CEAS Aeronautical Journal (2020) 11:745–766
https://doi.org/10.1007/s13272-020-00450-1

ORIGINAL PAPER

Lattice–Boltzmann simulations for complex geometries on high-performance computers

Andreas Lintermann[1,2] · **Wolfgang Schröder**[2,3]

Received: 28 December 2018 / Revised: 25 March 2020 / Accepted: 2 April 2020 / Published online: 13 May 2020

Abstract

Complex geometries pose multiple challenges to the field of computational fluid dynamics. Grid generation for intricate objects is often difficult and requires accurate and scalable geometrical methods to generate meshes for large-scale computations. Such simulations, furthermore, presume optimized scalability on high-performance computers to solve high-dimensional physical problems in an adequate time. Accurate boundary treatment for complex shapes is another issue and influences parallel load-balance. In addition, large serial geometries prevent efficient computations due to their increased memory footprint, which leads to reduced memory availability for computations. In this paper, a framework is presented that is able to address the aforementioned problems. Hierarchical Cartesian boundary-refined meshes for complex geometries are obtained by a massively parallel grid generator. In this process, the geometry is parallelized for efficient computation. Simulations on large-scale meshes are performed by a high-scaling lattice–Boltzmann method using the second-order accurate interpolated bounce-back boundary conditions for no-slip walls. The method employs Hilbert decompositioning for parallel distribution and is hybrid MPI/OpenMP parallelized. The parallel geometry allows to speed up the pre-processing of the solver and massively reduces the local memory footprint. The efficiency of the computational framework, the application of which to, e.g., subsonic aerodynamic problems is straightforward, is shown by simulating clearly different flow problems such as the flow in the human airways, in gas diffusion layers of fuel cells, and around an airplane landing gear configuration.

Keywords Lattice–Boltzmann methods · High-performance computing · Respiratory flows · Gas diffusion layers · Landing gear

1 Introduction and historical review

As computing power continuously increases, more and more complex engineering problems are investigated by means of computational fluid dynamics (CFD) methods. In research, high-performance computing (HPC) systems are commonly employed to solve large-scale multi-physics problems in fundamental and applied engineering. HPC architectures come in various flavors. The trend in HPC is towards heterogeneous architectures, which poses new challenges to simulation software. The TOP500 list of supercomputers[1] reflects this trend, e.g., the leading machine SUMMIT (status of 2020) of the Oak Ridge National Laboratory, U.S., is equipped with a massive amount of GPU accelerators. The adaption of software to run efficiently on such systems necessitates porting and tuning activities as well as multi-level parallelizations and optimized domain decompositioning methods.

A promising numerical method for the efficient simulation of flows are lattice–Boltzmann (LB) methods [3, 27, 31]. Their parallelization and their compute kernel implementation is simple, and intricate geometries can efficiently be treated. They are derived from lattice-gas cellular automata (LGCA). First theoretical work in [75] led to an application in fluid mechanics, i.e., to the first numerical solution of the Navier–Stokes equations via LGCA [20]. LGCA, however, suffer from their non-Galilean invariance

✉ Andreas Lintermann
A.Lintermann@fz-juelich.de

1 Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH, Wilhelm-Johnen-Straße, 52425 Jülich, Germany

2 Jülich Aachen Research Alliance Center for Simulation and Data Science (JARA-CSD), Seffenter Weg 23, 52074 Aachen, Germany

3 Institute of Aerodynamics and Chair of Fluid Mechanics, RWTH Aachen University, Wüllnerstr. 5a, 52062 Aachen, Germany

1 TOP500 list www.top500.org.

at high Reynolds numbers Re leading to statistical noise [59, 74]. McNamara and Zanetti [57] solved this problem by replacing the individual particle consideration by particle probability distribution functions (PPDFs). In [59], the collision matrix [31] is replaced by a single relaxation scheme. Higuera et al. [31] were the first to show results with the LB method for 2D flows in a channel containing a periodic array of obstacles. This is in line with the physical work of Bhatnagar–Gross–Krook and led to the numerical BGK model. Qian et al. [59] also introduce the $DxQy$ discretization model with x space dimensions and y discrete directions. In [42, 56, 74], the Boltzmann equation is derived and by a Chapman–Enskog development the Navier–Stokes equations. Based on this model, a lot of novel variants of the LB method for different applications were deduced.

The stability of the standard BGK method with a single relaxation time (SRT) is non-linear dependent on the flow solution and the collision frequency. The viscosity ν is limited by a stability threshold, i.e., for $Re \rightarrow \infty$ the viscosity cannot become infinitely small. Moment-based methods such as the multiple relaxation time (MRT) [12] and the cascaded LB (CLB) [22] methods are in this respect advantageous. In the MRT model, the relaxation is performed in moment space, which decouples the conservative and non-conservative variables. This leads to an increase in accuracy and stability. The CLB method also relaxes in moment space. Moments of the fourth order of the velocity components are, however, reconstructed from lower order moments in a cascade to capture oscillations in high-frequency space. Yet, this requires to use at least a $D3Q27$ discretization model. The entropic LB (ELB) method [36] increases the stability by an adaption of the relaxation time in each LB iteration. For the correction of instabilities, an entropy equation such as the Boltzmann entropy function or the Tsallis entropy function [5] is employed. The ELB model is stable even at high Reynolds numbers and more efficient than the MRT model [1]. The collision operator in the regularized LB (RLB) method [43] is separated into an equilibrium and non-equilibrium part. Under application of a term for the non-equilibrium stresses, the collision term is transformed into a solvable form. The RLB method is not as stable as the MRT, CLB, or ELB methods, and the computational costs are, however, similar to the standard BGK operator. Recently, the cumulant LB method [23] has been developed, which shows a smaller analytical and numerical error than the MRT method. The collision operator in this model is based on the so-called cumulants. The method is especially suited for high Reynolds number flows.

For the derivation of the equilibrium distribution function, a small Mach number Ma approximation is used [27], which limits the aforementioned methods to weakly compressible flow. This leads to a decoupling of the energy equation from the Navier Stokes equations. To satisfy the first law of thermodynamics or, in other words, to determine the temperature distribution, a multidistribution function (MDF) approach [26] is commonly used. Natural convection, i.e., a change of the density based on the temperature, can be realized by incorporating the Boussinesq approximation into the Navier–Stokes equations. The MDF method neglects, however, added work via compression and viscous heating at moderate Mach numbers. For $Ma > 0.3$, either more energy states are considered [71] or finite-difference approaches [37] are employed. The full Navier–Stokes equations can be derived from the discretized equations using discretization models with $y = 120$ or $y = 39$ [66]. Chen et al. [9] present a method based on this approach, which is capable of simulating flows at $Ma = 2.0$.

LB methods are, furthermore, suited for the simulation of multi-phase flows. Depending on which thermal equation of state (EOS) and maximum density ratio are considered, or how fast and accurate a solution is requested, different models have been developed. Color-gradient methods [25] distinguish between two differently colored fluids, each with an own set of PPDFs [60]. Although this method is quite accurate, high-density gradients may lead to numerical instabilities. Shan-Chen models [65] generate a phase separation by integrating attracting and repelling forces in the vicinity of phase interfaces. This, however, leads to spurious currents. In free-energy models [72], the issue of the non-monotonic thermal EOS is addressed. The EOS is incorporated into the pressure tensor of the Navier–Stokes equations. Like the color-gradient method, the free-energy methods suffer from limiting density ratios. The interface tracking methods [30] solve for two sets of PPDFs. Macroscopically, the Cahn–Hilliard and the Navier–Stokes equations are considered. Recent advances using interface tracking methods [45] allow high-density gradients on the order of $\mathcal{O}(10^3)$ and an efficient and accurate solution scheme.

In general, all aforementioned LB schemes operate on Cartesian meshes, which may lead to an excessive number of cells due to the unique grid spacing, e.g., in boundary-layer flows. The application of boundary-refined Cartesian meshes delivers in such cases only a slight reduction of the computational effort. Furthermore, the Courant–Friedrichs–Lewy (CFL) number is bound to unity. It is hence worth noting that there exist several works on non-uniform meshes [13, 39, 58]. In [58], a finite volume (FV) formulation of the LB equation is derived, extending the applicability of LB methods to irregular meshes. Krämer et al. [39] employ a semi-Lagrangian propagation step, where the streamed PPDF is reconstructed by a finite-element approach. This allows for non-unity CFL numbers, higher order spatial discretizations, and higher time steps. Di Ilio et al. [13] combine the standard Cartesian approach with an FV formulation and name it the hybrid LB method (HLBM). In this method, two meshes, a Cartesian mesh in the far field of a solid object and an unstructured body-fitted FV mesh in the vicinity of the object, are combined. In the overlapping region, information is exchanged via interpolation. Note that despite the advantages of these irregular mesh methods, they complicate automatic meshing and dynamic solution-adaptive refinement, which is why the approach in this paper employs hierarchical Cartesian meshes. For a good introduction and further details on LB simulations, the reader is referred to [3, 27].

The Institute of Aerodynamics and Chair of Fluid Mechanics, RWTH Aachen University, develops the framework Zonal Flow Solver (ZFS) [52], which is supported by the Jülich Aachen Research Alliance Center for Simulation and Data Science (JARA-CSD). This framework features a massively parallel Cartesian grid generator, parallelized geometries, and, amongst others, different LB methods to solve technical and biomedical engineering problems. The code is written in C++. In this contribution, the numerical methods implemented in ZFS are described and some application examples of large-scale simulations from different fields are presented as proof of concept. Note that despite the theory of the computational kernels is derived from available publications, the modularity, meshing, and parallelization approaches, also for the geometry, are unique to ZFS. That is, the paper aims at delivering details on a production code, which is under continuous development and frequently used. For a review of other LB simulation codes in the context of large-scale computations on HPC systems, the reader is referred to [70].

The paper is structured as follows. Section 2 discusses the numerical methods and corresponding performance aspects. Subsequently, Sect. 3 presents the application examples, i.e., flow computations in the human respiratory tract, in gas diffusion layers (GDLs) of fuel cells, and around a airplane landing gear configuration. In Sect. 4 a summary and a conclusion are given. Finally, in Sect. 5, other fields of application and an outlook are presented.

2 Numerical methods

The simulation of flows necessitates to follow a particular workflow path. First, the computational mesh corresponding to a flow problem and an associated geometry is generated. The mesh and a parallelized geometry are input to the multiphysics framework ZFS, which uses its solvers to compute an approximate solution of the Navier–Stokes equations. The following Sect. 2.1 describes the method to generate the mesh and a parallelized geometry. This is followed by a description of the LB solver in Sect. 2.2, which is used for the applications presented in Sect. 3. Some performance aspects of the simulation code are presented in Sect. 2.3. A workflow chart for the LB method is also shown in Fig. 1.

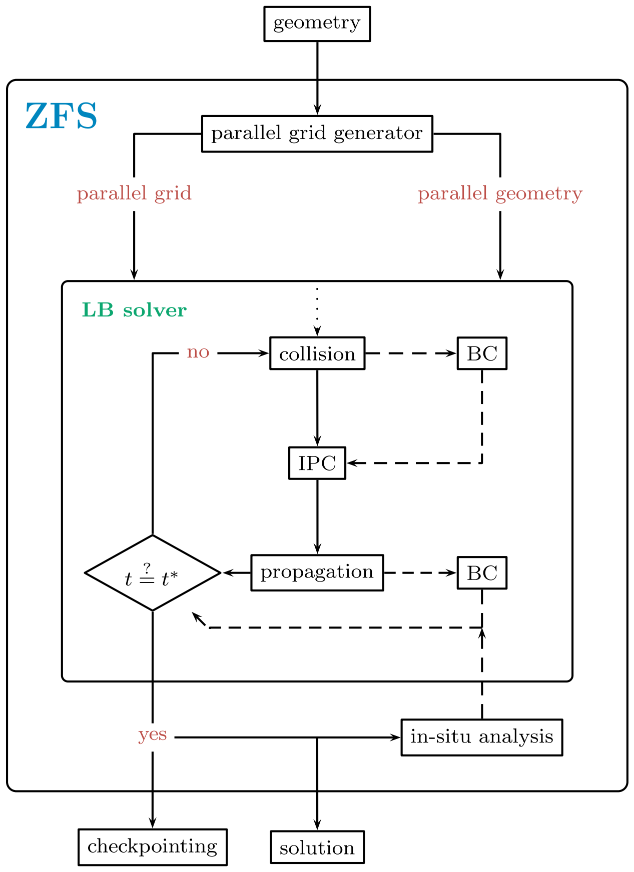


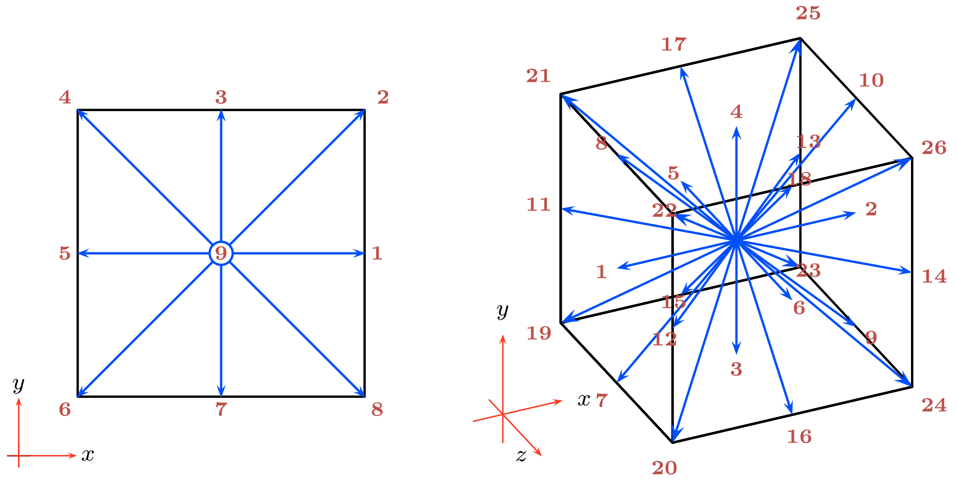
Fig. 1 Workflow overview of ZFS using the LB solver. At time step $t = t^*$, some event is triggered, i.e., checkpointing, solution output, or in-situ analysis is performed

2.1 Mesh generation

The mesh generator [53] is part of the framework ZFS and generates hierarchical Cartesian meshes. It is independent of the method employed for the simulation. It fully works in parallel using the Message Passing Interface (MPI) and OpenMP, and is able to construct meshes in a short amount of time on a large number of processes.

Parallel meshing begins with an I/O of the geometry, which is usually given in Standard Tessellation Language (STL). Each process places an initial cube around this geometry and starts to continuously subdivide the cube into smaller cubes. This constitutes an octree hierarchy with parent–child and neighborhood relations, and cells living on different levels l in the tree. The subdivision is performed until a level l_α is reached. In each iteration, cells that are outside the geometry are removed by a mixture of intersection tests and flooding algorithms. At l_α, all levels $l < l_\alpha$ are removed and the remaining cells are decomposed by a Hilbert space-filling curve [61]. Each MPI rank keeps only those cells which it is responsible for and creates rank-neighborhood information based on the local domain boundary cells, the

Fig. 2 *DxQy* LB discretization schemata in two and three dimensions, i.e., for the *D*2*Q*9, *D*3*Q*15, *D*3*Q*19, and *D*3*Q*27 LB models



(a) *D*2*Q*9 LB discretization scheme. The rest particles distribution is located at position 9 in the center.

(b) *D*3*Qy* LB discretization scheme with $y \in \{15, 19, 27\}$. The rest particle distributions for the various models is located in the center with indices 15, 19, 27.

so-called window cells. Continuous subdivision is then performed on the remainder of the cells up to a level $l_\beta > l_\alpha$. At this stage, the mesh is uniformly refined. The subsequent step creates locally refined meshes. The algorithm can refine regions, where higher resolutions are required based on user-defined geometrical objects or the distance of the boundary. For the former method, cells inside the defined geometrical shapes are refined, while for the latter method, the distance from the wall is measured and used as in indicator for refinement. Global cell neighborhood is restored using the window and the corresponding halo cells, which are created as copy of each window cell on a neighboring MPI rank. From the decomposition on l_α, a list of cells is defined, which is used in a preprocessing step to the solver for the weighted decomposition of the mesh. This list is also utilized for the parallelization of the geometry, i.e., for each cell in this list, the number of triangles and the corresponding triangle information is gathered and written to disk in parallel using either parallel NetCDF [47] or HDF5 [18]. These parallel libraries are in a final step also used to write the mesh to disk. For more details on the parallel mesh generator, the interested reader is referred to [49, 53].

2.2 Lattice–Boltzmann solver

ZFS implements different LB methods, i.e., the standard SRT, MRT, RLB, and CLB models. For spatial discretization, all these models employ the parallel octree mesh placed into memory via parallel I/O. The space is furthermore discretized using the *DxQy* schemata from [59]. Figure 2 shows the various discretization schemata implemented in ZFS. The inner box of Fig. 1 shows the basic algorithm of the LB methods, i.e., a collision step, which represents, in a statistical sense, the change of PPDFs due to cell-local collisions, is usually followed by a propagation step, which pushes the new PPDFs to the neighboring cells. Since the code is executed in parallel, window cells need to copy their information to the neighboring MPI ranks via inter-process communication (IPC) to their corresponding halo cells such that a valid propagation can be executed. Depending on the kind of boundary condition (BC), the BC is either executed right after collision and before IPC, or after propagation. After that, the iteration step t is advanced, and if $t = t^*$, a special event is executed. Such events can be checkpointing actions, solution I/O, in-situ analysis, or simply finishing the simulation at a target time step $t^* = t_{end}$.

The most frequently used LB models are the SRT with the *D*3*Q*27 scheme. The MRT, RLB, and CLB models come, however, with advanced features with respect to the range of applications and stability. The following describes these methods independent of the *DxQy* schemata. Furthermore, a method for large-eddy simulations (LES), the employed mesh refinement technique, and boundary conditions are described. Note that all methods have previously been validated in [15, 19, 50]. That is, in [15], the SRT method with mesh refinement is validated by simulating the flow past a circular cylinder at Reynolds numbers $Re = 40$ and $Re = 100$, and past a sphere at $100 \leq Re \leq 300$ and at $3700 \leq Re \leq 10{,}000$. For the latter case, the Reynolds number is on the same order as that of the landing gear configuration presented in Sect. 3.3. A detailed analysis of the averaged drag coefficient $\bar{C}_d$, the mean base pressure coefficient $\bar{C}_{pb}$, the mean non-dimensional length of the recirculation region $\bar{L}_r/D$, the mean separation angle $\bar{\Phi}_s$, the Strouhal number St for the large-scale vortex shedding, the mean wall-pressure coefficient $\bar{C}_p$, and the mean skin-friction $\bar{\tau}$ is given. The values and distributions are in good agreement with the other results from the literature. In [19], the MRT and the CLB methods are validated for the D3Q19 and D3Q27 models by simulations of plane Poiseuille flow at $Re = 200$, flow in a three-dimensional lid-driven cavity at yaw at $Re = 700$, and of a turbulent channel flow at $Re_\tau = 200$. Finally, in [50], the thermal MDF LB approach is validated by analyzing a boundary-layer flow over a heated flat plate at $Re = 10,000$ and a Prandtl number of $Pr = 1.0$.

2.2.1 SRT method

The SRT method [59] equates the PPDFs $f_i(r + \delta r, t + \delta t), i \in \{1 \dots, y\}$ with a single relaxation parameter ω_F by:

$$f_i(r + \delta r, t + \delta t) = f_i(r, t) + \omega_F \left(f_i^{eq}(r, t) - f_i(r, t)\right). \tag{1}$$

The parameter ω_F is a function of the inverse of the viscosity ν, that is:

$$\omega_F = \frac{c_s^2}{\nu + \delta t c_s^2/2}. \tag{2}$$

In Eq. (1), r represents the spatial location, δr is the grid distance, t is the time, δt is the time increment, and f_i^{eq} is the discretized Maxwellian equilibrium distribution function given by:

$$f_i^{eq}(r, t) = \rho t_p \underbrace{\left[1 + \frac{v_a \xi_{i,a}}{c_s^2} + \frac{v_a v_b}{2c_s^2} \cdot \left(\frac{\xi_{i,a}\xi_{i,b}}{c_s^2} - \delta_{ab}\right)\right]}_{\chi}, \tag{3}$$

where ρ is the density, t_p is a discretization scheme-dependent factor, see Appendix A, c_s is the speed of sound, $v_{a,b}$ and $\xi_{i,a,b}$ are fluid velocity and molecular velocity components, and δ_{ab} is the Kronecker delta with indices $a, b \in \{1, \dots, x\}$. The algorithm usually performs the collision and propagation steps in separate steps, i.e., an explicit scheme, alternating between collision and propagation operations, is used:

$$\hat{f}_i(r, t) = f_i(r, t) + \omega_F \left(f_i^{eq}(r, t) - f_i(r, t)\right) \tag{4}$$

$$f_i(r + \delta r, t + \delta t) = \hat{f}_i(r, t), \tag{5}$$

where the PPDFs with a <^> represent the post-collision PPDFs. The macroscopic variables can be obtained from the moments of the PPDFs, see, e.g., Hänel [27]:

$$\rho = \sum_i f_i(r, t) \tag{6}$$

$$\rho v_a = \sum_i \xi_{i,a} f_i(r, t) \tag{7}$$

$$\rho(e + v_a^2) = \frac{1}{2} \sum_i \xi_{i,a}^2 f_i(r, t) \tag{8}$$

$$\underbrace{\rho v_a v_b + p\delta_{ab} - \sigma_{ab}}_{\Pi_{ab}} = \sum_i \xi_{i,a}\xi_{i,b} f_i(r, t). \tag{9}$$

The temperature distribution can be simulated by an MDF approach [26], i.e., by additionally solving:

$$g_i(r + \delta r, t + \delta t) = g_i(r, t) + \omega_T \left(g_i^{eq}(r, t) - g_i(r, t)\right). \tag{10}$$

The parameter ω_T depends on the heat conduction coefficient κ of the fluid:

$$\omega_T = \frac{c_s^2}{\kappa + \delta t c_s^2/2}, \tag{11}$$

where κ is given by the Prandtl number $Pr = \nu/\kappa$ and $g_i^{eq}(r, t)$ equates to:

$$g_i^{eq}(r, t) = T t_p \chi, \tag{12}$$

with T representing the temperature. The macroscopic temperature variable is given by:

$$T = \sum_i g_i. \tag{13}$$

The advantages of the SRT method are that the implementation is straightforward for all $DxQy$ models and that the collision kernel is well suited for HPC simulations. The Reynolds number range and its quasi-incompressibility ansatz in the derivation of the equilibrium distribution function (see [27]) are, however, due to stability limitations and fixed collision frequencies ω_F and ω_T restricted to rather small Mach and Reynolds numbers.

2.2.2 MRT method

Unlike the SRT method, the MRT method [12] relaxes in momentum space and introduces individual collision frequencies for the various moments resulting in higher numerical stabilities [42]. Various physical processes in fluids such as viscous transport can be approximately described by mode coupling. The modes are directly related to the moments of the PPDFs. Since the collision frequencies of the moments are directly related to various transport coefficients, each mode can be controlled independently. This overcomes the fixed Prandtl number issue that the SRT models suffer from. The MRT equation can be written in vector notation and is given by:

$$\mathbf{f}(r+\delta r, t+\delta t) = \mathbf{f}(r,t) - \underline{\mathbf{M}}^{-1}\underline{\mathbf{K}}_{MRT} \cdot \left[\mathbf{m}(r,t) - \mathbf{m}^{eq}(r,t)\right], \tag{14}$$

with $\mathbf{f}$ being the vector of the PPDFs and moment and relaxation matrices $\underline{\mathbf{M}}$ and $\underline{\mathbf{K}}_{MRT}$, and vectors $\mathbf{m}^{eq}$ and $\mathbf{m}$. The relaxation matrix $\underline{\mathbf{K}}_{MRT}$ is a diagonal matrix holding the various collision frequencies. Example vectors and matrices for the $D2Q9$ MRT model are given in Appendix B.

Note that setting the diagonal elements of $\underline{\mathbf{K}}_{MRT}$ to ω_F yields the SRT method. The calculations of the macroscopic variables and the propagation are performed analogue to the SRT method, see Eqs. (6)–(8). That is, the MRT method only differs from the SRT method by the collision step. A stability analysis that considers the eigenvalues of the spectral matrix of the Fourier transform of Eqs. (1) and (14) underlines the stability advantages of the MRT method over the SRT method [42]. That is, the MRT method is advantageous for higher Reynolds number flows. Computationally, it is due to the required matrix operations more expensive than the SRT method. On modern supercomputers with decent vector units, the performance difference can, however, be considered negligible if the compiler is able to efficiently vectorize the code.

2.2.3 CLB method

Similarly to the MRT method, the CLB method [22] also relaxes in moment space, which reconstructs, however, the high-order non-hydrodynamic moments of the discrete velocity set up to the fourth order in a cascade from lower order moments. Short wave-length oscillations at higher Reynolds numbers are captured by these higher order moments yielding a stable numerical scheme. The advancement of the state vector of the PPDFs $\mathbf{f}$ is given by [22]:

$$\mathbf{f}(r+\delta r, t+\delta t) = \mathbf{f}(r,t) + \underline{\mathbf{K}}_{CLB} \cdot \mathbf{k} \tag{15}$$

where $\underline{\mathbf{K}}_{CLB} = (\mathbf{K}_1, \dots, \mathbf{K}_y)$ is a orthogonal transformation matrix that transforms the configuration space to the moment space to obtain Galilean invariance, e.g., the first hydrodynamic macroscopic quantities can be obtained by:

$$(\rho, \rho v_1, \dots, \rho v_x)^T = \mathbf{f} \cdot (\mathbf{K}_1, \dots, \mathbf{K}_{x+1}), \tag{16}$$

and $\mathbf{k}$ is the moment vector. The matrix $\underline{\mathbf{K}}_{CLB}$ and the vector $\mathbf{k}$ are exemplarily given in Appendix C for the $D2Q9$ model.

Concerning the limitations of the SRT method for $\nu \to 0$, or in other words $Re \to \infty$, the CLB method allows for higher numerical stability by enabling a decrease of the viscosity by many orders of magnitude as compared to the original model [22].

2.2.4 RLB method

Using the SRT method, the symmetry properties of the PPDFs are not necessarily fulfilled to reach the hydrodynamic limit. Therefore, Latt and Chopard [43] introduced the RLB method, which introduces a pre-collision regularization step to restore this symmetry. This regularization leads to enhanced stability and accuracy in the hydrodynamic regime of the employed scheme. The Chapman–Enskog expansion expands the PPDFs in powers of the Knudsen number $\epsilon = l_f/L$, where l_f is the mean free path and L a characteristic length. This leads to:

$$f_i(r,t) = f_i^{(0)}(r,t) + \epsilon f_i^{(1)}(r,t) + \epsilon^2 f_i^{(2)}(r,t) + \dots, \tag{17}$$

and $f^{eq}(r,t) = f_i^{(0)}(r,t)$ at zeroth order. The non-equilibrium parts are then given by:

$$f_i^{neq}(r,t) = f_i(r,t) - f_i^{eq}(r,t). \tag{18}$$

The non-equilibrium part of the PPDFs can furthermore be expressed as [11]:

$$f_i^{neq}(r,t) \approx f_i^{(1)}(r,t) = -\frac{\delta t}{\omega_F c_s^2}\left(\xi_{i,a}\xi_{i,b} - c_s^2\delta_{ab}\right) \cdot \frac{\partial \rho v_b}{\partial x_a}. \tag{19}$$

Inserting this into the non-equilibrium momentum flux tensor [cf. Eq. (9)] leads to:

$$\Pi_{ab}^{neq} = \Pi_{ab} - \sum_i \xi_{i,a}\xi_{i,b} f_i^{eq}(r,t) \tag{20}$$

$$\approx \sum_i \xi_{i,a}\xi_{i,b} f_i^{(1)}(r,t) \tag{21}$$

$$= -\frac{c_s^2}{\omega_F}(\nabla \cdot \rho \mathbf{v}). \tag{22}$$

Combining Eqs. (19) and (22) yields the regularization term:

$$f_i^{(1)}(r,t) = \frac{\delta t}{2c_s^4}\left(\xi_{i,a}\xi_{i,b} - c_s^2\delta_{ab}\right) \cdot \Pi_{ab}, \tag{23}$$

which can be used in $f_i^{reg}(r,t) = f_i^{eq}(r,t) + f_i^{(1)}(r,t)$ to set $\hat{f}_i(r,t) = f_i^{reg}(r,t)$ prior to the next collision.

2.2.5 Large-eddy simulations

LES computations employ the Smagorinsky sub-grid scale (SGS) model for LB methods [33, 68]. Spatial filtering is performed using coarsened meshes. To account for the filtered scales, the turbulent viscosity ν_t is added to the collision frequency (see Eq. (2)):

$$\omega_F^{SGS} = \frac{c_s^2}{(\nu + \nu_t) + \delta t c_s^2/2} \tag{24}$$

with Smagorinsky's constant C_s. The turbulent viscosity is given by:

$$\nu_t = (C_s \delta r)^2 \sqrt{2\bar{S}_{ab}^2}, \tag{25}$$

where $\bar{S}_{ab}$ is the filtered strain tensor, which can be obtained by:

$$\bar{S}_{ab} = \frac{\omega_F}{2\rho c_s^2} \sum_i \left[f_i(r,t) - f_i^{eq}(r,t)\right] \xi_{i,a}\xi_{i,b}. \tag{26}$$

2.2.6 Grid refinement

Simulations on multiple octree hierarchy levels are performed using the method of Dupuis and Chopard [14], i.e., by executing restriction and prolongation operations on succeeding levels l_κ and $l_{\kappa+1}$ in the corresponding overlapping regions. The method splits missing interpolated incoming PPDFs $\tilde{f}_i(r,t)$ from other levels into equilibrium and non-equilibrium parts, see Eqs. (18) and (19). The relation of the non-equilibrium PPDFs on the fine and the coarse level can then be expressed as:

$$\begin{aligned}\frac{f_{i,\kappa+1}(r,t)^{neq}}{f_{i,\kappa}(r,t)^{neq}} &= \frac{\delta t_{\kappa+1}}{\delta t_\kappa} \cdot \frac{\omega_{F,\kappa}}{\omega_{F,\kappa+1}} \frac{\left(\xi_{i,a}\xi_{i,b} - c_s^2\delta_{ab}\right)}{\left(\xi_{i,a}\xi_{i,b} - c_s^2\delta_{ab}\right)} \\ &= \frac{\delta r_{\kappa+1}}{\delta r_\kappa} \cdot \frac{\omega_{F,\kappa}}{\omega_{F,\kappa+1}}\end{aligned} \tag{27}$$

with $\delta t_{\kappa+1}/\delta t_\kappa = \delta r_{\kappa+1}/\delta r_\kappa$. For the transformations from fine to coarse and coarse to fine, this yields:

$$\begin{aligned}f_{i,\kappa+1}(r,t) =& \tilde{f}_i^{eq}(r,t) + (\tilde{f}_{i,\kappa}(r,t) - \tilde{f}_i^{eq}(r,t)) \\ &\cdot \underbrace{\frac{\delta r_{\kappa+1}}{\delta r_\kappa} \cdot \frac{\omega_{F,\kappa}}{\omega_{F,\kappa+1}}}_{\Theta_{\kappa+1}},\end{aligned} \tag{28}$$

$$\begin{aligned}f_{i,\kappa}(r,t) =& f_i^{eq}(r,t) + \left(f_{i,\kappa+1}(r,t) - f_i^{eq}(r,t)\right) \\ &\cdot \underbrace{\frac{\delta r_\kappa}{\delta r_{\kappa+1}} \cdot \frac{\omega_{F,\kappa+1}}{\omega_{F,\kappa}}}_{\Theta_\kappa}.\end{aligned} \tag{29}$$

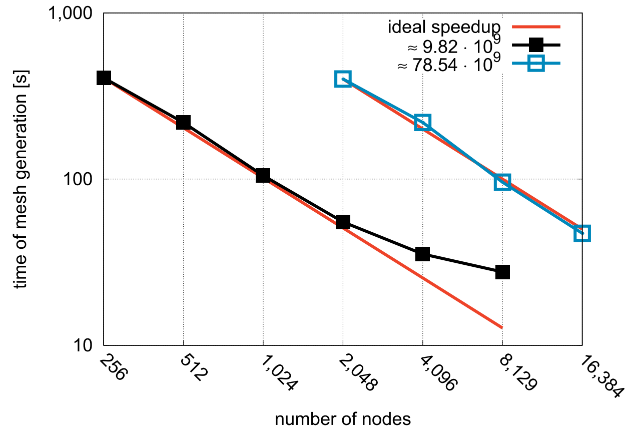
Keeping the viscosity constant across levels leads to the transformation factors $\Theta_{\kappa,\kappa+1}$, which depend on the ratios of the grid distances and the relaxation times on l_κ and $l_{\kappa+1}$. The same scheme can be applied to the MDF approach, however, using the same heat conduction coefficient on both levels and $\omega_{T,\kappa}$ and $\omega_{T,\kappa+1}$.

2.2.7 Boundary conditions

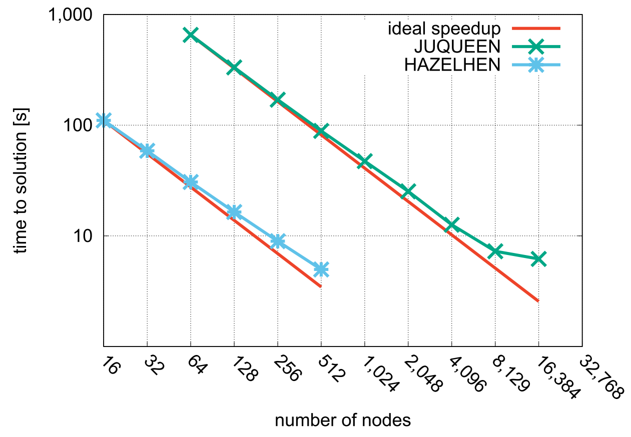
Various boundary conditions can be applied to simulate a large spectrum of numerical setups. In more detail, multiple von Neumann and Dirichlet conditions for in- and outflows, and periodic boundary conditions are implemented in ZFS. This also includes sponge layered, Saint-Venant/Wantzel, and pressure boundary conditions [51]. That is, for Dirichlet conditions, e.g., for the prescription of a velocity profile at an inlet or a pressure at the outlet, Eq. (3) is equated for a given velocity or density. An adaptive pressure-based outlet condition adapts according to the local Reynolds number. Such a boundary condition is frequently used at an outlet in combination with a Saint-Venant/Wantzel at an inlet in respiratory flow simulations to imitate the pressure drop caused by the expansion of the diaphragm; see Sect. 3.1. A von Neumann condition is fulfilled by extrapolating from interpolated values, e.g., velocity or density, of the inlet-/outlet-nearest neighbor cells. The Saint-Venant/Wantzel condition extrapolates the momentum. No-slip wall boundary conditions are at least of second-order accuracy. Among the most frequently used wall boundary conditions is the interpolated bounce-back method from Bouzidi et al. [6], which measures for cells intersecting the geometry the distance q from the cell center to the wall and uses this information to interpolate the bounced-back PPDF. The interpolation scheme is chosen based on $q < 0.5 \cdot \delta r$ and $0.5 \cdot \delta r \le q \le \delta r$. Furthermore, schemes from Ginzburg and D'Humières [24] and Yu et al. [76] are implemented. In the former, a multi-reflection boundary condition, which employs PPDFs at three nodes to find the unknown PPDFs at the boundary condition after a bounce-back, is proposed. In the latter, a double interpolated bounce-back using only a single equation for all values of q is applied. This boundary condition finds a PPDF at a location between the two or three wall-nearest nodes, which during the bounce-back will exactly stream to the wall. Subsequently, it performs the bounce-back and then interpolates the missing PPDF based on the PPDF at the wall.

2.3 Performance analysis

The scalability of the code is tested on two HPC systems. The CRAY HAZEL HEN system is located at the High-Performance Computing Center (HLRS) Stuttgart, Germany. The system consists of 7712 dual socket nodes containing each 2 Intel Haswell E5-2680v3 CPUs, each with 12 cores clocked at 2.5 GHz. The system has a peak performance of 7.4*PFlops* for 185, 088 cores. The nodes contain 128 GB of RAM. Parallel I/O is implemented via a Lustre File System (LFS), see [77]. The IBM BlueGene/Q system JUQUEEN [69] is located at the Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich, Germany, and consists of 28,672 nodes containing IBM PowerPC A2 CPUs at 1.6 GHz, 16



(a) Strong scalability of the massively parallel grid generator on JUQUEEN for the generation of $\approx 9.82 \cdot 10^9$ and $\approx 78.54 \cdot 10^9$ cells.



(b) Strong scalability of the SRT LB on JUQUEEN and HAZEL HEN for $t = 100$ iterations on $\approx 1.225 \cdot 10^9$ cells.

Fig. 3 Strong scalability performance results of the massively parallel grid generator on JUQUEEN and the SRT LB method on the JUQUEEN and HAZEL HEN systems

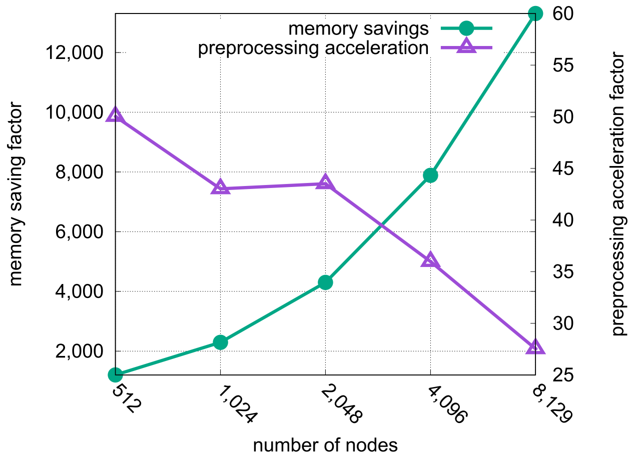


Fig. 4 Memory saving factor (left ordinate) and preprocessing acceleration factor (right ordinate) for computations on increasing node counts using parallelized geometries related to resources allocated on 512 JUQUEEN nodes using a serial geometry

cores, and 16 GB of RAM per node. The overall peak performance is 5.9*PFlops*. Due to its four-way SMT hardware threaded floating point units, it is capable of running a maximum number of 4 OpenMP threads per core. The JUQUEEN system uses the IBM LoadLeveler as job scheduler and has a 5D Torus network with a 40 GB/s bandwidth.

Figure 3a shows the performance of the mesh generation on the JUQUEEN system for two cases with $\approx 9.82 \cdot 10^9$ and $\approx 78.54 \cdot 10^9$ cells. The domains are cubic and periodic in all Cartesian directions. Both meshes have a base level of $l_\alpha = 7$. The smaller mesh is refined up to $l_\beta = 11$ and the finer mesh up to $l_\beta = 12$. From Fig. 3a, it can be seen that the mesh generation on for the smaller case scales well up to 2048 nodes. Defining the ratio of the expected run time under continuous bisection of the first run time on the lowest number of nodes and the actual measured time for the individual node counts as computational efficiency ϵ in percent, a value of $\epsilon = 92.06\%$ is achieved on 2048 nodes. The performance increase drops to $\epsilon = 46.01\%$ on 8192 nodes. A slightly superlinear scaling behavior up to 16,384 nodes is visible for the finer case, where $\epsilon = 106.0\%$ is reached. It should be noted that despite the scalability drops for the smaller case for higher node counts, the meshing process only requires $27.63s$ and $44.94s$ for the small and fine cases on 8192 and 16,384 nodes.

Figure 3b shows the results of a strong scaling experiment on both systems HAZEL HEN and JUQUEEN using the SRT method on a cubic domain with periodic boundary conditions in all directions. The computational mesh is uniformly refined, has levels from $l_\alpha = 8$ to $l_\beta = 10$, and consists of $\approx 1.225 \cdot 10^9$ cells. The analyses are performed using $n_H = 16, \dots, 512$ compute nodes on HAZEL HEN, each with 24 MPI ranks per node, and $n_J = 64, \dots, 16,384$ on JUQUEEN using 16 MPI ranks per node. The simulations are advanced for $t = 100$ iterations. Obviously, the computation shows a very good scalability up to 512 nodes on HAZEL HEN with efficiencies of $\epsilon = \{84.15\%, 77.1\%, 69.46\%\}$ on 128, 256, and 512 nodes. On JUQUEEN, the code scales up to 16,384 nodes and has efficiencies of $\epsilon = \{81.06\%, 70.4\%, 41.26\%\}$ on 4096, 8192, and 16,384 nodes.

To reduce the memory footprint of simulations and to accelerate the preprocessing performance of simulations, parallelized geometries are employed. Figure 4 shows the memory saving factors and preprocessing accelerator factors employing parallelized geometries in contrast to using serial geometries. The results are obtained for the simulation of the flow in a respiratory tract using a mesh consisting of $266.5 \cdot 10^6$ cells with levels $l_\alpha = 9$, $l_\beta = 10$, and $l_\gamma = 12$. The corresponding geometry consists of $7 \cdot 10^6$ triangles and consumes 1239*MB* in serial. The memory graph is based on the estimate that a geometry allocates twice as much space under a doubling of the number of nodes. The acceleration of the preprocessing bases on a comparison to a test run using a serial geometry on 8192 nodes. From Fig. 4, it is obvious that using a parallelized geometry massively reduces allocated space, i.e., from 512 up to 8192 nodes saving factors of 1204 and 13,306 are obtained. The preprocessing, which is for the serial geometry dominated by I/O, can be decreased by factors of 50 and 27.5 for 512 and 8192 nodes. That is, on one hand, parallelized geometries allow to have more memory available for the computation and, on the other hand, reduce the effort in preparing a simulation.

For further ideas to accelerate LB codes, e.g., for exascale computing or to port to GPUs, the reader is referred to [4, 70, 73].

3 Applications

To show the applicability of the methods presented in Sect. 2 for complex flows, three simulation examples are presented in the following. Section 3.1 discusses some results for the flow in a complex geometry such as the human respiratory system. Subsequently, the applicability for technical applications, i.e., for the simulation of the flow in porous media such as in GDLs of fuel cells and around an airplane landing gear configuration, are shown in Sects. 3.2 and 3.3.

3.1 Flow in the human respiratory tract

The flow in the human respiratory tract is quasi-incompressible and is mainly in the low Reynolds number and low Mach number regime, i.e., the $D3Q27$ SRT LB method is well suited for the simulation of such flows. Considering respiration, it is important, from a fluid mechanics point of view, to understand the functionalities of the human nasal cavity and what pathologies, or in fluid mechanics context, what flow phenomena might cause their reduction [51]. That is, the pressure loss along the airway can be used as an indication for strenuous respiration. A local consideration might even lead to the identification of anatomical structures causing diminished respiration. The distribution of the wall-shear stress can be used to find potential locations of inflammations, while the temperature distribution and the heating capability classify a nasal cavity to be protective for the lungs. The epiglottis and the larynx might lead to jets at inspiration [49]. Furthermore, particle deposition behavior in the nose as well as in the lung is of high interest, i.e., where and why, e.g., fine dust particles or Diesel aerosols deposit in the airway [55]. The human organism is a highly complex structure and simulations in the human respiratory tract are quite a challenge for today's simulation algorithms. ZFS, however, is capable of simulating such flows with LB methods on octree-based meshes. Especially large geometries that may be the output of algorithms extracting surfaces from computer tomography data [54] might become large in size and necessitate a geometrical parallelization approach as presented in Sects. 2.1 and 2.3.

Figure 5 shows the results of a simulation embedded in the original computer tomography data set that was computed on the JUQUEEN system. The computational mesh consists of $266.5 \cdot 10^6$ cells and has levels $l_\alpha = 9$, $l_\beta = 10$, and $l_\gamma = 12$, cf. Sect. 2.3. The Reynolds number, which is based on the hydraulic diameter of the trachea D_T, the kinematic viscosity of air, and the velocity corresponding to a volume flux of 360*ml/s* is $Re = 1,515$. The flow in the nasal cavity is primarily in the laminar regime. Depending on the local Reynolds number, i.e., the local shape of the anatomy and the respiration velocity, the flow can, however, become transitional [51]. Further downstream, the epiglottis and the larynx are responsible for the formation of a jet. The magnification in Fig. 5 shows that the flow becomes transitional in this region before it turns laminar again in the trachea. For visualizing unsteady vortical structures, the Δ-criterion [10] defined by

$$\Delta = \left(\frac{Q}{3}\right)^3 + \left[\frac{\det(\nabla \otimes \mathbf{v})}{2}\right]^2 > 0 \tag{30}$$

has been evaluated in the magnification. It is based on the Q-criterion [34]:

$$Q = \frac{1}{2}\left(|\Omega|^2 - |S|^2\right) > 0, \tag{31}$$

where

$$S = \frac{1}{2}\left[\nabla \otimes \mathbf{v} + (\nabla \otimes \mathbf{v})^T\right] \tag{32}$$

$$\Omega = \frac{1}{2}\left[\nabla \otimes \mathbf{v} - (\nabla \otimes \mathbf{v})^T\right] \tag{33}$$

are the strain and the vorticity tensor, $\mathbf{v} = (v_1, v_2, v_3)^T$ is the velocity vector, and $|\cdot|$ denotes the Frobenius norm.

Quantitative results are depicted in Fig. 6. As it is in many cases the nasal cavity which determines the respiratory capability, the total pressure loss from the nostrils to the pharynx, i.e.,

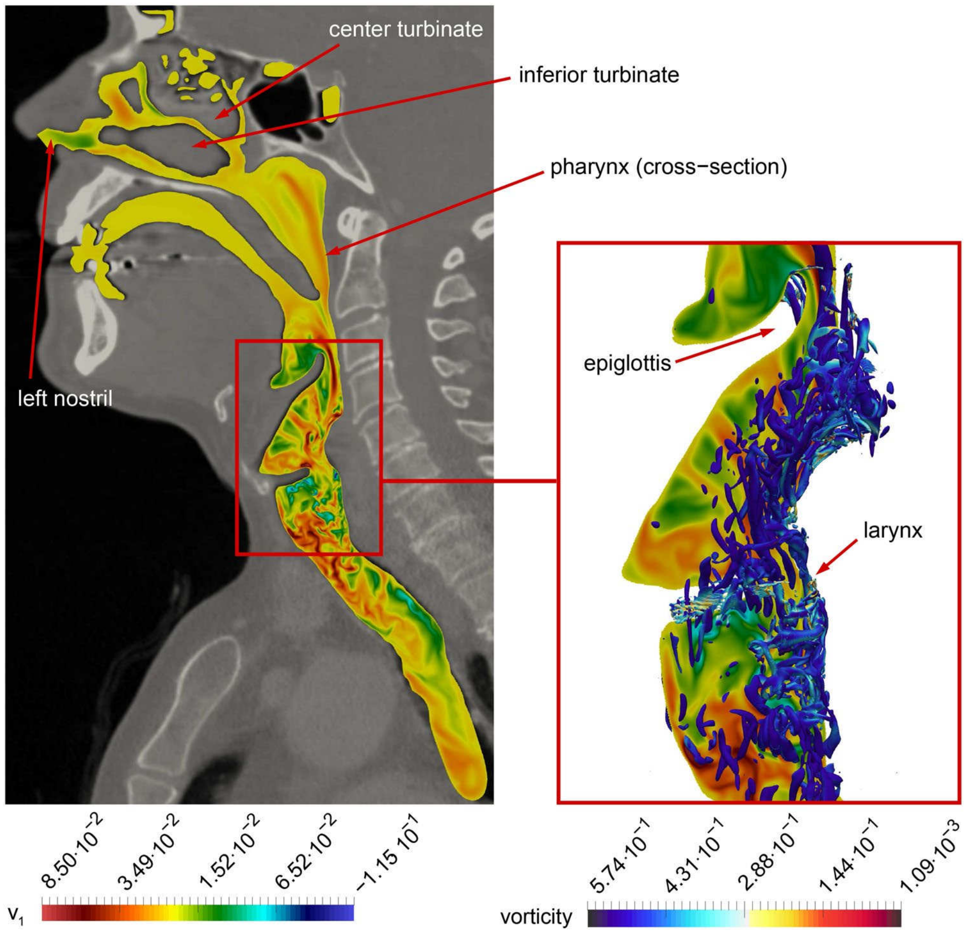


Fig. 5 Flow in the human respiratory tract. The cross section is colored by the v_1 velocity component of the flow. The magnification shows the larynx region. Emerging vortical structures are visualized by contours of the Δ-criterion. The contours are colored by the vorticity

$$\delta\hat{p}_{t,x_1} = \left(\underbrace{\tilde{p}_s(x_1) + \frac{\tilde{\rho}(x_1)}{2} \cdot |\tilde{\mathbf{v}}(x_1)|^2}_{\tilde{p}_d(x_1)} \right) - \left(\underbrace{\tilde{p}_s(x_2) + \frac{\tilde{\rho}(x_2)}{2} \cdot |\tilde{\mathbf{v}}(x_2)|^2}_{\tilde{p}_d(x_2)} \right), \tag{34}$$

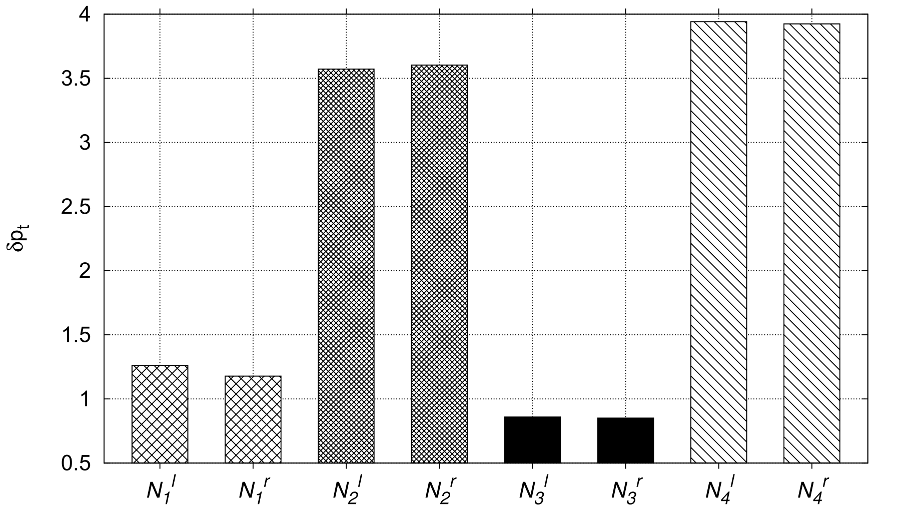
is juxtaposed to results in [51] in Fig. 6a. That is, the pressure loss of the nasal cavities $\mathcal{N}$ with indices $1, \ldots, 3$, is compared to the current case with index 4 for the left l and right r nasal cavity. In Eq. (34), the quantities $x_1 \in \{l, r\}$ and x_2 represent the locations of the left and right nostrils and the pharynx location (see Fig. 5). The static pressure $\tilde{p}_s$ and the dynamic pressure $\tilde{p}_d$ are averaged over their corresponding cross-sectional areas. Note that to allow for a comparison, the total pressure has been normalized according to [51], that is:

$$\delta p_{t,x_1} = \frac{1}{Ma^2}\Big[\big(\tilde{\rho}(x_1) - \tilde{\rho}(x_2)\big) + \frac{3}{2}\big(\tilde{\rho}(x_1) \cdot |\mathbf{v}(x_1)|^2 - \tilde{\rho}(x_2) \cdot |\mathbf{v}(x_2)|^2\big)\Big]. \tag{35}$$

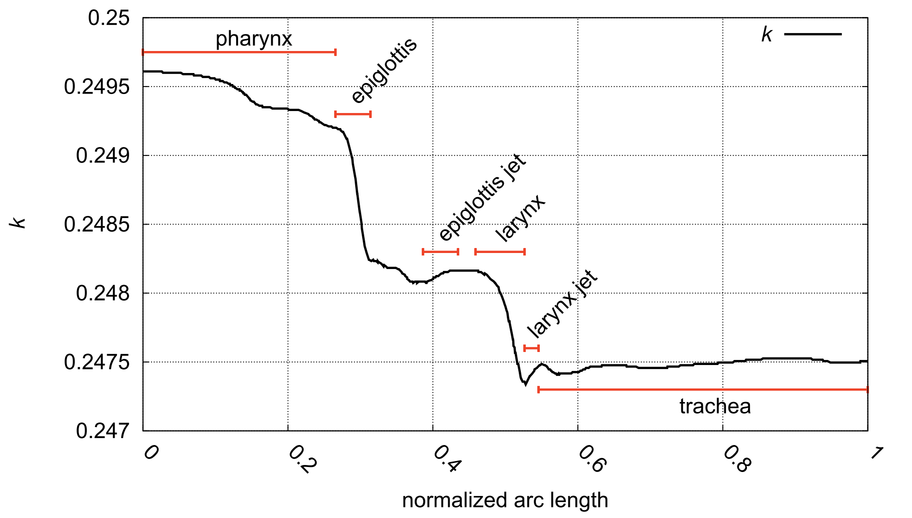
Case $\mathcal{N}_1$ is considered a healthy cavity, $\mathcal{N}_2$ a pathological case with swollen turbinates and a bend septum, and $\mathcal{N}_3$ is a case which underwent a septoplastic surgery. Obviously, the respiratory capability of case $\mathcal{N}_4$ is below that of the healthy and even the pathological case. It can be seen that strenuous respiration is present in both the left and right nasal cavity. It should, however, be mentioned that for $\mathcal{N}_{1,\ldots,3}$ a volume flux of 250 ml/s was assumed, which explains the comparative increased pressure loss for case $\mathcal{N}_4$.

The laminarization process along the airway is analyzed by considering the turbulent kinetic energy $k = 1/2 \cdot (\overline{v_1'^2} + \overline{v_2'^2} + \overline{v_3'^2})$ with temporally averaged non-dimensional fluctuating squared quantities $\overline{v_a'^2}$. It is sampled along the geometrical centerline of the anatomy from the pharynx region to the end of the trachea. The results along the arc length of the curve are shown in Fig. 6b. It can be seen that the flow is already quite energetic when reaching

Fig. 6 Analysis of the flow in the respiratory tract by comparing the pressure loss in the nasal cavity to literature values and by considering the turbulent kinetic energy along a centerline of the lower airway



(a) Comparison of the total pressure loss δp_t along the nasal cavities $\mathcal{N}_{1,\ldots,3}$ from [51] and the current case $\mathcal{N}_4$ for the left l and right r cavity.



(b) Turbulent kinetic energy k along the arc length of a centerline of the geometry from the pharynx to the end of the trachea.

the epiglottis, cf. Fig. 5. The corresponding restriction helps laminarizing the flow before the turbulent kinetic energy increases in the epiglottis jet region. Subsequently, the larynx constriction supports again laminarization and the larynx jet again increases the energy. In the trachea, the energy level stays at a low and almost constant level before the flow finally enters the lung.

3.2 Flow in gas diffusion layers of fuel cells

In fuel cells, it is important to have a uniform supply of oxygen and hydrogen in the reactive layer to reliably produce electricity. Therefore, GDLs [21, 41] are placed between the gas supply channels and the reactive layer. Such GDLs consist of porous material composed of dense arrays of carbon fibers, sometimes treated with hydrophobic substances to allow for enhanced water droplet transport across the layers. The optimal structure to homogenize the flow is, however, difficult to find, which is why numerical simulations are employed to understand the fundamental aspects of gas transport across the layers. The following simulations make use of the LB method with the SRT $D3Q27$ model using parallelized geometries.

Figure 7 shows the simulation setup, which covers 27 GDLs embedded in a $1mm^3$ cube. For the inlet, a uniform velocity profile is prescribed via a Dirichlet BC and the

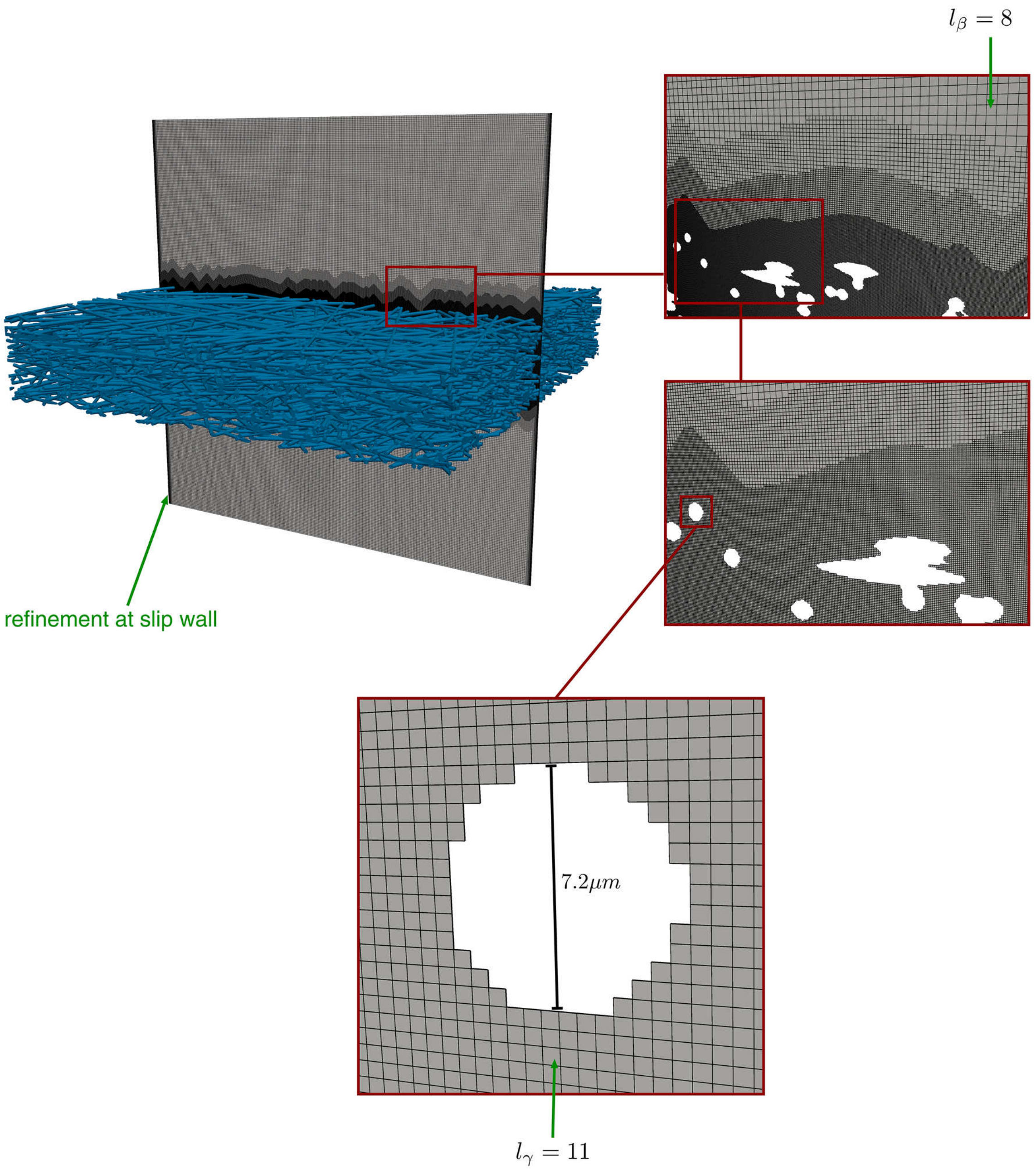


Fig. 7 Geometry and computational mesh of the GDL simulation case. The flow in a total of 27 layers is simulated. The computational mesh consists of hierarchy levels $l_\alpha = 7$, $l_\beta = 8$, and $l_\gamma = 11$ and of $2.032 \cdot 10^9$ cells

density is extrapolated using a von Neumann BC. In contrast, the outlet extrapolates the velocity and the density is prescribed. At the outer wall, a slip condition is used and the no-slip wall BC for at the fiber surfaces is realized via the interpolated bounce-back approach from [6]. The Reynolds number, which is based on the fiber diameter of $D = 7.2\mu m$, the kinematic viscosity of hydrogen, and a velocity $v_G = 1.8 \cdot 10^{-3} m/s$, is $Re_G = 6.47 \cdot 10^{-5}$. The computational mesh consists of hierarchy levels $l_\alpha = 7$, $l_\beta = 8$, and $l_\gamma = 11$ and of $2.032 \cdot 10^9$ cells. Figure 8 shows some results of the simulation, i.e., the velocity magnitude of the gas flow across the GDL is depicted via distributions in cross sections in the center of the GDL (horizontal and vertical) and right downstream of the GDL. The results match the small Reynolds number, i.e., the flow is dominated by diffusive effects. This is also corroborated by the smeared velocity distribution in the cross section right downstream of the GDL.

The simulations were performed on the JURECA system [40] at JSC Jülich. The JURECA supercomputer consists of

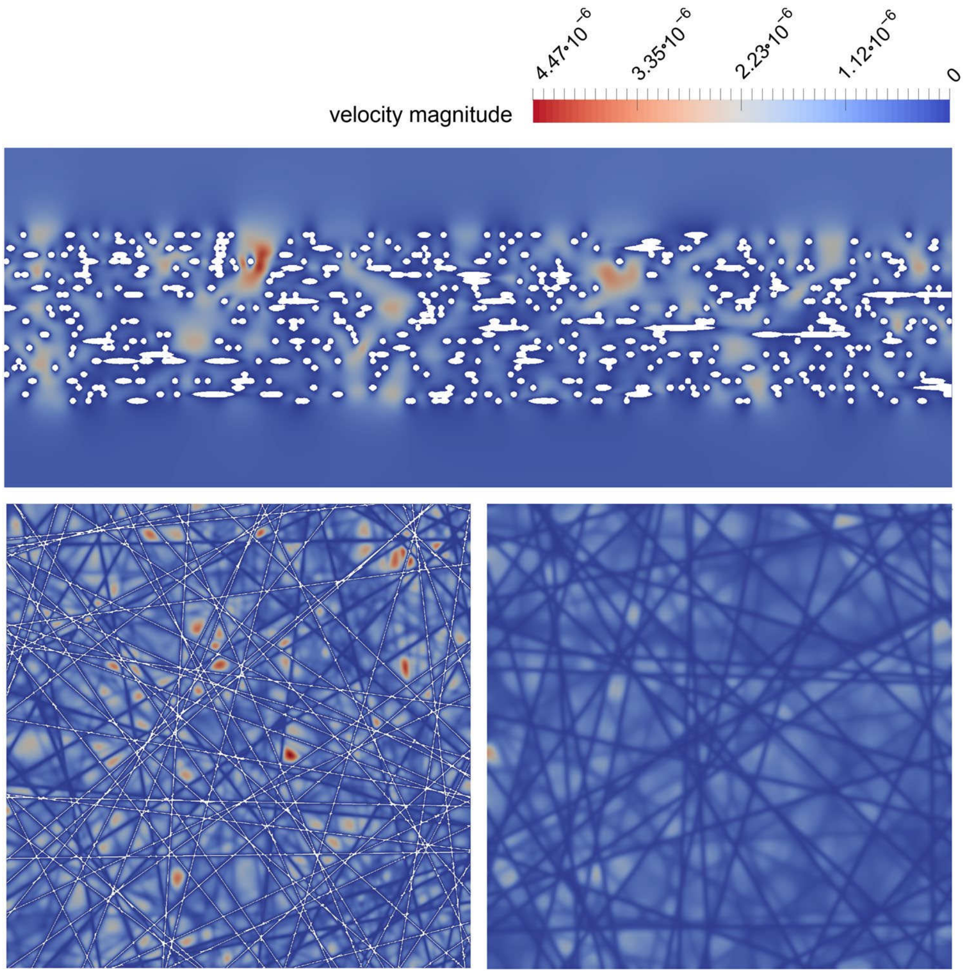


Fig. 8 Simulation results of the GDL simulation. The different cross-sections are colored by the velocity magnitude. Cross-section locations from left to right: vertical in the center of the GDL; horizontal in the center of the GDL; directly downstream of the GDL

1872 compute nodes, each equipped with a dual-socket system consisting of two Intel Xeon E5-2680 v3 Haswell CPUs. The CPUs are clocked at 2.5 GHz and have 12 cores each. That is, the whole system consists of 44,928 cores. 1605 compute nodes are equipped with 128 GB, 128 with 256 GB, and 64 with 512 GB DDR4 memory clocked at 2133 MHz. 75 of the compute nodes are furthermore equipped with two NVIDIA K80 GPUs each. Additionally, the JURECA has a booster module with 1640 compute nodes with one Intel Xeon Phi 7250-F Knights Landing CPUs (KNL) per node. Each KNL has 68 cores clocked at 1.4 GHz and is equipped with 96 GB memory plus 16 GB MCDRAM high-bandwidth memory. Altogether, the booster module has 111,520 CPU cores. The overall CPU, GPU, and KNL peak performances of JURECA are 1.8, 0.44, and 5 Petaflop.

3.3 Flow around an airplane landing gear configuration

To show the applicability of the LB method to subsonic aerodynamic problems, simulations are performed on the JURECA system at JSC for the flow around a front airplane landing gear configuration[2] as shown in Fig. 9a. The landing gear can be considered typical for vertical take-off landing (VTOL) jet airplanes with double front gear wheels. The MRT method described in Sect. 2.2.2 with the $D3Q27$ discretization scheme is used. The REYNOLDS number of this problem is $Re = v_\infty \cdot H/\nu = 0.5 \cdot 10^6$. It is based on the freestream velocity $v_\infty = 22.725m/s$, the landing gear height $H = 1.20m$ (see label in Fig. 9a), and the viscosity of air at a temperature of $T = 267.039K$ of $\nu = 15.15 \cdot 10^{-6} m^2/s$. The REYNOLDS number based on the diameter of the main hydraulic damper $d = 0.062m$ is $Re_d = 25,800$. Although the velocity and, hence, the REYNOLDS number are quite low compared to real inflight conditions, it represents a characteristic REYNOLDS number in the transition phase from vertical ascent to accelerated horizontal flight. Figure 9b shows

[2] The STL of the landing gear has been downloaded from https://grabcad.com/library/nose-landing-gear-10 and has been generated by GrabCAD user `jissi`. All content submitted by the Contributors To GrabCAD is for non-commercial use only, unless otherwise agreed in writing with the Contributor.

the computational mesh of the simulation which is uniformly refined up to a base level $l_\alpha = l_\beta = 7$. The mesh is furthermore boundary-refined in the vicinity of the gear and patch-refined in the wake of the gear up to level $l_\gamma = 11$. The inset of Fig. 9b shows a magnification of the computational mesh in the vicinity of the top airplane mount joint. The mesh has a total number of cells of $191.5 \cdot 10^6$. On level l_γ, the resolution is at $\delta r = 1.479mm$. To show that this resolution is sufficient, the non-dimensional wall distance y^+ is calculated on a temporally averaged simulation data set. The mean velocity components $\bar{v}_a$ and the mean density $\bar{\rho}$ are obtained by averaging the solution after $\iota_s = 300 \cdot 10^3$ LB iterations for another $\bar{\iota} = 200 \cdot 10^3$ LB iterations. The non-dimensional wall distance is given by:

$$y^+ = \frac{v_* \cdot r_w}{\nu}, \tag{36}$$

where $v_* = \sqrt{\tau_w / \rho}$ is the friction velocity with wall-shear stress $\tau_w = \mu(\partial v_\parallel / \partial r_w)$, and r_w, μ, and $v_\parallel$ are the coordinate normal to the wall, the dynamic viscosity, and the wall-tangential velocity component. The minimal distance $\phi(i)$ between field data points i and the landing gear geometry employs the signed distance calculation method by Baerentzen and Aanaes [2]. The wall-shear stress is linearly approximated by $\tau_w(i) \approx \mu[v_\parallel / \phi(i)]$. Together with Eq. (36), this leads to:

$$y^+(i) \approx \sqrt{\frac{\phi(i)}{\nu} \cdot v_\parallel}. \tag{37}$$

Figure 11a shows contours of $\phi(i)$ colored by $y^+(i)$. The maximum y^+ that is found in the wall-nearest cells, i.e., at a maximum distance of $\delta r/2$ on level l_γ is $max\{y^+(i)\} = 4.086$. Although $max\{y^+(i)\} > 1$, the majority of the cells has a value of $y^+(i) \leq 1$. Furthermore, since the flow is mainly shear-driven, the resolution of the computational mesh can be considered sufficient. The corresponding contour at $\phi = \delta r/2$, colored by y^+, is shown in Fig. 11b. Obviously, the maximum y^+ is found on the left and right of the stagnation lines on the main and minor dampers, the hydraulic dampers, the wheels, and on top and on the bottom of the wheels.

Figure 10a shows the flow in a cross section parallel to the main flow direction. The cross section is colored by the velocity magnitude. Obviously, the complex geometry in combination with the Reynolds number leads to a transition of undisturbed flow upstream of the landing gear to highly vortical structures in the wake region. This is also visible in Fig. 10b, which shows contours of the Δ-criterion colored by the velocity magnitude. The two insets in Fig. 10b show the magnifications of these structures in the top region of the geometry and in the vicinity of the small hydraulic hoses, which resemble hair-pin vortex-like shapes that detach in the near-wall region.

To quantitatively analyze the flow, the Reynolds stresses are computed over the LB iterations $\bar{\iota}$. The turbulent kinetic energy k and the mean velocities $\bar{v}_a$ along a line L^* that runs from the inlet to the outlet and crosses the center of the axle of the wheel construction are plotted over the normalized length of the line. The position of the line is shown in Fig. 10a. Figure 12 shows the strong increase of the turbulent kinetic energy in the impingement region of the wheel axle. The center of the wheels, i.e., the rotational axis, is located at $p^*_w = 0.3445$ (see Fig. 10a) and the wheel range is $p^*_{\delta w} = [0.2974, 0.3919]$. A slight decrease of k is visible in the wake region, before it normalizes towards an equilibrium further downstream. While the mean axial flow component $\bar{v}_1$ strongly decreases towards the object, an increase of $\bar{v}_2$ and $\bar{v}_3$ due to the geometry-induced side-way displacement of the flow in this regions is visible. Downstream of the gear, $\bar{v}_3$ experiences a stronger decrease than $\bar{v}_2$, which is due to the left and right mounted wheels forcing the flow to move inwards. The turbulent flow is, furthermore, analyzed by means of the power spectral density *PSD* calculated for the velocity components v_a and the density ρ at probe locations $p^*_{1,2} \approx 0.4$ (see Fig. 10a). Probe p^*_1 is located in the center of the geometry, while p^*_2 is located right behind the left wheel. Comparing the *PSD*s in Fig. 13a and Fig. 13b shows that the flow at p^*_2 is slightly more energetic in the higher wave number range and experiences a stronger drop towards non-resolved frequencies. Especially in the lower frequency range, the energy content at p^*_1 for component v_3 is higher than for p^*_2, which is probably due to the same effect as for the turbulent kinetic energy k at this location. Three wave numbers $k_w(p^*_1) = \{0.176 \cdot 10^{-3}, 0.251 \cdot 10^{-3}, 0.325 \cdot 10^{-3}\}$, as highlighted by the three vertical lines in Fig. 13a, seem to dominate the low wave number range for v_3. At p^*_2, a slight bump is visible for velocity component v_1 at $k_w(p^*_2) = 0.113 \cdot 10^{-3}$ (indicated by the vertical line in Fig. 13b). A similar behavior, although with a slightly higher wave number, is present for p^*_1.

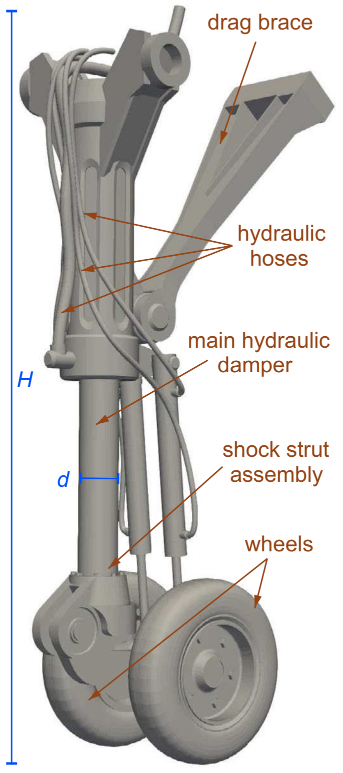
The results show the MRT method together with mesh refinement method and the Smagorinsky SGS model for LES computations to be a suitable combination to compute such an intricate flow over a highly complicated geometry.

4 Summary and conclusion

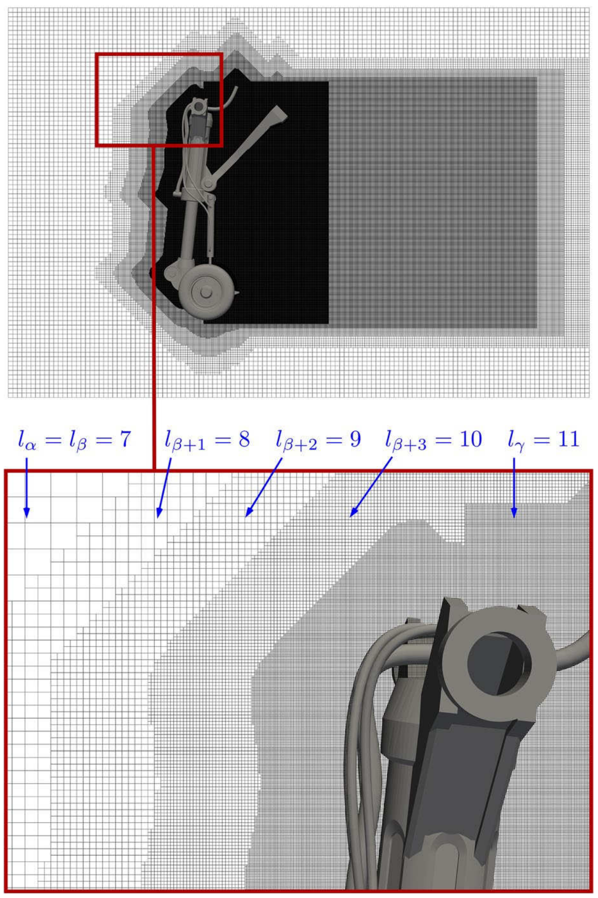
Over the last three decades, the LB method has grown mature and is nowadays used in a variety of physics and engineering applications. Methods for simple flow problems, flow in highly complex geometries, multi-phase, low- and high Mach, and Reynolds number flows have been developed. Furthermore, the LB algorithms are easy to parallelize and show good scaling behavior.

The simulation framework ZFS is developed by the Institute of Aerodynamics and Chair of Fluid Mechanics, RWTH Aachen University, and is supported by

Fig. 9 Geometric setup and computational mesh for the simulation of the turbulent flow around a landing gear configuration



(a) Geometric setup of a front landing gear configuration of a VTOL jet airplane.

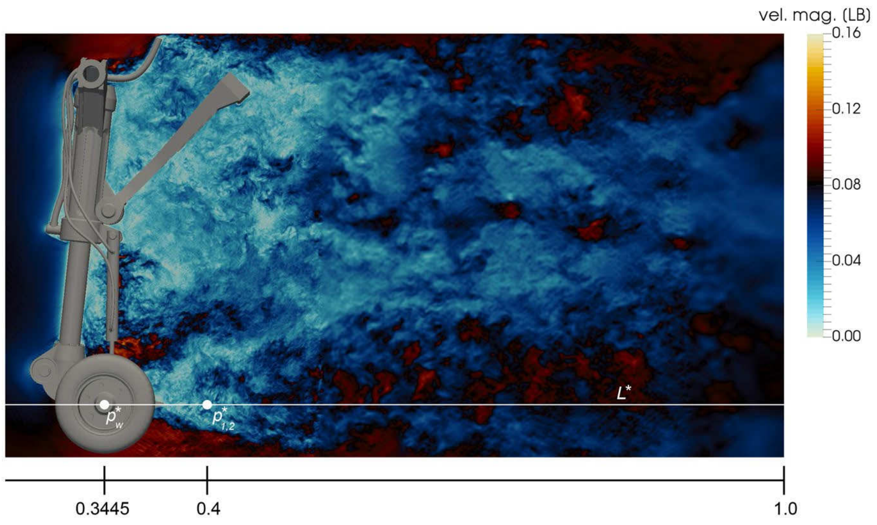


(b) Computational mesh for the simulation of the flow around a landing gear configuration. The mesh consists of levels $l_\alpha = l_\beta = 7, \dots, l_\gamma = 11$.

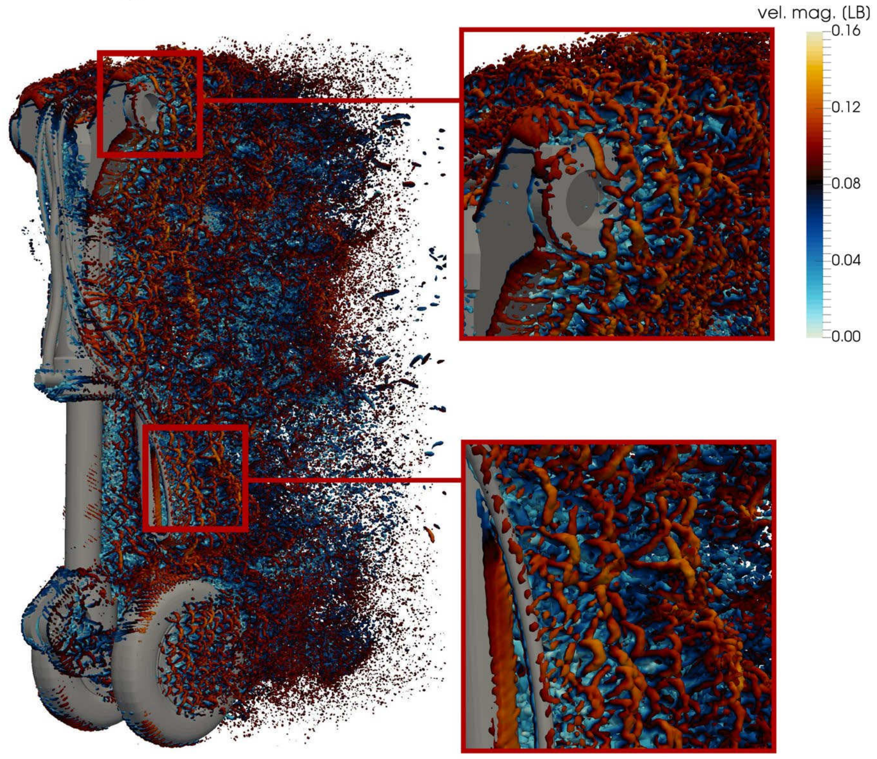
JARA-CSD. It features a massively parallel grid generator, which enables to construct hierarchical octree meshes on hundreds of thousands of compute cores in a small amount of time for the simulation of large-scale problems. It shows good scalability behavior on present HPC systems. Furthermore, parallelized geometries can easily be generated, which enable to massively reduce the memory footprint and the preprocessing times of simulations. For the simulation, different methods have been implemented with LB methods primarily being used for the simulation in complex geometries. The standard SRT, MRT, CLB, and RLB models as well as the thermal MDF approach, the refinement strategy, and the Smagorinsky LES approach for LB methods have been presented. All methods are hybrid MPI/OpenMP parallelized, use parallel I/O for output and feature internal data usage for in situ analyses of the results. The scalability of the SRT method has been shown up to 16,384 nodes on the PowerPC-based JUQUEEN system at JSC and up to 512 nodes on the Intel-based HAZEL HEN machine at HLRS. To show the applicability of these methods, three simulation examples have been presented. The first example considered the flow in the human respiratory tract. Highly resolved simulations were carried out and flow regime transitions such as laminar to transitional flow regions could be identified. The second example showed the applicability to porous media flows, i.e., the flow in GDLs of fuel cells has been investigated. The method showed to be capable of dealing with highly dense fiber structures. The flow in GDLs is dominated by low Reynolds numbers and diffusive effects. The third example is the flow around a front landing gear configuration of a VTOL jet airplane. The resolution of the simulation has been discussed by an analysis of the non-dimensional wall distance. The flow has been analyzed qualitatively by contours of the Δ-criterion and in-plane velocities as well as quantitatively by considering the turbulent kinetic energy derived from the Reynolds stress tensor and power spectral density analyses.

To conclude, with ZFS, an efficient multi-tool framework to implement various LB methods has been established. It allows to simulate laminar to transitional flow in and around complex geometries at low to medium as well

Fig. 10 Flow around a landing gear configuration



(a) Flow in the wake of the landing gear. The cross-section is located in the center between the wheels and is colored by the velocity magnitude. Furthermore, the line L^* and probe locations $p^*_{1,2}$ at which further analyses are performed are shown.



(b) Contours of the Δ-criterion colored by the velocity magnitude. The two insets show enlargements of the flow structures in the vicinity of the geometry.

as higher Reynolds numbers leading to turbulent flow and at low Mach numbers. The modularity of the framework, however, allows for an easy integration of methods such as the Cumulant LB or the interface tracking method for high Reynolds number and compressible, and multi-phase flows. That is, an extension towards aeronautical engineering applications with realistic inflight conditions is straightforward and is line with the research and development strategy of the developers.

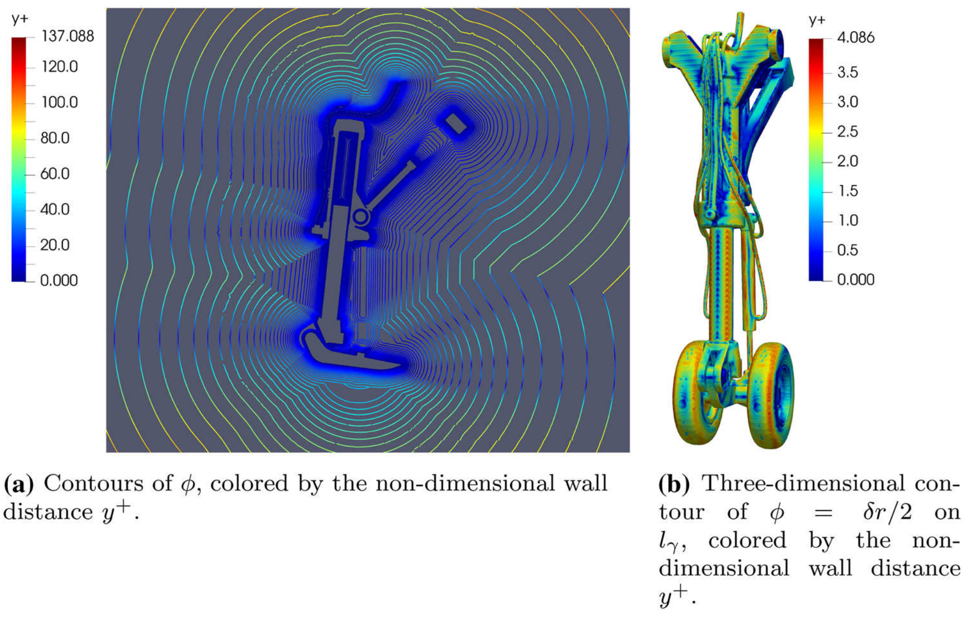

y+
137.088
120.0
100.0
80.0
60.0
40.0
20.0
0.000
y+
4.086
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.000
(a) Contours of ϕ, colored by the non-dimensional wall distance y^+.
(b) Three-dimensional contour of $\phi = \delta r/2$ on l_γ, colored by the non-dimensional wall distance y^+.

Fig. 11 Non-dimensional wall distance y^+ plotted on contours of the signed distance function ϕ

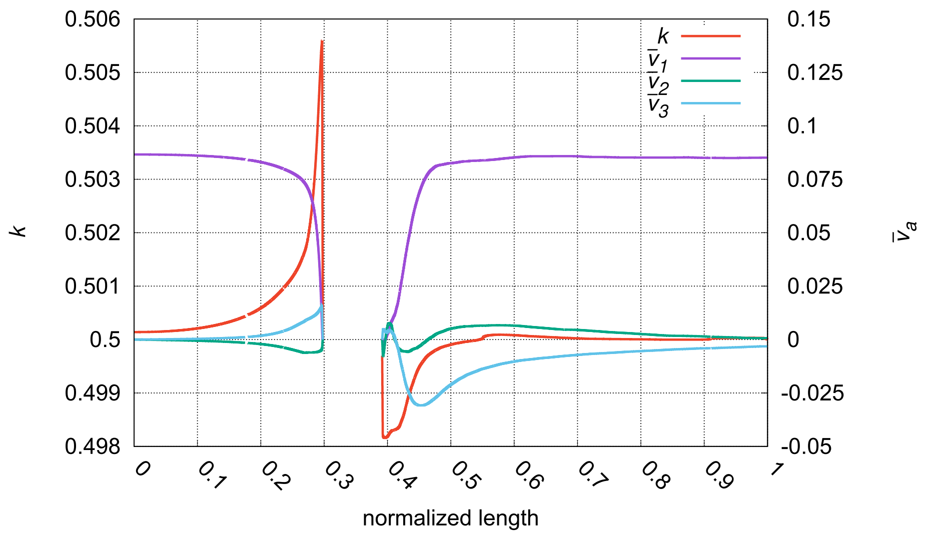


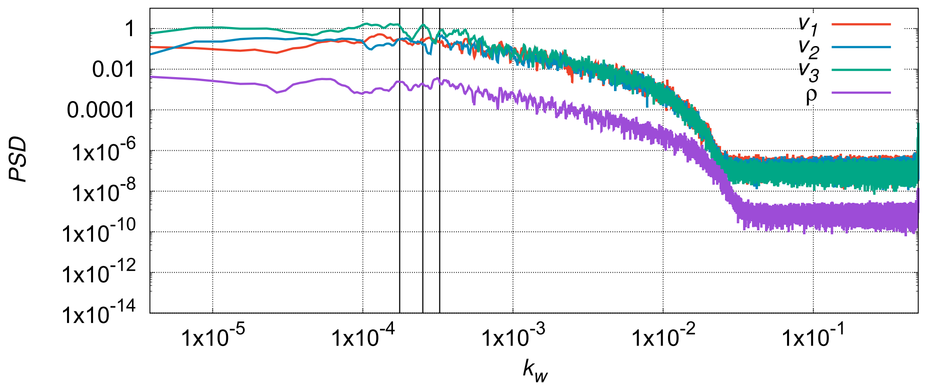
Fig. 12 Turbulent kinetic energy k and the mean velocity components $\bar{v}_a$ along the line L^* shown in Fig. 10a. All quantities are derived from the non-dimensional LB velocities

Note that ZFS is already capable of simulating such problems by means of an FV method solving the compressible Navier–Stokes equations, see, e.g., [64]. Interfaces of moving boundaries can be tracked by a level-set approach, cf. [29]. Particles can be traced with a Lagrangian approach [55]. Aeroacoustics are simulated with a coupled FV discontinuous Galerkin direct-hybrid approach [63] solving the flow problem and the acoustic perturbation equations [16]. That is, ZFS is a tool to simulate a variety of multi-physics problems in engineering and biomedicine. The modularity of the framework allows to couple any of the described methods and for easy model and method extension.

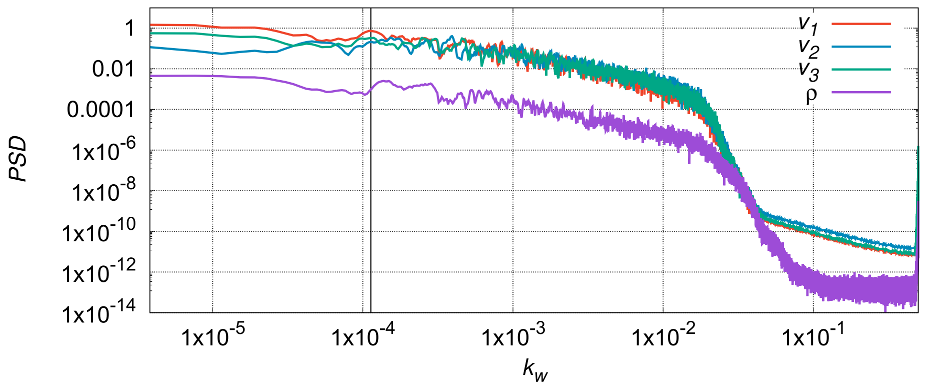
5 Further fields of application and outlook

The LB method is extremely appealing for a variety of fields of application. The cumulant LB method has, e.g., been used to simulate turbulent flows over a resolved urban canopy [46]. König and Fares [38] use the LB method to simulate transonic flows around the NASA Common Research model. The method is based on the trans- and supersonic method presented in [17, 67], i.e., it uses the D3Q39 model with a Hermite expansion to arrive at a fourth-order polynomial for the equilibrium PPDFs. Latt et al. [44] also use the D3Q39 model to

Fig. 13 Power spectral density *PSD* at probe locations $p^*_{1,2}$ (cf. Fig. 10a) over the wave number k_w



(a) Power spectral density PSD at probe location p_1^* (cf. Fig. 10a) over the wave number k_w.



(b) Power spectral density PSD at probe location p_2^* (cf. Fig. 10a) over the wave number k_w.

simulate supersonic flows in a sod shock tube and around a NACA0012 profile. The LB method is also used to solve computational aeroacoustic (CAA) problems, e.g., to study the community noise of urban air transportation vehicles [8] in combination with a Ffowcs–Williams and Hawkings acoustics model [7]. Another approach is to run a direct acoustic simulation, e.g., using a potential energy double-distribution-function (DDF) LB method [48]. Furthermore, the LB method can be used to simulate microfluidic behavior, where especially multi-phase flows, cf. Sect. 1, play an important role. For example, Harting et al. [28] simulate fluid flows in hydrophobic and rough microchannels as well as over surfaces covered by nano- or microscale gas bubbles. In [32], simulations of droplets manipulation generated in lab-on-chip (LOC) microfluidic T-junction are performed. For a good review on LB-microfluid simulations, the reader is also referred to [79]. LB methods are also well suited to simulate rarefied gas flows, where for Knudsen numbers $Kn > 0.1$ continuous approaches such as the FV method fail [27]. For example, rarefied gas flows in microchannels are simulated in [78] using the MRT method with a second-order slip condition. A Bosanquet-type effective viscosity is used to correlate the effective relaxation time with the local Knudsen number to account for varying rarefaction effects. In contrast, a Knudsen-number–relaxation-time relation using the viscosity-based mean free path and mean thermal speed is applied in [35] to simulate 2D and 3D microchannel flows.

Considering the applications presented in this manuscript, ZFS is currently extended by the finite cell method [62] to simulate fluid–structure interaction (FSI) problems in the context of human respiration, e.g., to analyze the nasal valve effect. Furthermore, ZFS will be extended by a multi-phase LB approach such as presented in [45] to track water droplets diffusing through GDLs. To go to realistic Reynolds numbers in non-VTOL jet airplane landing gear configurations, the cumulant LB method is currently being implemented and validated.

Acknowledgements Open Access funding is provided by Projekt DEAL. The authors gratefully acknowledge the Gauss Centre for Supercomputing e.V. (GCS)[3], and the Jülich Aachen Research Alliance Center for Simulation and Data Science (JARA-CSD)[4] Vergabegremium for funding this project by providing computing time on the supercomputers JUQUEEN and JURECA at Jülich Supercomputing Centre (JSC), and the HAZEL HEN systems at the High-Performance Computing Center Stuttgart (HLRS) through the John von Neumann Institute for Computing (NIC), and the JARA-HPC partition. For funding, the authors would like to thank the German Research Association (Deutsche Forschungsgemeinschaft, DFG, grant number: WE-2186/5), the Excellence Initiative of the Federal Government of Germany, and the German States (Jülich Aachen Research Alliance—JARA, ExIni 2005–2017), RWTH Aachen University and Forschungszentrum Jülich GmbH (JARA Center for Simulation and Data Sciences, JARA-CSD), and the Federal Ministry for Economic Affairs and Energy (Zukunft Innovation Mittelstand ZIM, Grant Number: ZF4427201 CS7). Furthermore, the authors would like to thank the Institute of Energy and Climate Research—Techno-economic Systems Analysis (IEK-3)[5] and the Cross-Sectional Team Visualization at JSC for the provision of the GDL geometry and for in-depth discussions.

Appendix A: LB weighting factors

The coefficients t_p, where p determines the number of non-zero components of the directions considered in the $DxQy$ scheme, for the calculation of f_i^{eq} is given by the values in Table 1.

Note that the direction $p = 0$ corresponds to the rest-particle distribution, i.e., the PPDF that represents the particle distribution for particles residing in the local volume of interest, or in other words, in the local computational cell.

Table 1 Direction-dependent weighting coefficients t_p

$DxQy$ scheme	$p = 0$	$p = 1$	$p = 2$	$p = 3$
$D2Q9$	4/9	1/9	1/36	–
$D3Q15$	2/9	1/9	1/72	–
$D3Q19$	1/3	1/18	1/36	–
$D3Q27$	8/27	2/27	1/54	1/216

Appendix B: MRT vectors and matrices

The momentum vectors of the MRT method are given by:

$$\mathbf{m} = \left(\rho, e, \epsilon, j_0, q_0, j_1, q_1, \tau_{00}, \tau_{11}\right)^T \tag{38}$$

and

$$\mathbf{m}^{eq} = \big(1, -2\rho + 3(j_0^2 + j_1^2), \rho - 3(j_0^2 + j_1^2), \\ j_0, -j_0, j_1, -j_1, j_0^2 + j_1^2, j_0 j_1\big)^T, \tag{39}$$

where e is related to the kinetic energy, ϵ is related to the kinetic energy squared, $j_{0,1}$ are the two momentum components, $q_{0,1}$ are proportional to the energy fluxes, and $\tau_{00,11}$ are related to the diagonal and off-diagonal components of the viscous stress tensor $\underline{\boldsymbol{\tau}}$.

The relaxation matrix $\underline{K}_{MRT}$ of the MRT method is given by:

$$\underline{K}_{MRT} = diag(0, s_1, s_2, 0, , s_4, 0, s_4, s_\nu, s_\nu), \tag{40}$$

where s_ν is related to the kinematic viscosity as $\nu = (s_\nu^1 - 1) \cdot c_s^2 \delta t$, s_1 is related to the bulk viscosity, and s_2 and s_4 are adjustable parameters that have no effects on the Navier–Stokes equations.

Note that to keep the description brief, the vectors are only given for the $D2Q9$ model. For further details, the reader is referred to [12].

Appendix C: CLB vector and matrix

Due its complexity, the CLB transformation matrix $\underline{\mathbf{K}}_{CLB}$ and the moment vector $\mathbf{k}$ are only given for the $D2Q9$ model. In this model, the transformation matrix reads:

$$\underline{\mathbf{K}}_{CLB} = (\mathbf{K}_1, \dots \mathbf{K}_9) = \begin{pmatrix} 1 & 0 & 0 & -4 & 0 & 0 & 0 & 0 & 4 \\ 1 & -1 & 1 & 2 & 0 & 1 & -1 & 1 & 1 \\ 1 & -1 & 0 & -1 & 1 & 0 & 0 & -2 & -2 \\ 1 & -1 & -1 & 2 & 0 & -1 & 1 & 1 & 1 \\ 1 & 0 & -1 & -1 & -1 & 0 & -2 & 0 & -2 \\ 1 & 1 & -1 & 2 & 0 & 1 & 1 & -1 & 1 \\ 1 & 1 & 0 & -1 & 1 & 0 & 0 & 2 & -2 \\ 1 & 1 & 1 & 2 & 0 & -1 & -1 & -1 & 1 \\ 1 & 0 & 1 & -1 & -1 & 0 & 2 & 0 & -2 \end{pmatrix} \tag{41}$$

and the moment vector $\mathbf{k} = (k_1, \dots, k_8)^T$:

[3] GCS http://www.gauss-centre.eu.

[4] JARA-CSD https://www.jara.org/de/forschung/center-for-simulation-and-data-sciences

[5] IEK-3 https://www.fz-juelich.de/iek/iek-3/EN/Home/home_node.html

$$\mathbf{k} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \lfloor \omega_3 \left[\rho(v_1^2 + v_2^2) - f_1 - f_3 - f_7 - f_5 - 2(f_8 + f_6 + f_2 + f_4 - \rho/3\right]/12 \rfloor \\ \lfloor \omega_4 \left[f_3 + f_7 - f_1 - f_5 + \rho(v_1^2 + v_2^2)\right]/4 \rfloor \\ \lfloor \omega_5 \left[(f_2 + f_6 - f_4 - f_8) - v_1 v_2 \rho\right]/4 \rfloor \\ \begin{bmatrix} \lfloor \omega_6(-\{[f_8 + f_6 - f_2 - f_4 - 2v_1^2 v_2 \rho + v_2(\rho - f_3 - f_7 - f_9)]/4 & \hookleftarrow \\ +v_1/2(f_2 - f_4 - f_8 + f_6)\}) - v_2/2(-3k_3 - k_4) - 2v_1 k_5 \rfloor & \end{bmatrix} \\ \begin{bmatrix} \lfloor \omega_7(-\{[f_6 + f_4 - f_8 - f_2 - 2v_2^2 v_1 \rho + v_1(\rho - f_5 - f_7 - f_9)]/4 & \hookleftarrow \\ +v_2/2(f_2 - f_6 - f_8 + f_4)\}) - v_1/2(-3k_3 - k_4) - 2v_2 k_5 \rfloor & \end{bmatrix} \\ \begin{bmatrix} \lfloor \omega_8(1/4\{/rho/9 - f_2 - f_4 - f_8 - f_6 + 2[v_1(f_2 - f_4 + f_8 - f_6) & \hookleftarrow \\ +v_2(f_2 + f_4 - f_8 - f_6)] + 4v_1 v_2(f_4 - f_2 + f_8 - f_6) & \hookleftarrow \\ -v_1^2(f_3 + f_2 + f_4 + f_7 + f_8 + f_6) + v_2^2(3v_1^2 \rho - f_1 - f_2 - f_4 & \hookleftarrow \\ -f_8 - f_6 - f_5)\}) - 2k_3 - 2v_1 k_7 - 2v_2 k_6 + 4v_1 v_2 k_5 - (3/2k_3 & \hookleftarrow \\ -k_4/2)(v_1^2 + v_2^2) & \end{bmatrix} \end{pmatrix}. \tag{42}$$

References

1. Aidun, C.K., Clausen, J.R.: Lattice–Boltzmann method for complex flows. Annu. Rev. Fluid Mech. **42**(1), 439–472 (2010). https://doi.org/10.1146/annurev-fluid-121108-145519
2. Baerentzen, J., Aanaes, H.: Signed distance computation using the angle weighted pseudonormal. IEEE Trans. Vis. Comput. Graph. **11**(3), 243–253 (2005). https://doi.org/10.1109/TVCG.2005.49
3. Benzi, R., Succi, S., Vergassola, M.: The lattice Boltzmann equation: theory and applications. Phys. Rep. **222**(3), 145–197 (1992). https://doi.org/10.1016/0370-1573(92)90090-M
4. Bernaschi, M., Fatica, M., Melchionna, S., Succi, S., Kaxiras, E.: A flexible high-performance Lattice Boltzmann GPU code for the simulations of fluid flows in complex geometries. Concurr. Comput. Pract. Exp. **22**(1), 1–14 (2010). https://doi.org/10.1002/cpe.1466
5. Boghosian, B.M., Yepez, J., Coveney, P.V., Wager, A.: Entropic lattice Boltzmann methods. Proc. R. Soc. A Math. Phys. Eng. Sci. **457**(2007), 717–766 (2001). https://doi.org/10.1098/rspa.2000.0689
6. Bouzidi, M., Firdaouss, M., Lallemand, P.: Momentum transfer of a Boltzmann–lattice fluid with boundaries. Phys. Fluids **13**(11), 3452–3459 (2001). https://doi.org/10.1063/1.1399290
7. Bres, G.a., Perot, F., Freed, D.: A Ffowcs Williams Hawkings Solver for Lattice-Boltzmann based computational aeroacoustics. In: 16th AIAA/CEAS Aeroacoustics Conference p. 15 (2010)
8. Casalino, D., van der Velden, W.C., Romani, G.: Community noise of urban air transportation vehicles. In: AIAA Scitech 2019 Forum. American Institute of Aeronautics and Astronautics, Reston, Virginia (2019). https://doi.org/10.2514/6.2019-1834
9. Chen, H., Gopalakrishnan, P., Zhang, R.: Recovery of Galilean invariance in thermal lattice Boltzmann models for arbitrary Prandtl number. Int. J. Mod. Phys. C **25**(10), 1450046 (2014). https://doi.org/10.1142/S0129183114500466
10. Chong, M.S., Perry, A.E., Cantwell, B.J.: A general classification of three-dimensional flow fields. Phys. Fluids A **2**(5), 765–777 (1990). https://doi.org/10.1063/1.857730
11. Chopard, B., Droz, M.: Cellular Automata Modeling of Physical Systems. Cambridge University Press, Cambridge (1998). https://doi.org/10.1017/CBO9780511549755
12. D'Humières, D., Ginzburg, I., Krafczyk, M., Lallemand, P., Luo, L.S.: Multiple-relaxation-time lattice Boltzmann models in three dimensions. Philos. Trans. Ser. A Math. Phys. Eng. Sci. **360**(1792), 437–51 (2002). https://doi.org/10.1098/rsta.2001.0955
13. Di Ilio, G., Chiappini, D., Ubertini, S., Bella, G., Succi, S.: Fluid flow around NACA 0012 airfoil at low-Reynolds numbers with hybrid lattice Boltzmann method. Comput. Fluids **166**, 200–208 (2018). https://doi.org/10.1016/j.compfluid.2018.02.014
14. Dupuis, A., Chopard, B.: Theory and applications of an alternative lattice Boltzmann grid refinement algorithm. Phys. Rev. E **67**(6), 1–7 (2003). https://doi.org/10.1103/PhysRevE.67.066707
15. Eitel-Amor, G., Meinke, M., Schröder, W.: A lattice-Boltzmann method with hierarchically refined meshes. Comput. Fluids **75**, 127–139 (2013). https://doi.org/10.1016/j.compfluid.2013.01.013
16. Ewert, R., Schröder, W.: Acoustic perturbation equations based on flow decomposition via source filtering. J. Comput. Phys. **188**(2), 365–398 (2003). https://doi.org/10.1016/S0021-9991(03)00168-2
17. Fares, E., Wessels, M., Zhang, R., Sun, C., Gopalaswamy, N., Roberts, P., Hoch, J., Chen, H.: Validation of a Lattice-Boltzmann approach for transonic and supersonic flow simulations. In: 52nd Aerospace Sciences Meeting, January, pp. 1–17. American Institute of Aeronautics and Astronautics, Reston, Virginia (2014). https://doi.org/10.2514/6.2014-0952
18. Folk, M., Pourmal, E.: Balancing performance and preservation lessons learned with HDF5. In: Proceedings of the 2010 Roadmap for digital preservation interoperability framework workshop on - US-DPIF '10, pp. 1–8 (2010). https://doi.org/10.1145/2039274.2039285
19. Freitas, R.K., Henze, A., Meinke, M., Schröder, W.: Analysis of Lattice–Boltzmann methods for internal flows. Comput. Fluids **47**(1), 115–121 (2011). https://doi.org/10.1016/j.compfluid.2011.02.019
20. Frisch, U., Hasslacher, B., Pomeau, Y.: Lattice-gas automata for the Navier–Stokes equation. Phys. Rev. Lett. **56**(14), 1505–1508 (1986). https://doi.org/10.1103/PhysRevLett.56.1505
21. Froning, D., Yu, J., Gaiselmann, G., Reimer, U., Manke, I., Schmidt, V., Lehnert, W.: Impact of compression on gas transport in non-woven gas diffusion layers of high temperature polymer electrolyte fuel cells. J. Power Sources **318**, 26–34 (2016). https://doi.org/10.1016/j.jpowsour.2016.03.102
22. Geier, M., Greiner, A., Korvink, J.G.: Cascaded digital lattice Boltzmann automata for high Reynolds number flow. Phys. Rev. E **73**(6), 066705 (2006). https://doi.org/10.1103/PhysRevE.73.066705
23. Geier, M., Schönherr, M., Pasquali, A., Krafczyk, M.: The cumulant lattice Boltzmann equation in three dimensions: theory and validation. Comput. Math. Appl. **70**(4), 507–547 (2015). https://doi.org/10.1016/j.camwa.2015.05.001
24. Ginzburg, I., D'Humières, D.: Multireflection boundary conditions for lattice Boltzmann models. Phys. Rev. E **68**(6), 066614 (2003). https://doi.org/10.1103/PhysRevE.68.066614
25. Gunstensen, A.K., Rothman, D.H., Zaleski, S., Zanetti, G.: Lattice Boltzmann model of immiscible fluids. Phys. Rev. A **43**(8), 4320–4327 (1991). https://doi.org/10.1103/PhysRevA.43.4320
26. Guo, Z., Shi, B., Zheng, C.: A coupled lattice BGK model for the Boussinesq equations. Int. J. Numer. Methods Fluids **39**(4), 325–342 (2002). https://doi.org/10.1002/fld.337
27. Hänel, D.: Molekulare Gasdynamik, Einführung in die kinetische Theorie der Gase und Lattice-Boltzmann-Methoden. Springer, Berlin (2004)
28. Harting, J., Kunert, C., Hyväluoma, J.: Lattice Boltzmann simulations in microfluidics: probing the no-slip boundary condition in hydrophobic, rough, and surface nanobubble laden microchannels. Microfluid. Nanofluid. **8**(1), 1 (2010). https://doi.org/10.1007/s10404-009-0506-6

29. Hartmann, D., Meinke, M., Schröder, W.: Differential equation based constrained reinitialization for level set methods. J. Comput. Phys. **227**(14), 6821–6845 (2008). https://doi.org/10.1016/j.jcp.2008.03.040
30. He, X., Chen, S., Zhang, R.: A Lattice Boltzmann scheme for incompressible multiphase flow and its application in simulation of Rayleigh-Taylor instability. J. Comput. Phys. **152**(2), 642–663 (1999). https://doi.org/10.1006/jcph.1999.6257
31. Higuera, F.J., Succi, S., Benzi, R.: Lattice gas dynamics with enhanced collisions. Europhys. Lett. (EPL) **9**(4), 345–349 (1989). https://doi.org/10.1209/0295-5075/9/4/008
32. Hoseinpour, B., Sarreshtehdari, A.: Lattice Boltzmann simulation of droplets manipulation generated in lab-on-chip (LOC) microfluidic T-junction. J. Mol. Liq. **297**, 111736 (2020). https://doi.org/10.1016/j.molliq.2019.111736
33. Hou, S., Sterling, J., Chen, S., Doolen, G.D.: A Lattice Boltzmann subgrid model for high reynolds number flows. Pattern Format Lattice Gas Autom **6**, 1–18 (1994)
34. Hunt, J., Wray, A., Moin, P.: Eddies, Stream, and Convergence Zones in Turbulent Flows. In: Turbulence Research, Report CTR, pp. 193–208 (1988)
35. Jeong, N.: Lattice Boltzmann approach for the simulation of rarefied gas flow in the slip flow regime. J. Mech. Sci. Technol. **27**(6), 1753–1761 (2013). https://doi.org/10.1007/s12206-013-0426-y
36. Karlin, I.V., Ferrante, A., Öttinger, H.C.: Perfect entropy functions of the Lattice Boltzmann method. Europhys. Lett. (EPL) **47**(2), 182–188 (1999). https://doi.org/10.1209/epl/i1999-00370-1
37. Kataoka, T., Tsutahara, M.: Lattice Boltzmann model for the compressible Navier–Stokes equations with flexible specific-heat ratio. Phys. Rev. E **69**(3), 035701 (2004). https://doi.org/10.1103/PhysRevE.69.035701
38. Konig, B., Fares, E.: Validation of a Transonic Lattice-Boltzmann Method on the NASA Common Research Model. In: 54th AIAA Aerospace Sciences Meeting, January, pp. 1–14. American Institute of Aeronautics and Astronautics, Reston, Virginia (2016). https://doi.org/10.2514/6.2016-2023
39. Krämer, A., Küllmer, K., Reith, D., Joppich, W., Foysi, H.: Semi-Lagrangian off-lattice Boltzmann method for weakly compressible flows. Phys. Rev. E **95**(2), 023305 (2017). https://doi.org/10.1103/PhysRevE.95.023305
40. Krause, D., Thörnig, P.: JURECA: modular supercomputer at Jülich Supercomputing Centre. J. Large-scale Res. Facilit. JLSRF **4**, A132 (2018). https://doi.org/10.17815/jlsrf-4-121-1
41. Kvesić, M., Reimer, U., Froning, D., Lüke, L., Lehnert, W., Stolten, D.: 3D modeling of a 200 cm^2 HT-PEFC short stack. Int. J. Hydrogen Energy **37**(3), 2430–2439 (2012). https://doi.org/10.1016/j.ijhydene.2011.10.055
42. Lallemand, P., Luo, L.S.: Theory of the lattice Boltzmann method: dispersion, dissipation, isotropy, Galilean invariance, and stability. Phys. Rev. E **61**(6), 6546–6562 (2000). https://doi.org/10.1103/PhysRevE.61.6546
43. Latt, J., Chopard, B.: Lattice Boltzmann method with regularized pre-collision distribution functions. Math. Comput. Simul. **72**(2–6), 165–168 (2006). https://doi.org/10.1016/j.matcom.2006.05.017
44. Latt, J., Coreixas, C., Beny, J., Parmigiani, A.: Efficient supersonic flows through high-order guided equilibrium with lattice Boltzmann. Philos. Trans. R. Soc. A Math. Phys. Eng. Sci. (2019)
45. Lee, T., Liu, L.: Lattice Boltzmann simulations of micron-scale drop impact on dry surfaces. J. Comput. Phys. **229**(20), 8045–8063 (2010). https://doi.org/10.1016/j.jcp.2010.07.007
46. Lenz, S., Schönherr, M., Geier, M., Krafczyk, M., Pasquali, A., Christen, A., Giometto, M.: Towards real-time simulation of turbulent air flow over a resolved urban canopy using the cumulant lattice Boltzmann method on a GPGPU. J. Wind Eng. Ind. Aerodyn. **189**, 151–162 (2019). https://doi.org/10.1016/j.jweia.2019.03.012
47. Li, J., Zingale, M., Liao, W.k., Choudhary, A., Ross, R., Thakur, R., Gropp, W., Latham, R., Siegel, A., Gallagher, B.: Parallel netCDF: A High-Performance Scientific I/O Interface. In: Proceedings of the 2003 ACM/IEEE conference on Supercomputing - SC '03, p. 39. ACM Press, New York (2003). https://doi.org/10.1145/1048935.1050189
48. Li, K., Zhong, C.: Aeroacoustic simulations using compressible lattice Boltzmann method. Adv. Appl. Math. Mech. **8**(5), 795–809 (2016). https://doi.org/10.4208/aamm.2015.m1083
49. Lintermann, A.: Efficient parallel geometry distribution for the simulation of complex flows. In: Proceedings of the VII European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress 2016), pp. 1277–1293. Athens (2016). https://doi.org/10.7712/100016.1885.5067
50. Lintermann, A., Meinke, M., Schröder, W.: Investigations of the inspiration and heating capability of the human nasal cavity based on a Lattice-Boltzmann method. In: Proceedings of the ECCOMAS Thematic International Conference on Simulation and Modeling of Biological Flows (SIMBIO 2011). Brussels, Belgium (2011)
51. Lintermann, A., Meinke, M., Schröder, W.: Fluid mechanics based classification of the respiratory efficiency of several nasal cavities. Comput. Biol. Med. **43**(11), 1833–1852 (2013). https://doi.org/10.1016/j.compbiomed.2013.09.003
52. Lintermann, A., Meinke, M., Schröder, W.: Zonal Flow Solver (ZFS): a highly efficient multi-physics simulation framework. Int. J. Comput. Fluid Dyn. (2020). https://doi.org/10.1080/10618562.2020.1742328
53. Lintermann, A., Schlimpert, S., Grimmen, J., Günther, C., Meinke, M., Schröder, W.: Massively parallel grid generation on HPC systems. Comput. Methods Appl. Mech. Eng. **277**, 131–153 (2014). https://doi.org/10.1016/j.cma.2014.04.009
54. Lintermann, A., Schröder, W.: A hierarchical numerical journey through the nasal cavity: from nose-like models to real anatomies. Flow Turbulence Combust. (2017). https://doi.org/10.1007/s10494-017-9876-0
55. Lintermann, A., Schröder, W.: Simulation of aerosol particle deposition in the upper human tracheobronchial tract. Eur. J. Mech. B. Fluids **63**, 73–89 (2017). https://doi.org/10.1016/j.euromechflu.2017.01.008
56. Luo, L.S.: Unified theory of lattice boltzmann models for nonideal gases. Phys. Rev. Lett. **81**(8), 1618–1621 (1998). https://doi.org/10.1103/PhysRevLett.81.1618
57. McNamara, G.R., Zanetti, G.: Use of the Boltzmann equation to simulate lattice-gas automata. Phys. Rev. Lett. **61**(20), 2332–2335 (1988). https://doi.org/10.1103/PhysRevLett.61.2332
58. Nannelli, F., Succi, S.: The lattice Boltzmann equation on irregular lattices. J. Stat. Phys. **68**(3–4), 401–407 (1992). https://doi.org/10.1007/BF01341755
59. Qian, Y.H., D'Humières, D., Lallemand, P.: Lattice BGK models for Navier–Stokes equation. Europhys. Lett. (EPL) **17**(6), 479–484 (1992). https://doi.org/10.1209/0295-5075/17/6/001
60. Rothman, D.H., Keller, J.M.: Immiscible cellular-automaton fluids. J. Stat. Phys. **52**(3–4), 1119–1127 (1988). https://doi.org/10.1007/BF01019743
61. Sagan, H.: Space-Filling Curves, 1 edn. Universitext. Springer, New York (1994). https://doi.org/10.1007/978-1-4612-0871-6
62. Schillinger, D., Ruess, M.: The finite cell method: a review in the context of higher-order structural analysis of CAD and image-based geometric models. Arch. Comput. Methods Eng. **22**(3), 391–455 (2015). https://doi.org/10.1007/s11831-014-9115-y
63. Schlottke-Lakemper, M., Yu, H., Berger, S., Meinke, M., Schröder, W.: A fully coupled hybrid computational aeroacoustics

method on hierarchical Cartesian meshes. Comput. Fluids **144**, 137–153 (2017). https://doi.org/10.1016/j.compfluid.2016.12.001
64. Schneiders, L., Günther, C., Meinke, M., Schröder, W.: An efficient conservative cut-cell method for rigid bodies interacting with viscous compressible flows. J. Comput. Phys. **311**, 62–86 (2016). https://doi.org/10.1016/j.jcp.2016.01.026
65. Shan, X., Chen, H.: Lattice Boltzmann model for simulating flows with multiple phases and components. Phys. Rev. E **47**(3), 1815–1819 (1993)
66. Shan, X., He, X.: Discretization of the velocity space in the solution of the Boltzmann equation. Phys. Rev. Lett. **80**(1), 65–68 (1998). https://doi.org/10.1103/PhysRevLett.80.65
67. Shan, X., Yuan, X., Hudong, C.: Kinetic theory representation of hydrodynamics: a way beyond the Navier-Stokes equation. J. Fluid Mech. **550**, 413 (2006). https://doi.org/10.1017/S0022112005008153
68. Smagorinsky, J.: General circulation experiments with the primitive equations. Mon. Weather Rev. **91**(3), 99–164 (1963). https://doi.org/10.1175/1520-0493(1963)091<0099:GCEWTP>2.3.CO;2
69. Stephan, M., Docter, J.: JUQUEEN: IBM blue gene/Q® supercomputer system at the Jülich supercomputing centre. J. Large-scale Res. Facilit JLSRF **1**, A1 (2015). https://doi.org/10.17815/jlsrf-1-18
70. Succi, S., Amati, G., Bernaschi, M., Falcucci, G., Lauricella, M., Montessori, A.: Towards exascale Lattice Boltzmann computing. Comput. Fluids **181**, 107–115 (2019). https://doi.org/10.1016/j.compfluid.2019.01.00
71. Sun, C.: Adaptive lattice Boltzmann model for compressible flows: viscous and conductive properties. Phys. Rev. E **61**(3), 2645–2653 (2000). https://doi.org/10.1103/PhysRevE.61.2645
72. Swift, M.R., Osborn, W.R., Yeomans, J.M.: Lattice Boltzmann simulation of nonideal fluids. Phys. Rev. Lett. **75**(5), 830–833 (1995). https://doi.org/10.1103/PhysRevLett.75.830
73. Wittmann, M., Zeiser, T., Hager, G., Wellein, G.: Modeling and analyzing performance for highly optimized propagation steps of the lattice Boltzmann method on sparse lattices pp. 1–9 (2014)
74. Wolf-Gladrow, D.A.: Lattice-Gas Cellular Automata and Lattice Boltzmann Models—An Introduction. Springer, Berlin (2000)
75. Wolfram, S.: Statistical mechanics of cellular automata. Rev. Mod. Phys. **55**(3), 601–644 (1983). https://doi.org/10.1103/RevModPhys.55.601
76. Yu, D., Mei, R., Luo, L.S., Shyy, W.: Viscous flow computations with the method of lattice Boltzmann equation. Prog. Aerosp. Sci. **39**(5), 329–367 (2003). https://doi.org/10.1016/S0376-0421(03)00003-4
77. Yu, W., Vetter, J., Canon, R.S., Jiang, S.: Exploiting Lustre File Joining for Effective Collective IO. In: Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid '07), pp. 267–274. IEEE (2007). https://doi.org/10.1109/CCGRID.2007.51
78. Yuan, Y., Rahman, S.: Extended application of lattice Boltzmann method to rarefied gas flow in micro-channels. Phys. A **463**, 25–36 (2016). https://doi.org/10.1016/j.physa.2016.06.123
79. Zhang, J.: Lattice Boltzmann method for microfluidics: models and applications. Microfluid. Nanofluid. **10**(1), 1–28 (2011). https://doi.org/10.1007/s10404-010-0624-1