SoftwareX 9 (2019) 49–54

Contents lists available at ScienceDirect

SoftwareX

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

ExWave: A high performance discontinuous Galerkin solver for the acoustic wave equation

S. Schoeder *, W.A. Wall, M. Kronbichler

Institute for Computational Mechanics, Technical University of Munich, Germany

ARTICLE INFO

Article history:
Received 17 September 2018
Received in revised form 2 January 2019
Accepted 2 January 2019

MSC:
65M60
65Y20
68N19

Keywords:
Matrix-free methods
Discontinuous Galerkin methods
Acoustic wave equation

ABSTRACT

A high performance implementation of a discontinuous Galerkin discretization with explicit Runge–Kutta and arbitrary derivative (ADER) time integration schemes is presented to solve the acoustic wave equation. For ADER, both a global and a local time stepping variant is supplied. The implementation is based on the matrix-free framework of the `deal.II` finite element library providing efficient evaluation routines for quadrilaterals and hexahedra. The implementation is generic and its applicability is demonstrated for academic examples as well as real world problems like urban acoustics. We present the physical and numerical problem description, the general code structure, and the design principles.

Code metadata

Current code version	v1.0
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2018_172
Legal Code License	LGPL v.2.1
Code versioning system used	Git
Software code languages, tools, and services used	C++, MPI
Compilation requirements, operating environments & dependencies	cmake, ctest, `deal.II` version 9.0/9.1, p4est
If available Link to developer documentation/manual	*github.com/kronbichler/exwave*
Support email for questions	kronbichler@lnm.mw.tum.de

1. Motivation and significance

The acoustic wave equation is used in a wide range of applications. Amongst others it allows to predict room acoustics for construction projects or to optimize urban planning and city design in terms of noise. Countless numerical tools exist to solve the acoustic wave equation each with its advantages and disadvantages. Most challenging is the accurate simulation of high frequency waves because they require high spatial and temporal resolution. One group of solution techniques circumventing the difficulties induced by high frequencies and applicable to room acoustics are geometric methods based on ray-tracing, which however are not sufficiently accurate for low-frequencies and diffraction [1]. Another approach is to assume that acoustic waves propagate diffusely in rooms after a sufficient number of reflections and a diffusion equation model is solved (for example, see [2]).

The prediction of acoustics over a wide frequency range requires to solve the acoustic wave equation either with purely numerical schemes with a sufficiently fine discretization or semi-analytic schemes relying partly on fundamental solutions of the wave equation. Straightforward finite difference time domain (FDTD) methods are widely used but due to limited computational resources only applied to lower frequencies [3]. More recently, adaptive rectangular decomposition has been proposed, which is a domain decomposition technique relying on the analytic solution of the wave equation in rectangles and additional interface handling [4,5]. They are currently limited to homogeneous sound speed distributions. Besides finite difference schemes, also conventional finite elements, mixed elements, spectral elements, discontinuous Galerkin (DG) methods and finite volume methods have been applied to the acoustic wave equation, see [6–10], respectively.

* Corresponding author.
E-mail address: schoeder@lnm.mw.tum.de (S. Schoeder).
URL: http://www.lnm.mw.tum.de (S. Schoeder).

https://doi.org/10.1016/j.softx.2019.01.001

Table 1
Implemented time integration schemes. The indicated orders refer to the order of the time integration scheme. For ADER, also order $k+2$ can be reached if a reconstruction as in [12] is applied.

Name	Description	Order	Stages
`ExplEuler`	Explicit Euler	1	1
`clRK4`	Classical RK	4	4
`LSRK45R2`	Low-storage RK, two registers [17]	4	5
`LSRK45R3`	Low-storage RK, three registers [17]	4	5
`LSRK33R2`	Low-storage RK, two registers [17]	3	3
`LSRK59R2`	Low-storage RK, two registers [17]	5	9
`SSPRK`	Strong-stability preserving RK [18]	4	8
`ADER`	ADER as in (4) or [12]	$k+1$	–
`ADERLTS`	ADER LTS as in [12]	$k+1$	–
`ADERADCONFULL`	adjoint consistent ADER [12]	$k+1$	–

The basis for this work are the publications [11,12] on DG methods for the acoustic wave equation. In this contribution, we present a high performance high-order DG solver for the acoustic wave equation that is general in terms of geometries, frequency range, and speed of sound distributions and is combined with explicit Runge–Kutta as well as arbitrary derivative (ADER) time stepping schemes [13]. Our code supports adaptive mesh refinement in space as well as local time stepping in time, to concentrate the computational work in the regions where the most interesting physics happen. The implementation is based on the `deal.II` finite element library [14] supplying efficient matrix-free evaluation routines for quadrilaterals and hexahedra [15,16]. The algorithm provides a very high performance on modern hardware in terms of throughput, degrees of freedom processed per second, and scalability [13].

ExWave is valuable for the scientific computing community because it balances high performance and code optimizations with applicability to real world problems and available hardware. For the community of computational acoustics, this code allows to approach currently unexplored scenarios because of its high flexibility in geometry, mesh generation, or in terms of adaptivity (as in [12]) but also delivers solutions of predictable accuracy in reasonable time with moderate computational resources.

Theoretical background

The acoustic wave equation written as first order system in terms of the pressure p and the particle velocity $\boldsymbol{v}$ is given as

$$\frac{\partial \boldsymbol{v}}{\partial t} + \frac{1}{\rho}\nabla p \quad = 0, \tag{1}$$

$$\frac{\partial p}{\partial t} + c^2\rho\nabla\cdot\boldsymbol{v} = 0, \tag{2}$$

with the speed of sound c and the mass density ρ. This equation holds on a d-dimensional spatial domain $\Omega \in \mathbb{R}^d$ and in the time interval $[0, T]$ with final time T. The first equation represents the conservation of momentum while the second enforces the conservation of mass. The acoustic wave equation is accompanied by initial conditions on the pressure and the velocity as well as boundary conditions. Currently, Dirichlet boundary conditions for the pressure, homogeneous Neumann boundary conditions for the normal component of the velocity, and a first-order absorbing boundary condition are implemented in ExWave.

Spatial discretization is carried out by the DG method using the accurate fluxes from the hybridized DG method as described in [11]. Temporal discretization is carried out by explicit Runge–Kutta schemes as presented in [11] or by ADER time stepping as derived in [12,13]. The solution is expressed by time-dependent unknowns summarized in the vectors $\boldsymbol{V}, \boldsymbol{P}$. The update rule for an s-stage Runge–Kutta scheme reads

$$\begin{bmatrix}\boldsymbol{V}_{t_{i+1}}\\ \boldsymbol{P}_{t_{i+1}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{V}_{t_i}\\ \boldsymbol{P}_{t_i}\end{bmatrix} + \Delta t\sum_{j=1}^{s} b_j\boldsymbol{K}_j \quad \text{with} \tag{3}$$

$$\boldsymbol{K}_j = -\mathbb{Q}^{-1}\mathbb{K}\left(\begin{bmatrix}\boldsymbol{V}_{t_i}\\ \boldsymbol{P}_{t_i}\end{bmatrix} + \Delta t\sum_{l=1}^{j-1} a_{jl}\boldsymbol{K}_l\right),$$

and the coefficients a_{jl}, b_j from the Butcher tableau of the respective scheme. Therein, Δt is the time step and the matrices $\mathbb{Q}$ and $\mathbb{K}$ are the mass and stiffness matrix resulting from the spatial discretization as shown in [12]. For ADER time discretization, the update rule is

$$\begin{bmatrix}\boldsymbol{V}_{t_{i+1}}\\ \boldsymbol{P}_{t_{i+1}}\end{bmatrix} = \begin{bmatrix}\boldsymbol{V}_{t_i}\\ \boldsymbol{P}_{t_i}\end{bmatrix} - \mathbb{Q}^{-1}\mathbb{K}\mathbb{Q}^{-1}\sum_{j=0}^{k+1}\frac{\Delta t^{j+1}}{(j+1)!}(-1)^j \times \int_\Omega \mathbb{N}^{\mathrm{T}}\mathbb{S}^j\mathbb{N}\mathrm{d}\Omega\begin{bmatrix}\boldsymbol{V}_{t_i}\\ \boldsymbol{P}_{t_i}\end{bmatrix}, \tag{4}$$

where k is the polynomial degree of the utilized shape functions, $\mathbb{N}$ is the matrix holding the shape functions, the operator $\mathbb{S}$ comprises the spatial derivatives representing the wave equation, see [12] for details. Both methods (3) and (4) yield optimal convergence of order $k+1$ in the pressure p and the velocity $\boldsymbol{v}$. By means of reconstruction and postprocessing, even superconvergent results of order $k+2$ in p can be obtained.

Numerical example

While the implementation is quite general (see e.g. the example in Section 3.3), we will explain it along with an academic example for which the parameter file is provided in the current code version. The example is a vibrating membrane problem for which analytical solutions are available and implemented, therefore allowing for straightforward convergence tests. On the d-dimensional cube with $\Omega = [0,1]^d$, a vibrating membrane with m modes per coordinate obeys the exact solution

$$p = \cos\left(m\sqrt{d}\pi t\right)\cdot\prod_{e=1}^{d}\sin(m\pi x_e) \tag{5}$$

with homogeneous Dirichlet boundary conditions. adaptations to more general setups can be easily introduced by extensions of the code as shown in Section 3.3.

The parameter file allows to adapt the spatial dimension ($d = 2,3$), the spatial discretization (i.e., mesh, mesh refinement, mesh transformation). Additionally, parameters concerning the temporal discretization can be adapted. For time integration, one can choose between the schemes listed in Table 1. Also, the user inputs the final time, output time steps, and the Courant number from which the time step is derived. The parameter m in Eq. (5) is specified by `membrane_modes`. Other than that, ADER and ADER LTS specific parameters may be set with `use_ader_post` enabling the reconstruction and `spectral_evaluation` referring to the fast evaluation introduced in [13].

2. Software description

The presented code is based on version 9.1-pre[1] of the `deal.II` finite element library [14]. In order to ensure the functionality and correctness of the code and to ease further developments, unit tests checking the convergence orders for several setups are incorporated using the `ctest` testing framework.

[1] Any commit after 102d808 from Nov. 28, 2018.

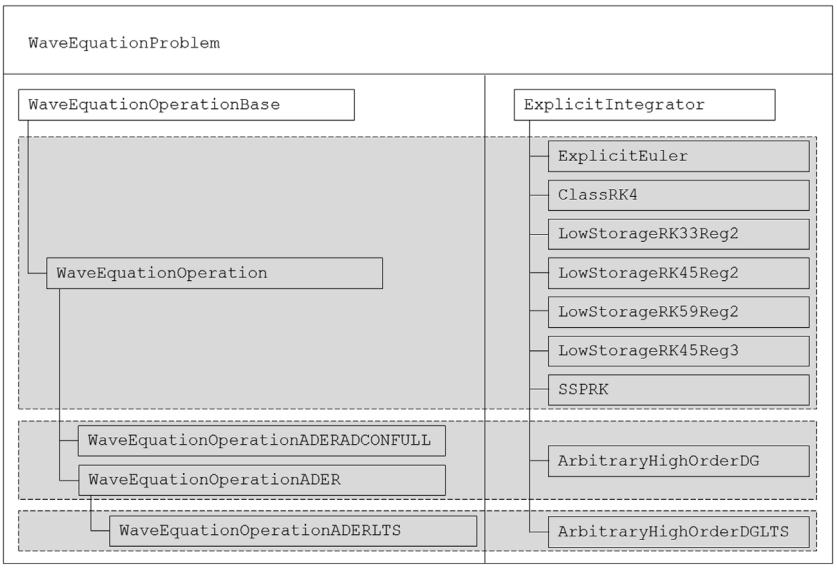


Fig. 1. The main class `WaveEquationProblem` contains an operator to evaluate the spatial discretization `WaveEquationOperationBase` and a time integrator derived from `ExplicitIntegrator`. The shaded regions indicate how spatial and temporal operator are combined.

The main class of ExWave is `WaveEquationProblem`. Its method `run()` executes the time loop. The main components are a time integrator derived from `ExplicitIntegrator` and a spatial operator derived from `WaveEquationOperationBase`, as shown in Fig. 1. The time integrators execute the vector updates and call the spatial operator application. For ADER, spatial and temporal evaluation are strongly interlinked and the entire evaluation takes place in `WaveEquationOpeationADER`. The LTS requires a complex update call, which is handled by a `ClusterManager` which in turn is called by `WaveEquationOperationADERLTS`.

The class `WaveEquationOperation` is templated on the dimension d and the polynomial degree k of the shape functions. It relies heavily on the `MatrixFree` class of the `deal.II` library and uses the optimized evaluation routines `FEEvaluation` and `FEFaceEvaluation`. Matrix-free operator evaluation allows for a much higher performance compared to classical matrix-based schemes, which is due to a higher arithmetic density. Also, fast integration techniques relying on sum factorization utilizing the tensor product structure of the shape functions are used and explicit cross-cell vectorization is enabled. Details on matrix-free methods with sum factorization techniques and performance in the context of DG for the acoustic wave equation can be found in [15,16] and [13], respectively. `WaveEquationOperationBase` can be flexibly switched between single and double precision by a typedef `value_type`.

In the file `time_integrators.h`, not only the time integrators but also an optimized vector updater are implemented. The vector updater `RKVectorUpdater` merges several vector updates into a single loop over the entries and thereby reduces the number of required vector reads from five to two per stage. This contribution allows one to increase performance by a factor of 1.7× for LSRK45R2 on modern processors where performance is typically memory-bandwidth limited [13].

One aspect of the code we want to mention explicitly is that it allows to run an iterative CFL stability analysis based on a stability criterion in terms of L_2 pressure errors to find a tight fit of the critical Courant number for a certain problem configuration.

3. Illustrative examples

In this chapter, one academic example is presented, followed by two performance tests, and last a representative for real world problems.

3.1. A convergence test

The correctness of the methods and code is demonstrated with convergence tests. To run a convergence test based on the vibrating membrane, a basic set of input parameters must be specified and then several simulations are run varying the parameter `n_refinements`. The L_2 pressure error at the final time is output.

Results are shown for two different exemplary configurations. The first setup is a two-dimensional geometry and the temporal discretization relies on ADER LTS. Furthermore, the parameter `use_ader_post=true` is set to obtain superconvergent results [12]. The second setup is three-dimensional and temporal discretization is based on LSRK33R2. For both setups, the number of membrane modes in the analytic solution is set to the polynomial degree of the shape functions and `cfl_number=0.1` with `n_initial_intervals = 5`. All other parameters are set as in the default parameter file.

Fig. 2 summarizes the results for the tested polynomial degrees $k = 1, 2, 3, 4, 5, 6$. The expected convergence orders of $k + 1$ for the pressure p and $k + 2$ for the postprocessed pressure p^* are obtained for all tested polynomial degrees with ADER LTS. For LSRK33R2, the expected convergence rates are only obtained for coarse discretizations. For fine discretizations, the temporal error dominates because LSRK33R2 is only of order three.

3.2. Performance evaluation

We compare ADER and LSRK45R2 in terms of the throughput on 28 cores of a dual-socket Intel Xeon Broadwell E5-2690v4 machine operating at 2.6 GHz, compiled with the g++ compiler, version 6.2, at optimization level `-march=haswell -O3 -funroll-loops`. The setup is as in Section 3.1 with a three dimensional geometry meshed with 80^3 elements for $k = 1, 2, 3$ and 40^3 elements for $k = 4, \ldots, 12$. Fig. 3 plots the results. A throughput up to $6.33 \cdot 10^8$ degrees of freedom per second for ADER with $k = 3$ is reached, which is 3.9 times higher than the respective throughput for LSRK45R2. For higher polynomial degrees, the advantage of ADER decreases slowly due to the fact that ADER is of order $k + 1$ with additional computations for the higher orders while LSRK45R2 is of order four for all polynomial degrees.

To demonstrate parallel capabilities, we perform a strong scaling experiment with h-adaptivity based on a three-dimensional

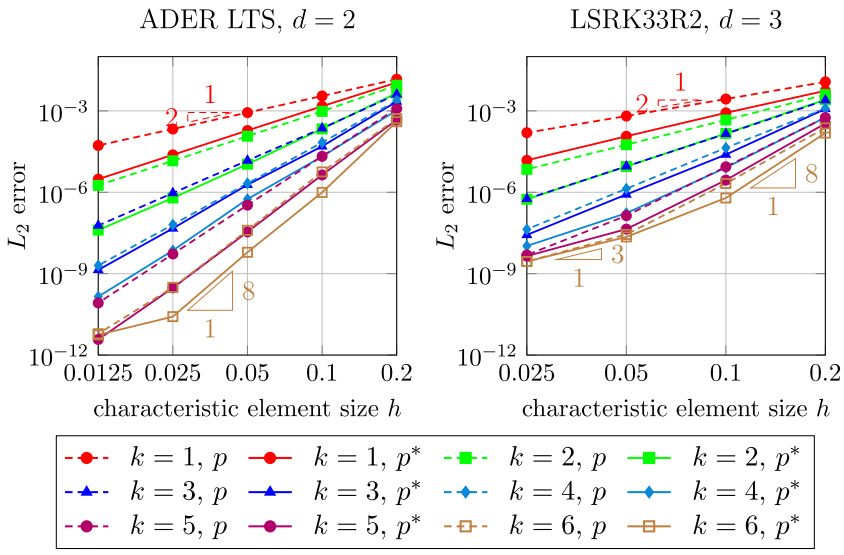


Fig. 2. Convergence study in two dimensions for ADER LTS and in three dimension for LSRK33R2.

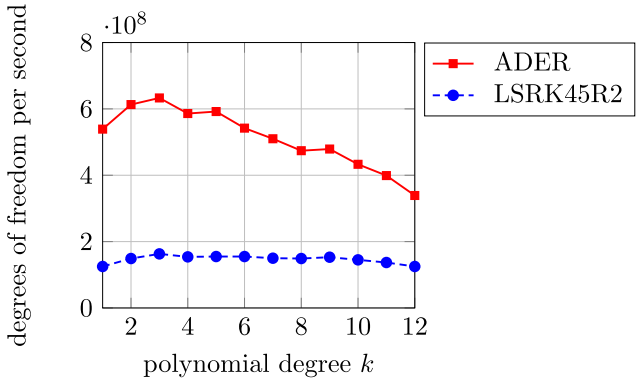


Fig. 3. Throughput for ADER and LSRK45R2 for $d = 3$ and $k = 1, \ldots, 12$ on one node of 28 cores.

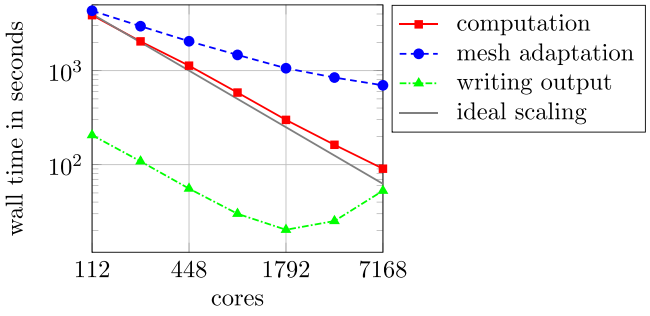


Fig. 4. Strong scaling of full simulation with 43,000 time steps and up to 180 million spatial unknowns with ADER for $d = 3$ and $k = 5$.

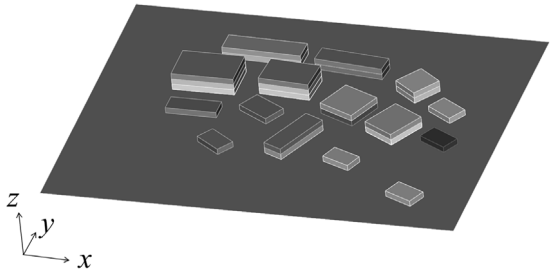
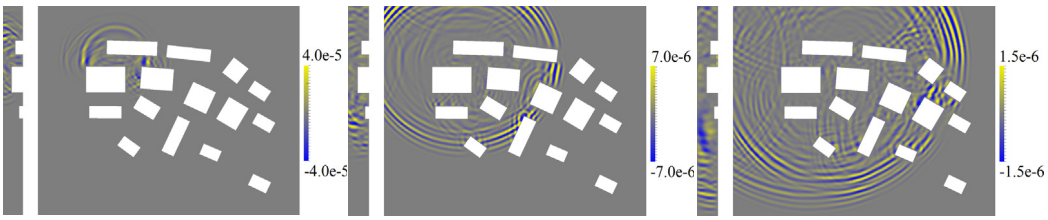


Fig. 5. Geometry of the training village.

setup as in [13], Section 4.7, incorporating a material inhomogeneity with an impedance mismatch of four and an unstructured mesh capturing a curved interface between the two materials with a high-order mapping. An adaptive update is carried out every 500 time steps at a Courant number of 0.1. Fig. 4 shows the results obtained on the SuperMUC Phase 2 system using 112 to 7168 cores of Intel Haswell E5-2697 v3, running at a frequency of 2.1 GHz. The computational time scales almost ideally while a slowdown can be observed for the adaptation as explained in [19–21] and due to the fact that data structures for h-adaptivity in `deal.II` are not as optimized as the matrix-free implementations. The weak scaling is better since the main cost of the adaptivity is due to the cost of `deal.II`'s generic transfer routines between mesh levels, the geometry evaluation at ghost cells, and cost of data transfer while repartitioning. Writing output scales good for small numbers of cores but degrades after 1792 cores.

3.3. Urban acoustics

This example examines sound propagation in a village representing outdoor acoustics, which is relevant for urban planning and city design [22] or useful for gun shot localization in the context of crime control [23]. The geometry under consideration is based on the artificial village presented in [24,25], where a FDTD method and an adaptive rectangular decomposition approach are used to solve the acoustic wave equation. The geometry consists of fifteen buildings of different height. Fig. 5 depicts the geometry of the buildings. The computational domain is a cuboid of size $175 \times 140 \times 14$ with the buildings cut out. Surfaces corresponding to walls, roofs, and the ground are assumed to be perfectly reflecting, whereas the first order absorbing boundary condition is applied on all other boundaries. A sound source is located at (62, 104, 1) corresponding to SP1 from [25]. To read in the externally generated mesh and set corresponding boundary conditions and initial pressure fields, adaptations to `input_parameters.h` are necessary as shown in the branch `urbanacoustics` of the supplied software. In [25], simulations were run on a mesh consisting of $1.1 \cdot 10^7$ grid points and a time step size of $\Delta t = 3.85 \cdot 10^{-4}$ with 2000 time steps which corresponds to a simulation frequency of 450 Hz, taking 20 min on a single core CPU machine. Here, we run



(a) Pressure at $t = 0.1$ (b) Pressure at $t = 0.2$ (c) Pressure at $t = 0.3$

Fig. 6. Pressure snapshots in the training village on the *xy* plane at $z = 1$ and on the *yz* plane at $x = 62$.

simulations on a discretization of average grid spacing 1.2, $k = 3$, resulting in $1.15 \cdot 10^7$ grid points with $\Delta t = 2.45 \cdot 10^{-5}$ and 19704 time steps to reach the same final time. Fig. 6 plots several pressure snapshots to give an impression of the sound propagation patterns. Simulation on 28 cores of a two socket Intel Xeon E5-2690 v4 Broadwell 2.6 GHz system requires $2.5 \cdot 10^3$ seconds which corresponds to 76 CPU minutes for 2000 time steps. Hence, a computation on the same geometry with the same number of grid points and time steps is only 3.8 times slower compared to the adaptive rectangular decomposition. This is a very good result, considering that the adaptive rectangular decomposition relies on a semi-analytic approach using the discrete cosine transform in the decomposed rectangles and considering that their time integration is not of order five but a two step type. A comparison of the computational performance considering accuracy and temporal stability should be addressed by future work.

4. Impact

The presented algorithm was used in several research papers [11–13]. The code impacts two communities. First, for the scientific computing community, this code is a representative of a very efficient implementation of matrix-free operator evaluation with throughput close to much simpler finite difference methods, despite support for complex meshes, including local mesh refinement and curved meshes. The code allows to run conclusive performance tests as shown in [13] and Section 3.2, and therefore enables researchers concerned with high performance computing to test and validate new methods to reduce computational time. This new concept can be applied in a wide variety of problems beyond the acoustic wave equation, such as other wave propagation problems in electromagnetics or seismics. Furthermore, the Runge–Kutta related code part can be easily adapted to nonlinear problems, such as the compressible Navier–Stokes equations with explicit time stepping schemes. As demonstrated by results in [26], the present concepts are faster by a factor three to five over the state of the art in the DG literature. In addition, it is an ideal test bed for introducing advanced features, such as cut finite element technology for representing complicated geometries on unfitted meshes, e.g. by the ongoing work [27].

Secondly, this code will have an impact on computational acoustics. Historically dominated by finite difference and semi-analytic methods, we now have a code at hand which is not only high performing but also applicable to problems of practical relevance. It offers more flexibility compared to finite difference solvers, e.g. in terms of meshes and adaptivity.

5. Conclusions

We presented the high performance code ExWave to solve the acoustic wave equation based on higher order DG spatial discretization in combination with explicit Runge–Kutta and ADER global and local time stepping relying on the `deal.II` finite element library, or, more precisely, the matrix-free framework supplying fast quadrature with sum factorization. Also, ExWave allows to validate implementations in terms of spatial and temporal convergence, to determine temporal stability limits in terms of a CFL stability analysis, as well as to measure and compare computational performance for different discretizations. Urban acoustic simulations, as exemplary shown in Section 3.3, enable city planning or street canyon design minimizing noise exposure. A potential extension is the implementation of a variety of boundary conditions that are commonly used in room and urban acoustics. A future study should compare this solver to adaptive rectangular decomposition methods not only in terms of computational time but also accuracy.

Acknowledgments

This work was supported by the German Research Foundation (DFG) and the Technical University of Munich within the funding programme Open Access Publishing. Additionally, the authors acknowledge support by the German Research Foundation (DFG) through the project “High-order discontinuous Galerkin for the exa-scale" (ExaDG) within the priority program “Software for Exascale Computing" (SPPEXA), grant agreement no. KR4661/2-1 and WA1521/18-1.

References

[1] Siltanen S, Lokki T, Savioja L. Rays or waves? Understanding the strengths and weaknesses of computational room acoustics modeling techniques. In: Proceedings of the international symposium on room acoustics. Melbourne, Australia; 2010, p. 1–6.
[2] Escolano J, Navarro J, Lopez J. On the limitation of a diffusion equation model for acoustic predictions of rooms with homogeneous dimensions. J Acoust Soc Am 2010;128(4):1586–9. http://dx.doi.org/10.1121/1.3479756.
[3] Botteldooren D. Finite-difference time-domain simulation of low-frequency room acoustic problems. J Acoust Soc Am 1995;98(6):3302–8. http://dx.doi.org/10.1121/1.413817.
[4] Raghuvanshi N, Narain R, Lin MC. Efficient and accurate sound propagation using adaptive rectangular decomposition. IEEE Trans Vis Comput Graphics 2009;15(5):789–801. http://dx.doi.org/10.1109/TVCG.2009.28.
[5] Morales N, Mehra R, Manocha D. A parallel time-domain wave simulator based on rectangular decomposition for distributed memory architectures. Appl Acoust 2015;97:104–14. http://dx.doi.org/10.1016/j.apacoust.2015.03.017.
[6] Bangerth W, Rannacher R. Finite element approximation of the acoustic wave equation: error control and mesh adaptation. East-West J Numer Math 1999;7(4):232–82.
[7] Cohen G, Fauqueux S. Mixed finite element with mass-lumping for the transient wave eqaution. J Comput Acoust 2000;8(1):171–88. http://dx.doi.org/10.1142/S0218396X0000011X.
[8] Komatitsch D, Tromp J. Introduction to the spectral element method for three-dimensional seismic wave propagation. Geophys J Int 1999;139:806–22. http://dx.doi.org/10.1046/j.1365-246x.1999.00967.x.
[9] Dumbser M, Käser M. An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – II. The three-dimensional isotropic case. Geophys J Int 2006;167:319–36. http://dx.doi.org/10.1111/j.1365-246X.2006.03120.x.
[10] LeVeque RJ. Finite volume methods for hyberbolic problems. Cambridge texts in applied mathematics, Cambridge; 2002, http://dx.doi.org/10.1017/CBO9780511791253.

[11] Kronbichler M, Schoeder S, Müller C, Wall W. Comparison of implicit and explicit hybridizable discontinuous Galerkin methods for the acoustic wave equation. Internat J Numer Methods Engrg 2016;106(9):712–39. http://dx.doi.org/10.1002/nme.5137.

[12] Schoeder S, Kronbichler M, Wall W. Arbitrary high-order explicit hybridizable discontinuous Galerkin methods for the acoustic wave equation. J Sci Comput 2018;76(2):1–38. http://dx.doi.org/10.1007/s10915-018-0649-2.

[13] Schoeder S, Kormann K, Wall W, Kronbichler M. Efficient explicit time stepping of high order discontinuous Galerkin schemes for waves. SIAM J Sci Comput 2018;40(6):C803–26. http://dx.doi.org/10.1137/18M1185399.

[14] Alzetta G, Arndt D, Bangerth W, Boddu V, Brands B, Davydov D, Gassmoeller R, Heister T, Heltai L, Kormann K, Kronbichler M, Maier M, Pelteret J-P, Turcksin B, Wells D. The `deal.II` library, version 9.0. J Numer Math 2018;26(4):173–83. http://dx.doi.org/10.1515/jnma-2018-0054.

[15] Kronbichler M, Kormann K. A generic interface for parallel cell-based finite element operator application. Comput Fluids 2012;63:135–47. http://dx.doi.org/10.1016/j.compfluid.2012.04.012.

[16] Kronbichler M, Kormann K. Fast matrix-free evaluation of discontinuous Galerkin finite element operators, arXiv preprint (2017) http://arxiv.org/abs/1711.03590v1.

[17] Kennedy CA, Carpenter MH, Lewis RM. Low-storage, explicit Runge-Kutta schemes for the compressible Navier–Stokes equations. Appl Numer Math 2000;35:177–219. http://dx.doi.org/10.1016/S0168-9274(99)00141-5.

[18] Kubatko EJ, Yeager BA, Ketcheson DI. Optimal strong-stability-preserving Runge–Kutta time discretizations for discontinuous Galerkin methods. J Sci Comput 2014;60:313–44. http://dx.doi.org/10.1007/s10915-013-9796-7.

[19] Burstedde C, Wilcox LC, Ghattas O. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. SIAM J Sci Comput 2011;33(3):1103–33. http://dx.doi.org/10.1137/100791634.

[20] Burstedde C, Holke J. A tetrahedral space-filling curve for non-conforming adaptive meshes, arXiv preprint (2017) https://arxiv.org/abs/1509.04627v2.

[21] Bangerth W, Burstedde C, Heister T, Kronbichler M. Algorithms and data structures for massively parallel generic finite element codes. ACM Trans Math Software 2011;38(2). http://dx.doi.org/10.1145/2049673.2049678.

[22] Kang J. Urban sound enviroment. 2 Park Square, Milton Park, Abingdon, OXon PX14 4RN: Taylor and Francis; 2007.

[23] Ledeczi A, Volgyesi P, Maroti M, Simon G, Balogh G, Nadas A, Kusy B, Dora S, Pap G. Multiple simultaneous acoustic source localization in urban terrain. In: Fourth international symposium on information processing in sensor networks. 2005, p. 491–6. http://dx.doi.org/10.1109/IPSN.2005.1440982.

[24] Albert DG, Liu L. The effect of buildings on acoustic pulse propagation in an urban environment. J Acoust Soc Am 2010;127(3):1335–46. http://dx.doi.org/10.1121/1.3277245.

[25] Morales N, Mehra R, Manocha D. Acoustic pulse propagation in an urban environment using a three-dimensional numerical simulation. J Acoust Soc Am 2014;135(6). http://dx.doi.org/10.1121/1.4874495.

[26] Fehn N, Wall W, Kronbichler M. A matrix-free high-order discontinuous Galerkin compressible Navier–Stokes solver: A performance comparison of compressible and incompressible formulations for turbulent incompressible flows. Internat J Numer Methods Fluids 2019;89(3):71–102. http://dx.doi.org/10.1002/fld.4683.

[27] Schoeder S, Sticko S, Kreiss G, Kronbichler M. High order cut discontinuous galerkin methods with local time stepping for acoustics. 2018, submitted for publication.