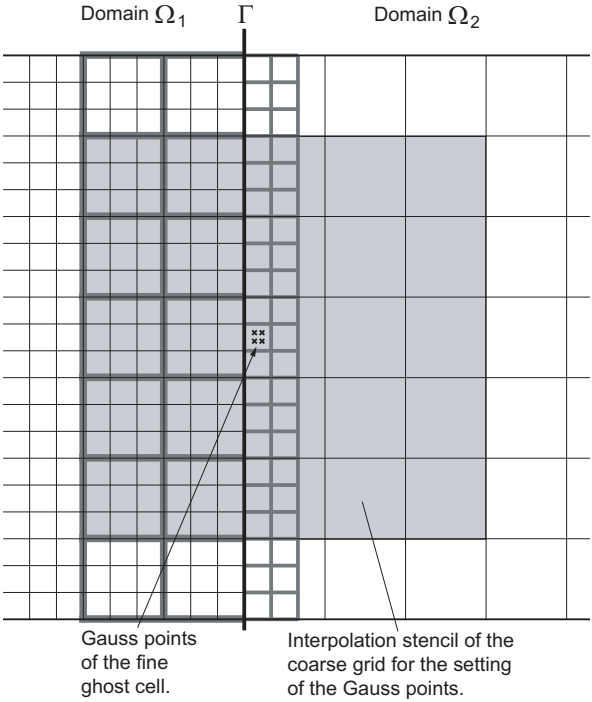


Figure 2.18: Interpolation from a coarse grid onto the Gauss points of a fine grid. The fifth order interpolation stencil (grey shaded area) is symmetrical.

2.3.6 Setting the Ghost Elements

In the following, the algorithms for the setting of the coupling boundary conditions are described for the different element types. After the data exchange between connecting domains is complete, all necessary information for the actual setting of the ghost elements is available. The latter is done in the framework of the usual boundary condition treatment that has to be performed in each domain for every iteration step. If the domain takes part in a subcycle and its current time level t^c lies in-between the data exchanges, there is no current data available for the ghost elements. In this case, a CK procedure must be performed, using the spatial derivatives of the state vector. After this, both the state vector $\vec{U}$ and its Taylor series in time

$$\vec{U}(\vec{X}, t^n + \tau) = \vec{U}(\vec{X}, t^n) + \sum_{k=1}^{n_{Poly}} \frac{\tau^k}{k!} \frac{\partial \vec{U}^k(\vec{X}, t^n)}{\partial^k t}, \tag{2.126}$$

which is based on the time level t^n of the last data exchange are available for every ghost- or Gauss point (see section 2.4 for details) at position $\vec{X}$. The Taylor series can be developed up to the degree n_{Poly}, which is bounded by the order of accuracy of the source domain and the interpolation from it.

Algorithm 2.3.1. ***FD Coupling Ghost Points*** *For all target ghost points T:*

1. *Lift the state vector from the time level of the data exchange t^n up to the current time level t^c at the ghost point position $\vec{X}_T$:*

$$\vec{U}_T(\vec{X}_T, t^c) = \vec{U}_T(\vec{X}_T, t^n) + \sum_{k=1}^{n_{Poly}} \frac{\Delta t^k}{k!} \frac{\partial \vec{U}_T^k(\vec{X}_T, t^n)}{\partial^k t}, \tag{2.127}$$

   with $\Delta t = t^c - t^n$.

2. *If the Taylor-FD method is used, also provide current data for the time derivatives of $\vec{U}$, which can be obtained like the state itself.*

Algorithm 2.3.2. ***FV Coupling Ghost Cells*** *For all target ghost cells T:*

1. *Lift the state vector at every Gauss point position $\vec{X}_{T,i_{GP}}$ from the time level of the data exchange t^n up to the current time level t^c. The Gauss points may have different source time levels t^n if the cell is a multi-domain cell.*

2. *Perform the Gauss integration in order to obtain the mean-value in the cell:*

$$\vec{U}_T = \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \vec{U}_{T,i_{GP}}, \tag{2.128}$$

with the integration weights $\omega_{i_{GP}}$ *from* (2.89).

Algorithm 2.3.3. ***DG Coupling Ghost Cells*** *For all target ghost cells* T*:*

1. *Lift the state vector at every Gauss point position* $\vec{X}_{T,i_{GP}}$ *from the time level of the data exchange* t^n *up to the current time level* t^c*. The Gauss points may have different source time levels* t^n *if the cell is a multi-domain cell.*

2. *Perform the projection onto the cell's degrees of freedom:*

$$\vec{\hat{U}}_{T,i_{DOF}} = \sum_{i_{GP}=1}^{n_{GP}} \frac{\omega_{i_{GP}} \cdot \phi_{i_{GP},i_{DOF}} \cdot \vec{U}_{T,i_{GP}}}{M_{i_{DOF},i_{DOF}}}, \tag{2.129}$$

with $\vec{\hat{U}}_{i_{DOF}} = (\hat{\rho}, \hat{u}, \hat{v}, \hat{w}, \hat{p})^T_{i_{DOF}}$ *(exemplarily for primitive variables), the integration weights* $\omega_{i_{GP}}$ *from* (2.96)*, the DG method's mass matrix* $M_{i_{DOF},i_{DOF}}$ *(Dumbser [18]) and the value of the basis function* $\phi_{i_{GP},i_{DOF}}$ *at the Gauss point position* $\vec{X}_{T,i_{GP}}$ *for the degree of freedom* i_{DOF}*.*

2.3.7 Pre-Integration

It is obvious from the described interpolation procedures that the computational effort increases proportionally with the number of target points that have to be set. For high order DG and FV cells, this number can become very high. Under certain circumstances it makes sense to sum up the interpolation coefficients and integration weights in the initialization part prior to the calculation loop, which is referred to as "pre-integration". This means effectively that during the data exchange, only one interpolation per cell has to be performed, instead of one interpolation per Gauss point. This resembles a "condensation" of all Gauss points per element onto one single point, but in fact all Gauss points are actually considered in the pre-multiplied factors!
The conditions, under which a pre-integration can be performed, are:

1. An FV element must originate from a linearized domain.
2. A DG element may originate from a nonlinear or a linearized domain.
3. The interpolation stencil must be the same for every target point in the considered target cell (Fig. 2.18 gives an example). For instance, the target cell must not be a multi-domain cell! Often the same stencil can be used if the target cell is much smaller than the grid size on the source domain. Also "unstructured stencils" are allowed, for example one single, large DG cell which completely covers a small target element.

This way, instead of getting the values at the single Gauss point positions after the data exchange, one obtains immediately the mean-value (FV cell) or the degrees of freedom (DG cell) in the element.
However, if a CK procedure becomes necessary in order to lift the considered cell up to the current time level (see sections 2.3.6 and 2.4), the values at the Gauss point positions are required nevertheless in some cases.
For DG schemes, the volume quadrature points are needed for the projection onto the degrees of freedom (see (2.129)). Hence, the polynomial inside the DG cell has to be evaluated for state and spatial derivatives at the Gauss points first. Then the coefficients for the Taylor series in time can be determined and the Gauss point values are lifted to the current time level.
The state and its spatial derivatives at all Gauss points are also required in case of nonlinear FV elements, for which *no* pre-integration can be made: The nonlinear CK procedure depends on the Jacobians $\underline{\underline{A}}(\vec{U})$, $\underline{\underline{B}}(\vec{U})$ and $\underline{\underline{C}}(\vec{U})$, which themselves depend on the state $\vec{U}(\vec{X})$ at the Gauss point positions $\vec{X}_{i_{GP}}$. Different to a DG element, no evaluation at those positions can be made later,

only the mean-value is available. Therefore, just linearized elements will be allowed for pre-integration, as their Jacobians and thus the CK procedure are independent of the state.
A demonstration shall be given for a FV cell. The procedure works similar for DG elements, for which additionally the basis functions and mass matrices have to be considered ((2.129)) for the projection.
Following (2.128) and (2.103), the mean-value in a FV coupling cell T is obtained at the time level t^n of the data exchange by:

$$\begin{aligned}\vec{\bar{U}}_T(t^n) &= \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \vec{U}_{T,i_{GP}}(t^n) \\ &= \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \left(\sum_{i_S=1}^{n_S} L_{i_S} \cdot \vec{U}_{i_S}(t^n) \right)_{T,i_{GP}}, \end{aligned} \tag{2.130}$$

or generally, if spatial derivatives are also sought (with (2.105)):

$$\begin{aligned}\frac{\partial^{p+q+r}\vec{\bar{U}}_T(t^n)}{\partial x^p \partial y^q \partial z^r} &= \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \frac{\partial^{p+q+r}\vec{U}_{T,i_{GP}}(t^n)}{\partial x^p \partial y^q \partial z^r} \\ &= \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \left(\sum_{i_S=1}^{n_S} \frac{\partial^{p+q+r} L_{i_S}}{\partial x^p \partial y^q \partial z^r} \cdot \vec{U}_{i_S}(t^n) \right)_{T,i_{GP}} \end{aligned} \tag{2.131}$$

where $\frac{\partial^{p+q+r}\vec{\bar{U}}_T(t^n)}{\partial x^p \partial y^q \partial z^r} = \vec{\bar{U}}_T(t^n)$ for $p,q,r = 0$.
If every Gauss point i_{GP} of the target cell has the same interpolation stencil with values $\vec{U}_{i_S}(t^n)$, (2.131) yields:

$$\begin{aligned}\frac{\partial^{p+q+r}\vec{\bar{U}}_T(t^n)}{\partial x^p \partial y^q \partial z^r} &= \sum_{i_S=1}^{n_S} \left(\underbrace{\sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \frac{\partial^{p+q+r} L_{i_S}}{\partial x^p \partial y^q \partial z^r}}_{Pre-integration} \cdot \vec{U}_{i_S}(t^n) \right)_{T,i_S} \\ &= \sum_{i_S=1}^{n_S} \frac{\partial^{p+q+r}\Psi_{i_S}}{\partial x^p \partial y^q \partial z^r} \cdot \vec{U}_{i_S}(t^n). \end{aligned} \tag{2.132}$$

The coefficients $\frac{\partial^{p+q+r}\Psi_{i_S}}{\partial x^p \partial y^q \partial z^r}$ can be conveniently computed and saved prior to the actual calculation.

For mean-values at the current time level t^c, time derivatives are required, which can be again obtained by the CK procedure:

$$\vec{\bar{U}}_T(t^c) = \vec{\bar{U}}_T(t^n) + \sum_{k=1}^{n_{Poly}} \frac{\Delta t^k}{k!} \frac{\partial \vec{\bar{U}}_T^k(t^n)}{\partial^k t}, \tag{2.133}$$

with

$$\frac{\partial^k \vec{\bar{U}}_T(t^n)}{\partial t^k} = CK_k \left(\frac{\partial^{p+q+r} \vec{\bar{U}}_T(t^n)}{\partial x^p \partial y^q \partial z^r} \right), \ \forall \ 0 \leq p+q+r \leq k. \tag{2.134}$$

This is only valid for linearized FV elements, where the CK procedure becomes linear itself, as the Jacobians do not depend on the states in the Gauss points:

$$\frac{\partial^k \vec{\bar{U}}_T(t^n)}{\partial t^k} \equiv \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot \frac{\partial^k \vec{\bar{U}}_{T,i_{GP}}(t^n)}{\partial t^k}, \tag{2.135}$$

$$CK_k \left(\frac{\partial^{p+q+r} \vec{\bar{U}}_T(t^n)}{\partial x^p \partial y^q \partial z^r} \right) \equiv \sum_{i_{GP}=1}^{n_{GP}} \omega_{i_{GP}} \cdot CK_k \left(\frac{\partial^{p+q+r} \vec{\bar{U}}_{T,i_{GP}}(t^n)}{\partial x^p \partial y^q \partial z^r} \right), \tag{2.136}$$

with $CK_k = CK_k(\underline{\underline{A}}, \underline{\underline{B}}, \underline{\underline{C}}, \frac{\partial^{p+q+r} \vec{U}}{\partial x^p \partial y^q \partial z^r})$ for the linearized Euler equations.
The improvement in efficiency is demonstrated on a 2D example that is very similar to the planar wave test cases in the convergence study section 2.7.1.1. Here, two linearized eighth order domains – a FV domain Ω_1 with $[80 \times 80]$ cells and a FD domain Ω_2 with $[37 \times 37]$ domain points – are connected. The time step ratio for subcycling is $\frac{\Delta t_{\Omega_1}}{\Delta t_{\Omega_2}} = 2$. A planar, right-moving characteristic wave is initialized. All domain boundaries are coupling boundaries or periodic coupling boundaries. This means that not only Ω_1 has a common boundary with Ω_2 but also every outer boundary is connected periodically with the opposite side by coupling. Thus, both Ω_1 and Ω_2 are also linked to themselves, which results in a quite large number of coupling ghost cells. By doing so, the coupling overhead becomes large compared to the time which is spent in the actual solvers. Note that in later practical examples, the overhead is *small* against the solver CPU time. The simulation is run with the same parameters as in section 2.7.1.1. Table 2.10 shows the statistics for the calculation with and without pre-integration prior to the actual iteration loop. Without pre-integration, each ghost cell of the $\mathcal{O}8$ FV domain contains 16 Gauss points. With pre-integration on the other hand, the number of interpolations per ghost cell is reduced to a single one. Hence, the interpolation effort for the FV cells

reduces to $\frac{1}{16}$th, resulting in an reduced computational effort of $\frac{1}{14}$th compared to the calculation without pre-integration. Note that a base overhead is always present, for example for the CK procedure and for the exchange of the FD ghost points.

Without pre-int.	# Coupling elem.	# Interpol. targets	$t_{CPU}[s]$
Ω_1	1344	**21504**	57.5
Ω_2	656	656	3.5
Total	2000	22160	7866.8
Coupling			**7805.9**
With pre-int.	# Coupling elem.	# Interpol. targets	$t_{CPU}[s]$
Ω_1	1344	**1344**	18.9
Ω_2	656	656	3.6
Total	2000	2000	566.0
Coupling			**543.5**

Table 2.10: Pre-integration statistics for the FV-FD planar wave case.

2.4 The Coupling of Different Time Steps

The explicit time integration methods which are used in the calculation domains must follow the CFL stability condition

$$\Delta t \sim \frac{CFL \cdot \Delta x}{a} \tag{2.137}$$

for dominating convective terms. If the diffusive character prevails for the Navier-Stokes equations, the maximum time step of an iteration is limited by

$$\Delta t \sim \frac{DFL \cdot \Delta x^2}{\nu}, \tag{2.138}$$

where Δx is the characteristic element size, a is the maximum convection speed in the cell and ν is the kinematic viscosity.
It is obvious from (2.137) and (2.138), that domains with very different grid spacing will also have very different maximum time steps. Usually, the domains with larger time steps would have to adopt the smallest Δt for each iteration. However, according the domain decomposition philosophy, it would be very convenient to allow the largest possible time steps on each domain. In order to implement this form of "local time stepping", a subcycling method is introduced.
The idea of multi-size meshes with different time steps has been examined by Tam et al. [106] for DRP schemes on particularly designed grids. This basic idea has been extended to arbitrary time step ratios by Schwartzkopff [88] for his coupling approach and has now been implemented for arbitrary meshes in the generalized domain decomposition framework for FV, FD and DG schemes. The data between the domains are exchanged at common time levels, as described in section 2.3. After the exchange, the largest possible time steps are estimated in each domain. Then the Δt's are sorted with respect to their size and adjusted such that they are always equal or a multiple of the previous smaller one. The Δt's of the domains are also adjusted if a data output is imminent in order to match the desired output time. Note that time steps may be only *decreased*, as the stability condition must not be violated. The idea of subcycling is illustrated in Fig. 2.19: Four domains Ω_0, Ω_1, Ω_2 and Ω_3 are coupled in a calculation. All four domains possess different time steps ($\Delta t_{\Omega_0} < \Delta t_{\Omega_1} < \Delta t_{\Omega_2} < \Delta t_{\Omega_3}$) which have already been adjusted for common time levels. Depending on how many of the domains have reached the same absolute time level, subcycle 1, 2 or 3 is complete. For example after subcycle 1, domain Ω_0 has made five time steps, shares now a common time level with

Ω_1 and the domains can exchange data at their boundary $\Gamma_{\Omega_0,\Omega_1}$. When subcycle 3 is complete, all of the domains have arrived at the same level.
The updating procedure itself is illustrated in Fig. 2.20 for the exchange between two domains Ω_1 and Ω_2. The Cauchy-Kovalevskaja procedure is used to provide time-accurate values for the ghost elements which take part in iterations *between* data exchanges. The starting point is a Taylor series in time

$$\vec{U}(\vec{X}, t^n + \tau) = \vec{U}(\vec{X}, t^n) + \sum_{k=1}^{n_{Poly}} \frac{\tau^k}{k!} \frac{\partial \vec{U}^k(\vec{X}, t^n)}{\partial^k t}, \tag{2.139}$$

with which, once the time derivatives at the target point $\vec{X}$ are known, the state at an arbitrary time $t^c = t^n + \tau$ can be calculated easily. The unknown time derivatives can be replaced by spatial derivatives with the CK procedure (Kowalevsky [120]). This procedure is used in Lax-Wendroff type time integration methods (Lax and Wendroff [62], Hirsch [45]) and is also a main ingredient of the ADER schemes. The CK procedure makes use the original PDE and can be written as a function of the spatial derivatives:

$$\frac{\partial^k \vec{U}(\vec{X}, t^n)}{\partial t^k} = CK_k \left(\frac{\partial^{p+q+r} \vec{U}_T(\vec{X}, t^n)}{\partial x^p \partial y^q \partial z^r} \right), \ \forall \ 0 \leq p+q+r \leq k. \tag{2.140}$$

To give a very simple example, the CK procedure for the kth derivative for linear systems of equations in 3D reads

$$\frac{\partial^k \vec{U}(\vec{X}, t^n)}{\partial t^k} = (-1)^k \left(\underline{\underline{A}} \frac{1}{\partial x} + \underline{\underline{B}} \frac{1}{\partial y} + \underline{\underline{C}} \frac{1}{\partial z} \right)^k \partial^k \vec{U}(\vec{X}, t^n), \tag{2.141}$$

where the differential operators act on $\vec{U}(\vec{X}, t^n)$. Efficient algorithms for a variety of linear and nonlinear systems of equations (e.g., LEE, Euler, Navier-Stokes) have been developed [18–20, 23, 24, 26, 88, 109, 115, 116] and are available in the domain decomposition framework.
As soon as the time derivatives have been determined, they can be stored as coefficients for the Taylor series (2.139) for efficiency purposes: Especially in the nonlinear case, the CK procedure becomes computationally expensive if it is repeated for every iteration. The coefficients are valid until the next data exchange, where another CK procedure determines a new set of time derivatives at the respected target point. It shall be emphasized, that the provided Taylor data for the ghost elements result from the interpolation from the domain with the larger time step. As a consequence, these ghost elements inherit the stability condition from the neighbor grid and every evaluation of the Taylor series between two data exchanges is in the stable range.

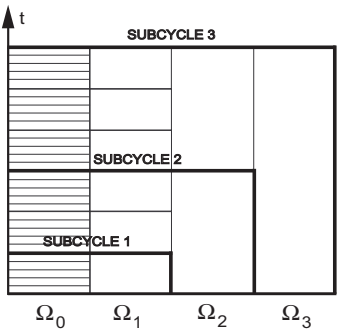


Figure 2.19: Subcycles for a calculation with four differently sized domains.

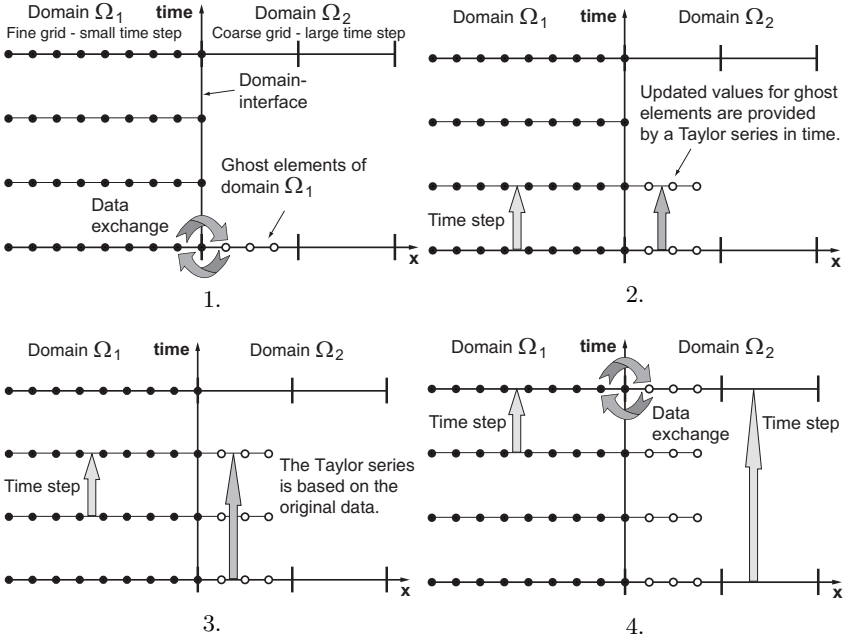


Figure 2.20: The subcycling procedure.

2.5 The Implementation of the KOP3D Framework

The tool *KOP3D* has been developed as a coupling framework for stand-alone research codes. The framework and the solvers are written in Fortran 95 programming language. Figure 2.21 depicts the structure of the programs: The stand-alone CFD and acoustics solvers for either unstructured or structured meshes consist of single modules, which can be fully embedded in the decomposition framework. There are no separate codes anymore in the compiled version of *KOP3D*, only one executable exists. Note that the latter is not the case for the coupling with external codes that do not have the same modular structure (see section 2.6.1), where two programs need to be executed. The

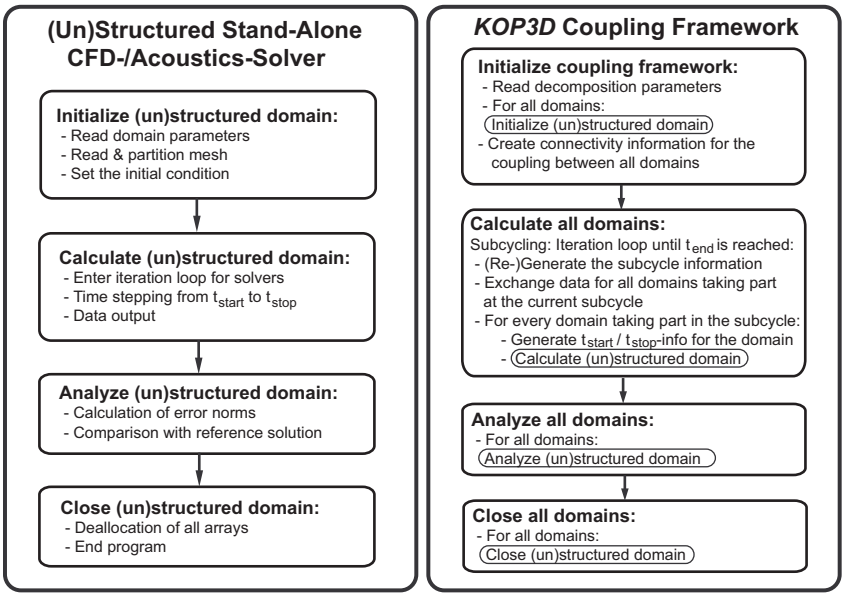


Figure 2.21: Program structures: The building blocks of the stand-alone CFD & CAA solvers are part of the domain decomposition framework *KOP3D*.

decomposition of the total domain into subdomains is handled manually and not by a grid assembly software. Every subdomain is operated by exactly one solver, for which a separate parameter file and a mesh file must be provided. A main parameter file contains the paths to the subdomain definitions and specific information regarding the coupling procedure. The MPI parallelization *inside* the subdomains is organized completely code-internal. The simplified structure of the framework and the algorithms are shown for clarity in the serial version of the code. The MPI-parallel implementation uses optimized loops and a more efficient data structure (target point oriented arrays for better vectorization, etc.). More details about the MPI communication in *KOP3D* can be found in Klimach et al. [56].
The following algorithms are elemental features of the coupling framework:

Algorithm 2.5.1. ***Create connectivity***

1. *Estimate the time steps for each domain: Coupling ghost elements which require derivatives for the CK procedure (section 2.3.3) are marked.*
2. *Generate the boundary polygon information for all domains.*
3. *Create the coupling information for all domains:*
   a) *Construct ghost elements, if necessary (e.g., DG methods), and allocate the data structure.*
   b) *Calculate the number and position of the target points for the interpolation for all coupling ghost elements (section 2.3.2).*
   c) *For all target points in each ghost element:*
      i. *Search for the source domain for the interpolation by checking the boundary polygons.*
      ii. *Get the interpolation stencil information from the source domain (stencil element indices, number of elements, interpolation weights, etc.) for state and derivatives at the target point position (section 2.3.3, 2.3.4, 2.3.5).*
4. *Pre-integrate coupling ghost elements if possible, in order to reduce the amount of data (section 2.3.7).*

Algorithm 2.5.2. ***Exchange data***

1. *Set all non-coupling boundary conditions (e.g., inflow, outflow, wall) for all structured domains. These data may be needed for the interpolation.*
2. *For all structured domains: Get values from the source domains for all target points marked for "asymmetric", "CK-supported symmetric" or "symmetric without ghost elements" interpolation (section 2.3.5).*
3. *For all structured domains: Set the boundary conditions for all coupling ghost elements which already possess the complete set of information (section 2.3.6). These cells may be required for the further data exchange.*
4. *For all structured domains: Get values from the source domains for all target points marked for "symmetric with ghost elements" interpolation (section 2.3.5).*
5. *For all unstructured domains: Get values from the source domains for all target points.*

Algorithm 2.5.3. ***Get values from source domains***
Loop over the target point's stencil elements in the source domain:

1. *If the stencil element is a structured ghost element and is not set yet: Provide temporary data with a Taylor series in time (section 2.3.5).*
2. *If the source element is a DG cell: Evaluate the DG polynomial at the target position.*
3. *Perform a state vector conversion from the source system into the target system, if necessary (section 2.2.4).*
4. *Multiply the stencil element's value with its weight and add it to the result calculated previously in the loop. After the loop over the stencil elements is complete, the target point is set (equations* (2.105)*,* (2.115) *and* (2.117) *in sections 2.3.3 and 2.3.4).*

Besides those core elements of the coupling framework, a large variety of modifications and features associated with the domain decomposition has been implemented in the different solvers. For example the setting of the coupling ghost elements (see section 2.3.6) is directly integrated into the solver's boundary condition treatments. Another example is the ability to cut holes into structured domains. With this feature, it is not necessary to arrange several

subdomains around a center domain (see section 3 for examples), which can be cumbersome. The hole cutting is realized by either ignoring the cells inside the hole in the solver algorithms or by simply overriding the ghost cells at the hole boundary.

2.6 The Coupling of Different System Architectures

This section explores an entirely different domain decomposition approach. After having adapted the numerical instruments locally for acceleration purposes, the computational domain is now distributed to different computer systems.
The numerics in solver routines can differ considerably, depending on the discretization and the data structure. As a result, different solvers show a different behavior with respect to vectorization. Usually, codes for Cartesian grids can be optimized in order to achieve a high computational performance on vector machines. On the other hand, very local solvers, such as the FV or DG schemes on unstructured grids, show their best parallel performance on scalar machines. Due to that, a machine independent coupling mechanism has been implemented which enables the distribution of different solvers to separate computer systems.
Two possible ways are examined in the following: At first, an external stand-alone code for Direct Numerical Simulations (DNS) is linked to the *KOP3D* framework by using the TCP protocol. Both programs remain stand-alone and can be run on different computer systems at the same time. This method has the advantage that a specialized code with completely different data structure and programming philosophy can be included in the existing solver framework. The second approach distributes the *internal* solvers of the *KOP3D* framework to different machines. Here, the code framework is most coherent and the validated coupling algorithms can be employed without modifications.

2.6.1 Connection to External Codes

An unstructured DG code for acoustics from the *KOP3D* framework is coupled with a structured FD code for DNS. All features from the previous sections (the change of equations, grid types, methods, time steps, etc.) are exploited for this purpose. The required domain information of course needs to be communicated between the separate solvers.
On the FD side, the *NS3D* code (Babucke et al. [3]), based on the three-dimensional unsteady compressible Navier-Stokes equations is used. Spatial discretization in streamwise x- and normal y-direction is done by $\mathcal{O}6$ compact

finite differences. The flow is assumed to be periodic in spanwise direction, therefore a spectral ansatz is applied for the z-direction. High wave numbers, which are generated by the non-linear terms in the Navier-Stokes equations, are damped by applying alternating up- and downwind-biased finite differences for the convective terms (Kloker [58]). The second derivatives are evaluated directly, resolving the viscous terms more precisely (Babucke et al. [2]). Time integration is done using the standard $\mathcal{O}4$ Runge-Kutta scheme. Using MPI [74], the code is parallelized in a block structured manner in the xy-plane, while a shared memory parallelization is performed in spanwise direction. Good vectorization properties provide a fast and efficient usage of vector computers. More details on the *NS3D* code can be found in Babucke et al. [2, 3].
The three-dimensional acoustic code uses the ADER-DG method for the linearized Euler equations on unstructured tetrahedrons. The solver is a good testbed for the coupling procedure: It is most efficient on scalar parallel computers due to its locality and it is already embedded in the *KOP3D* framework. On the other side, the unstructured ADER-DG method is computationally more expensive than for example a structured FD method. Hence, it would not be the method of choice in a domain which is far away from complex geometries. However, this constraint will be accepted in this proof of concept, as the aim is to couple two codes with very distinct behavior. In the following setup, the DNS code runs on the NEC-SX8 vector machine of the High Performance Computing Center Stuttgart (HLRS), while the acoustic code is executed on its scalar front end machine TX7 with Itanium II (IA64) processors. Note that the coupling mechanism is not limited to these specific machines: Other computer configurations are possible and were successfully tested.
The communication between the platforms is based on the Transmission Control Protocol (TCP [78]), consisting of a server and a client. The server opens a port and waits for the client to connect. To preclude the server from being started before the client, the server is not run inside a queuing system. Here, the DG code acts as the server, running interactively before the FD code is submitted to the queue of the SX8 vector machine. The IP address and the port of the server must be specified for the client. The communication itself is done by simply writing on or reading from the socket.
The initial communication defines the array sizes, the DNS code sends the base flow to the acoustic code and the time step ratio is determined. For the interpolation, the positions of the ghost points and a sufficiently large strip from the actual domain must be communicated to the acoustic code. The latter organizes the entire coupling procedure, *including* the interpolation from FD to DG and the determination of the subcycles. It is assumed, that the time step

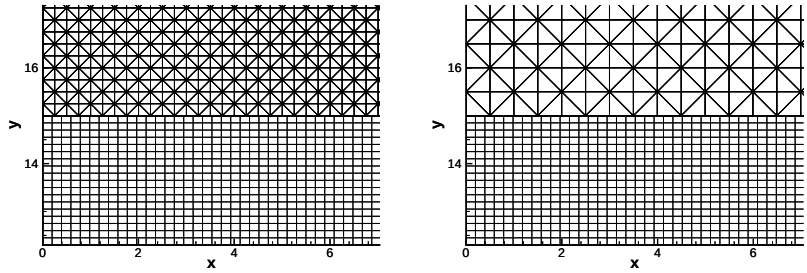


Figure 2.22: Detailed view of the FD (bottom) and the DG grid (top). Left: Fine DG grid. Right: Coarse DG grid.

of the acoustic domain is equal or greater than the one of the DNS domain. Consequently, the DG domain also calculates and provides boundary data with the CK procedure for the Runge-Kutta stages of the structured code. It seems reasonable to let the acoustic code handle the coupling overhead because in a real life scenario, the DNS is considered to be the most expensive part. At every common time step, the set of primitive variables (ρ, u, v, w, p) near the boundary is exchanged. The communication is finished by closing the socket.
A big advantage of using the network connection instead of standard MPI lies in the fact, that two separate executables remain. Thus, the two codes do not need to be merged. Each code is independent of the internal parallelization concept of the other code, since only the first MPI process communicates with the connected program. The data distribution to the other internal processes is handled by each program separately.
The external coupling method is validated with the example of a wave package that is transported from an interior DNS domain to an outer unstructured acoustic domain. A uniform mean flow is chosen with $Ma_\infty = 0.5$ in streamwise direction, resulting in an ambient pressure of $p_\infty = 1/(\gamma \cdot Ma_\infty^2) = 2.85714$. The temperature is $T_\infty = 280K$, containing a temperature disturbance of $0.001 \cdot T_\infty$ with a radius of 1.0 located at $x = 39$ and $y = 0$ in the DNS domain. This leads to a circular pressure pulse (which is emitted in all directions) and to a temperature spot (which is convected with the streamwise mean flow). The Reynolds number $Re = 500$ and the Prandtl number $Pr = 0.71$, which describe

the viscous terms, are of course only meaningful for the DNS domain.
The DNS grid extents are $-15 \leq y \leq 15$ in normal and $0 \leq x \leq 115$ in streamwise direction. In order to prevent disturbances that are convected with the mean flow from reaching the outflow boundary, the grid is stretched in x-direction from $x = 60$ onwards. The spanwise extent is $z = [0, 0.5]$ for the fine and $z = [0, 1]$ for the coarse grid, respectively. The DG domains are located above and below the FD grid with the same dimensions in streamwise and spanwise directions and a range of $y = \pm 25$. The wave propagation is examined by comparing discretization orders on different grids, the latter are depicted in Fig. 2.22.
Table 2.11 shows the different test configurations. The time step ratio is set to one for the considered cases. However, the Runge-Kutta method requires values at intermediate time levels, which makes the CK procedure still necessary despite the common time step. The time step limitations for the ADER-DG scheme is a minor problem for "real world" computations where the spatial resolution of the DNS is much higher than in the DG domain, where only long wave acoustics need to be resolved.
Case D is a large DNS which is performed for comparison purposes. The domain has three times the size of the original DNS, with $y = \pm 45$. Hence, the wave does not reach the boundary in the considered time. As we can exclude errors due to boundary conditions, this DNS solution may serve as reference data. The resulting pressure distribution in the xy-plane at time $t = 9.4248$ is shown for the single domain case in Fig. 2.23. The wave crossing the coupling interface is shown for case C in Fig. 2.24, using the polynomial information inside the DG elements in order to plot the solution. Only the integral mean-values are considered for visualization in Figure 2.25, depicting also the grid configuration. Despite the different equations and discretization used in both codes, the wave crosses the coupling plane with almost no reflections. The

Case	DNS grid (x y z)	$\#Elem_{,DG}$	$\mathcal{O}_{,DG}$
A	401 x 201 x 5	368800	2
B	401 x 201 x 5	368800	3
C	401 x 201 x 5	92400	4
D	401 x 603 x 5		

Table 2.11: Domain configurations for the external coupling.

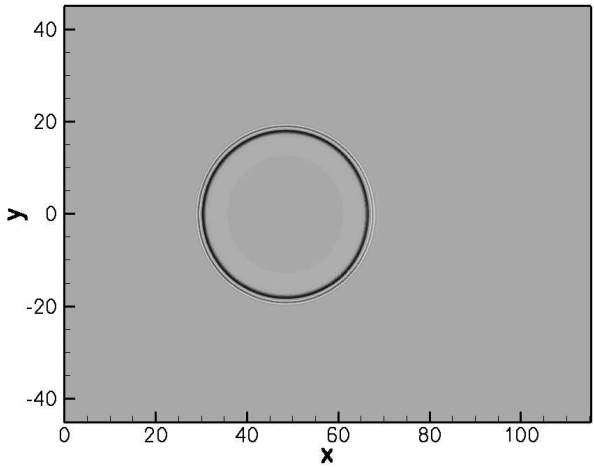


Figure 2.23: Instantaneous pressure field at time $t = 9.4248$, obtained by the DNS without coupling.

pressure distribution along a constant x-position at two different points of time is shown in Figs. 2.26 and 2.28. The detail views in Fig. 2.27 and 2.29 illustrate the differences between various DG orders and the single domain DNS solution. At time $t = 7.8539$, when the pulse passes the coupling interface, a small amplitude error can be observed. This error may correspond to minor reflections at the interface. Later in time, the amplitudes are about equal for both the coupled and the single domain calculation (Fig. 2.29). The phase error decreases with higher DG order. It must be taken into account that the Navier-Stokes equations contain viscous terms, while in the outer domain the linear Euler equations are solved. This might explain the better accordance of the amplitudes at the later time step (Fig. 2.29).
As mentioned above, the machines used for this test are a NEC-SX8 for the FD code and an Itanium II machine for the DG code. The computational performance for the considered cases is summarized in Table 2.12. The total network communication per time step between the SX8 and the Itanium II machine takes less than one second and transfers about five megabyte. The CPU time of the FD code varies slightly between approximately 500 to 600 seconds. The CPU time of the DG method grows with the order of the scheme and ranges

from 6.5 to 53 hours for 800 time steps. This underlines especially the superior performance of the structured code on the vector computer. The DG code performs slower on the scalar cluster, but this is an expected behavior: First of all, the large number of degrees of freedom in the DG domain indicates, that the acoustic grid has been chosen quite over-resolved. Furthermore, the front end computer is not equipped with especially fast CPUs and was heavily used by other applications. It has been observed, that the DG code runs much more efficient on other parallel clusters and does not reach its optimal performance on this specific machine. Nevertheless, the machine was chosen for testing in order to circumvent other technical restrictions, such as firewall and network speed issues. Last but not least, while the unstructured method is highly suitable in the vicinity of complex geometries, a much cheaper structured scheme (e.g., ADER-FD) can be used for far field computations. This code could be also run on a vector computer with a better efficiency than the DNS code (due to a coarser grid, LEE), the same degree of optimization provided.

Case	A	B	C	D
$\mathcal{O}_{,DG}$	2	3	4	-
t_{real} [s]	24079	60503	73869	782
$t_{CPU,DG}$ [s]	23560	59230	72869	-
$t_{CPU,FD}$ [s]	576	571	507	2031
$\#CPU_{,DG}$	1	1	1	-
$\#CPU_{,FD}$	1	1	1	3
Δt	3.93E-03	3.93E-03	3.93E-03	1.57E-02
$\#Iter$	800	800	800	800
$\#Elem_{,FD}$	403005	403005	403005	1209015
$\#Elem_{,DG}$	368800	368800	92400	-
$\#DOF_{,DG}$	1475200	3688000	1848000	-
Com.Data [MB]	3.99	3.99	4.82	-

Table 2.12: Domain statistics for the external coupling.

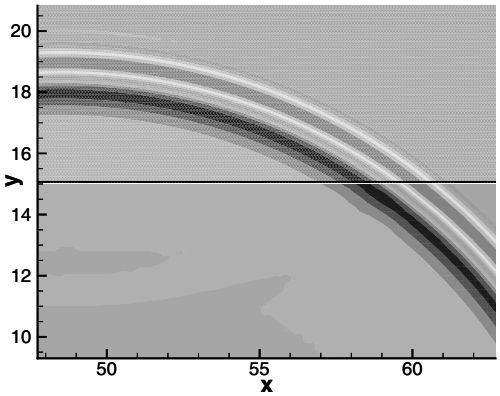


Figure 2.24: Acoustic wave crossing the coupling plane at $t = 9.4248$ for the $\mathcal{O}4$ DG method on the coarse grid.

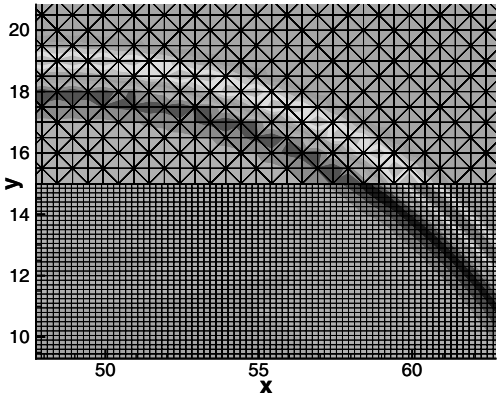


Figure 2.25: FD and DG grids with integral mean-values of the DG elements at $t = 9.4248$ for the $\mathcal{O}4$ DG method on the coarse grid.

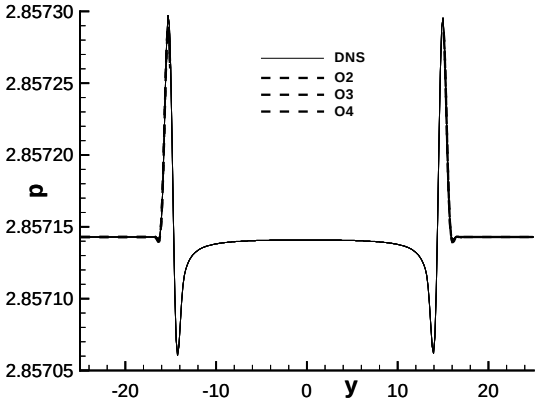


Figure 2.26: Pressure distribution at $x = 52.03$ and $t = 7.8539$.

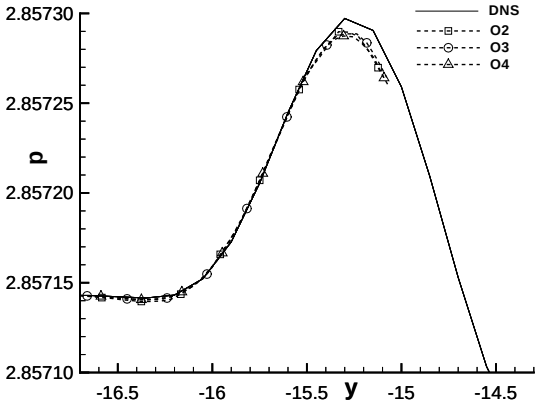


Figure 2.27: Close-up of the pressure distribution at $x = 52.03$ and $t = 7.8539$.

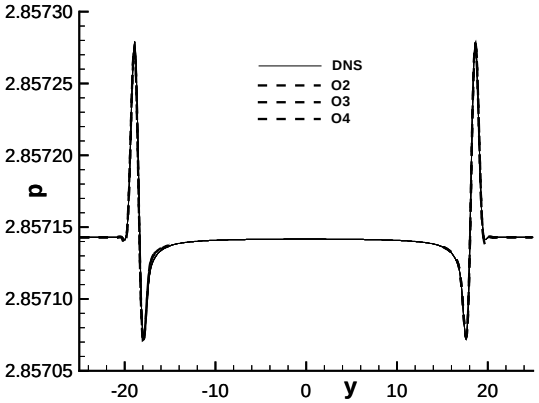


Figure 2.28: Pressure distribution at $x = 52.03$ and $t = 9.4248$.

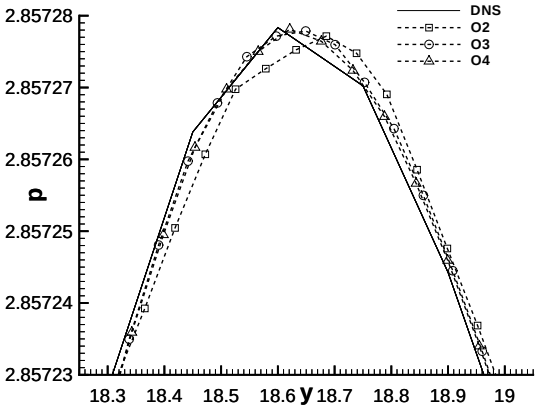


Figure 2.29: Close-up of the pressure distribution at $x = 52.03$ and $t = 9.4248$.

2.6.2 Code-Internal Distribution onto Different Systems

The distribution of internal *KOP3D* solvers to different computer architectures is organized with the PACX-MPI library (Beisel et al. [6], Keller et al. [53]). The handling of this environment resembles very much using standard MPI with the additional capability of inter-cluster communication. Solver internally, the regular native MPI libraries are used. It is emphasized, that with this kind of coupling only one *KOP3D* executable exists, which has access to different types of processors, regardless of their physical position. More details about the integration of PACX-MPI and the communication layout inside the *KOP3D* framework can be found in Klimach et al. [55, 56], where also the following example is taken from.
The setup of the test case is largely identical to the sphere scattering in the example section 3.3. Some modifications are made, though: In the unstructured center domain around the sphere (9874 tetrahedrons), the order of the ADER-DG method is increased to $\mathcal{O}8$. The structured $\mathcal{O}8$ ADER-FD far field domain is enlarged to the extents $102.2 \times 57 \times 57$ and contains now about 42 million grid points. The main solver routine of the ADER-FD method shows a good vectorization ratio on the NEC-SX8 machine, where it operates at ca. 14 GFlop/s (87.5% of the peak performance). Compared to that, the unstructured ADER-DG scheme performs poorly with less than 100 MFlop/s on the vector machine. In order to examine the distributed simulation, three calculations are performed: At first, the total domain is computed on one single Itanium II (IA64) processor on the scalar front end cluster. In the second calculation, the simulation is run on a single NEC-SX8 processor. Last but not least, the unstructured domain is distributed to one single Itanium II processor while the structured domain is run on a single NEC-SX8 CPU. Table 2.13 shows the simulation times. It is obvious, that the unstructured code becomes very slow on the vector processor and performs much better on the scalar machine. For the structured part, the situation is vice versa. It is so efficient, that the acceleration in the structured part outweighs the slower unstructured part in the SX8 single processor calculation and reduces the elapsed time to 39% of the IA64 single processor run. Finally, the distributed simulation manages to combine the best performances from the single processors and requires the least computational effort. The PACX-MPI run takes only about 58% of the wall-clock time of a single SX8 calculation with two CPUs, linear scaling provided. Now the distributed simulation can be further accelerated by parallelization in each subdomain.

	1×IA64	1×NEC-SX8	1×IA64 + 1×NEC-SX8
Unstr. Domain [s]	2994	7746	3019
Str. Domain [s]	23887	2871	2869
Coupling [s]	1012	321	554
Waiting in MPI [s]	0	0	164
Elapsed time [s]	**27925**	**10966**	**3207**

Table 2.13: Domain statistics for the internal coupling.

2.7 Validation

2.7.1 Global Convergence

In the following, the behavior towards convergence for coupling procedure is tested in 2D and 3D. The convergence studies demonstrate, that high order of accuracy can be maintained globally for the coupling procedure. Schwartzkopff [88] achieved global convergence for his coupling procedure. The linearized Euler equations were solved with the ADER-FV method on structured grids for the advection of a Gaussian pulse in density. Dumbser [18] observed the same behavior for the coupling of ADER-DG domains with other ADER-DG or -FV domains, again in 2D. Both Schwartzkopff and Dumbser underlined the importance of the CK procedure for the coupling elements: If the CK procedure was turned off, only first order of accuracy could be obtained.
Now the convergence properties of the Gauss point coupling approach is examined in several test cases: For the connection of linearized domains, a planar wave in the characteristic variables is chosen in order to test the complete set of equations. For nonlinear domains, the so-called Shu vortex is used. Finally, a Gaussian pulse in density is used for testing the coupling of nonlinear domains with linear ones. Mixed grids (unstructured/structured), methods (DG/FV/FD) and equations (linearized, nonlinear) are preferred for the tests, as they are considered more challenging. These test cases are regarded as a representative selection of all possible configurations.

2.7.1.1 Planar Wave

For linearized domains, a planar wave is initialized such that it contains only fluctuations in the right-moving characteristic wave with the eigenvalue $u + c$:

$$\vec{w}' = \vec{\tilde{w}} \cdot \sin(\vec{k} \cdot \vec{X}), \tag{2.142}$$

with $\vec{w}' = (w_1', w_2', w_3', w_4', w_5')$ being the perturbation of the characteristic variable vector, $\vec{\tilde{w}} = (0, 0, 0, 0.001, 0)$ containing the amplitude of the perturbation and $\vec{k} = (2\pi, 0, 0)$ being the wave number vector. The primitive fluctuations are obtained by a transformation with the Eigenvector matrix R, which belongs to the set of Eigenvectors $\vec{\lambda} = (u, u, u, u + c, u - c)$:

$$\vec{u}' = \underline{\underline{R}}\vec{w}', \tag{2.143}$$

$$\underline{\underline{R}} = \begin{pmatrix} n_1 & n_2 & n_3 & \frac{\rho_0}{2c_0} & \frac{\rho_0}{2c_0} \\ 0. & -n_3 & n_2 & \frac{n_1}{2} & -\frac{n_1}{2} \\ n_3 & 0. & -n_1 & \frac{n_2}{2} & -\frac{n_2}{2} \\ -n_2 & n_1 & 0. & \frac{n_3}{2} & -\frac{n_3}{2} \\ 0. & 0. & 0. & \frac{\rho_0}{2c_0}, & \frac{\rho_0}{2c_0} \end{pmatrix} \tag{2.144}$$

with $\vec{n} = (n_1, n_2, n_2) = (1, 0, 0)$ being the normal vector of the wave. The background flow is $\rho_0 = 1.0$, $u_0 = 0.0$, $v_0 = 0.0$, $w_0 = 0.0$ and $p_0 = 0.714285714$. The result is a wave in the primitive variables $\vec{u}' = (\rho', u', v', w', p')$, which travels with the speed of sound $c_0 = 1.0$ in the direction of the positive x-Axis and has a wavelength of $\lambda = 1$.
In the first convergence study, an unstructured ADER-DG domain Ω_1 is coupled with a structured ADER-FV domain Ω_2, which is again connected to a Cartesian ADER-FD domain Ω_3. Each domain uses a fourth order method. All meshes are of length $l = 1$, thus one wavelength fits exactly in each domain (Fig 2.30). At the left and right boundaries, periodic boundary conditions are imposed by a virtual displacement of the ghost cells. Hence, there is also a coupling between Ω_1 and Ω_3. Note that the meshes are non-matching at their coupling interfaces. Because the methods and the grid spacing change between the domains, there are also considerable jumps in the time steps, for example a jump by a factor of 24 between Ω_1 and Ω_3. In order to determine the global order of convergence, the meshes are refined while keeping the ratios between the coupled domains constant. Table A.7 provides details regarding the mesh spacings $N_{\Delta h}$ in each domain and the subcycles. The calculation is run for 30 oscillation periods, so the initialized wave travels ten times across

the complete domain. At the end of the calculation, the error norms based on the perturbation pressure p' are calculated by comparing the solution to the initial condition. Table A.7 shows fourth order convergence rates on each single domain, hence the global order of accuracy is maintained despite the domain decomposition. The other relevant perturbation variables (ρ', u') yield similar results. The calculation is repeated without applying the CK procedure to the coupling ghost cells. As a result, the global order of accuracy breaks down to first order (Table A.8).
The coupling method is also capable of maintaining *very* high order between mixed grids. This is demonstrated in the next example, which is very similar to the previous one: Two domains (ADER-DG and ADER-FD) are connected, both employ eighth order methods. The methods are so highly accurate that the previous grids cannot be used for a convergence study: The error norms quickly decrease below the practical machine accuracy of the processor ($< 10^{-11}$). Therefore, the grids are made very coarse (Table A.9) and the wave number is doubled ($k = 4\pi$). Nevertheless, the zeroth refinement stage, which is depicted in Fig. 2.31, resolves the wave completely in both domains. Note that only a visualization which considers the degrees of freedom in the DG elements is able to show the actual solution. If only mean-values are plotted in the DG domain, the visualization delivers a false impression. On the coarsest structured grid, the initial resolution is 5 points per wavelength. The error norms in Table A.9 show eighth order convergence for both domains.
Last but not least, the global convergence for linearized domains is shown for a 3D example. The planar wave has again a wave number of $k = 2\pi$, fourth order methods are used (ADER-DG, ADER-FD, Table A.10). The regular unstructured tetrahedral meshes were created by subdividing hexahedra into five equal tetrahedra. The meshes do not conform at their connecting interface (Fig. 2.32). Nevertheless, fourth order convergence is achieved globally.

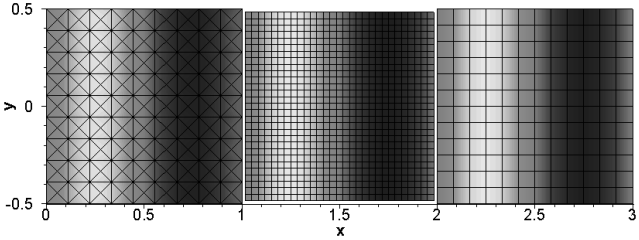


Figure 2.30: Contour plot of p' for the planar wave case, $\mathcal{O}4$, 2D, refinement stage #1.

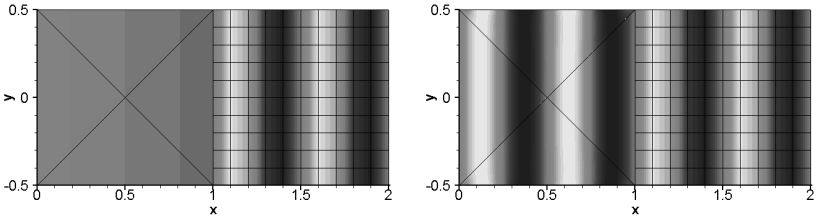


Figure 2.31: Contour plot of p' for the planar wave case, $\mathcal{O}8$, 2D, refinement stage #0. *Left:* mean-values are plotted in the DG domain. *Right:* Actual resolution.

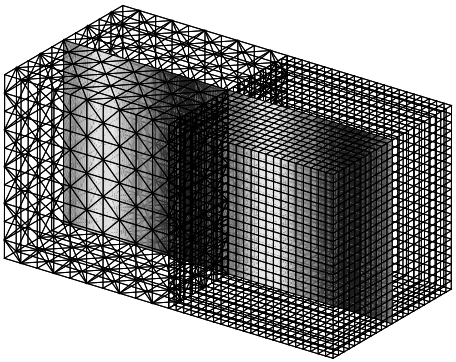


Figure 2.32: Contour plot of p' for the planar wave case, $\mathcal{O}4$, 3D, refinement stage #1.

2.7.1.2 Shu Vortex

The coupling of nonlinear domains is tested on the Shu vortex example (Hu and Shu [48]). The convection of a smooth, isentropic vortex in 2D is considered, the governing equations are the Euler equations. The initial condition is a linear superposition of a homogeneous background field $\rho_0 = 1.0$, $u_0 = 1.0$, $v_0 = 0.0$, $p_0 = 1.0$ and perturbations ρ', u', v', p'. The velocity perturbations u' and v' and the perturbations of entropy $S = \frac{p}{\rho^\gamma}$ and temperature T of the vortex are given by

$$\begin{pmatrix} u' \\ v' \end{pmatrix} = \frac{\epsilon}{2\pi} e^{\frac{1-r^2}{2}} \begin{pmatrix} -(y-y_0) \\ (x-x_0) \end{pmatrix}, \tag{2.145}$$

$$S' = 0, \tag{2.146}$$

$$T' = -\frac{(\gamma-1)\,\epsilon^2}{8\gamma\pi^2} e^{1-r^2}, \tag{2.147}$$

with $r^2 = (x-5)^2 + (y-5)^2$, the vortex strength $\epsilon = 5$ and $\gamma = 1.4$. A relationship between density, pressure and static temperature can be defined in a non-dimensional way, so that the gas constant becomes equal to unity:

$$\frac{p}{\rho} = T. \tag{2.148}$$

The density and pressure perturbations then become

$$\rho' = \left(1 + T'\right)^{\frac{1}{\gamma-1}} - 1, \tag{2.149}$$

$$p' = \left(1 + T'\right)^{\frac{\gamma}{\gamma-1}} - 1. \tag{2.150}$$

The vortex is initialized at $t = 0$ in an unstructured ADER-DG domain with its center at $(x_0, y_0) = (5, 0)$. It travels with speed $u_0 = 1.0$ in the direction of the positive x-axis and crosses the domain interface at $x = 10$, entering the Cartesian ADER-FV domain (Fig. 2.33). All boundaries are coupled periodically, therefore the vortex takes its initial position on the triangular mesh at $t = 20$, when the calculation stops and the error norms for the total energy ρE are calculated. Table A.11 shows the setup for the refinement stages for all grids and fourth order convergence rates for Ω_1, where the comparison with the initial condition is performed. Note that the initial ratio between the DG domain's time step and the time step on the FV domain is $\frac{1}{14}$, then reduces to $\frac{1}{8}$ when the vortex is located on the structured grid and finally increases again to $\frac{1}{14}$ when the vortex is back on the DG mesh. This is due to the fact that the maximum velocities in the calculation, which are attributed to the vortex, lead to a smaller Δt in the dynamic time step calculation.

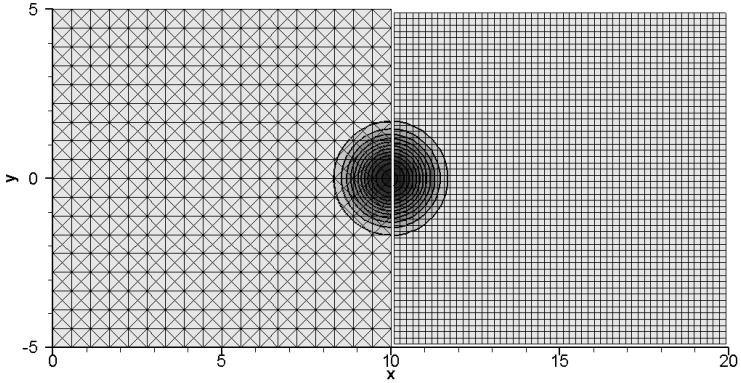


Figure 2.33: Contour plot of p for the Shu vortex case, refinement stage #4, at $t = 5.0$: The vortex is crossing the interface between the DG and the FV domain.

2.7.1.3 Gaussian Pulse

In case of decompositions with mixed equations (nonlinear-linear), the previous test cases are not suitable for a convergence study. Therefore, the advection of a Gaussian pulse in density is considered. In the nonlinear domain, a Gaussian fluctuation in the density variable is initialized and added to the otherwise constant flow:

$$\rho'(t = 0.0) = e^{-\frac{1}{2}\frac{(x-x_c)^2+(y-y_c)^2+(z-z_c)^2}{\sigma^2}}, \tag{2.151}$$

with the center coordinates $(x_c, y_c, z_c)_{\Omega_1} = (0.5, 0, 0)$, the pulse halfwidth $\sigma = 0.1$ and the amplitude $\hat{\rho}' = 0.01$. The mean flow is $\rho_0 = 1.0$, $u_0 = 1.0$, $v_0 = 0.0$, $w_0 = 0.0$ and $p_0 = 0.714285714$. In the linearized domain, these are also the values for the background flow. The other perturbation variables are set to zero. Very similar to the Shu vortex case, the Gaussian pulse is transported from the nonlinear unstructured grid onto the linearized structured domain and then back onto the unstructured one, where the simulation stops (Fig. 2.34 and 2.35). This time, the error norms are based on the density ρ. Using fourth order solvers, the convergence rates for the domain decomposition deliver the same order of accuracy in 2D and 3D (Tables A.12 and A.13). In the 3D case, the nonlinear Rec-FV scheme is used on the unstructured grid.

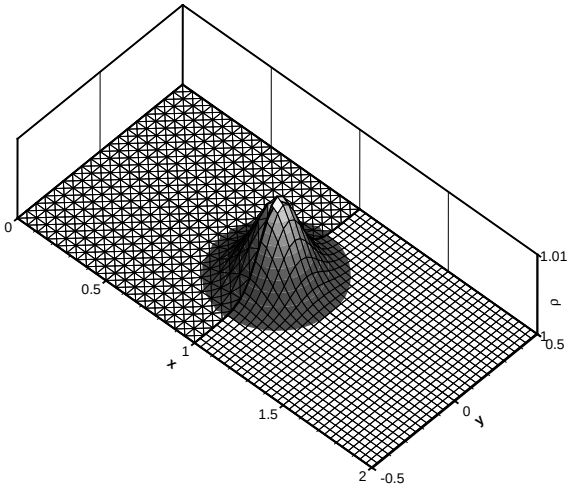


Figure 2.34: Contour plot of ρ for the Gaussian pulse case, $\mathcal{O}4$, 2D, refinement stage #4, at $t = 0.5$: The pulse is crossing the interface between the DG and the FV domain.

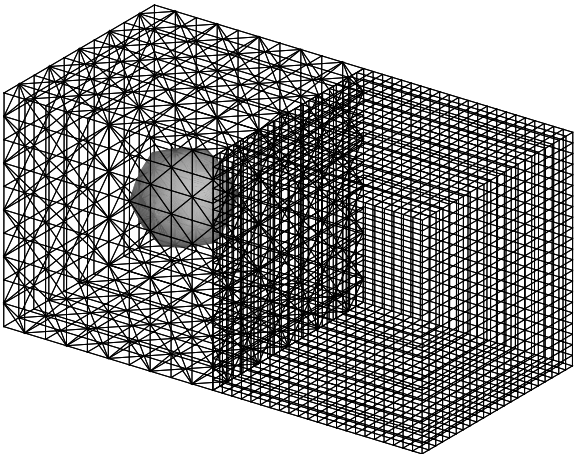


Figure 2.35: Isosurface plot of ρ for the Gaussian pulse case, $\mathcal{O}4$, 3D, refinement stage #1, at $t = 0.0$.

2.7.2 High-Frequency Perturbations

When grids with different mesh spacings are connected, it might occur that a flow feature or an acoustic wave is well resolved on one grid, while it cannot be maintained anymore on the other. An interesting question is now how the solution behaves in a case like this. Schwartzkopff [88] set up a quasi one-dimensional test for his coupling method: He let a single long wave pulse with superimposed high-frequency perturbations cross the interface from a fine grid to a coarse one. While the long wave pulse could be resolved on the coarse grid, the high-frequency perturbations were simply "filtered out" at the coupling boundary. A similar, but more detailed examination is performed for the proposed coupling approach.
The setup for this two-dimensional numerical experiment is as follows: An unstructured ADER-DG domain is surrounded by a structured ADER-FD grid. The unstructured grid will remain fixed in all computations, while the structured mesh is either very fine (resolving all occurring waves) or very coarse (only partially resolving the waves). Figure 2.36 shows the grid layout, Table 2.14 lists the domain properties. The governing equations are the linearized Euler equations with the background values $\rho_0 = 1.0$, $u_0 = 0.0$, $v_0 = 0.0$, $p_0 = 0.714285714$ and the speed of sound $c = \sqrt{\gamma \cdot \frac{p_0}{\rho_0}} = 1.0$ with $\gamma = 1.4$.

An oscillating source S (for a similar implementation, see section 3.1) is located in the center of the unstructured domain:

$$S = A_1 \cdot \sin(\omega_1 t) + A_2 \cdot \sin(\omega_2 t) + A_3 \cdot \sin(\omega_3 t), \tag{2.152}$$

where $A_1 = 0.05$, $A_2 = 0.04$, $A_3 = 0.03$, $\omega_1 = 2\pi$, $\omega_2 = 5\pi$ and $\omega_3 = 13\pi$. Note that the source term is only evaluated in the single triangle which contains the center coordinate $(x, y)_{Source} = (0, 0)$. The resulting frequencies $f_i = \frac{\omega_i}{2\pi}$ and wave lengths $\lambda_i = \frac{c}{f_i}$ for the emitted acoustic waves are $f_1 = 1.0$, $f_2 = 2.5$, $f_3 = 6.5$, $\lambda_1 = 1.0$, $\lambda_2 = 0.4$ and $\lambda_3 = 0.153846$.
The spacing and order of accuracy on the unstructured grid are chosen such

Domain	Method	$\mathcal{O}$	Outer Dimensions	Δh
Unstr. 1	ADER-DG	4	$[-2.00, +2.00] \times [-2.00, +2.00]$	$\frac{1}{20}$
Str. 1	ADER-FD	8	$[-10.00, +10.00] \times [-10.00, +10.00]$	$\frac{1}{8}$ / $\frac{1}{65}$

Table 2.14: Domain properties for the high-frequency perturbation test case.

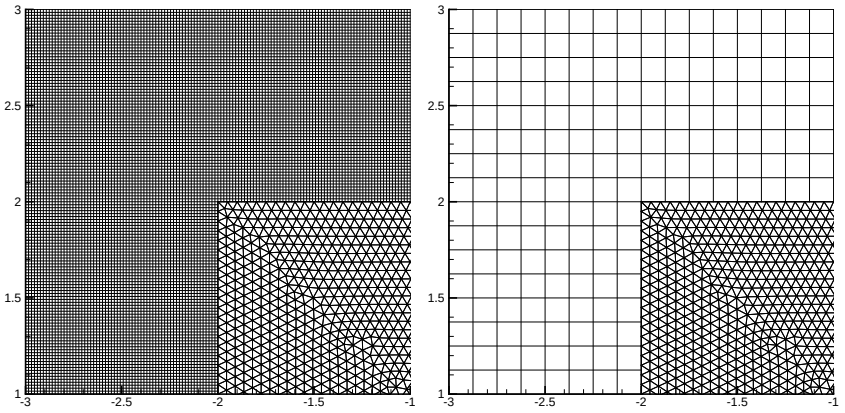


Figure 2.36: Close-up of the grids for the high-frequency perturbation test case. The unstructured mesh in the center remains the same, while the structured grid is either fine (*left*) or coarse (*right*).

that all frequencies are well resolved. For the smallest wavelength λ_3, the points per wavelengths are about $PPW_{DG,3} \approx \frac{\lambda_3 \cdot \mathcal{O}}{\Delta h} = 12.3$. Note that the order of accuracy is included in this rule of thumb for the resolution (a specialty of DG schemes).

For the first calculation, the spacing of the structured mesh is chosen so fine that – in combination with the high order scheme – all waves are fully resolved on the structured grid: $PPW_{FD,3} = \frac{\lambda_3}{\Delta h} = 10$. The result is shown in Fig. 2.37: The acoustic waves propagate in concentric circles. No reflections at the domain coupling interface are visible. At the outer boundary, a sponge layer (see section 3.3 for details) is employed, which slowly absorbs the waves in order to avoid reflections. A second calculation is performed with a much coarser structured grid that has a resolution of $PPW_{FD,1} = 8$, $PPW_{FD,2} = 3.2$ and $PPW_{FD,3} = 1.23$ for the wavelengths λ_1,λ_2 and λ_3. It is clear that the low frequency perturbations with λ_1 can be well resolved with 8 PPW. However, for medium wavelength λ_2, the 3.2 PPW are near the threshold of the theoretical limit of 2 PPW (wiggle mode) and is too low, even for a very high order method. Last but not least, the very short wavelength λ_3 cannot – even in theory – be resolved with 1.23 PPW. It shall be emphasized that those badly- and under-resolved frequencies are usually filtered out of the solution by an appropriate algorithm for today's state-of-the-art solvers (Kloker [57, 58]).

Figure 2.38, 1. shows the result of the calculation: Like in the previous case, all waves are fully resolved on the unstructured grid, but only the low-frequency waves make it onto the coarse grid and get propagated there. Exceptions seem to be the diagonals in the domains, where also the medium frequency f_2 is maintained quite well.
Last but not least, three independent calculations with only one frequency at a time are performed for comparison purposes. The structured mesh is again the coarse one. As expected, the low frequency f_1 is fully resolved and no reflections occur (Fig. 2.38, 2.). In the medium frequency case f_2, the waves vanish in direction of the x- and y-axis, but are still resolved in the diagonal direction (Fig. 2.38, 3.). This seems to be a unique property of the structured ADER-FV and ADER-FD methods, which employ volume stencils for the reconstruction. It is interesting in this context, that Schwartzkopff et al. [25, 88] observed enhanced CFL stability limits along the diagonals. It shall be pointed out that this is not the case for general FV and FD methods, for which the solution quality is usually worse in diagonal direction. The highest frequency f_3 vanishes almost completely on the structured grid(Fig. 2.38, 4.). Only a few perturbations remain, but turn out to be very small.
For a further examination, a DFT (Discrete Fourier Transform) is conducted for all test cases. The perturbation pressure signal $p'(t)$ is recorded and evaluated at several positions: $P_1 = (1.75, 0)$ and $P_2 = (1.2374, 1.2374)$ are located in the unstructured domain on a circle with radius $r = 1.75$, $P_3 = (5, 0)$ and $P_4 = (3.5355, 3.5355)$ are located in the structured domain on a circle with radius $r = 5$ (at an angle of $\Theta = 0^\circ$ and $\Theta = 45^\circ$).
Figure 2.39 shows the spectra of the pressure amplitudes $\hat{p}$: For the fully resolved calculation, only amplitudes at the three source frequencies are captured on both grids. For the partially unresolved case with the coarse structured grid, a few small additional peaks at some side frequencies are visible in the unstructured domain. On the structured grid, the low frequency peak at $f_1 = 1.0$ remains while the others vanish, as the higher frequencies are not resolved on the grid. An exception is the peak at $f_2 = 2.5$ at 45°, where the waves are partially resolved. Small additional peaks can be also observed here.
In order to find a relation between the source frequencies and the small side peaks, the Fourier spectra are compared to the spectra of the computations with the isolated source frequencies (Fig. 2.40 and 2.41). While the main peaks are identical in the unstructured center domain, the side peaks appear solely for the high angular frequency ω_3, which falls probably below the so-called aliasing limit for FD methods. Then the dispersion relation $k \propto \omega$ is not valid anymore, with k being the wave number and ω being the angular frequency of the wave.

In this case, the group velocity $v_G = \frac{d\omega}{dk}$ differs from the phase velocity $c = \frac{\omega}{k}$ and can even reverse its direction (Vogel [124], Kloker [57, 58]).
The side amplitudes in the structured domain appear almost exclusively (a tiny peak is visible also for the second harmonic frequency) for the calculations with the badly resolved wavelengths. They are very small in comparison to the main amplitudes, an exception is the f_2-peak on the diagonal. A further numerical experiment showed, that this peak soon vanishes, when the mesh is coarsened a bit more (at $PPW \approx 2.25$), due to under-resolution.
In conclusion, satisfying results are obtained, even without an artificial filtering of underresolved frequencies.

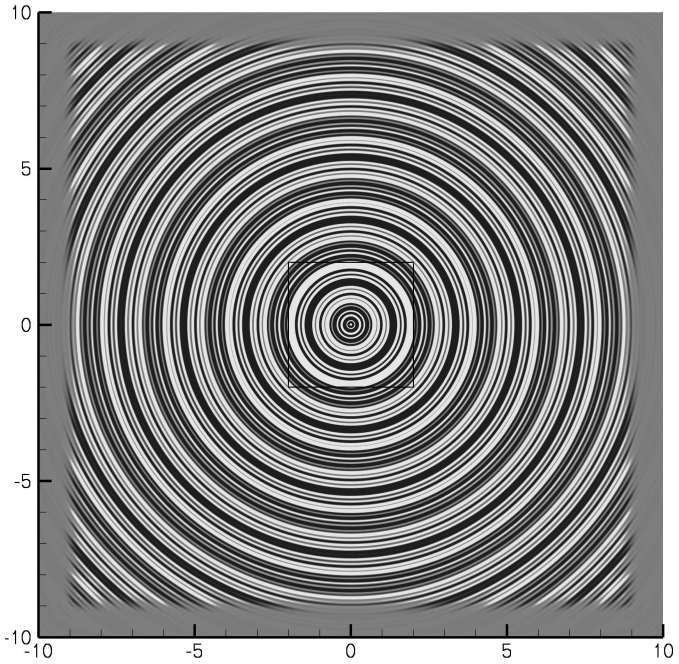
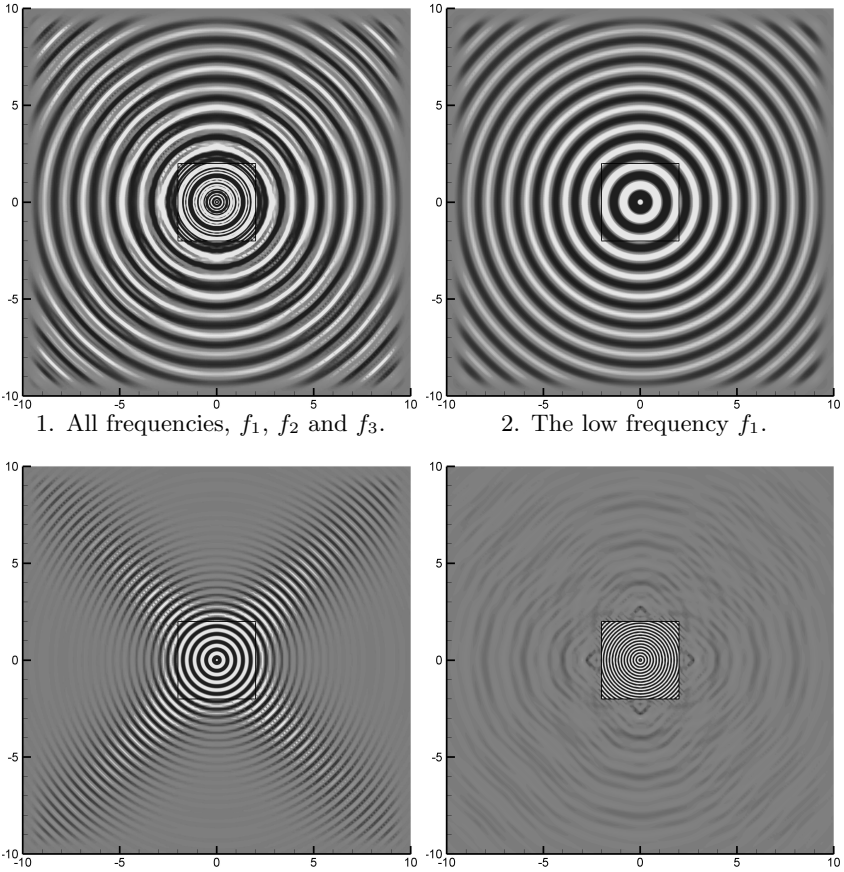
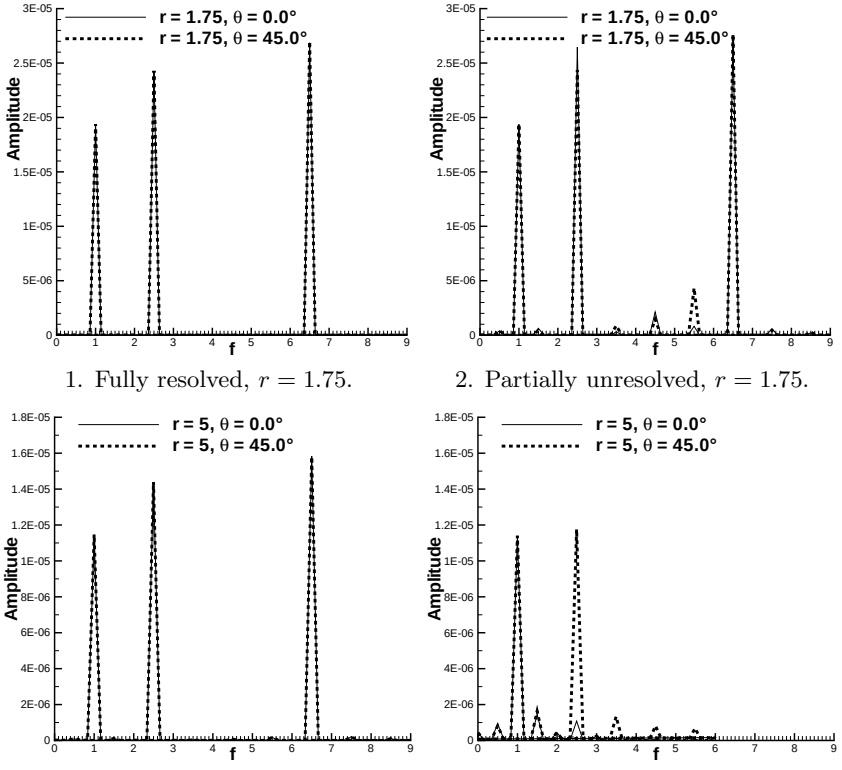


Figure 2.37: Contour plot of the fully resolved high-frequency perturbation test case at $t = 20$.



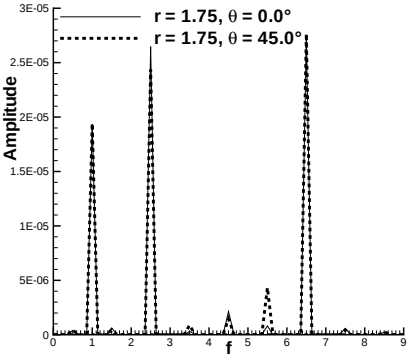
1. All frequencies, f_1, f_2 and f_3.
2. The low frequency f_1.
3. The medium frequency f_2.
4. The high frequency f_3.

Figure 2.38: Contour plots for the test case with the coarse structured grid at $t = 20$.

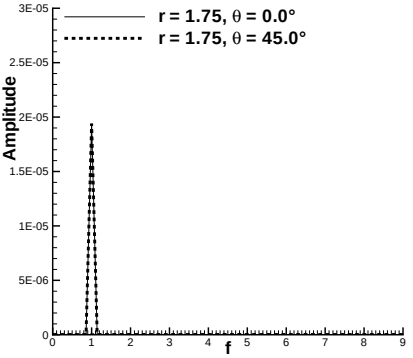


1. Fully resolved, $r = 1.75$.
2. Partially unresolved, $r = 1.75$.
3. Fully resolved, $r = 5$.
4. Partially unresolved, $r = 5$.

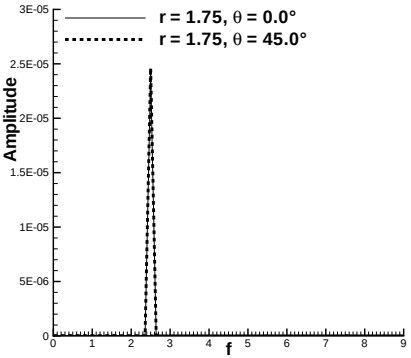
Figure 2.39: Fourier spectra of the pressure amplitude $\hat{p}$ at different positions in the computational domain. *Left column:* Using the fine structured grid. *Right column:* Using the coarse structured grid. *Top row:* $r = 1.75$ is located in the unstructured center domain, thus all frequencies are resolved. *Bottom row:* $r = 5$ is located in the structured domain, thus the frequencies are either fully (fine grid) or partially (coarse grid) resolved.



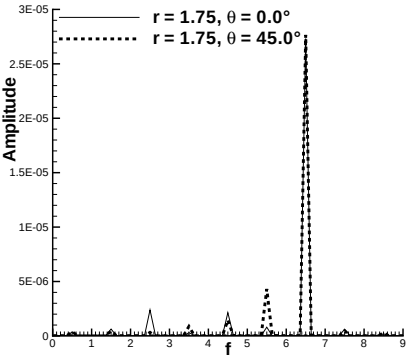
1. Source includes ω_1, ω_2 and ω_3.



2. Source only with ω_1.

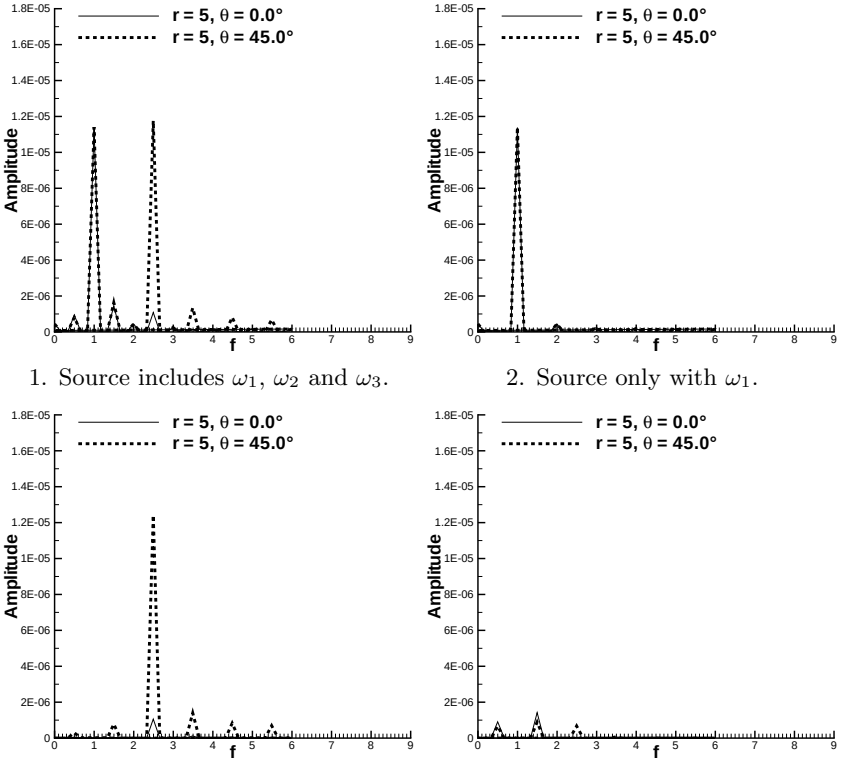


3. Source only with ω_2.



4. Source only with ω_3.

Figure 2.40: Fourier spectra of $\hat{p}$ at $r = 1.75$ in the *unstructured* domain for the partially resolved case.



1. Source includes ω_1, ω_2 and ω_3.
2. Source only with ω_1.
3. Source only with ω_2.
4. Source only with ω_3.

Figure 2.41: Fourier spectra of $\hat{p}$ at $r = 5$ in the *structured* domain for the partially resolved case.

2.7.3 Reflections

The behavior towards reflections at the coupling boundaries shall be examined. Schwartzkopff [88] gave an estimate for the magnitude of possible reflections, when nonlinear Euler equations are coupled with linearized Euler equations. He pointed out that the reflected waves are due to the jump in the wave propagation speed, which changes from a nonlinear $u+c = u+\sqrt{\gamma p/\rho}$ to a linearized $u_0+c_0 = u_0+\sqrt{\gamma p_0/\rho_0}$. This change in propagation speed acts as an artificial material interface. When a wave traveling in one medium impinges perpendicularly on the boundary of a second, different medium, one part will be reflected, one part will go through (general physics of wave propagation, Gerthsen [124]). Its overall intensity will be conserved:

$$I_I = I_R + I_T, \tag{2.153}$$

where the indices denote the impinging, reflected and transmitted waves. The intensities for the waves are

$$I_R = I_I \cdot R = I_I \cdot \left(\frac{n_2 - n_1}{n_2 + n_1}\right)^2, \tag{2.154}$$

$$I_T = I_I \cdot T = I_I \cdot \frac{4 \cdot n_2 \cdot n_1}{(n_2 + n_1)^2}, \tag{2.155}$$

with n_1 and n_2 being the refraction indices of the two media. If the wave travels from a nonlinear domain 1 onto a linear domain 2, these refraction indices will be $n_1 = \frac{1}{u+c}$ and $n_2 = \frac{1}{u_0+c_0}$. Assuming that linearization is allowed and therefore the velocity u normal to the interface can be split into $u = u_0 + u'$, the reflection index $R = \frac{I_R}{I_I}$ can be written as

$$\begin{aligned} R &= \left(\frac{n_2 - n_1}{n_2 + n_1}\right)^2 \\ &= \left(\frac{\frac{1}{u_0+c_0} - \frac{1}{u+c}}{\frac{1}{u_0+c_0} + \frac{1}{u+c}}\right)^2 \\ &= \left(\frac{u' + c - c_0}{u' + 2u_0 + c + c_0}\right)^2. \end{aligned} \tag{2.156}$$

One would expect reflections with an amplitude of $\hat{p}'_R = R \cdot \hat{p}'_I$, for an impinging pressure pulse with a perturbation amplitude $\hat{p}'_I$.
R (and therefore the reflections) will be small ($\ll 1.0$) if it can be assumed that

the speeds of sound c and c_0 are significantly larger than the perturbation velocity u' and that the difference between c and c_0 is not very big. Furthermore, the background velocity u_0 should be smaller than c_0. If the coupling boundary is placed in an area, where a linearized treatment is valid, these assumptions are reasonable.
The estimation for the reflections is checked for several constellations of methods, grids and time steps. The underlying test case is a Gaussian-shaped pressure pulse, that is initialized in the center $(x_c, y_c) = (0, 0)$ of the calculation domain:

$$p' = A \cdot \exp\left(-\frac{1}{2b^2}\left((x - x_c)^2 + (y - y_c)^2\right)\right), \tag{2.157}$$

with the amplitude $A = 0.05$, the halfwidth $b = 0.2$ and the perturbation pressure p'. The other perturbation variables, ρ', u' and v', are set to zero. The governing equations are the nonlinear and the linearized Euler equations. For linearized calculations, the background values are $\rho_0 = 1.0$, $u_0 = 0.0$, $v_0 = 0.0$ and $p_0 = \frac{1}{\gamma}$ with $\gamma = 1.4$. For nonlinear calculations, those background values are added to the initial condition.
The calculation domain consists of two subdomains, an inner one with the dimensions $[-3, 3] \times [-3, 3]$ and an outer one with the dimensions $[-10, 10] \times [-10, 10]$. At $t = 0$, the pressure pulse begins to collapse and propagates into the computational domain. The calculation stops at $t = 8$, before the pulse arrives at the outer domain boundaries and possible reflections from there can occur (Fig. 2.42).
The pressure signal is picked up at a radius of $r = 2$ at two points $P_1 = (2, 0)$ and $P_2 = (\sqrt{2}, \sqrt{2})$ in the inner domain. In order to determine the reflections, the difference between the pressure signal and the signal from a reference solution on an undivided domain is calculated:

$$\Delta p = p - p_{ref}. \tag{2.158}$$

First of all, two Cartesian $\mathcal{O}4$ ADER-FV domains with cell size $\Delta h_0 = 0.0625$ are coupled, hence the interfaces match perfectly. The time step is set to $\Delta t = 0.4$ in both domains. If both domains are either nonlinear or linear, no reflections occur: The difference between the partitioned solution and the reference calculation is not measurable ($\ll 10^{-12}$, see Fig. 2.43, 1.). This changes, when linearized Euler equations are employed in the outer domain, while the inner domain remains nonlinear: P_1 picks up reflections of a magnitude of $\Delta p \leq 0.88 \cdot 10^{-5}$ (Fig. 2.43, 2.).
The following maximum amplitudes for the primitive states were picked up at

P_1: $\hat{\rho}_I = 1.00583$, $\hat{u}_I = u' = 6.13467 \cdot 10^{-3}$, $\hat{v}_I = v' = 0.0$ and $\hat{p}_I = 0.72012$. With $c = \sqrt{\gamma \hat{p}_I / \hat{\rho}_I} = 1.00116$, $c_0 = 1.0$, $u_0 = 0.0$ and an amplitude of the perturbation pressure of $\hat{p}'_I = \hat{p}_I - p_0 = 5.83429 \cdot 10^{-3}$, equation (2.156) yields an expected maximum reflection of $\hat{p}'_R = 1.2 \cdot 10^{-5}$. In fact, the first observed reflection has only about $\frac{1}{2}$, the second one about $\frac{3}{4}$ of this amplitude. As it will be shown later, the second pulse is larger due to superposition of two reflections. The shape and magnitude of the reflections are not affected when the numerical methods, the time steps and even the grids are changed: Figure 2.43, 3. shows the results for a nonlinear $\mathcal{O}4$ ADER-DG domain on an unstructured grid ($\Delta h_0 = 0.0625$ for regular triangles), coupling with a linear $\mathcal{O}8$ ADER-FD domain (Cartesian, $\Delta h_0 = 0.0625$ for the point spacing). Note that the DG domain has a time step ratio of $\frac{1}{10}$ in comparison with the FD domain! Finally, the method on the inner domain is replaced by the Rec-FV scheme on the same unstructured grid (Fig. 2.43, 4.). All diagrams share a great resemblance to the original case!

The influence of the pressure amplitude on the reflections is studied next. The original case is considered again, two structured $\mathcal{O}4$ ADER-FV domains, the inner one solving the nonlinear Euler equations, the outer solving the linearized ones. Now the amplitudes of the initial pulse are varied from $A = 2A_0$ to $A = \frac{1}{2}A_0$ and finally $A = \frac{1}{4}A_0$. The resulting reflections are exactly $\Delta p = 4\Delta p_0$, $\Delta p = \frac{1}{4}\Delta p_0$ and $\Delta p = \frac{1}{16}\Delta p_0$ (Fig. 2.44). Thus, the reflections decay proportionally with A^2, which confirms the assumptions made for the reflection index!

For the next test, the mesh spacings are refined ($\Delta h = \frac{1}{2}\Delta h_0$) and coarsened ($\Delta h = 2\Delta h_0$) equally on both grids. Figure 2.45, 1.-2. shows, that the influence on the reflections is minimal. Also increasing the order of accuracy in both domains from $\mathcal{O}4$ to $\mathcal{O}8$ leaves the shape and magnitude of Δp unchanged (Fig. 2.45, 3.).

Several Δp peaks are picked up at the points P_1 and P_2 (Fig. 2.45, 4.). Their temporal distribution and magnitude explain their origin, when the speed of sound ($c = 1.0$) and the angles of incidence and reflection are considered. At $P_1 = (2, 0)$, three peaks occur: The first one happens at about $t = 4$ and is obviously the direct reflection from the right coupling boundary at $x = 3$. The second signal is captured at $t = 6.3$ and considerably larger than the first one. It is the sum of the two reflections at the upper ($y = 3$) and the lower ($y = -3$) boundary. Last but not least, there is a weak third pulse at about $t = 8$, which is the reflection from the left coupling boundary at $x = -3$. For $P_2 = (\sqrt{2}, \sqrt{2})$, which lies on the 45° diagonal, only two main peaks can be observed: The first one at $t = 4.8$ originates from the reflections at the right and upper coupling

boundary and is the strongest one. The other one occurs at $t = 7.5$ and turns out to be the reflection from the left and lower boundary.
At the very end, a jump in the grid size is allowed: Again, two structured FV domains are coupled, but this time, the outer domain employs twice the spacing of the inner one ($\Delta h = 2\Delta h_0$). Small reflections ($\Delta p \leq 2.8 \cdot 10^{-6}$) are observed now, even if both domains solve the same set of equations (linear or nonlinear, Fig. 2.46, 1.-2.). If mixed equations are used, the single reflections superpose (Fig. 2.46, 3.). The origin of the reflections can be related to the sudden jump in resolution: Another calculation with equal grid sizes, linearized equations, but a jump in the order (from $\mathcal{O}4$ to $\mathcal{O}8$) between the domains produces very similar looking reflections (Fig. 2.46, 4.). However, they are about two magnitudes weaker: The effect due to the jump is is not very strong, as the solution is already well resolved with the fourth order scheme on the given grid. The temporal distribution of the pulses resembles the previous cases.
In conclusion, it has been observed that the reflections caused by changing the equations from nonlinear to linear are generally small compared to the wave amplitudes. This follows directly from equation (2.156). The reflections caused by a jump in resolution depend on the magnitude of the jump. However, they were observed to be even smaller than the ones caused by switching the equations. A prediction for the error regarding the transmitted waves would be of interest, too. Yet a comparison to a reference solution, for example at radius $r = 5.0$ in the outer domain, is difficult. The linearized solution could be either compared to the unpartitioned nonlinear calculation (minus the background state) or to an unpartitioned linear calculation. In fact, a test delivered diametrical results. Slight shifts in phase due to different propagation speeds in the domains produce differences for the location of the pulse. However, following from equation 2.153, the error for the transmitted waves must possess the same magnitude as the reflections, as $R + T = 1.0$.

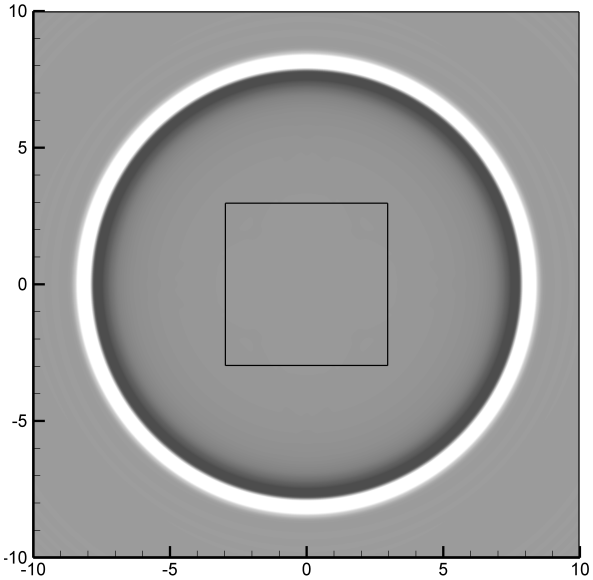
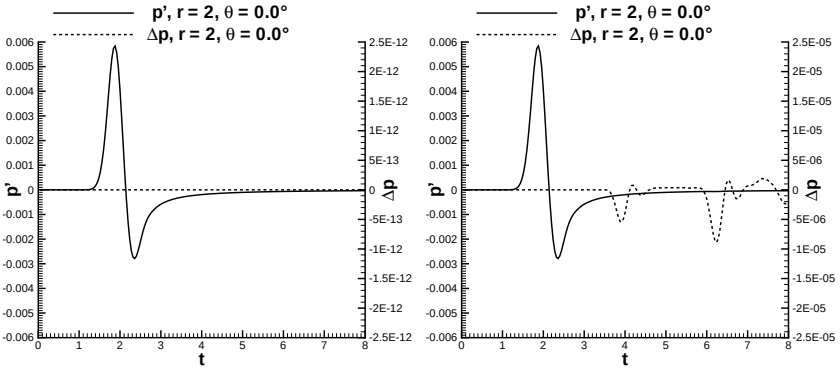
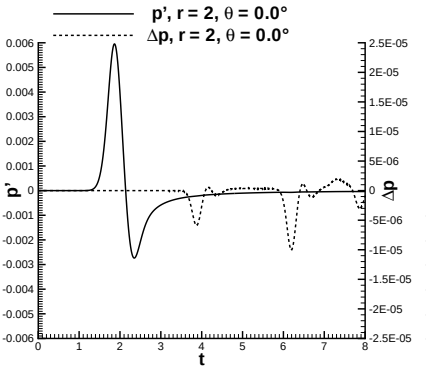


Figure 2.42: Contour plot of the expanding pressure pulse at $t = 8$. The rectangle in the middle marks the boundaries of the inner domain.

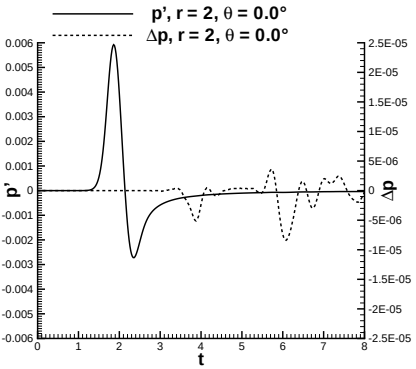


1. FV-nonlin ⇌ FV-nonlin

2. FV-nonlin ⇌ FV-lin

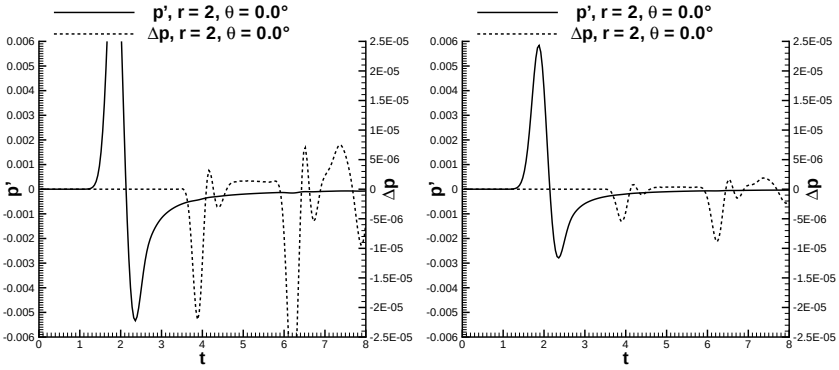


3. DG-nonlin ⇌ FD-lin

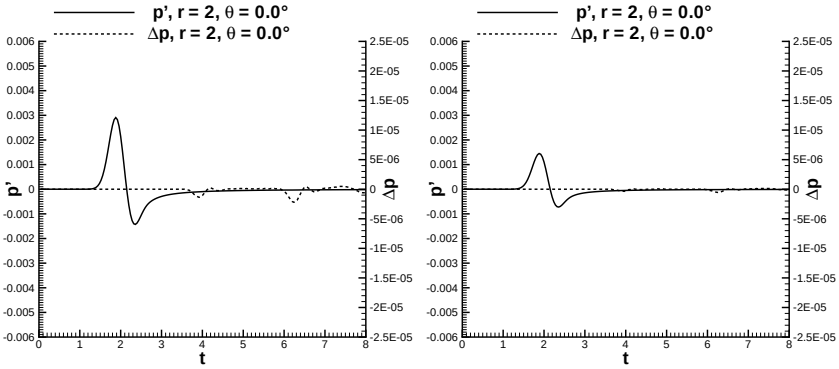


4. Rec-FV-nonlin ⇌ FD-lin

Figure 2.43: Reflections for several domain constellations, involving different methods and grids.

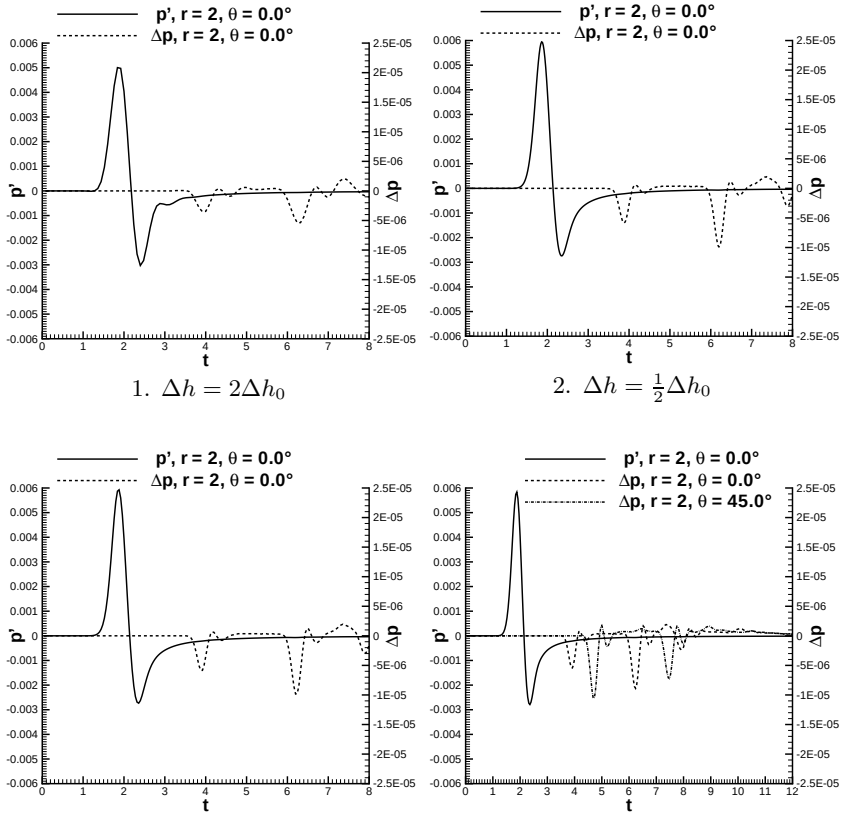


1. $A = 2A_0$ 2. $A = A_0$



3. $A = \frac{1}{2}A_0$ 4. $A = \frac{1}{4}A_0$

Figure 2.44: FV-nonlin $\rightleftharpoons$ FV-lin: The reflections scale with A^2.



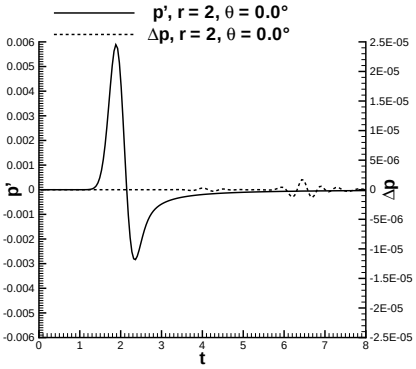
1. $\Delta h = 2\Delta h_0$

2. $\Delta h = \frac{1}{2}\Delta h_0$

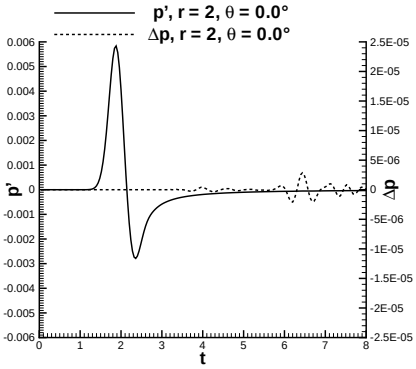
3. $\Delta h = \Delta h_0$, $\mathcal{O}8$

4. Signals in P_1 and P_2

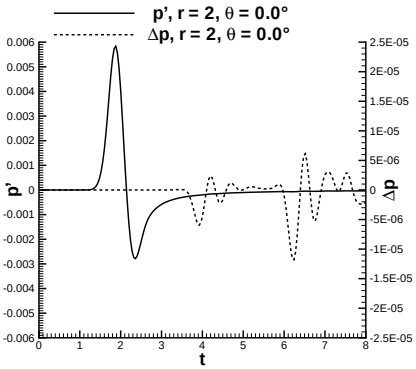
Figure 2.45: FV-nonlin $\rightleftharpoons$ FV-lin: 1.-3.: The reflections are not affected by the mesh size or the order of accuracy. 4.: The temporal development of the reflections indicate their origin.



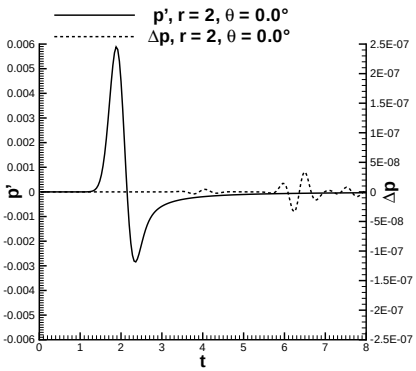
1. FV-lin, $\Delta h_0 \rightleftharpoons$ FV-lin, $2\Delta h_0$



2. FV-nonlin, $\Delta h_0 \rightleftharpoons$ FV-nonlin, $2\Delta h_0$



3. FV-nonlin, $\Delta h_0 \rightleftharpoons$ FV-lin, $2\Delta h_0$



4. FV-lin $\mathcal{O}8 \rightleftharpoons$ FV-lin $\mathcal{O}4$

Figure 2.46: Reflections occur due to a jump in grid size (1.-3.) as well as due to a jump in the order (4.).

2.7.4 Efficiency Estimates

To provide a general efficiency estimate for the domain decomposition approach is a very difficult task. It is clear that one could examine all single coupling aspects independently. However, only the interaction of *all* coupling options can reveal the true potential regarding a computational speed-up. Then it depends on of the actual problem which decomposition features can be exploited and contribute most to an acceleration.
This problem is similar to the difficulty, how to compare different numerical methods. This is by the way one of the sub-categories in the efficiency estimate of the domain decomposition, as a variety of completely different solvers can be used in the same calculation. Each of them must be carefully analyzed regarding its accuracy and efficiency first (which has been done by the developers). Afterwards, they have to be compared *against* each other. This puts already a severe limitation on the grids and the equations that can be used in such a test case. Some of the more complex applications could not be calculated with just *one* of the solvers that is maybe otherwise used in a subdomain of the decomposed same problem. For example a FD code for linearized wave propagation on regular structured grids is surely not able to compute nonlinear fluid dynamics around complex geometries. Hence, a comparison can only take place for simple, standardized benchmark problems which can be solved by all of the methods. The accuracy with regard to a reference solution can be determined along with the required computational effort then.
This means in the domain decomposition context, that an analysis similar to Nordström et al. [77] could be performed. He examined the coupling efficiency between two domains (one unstructured FV domain, one structured FD domain) by contrasting a decomposed simulation with a single unstructured FV calculation and comparing the L_2 errors. Then the relative speed-up can be determined for the coupling parameters that deliver the same errors as the single domain run. However, the significance of such an estimate for more complex examples or even real world calculations is probably not very high. First of all, it is not possible to consider every single feature of the domain coupling approach (subcycling, methods, grids, order of accuracy, equations, etc.), either because it does not make sense for the test case or the number of different constellations would be too large to test them all. Furthermore, in order to be able to compare the results with a reliable reference solution, one is confined to specific examples (e.g., problems with an analytical solution).
On this account, the efficiency analysis is performed only for the concrete examples in section 3. If possible or available, a comparison to a single domain calcu-

lation or a reference solution regarding quality and effort is drawn. Also, significant problem independent efficiency factors (such as $\frac{t_{CPU}}{Elem \cdot Iter}$ and $\frac{t_{CPU}}{DOF \cdot Iter}$) and problem dependent factors ($\frac{t_{CPU}}{DOF \cdot t_{Sim}}$, the subcycling ratio $\frac{\Delta t}{\Delta t_{min}}$, t_{CPU} per domain volume, etc.) are provided together with the calculation parameters (e.g., numbers of elements, iterations, time steps).

3 Numerical Examples

3.1 Multiple Cylinder Scattering

The following problem was posed by Scott E. Sherer [98] in order to test the ability of high order CAA codes to handle complex geometries. Furthermore, this example for the *Fourth CAA Workshop on Benchmark Problems* (Category 2, Case 2) is suitable to demonstrate the numerical robustness, longtime stability and the treatment of non-reflecting boundary conditions of a scheme. Case 2, the scattering of sound from three rigid circular cylinders of different sizes is chosen for the validation of the domain decomposition, as it can be considered the more challenging problem. Sherer et al. [97, 100] provide also analytical and numerical reference data for this case.
The example is governed by the two-dimensional linearized Euler equations with the background values $\rho_0 = 1.0$, $u_0 = 0.0$, $v_0 = 0.0$ and $p_0 = 0.714285714$ and the speed of sound $c = \sqrt{\gamma \cdot \frac{p_0}{\rho_0}} = 1.0$ with $\gamma = 1.4$:

$$\vec{U}'_t + \underline{\underline{A}}\vec{U}'_x + \underline{\underline{B}}\vec{U}'_y = \vec{S}, \tag{3.1}$$

with $\vec{U}' = (\rho', u', v', p')^T$, $\vec{S} = (0, 0, 0, S)^T$ and the Jacobians

$$\underline{\underline{A}} = \begin{pmatrix} u_0 & \rho_0 & 0 & 0 \\ 0 & u_0 & 0 & \frac{1}{\rho_0} \\ 0 & 0 & u_0 & 0 \\ 0 & \gamma \cdot p_0 & 0 & u_0 \end{pmatrix}, \; \underline{\underline{B}} = \begin{pmatrix} v_0 & 0 & \rho_0 & 0 \\ 0 & v_0 & 0 & 0 \\ 0 & 0 & v_0 & \frac{1}{\rho_0} \\ 0 & 0 & \gamma \cdot p_0 & v_0 \end{pmatrix}. \tag{3.2}$$

The spatially distributed and axisymmetric acoustic source term is located at $(x, y)_{Source} = (0, 0)$ and oscillates in time:

$$S = exp\left[-\ln 2 \cdot \frac{(x - x_{Source})^2 + (y - y_{Source})^2}{b^2} \right] \cdot \sin(\omega t), \tag{3.3}$$

where $\omega = 8\pi$ and $b = 0.2$. Manoha et al. [70] stated, that their results differed from the provided analytical solution by a factor $\frac{1}{\gamma - 1}$, which has been also observed in the following calculation. Hence, the presented numerical results

are scaled by a factor $(\gamma - 1) = 0.4$ for comparison purposes.
The computational setup is as follows: Inside the computational domain, three solid cylinders with radii $r_1 = 0.5$, $r_2 = 0.375$ and $r_3 = 0.375$ are located at $L_1 = (-3, 0)$, $L_2 = (3, 4)$ and $L_3 = (3, -4)$. Although the problem is axisymmetric with respect to the x-axis, the full problem in a total area of $[-10, 10] \times [-8, 8]$ is calculated. This area is discretized with three different domains: A structured ADER-FD domain for the far field, a structured Taylor-FD domain containing the source term and an unstructured ADER-DG domain in the vicinity of the cylinders (Fig. 3.2). Note that although the unstructured grids are physically apart, the single unstructured domain contains all three cylinders! A wavelength of $\lambda = \frac{c}{f} = 0.25$ with $f = \frac{\omega}{2\pi} = 4.0$ results from (3.3). In all computations, the structured grid sizes are fixed: On the Cartesian ADER-FD grid, a spacing of $\Delta h = 0.05$ results in a resolution of 5 points per wavelength (PPW). In order to resolve the source very well, 10 PPW ($\Delta h = 0.025$) are used for the Taylor-FD domain in the source region. Two different unstructured grids are used for the calculations: The rather fine and uniform mesh has a spacing of $\Delta h = 0.05$ on the cylinder surfaces and at the coupling boundaries. The coarser grid has also a spacing of $\Delta h = 0.05$ on the cylinder surfaces, but coarsens to $\Delta h = 0.1$ at the coupling boundaries. Note that the finer cylinder spacing is due to the fact, that the problem demands a recording of the pressures exactly on the surface. For DG methods, it is quite difficult to provide a value for the resolution in PPW, but the rule of thumb for the employed ADER-DG scheme is, that the element's order of accuracy equals about the PPW per edge. As a fourth order ADER-DG method is used in every case, the resolution is therefore about 20 PPW for $\Delta h = 0.05$ and 10 PPW for a $\Delta h = 0.1$ spacing.
In order to avoid reflections at the boundary, a sponge layer, which is described in more detail in section 3.3, is employed with the sponge parameters $L = 2$ and $s = 4$. Three different calculations are performed on a single Intel Xeon 5150 $2.66GHz$ core. After a simulation time of about $t = 30 - 40$, the solution becomes periodical. Hence, all simulations are run until $t = 50$ (Fig. 3.1). The root mean square pressure

$$p_{rms} = \sqrt{(\int_{t_0}^{t_0+T} (p')^2 dt)/(\int_{t_0}^{t_0+T} dt)} = \sqrt{< (p')^2 >} \tag{3.4}$$

is determined along the cylinder surfaces and along the centerline $x_{CL} = [-8, 8],\ y = 0$ (Fig. 3.3 and 3.4), where T denotes an oscillation period.

First off all, the fine unstructured $\mathcal{O}4$ ADER-DG domain is coupled with the structured $\mathcal{O}8$ Taylor-FD source domain and a structured $\mathcal{O}8$ ADER-FD domain. Note that the order for the interpolation from the ADER-FD grid onto the Taylor-FD ghost points is actually set to $\mathcal{O}9$ for the time derivatives. While the results along the left cylinder surface are in good agreement with the analytical solution, the results for the smaller cylinders clearly show a shift in phase and amplitude. Because the acoustic waves must cross the relatively coarse far field grid on their way from the source to the outer cylinders, it is suspected that maybe the waves are still insufficiently resolved. Note that the waves are well resolved on their way to the left cylinder, as they travel directly from the Taylor-FD grid into the ADER-DG domain.
Therefore, the order of accuracy in the ADER-FD domain is raised to $\mathcal{O}12$ in a second computation. All grids and other parameters remain the same. This time, the numerical solution matches the analytical one very well on all cylinders and along the centerline, which confirms the previous assumption! The $\mathcal{O}12$ method preserves the wave properties even on a coarse grid.
The fine unstructured grid is replaced by the coarser one in the third calculation. Its elements at the coupling boundaries are now a lot coarser than the spacing on the neighboring grid. Again, all other parameters remain the same, including the structured $\mathcal{O}12$ far field domain. The results along the cylinders (Fig. 3.3, right) and along the centerline are almost identical to the ones on the fine grid, although the jump in grid sizes is now significant at the boundaries and less elements are computed.
The parameters and the performance of the calculations are summarized in Tables 3.1 and 3.2, the dimensions of the calculation domain are given in Table 3.3. It can be seen, that the $\mathcal{O}12$ method in the far field domain is about twice as expensive as the $\mathcal{O}8$ scheme. However, the increase is only about 10% of the total CPU time. This is because the DG method on the unstructured grid consumes most of the computational resources, being approximately one magnitude more expensive than the structured methods in terms of CPU time per DOF and simulation time. Note that this factor is largely dependent on the calculation scenario! In terms of CPU time per DOF and iterations, the DG scheme is actually more efficient, but each iteration has a relatively small time step. It is emphasized, that DOF denotes here the entire state *vector* for a degree of freedom, not each component (e.g., pressure or inner energy) separately. If one is interested the latter case, the total number of DOFs must be scaled up with the number of variables n_{Var} (e.g., $n_{Var} = 4$ for 2D Euler equations) and the CPU times per DOF must be scaled down with $\frac{1}{n_{Var}}$. Last but not least, the use of the coarser grid decreases the overall wall-clock time by one third,

although the time step remains about the same due to the fine spacing at the cylinder surfaces. Less elements need to be calculated for every iteration, while the coarse cells still ensure a very good solution. The most expensive computation (fine unstructured grid, structured $\mathcal{O}12$ far field domain) with 196633 degrees of freedom took $1.5h$ on a single CPU in a $[-10, 10] \times [-8, 8]$ calculation domain until simulation time $t = 50$. Sherer et al. [100] performed a total of 20000 iterations with a time step $\Delta t = 0.002$, hence calculated until a final simulation time of $t = 40$. With a resolution of 8 PPW (277000 grid points), they spent $2.87h$ on 26 processors, thus required a total of $74.62CPUh$. The processor type was not stated. Their actual calculation area in the domain (axisymmetric half-model, including the sponge zone: $[-20, 20] \times [0, 20]$) was $[-10, 10] \times [0, 10]$.

Domain	$\mathcal{O}$	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	4	1.964E-05	1.964E-06	1.117E-03	1.758E-03	1
Str. 1	8	3.159E-06	3.159E-06	9.982E-05	3.164E-02	18
Str. 2	8	5.518E-06	5.518E-06	4.489E-04	1.582E-02	9

Fine unstructured grid, $\mathcal{O}8$ in the ADER-FD domain.

Domain	$\mathcal{O}$	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	4	1.950E-05	1.950E-06	1.109E-03	1.758E-03	1
Str. 1	12	6.261E-06	6.261E-06	1.978E-04	3.164E-02	18
Str. 2	8	5.573E-06	5.573E-06	3.523E-04	1.582E-02	9

Fine unstructured grid, $\mathcal{O}12$ in the ADER-FD domain.

Domain	$\mathcal{O}$	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	4	2.018E-05	2.018E-06	1.171E-03	1.723E-03	1
Str. 1	12	6.634E-06	6.634E-06	1.925E-04	3.445E-02	20
Str. 2	8	5.728E-06	5.728E-06	3.325E-04	1.723E-02	10

Coarse unstructured grid, $\mathcal{O}12$ in the ADER-FD domain.

Table 3.1: Efficiency factors for the multiple cylinder scattering calculations.

Domain	PPW	t_{CPU}[s]	t_{CPU}[%]	#Elem	#DOF	#Iter
Unstr. 1	≈ 20	3402.5	70.26	6090	60900	28449
Str. 1	5	608.3	12.56	121882	121882	1580
Str. 2	10	241.6	4.99	13851	13851	3161
Coupling	-	590.1	12.19	-	-	-
Total	-	4842.5	100.00	141823	196633	-

Fine unstructured grid, $\mathcal{O}8$ in the ADER-FD domain.

Domain	PPW	t_{CPU}[s]	t_{CPU}[%]	#Elem	#DOF	#Iter
Unstr. 1	≈ 20	3377.6	62.08	6090	60900	28449
Str. 1	5	1205.7	22.16	121882	121882	1580
Str. 2	10	244.0	4.48	13851	13851	3161
Coupling	-	613.8	11.28	-	-	-
Total	-	5441.0	100.00	141823	196633	-

Fine unstructured grid, $\mathcal{O}12$ in the ADER-FD domain.

Domain	PPW	t_{CPU}[s]	t_{CPU}[%]	#Elem	#DOF	#Iter
Unstr. 1	≈ 10	1659.3	45.15	2833	28330	29030
Str. 1	5	1173.2	31.93	121882	121882	1451
Str. 2	10	230.3	6.27	13851	13851	2903
Coupling	-	612.0	16.65	-	-	-
Total	-	3674.8	100.00	138566	164063	-

Coarse unstructured grid, $\mathcal{O}12$ in the ADER-FD domain.

Table 3.2: Domain statistics for the multiple cylinder scattering calculations.

Domain	Outer Dimensions	Area	Area[%]
Unstr. 1, Cyl. 1	$[-4.00, -2.00] \times [-1.00, +1.00]$	3.2	1.00
Unstr. 1, Cyl. 2	$[+2.25, +3.75] \times [+3.25, +4.75]$	1.8	0.57
Unstr. 1, Cyl. 3	$[+2.25, +3.25] \times [-4.75, -3.25]$	1.8	0.57
Unstr. 1		6.8	2.1
Str. 1	$[-10.00, +10.00] \times [-8.00, +8.00]$	303.0	95.2
Str. 2	$[-2.00, +2.25] \times [-1.00, +1.00]$	8.5	2.7
Total	$[-10.00, +10.00] \times [-8.00, +8.00]$	318.3	100.0

Table 3.3: Dimensions of the different domains.

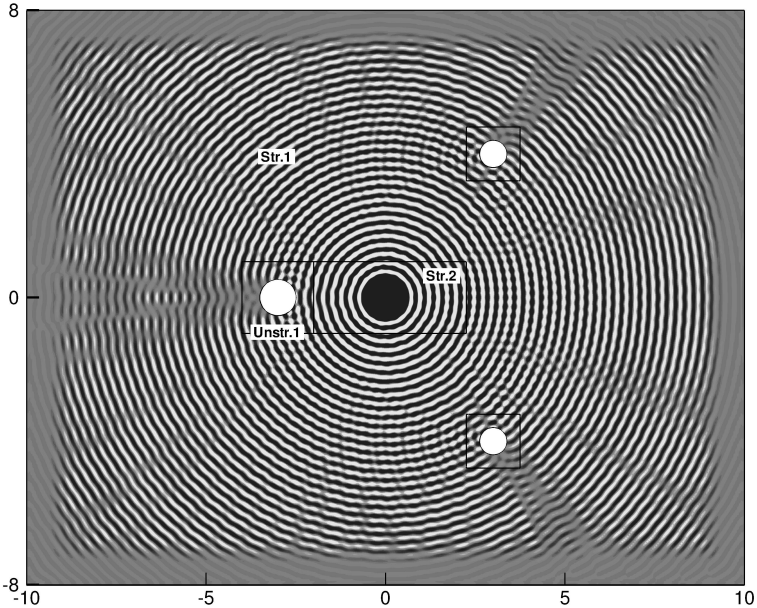


Figure 3.1: Contour plot of p' at $t = 50$.

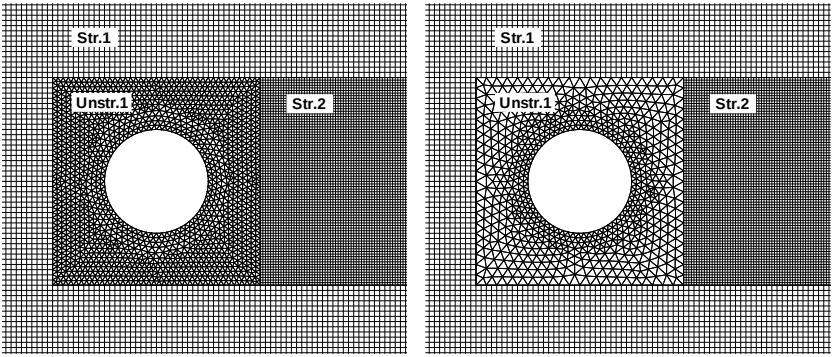


Figure 3.2: Grid configurations for the multiple cylinder scattering example: Left: Fine unstructured grid. Right: Coarse unstructured grid.

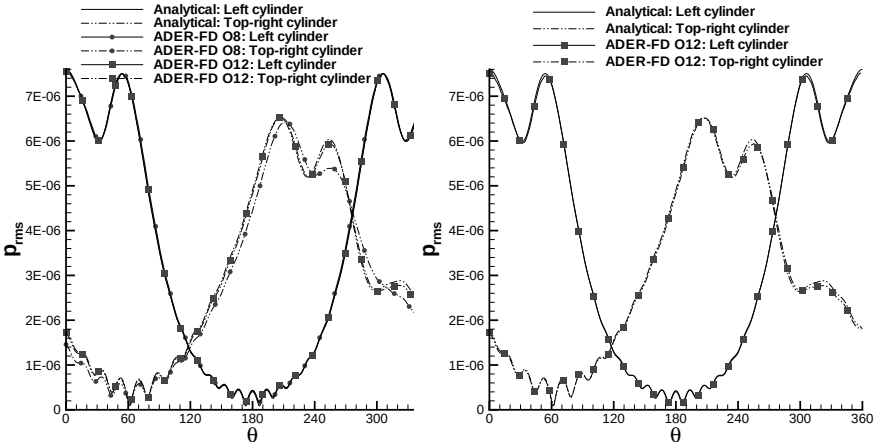


Figure 3.3: p_{rms} along the cylinder surfaces. Left: Fine unstructured grid. Right: Coarse unstructured grid.

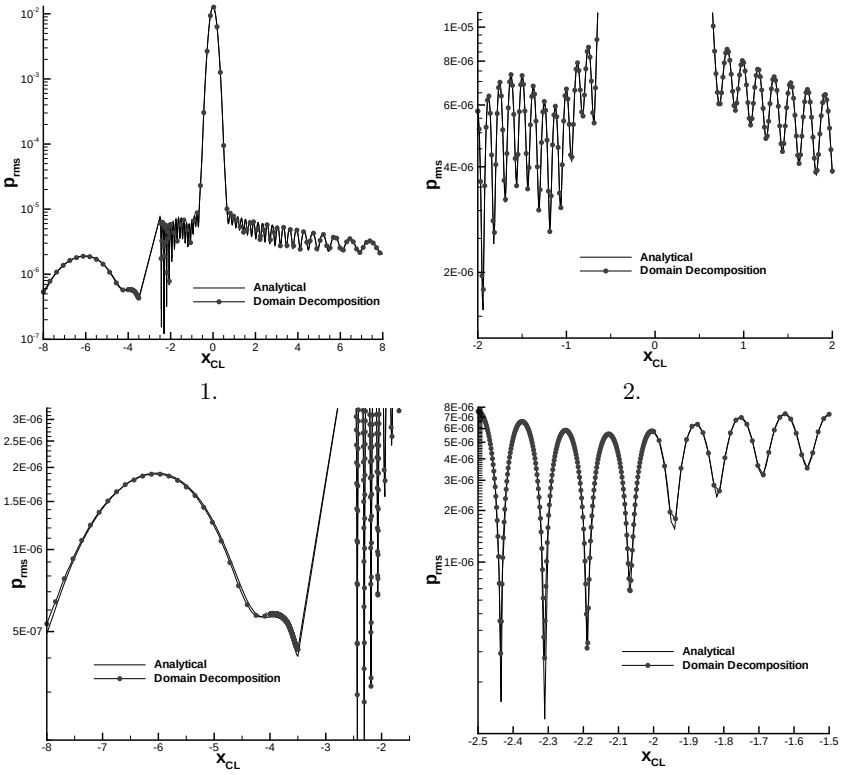


Figure 3.4: p_{rms} along the centerline ($x_{CL} = [-8, 8]$, $y = 0$) for the fine unstructured grid and the $\mathcal{O}12$ ADER-FD domain. 1.: Across the computational domain. 2.-4.: Close-ups of different regions.

3.2 Von Karman Vortex Street

This validation example describes a classical laminar vortex shedding behind a two-dimensional circular cylinder (Fig. 3.5). The diameter of the cylinder is $D = 1$, the ambient density, velocity and pressure are $\rho_\infty = 1.0$, $u_\infty = 0.2$, $v_\infty = 0.0$ and $p_\infty = 0.71428571428$. The resulting Mach number is $Ma = 0.2$. The dynamic viscosity is chosen as $\mu = 0.00133333$, hence the Reynolds number based on the diameter is $Re = 150$. At this Reynolds number, vortex pairs are shed periodically from the downstream side of the cylinder and the laminar flow is basically two-dimensional. This case has been studied extensively in the past, so the results can be compared with both numerical (Inoue et al. [49], Müller [75]) and experimental data (Roshko [86]).
The contour plot of the pressure in Fig. 3.5 shows a close-up of the calculation domain and its composition of different domains, methods, grid types and orders of accuracy. A zoom into the actual grids is plotted in Figs. 3.6 and 3.8. The unstructured domains employ triangular elements, the structured ones contain Cartesian quadrilaterals. The extents of the overall area are $[1200 \times 1200]$, while the total unstructured inner region around the cylinder is only $[35 \times 14]$. See Tables 3.4 and 3.5 for area fractions, CPU times and other domain parameters. Also the time step ratios are given: In the outer regions, Δt is 3, 9, 63 and even 315 times larger than in the unstructured innermost domain. There, the time step is mainly restricted by the small cell sizes which are required for a sufficiently resolved boundary layer (Fig. 3.9). At the outer boundary of the acoustic far field domain, a sponge layer (see section 3.3) is employed to avoid reflections (Fig. 3.7). Note that the eighth order calculation in the far field is very inexpensive (2.27% of the CPU time for 96.4% of the total area, Table 3.5)!
The computation was performed on a single Intel Xeon 5150 $2.66GHz$ core, the overall wall-clock time was $62.19h$. The final simulation time is $t = 2500$, which is far beyond reaching periodicity of the emitted sound in the whole domain (at $t = 600$, the first acoustic waves reach the upper and lower domain boundary; CPU time at $t = 600$: $14.92h$).
As the cylinder enters the $Ma = 0.2$ flow abruptly at $t = 0$, an initial non-physical perturbation is produced and convected in the downstream direction. It can still be seen at $t = 2500$ in the right part of Fig. 3.7. However, this perturbation does not emit spurious noise. The wavelength of the acoustic perturbations in y-direction ($v = 0.0$) is $\lambda = 27.27$. A Fourier analysis at various points in the calculation domain ($P_1 = (20, 0)$, $P_2 = (0, 20)$, $P_3 = (0, 100)$, $P_4 = (0, 300)$, $P_5 = (100, 100)$, $P_6 = (300, 300)$) returns a fre-

quency of $f = 0.03667$ and thus a Strouhal number of $Str = \frac{D \cdot f}{u_\infty} = 0.18335$. The frequency is observed both in the wake and in the acoustic region in the far field and corresponds to the periodical laminar separation at the cylinder. This is in good agreement with the experimental and numerical references (Roshko [86]: Values range from $Str = 0.179 - 0.182$, Inoue et al. [49]: $Str = 0.183$, Müller [75]: $Str = 0.1831$. The values for the lift and drag coefficients were sampled during the simulation time $t = [1500, 2500]$ (Fig. 3.12): The mean drag coefficient is $\bar{C}_D = 1.3309$, its amplitude is $\hat{C}_D = 0.0257$ and the lift coefficient amplitude is $\hat{C}_L = 0.5192$ (Inoue et al.: $\bar{C}_D = 1.32$ at $Ma = 0.1$, $\hat{C}_D = 0.026$, $\hat{C}_L = 0.52$, Müller: $\bar{C}_D = 1.34$, $\hat{C}_D = 0.02614$, $\hat{C}_L = 0.5203$). In Fig. 3.10, the amplitude of the instantaneous fluctuation pressure $\tilde{p}' = p(x, y, t) - \bar{p}(x, y)$ with the mean flow $\bar{p}(x, y) = \frac{1}{T} \int_{t_0}^{t_0+T} p(x, y, t) dt$ is scaled with $\frac{1.}{Ma_\infty^{2.5}}$. The data are compared to the results by Inoue et al. [49] and Müller [75]. They are in good accordance especially with the results by Inoue et al. and close to the results of Müller. Finally, the dipole-like directivity of the sound field is depicted by Fig. 3.11.

Domain	Method	$\mathcal{O}$	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{Elem \cdot t_{Sim}}$	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	FV, N.S.	4	4.681E-05	1.151E-02	4.064E-03	1
Unstr. 2	FV, N.S.	4	4.854E-05	3.978E-03	1.219E-02	3
Unstr. 3	FV, N.S.	4	4.483E-05	1.224E-03	3.657E-02	9
Unstr. 4	FV, N.S.	4	4.631E-05	1.265E-03	3.657E-02	9
Str. 1	FD, LEE	8	7.502E-06	5.855E-06	1.280E+00	315
Str. 2	FV, EE	4	3.758E-05	1.466E-04	2.560E-01	63
Str. 3	FV, N.S.	2	1.245E-05	4.856E-05	2.560E-01	63
Str. 4	FV, N.S.	4	5.821E-05	2.271E-04	2.560E-01	63

Table 3.4: Efficiency factors for the different Von Karman domains.

Domain	Outer Dimensions	$t_{CPU}[\%]$	Area[%]	#Elem	#Iter
Unstr. 1	$r_1 = 0.5 \rightarrow r_2 = 0.9$	58.16	0.0001	4526	614565
Unstr. 2	$r_1 = 0.9 \rightarrow r_2 = 1.5$	6.83	0.0003	1538	204855
Unstr. 3	$r_1 = 1.5 \rightarrow r_2 = 3.5$	2.97	0.0022	2169	68285
Unstr. 4	$[-5, 30] \times [-7, 7]$	7.72	0.0314	5463	68285
Str. 1	$[-600, 600] \times [-600, 600]$	2.27	96.4445	347885	1951
Str. 2	$[-40, 100] \times [-40, 40]$	6.23	0.6757	38032	9755
Str. 3	$[100, 600] \times [-40, 40]$	8.68	2.7778	160000	9755
Str. 4	$[30, 100] \times [-7, 7]$	0.99	0.0681	3920	9755
Coupling	-	6.15	-	-	-
Total	-	100.0	100.0	563533	-

Table 3.5: Domain parameters and properties.

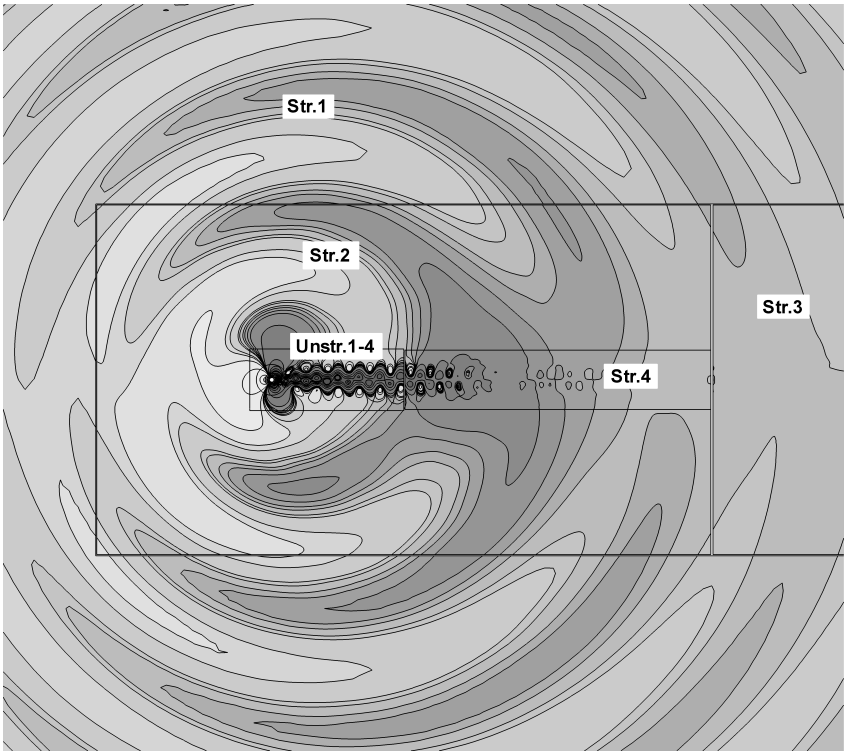


Figure 3.5: Von Karman vortex street at $t = 2500$; *Unstructured domains 1-4, cylinder:* Navier-Stokes equations, triangular meshes, nonlinear FV scheme, $\mathcal{O}4$. *Structured domain 1, acoustic far field:* Linearized Euler equations, Cartesian mesh, FD scheme, $\mathcal{O}8$. *Structured domain 2, near field:* nonlinear Euler equations, Cartesian mesh, FV scheme, $\mathcal{O}4$. *Structured domain 3, damping zone:* Navier-Stokes, Cartesian mesh, FV scheme, $\mathcal{O}2$. *Structured domain 4, wake:* Navier-Stokes, Cartesian mesh, FV scheme, $\mathcal{O}4$.

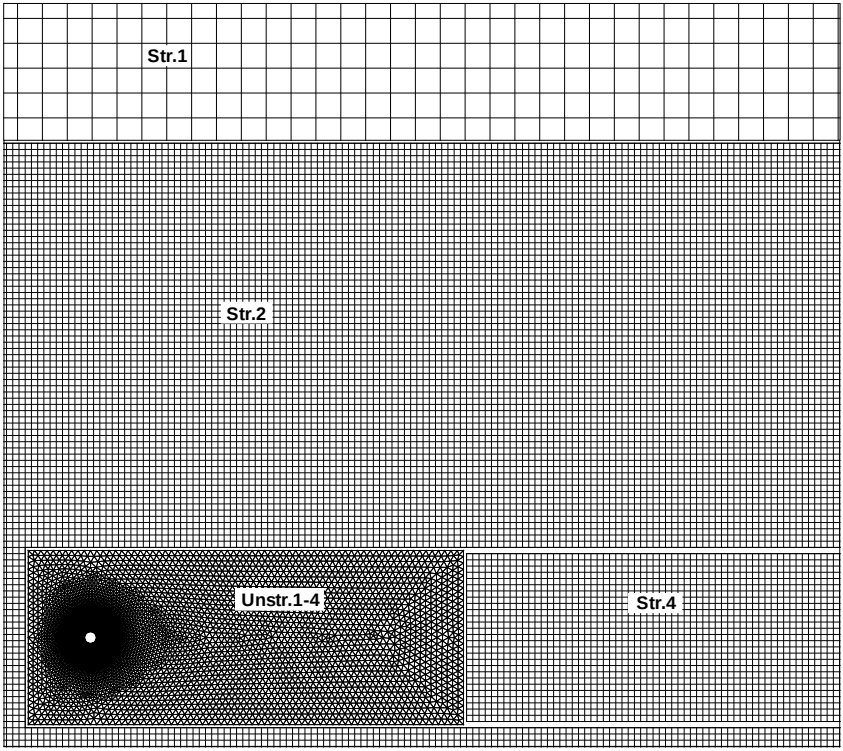


Figure 3.6: Grid topology: Depicted are the fine unstructured triangular grids around the cylinder, the coarser structured mesh in the near field/wake region and the coarse acoustic grid for the far field.

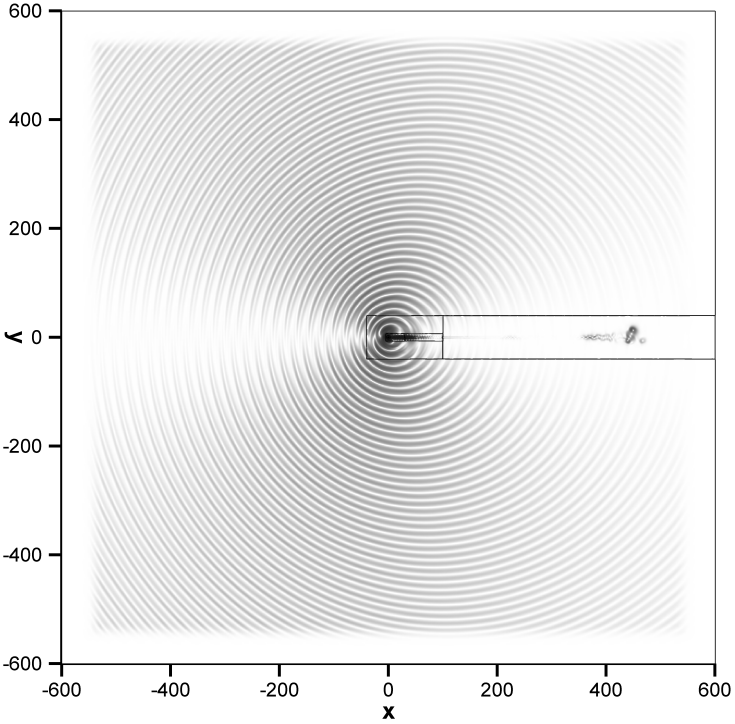


Figure 3.7: The total simulation area at $t = 2500$. A sponge layer absorbs the sound waves at the boundaries. The non-physical initial perturbation is still visible downstream at the right side.

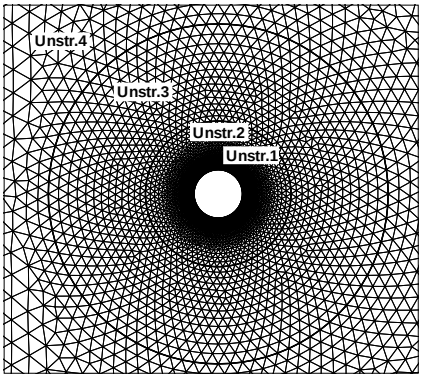


Figure 3.8: Close-up of the innermost unstructured grids in the direct vicinity of the cylinder.

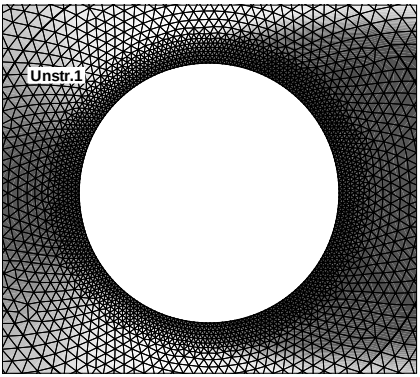


Figure 3.9: Grid cells for a well resolved boundary layer. The contours depict the absolute value of the velocity $u_{abs} = \sqrt{u^2 + v^2}$.

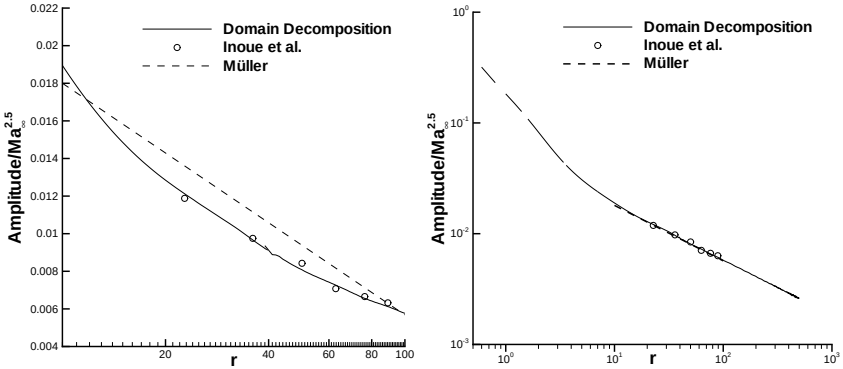


Figure 3.10: Amplitude of the scaled instantaneous fluctuation pressure $\frac{\tilde{p}'}{Ma_\infty^{2.5}}$ on the positive y-axis at $x = 0$. Left: Close-up. Right: Across the whole domain.

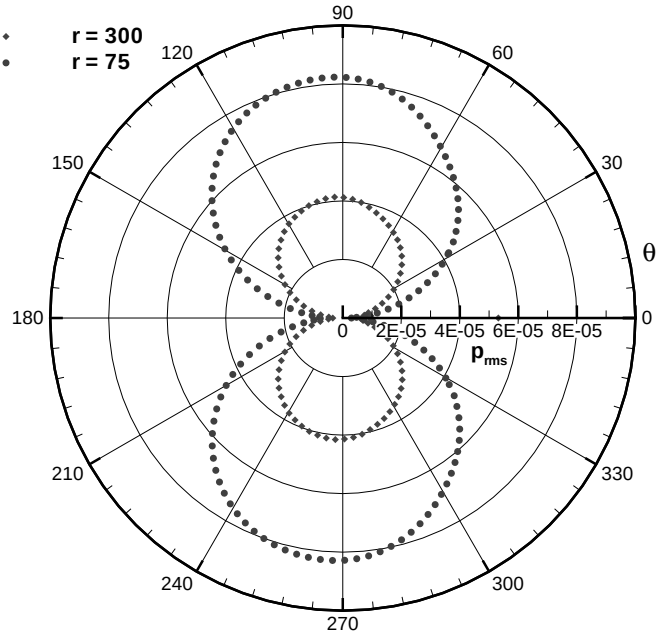


Figure 3.11: Polar plots of the root mean square of the fluctuation pressure at $r = 75$ and $r = 300$.

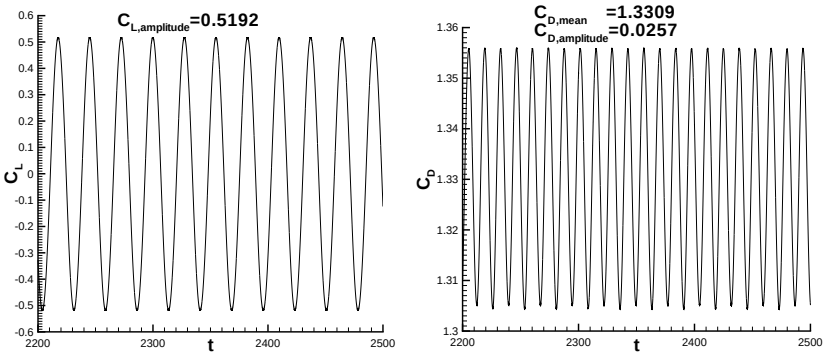


Figure 3.12: Lift and drag coefficients.

3.3 Sphere Scattering

The scattering of sound by a solid sphere was proposed by Morris [73] for the *Second CAA Workshop on Benchmark Problems* (Category 1, Problems 3 and 4). He also provided details about the analytical solution, including the numerical evaluation of spherical Hankel functions [72]. The governing equations in this three-dimensional example are the linearized Euler equations with the mean flow variables $\rho_0 = 1.0$, $u_0 = 0.0$, $v_0 = 0.0$, $w_0 = 0.0$ and $p_0 = 0.714285714$. The speed of sound is $c = \sqrt{\gamma \cdot \frac{p_0}{\rho_0}} = 1.0$ with $\gamma = 1.4$. A solid sphere with the radius $r = 1$ is located at $\vec{X}_{Sphere} = [0, 0, 0]$. In its vicinity at $\vec{X}_{Source} = [2, 0, 0]$, a spatially distributed source S on the right hand side of the pressure term oscillates:

$$S = -A \cdot exp[-B \cdot \ln 2 \cdot ((x - x_{Source})^2 + y^2 + z^2)] \cdot \cos \omega t, \qquad (3.5)$$

with $A = 0.01$, $B = 16$, $x_{Source} = 2$. In the following, $\omega = 2\pi$ (Problem 3) is chosen as frequency, thus the expected wavelength of the generated sound is $\lambda = \frac{c}{f} = 1$ with $f = \frac{\omega}{2\pi} = 1$.
In order to keep the additional computational effort for the source term low, it is evaluated only until a given radius from the source center. For $r = (x - x_{Source})^2 + y^2 + z^2 = 0$, Eq. 3.5 yields $S_0 = S(r = 0) = -0.01$. Due to the exponential decay, the source term's cut-off radius is set to $r = 1.5$: $\frac{S(r=1.5)}{S_0} = 1.46 \cdot 10^{-11}$.
The overall computational domain is divided into one unstructured and one structured volume (Fig. 3.13).
A sponge layer of the following form is employed on the structured finite difference grid in order to avoid reflections:

$$\tilde{u}_{ijk}^{n+1} = (1 - \sigma)\, u_{ijk}^{n+1} + \sigma u_{in}\left(x_i, y_j, z_k, t^{n+1}\right), \qquad (3.6)$$

where u_{ijk}^{n+1} is the time update of the numerical solution as computed by the numerical scheme, $\tilde{u}_{ijk}^{n+1}$ is the modified numerical solution after applying the sponge layer and u_{in} is the prescribed solution, which must be given as a function of $\vec{X}$ and t at the boundary. The sponge parameter $0 \leq \sigma \leq 1$ is defined as

$$\sigma = \begin{cases} \left(\frac{L-\delta}{L}\right)^s & \text{if} \quad \delta \leq L \\ 0 & \text{if} \quad \delta > L \end{cases} . \qquad (3.7)$$

In (3.7), L denotes the thickness of the sponge layer, δ is the distance of grid point P_{ijk} to the nearest boundary of the computational domain, and s is the

power of the sponge layer. In the following computations, the sponge parameters are chosen as $L = 1$ and $s = 4$.
The domain $[-1.75, -1.75, -1.75] \times [3.75, 1.75, 1.75]$ in the direct vicinity of the sphere consists of an unstructured tetrahedral mesh and includes the source region. The $\mathcal{O}4$ ADER-DG method is employed and the time step in this domain is $\Delta t = 1.664 \cdot 10^{-3}$ (CFL=0.3). The elements directly at the sphere surface are slightly refined with an interval size of $\Delta h = 0.2$, compared to an interval size of $\Delta h = 0.4$ at the outer boundaries. The interval size denotes the average edge length of a surface triangle. The minimum inner sphere radius of a tetrahedron is $l_{min} = 0.0194$. As it is not trivial to quantify the resolution of the ADER-DG method on tetrahedral grids, the following approximation is made: The number of degrees of freedom per element in the unstructured domain is $n_{DOF} = \frac{\mathcal{O}\cdot(\mathcal{O}+1)\cdot(\mathcal{O}+2)}{6} = 20$. By averaging the total number of DOFs for the considered volume and for each spatial direction, one obtains approximately 14.62 DOFs per wavelength for the unstructured grid. A total of 1520 ghost tetrahedrons couple with the structured grid at the domain interface, which results in 97280 connecting Gauss points. The unstructured mesh remains fixed in the following computations.
In the first calculation, the outer far field domain $[-6, -6, -6] \times [8, 6, 6]$ consists of a Cartesian mesh with an interval size of $\Delta h = 0.1$ and thus a resolution of 10 PPW. The finite difference scheme has an order of accuracy of $\mathcal{O}8$. A total of 31328 ghost points are required for the coupling with the unstructured grid. In a second calculation, the interval size of the structured grid is doubled ($\Delta h = 0.2$), hence the resolution is 5 PPW. For this configuration, the total number of structured ghost points coupling with the unstructured domain is only 6264. Again, the $\mathcal{O}8$ FD scheme is employed.
Last but not least, the domain decomposition results are compared to a calculation that has been performed on a single tetrahedral grid, using the $\mathcal{O}4$ ADER-DG method. The calculation domain is spherical with its center at $[1, 0, 0]$ and a radius of $r = 7$. In order to keep the grid similar to the one in the decomposition approach, the grid has the same size $\Delta h = 0.2$ on the sphere surface. At the boundary, a spacing of $\Delta h = 0.4$ is imposed, resulting in a grid for a well resolved solution.
All computations were performed on a single Intel Xeon 5150 $2.66GHz$ core. Although the solution reaches periodicity in every point of the domain very quickly (in a simulation time of less than $t = 5$), the calculation was run until $t = 20$, in order to ensure a sufficiently large sample for the analysis.
Figure 3.15 shows the directivity plot of the root mean square fluctuation pressure. It was picked up at several radii in the xy-plane around the sphere center

(Fig. 3.14), $r = 1$ (surface of the sphere), $r = 3$ and $r = 5$. The results of the three calculations are all in good agreement with the reference solution.
Tables 3.6 and 3.7 show the statistics for the three computations: Both decomposition calculations are considerably faster than the single ADER-DG calculation in terms of wall-clock time. The tables also provide a reason why: If the DOFs per DG element are considered as the points that are necessary for the resolution of the waves, the efficiency of the DG and the FD method can be compared for this specific example. DOF denotes here the entire state *vector* for a degree of freedom, not each component (e.g., pressure) separately. Otherwise, the total number of DOFs must be scaled up with the number of variables n_{Var} and the CPU times per DOF must be scaled down with $\frac{1}{n_{Var}}$ (here: $n_{Var} = 5$). The CPU time per DOF and per simulation time unit $\frac{t_{CPU}}{DOF \cdot t_{Sim}}$ shows an advantage of about one magnitude for the structured $\mathcal{O}8$ FD method (Table 3.6). This advantage is strongly example dependent because it arises from the much bigger time step that can be made on the structured grid per iteration. The time step on the unstructured grid is limited by the DG method itself and by the finer grid that is required in order to resolve the sphere. On the other hand, the CPU time per element and iteration $\frac{t_{CPU}}{Elem \cdot Iter}$ permits an example independent (and only processor specific) statement about the methods' cost per iteration. Here, the $\mathcal{O}4$ ADER-DG method is again more expensive but becomes actually cheaper than the FD scheme if the CPU time per DOF and iteration is considered. However, the time step Δt per iteration is much greater for the structured grid because of a less restrictive stability condition and a more regular grid. Note that the edge spacing $\Delta h = 0.2$ of the coarsest FD grid is the same as the interval size $\Delta h = 0.2$ for the surface triangles on the sphere. Last but not least, a comparison between the two domain decomposition calculations (Table 3.7) demonstrates the benefit of using high order methods: The structured far field grid covers almost 97% of the total domain volume. By decreasing the resolution from 10 PPW to 5 PPW, the calculation effort is reduced by a factor of almost $\frac{1}{16}$th! This is due to the two times bigger time step on the coarser grid, furthermore only $\frac{1}{2^3}$ of the grid points are needed. At the same time, the sound waves are still resolved well by the high order method, while a lower order method, for example $\mathcal{O}4$, begins to show dissipation errors for 5 PPW at the outer regions of the domain (Fig. 3.15).

Domain	$\mathcal{O}$	PPW	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$
Unstr. 1	4	≈ 15	8.37E-05	4.19E-06	2.52E-03
Str. 1	8	10	1.39E-05	1.39E-05	4.64E-04

Domain decomposition, 10 PPW in the structured domain.

Domain	$\mathcal{O}$	PPW	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$
Unstr. 1	4	≈ 15	8.37E-05	4.19E-06	2.36E-03
Str. 1	8	5	1.53E-05	1.53E-05	2.55E-04

Domain decomposition, 5 PPW in the structured domain.

Domain	$\mathcal{O}$	PPW	$\frac{t_{CPU}[s]}{Elem \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot Iter}$	$\frac{t_{CPU}[s]}{DOF \cdot t_{Sim}}$
Unstr. 1	4	≈ 15	6.94E-05	3.47E-06	2.06E-3

Single ADER-DG domain.

Table 3.6: Efficiency factors for the domain decomposition and the single unstructured domain calculation.

Domain	t_{CPU}[s]	t_{CPU}[%]	Vol.	Vol.[%]	#Elem	#DOF	#Iter	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	9941	25.5	63.2	3.1	9874	197480	12024	1.664E-03	1
Str. 1	18068	46.4	1948.6	96.9	1948625	1948625	668	2.995E-02	18
Coupling	10958	28.1	-	-	-		-	-	-
Total	**38968**	100.0	2011.8	100.0	1958499	2146105	-	-	-

Domain decomposition, 10 PPW in the structured domain.

Domain	t_{CPU}[s]	t_{CPU}[%]	Vol.	Vol.[%]	#Elem	#DOF	#Iter	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	9316	72.1	63.2	3.1	9874	197480	12024	1.664E-03	1
Str. 1	1301	10.1	1948.6	96.9	255362	255362	334	5.989E-02	36
Coupling	2305	17.8	-	-	-		-	-	-
Total	**12922**	100.0	2011.8	100.0	265236	462716	-	-	-

Domain decomposition, 5 PPW in the structured domain.

Domain	t_{CPU}[s]	t_{CPU}[%]	Vol.	Vol.[%]	#Elem	#DOF	#Iter	Δt	$\frac{\Delta t}{\Delta t_{min}}$
Unstr. 1	**141394**	100.0	1432.6	100.0	171687	3433740	11863	1.686E-03	1

Single ADER-DG domain.

Table 3.7: Statistics for the domain decomposition and the single unstructured domain calculation.

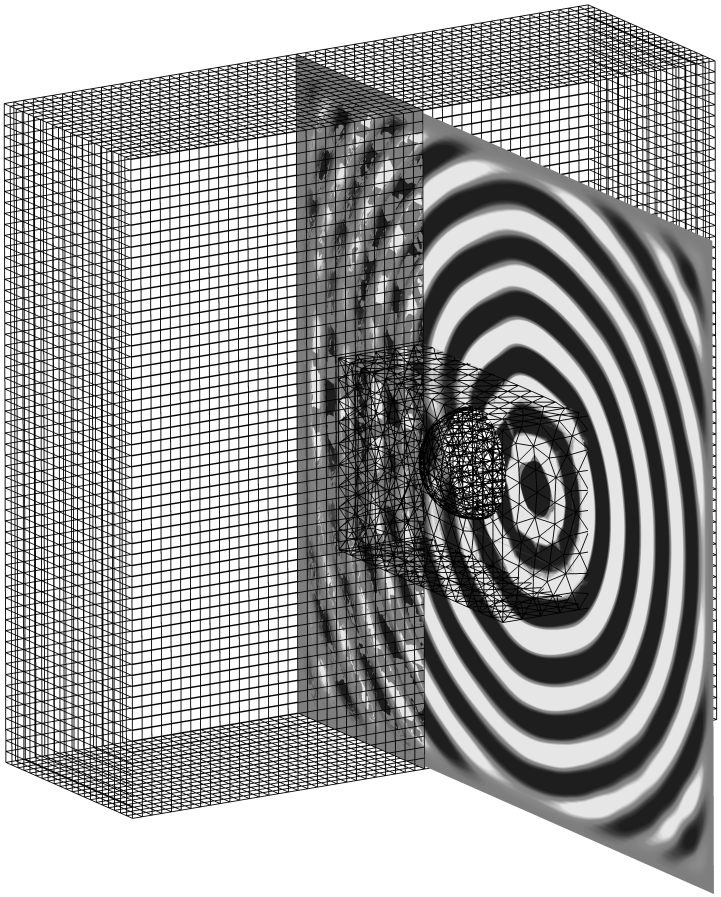


Figure 3.13: A cut-out of the simulation domain including the structured and unstructured mesh as well as a slice of the fluctuation pressure contour plot. The structured mesh is shown in the 5 PPW configuration.

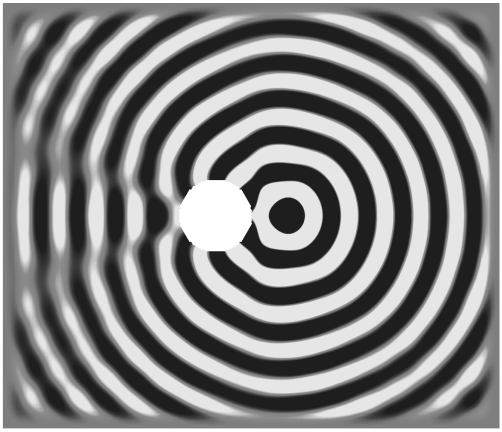


Figure 3.14: Contour plot of the fluctuation pressure in the xy-plane.

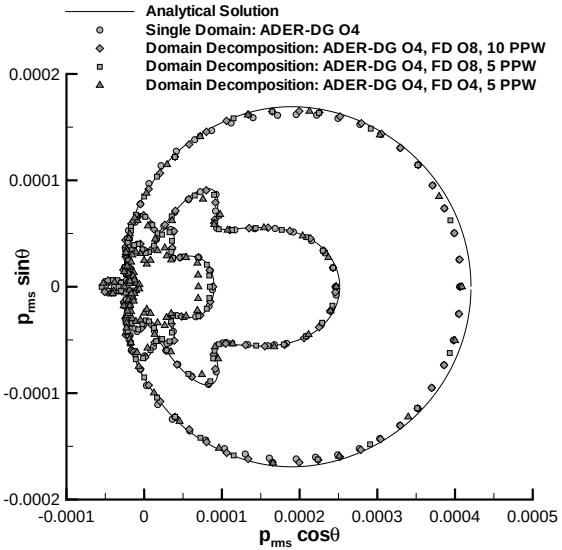


Figure 3.15: Directivity plots of the RMS fluctuation pressure.

3.4 Supersonic Free Jet

The applicability of the coupling framework to real life problems shall be demonstrated in this last example. The chosen scenario, a gas ejector with a supersonic free jet at $Ma = 1.4$, simulates a common pneumatics component in industrial applications. Both experimental and numerical reference data are available (Schönrock [87]).
The setup consists of a round duct with an inner diameter of $4.6mm$. Four kidney-shaped orifices, the actual outlets, are located on its bottom at $z = 0mm$ (Figs. 3.17- 3.18). For the simulation, each orifice extends to a depth of $z = -0.5mm$, where the following inflow conditions are prescribed: Air ($\gamma = 1.4$, $R = 287.058 \frac{J}{kg \cdot K}$) enters the domain with a velocity in z-direction (nozzle axis) of $w = 406.712 \frac{m}{s}$, a pressure of $160000Pa$ and a temperature of $210K$. This results in a local Mach number of $Ma = 1.4$ and a density of $\rho = 2.654 \frac{kg}{m^3}$. As the ambient conditions are $\rho_\infty = 1.188 \frac{kg}{m^3}$, $u_\infty = v_\infty = w_\infty = 0.0 \frac{m}{s}$ and $p_\infty = 100000Pa$, the inflow immediately forms under-expanded jets. The Reynolds number based on the width of the four orifices is $Re = 30000$ ($\mu = 1.831 \cdot 10^{-5} \frac{kg}{m \cdot s}$, $Pr = 0.72$).
The consequence is a flow field that is dominated by a system of shock cells inside the duct, breaking up outside into a heavily fluctuating flow with strong acoustic efficiency. Here, the simulation of the emitted noise must capture non-linear wave propagation and the acoustic feedback from the walls. The problem demands a great deal of the simulation: Due to the delicate geometry and the small-scale shock cells, the grid spacing needs to be very fine in the vicinity of the nozzle. Along with high jet velocities, this is reflected in very small time steps. The overall number of elements becomes very large, although a calculation with DNS resolution is not considered (Kolmogorov scale $\sim 10^{-7}$m, Schönrock [87]). It shall be emphasized at this point, that no subgrid-scale model was employed but the numerical viscosity of the solvers and a coarser grid. Figures 3.16- 3.18 illustrate the decomposition of the domain into three different volumes: An unstructured tetrahedral mesh ($\Delta h \approx 0.8 \cdot 10^{-4} m$, 2.06 million cells) is employed in the nozzle area. The structured core region of the outside flow requires also fine elements for the resolution of shear layers and vortices ($\Delta h = 1.0 \cdot 10^{-4} m$, 9 million elements), while the extended size of the surrounding acoustic domain results in 25 million hexahedrons despite a coarser mesh ($\Delta h = 2.0 \cdot 10^{-4} m$). Table 3.8 provides details about the domain parameters and the employed methods. In all domains, the Navier-Stokes equations are solved with second order FV methods (WENO reconstruction with param-

eters $\lambda_C = 10^4$, $\epsilon = 1.0$, $r = 3$). The HLLE flux is employed and the CFL number is set to 0.25 for the structured domains and to 0.5 for the unstructured one. As the grid resolution is quite high, only first order interpolation is used between the connected volumes. At the outer boundaries of the domain, a sponge layer (section 3.3) is applied with power $s = 2$ and different thicknesses at the lower ($L = 0.003m$), lateral ($L = 0.006m$) and upper side ($L = 0.010m$). The calculation was performed for a simulation time of $0.8ms$ in $24h$ on the *BW-GRID* cluster of the High Performance Computing Center Stuttgart (HLRS). A total of 1024 cores (two Intel Xeon 5440 $2.83GHz$ quad-core CPUs on 128 nodes) were distributed to the unstructured domain (512 cores) and to the structured ones (256 cores each). The accumulated CPU time was therefore $t_{CPU,acc} = 24576CPUh$. The different MPI partitions in each subdomain can be seen in the cut through the overall volume in Fig. 3.16.
The contour plot in Fig. 3.19 clearly shows the propagation of the emitted sound waves into the far field. At two different points inside the domain ($P_1 = (0.01697m, 0.0m, 0.004m)$, $P_2 = (0.01697m, 0.0m, 0.06m)$), the pressure signal was picked up and converted to SPL spectra by a non-averaging FFT (Fast Fourier Transformation). The domain decomposition results for the audible range are in good agreement with the experimental and numerical reference data by Schönrock (Figs. 3.20- 3.21). Note that the experimental recordings were frequency averaged by the measurement system over a longer time. The low frequencies require a larger sample interval ΔT_{Sample}, as the total simulation time of $0.8ms$ corresponds to a frequency of $1.25kHz$. As only the periodical part of the data set can be properly evaluated (beginning at about $t = 0.35ms$ for P_1 and $t = 0.59ms$ for P_2, where T_{Sample} starts), the threshold for reliable FFT results lies considerably above this value. Moreover, a careful parameter study for the upper sponge region is expected to improve the results for P_2, which is close to the base flow and thus directly affected by a non-ideal outflow condition. Nevertheless, the simulation predicts the OASPL (Overall Sound Pressure Level) in both points very well.
For this calculation, which has not been especially optimized with regard to accuracy or efficiency, the results are considered quite respectable. Here, a thorough examination of the domain parameters and coupling options (grid size, order of the methods, interpolation order, grid size ratio) has the potential to increase the quality of the solution, while reducing the computational effort at the same time. Furthermore, compromises had to be made for the distribution of computational resources to the subdomains: While the expensive unstructured part should have gotten as many processors as possible, the cheaper but bigger structured part underlied memory limitations ($2GB$ per

core on the considered system) and was forced to use more CPUs than for a well-balanced computation. Hence, some processors carried a higher computational load, others managed to complete their time stepping earlier and had to wait for the data exchange or for MPI information. Note that the percentage of $t_{CPU,acc}$ in Table 3.8 refers in case of the subdomains to the CPU time actually spent in the *solver* routines, while the fraction of the data exchange contains the overall coupling effort of all domains *plus* the idling times spent waiting for other partitions to synchronize. Thus, an unbalanced CPU distribution always results in a higher coupling overhead.
Regardless of these restrictions, the performance is convincing: Schönrock estimates 1400 CPU days per millisecond simulation time for his LES calculations (*ANSYS CFX*), which is close to the 1024 CPU days per $0.8ms$ for the domain decomposition. It is stressed however, that a much larger number of finite volume cells (totalling 36 million) were calculated with a robust WENO scheme and no simplifications exploiting the symmetrical geometry were made (Schönrock: Half-model with 3.38 million elements). Last but not least, the problem is a good candidate for a distributed computing approach as examined in section 2.6.2: While the unstructured part around the nozzle would still be run on a parallel cluster with scalar CPUs, the Cartesian far field domains perform much better on vector architectures (about a magnitude for the NEC-SX8, see Table 2.6 in section 2.1.4). A direct noise computation of the audible spectrum could then be realized for a far bigger volume, for example in a $1m^3$ cube.

Domain	Method	$\mathcal{O}$	Δt	$\frac{\Delta t}{\Delta t_{min}}$	#Elem	#Iter
Unstr. 1	Rec-FV	2	4.548E-09	1	2057518	173768
Str. 1	STE-FV	2	3.184E-08	7	$9 \cdot 10^6$	24821
Str. 2	STE-FV	2	6.368E-08	14	$25 \cdot 10^6$	12410

Domain	Outer Dimensions [m]	Vol.[%]	$t_{CPU,acc}$ [%]
Unstr. 1	$[-0.0050, +0.0050]^2 \times [-0.0050, -0.0110]$	0.55	43.82
Str. 1	$[-0.0075, +0.0075]^2 \times [-0.0050, -0.0350]$	3.95	15.06
Str. 2	$[-0.0250, +0.0250]^2 \times [-0.0050, -0.0750]$	95.50	4.38
Coupling	-	-	36.74

Table 3.8: Domain parameters and statistics.

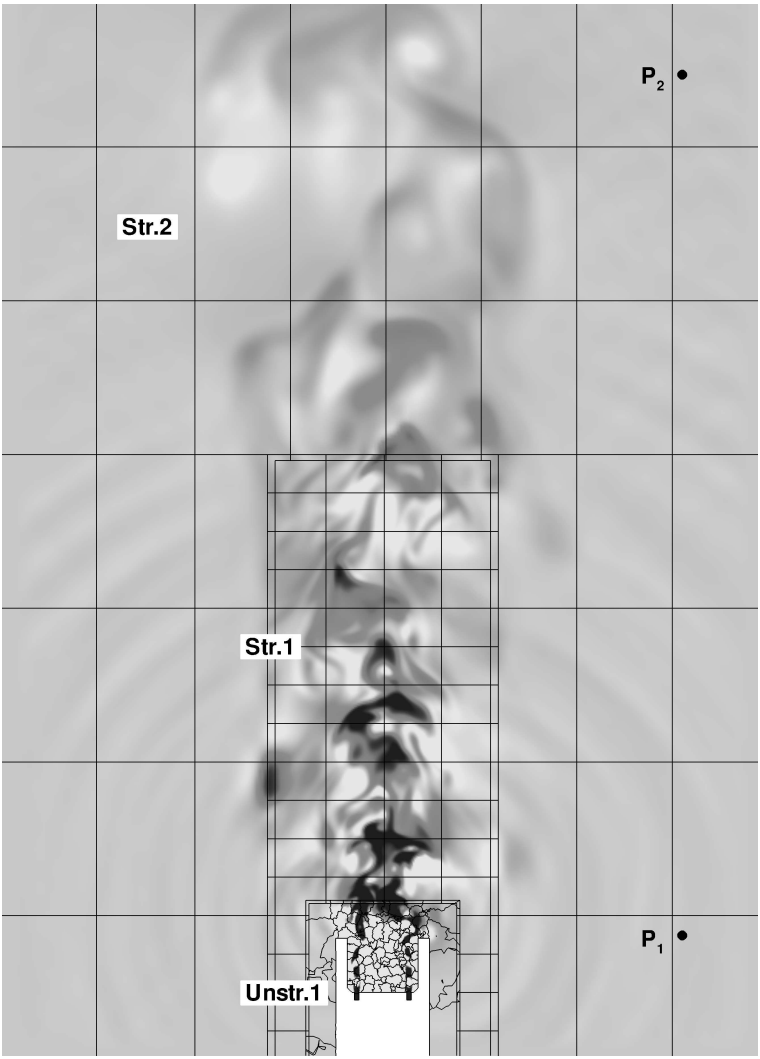


Figure 3.16: Contour plot of ρ in the total simulation area at $t = 0.8ms$. A slice in the xz-plane at $y = 0.0m$ is shown. The lines depict the edges of the MPI partitions in the subdomains.

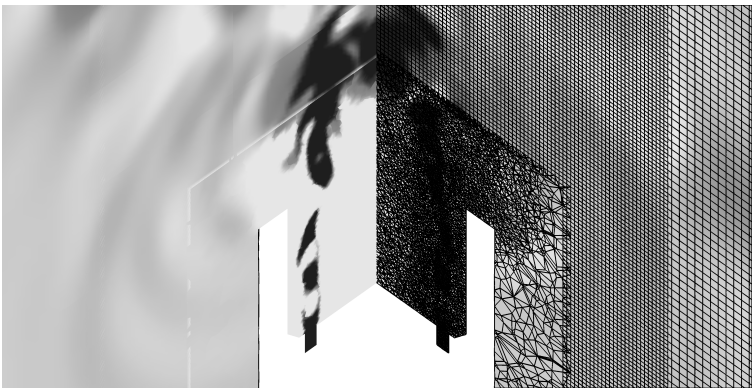


Figure 3.17: Close-up of the nozzle flow and the different grids.

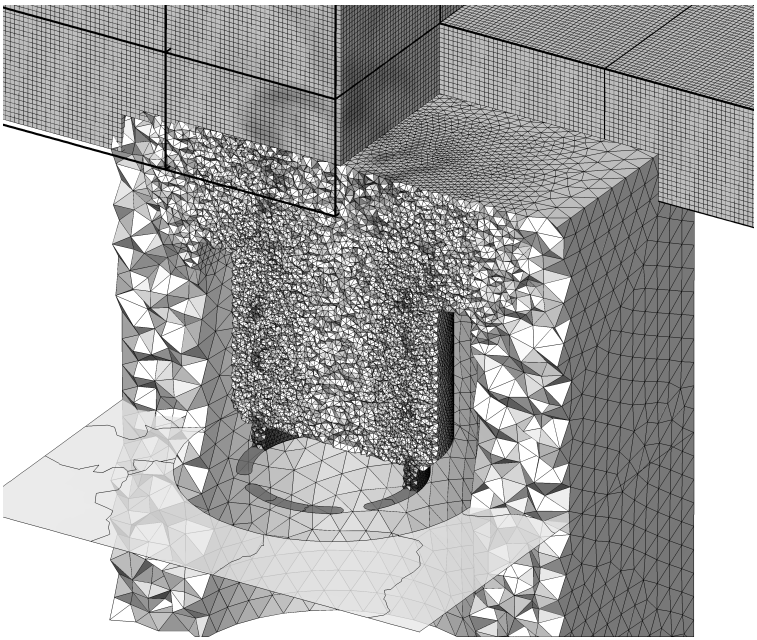


Figure 3.18: A cut-out of the nozzle geometry. On top, MPI partitions of the fine structured core region are shown.

Figure 3.19: Contour plot of the emitted sound waves (ρ depicted).

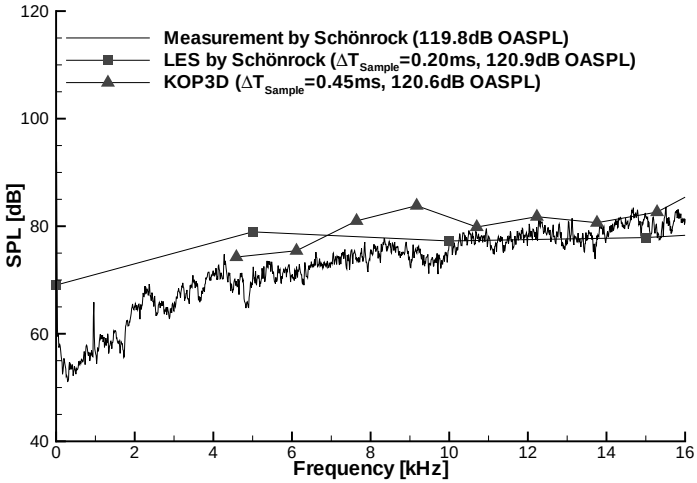


Figure 3.20: SPL spectra at $P_1 = (0.01697m, 0.0m, 0.004m)$.

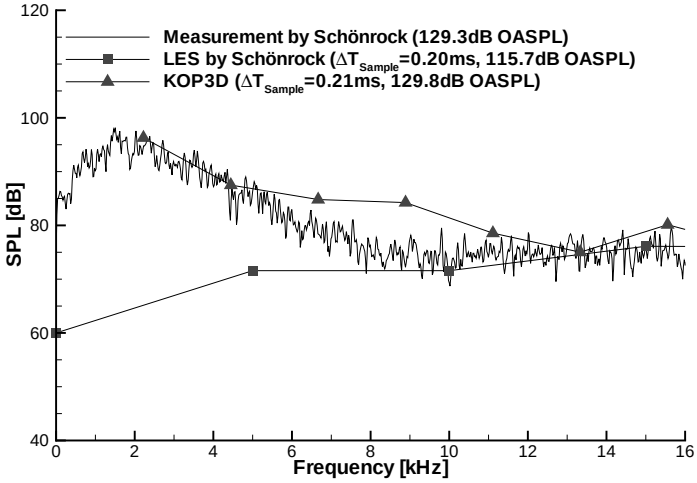


Figure 3.21: SPL spectra at $P_2 = (0.01697m, 0.0m, 0.06m)$.

4 Conclusions

A domain decomposition approach for the direct simulation of aeroacoustic problems has been developed and implemented in a coherent code framework. The basic concept is to combine different numerical methods, equations, grids and time steps for a greater efficiency. It is of secondary importance for the domain decomposition philosophy how evolved and thus complex the single solvers in the subdomains are, i.e. if they feature local time stepping algorithms, dynamic grid adaptation and so on. In fact, an increase in computational performance can already be realized by exploiting the basic properties of different schemes.

The coupling approach connects different classes of methods (DG, FV, FD) on structured and unstructured grids for the solution of the Navier-Stokes, Euler and linearized Euler equations. The optimal time step can be chosen in each subdomain by employing a subcycling technique. The numerical flux at the interfaces may be discontinuous, while reflections at the coupling boundaries remain very small if the grids are connected in a reasonable fashion. High order convergence rates are maintained globally if the order of the interpolation procedure is at least equal to the order of accuracy in the subdomains. No significant spurious effects arise from high frequency waves traveling onto coarse grids which cannot resolve their wavelength. Conveniently, these waves are filtered out automatically. Various numerical examples demonstrated, that the coupling approach is especially suitable for far field computations with embedded complex geometries.

A certain basic knowledge of the occurring flow and the acoustic phenomena is the premise for the decomposition method. This means, that a preselection of appropriate parameters (solvers, equations, grid spacing, time step ratios, interpolation and domain order, etc.) is required for every problem that ought to be solved. It must be kept in mind that the chosen parameters should harmonize and match the desired accuracy and resolution of the phenomena. Otherwise, reflections will occur. Some thoughts need to be put into the manual arrangement of the subdomains. Because the approach is basically non-overlapping, the amount of exchanged coupling data scales mainly with the surface of the domain boundaries, which should therefore be as compact as possible. This re-

sembles the requirements for grid partitioning for parallel computations. Note that due to the non-overlapping nature of the decomposition and in contrast to traditional overset grid methods, the order of interpolation is not an important factor during the *generation* of the subdomains. This facilitates the subsequent adaptation of the domain order. The only limitation here is, that ghost elements must not reach into a solid geometry. In order to ease the pre-processing, an auto-adaptive decomposition algorithm based on an error-estimator that dynamically locates and distributes coupling boundaries is imaginable, yet intricate. Because the calculation domain is composed of domains with very different properties, the implementation of an effective load balancing becomes more complex for parallel computations than for a stand-alone code. On the other hand, one is rewarded with the option to distribute the different domains to separate and for each method optimal computer architectures.
Changing from nonlinear to linearized equations is permitted only, when all nonlinear effects have vanished. This implies, that no discontinuities are permitted to cross such boundaries. Strictly speaking, this is also the case for an interface between nonlinear domains, as the current implementation uses central stencils whenever possible for the interpolation. Nevertheless, the supersonic free jet example proved, that the coupling method remains robust even in scenarios where strongly nonlinear flow features pass the domain interfaces. It is emphasized, that no special grid-to-grid filtering to remove oscillations was applied in this case. However, in order to adapt the interpolation procedure better to highly nonlinear effects, either a filtering operator or a WENO-like method could be added. The latter may be directly extracted from the STE-FV and Rec-FV schemes.
The available Navier-Stokes solvers in the *KOP3D* framework are only designed for DNS computations, as no turbulence modeling is included so far. The coupling with RANS (Reynolds-averaged Navier-Stokes) solvers or, given the unsteady nature of aeroacoustic problems, LES (Large Eddy Simulation) methods raises new questions and makes higher demands on a decomposition algorithm, as additional turbulence parameters must be exchanged where required. Finally, the coupling framework is by nature not restricted to CFD and CAA. It would be interesting to explore, how to fit in other equations and schemes, such as the magnetohydrodynamic (MHD) and Maxwell equations, the particle in cell (PIC) method, etc., enabling simulations in a wide range of disciplines.

A Convergence Tables

$\mathcal{O}2$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.04737E-02	2.29786E-02	3.49250E-02	–	–	–
#20	3.19661E-03	3.56202E-03	5.82578E-03	2.68	2.69	2.58
#30	1.10493E-03	1.23906E-03	2.03622E-03	2.62	2.60	2.59
#40	5.40552E-04	6.08576E-04	1.00027E-03	2.49	2.47	2.47
#50	3.14933E-04	3.54696E-04	5.84191E-04	2.42	2.42	2.41
$\mathcal{O}3$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.61282E-02	1.80764E-02	2.77849E-02	–	–	–
#20	2.11114E-03	2.33846E-03	3.85006E-03	2.93	2.95	2.85
#30	6.23669E-04	6.94327E-04	1.15041E-03	3.01	2.99	2.98
#40	2.63210E-04	2.92626E-04	4.84141E-04	3.00	3.00	3.01
#50	1.34543E-04	1.49524E-04	2.47503E-04	3.01	3.01	3.01
$\mathcal{O}4$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.91198E-03	2.25226E-03	4.19146E-03	–	–	–
#20	8.69657E-05	1.04884E-04	1.82361E-04	4.46	4.42	4.52
#30	1.48744E-05	1.80055E-05	3.04336E-05	4.36	4.35	4.42
#40	4.38057E-06	5.32006E-06	8.83494E-06	4.25	4.24	4.30
#50	1.73429E-06	2.09807E-06	3.48888E-06	4.15	4.17	4.16

Table A.1: STE-FV, 2D, central reconstruction: Convergence rates and error norms.

$\mathcal{O}5$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.05273E-03	2.30478E-03	4.14604E-03	–	–	–
#20	7.34918E-05	8.76462E-05	1.73181E-04	4.80	4.72	4.58
#30	9.95859E-06	1.19068E-05	2.36087E-05	4.93	4.92	4.91
#40	2.38565E-06	2.85105E-06	5.61898E-06	4.97	4.97	4.99
#50	7.83453E-07	9.37029E-07	1.85745E-06	4.99	4.99	4.96
$\mathcal{O}6$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	3.61850E-04	4.14211E-04	7.37854E-04	–	–	–
#20	4.38007E-06	4.98573E-06	8.51183E-06	6.37	6.38	6.44
#30	2.99752E-07	3.43619E-07	5.88158E-07	6.61	6.60	6.59
#40	4.55975E-08	5.25603E-08	9.18576E-08	6.55	6.53	6.45
#50	1.08539E-08	1.25344E-08	2.21599E-08	6.43	6.42	6.37
$\mathcal{O}7$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	3.69265E-04	4.34302E-04	7.25702E-04	–	–	–
#20	4.37934E-06	4.95399E-06	8.41545E-06	6.40	6.45	6.43
#30	2.73041E-07	3.08513E-07	5.25292E-07	6.84	6.85	6.84
#40	3.74378E-08	4.20525E-08	7.10669E-08	6.91	6.93	6.95
#50	7.90692E-09	8.89710E-09	1.50083E-08	6.97	6.96	6.97
$\mathcal{O}8$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.14048E-04	1.29070E-04	1.86552E-04	–	–	–
#20	3.69120E-07	4.07561E-07	6.37447E-07	8.27	8.31	8.19
#30	1.08041E-08	1.20405E-08	1.89161E-08	8.71	8.69	8.68
#40	8.82524E-10	9.84998E-10	1.56025E-09	8.71	8.70	8.67
#50	1.28076E-10	1.43008E-10	2.28300E-10	8.65	8.65	8.61

Table A.2: STE-FV, 2D, central reconstruction: Convergence rates and error norms (continuation).

$\mathcal{O}2$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.05430E-02	2.30661E-02	3.50576E-02	–	–	–
#20	3.19998E-03	3.56533E-03	5.83050E-03	2.68	2.69	2.59
#30	1.10554E-03	1.23948E-03	2.03586E-03	2.62	2.61	2.60
#40	5.40721E-04	6.08669E-04	1.00006E-03	2.49	2.47	2.47
#50	3.14988E-04	3.54724E-04	5.84136E-04	2.42	2.42	2.41
$\mathcal{O}3$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.61190E-02	1.80658E-02	2.77692E-02	–	–	–
#20	2.10978E-03	2.33695E-03	3.84759E-03	2.93	2.95	2.85
#30	6.23261E-04	6.93872E-04	1.14966E-03	3.01	2.99	2.98
#40	2.63036E-04	2.92433E-04	4.83819E-04	3.00	3.00	3.01
#50	1.34454E-04	1.49426E-04	2.47339E-04	3.01	3.01	3.01
$\mathcal{O}4$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.91476E-03	2.25526E-03	4.20008E-03	–	–	–
#20	8.69411E-05	1.04836E-04	1.82354E-04	4.46	4.43	4.53
#30	1.48641E-05	1.79904E-05	3.04074E-05	4.36	4.35	4.42
#40	4.37658E-06	5.31481E-06	8.82594E-06	4.25	4.24	4.30
#50	1.73255E-06	2.09585E-06	3.48511E-06	4.15	4.17	4.16
$\mathcal{O}5$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.05219E-03	2.30406E-03	4.14394E-03	–	–	–
#20	7.34859E-05	8.76322E-05	1.73162E-04	4.80	4.72	4.58
#30	9.95720E-06	1.19055E-05	2.36058E-05	4.93	4.92	4.91
#40	2.38541E-06	2.85077E-06	5.61826E-06	4.97	4.97	4.99
#50	7.83374E-07	9.36941E-07	1.85729E-06	4.99	4.99	4.96
$\mathcal{O}6$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	3.61942E-04	4.14331E-04	7.38920E-04	–	–	–
#20	4.38282E-06	4.99002E-06	8.52943E-06	6.37	6.38	6.44
#30	3.00164E-07	3.44128E-07	5.89108E-07	6.61	6.60	6.59
#40	4.56838E-08	5.26697E-08	9.20674E-08	6.54	6.52	6.45
#50	1.08815E-08	1.25669E-08	2.22203E-08	6.43	6.42	6.37

Table A.3: STE-FV, 2D, WENO reconstruction ($\lambda_C = 10^4$, $\epsilon = 1.0$, $r = 3$): Convergence rates and error norms.

$\mathcal{O}7$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	3.69553E-04	4.34513E-04	7.25768E-04	–	–	–
#20	4.37444E-06	4.94907E-06	8.40296E-06	6.40	6.46	6.43
#30	2.72718E-07	3.08083E-07	5.24510E-07	6.84	6.85	6.84
#40	3.73824E-08	4.19886E-08	7.09645E-08	6.91	6.93	6.95
#50	7.89423E-09	8.88315E-09	1.49822E-08	6.97	6.96	6.97
$\mathcal{O}8$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	1.14103E-04	1.29104E-04	1.86699E-04	–	–	–
#20	3.68132E-07	4.06514E-07	6.35193E-07	8.28	8.31	8.20
#30	1.07602E-08	1.19906E-08	1.88228E-08	8.71	8.69	8.68
#40	8.76811E-10	9.78821E-10	1.54896E-09	8.72	8.71	8.68
#50	1.26978E-10	1.41777E-10	2.26118E-10	8.66	8.66	8.62

Table A.4: STE-FV, 2D, WENO reconstruction ($\lambda_C = 10^4$, $\epsilon = 1.0$, $r = 3$): Convergence rates and error norms (continuation).

$\mathcal{O}2$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.89796E-02	3.19857E-02	4.67351E-02	–	–	–
#20	5.48121E-03	6.13170E-03	8.83717E-03	2.40	2.38	2.40
#30	2.14337E-03	2.38771E-03	3.49701E-03	2.32	2.33	2.29
#40	1.10116E-03	1.23130E-03	1.85071E-03	2.32	2.30	2.21
#50	6.62561E-04	7.41668E-04	1.13465E-03	2.28	2.27	2.19
$\mathcal{O}3$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.06429E-02	2.29550E-02	3.65170E-02	–	–	–
#20	2.15897E-03	2.38665E-03	3.79602E-03	3.26	3.27	3.27
#30	5.80660E-04	6.41770E-04	1.02058E-03	3.24	3.24	3.24
#40	2.30140E-04	2.54239E-04	4.03856E-04	3.22	3.22	3.22
#50	1.13416E-04	1.25452E-04	1.99073E-04	3.17	3.17	3.17
$\mathcal{O}4$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.61263E-03	3.29318E-03	6.05422E-03	–	–	–
#20	1.36035E-04	1.59683E-04	2.99382E-04	4.26	4.37	4.34
#30	2.09447E-05	2.48848E-05	4.70280E-05	4.61	4.58	4.57
#40	5.71588E-06	6.82022E-06	1.29727E-05	4.51	4.50	4.48
#50	2.12462E-06	2.54718E-06	4.84394E-06	4.44	4.41	4.41
$\mathcal{O}5$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.65564E-03	3.36121E-03	5.46644E-03	–	–	–
#20	1.30879E-04	1.53088E-04	2.69909E-04	4.34	4.46	4.34
#30	1.84553E-05	2.12637E-05	3.75036E-05	4.83	4.87	4.87
#40	4.45264E-06	5.11490E-06	8.97910E-06	4.94	4.95	4.97
#50	1.46827E-06	1.68495E-06	2.94278E-06	4.97	4.98	5.00
$\mathcal{O}6$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	7.24230E-04	8.08338E-04	1.34210E-03	–	–	–
#20	1.10397E-05	1.24318E-05	2.02696E-05	6.04	6.02	6.05
#30	9.14009E-07	1.02881E-06	1.70329E-06	6.14	6.15	6.11
#40	1.59141E-07	1.78505E-07	2.94312E-07	6.08	6.09	6.10
#50	4.12626E-08	4.62842E-08	7.58737E-08	6.05	6.05	6.07

Table A.5: STE-FV, 3D, central reconstruction: Convergence rates and error norms.

$\mathcal{O}2$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.99052E-02	3.29686E-02	4.74475E-02			
#20	5.50447E-03	6.17313E-03	8.91725E-03	2.44	2.42	2.41
#30	2.14467E-03	2.39373E-03	3.52853E-03	2.32	2.34	2.29
#40	1.10151E-03	1.23282E-03	1.85982E-03	2.32	2.31	2.23
#50	6.62655E-04	7.42180E-04	1.13789E-03	2.28	2.27	2.20
$\mathcal{O}3$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.06316E-02	2.29416E-02	3.64970E-02			
#20	2.15750E-03	2.38506E-03	3.79328E-03	3.26	3.27	3.27
#30	5.80265E-04	6.41326E-04	1.01986E-03	3.24	3.24	3.24
#40	2.29975E-04	2.54060E-04	4.03555E-04	3.22	3.22	3.22
#50	1.13336E-04	1.25363E-04	1.98919E-04	3.17	3.17	3.17
$\mathcal{O}4$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.59387E-03	3.28205E-03	5.99269E-03			
#20	1.35057E-04	1.58734E-04	2.96892E-04	4.26	4.37	4.34
#30	2.08589E-05	2.47982E-05	4.68616E-05	4.61	4.58	4.55
#40	5.69960E-06	6.80208E-06	1.29518E-05	4.51	4.50	4.47
#50	2.11902E-06	2.54103E-06	4.83567E-06	4.43	4.41	4.42
$\mathcal{O}5$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	2.65496E-03	3.36050E-03	5.46564E-03			
#20	1.30867E-04	1.53072E-04	2.69843E-04	4.34	4.46	4.34
#30	1.84544E-05	2.12621E-05	3.74995E-05	4.83	4.87	4.87
#40	4.45230E-06	5.11454E-06	8.97842E-06	4.94	4.95	4.97
#50	1.46821E-06	1.68483E-06	2.94257E-06	4.97	4.98	5.00
$\mathcal{O}6$	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
#10	7.19342E-04	8.02557E-04	1.33187E-03			
#20	1.10285E-05	1.24217E-05	2.02945E-05	6.03	6.01	6.04
#30	9.15070E-07	1.03004E-06	1.70547E-06	6.14	6.14	6.11
#40	1.59479E-07	1.78878E-07	2.94855E-07	6.07	6.09	6.10
#50	4.13685E-08	4.64023E-08	7.60501E-08	6.05	6.05	6.07

Table A.6: STE-FV, 3D, WENO reconstruction ($\lambda_C = 10^4$, $\epsilon = 1.0$, $r = 3$): Convergence rates and error norms.

#	$N_{\Delta h(\Omega_1/\Omega_2/\Omega_3)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	06/20/08	1/12/24	lin.DG/lin.FV/lin.FD	unstr./str./str.
1	09/30/12	1/12/24	lin.DG/lin.FV/lin.FD	unstr./str./str.
2	12/40/16	1/12/24	lin.DG/lin.FV/lin.FD	unstr./str./str.
3	15/50/20	1/12/24	lin.DG/lin.FV/lin.FD	unstr./str./str.
4	18/60/24	1/12/24	lin.DG/lin.FV/lin.FD	unstr./str./str.

Ω_1	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.18213E-04	1.30709E-04	1.83961E-04	–	–	–
1	2.51319E-05	2.78609E-05	3.94997E-05	3.82	3.81	3.79
2	7.90773E-06	8.79779E-06	1.24388E-05	4.02	4.01	4.02
3	3.22375E-06	3.57748E-06	5.05048E-06	4.02	4.03	4.04
4	1.54138E-06	1.71291E-06	2.41950E-06	4.05	4.04	4.04
Ω_2	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.18213E-04	1.30709E-04	1.83961E-04	–	–	–
1	2.51319E-05	2.78609E-05	3.94997E-05	3.82	3.81	3.79
2	7.90773E-06	8.79779E-06	1.24388E-05	4.02	4.01	4.02
3	3.22375E-06	3.57748E-06	5.05048E-06	4.02	4.03	4.04
4	1.54138E-06	1.71291E-06	2.41950E-06	4.05	4.04	4.04
Ω_3	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.21881E-04	1.33810E-04	1.85046E-04	–	–	–
1	2.62056E-05	2.86817E-05	3.88380E-05	3.79	3.80	3.85
2	8.20807E-06	9.02662E-06	1.23395E-05	4.04	4.02	3.99
3	3.31588E-06	3.65675E-06	5.03636E-06	4.06	4.05	4.02
4	1.58032E-06	1.74552E-06	2.41555E-06	4.06	4.06	4.03

Table A.7: Domain Decomposition, 2D, $\mathcal{O}4$: Convergence rates, error norms, grid properties, subcycles and methods for the planar wave case.

Ω_1	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	7.07886E-04	7.83087E-04	1.13128E-03	–	–	–
1	4.82407E-04	5.35710E-04	7.62069E-04	0.95	0.94	0.97
2	3.51389E-04	3.90166E-04	5.56204E-04	1.10	1.10	1.09
3	2.77576E-04	3.08359E-04	4.37229E-04	1.06	1.05	1.08
4	2.30420E-04	2.56031E-04	3.78102E-04	1.02	1.02	0.80

Ω_2	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	6.92664E-04	7.69368E-04	1.08723E-03	–	–	–
1	4.82993E-04	5.35854E-04	7.61952E-04	0.89	0.89	0.88
2	3.51458E-04	3.89742E-04	5.52335E-04	1.11	1.11	1.12
3	2.77682E-04	3.08516E-04	4.37168E-04	1.06	1.05	1.05
4	2.30467E-04	2.55946E-04	3.61976E-04	1.02	1.02	1.04
Ω_3	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	6.60394E-04	7.27190E-04	1.04540E-03	–	–	–
1	4.75237E-04	5.18690E-04	7.23677E-04	0.81	0.83	0.91
2	3.49740E-04	3.85889E-04	5.56581E-04	1.07	1.03	0.91
3	2.77393E-04	3.06725E-04	4.43620E-04	1.04	1.03	1.02
4	2.31267E-04	2.55185E-04	3.62127E-04	1.00	1.01	1.11

Table A.8: Domain Decomposition, 2D, $\mathcal{O}4$: Convergence rates and error norms for the planar wave case, *without* applying the Cauchy-Kovalevskaja procedure to the coupling cells.

#	$N_{\Delta h(\Omega_1/\Omega_2)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	1/10	1/6	lin.DG/lin.FD	unstr./str.
1	2/20	1/6	lin.DG/lin.FD	unstr./str.
2	3/30	1/6	lin.DG/lin.FD	unstr./str.
3	4/40	1/6	lin.DG/lin.FD	unstr./str.
4	5/50	1/6	lin.DG/lin.FD	unstr./str.

Ω_1	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.00371E-04	1.11256E-04	1.84834E-04	–	–	–
1	1.11161E-06	1.25587E-06	2.31053E-06	6.50	6.47	6.32
2	4.34141E-08	4.83441E-08	9.69821E-08	8.00	8.03	7.82
3	4.35598E-09	4.85757E-09	9.91102E-09	7.99	7.99	7.93
4	7.47196E-10	8.32115E-10	1.62499E-09	7.90	7.91	8.10
Ω_2	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	8.94465E-05	1.02979E-04	1.59922E-04	–	–	–
1	1.16003E-06	1.31963E-06	2.54569E-06	6.27	6.29	5.97
2	4.41855E-08	4.90505E-08	8.21517E-08	8.06	8.12	8.47
3	4.40212E-09	4.88321E-09	7.56169E-09	8.02	8.02	8.29
4	7.50843E-10	8.36011E-10	1.32063E-09	7.93	7.91	7.82

Table A.9: Domain Decomposition, 2D, $\mathcal{O}8$: Convergence rates, error norms, grid properties, subcycles and methods for the planar wave case.

#	$N_{\Delta h(\Omega_1/\Omega_2)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	05/11	1/25	lin.DG/lin.FD	unstr./str.
1	10/22	1/25	lin.DG/lin.FD	unstr./str.
2	15/33	1/25	lin.DG/lin.FD	unstr./str.
3	20/44	1/25	lin.DG/lin.FD	unstr./str.
4	25/55	1/25	lin.DG/lin.FD	unstr./str.

Ω_1	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	3.77289E-06	4.17450E-06	6.10200E-06	–	–	–
1	2.25286E-07	2.50413E-07	3.59185E-07	4.07	4.06	4.09
2	4.37562E-08	4.85673E-08	7.07303E-08	4.04	4.05	4.01
3	1.36943E-08	1.52169E-08	2.19622E-08	4.04	4.03	4.07
4	5.57810E-09	6.19432E-09	8.94730E-09	4.02	4.03	4.02
Ω_2	$L_1(p')$	$L_2(p')$	$L_\infty(p')$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	3.95460E-06	4.35040E-06	5.97122E-06	–	–	–
1	2.32878E-07	2.57918E-07	3.56685E-07	4.09	4.08	4.07
2	4.43935E-08	4.92285E-08	6.87019E-08	4.09	4.08	4.06
3	1.39280E-08	1.54456E-08	2.16409E-08	4.03	4.03	4.02
4	5.65727E-09	6.27489E-09	8.84977E-09	4.04	4.04	4.01

Table A.10: Domain Decomposition, 3D, $\mathcal{O}4$: Convergence rates, error norms, grid properties, subcycles and methods for the planar wave case.

#	$N_{\Delta h(\Omega_1/\Omega_2)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	06/18	[1/14, 1/8]	nonlin.DG/nonlin.FV	unstr./str.
1	09/27	[1/14, 1/8]	nonlin.DG/nonlin.FV	unstr./str.
2	12/36	[1/14, 1/8]	nonlin.DG/nonlin.FV	unstr./str.
3	15/45	[1/14, 1/8]	nonlin.DG/nonlin.FV	unstr./str.
4	18/54	[1/14, 1/8]	nonlin.DG/nonlin.FV	unstr./str.

Ω_1	$L_1(\rho E)$	$L_2(\rho E)$	$L_\infty(\rho E)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.01685E+00	1.96658E-01	1.50482E-01	–	–	–
1	3.42911E-01	6.87289E-02	4.84860E-02	2.68	2.59	2.79
2	1.29988E-01	2.72806E-02	1.90748E-02	3.37	3.21	3.24
3	5.61519E-02	1.18841E-02	8.06800E-03	3.76	3.72	3.86
4	2.75682E-02	5.82477E-03	3.85123E-03	3.90	3.91	4.06

Table A.11: Domain Decomposition, 2D, $\mathcal{O}4$: Convergence rates, error norms, grid properties, subcycles and methods for the Shu vortex case.

#	$N_{\Delta h(\Omega_1/\Omega_2)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	09/15	1/25	nonlin.DG/lin.FD	unstr./str.
1	12/20	1/25	nonlin.DG/lin.FD	unstr./str.
2	15/25	1/25	nonlin.DG/lin.FD	unstr./str.
3	18/30	1/25	nonlin.DG/lin.FD	unstr./str.
4	27/45	1/25	nonlin.DG/lin.FD	unstr./str.

Ω_1	$L_1(\rho)$	$L_2(\rho)$	$L_\infty(\rho)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.10750E-04	2.44962E-04	1.27112E-03	–	–	–
1	5.04587E-05	1.16428E-04	6.55195E-04	2.73	2.59	2.30
2	2.32009E-05	5.61826E-05	3.31358E-04	3.48	3.27	3.06
3	1.14153E-05	2.88341E-05	1.73547E-04	3.89	3.66	3.55
4	2.29637E-06	5.93430E-06	3.60376E-05	3.96	3.90	3.88

Table A.12: Domain Decomposition, 2D, $\mathcal{O}4$: Convergence rates, error norms, grid properties, subcycles and methods for the Gaussian pulse case.

#	$N_{\Delta h(\Omega_1/\Omega_2)}$	$\frac{\Delta t}{\Delta t_{\Omega_1}}$	Methods	Grids
0	05/11	1/25	nonlin.Rec-FV/lin.FD	unstr./str.
1	10/22	1/25	nonlin.Rec-FV/lin.FD	unstr./str.
2	15/33	1/25	nonlin.Rec-FV/lin.FD	unstr./str.
3	20/44	1/25	nonlin.Rec-FV/lin.FD	unstr./str.
4	25/55	1/25	nonlin.Rec-FV/lin.FD	unstr./str.

Ω_1	$L_1(\rho)$	$L_2(\rho)$	$L_\infty(\rho)$	$\mathcal{O}_{L_1}$	$\mathcal{O}_{L_2}$	$\mathcal{O}_{L_\infty}$
0	1.35786E-04	2.86086E-04	3.07690E-03	–	–	–
1	2.19331E-05	7.43748E-05	1.38471E-03	2.63	1.94	1.15
2	4.35259E-06	1.72701E-05	3.26638E-04	3.99	3.60	3.56
3	1.39404E-06	5.54302E-06	1.19470E-04	3.96	3.95	3.50
4	5.83062E-07	2.30490E-06	4.26862E-05	3.91	3.93	4.61

Table A.13: Domain Decomposition, 3D, $\mathcal{O}4$: Convergence rates, error norms, grid properties, subcycles and methods for the Gaussian pulse case.

Bibliography

[1] I. Ali, M. Escobar, M. Kaltenbacher, and S. Becker. Time domain computation of flow induced sound. *Computers & Fluids*, 2007. doi: 10.1016/j.compfluid.2007.02.011.

[2] A. Babucke, M. Kloker, and U. Rist. DNS of a plane mixing layer for the investigation of sound generation mechanisms. *Computers & Fluids*, 37:360–368, 2008. doi:10.1016/j.compfluid.2007.02.002.

[3] A. Babucke, J. Linn, M.J. Kloker, and U. Rist. Direct numerical simulation of shear flow phenomena on parallel vector computers. In *High performance computing on vector systems: Proceedings of the High Performance Computing Center Stuttgart, 2005*. Springer Verlag Berlin, 2006.

[4] D. Balsara and C.W. Shu. Monotonicity preserving weighted essentially non-oscillatory schemes with increasingly high order of accuracy. *Journal of Computational Physics*, 160:405–452, 2000.

[5] J. Becker. *Entwicklung eines effizienten Verfahrens zur Lösung hyperbolischer Differentialgleichungen*. PhD thesis, Universität Freiburg, Institut für Angewandte Mathematik, 2000.

[6] T. Beisel, E. Gabriel, and M. Resch. An extension to MPI for distributed computing on MPPs. pages 75–82, 1997.

[7] J. Blazek. *Computational Fluid Dynamics: Principles and Applications*. Elsevier Science, 2007.

[8] J.C. Butcher. *The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods*. Wiley, 1987.

[9] B. Cockburn, S. Hou, and C.-W. Shu. The Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case. *Mathematics of Computation*, 54:545–581, 1990.

[10] B. Cockburn, G. E. Karniadakis, and C.-W. Shu. *Discontinuous Galerkin Methods*. Lecture Notes in Computational Science and Engineering. Springer, 2000.

[11] B. Cockburn, S.-Y. Lin, and C.-W. Shu. TVB Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation laws III: One dimensional systems. *Journal of Computational Physics*, 84:90–113, 1989.

[12] B. Cockburn and C.-W. Shu. TVB Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation laws II: General framework. *Mathematics of Computation*, 52:411–435, 1989.

[13] B. Cockburn and C.-W. Shu. The Runge-Kutta local projection P1-Discontinuous Galerkin finite element method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, 25:337–361, 1991.

[14] B. Cockburn and C.-W. Shu. The Runge-Kutta local projection Discontinuous Galerkin finite element method for conservation laws V: multidimensional systems. *Journal of Computational Physics*, 141:199–224, 1998.

[15] C.-A. Coclici. *Domain decomposition methods and far-field boundary conditions for two-dimensional compressible flows around airfoils*. PhD thesis, Universität Stuttgart, November 1998.

[16] C.-A. Coclici and W.L. Wendland. Analysis of heterogeneous domain decomposition for compressible viscous flows. *Mathematical Models and Methods in Applied Sciences*, 11:565–599, 2000.

[17] J. Delfs. An overlapped grid technique for high resolution CAA schemes for complex geometries. In *Collection of Technical Papers. Vol. 2 (A01-30800 07-71). AIAA/CEAS Aeroacoustics Conference and Exhibit, 7th, Maastricht, Netherlands*, May 2001. AIAA-2001-2199.

[18] M. Dumbser. *Arbitrary High-Order Schemes for the Solution of Hyperbolic Conservation Laws in Complex Domains*. PhD thesis, Universität Stuttgart, Institut für Aerodynamik und Gasdynamik, 2005. Shaker Verlag, Aachen.

[19] M. Dumbser and M. Käser. Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems. *Journal of Computational Physics*, 221(2):693–723, 2007.

[20] M. Dumbser, M. Käser, V. Titarev, and E.F. Toro. Quadrature-free non-oscillatory finite volume schemes on unstructured meshes for nonlinear hyperbolic systems. *Journal of Computational Physics*, 226(2):715–736, 2007.

[21] M. Dumbser and C.-D. Munz. ADER discontinuous Galerkin schemes for aeroacoustics. *Comptes Rendus Mécanique*, 333:683–687, 2005.

[22] M. Dumbser and C.-D. Munz. Arbitrary high order discontinuous Galerkin schemes. In S. Cordier, T. Goudon, M. Gutnic, and E. Sonnendrücker, editors, *Numerical Methods for Hyperbolic and Kinetic Problems*, IRMA Series in Mathematics and Theoretical Physics, pages 295–333. EMS Publishing House, 2005.

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

[24] M. Dumbser and C.-D. Munz. On source terms and boundary conditions using arbitrary high order discontinuous Galerkin schemes. *Applied Mathematics and Computer Science*, 17(3):297–310, 2007.

[25] M. Dumbser, T. Schwartzkopff, and C.-D. Munz. Arbitrary high order finite volume schemes for linear wave propagation. In *Notes on Numerical Fluid Mechanics and Multidisciplinary Design (NNFM)*. Springer Verlag, 2005.

[26] R.W. Dyson. Technique for very high order nonlinear simulation and validation. Technical Report TM-2001-210985, NASA, 2001.

[27] M. Escobar, I. Ali, M. Kaltenbacher, and S. Becker. Investigation of the acoustic sources and their propagation for a FV/FE coupled computation using different grids. In *Proceedings of the 11th AIAA/CEAS Aeroacoustics Conference*, Monterey, California, USA, May 2005. AIAA-2005-3017.

[28] R. Ewert and W. Schröder. Acoustic perturbation equation based on flow decomposition via source filtering. *Journal of Computational Physics*, 188:365–398, 2003.

[29] F. Farassat and M.K. Myers. Extension of Kirchoff formula to radiation from moving surfaces. *Journal of Sound Vibration*, 123:451–460, 1988.

[30] J.E. Ffowcs Williams and D.L. Hawkins. Sound generation by turbulence and surfaces in arbitrary motion. *Philosophical Transactions of the Royal Society*, 264:321–342, 1969.

[31] B. Flemisch, M. Kaltenbacher, and B.I. Wohlmuth. Elasto-acoustic and acoustic-acoustic coupling on non-matching grids. *International Journal for Numerical Methods in Engineering*, 67(13):1791–1810, 2006.

[32] Fraunhofer-Institute SCAI. MpCCI - multidisciplinary simulations through code coupling. http://www.mpcci.de.

[33] J.B. Freund, S.K. Lele, and P. Moin. Matching of near/far-field equation sets for direct computation of aerodynamic sound. In *15th AIAA Aeroacoustics Conference*, October 1993. AIAA-1993-4326.

[34] J.B. Freund, S.K. Lele, and P. Moin. Direct numerical simulation of a mach 1.92 turbulent jet and its sound field. *AIAA Journal*, 38(11):2023–2031, 2000.

[35] G. Gassner, F. Lörcher, and C.-D. Munz. A contribution to the construction of diffusion fluxes for finite volume and discontinuous Galerkin schemes. *Journal of Computational Physics*, 224(2):1049–1063, 2007.

[36] G. Gassner, F. Lörcher, and C.-D. Munz. A discontinuous Galerkin scheme based on a space-time expansion. I. inviscid compressible flow in one space dimension. *Journal of Scientific Computing*, 32(2):175–199, 2007. doi:10.1007/s10915-007-9128-x.

[37] 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. *Journal of Scientific Computing*, 34(3):260–286, 2008.

[38] G. Gassner, F. Lörcher, C.-D. Munz, and J. Hesthaven. Polymorphic nodal elements and their application in discontinuous Galerkin methods. *submitted to Journal of Computational Physics*, 2008.

[39] S.K. Godunov. Finite difference methods for the computation of discontinuous solutions of the equations of fluid dynamics. *Mat. Sb.*, 47:271–306, 1959.

[40] S. Gottlieb and C.-W. Shu. Total variation diminishing Runge-Kutta schemes. ICASE 96-50, NASA Langley Research Center, Hampton, USA, 1996.

[41] A. Harten, B. Engquist, S. Osher, and S.R. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes III. *Journal of Computational Physics*, 71:231–303, 1987.

[42] A. Harten, P.D. Lax, and B. van Leer. On upstream differencing and Godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, 1983.

[43] W. Henshaw and K. Chand. Overture. Lawrence Livermore National Laboratories. https://computation.llnl.gov/casc/Overture.

[44] J. S. Hesthaven and T. Warburton. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications.* Springer Verlag, New York, 2008.

[45] C. Hirsch. *Numerical Computation of Internal and External Flows Vol I: Fundamentals of Numerical Discretisation.* Wiley, 1988.

[46] C. Hirsch. *Numerical Computation of Internal and External Flows Vol II: Computational Methods for Inviscid and Viscous Flow.* Wiley, 1988.

[47] Paul Houston, Christoph Schwab, and Endre Sueli. Discontinuous hp-finite element methods for advection-diffusion-reaction problems. *SIAM Journal of Numerical Analysis*, 39(6):2133–2163, 2001.

[48] C. Hu and C.W. Shu. Weighted essentially non-oscillatory schemes on triangular meshes. *Journal of Computational Physics*, 150:97–127, 1999.

[49] O. Inoue and N. Hatakeyama. Sound generation by a two-dimensional circular cylinder in a uniform flow. *Journal of Fluid Mechanics*, 471:285–314, 2002.

[50] M.J. Ivings, D.M. Causon, and E.F. Toro. On Hybrid High Resolution Upwind Methods for Multicomponent Flows. *ZAMM Zeitschrift für Angewandte Mathematik und Mechanik*, 77:645–668, 1997.

[51] G.-S. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, pages 202–228, 1996.

[52] M. Kaltenbacher, M. Escobar, G. Link, I. Ali, S. Becker, and F. Schäfer. Computational aeroacoustics using MpCCI as coupling interface between fluid mechanics and acoustics. *6th MpCCI User Forum*, pages 52–63, 2005.

[53] R. Keller and M. Mueller. PACX-MPI project homepage. http://www.hlrs.de/organization/amt/projects/pacx-mpi.

[54] C.M. Klaij, J. J. W. van der Vegt, and H. van der Ven. Spacetime discontinuous Galerkin method for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 217(2):589–611, 2006.

[55] H. Klimach, S. Roller, and C.-D. Munz. Heterogeneous parallelism of aero-acoustic applications using PACX-MPI. In *Proceedings of 7th TeraFlop Workshop*, Tohoku University, Sendai, Japan, November 2007.

[56] H. Klimach, S. Roller, J. Utzmann, and C.-D. Munz. Parallel coupling of heterogeneous domains with KOP3D using PACX-MPI. In *Proceedings of ParCFD 2007*, Antalya, Turkey, Mai 2007.

[57] M. Kloker. *Lösungseigenschaften von Finite-Differenzen-Verfahren - Diagrammkatalogreihe*. Institut für Aerodynamik und Gasdynamik der Universität Stuttgart, 1997. Katalog 1-3.

[58] M. J. Kloker. A robust high-resolution split-type compact FD scheme for spatial DNS of boundary-layer transition. *Applied Scientific Research*, 59:353–377, 1998.

[59] F. Krey. Coupling of FASTEST-3D and Hydsol for fluid-acoustic interaction: Implementation and simulation. Studienarbeit, 2007. Lehrstuhl für Strömungsmechanik, Friedrich-Alexander Universität Erlangen-Nürnberg.

[60] P.D. Lax. Weak solutions of nonlinear hyperbolic equations and their numerical approximation. *Communications on Pure and Applied Mathematics*, 7:159–193, 1954.

[61] P.D. Lax and B. Wendroff. Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237, 1960.

[62] P.D. Lax and B. Wendroff. Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237, 1960.

[63] S.K. Lele. Direct computation of aerodynamic noise. In M. Y. Hussaini J. C. Hardin, editor, *Computational Aeroacoustics*, pages 325–334. Springer Verlag, 1993.

[64] M.J. Lighthill. On sound generated aerodynamically - 1. General theory. *Proceedings of the Royal Society of London*, 211(A1107):564–587, 1952.

[65] M.J. Lighthill. On sound generated aerodynamically - 2. Turbulence as a source of sound. *Proceedings of the Royal Society of London*, 222(A1148):1–32, 1954.

[66] X.D. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115:200–212, 1994.

[67] F. Lörcher. Arbitrary high order accurate time integration schemes for linear problems. In *Proceedings of the ECCOMAS CFD 2006*, Egmond aan Zee, The Netherlands, September 2006.

[68] F. Lörcher, G. Gassner, and C.-D. Munz. An explicit discontinuous Galerkin scheme with local time-stepping for general unsteady diffusion equations. *Journal of Computational Physics*, 227(11):5649–5670, 2008.

[69] F. Lörcher and C.-D. Munz. Lax-Wendroff-type schemes of arbitrary order in several space dimensions. *IMA Journal of Numerical Analysis*, 27:593–615, 2007. doi:10.1093/imanum/drl031.

[70] E. Manoha, S. Redonnet, R. Guenanff, and M. Terracol. Category 2: Complex geometry. In *Proceedings of Fourth Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, September 2004. NASA/CP-2004-212954.

[71] R.L. Meakin. Composite overset structured grids. In J.F. Thompson, B.K. Soni, and N.P. Weatherill, editors, *Handbook of Grid Generation*. CRC Press, 1998.

[72] P.J. Morris. Scattering of sound from a spatially distributed, spherically symmetric source by a sphere. *Journal of the Acoustical Society of America*, 98:3536–3539, 1995.

[73] P.J. Morris. Scattering of sound by a sphere: Category 1, problems 3 and 4. In *Proceedings of Second Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, October 1997. NASA/CP-1997-3352.

[74] MPI-Forum. MPI a message-passing interface standard. Technical Report CS-94-230, University of Tennessee, 1995.

[75] B. Müller. High order numerical simulation of aeolian tones. *Computers & Fluids*, 2007. doi: 10.1016/j.compfluid.2007.02.008.

[76] C.-D. Munz and T. Westermann. *Numerische Behandlung gewöhnlicher und partieller Differenzialgleichungen*. Springer, 2006.

[77] J. Nordström and J. Gong. A stable hybrid method for hyperbolic problems. *Journal of Computational Physics*, 212:436–453, 2006.

[78] J. Postel. RFC-793: Transmission Control Protocol, 1981.

[79] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in Fortran77*, volume 1. Cambridge University Press, 2 edition, 1996.

[80] J. Qiu. A numerical comparison of the Lax-Wendroff discontinuous Galerkin method based on different numerical fluxes. *Journal of Scientific Computing*. DOI: 10.1007/s10915-006-9109-5.

[81] J. Qiu, M. Dumbser, and C.W. Shu. The discontinuous Galerkin method with Lax-Wendroff type time discretizations. *Computer Methods in Applied Mechanics and Engineering*, 194:4528–4543, 2005.

[82] J. Qiu, B. C. Khoo, and C.-W. Shu. A numerical study for the performance of the Runge-Kutta discontinuous Galerkin method based on different numerical fluxes. *Journal of Computational Physics*, 212:540 – 565, 2006.

[83] J. Qiu and C.-W. Shu. Finite difference WENO schemes with Lax-Wendroff type time discretization. *SIAM J. Sci. Comput.*, 24(6):2185–2198, 2003.

[84] A. Quarteroni and A. Valli. *Domain Decomposition methods for partial differential equations*. Numerical Mathematics and scientific computation. Oxford Science Publications, 1999.

[85] P.L. Roe. Approximate Riemann solvers, parameters, vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[86] A. Roshko. On the development of turbulent wakes from vortex streets. Technical Report NACA Report 1191, Deutsche Forschungsanstalt für Luft- und Raumfahrt e.V., 1954.

[87] O. Schönrock. Aeroacoustic simulation using SAS-SST turbulence model in ANSYS CFX. In *Proceedings of Int. Conf. on Jets, Wakes and Separated Flows, ICJWSF-2008*, Technical University of Berlin, Germany, September 2008.

[88] T. Schwartzkopff. *Finite-Volumen Verfahren hoher Ordnung und heterogene Gebietszerlegung für die numerische Aeroakustik*. PhD thesis, Universität Stuttgart, Institut für Aerodynamik und Gasdynamik, 2005.

[89] T. Schwartzkopff, M. Dumbser, and C.-D. Munz. CAA using domain decomposition and high order methods on structured and unstructured meshes. In *10th AIAA/CEAS Aeroacoustic Conference*, Manchester, GB, May 2004. AIAA-2004-2964.

[90] T. Schwartzkopff, M. Dumbser, and C.-D. Munz. Fast high order ADER schemes for linear hyperbolic equations. *Journal of Computational Physics*, 197:532–539, 2004.

[91] T. Schwartzkopff, C.-D. Munz, and E.F. Toro. ADER: A high order approach for linear hyperbolic systems in 2D. *Journal of Scientific Computing*, 17(1-4):231–240, 2002.

[92] T. Schwartzkopff, C.-D. Munz, E.F. Toro, and R.C. Millington. ADER-2D: A very high-order approach for linear hyperbolic systems. In *Proceedings of ECCOMAS CFD Conference 2001*, September 2001.

[93] T. Schwartzkopff, C.-D. Munz, E.F. Toro, and R.C. Millington. The ADER approach in 2D. In T. Sonar, editor, *Discrete Modelling and Discrete Algorithms on Continuum Mechanics*, pages 207–216, Berlin, 2001. Logos Verlag.

[94] J.H. Seo and Y.J. Moon. Perturbed compressible equations for aeroacoustic noise prediction at low mach numbers. *AIAA Journal*, 43(8):1716–1724, 2005.

[95] J.H. Seo and Y.J. Moon. Linearized perturbed compressible equations for low mach number aeroacoustics. *Journal of Computational Physics*, 218:702–719, 2006.

[96] H. Shen and C.K.W. Tam. Three-dimensional numerical simulation of the jet screech phenomenon. *AIAA Journal*, 40(1):33–41, 2001.

[97] S.E. Sherer. Acoustic scattering from multiple circular cylinders: Category 2, problems 1 and 2, analytic solution. In *Proceedings of Fourth Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, September 2004. NASA/CP-2004-212954.

[98] S.E. Sherer. Multi-geometry scattering problem. In *Proceedings of Fourth Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, September 2004. NASA/CP-2004-212954.

[99] S.E. Sherer and J.N. Scott. High-order compact finite-difference methods on general overset grids. *Journal of Computational Physics*, 210:459–496, 2005.

[100] S.E. Sherer and M.R. Visbal. High-order overset-grid simulations of acoustic scattering from multiple cylinders. In *Proceedings of Fourth Computational Aeroacoustics (CAA) Workshop on Benchmark Problems*, September 2004. NASA/CP-2004-212954.

[101] C.-W. Shu. Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws. *NASA/CR-97-206253 ICASE Report No.97-65*, November 1997.

[102] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock capturing schemes II. *Journal of Computational Physics*, 83:32–78, 1989.

[103] J. Steger, F.C. Dougherty, and J. Benek. A chimera grid scheme. In K.N. Ghia and U. Ghia, editors, *Advances in Grid Generation*, volume 5, pages 59–69. ASME FED, 1983.

[104] A.H. Stroud. *Approximate Calculation of Multiple Integrals*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1971.

[105] C.K.W. Tam and F. Hu. An optimized multi-dimensional interpolation scheme for computational aeroacoustics applications using overset grids. In *Proceedings of the 10th AIAA/CEAS Aeroacoustics Conference*, Manchester, Great Britain, May 2004. AIAA-2004-2812.

[106] C.K.W. Tam and K.A. Kurbatskii. Multi-size-mesh multi-time-step dispersion-relation-preserving scheme for multiple-scales aeroacoustic problems. *International Journal of Computational Fluid Dynamics*, 17:119–132, 2003.

[107] C.K.W. Tam, J.M. Seiner, and J.C. Yu. Proposed relationship between broadband shock associated noise and screech tones. *Journal of Sound and Vibration*, pages 110–309, 1986.

[108] C.K.W. Tam and J.C. Webb. Dispersion-relation-preserving finite difference schemes for computational acoustics. *Journal of Computational Physics*, 107(2):262–281, 1993.

[109] V.A. Titarev and E.F. Toro. ADER schemes for three-dimensional nonlinear hyperbolic systems. *Journal of Computational Physics*, 204:715–736, 2005.

[110] E.F. Toro. Viscous Flux Limiters. In J.B. Vos, A. Rizzi, and I.L. Ryhming, editors, *Notes on Numerical Fluid Mechanics*, volume 35, 1991.

[111] E.F. Toro. Primitive, conservative and adaptive schemes for hyperbolic conservation laws. In E.F. Toro and J.F. Clarke, editors, *Numerical Methods for Wave Progagation*, pages 323–385. Kluwer Academic Publishers, 1998.

[112] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer, second edition, 1999.

[113] E.F. Toro and R.C. Millington. ADER: High-order non-oscillatory advection schemes. In *Proceedings of the 8th International Conference on Nonlinear Hyperbolic Problems*, February 2000. Preprint.

[114] E.F. Toro, R.C. Millington, and L.A.M Nejad. Towards very high order Godunov schemes. In E.F. Toro, editor, *Godunov Methods. Theory and Applications*, pages 907–940. Kluwer/Plenum Academic Publishers, 2001.

[115] E.F. Toro and V.A. Titarev. Solution of the generalized Riemann problem for advection-reaction equations. *Proceedings of the Royal Society of London*, pages 271–281, 2002.

[116] E.F. Toro and V.A. Titarev. ADER schemes for scalar hyperbolic conservation laws with source terms in three space dimensions. *Journal of Computational Physics*, 202:196–215, 2005.

[117] E.F. Toro and V.A. Titarev. TVD fluxes for the high-order ADER schemes. *Journal of Scientific Computing*, 24(3):285–309, 2005.

[118] E.F. Toro and V.A. Titarev. MUSTA fluxes for systems of conservation laws. *Journal of Computational Physics*, 216:403–429, 2006.

[119] J. Utzmann, T. Schwartzkopff, M. Dumbser, and C.-D. Munz. Heterogeneous domain decomposition for computational aeroacoustics. *AIAA Journal*, 44(10):2231–2250, October 2006.

[120] S. v. Kowalevsky. *Zur Theorie der Partiellen Differentialgleichungen*. Dissertation, Philosophische Facultät zu Göttingen, 1874. available online: http://www.digizeitschriften.de.

[121] J. J. W. van der Vegt and H. van der Ven. Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows: I. General formulation. *Journal of Computational Physics*, 182(2):546–585, 2002.

[122] B. van Leer. Towards the ultimate conservative difference scheme V: A second order sequel to Godunov's method. *Journal of Computational Physics*, 32:101–136, 1979.

[123] B. van Leer and S. Nomura. Discontinuous Galerkin for diffusion. In *17th AIAA Computational Fluid Dynamics Conference*, AIAA-2005-5108, June 6-9 2005.

[124] H. Vogel. *Gerthsen Physik*. Springer, 19th edition, 1997.

[125] P. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *Journal of Computational Physics*, 54:115–173, 1984.

List of Tables

List of Figures

Lebenslauf

Zur Person

Geburtsdatum: 17.11.1977
Geburtsort: Bamberg
Familienstand: Ledig

Beruflicher Werdegang

2004 – 2008 Wissenschaftlicher Mitarbeiter am Institut für Aerodynamik und Gasdynamik der Universität Suttgart
Dissertation: *A Domain Decomposition Method for the Efficient Direct Simulation of Aeroacoustic Problems*

1998 – 2004 Studium der Luft- und Raumfahrttechnik, Universität Stuttgart
Abschluss: Diplomingenieur der Luft- und Raumfahrttechnik
Diplomarbeit im Vertiefungsfach Strömungslehre: *The Construction of High-Order Residual-Distribution Schemes and their Comparison to DG-Schemes*, angefertigt an der University of Michigan, Ann Arbor, USA
Studienarbeit im Vertiefungsfach Raumfahrtsysteme: *Simulation der Plasma-Umströmung eines Massenspektrometers im Plasmawindkanal mittels der DSMC-Methode*

1997 – 1998 Zivildienst in der Lebenshilfe Bamberg

Schulbildung

1988 – 1997 Ehrenbürg Gymnasium Forchheim, Bayern
Abschluss: Allgemeine Hochschulreife
1984 – 1988 Grundschule Hallerndorf

Stuttgart, Oktober 2008 Jens Utzmann